MITSUBISHI ELECTRIC
Changes for the Better

for a greener tomorrow

FATEC

Mitsubishi Programmable Controllers Training Manual MELSEC iQ-R Series Basic Course (for GX Works3)

SAFETY PRECAUTION

(Always read these instructions before using the products.)

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.
- Before opening the safety cover, turn off the power or ensure the safety.
- Do not touch the movable portion.

⚠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 mounting or removing the module.
  Failure to do so may result in malfunctions of the module or electric shock.
- When the demonstration machine (such as X/Y table) emits abnormal odor/sound, press the "Power switch" or "Emergency switch" to turn off.
- When a problem occurs, notify the instructor as soon as possible.

MEMO

REVISIONS

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

Revision date	*Manual number	Description
February 2016	SH(NA)-081898ENG-A	First edition

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

© 2016 MITSUBISHI ELECTRIC CORPORATION

MEMO

CONTENTS

SAFETY PRECAUTION . . . . . A - 1
REVISIONS . . . . . A - 3
INTRODUCTION . . . . . A - 9
RELEVANT MANUALS . . . . . A - 9

CHAPTER 1 BASICS OF A PROGRAMMABLE CONTROLLER 1 - 1

1.1 Programming Languages . . . . . 1 - 1
1.2 Program . . . . . 1 - 2
1.3 System Configuration . . . . . 1 - 6
1.3.1 Overall configuration . . . . . 1 - 6
1.4 Memory Configuration of the CPU Module . . . . . 1 - 11
1.5 External I/O Signals and I/O Numbers . . . . . 1 - 13
1.6 System Configuration and I/O Numbers of the Demonstration Machine . . . . . 1 - 17

CHAPTER 2 OPERATING GX Works3 2 - 1

2.1 Main Functions of GX Works3 . . . . . 2 - 1
2.2 Operations Before Creating a Ladder Program . . . . . 2 - 4
2.2.1 Starting GX Works3 . . . . . 2 - 4
2.2.2 Creating a new project . . . . . 2 - 5
2.3 Preparations for Stating the CPU Module . . . . . 2 - 8
2.3.1 Installing a battery . . . . . 2 - 9
2.3.2 Inserting or removing an extended SRAM cassette . . . . . 2 - 9
2.3.3 Inserting and removing an SD memory card . . . . . 2 - 11
2.3.4 Specifying connection destination . . . . . 2 - 12
2.3.5 Initializing the CPU module . . . . . 2 - 15
2.3.6 Clearing the error history of CPU module . . . . . 2 - 18
2.3.7 Setting the clock of the CPU module . . . . . 2 - 19
2.4 Creating a Ladder Program . . . . . 2 - 20
2.4.1 Creating a ladder program by entering devices and labels . . . . . 2 - 21
2.4.2 Creating a ladder program with function keys . . . . . 2 - 24
2.4.3 Creating a ladder program with tool buttons . . . . . 2 - 27
2.5 Converting a Created Ladder Program . . . . . 2 - 30
2.6 Reading/Writing Data from/to the Programmable Controller CPU . . . . . 2 - 31
2.6.1 Writing data to the CPU module . . . . . 2 - 33
2.6.2 Reading data from the CPU module . . . . . 2 - 35
2.7 Monitoring the Ladder . . . . . 2 - 37
2.8 Diagnosing the Programmable Controller CPU . . . . . 2 - 40
2.9 Editing a Ladder Program . . . . . 2 - 42
2.9.1 Modifying a part of a ladder program . . . . . 2 - 42
2.9.2 Drawing a line . . . . . 2 - 44
2.9.3 Deleting a line . . . . . 2 - 46
2.9.4 Inserting a row . . . . . 2 - 47
2.9.5 Deleting a row . . . . . 2 - 49
2.9.6 Cutting or copying a ladder . . . . . 2 - 50
2.10 Verifying Data . . . . . 2 - 52
2.11 Saving a Created Ladder Program . . . . . 2 - 54
2.11.1 Saving a program in the single file format . . . . . 2 - 54
2.11.2 Saving a program in the workspace format . . . . . 2 - 55

2.12 Opening a Saved Project . . . . . 2 - 57
2.13 Opening a Project in Another Format . . . . . 2 - 58

CHAPTER 3 DEVICES AND PARAMETERS OF A PROGRAMMABLE CONTROLLER 3 - 1

3.1 Devices . . . . . 3 - 1
3.2 Parameters . . . . . 3 - 3

CHAPTER 4 SEQUENCE INSTRUCTIONS AND BASIC INSTRUCTIONS -PART 1- 4 - 1

4.1 Instructions Described in This Chapter . . . . . 4 - 1
4.1.1 Instructions not described in this chapter -Part 1- . . . . . 4 - 2
4.1.2 Instructions not described in this chapter -Part 2- . . . . . 4 - 3
4.2 Differences Between [OUT] and [SET]/[RST] . . . . . 4 - 4
4.2.1 [OUT] (Coil output) . . . . . 4 - 4
4.2.2 [SET]/[RST](Setting/resetting devices) . . . . . 4 - 5
4.3 Measuring Timers (Timer, High-speed Timer, Retentive Timer) . . . . . 4 - 6
4.4 Counting with a Counter . . . . . 4 - 9
4.5 [PLS] (Turning on a Specified Device for One Scan at the Rising Edge of an Input Condition)
[PLF] (Turning on a Specified Device for One Scan at the Falling Edge of an Input Condition) . . . . . 4 - 19
4.6 [CJ] (Conditional Jump of the Non-Delay Execution Type)
[SCJ] (Conditional Jump Executed After One Scan) . . . . . 4 - 25
4.7 Exercise . . . . . 4 - 30
4.7.1 Exercise 1 . . . . . 4 - 30
4.7.2 Exercise 2 . . . . . 4 - 31
4.7.3 Exercise 3 . . . . . 4 - 33

CHAPTER 5 BASIC INSTRUCTIONS -PART 2- 5 - 1

5.1 Notation of Values (Data) . . . . . 5 - 1
5.2 Transfer Instructions . . . . . 5 - 10
5.2.1 [MOV(P)] (Transferring 16-bit data) . . . . . 5 - 10
5.2.2 [FMOV(P)] (Transferring the same data in a batch)
[BMOV(P)] (Transferring block data in a batch) . . . . . 5 - 18
5.3 Comparison Operation Instructions . . . . . 5 - 23
5.4 Arithmetic Operation Instructions . . . . . 5 - 27
5.4.1 [+(P)] (Addition of 16-bit binary data)
[-(P)] (Subtraction of 16-bit binary data) . . . . . 5 - 27
5.4.2 [*(P)] (Multiplication of 16-bit binary data)
[/(P)] (Division of 16-bit binary data) . . . . . 5 - 30
5.4.3 32-bit data instructions and their necessities . . . . . 5 - 35
5.5 External Setting of Timer/Counter Values and External Display of Current Values . . . . . 5 - 38
5.6 Exercise . . . . . 5 - 40
5.6.1 [Exercise 1] MOV-1 . . . . . 5 - 40
5.6.2 [Exercise 2] MOV-2 . . . . . 5 - 41
5.6.3 [Exercise 3] Comparison instruction . . . . . 5 - 42
5.6.4 [Exercise 4] +, - . . . . . 5 - 43
5.6.5 [Exercise 5] *, / . . . . . 5 - 44
5.6.6 [Exercise 6] D*, D/ . . . . . 5 - 45

CHAPTER 6 HOW TO USE OTHER FUNCTIONS 6 - 1

6.1 Online Test Function . . . . . 6 - 1

6.1.1 Forced on/off of the device (Y) . . . . . 6 - 2
6.1.2 Setting/resetting of the device (M) . . . . . 6 - 2
6.1.3 Current value change of the device (T) . . . . . 6 - 3
6.1.4 Reading error steps . . . . . 6 - 4
6.1.5 Remote RUN/STOP . . . . . 6 - 5
6.2 Creating the Module Configuration . . . . . 6 - 7
6.3 Device Batch Replacement . . . . . 6 - 10
6.3.1 Replacing device numbers in a batch . . . . . 6 - 10
6.3.2 Changing normally open contacts ↔ normally closed contacts of specified devices in a batch . . . . . 6 - 12
6.4 Online Change . . . . . 6 - 14
6.5 Watch Window . . . . . 6 - 15
6.6 How to Create Comments . . . . . 6 - 16

CHAPTER 7 NEW FUNCTIONS OF MELSEC iQ-R/GX Works3 7 - 1

7.1 Features of MELSEC iQ-R . . . . . 7 - 1
7.2 Differences Between the MELSEC-Q Series and the MELSEC iQ-R Series . . . . . 7 - 4
7.3 Functions of GX Works3 . . . . . 7 - 5

APPENDICES App. - 1

Appendix 1 I/O Control Mode . . . . . App. - 1
Appendix 1.1 Direct mode . . . . . App. - 1
Appendix 1.2 Refresh mode . . . . . App. - 2
Appendix 1.3 Comparisons between direct mode and refresh mode . . . . . App. - 3
Appendix 2 List of Special Relay Areas . . . . . App. - 4
Appendix 3 List of Special Register Areas . . . . . App. - 5
Appendix 4 Program Examples . . . . . App. - 6
Appendix 4.1 Flip-flop ladder . . . . . App. - 6
Appendix 4.2 One-shot ladder . . . . . App. - 8
Appendix 4.3 Long-time timer . . . . . App. - 9
Appendix 4.4 Off delay timer . . . . . App. - 10
Appendix 4.5 On delay timer (momentary input) . . . . . App. - 11
Appendix 4.6 On/off repeat ladder . . . . . App. - 11
Appendix 4.7 Preventing chattering inputs . . . . . App. - 11
Appendix 4.8 Ladder with common lines . . . . . App. - 12
Appendix 4.9 Time control program . . . . . App. - 13
Appendix 4.10 Clock ladder . . . . . App. - 14
Appendix 4.11 Star-delta starting of an electric motor . . . . . App. - 16
Appendix 4.12 Displaying the elapsed time and outputting before time limit . . . . . App. - 17
Appendix 4.13 Retentive timer . . . . . App. - 18
Appendix 4.14 Switching timer setting values with external switches . . . . . App. - 19
Appendix 4.15 Setting a counter with external switches . . . . . App. - 20
Appendix 4.16 Measuring the operating time . . . . . App. - 22
Appendix 4.17 Measuring the cycle time . . . . . App. - 22
Appendix 4.18 Application example of (D)CML(P) . . . . . App. - 23
Appendix 4.19 Dolly line control . . . . . App. - 24
Appendix 4.20 Compressor sequential operation with ring counters . . . . . App. - 26
Appendix 4.21 Application example to a positioning control . . . . . App. - 30
Appendix 4.22 Application example using the index register (Z) . . . . . App. - 31
Appendix 4.23 Application example of FIFO instructions . . . . . App. - 33
Appendix 4.24 Application example of data shifting . . . . . App. - 36

Appendix 4.25 Program example: Square root operations . . . . . App. - 39
Appendix 4.26 Program example: Multiplication with the nth power . . . . . App. - 40
Appendix 4.27 Displaying the number of failures and failure number in a failure detection program . . . . . App. - 41
Appendix 5 Memory and Files to be Handled by the CPU Module . . . . . App. - 45
Appendix 6 Checking and Setting Shortcut Keys. . . . . App. - 47
Appendix 7 Index Modification . . . . . App. - 48
Appendix 8 FB (Function Block). . . . . App. - 51
Appendix 8.1 FB . . . . . App. - 51
Appendix 8.1.1 FB conversion . . . . . App. - 52
Appendix 8.1.2 Advantages of using FBs. . . . . App. - 53
Appendix 8.1.3 FB library . . . . . App. - 56
Appendix 8.1.4 Precautions for using FBs . . . . . App. - 58
Appendix 8.2 Creating a program using FBs. . . . . App. - 59

INTRODUCTION

This textbook describes a programmable controller, the methods of editing a program with GX Works3, sequence instructions and standard functions/function blocks for helping users to understand the programming for the MELSEC iQ-R series.

RELEVANT MANUALS

Manual name [manual number]	Description	Available form
MELSEC iQ-R CPU Module User's Manual (Startup) [SH-081263ENG]	Performance specifications, procedures before operation, and troubleshooting of the CPU module	e-Manual EPUB PDF
MELSEC iQ-R CPU Module User's Manual (Application) [SH-081264ENG]	Memory, functions, devices, and parameters of the CPU module	e-Manual EPUB PDF
MELSEC iQ-R Programming Manual (Program Design) [SH-081265ENG]	Program specifications such as of ladder programs and ST programs, and labels	e-Manual EPUB PDF
MELSEC iQ-R Programming Manual (Instructions, Standard Functions/Function Blocks) [SH-081266ENG]	Instructions for the CPU module, instructions dedicated for intelligent function modules, and standard functions/function blocks	e-Manual EPUB PDF
MELSEC iQ-R Module Configuration Manual [SH-081262ENG]	System configuration, specifications, mounting, wiring, and maintenance and inspection required for using the MELSEC iQ-R series programmable controller	e-Manual EPUB PDF
GX Works3 Operating Manual [SH-081215ENG]	System configuration of GX Works3, parameter setting, and operation method of the online function	e-Manual EPUB PDF

Point

e-Manual refers to the Mitsubishi FA electronic book manuals that can be browsed using a dedicated tool.
e-Manual has the following features:

- Required information can be cross-searched in multiple manuals.
- Other manuals can be accessed from the links in the manual.
- The hardware specifications of each part can be found from the product figures.
- Pages that users often browse can be bookmarked.

MEMO

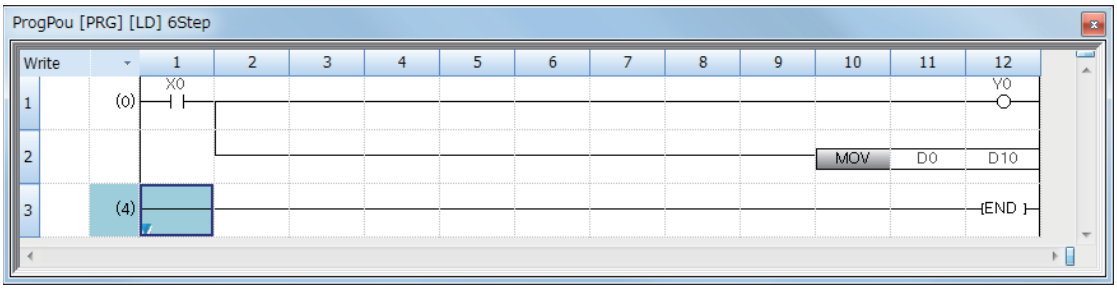
1 BASICS OF A PROGRAMMABLE CONTROLLER

1.1 Programming Languages

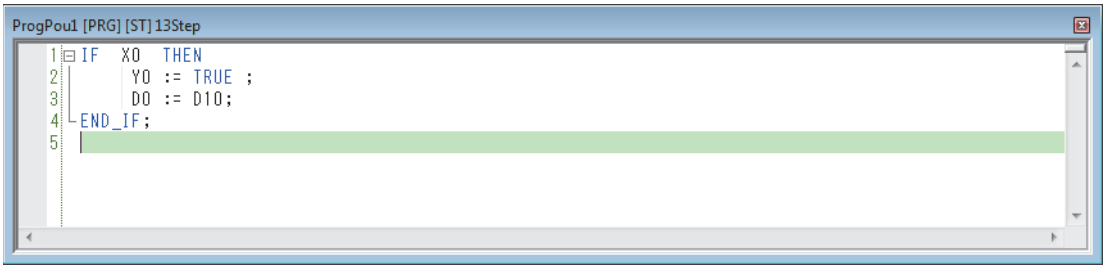
With the MELSEC iQ-R series, an optimal programming language can be selected and used according to the application.

Programming language	Description
Ladder diagram (Ladder)	A graphic language which describes ladders consisting of contacts and coils. This language is used to describe logical ladders using symbolized contacts and coils to enable easy-to-understand sequence control.
Structured text language (ST)	A textual language used to describe programs using statements (such as IF) and operators. Compared with the ladder diagram, this language can describe hard-to-describe operation processing concisely and legibly, and therefore is suitable for programming complicated arithmetic operations and comparison operations. Also, as with C, ST language can describe syntax control such as selective branches with conditional statements and repetitions with iteration statements, and thus can describe easy-to-understand, concise programs.

Ladder diagram (Ladder)



Structured text language (ST)



The same operation is described in each language.

Point

Programming in ladder is suitable for users who have knowledge and experience of sequence control and logical ladders. Programming in ST is suitable for users who have knowledge and experience of C programming.

1.2 Program

Point

- This section describes a flow from input signals to output signals.
- This section describes that a programmable controller repeatedly executes operations (scans a program).

When a programmable controller is regarded as a control circuit, it can be described with an input circuit, output circuit, and internal sequence.

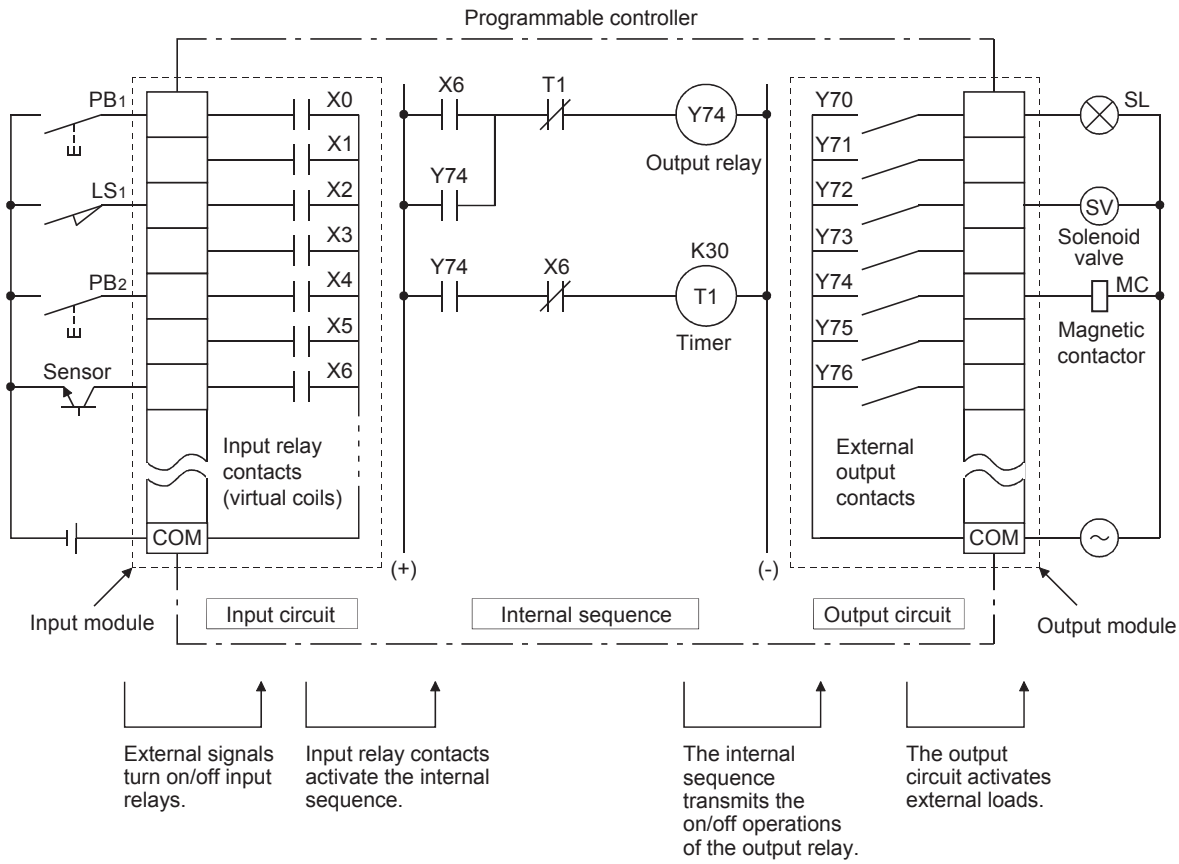


Figure 1.1 Configuration of a programmable controller

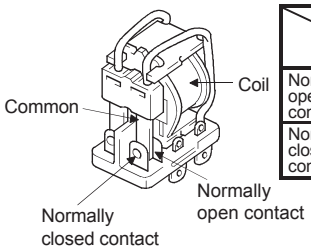
A programmable controller is an electronic device having a microcomputer at the center. A programmable controller can be regarded as a collection of relays, timers, and counters. As shown in Figure 1.1, normally open contacts and normally closed contacts are connected in series or in parallel and coils are turned on or off.

Point

A "relay", which is also called an electromagnetic relay, is a switch that relays signals. A relay is a key component that makes up a logic circuit.

1) Passing a current through a coil → Energization
- A normally open contact is closed (continuity state).
- A normally closed contact is open (non-continuity state).

2) Stopping a current flowing through a coil → Deenergization
- A normally open contact is open (non-continuity state).
- A normally closed contact is closed (continuity state).



	Coil OFF (always)	Coil ON (in operation)
Normally open contact	Non-continuity state	Continuity state
Normally closed contact	Continuity state	Non-continuity state

Operation of the internal sequence

The following describes the flow of signals in the internal sequence in figure 1.1.

1. When the sensor turns on, the coil of the input relay X6 is energized.

2. When the coil of the input relay X6 is energized, the normally open contact X6 goes in the continuity state and the coil of the output relay Y74 is energized.
(As the timer has not been energized at this time, the normally closed contact is in the continuity state.)

3. Once the coil of the output relay Y74 is energized, the external output contact Y74 goes in the continuity state and the magnetic contactor (MC) is turned on.

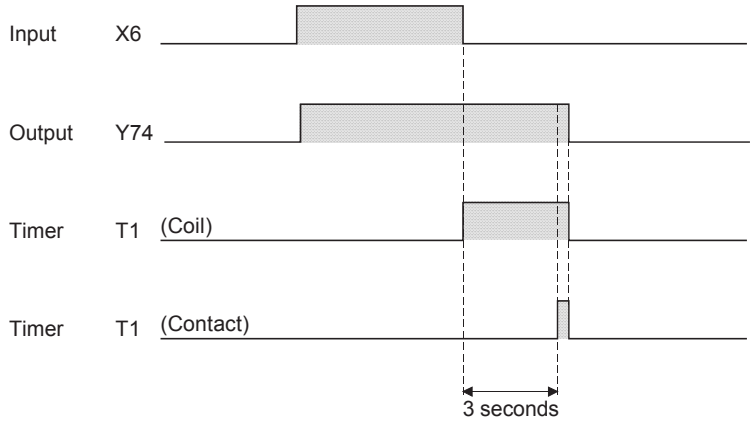
4. Turning off the sensor deenergizes the coil of the input relay X6 and the normally open contact X6 goes in the non-continuity state.
As the self-holding normally open contact Y74 of the output relay is in the continuity state, the coil remains energized. (Self-holding operation)

5. When the coil of the output relay Y74 is energized (with the normally open contact Y74 in the continuity state), turning off the sensor (with the normally closed contact X6 in the continuity state) energizes the coil of the timer T1 and the timer starts measuring time.
In three seconds (K30 means 3.0 seconds), the normally open contact of the timer goes in the continuity state and the normally closed contact goes in the non-continuity state.

6. As a result, the coil of the output relay Y74 is deenergized and the load magnetic contactor drops.
The self-holding status of the output relay is released.

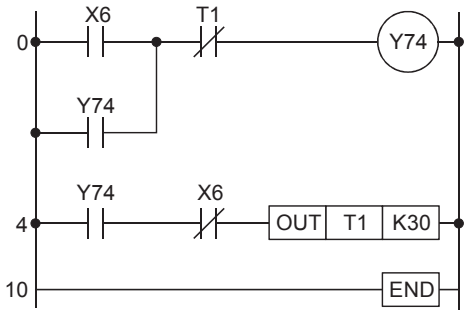
Timing chart

The following figure shows the timing chart of the operations of the input/output relays and timer.



Program

The internal sequence is regarded as programs of a programmable controller. Programs are stored in the program memory in the form similar to the following instruction list.



(a) Ladder diagram

Step number	Instruction word	Device
0	LD	X6
1	OR	Y74
2	ANI	T1
3	OUT	Y74
4	LD	Y74
5	ANI	X6
6	OUT	T1 K30
10	END	

Repetitive operation

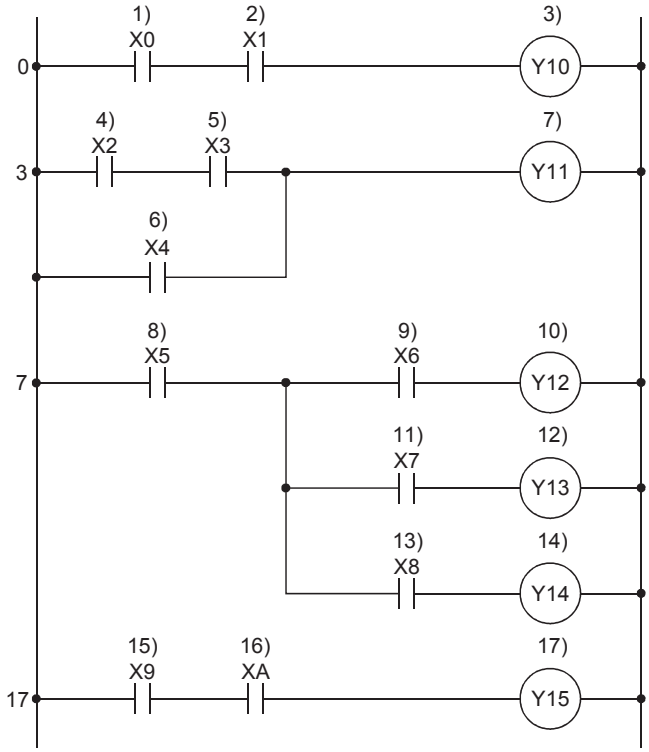
(b) Instruction list (program list)

Figure 1.2 Program

- A program consists of a large number of instruction words and devices.
- An instruction consists of instruction words and devices. Instructions are numbered to represent the order of operations. The numbers are called step numbers. (Instruction words are also called "instructions".)
- The number of steps differs depending on the types of instructions in use and the method of setting numerical values to be used for I/O numbers and operations.
  (The more complex an operation is, the more steps are needed.)
- An instruction is repeatedly executed starting from the step number 0 to the END instruction. (This is called "repetitive operation", "cyclic operation", or "scanning".) The time taken for one cycle is called operation cycle (scan time).
- The number of steps from the step number 0 to the END instruction is the length or size of a program.
- Programs are stored in the program memory inside a CPU module. The operation processing is executed in units of one ladder block.
  One ladder block starts with an operation start instruction (LD, LDI) and ends with an OUT instruction (including data instructions).

Program processing sequence

A CPU module executes operations in series from the start step of the program memory from left to right and from top to bottom (in the order of 1), 2) ... and 17)) in units of a ladder block as shown below.



1.3 System Configuration

Point

This section describes the basic configuration of a programmable controller system.

This section describes the MELSEC iQ-R series system configuration.

1.3.1 Overall configuration

The MELSEC iQ-R series programmable controller system is configured by mounting modules on a base unit.
A power supply module is mounted on the power supply slot located on the left end of a main base unit, and a CPU module is mounted on the CPU slot located on the right side of the power supply slot. Modules other than the power supply module are mounted on the slots located on the right side of the CPU slot.
Up to 7 extension base units can be added and up to 64 modules can be mounted in the entire system to build a large system.

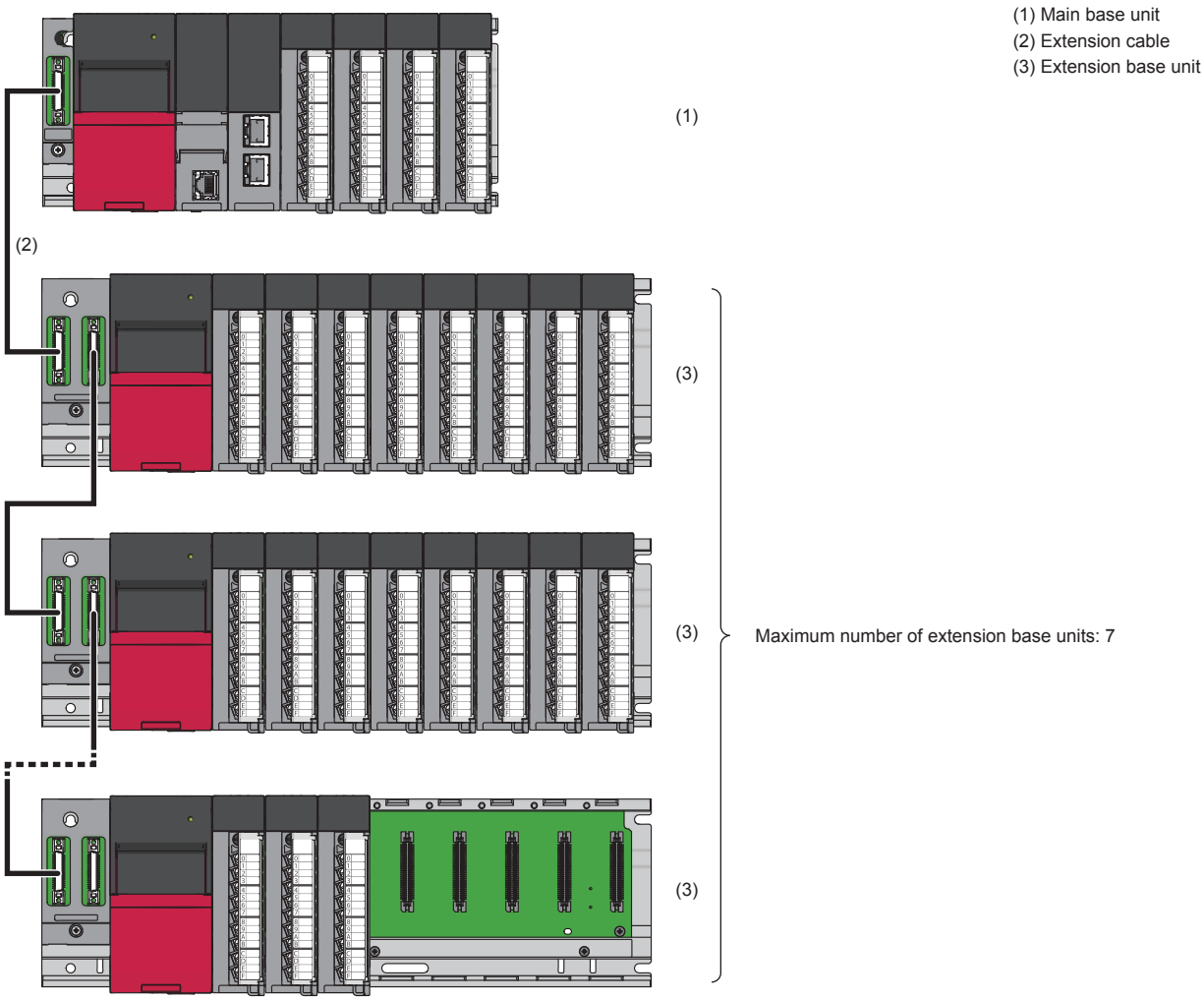
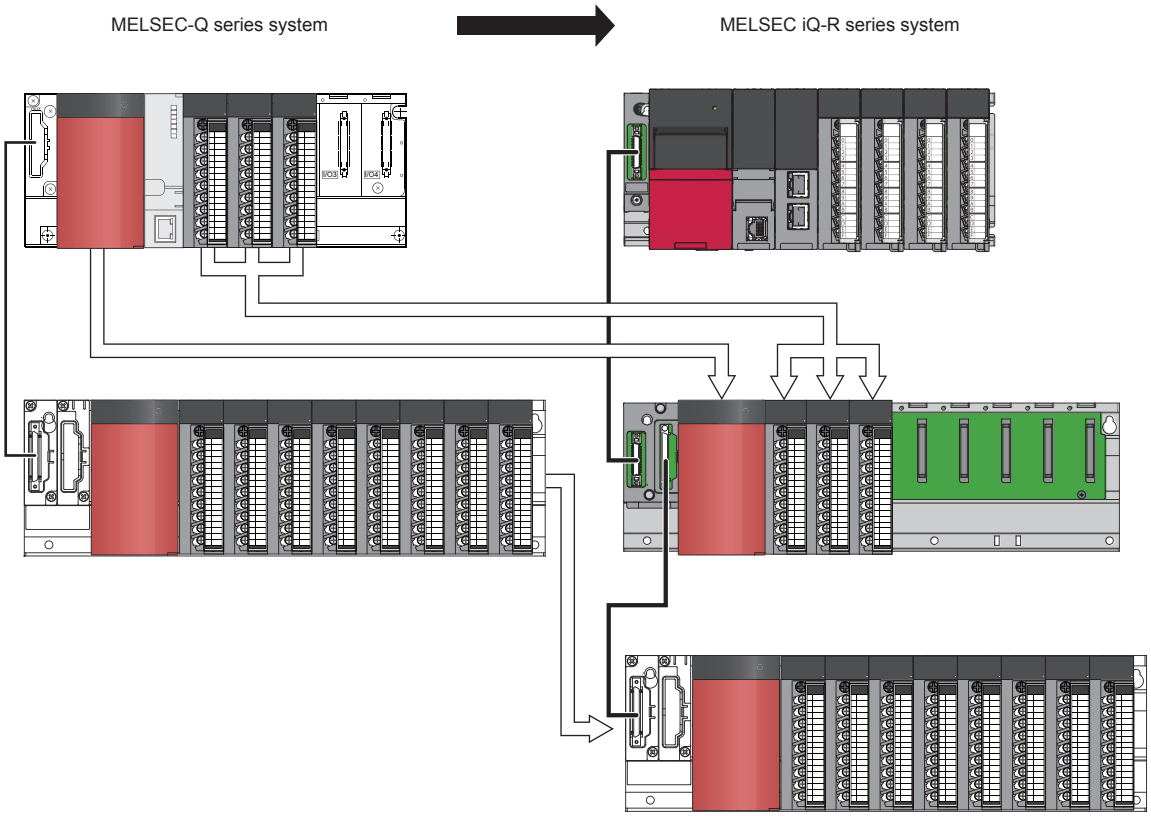


Figure 1.3 Overall configuration

Point

MELSEC-Q series modules and base units can be used by connecting the RQ extension base unit in the MELSEC iQ-R series system.
MELSEC-Q series power supply modules, I/O modules, and intelligent function modules can be mounted on the RQ extension base unit.
(📖 MELSEC iQ-R Module Configuration Manual)



Base unit

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

Type	Model	Description
Main base unit	R35B	5 slots
	R38B	8 slots
	R312B	12 slots
Extension base unit	R65B	5 slots
	R68B	8 slots
	R612B	12 slots
RQ extension base unit (For mounting MELSEC-Q series modules)	RQ65B	5 slots
	RQ68B	8 slots
	RQ612B	12 slots

Power supply module

Type	Model	Input	Output
AC power supply module	R61P	100 to 240VAC	5VDC/6.5A
DC power supply module	R63P	24VDC	5VDC/6.5A

CPU module

■Programmable controller CPU

Model	Program capacity (maximum)	Basic operation processing speed (LD instruction)	Maximum number of I/O points that can be connected to a programmable controller
R04CPU	40K steps	0.98ns	4096 points
R08CPU	80K steps		
R16CPU	160K steps		
R32CPU	320K steps		
R120CPU	1200K steps		

■Motion CPU

Model	Number of controlled axes	Operation cycle [ms]	Servo program capacity [step]
R16MTCPU	16 axes	0.222, 0.444, 0.888, 1.777, 3.555, 7.111	32K
R32MTCPU	32 axes (16 axes × 2 systems)	0.222, 0.444, 0.888, 1.777, 3.555, 7.111	32K

I/O module

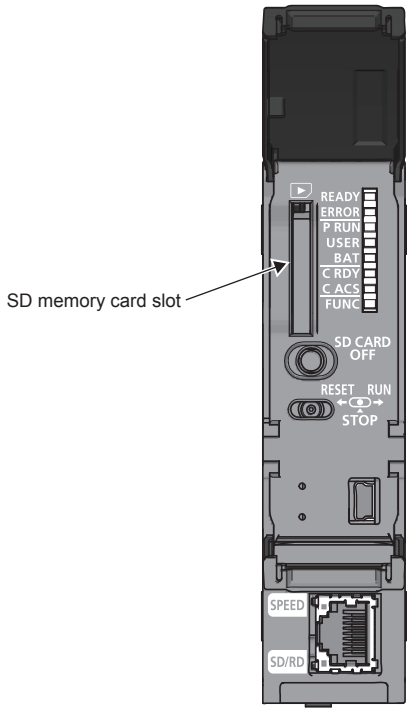
Type	Model	Description
Input module	RX10	AC input: 16 points, 100 to 240VAC (50/60Hz)
	RX40C7	DC input: 16 points, 24VDC, 7.0mA
	RX41C4	DC input: 32 points, 24VDC, 4.0mA
	RX42C4	DC input: 64 points, 24VDC, 4.0mA
Output module	RY10R2	Relay output: 16 points, 24VDC/2A, 240VAC/2A
	RY40NT5P	Transistor (sink) output: 16 points, 12 to 24VDC, 0.5A
	RY41NT2P	Transistor (sink) output: 32 points, 12 to 24VDC, 0.2A
	RY42NT2P	Transistor (sink) output: 64 points, 12 to 24VDC, 0.2A
	RY40PT5P	Transistor (source) output: 16 points, 12 to 24VDC, 0.5A
	RY41PT1P	Transistor (source) output: 32 points, 12 to 24VDC, 0.1A
	RY42PT1P	Transistor (source) output: 64 points, 12 to 24VDC, 0.1A
I/O combined module	RH42C4NT2P	DC input: 32 points, 24VDC, 4.0mA Transistor (sink) output: 32 points, 12 to 24VDC, 0.2A

Memory card

A CPU module has a built-in memory for storing parameters and programs as standard. Thus, users can execute programs without a memory card.
Users can use a memory card (SD memory card) to store a large amount of data such as boot data, comment data, logging data, database, and others.

This section describes the performance specifications of SD memory cards.

Item		L1MEM-2GBSD	L1MEM-4GBSD
Type		SD	SDHC
Capacity		2G bytes	4G bytes
Number of storable files		256	32767
Number of storable folders		256	32767
Number of insertions and removals		Limited to 500 times	
External dimensions	Height	32mm	
	Width	24mm	
	Depth	2.1mm	
Weight		2g	

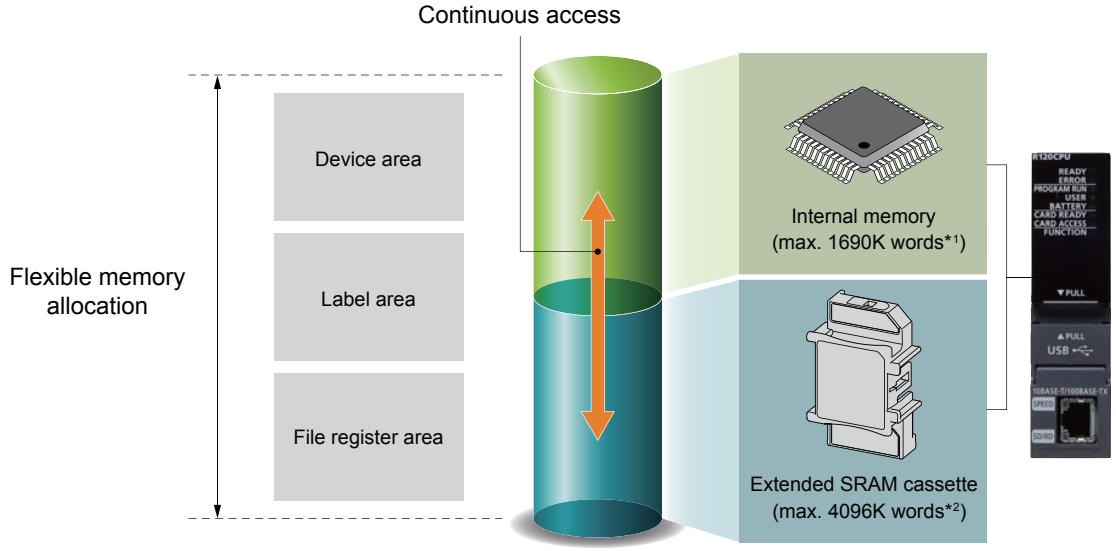


Precautions

All SD memory cards to be used in the CPU module need to be formatted. An SD memory card is unformatted when purchased. Before using the card, insert it into the CPU module and format it using the engineering tool. Do not format an SD memory card using a personal computer. (📖 GX Works3 Operating Manual)

Extended SRAM cassette

Users can extend device/label memory areas up to 5786K words by mounting an extended SRAM cassette on the programmable controller CPU. Users can assign device/label ranges and others in the extended areas as the areas connected with the built-in memory of the CPU module. Thus, users do not need to consider boundaries between each memory area, allowing them to do programming easily.



*1 For the R120CPU

*2 For the NZ2MC-8MBS (8M bytes)

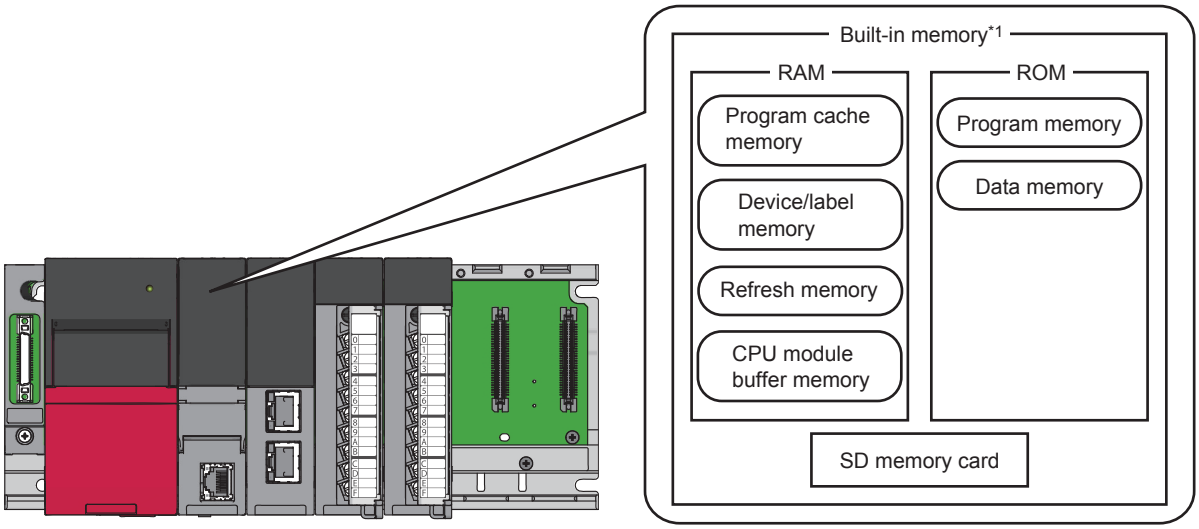
When users write a security key in an extended SRAM cassette, the cassette plays a security function. This function prevents programs from being executed with the CPU module that does not have the same security key as the one written in the program file.

1.4 Memory Configuration of the CPU Module

Point

This section describes the role of each memory.

The following shows the memory configuration of the CPU module.



*1 The built-in memory is a generic term for internal memory areas of the CPU module.

Point

The usage of the memory can be checked from the engineering tool. For details, refer to the following.

GX Works3 Operating Manual

RAM

This memory is for using file registers, local devices, and sampling trace files without a memory card. Using RAM areas as file registers enables users to quickly access the areas in the same way as data registers. This memory is also used for storing module error collection files.

■Program cache memory

This memory is used for program operations.

When the system is powered on or the CPU module is reset, programs stored in the program memory are transferred into the program cache memory and executed.

■Device/label memory

The device/label memory has the following areas.

Area		Application
Device area		User device
Label area	Label area	Global label and local label
	Latch label area	Global label and local label with latch specified
Local device area		Local device (excluding index register)
File storage area		File register file and other data*2

*2 File register files which are stored in the area for storing file register files can be written or read in file unit.

■Refresh memory

This memory is used to store data used to refresh communication with the intelligent function module.

■CPU buffer memory

This memory is used by the Ethernet function or in data communication between multiple CPU modules.

ROM

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

■Program memory

This memory is used to store necessary programs and parameters for the CPU module to perform operations.
Programs stored in the program memory are transferred into the program cache memory and executed.

■Data memory

This memory is used to store parameter files, device comment files, and/or other folders/files.

SD memory card

This memory is used to store the folder/files created by a function using the SD memory card and other folders/files.
The folder configuration is the same as the data memory.

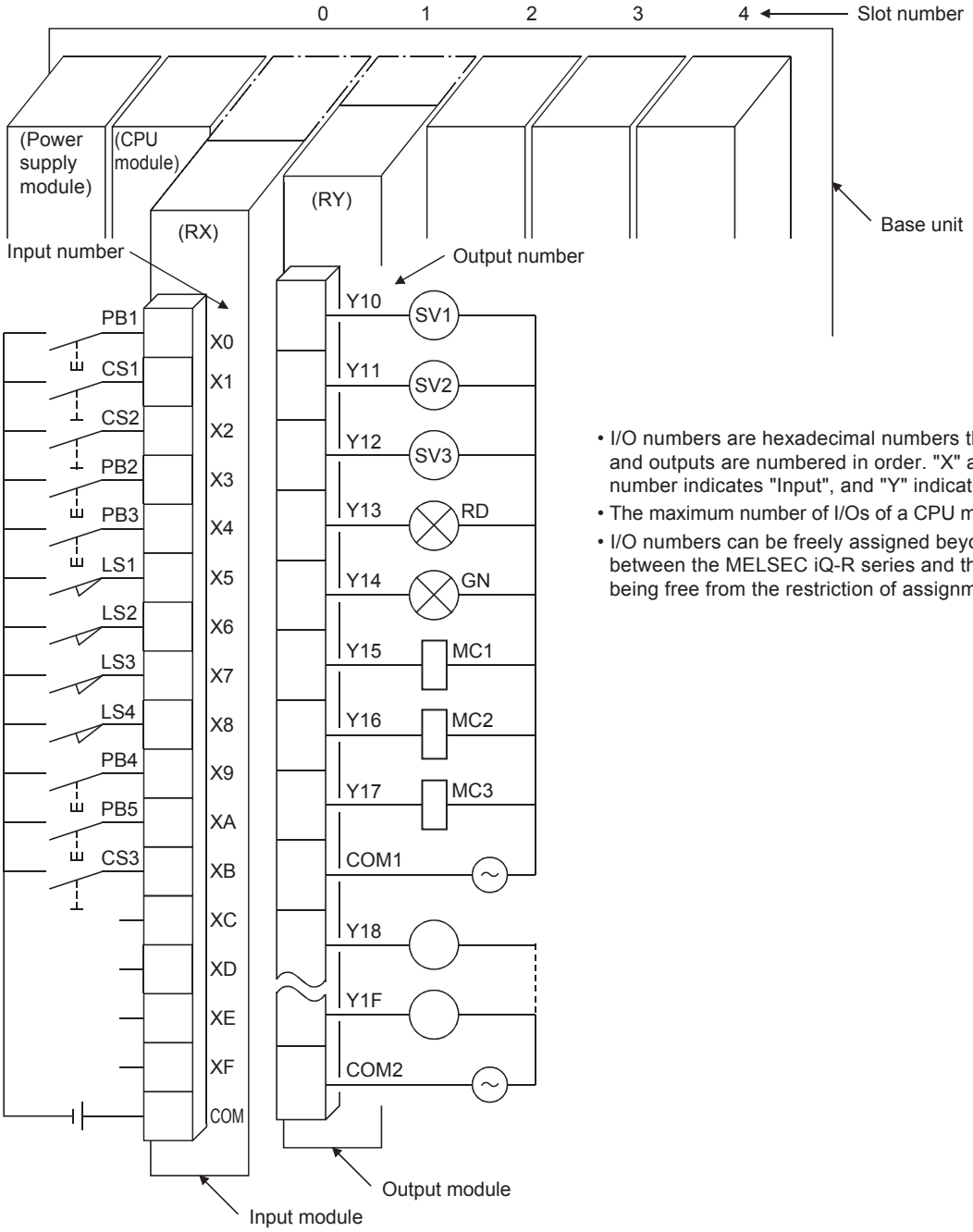
1.5 External I/O Signals and I/O Numbers

Point

This section describes how to assign I/O numbers for the MELSEC iQ-R series.

Wiring of I/O devices

Signals output from external input devices are replaced with the input numbers determined depending on the mounting position and terminal numbers of an input module, and used in programs. For outputs (coils) of operation results, the output numbers determined depending on the mounting position and the terminal numbers of an output module where an external output module has been connected.

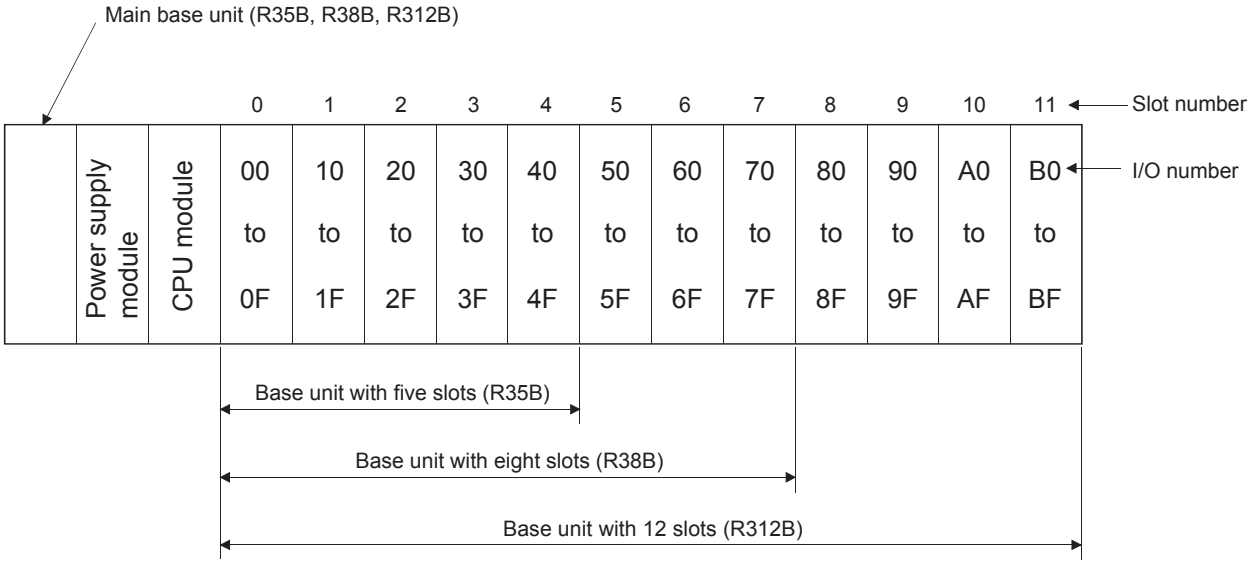


- I/O numbers are hexadecimal numbers that start from 0. Inputs and outputs are numbered in order. "X" at the beginning of a number indicates "Input", and "Y" indicates "Output".
- The maximum number of I/Os of a CPU module module is 4096.
- I/O numbers can be freely assigned beyond a boundary between the MELSEC iQ-R series and the MELSEC-Q series, being free from the restriction of assignment orders.

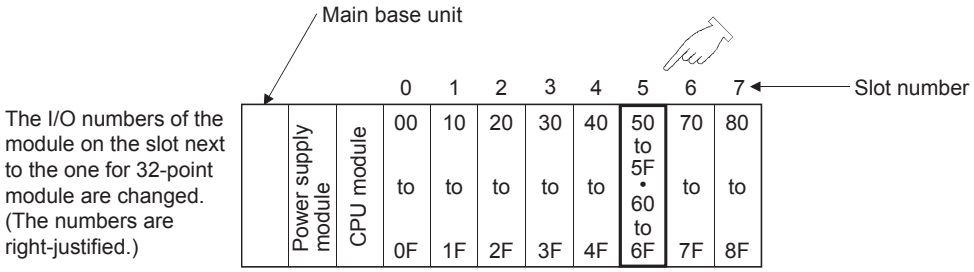
Figure 1.4 Wiring of I/O devices

I/O numbers of a main base unit

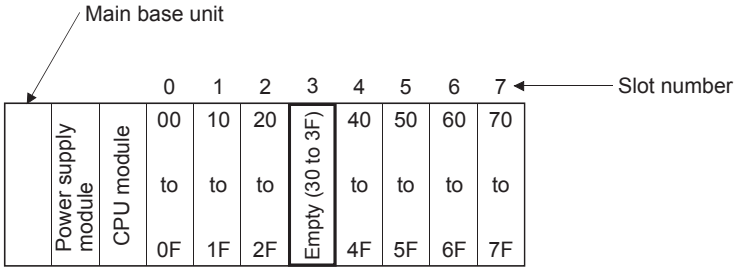
I/O numbers of I/O modules mounted on a main base unit are assigned as follows. I/O numbers of an intelligent function module are assigned in the same way.



- I/O numbers of one slot (one module) are assigned in ascending order in increments of 16 points (0 to FH).
  The case where 16-point modules have been mounted to all slots is used as standard.
  For example, the following figure shows I/O numbers of when a 32-point module is mounted to the fifth slot.



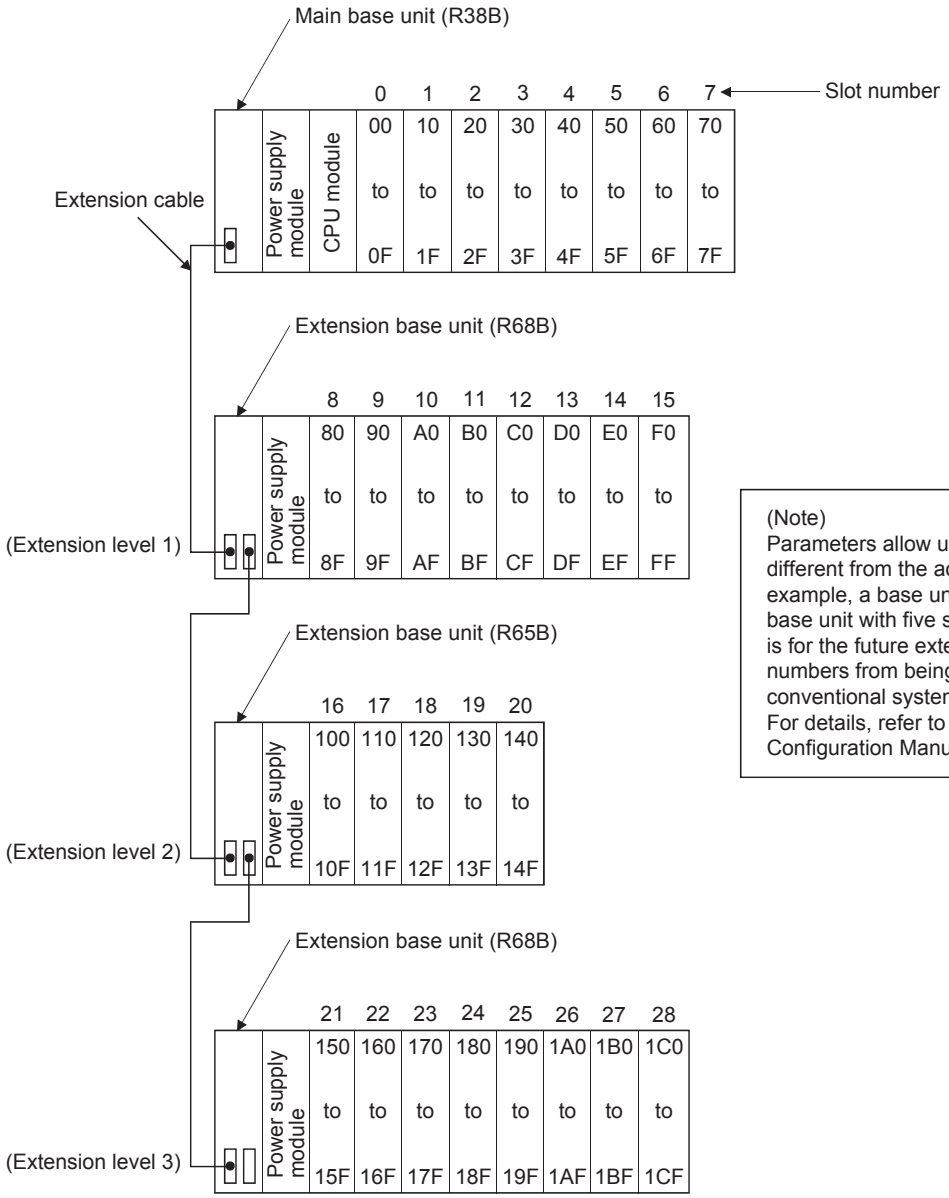
- I/O numbers are assigned to empty slots (slot where no I/O modules are mounted).
  For example, when the third slot is empty, I/O numbers are assigned as follows in the initial setting.
  The number of assigned points can be changed in the setting.



I/O numbers of an extension base unit

Connect extension base units when the number of slots of the main base unit is insufficient.

I/O numbers are assigned as follows in the initial setting. I/O numbers of an intelligent function module are assigned in the same way.



(Note)
Parameters allow users to set the number of slots different from the actual number of slots. For example, a base unit with 12 slots can be set as a base unit with five slots and vice versa. This setting is for the future extension, or to prevent I/O numbers from being unintentionally shifted when a conventional system is replaced with a new one. For details, refer to the MELSEC iQ-R Module Configuration Manual.

- I/O numbers are also assigned to the slots on an extension base unit in ascending order in increments of 16 points.
- As the start I/O number of an extension base unit, the number after the last number of the main base unit or the previous extension base unit is used.
- To empty slots and the areas with no slots, the value "0" can be assigned with parameters.
- Up to seven base units can be extended including the extension base unit, RQ extension base unit, and MELSEC-Q series extension base unit.

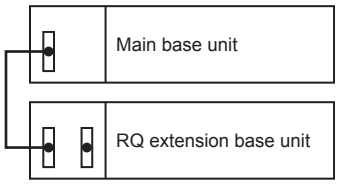
Point

This section describes the connection between the RQ extension base unit and MELSEC-Q series extension base unit.

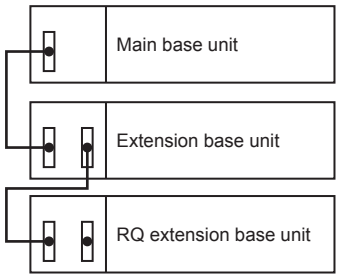
The RQ extension base unit is connected to the lower level of the main base unit or MELSEC iQ-R series extension base unit with a MELSEC iQ-R series extension cable.

(📖 MELSEC iQ-R Module Configuration Manual)

- When the RQ extension base unit is connected to the lower level of the main base unit

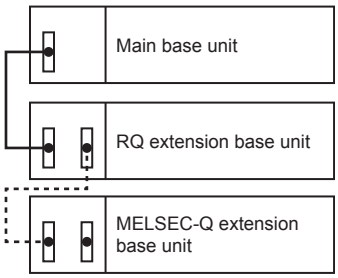


- When the RQ extension base unit is connected to the lower level of the extension base unit

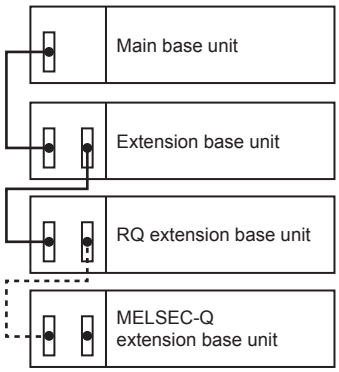


When additional MELSEC-Q series modules are mounted, the MELSEC-Q series extension base unit is connected to the lower level of the RQ extension base unit with a MELSEC-Q series extension cable. (The dot lines show the MELSEC-Q series extension cables.)

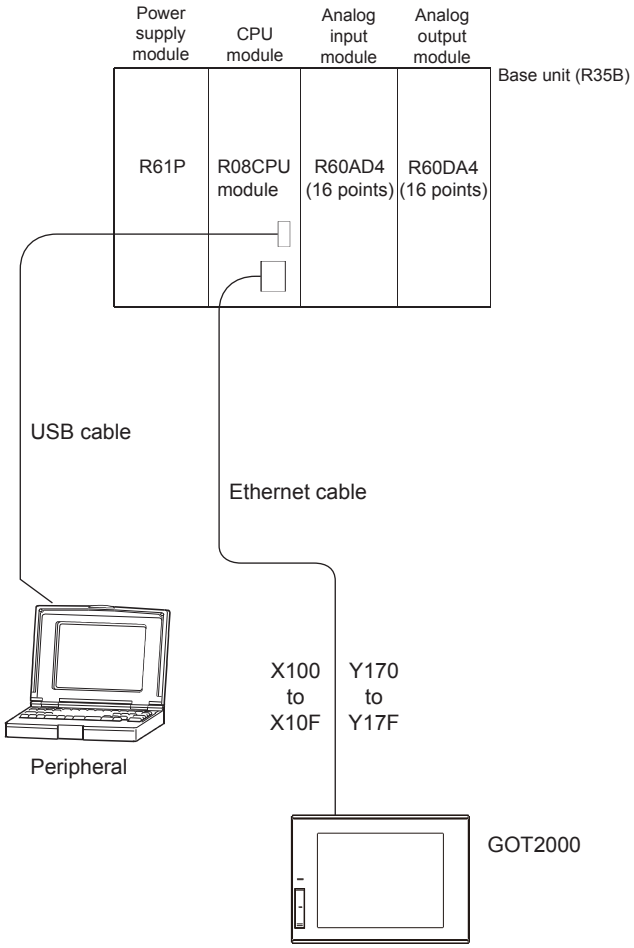
- When the RQ extension base unit is connected to the lower level of the main base unit



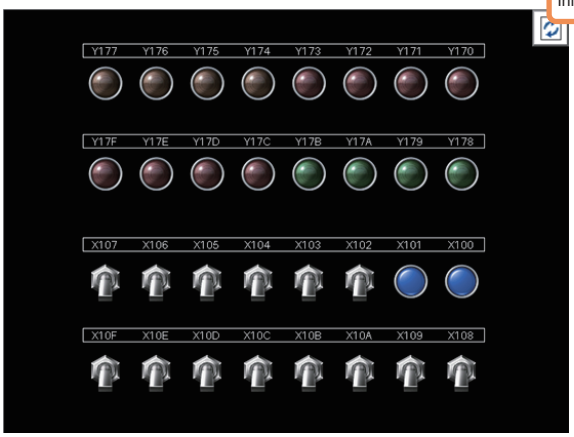
- When the RQ extension base unit is connected to the lower level of the extension base unit



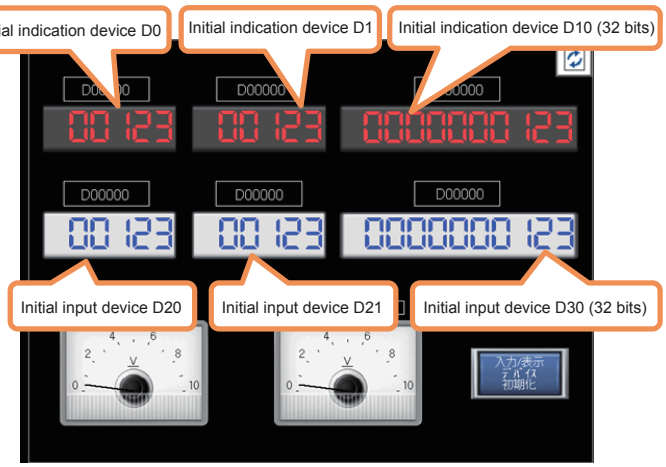
1.6 System Configuration and I/O Numbers of the Demonstration Machine



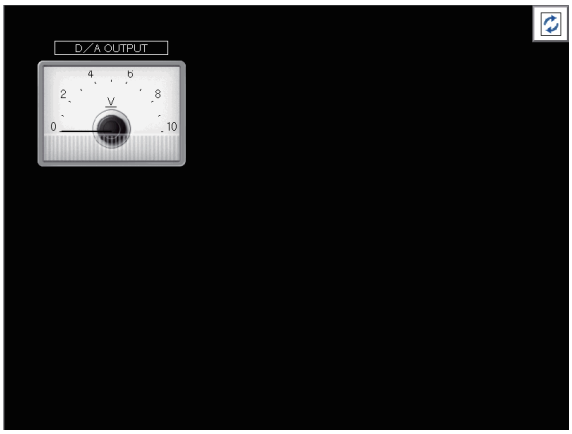
GOT screen display



Screen 1



Screen 2



Screen 3

Upper section: The indication device can be changed.
Lower section: Data is displayed.

Upper section: The input device can be changed.
Lower section: The input data can be set and displayed.

- Touching switches the screen.
- The initial value is automatically set to a device number in the upper section (trigger action function).
- Touching the "Initialize Input/Indication Device" button also initializes the device number.

2 OPERATING GX Works3

Point

This chapter describes the basic operations of GX Works3.

GX Works3 is an engineering tool for setting, programming, debugging, and maintenance of projects for the MELSEC iQ-R series programmable controllers and others on Windows®.
Compared with GX Works2, the functionality and operability of GX Works3 have been improved.
For changes in the window display, refer to the following.
📖 MELSEC iQ-R Module Configuration Manual

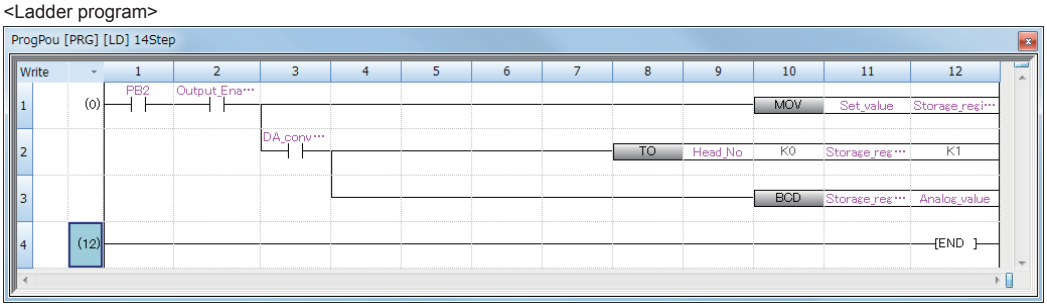
2.1 Main Functions of GX Works3

GX Works3 manages programs and parameters in a project for each CPU module.
GX Works3 has the following main functions.

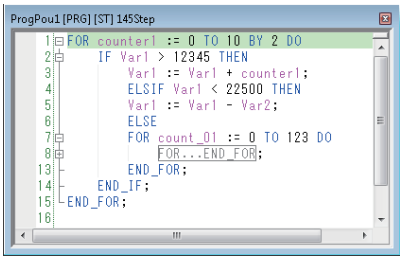
Creating programs

Users can create programs in a desired programming language, such as ladder or ST, depending on the processing.

<Ladder program>

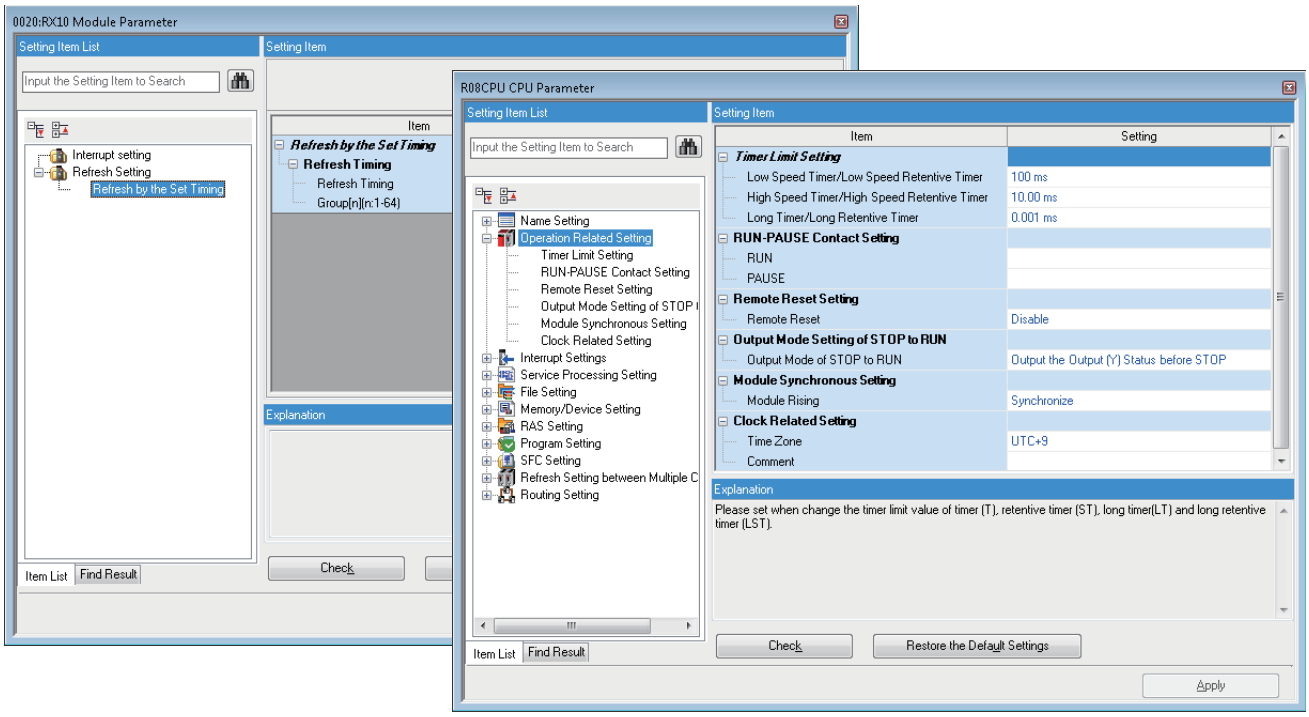


<ST program>



Setting parameters

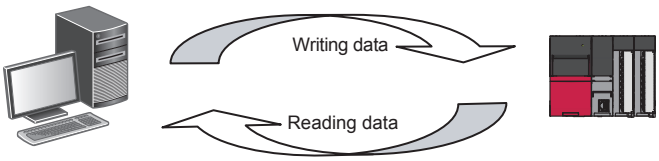
Users can set parameters for CPU modules, I/O modules, and intelligent function modules.



Reading/writing data from/to the CPU module

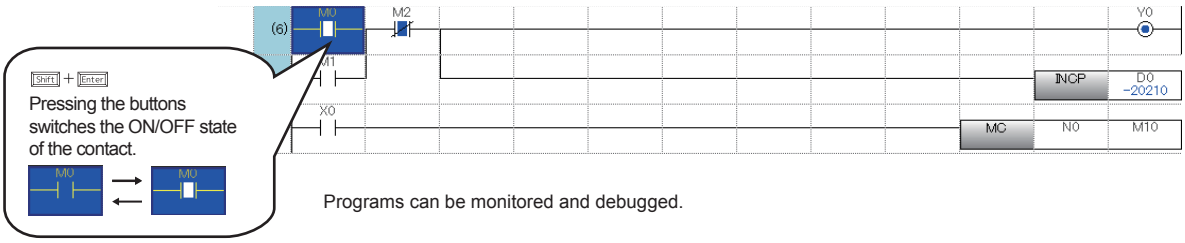
Users can read/write created sequence programs from/to the CPU module by using the "Write to PLC" and "Read from PLC" functions.

Users can edit sequence programs with the online change function even while the CPU module is in the RUN state.



Monitoring and debugging programs

Users can write created sequence programs to the CPU module and monitor data during operation, such as device values.

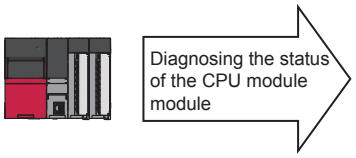


Programs can be monitored and debugged.

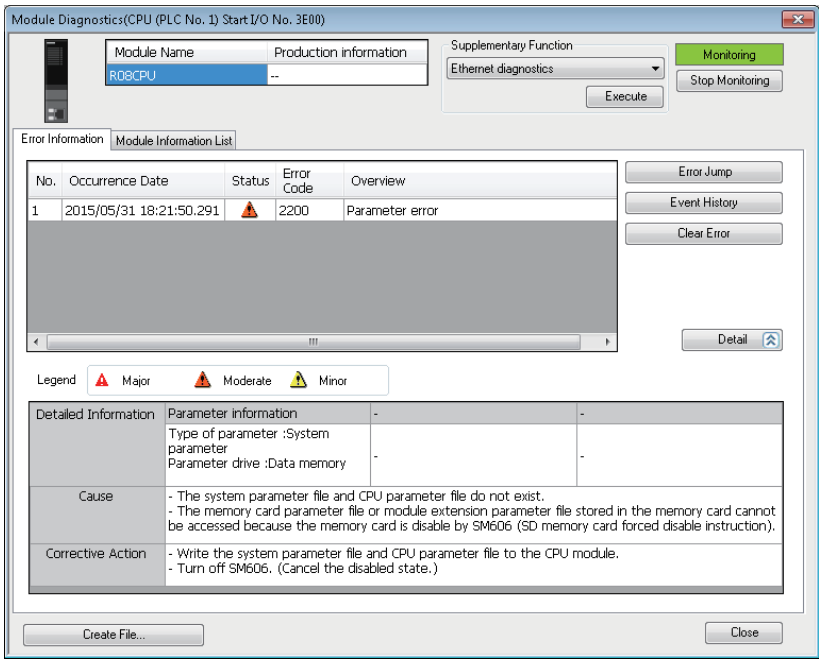
Diagnostic function

GX Works3 make diagnoses on the current error status and error history of the CPU module or network. With the diagnostic function, system recovery can be completed in a short time.

The system monitor shows detailed information on intelligent function modules and others. This feature helps users to shorten the time taken for system recovery when an error occurs.



Diagnosing the CPU module module ("Module Diagnostics" window)



2.2 Operations Before Creating a Ladder Program

2.2.1 Starting GX Works3

Operating procedure

Select [MELSOFT] ⇨ [GX Works3] ⇨ [GX Works3] from the Windows® Start menu*1.

*1 Select [Start] ⇨ [All apps] or [Start] ⇨ [All Programs].

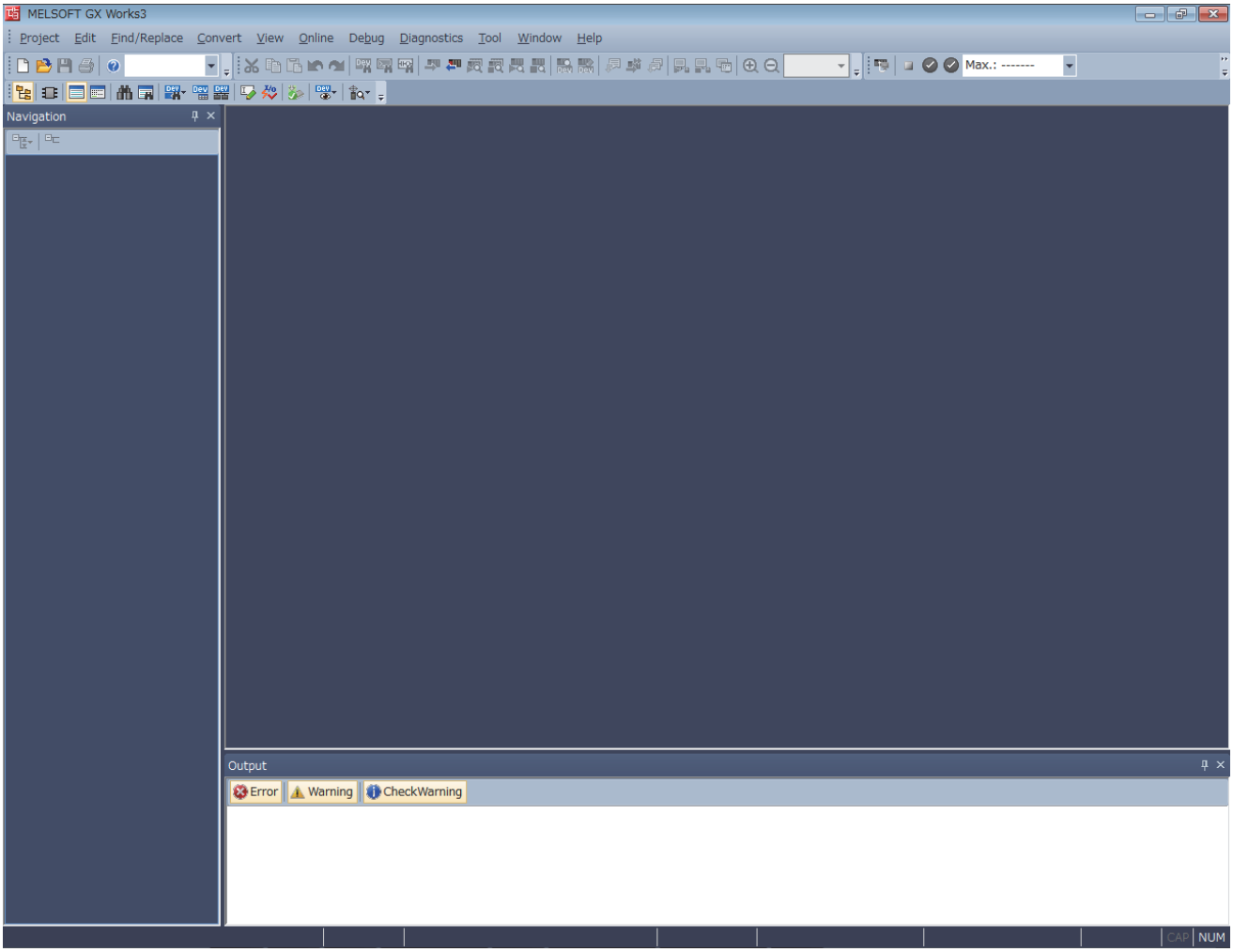
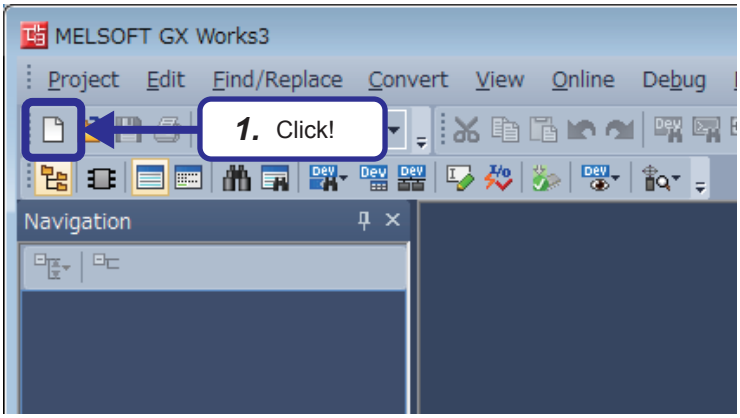


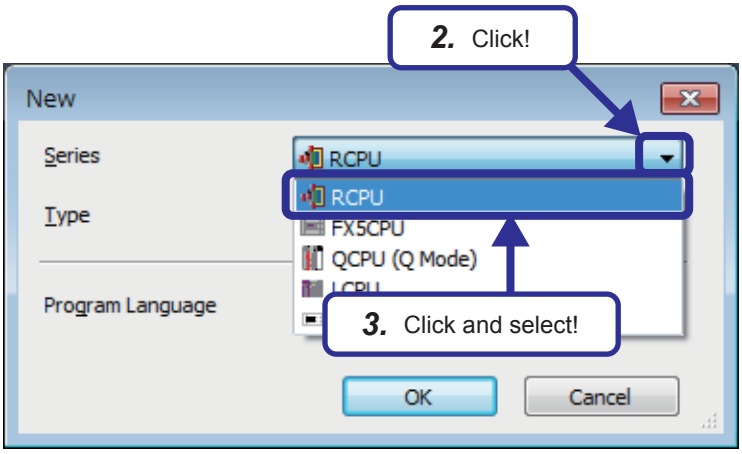
Figure 2.1 Startup window of GX Works3

2.2.2 Creating a new project

Operating procedure

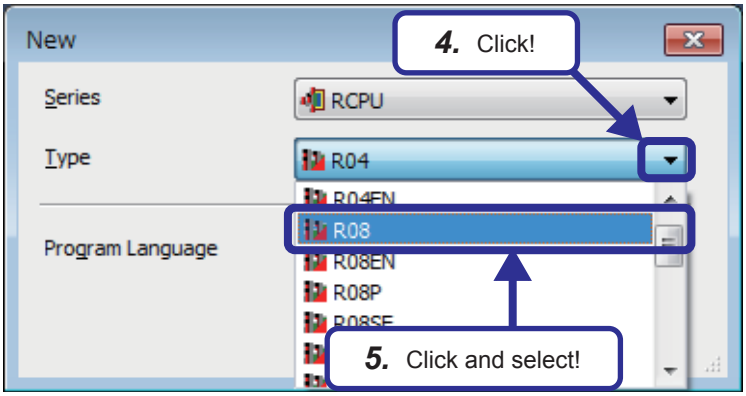


1. Click on the toolbar or select [Project] ⇨ [New] from the menu (Ctrl + N).



2. Click the list button of "Series".

3. Select "RCPU" from the drop-down menu.

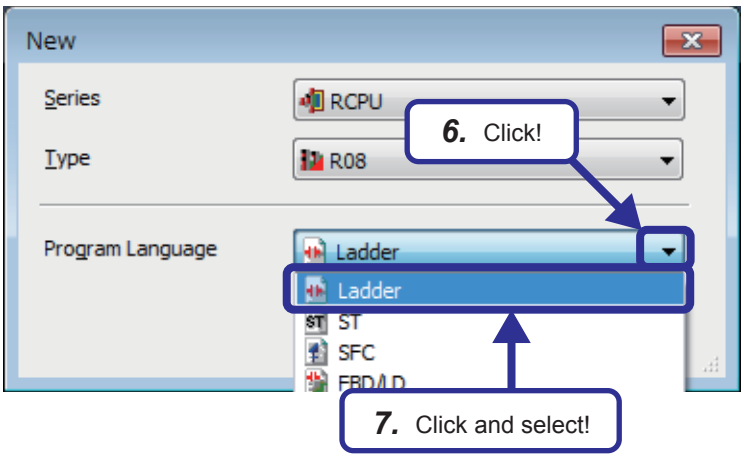


4. Click the list button of "Type".

5. Select "R08" from the drop-down menu.

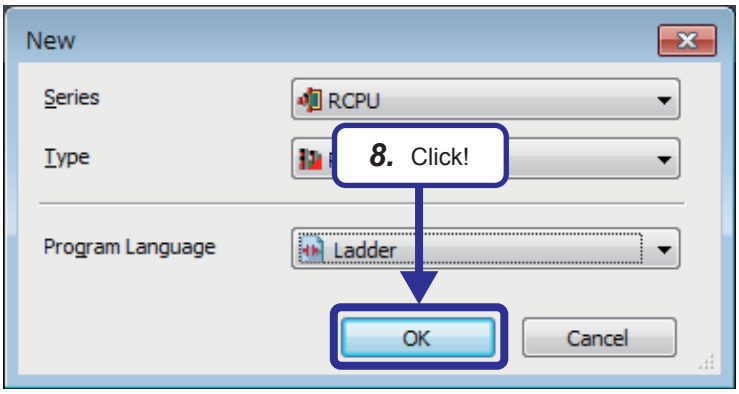
(To the next page)

(From the previous page)

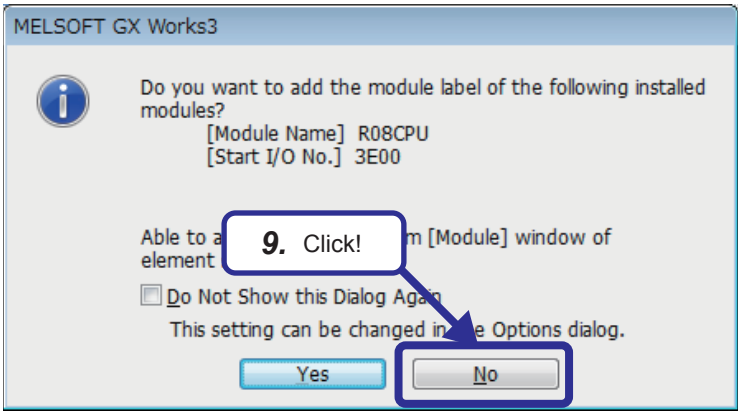


6. Click the list button of "Program Language".

7. Select "Ladder" from the drop-down menu.



8. Click the [OK] button.

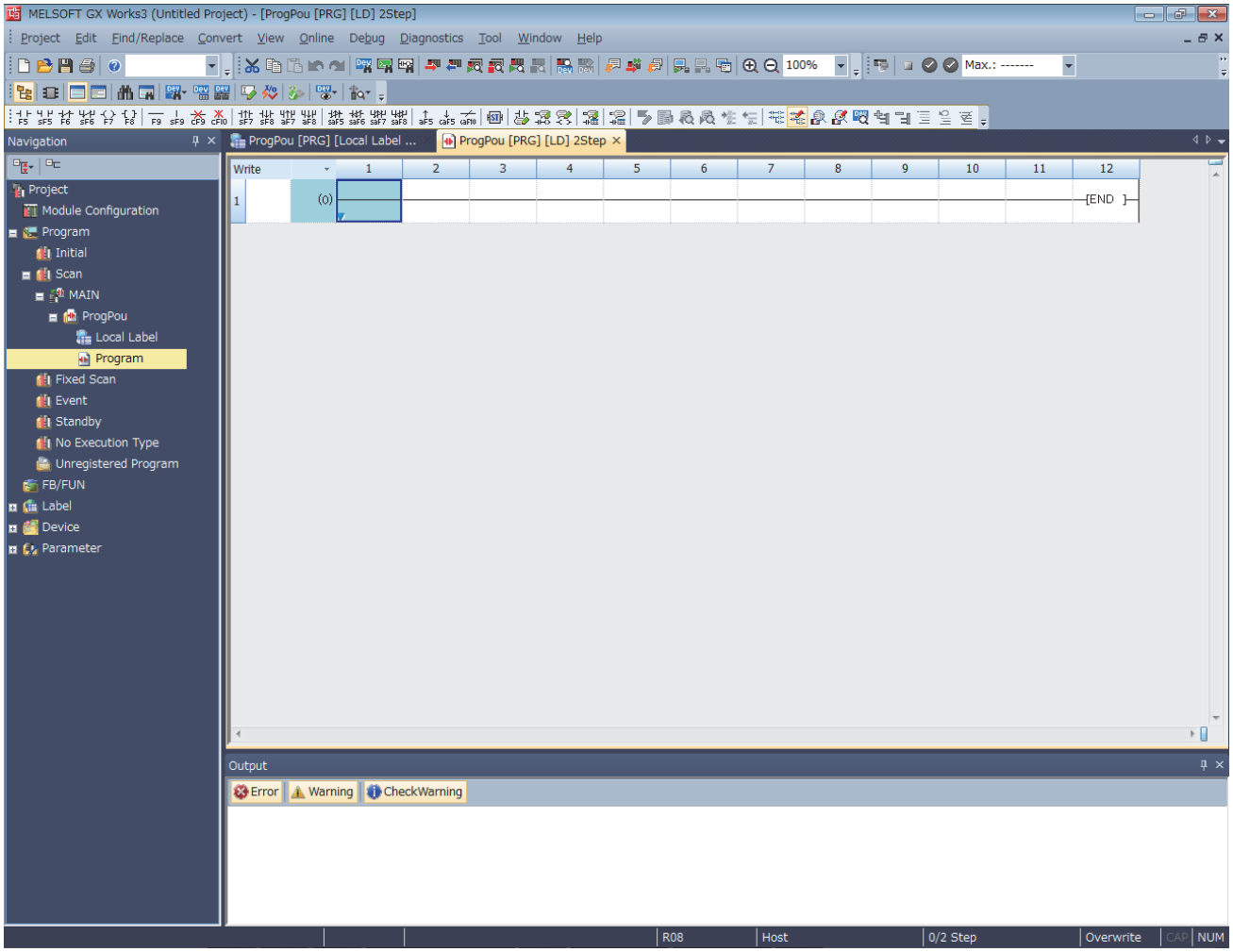


9. The confirmation window for adding the module label of the selected module type ("R08" in this case) appears. Click the [No] button.

(To the next page)

(From the previous page)

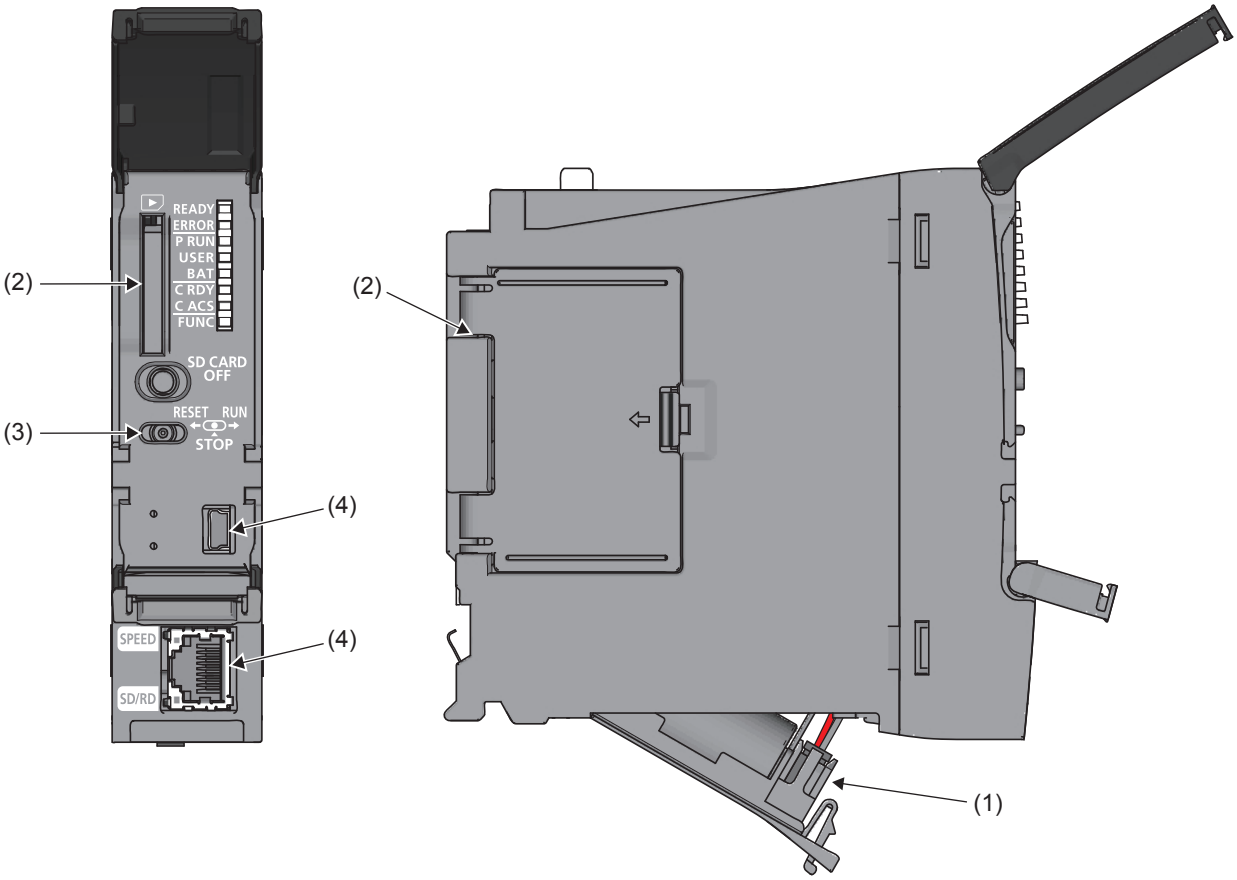
10. A new project is created.



2.3 Preparations for Stating the CPU Module

Before writing a program to the CPU module, configure the switch setting, initialize the built-in memory, and make other preparations.

Perform the following (1) to (5) operations.



(1) Connecting a battery (☞ Page 2 - 9 Installing a battery)
A battery has not been connected at the factory shipment. Connect a battery connector.

(2) Inserting an extended SRAM cassette and an SD memory card
(☞ Page 2 - 9 Inserting or removing an extended SRAM cassette, ☞ Page 2 - 11 Inserting and removing an SD memory card)
Insert an extended SRAM cassette or an SD memory card or both to the CPU module as needed.

(3) Setting the switch position
Set the RUN/STOP/RESET switch to the STOP position.

(4) Connecting a personal computer to the CPU module using a USB cable or an Ethernet cable

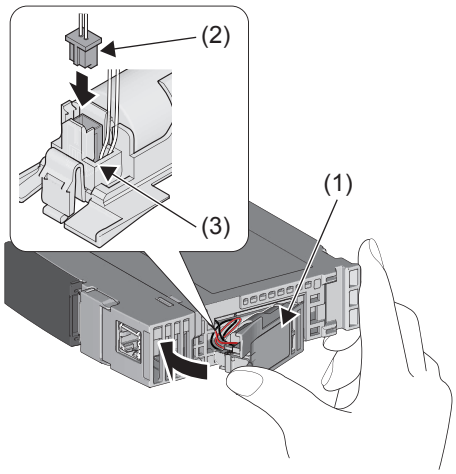
(5) Powering on the system
Check the following items, and then power on the system.
- A cable is correctly connected to the power supply.
- The power supply voltage is within the specified range.
- The CPU module is in the STOP state.

2.3.1 Installing a battery

Install a battery to the CPU module.

■Installation procedure

The connector plug of the Q6BAT is disconnected from the jack of the CPU module before shipment. To use the battery, connect the connector, following the procedure below.

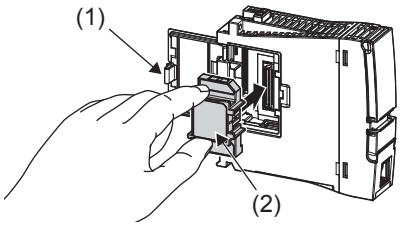


1. Open the battery cover located on the bottom of the CPU module.
2. Check that the Q6BAT (1) is correctly installed.
3. Check the direction and securely insert the connector plug of the Q6BAT (2) to the jack (3) of the CPU module.
4. Close the battery cover.

2.3.2 Inserting or removing an extended SRAM cassette

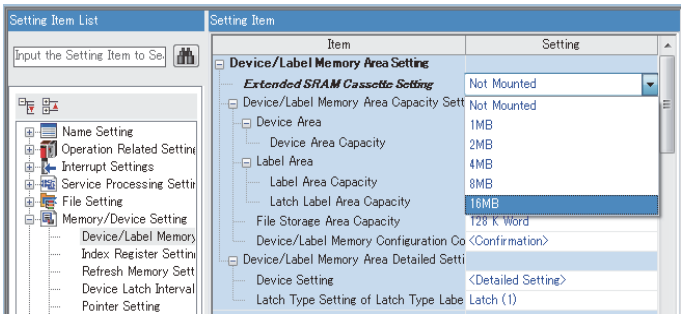
■Insertion procedure

Insert an extended SRAM cassette while the programmable controller is powered off.



1. Open the cassette cover (1) located on the side of the CPU module.
2. Hold the top and bottom of the tab (2) of an extended SRAM cassette (with the notched edge facing to the right), and insert the cassette straight into the connector. After inserting the cassette, check that it is inserted completely.
3. Close the cover, and mount the CPU module on the base unit.
4. Power on the programmable controller.
5. Set the capacity of the inserted cassette in the CPU parameters ("Extended SRAM Cassette Setting") using the engineering tool.

   [CPU Parameter] ⇨ [Memory/Device Setting] ⇨ [Device/Label Memory Area Setting] ⇨ [Extended SRAM Cassette Setting]



6. Using the engineering tool, check that SM626 (Extended SRAM cassette insertion flag) is on.

Precautions

- When the extended SRAM cassette is removed, all of the data on the device/label memory are erased. Back up the program and data before replacing the cassette.
- If the capacity of the extended SRAM cassette differs before and after the replacement, the ERROR LED of the CPU module may flash. But, it is not an error. Change the capacity setting in the CPU parameters. (Refer to step 5 above.)

Restriction

The extended SRAM cassette for the Universal model QCPU (Q4MCA-□MBS) cannot be used.

■Removal procedure

Remove the extended SRAM cassette while the programmable controller is powered off.

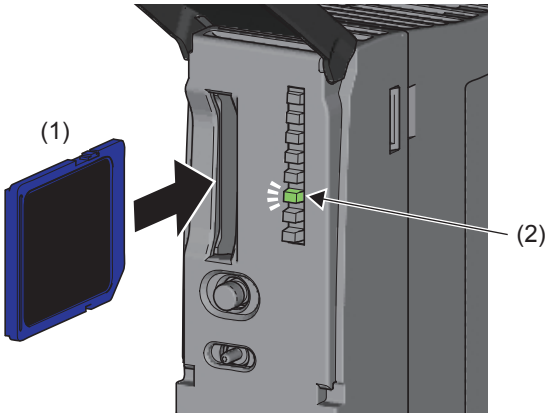
1. Read the data on the device/label memory from the CPU module, and save it in advance using the engineering tool. (When the extended SRAM cassette is removed, all of the data on the device/label memory are erased.)
2. Power off the programmable controller.
3. Remove the CPU module from the base unit, and open the cassette cover located on the side of the CPU module.
4. Hold the top and bottom of the tab of the extended SRAM cassette, and pull the cassette straight out of the connector.
5. Close the cover, and mount the CPU module back on the base unit.
6. Power on the programmable controller.
7. Set "Extended SRAM Cassette Setting" in the CPU parameters to "Not Mounted".

2.3.3 Inserting and removing an SD memory card

Insert an SD memory card to the CPU module as needed.

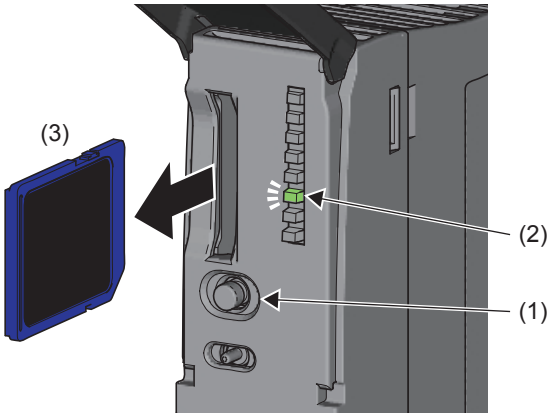
■Insertion procedure

Check the direction and insert an SD memory card, following the procedure below.



1. Insert an SD memory card (1) into the card slot until it clicks with the notched edge in the direction as illustrated. After inserting the cassette, check that it is inserted completely. Poor contact may cause malfunction.
2. The CARD READY LED (2) starts flashing. When the card is ready to be used, the CARD READY LED stops flashing and turns on.
3. If the CARD READY LED does not turn on even after the card is inserted, check that SM606 (SD memory card forced disable instruction) and SM607 (SD memory card forced disable status flag) are off.

■Removal procedure



1. Press the SD memory card access control switch (1) for one second or longer to disable access to the card.
2. The CARD READY LED (2) flashes during the access stop processing, and turns off upon completion of the processing.
3. Push in and release the SD memory card (3), and then pull the card out of the slot.

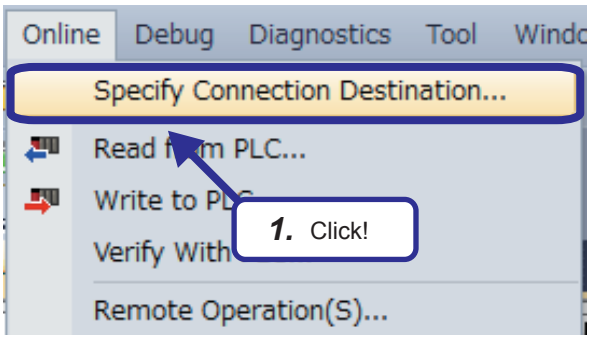
Precautions

- Follow the procedure above when inserting or removing the SD memory card while the system is powered on. If not, the data on the SD memory card may corrupt.
- If any function that accesses the SD memory card is being executed when the SD memory card access control switch is pressed to remove the card, the CARD READY LED turns off after the processing of the function is completed. For this reason, the time required until the LED turns off differs depending on the function being executed.
- If SM605 (Memory card remove/insert prohibit flag) is on, the CARD READY LED does not turn off even if the SD memory card access control switch is pressed. If not, turn on SM606 (SD memory card forced disable instruction) to forcibly disable access to the card.

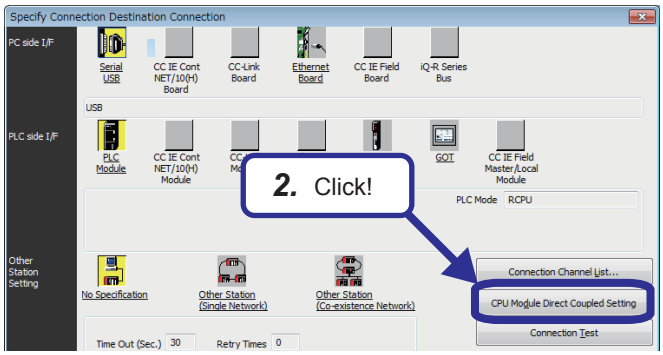
2.3.4 Specifying connection destination

Specify the connection destination for accessing the CPU module.

Operating procedure

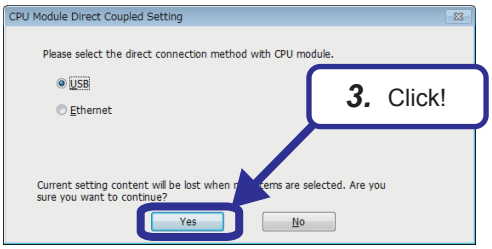


1. Select [Online] → [Specify Connection Destination] from the menu of the engineering tool.

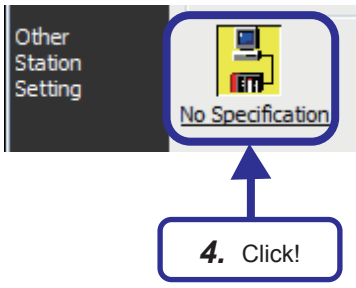


2. Click the [CPU Module Direct Coupled Setting] button on the "Specify Connection Destination Connection" window.

The "CPU Module Direct Coupled Setting" dialog box appears.



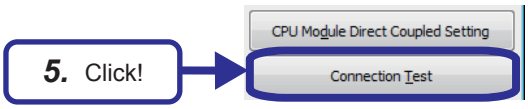
3. Select the connection method, and click the [Yes] button.



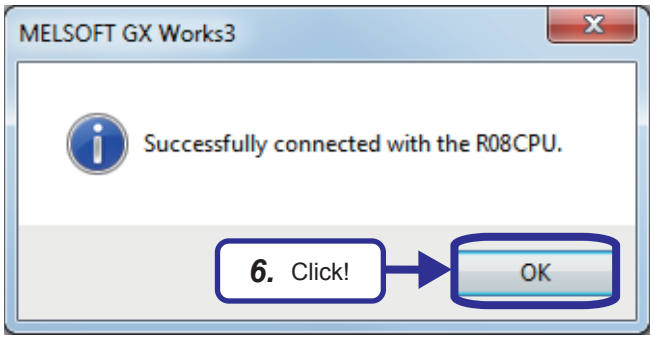
4. Click "No Specification" of "Other Station Setting".

(To the next page)

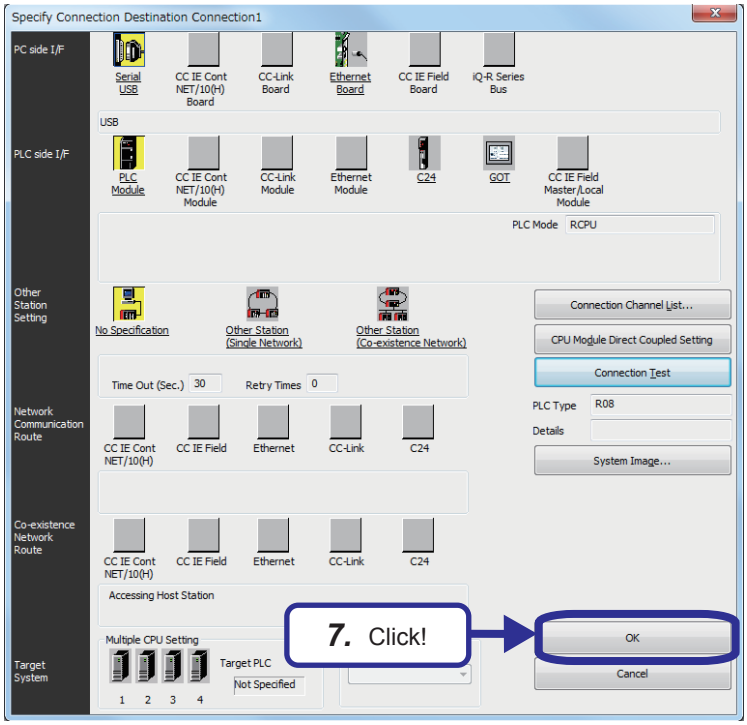
(From the previous page)



5. Click the [Connection Test] button.



6. Check that the CPU module is successfully connected, and click the [OK] button.



7. Click the [OK] button.

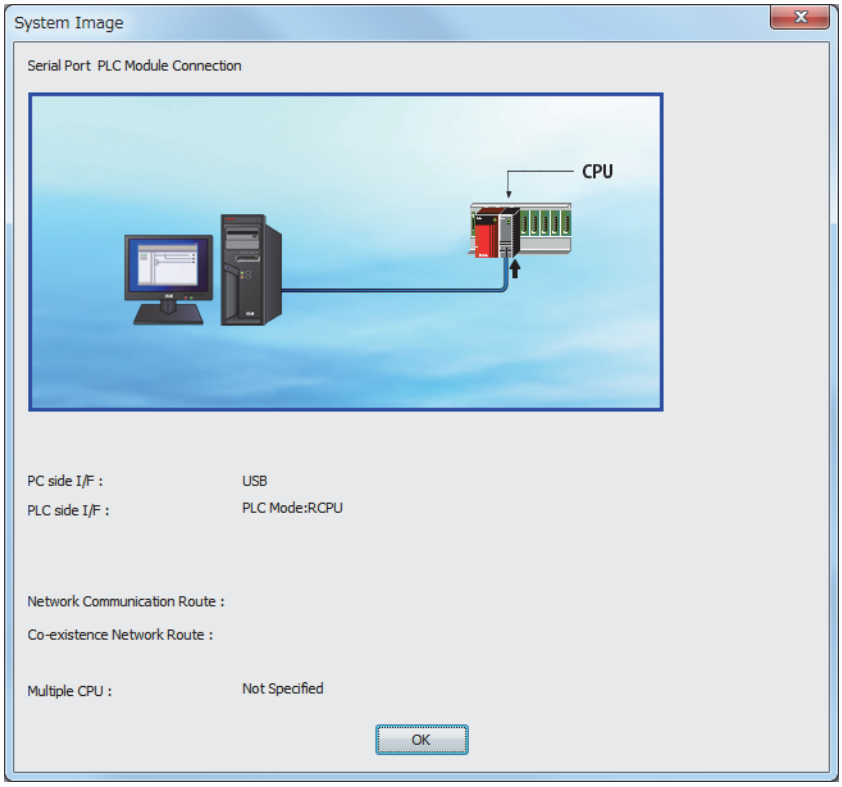
Point

- CPU Module Direct Coupled Setting

  With this setting, the personal computer is directly connected to the CPU module (connection destination) with a USB cable or an Ethernet cable. This setting is convenient for switching the connection destination between another station and the own station.

- System Image...

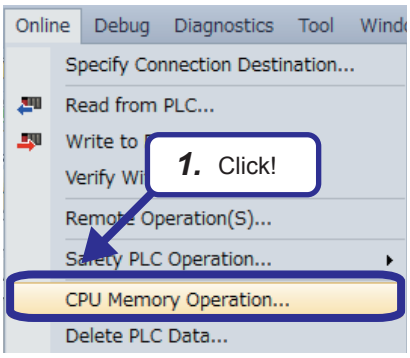
  The set connection route is displayed in an illustration.



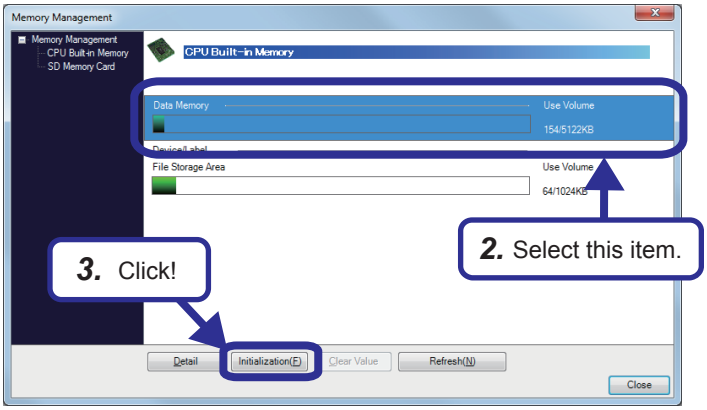
2.3.5 Initializing the CPU module

Initialize the RCPU.

Operating procedure

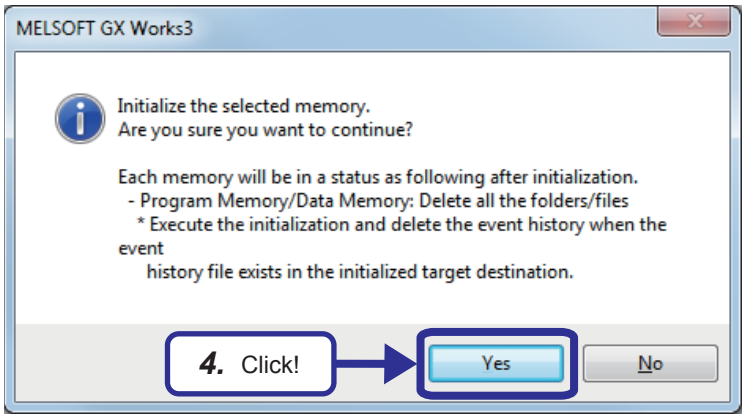


1. Click [Online] → [CPU Memory Operation] from the menu.



2. Select "Data Memory" in the "Memory Management" dialog box.

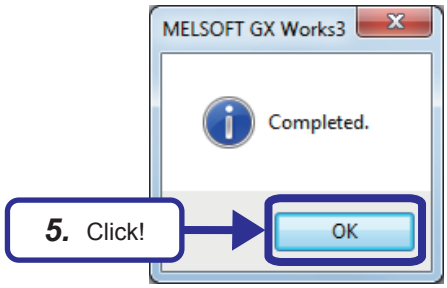
3. Click the [Initialization] button.



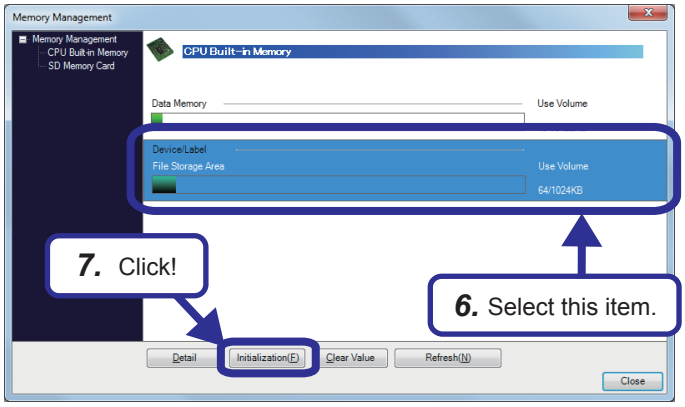
4. The confirmation dialog box appears. Click the [Yes] button.

(To the next page)

(From the previous page)

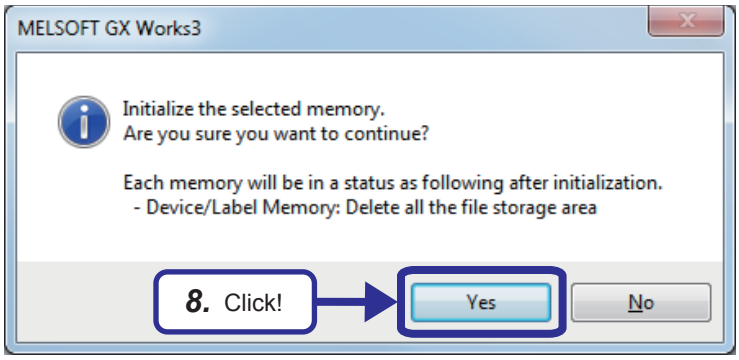


5. When the initialization is completed, the dialog box shown on the left appears. Click the [OK] button.



6. Select "Device/Label".

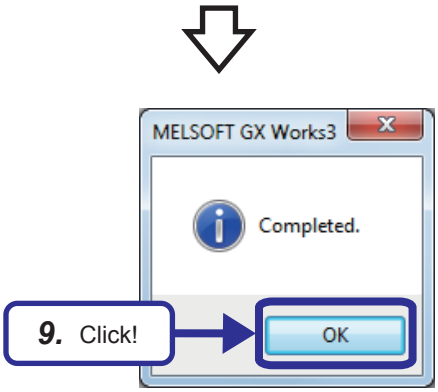
7. Click the [Initialization] button.



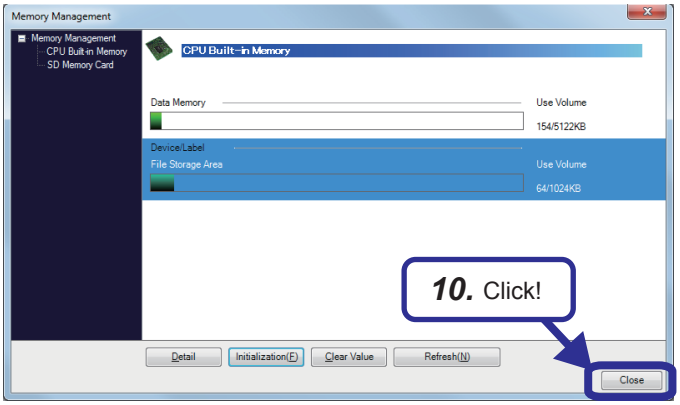
8. The confirmation dialog box appears. Click the [Yes] button.

(To the next page)

(From the previous page)



9. When the initialization is completed, the dialog box shown on the left appears. Click the [OK] button.

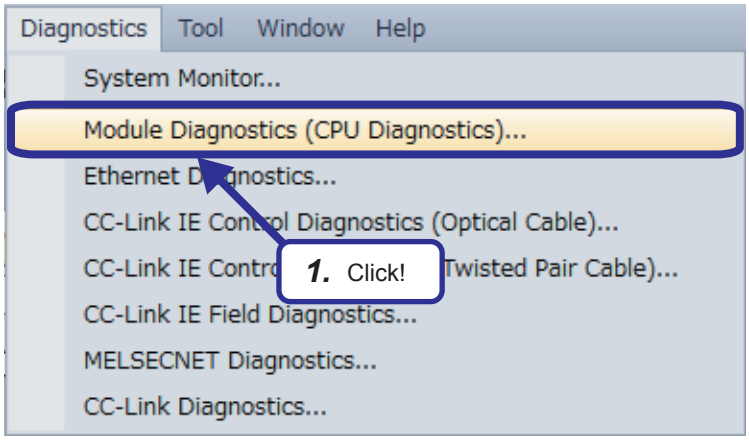


10. When the initialization processing is completed, click the [Close] button to close the dialog box.

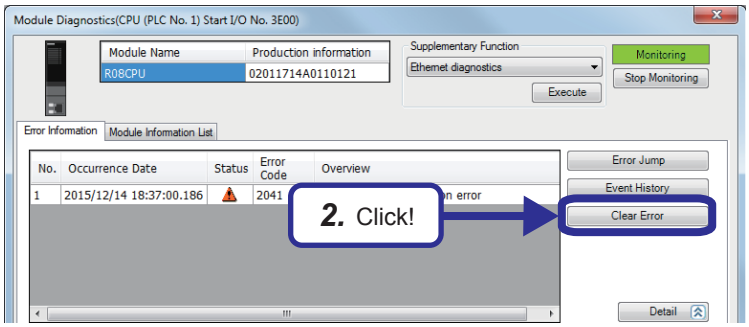
2.3.6 Clearing the error history of CPU module

Clear the error history data of the RCPU.

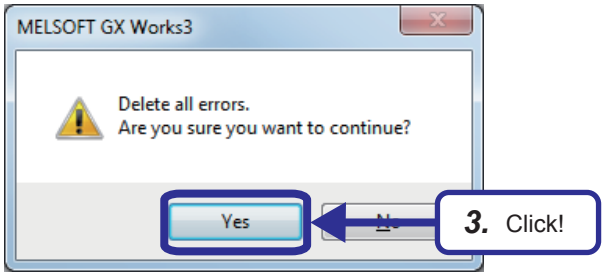
Operating procedure



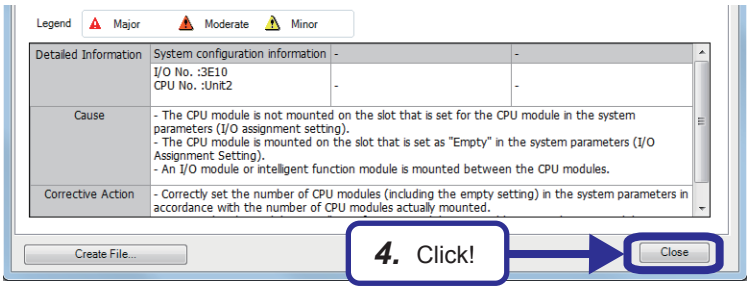
1. Click [Diagnostics] → [Module Diagnostics (CPU Diagnostics)] from the menu.



2. The dialog box shown on the left appears. Click the [Clear Error] button.



3. The confirmation dialog box appears. Click the [Yes] button.

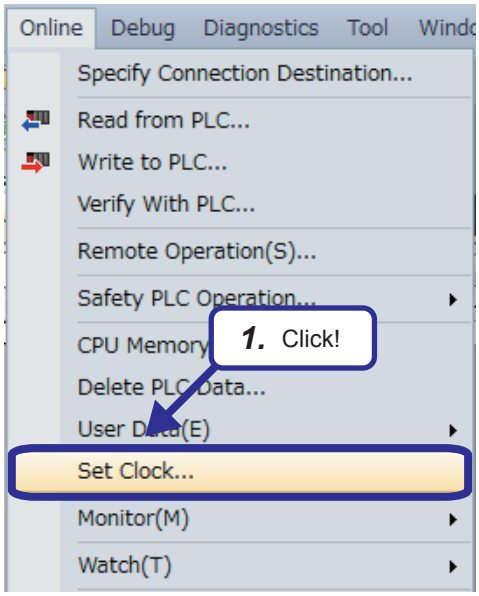


4. Click the [Close] button to close the dialog box.

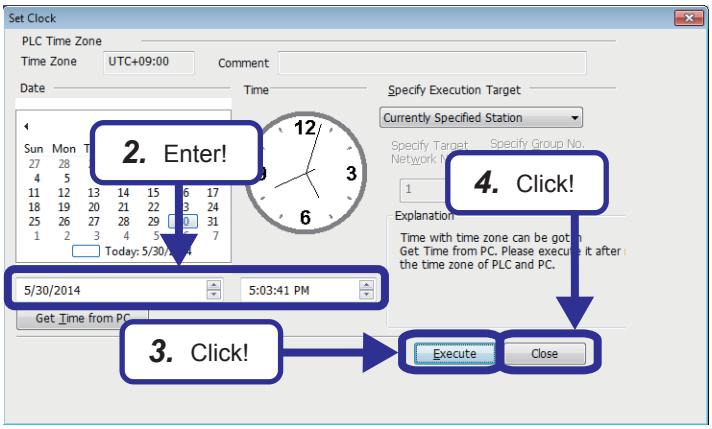
2.3.7 Setting the clock of the CPU module

The year, month, day, hour, minute, second, and day of week can be set to the clock element of the CPU module.
To use the clock function, use GX Works3 or a program.
Set the clock and read the setting with GX Works3.

Operating procedure



1. Click [Online] → [Set Clock] from the menu to display the "Set Clock" dialog box.



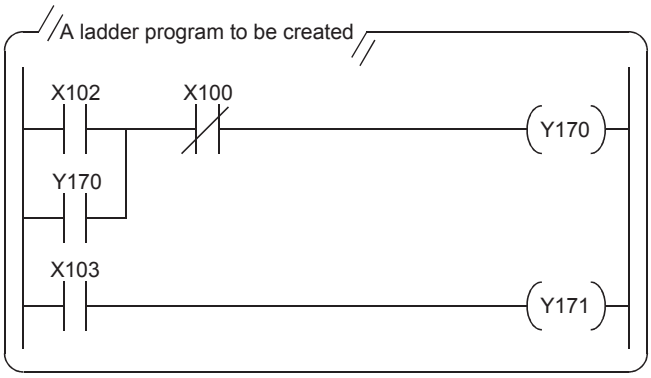
2. Set a year, month, day, hour, minute, second, and day of week on the "Set Clock" dialog box.

3. Click the [Execute] button.

4. Click the [Close] button.

2.4 Creating a Ladder Program

Operating procedure

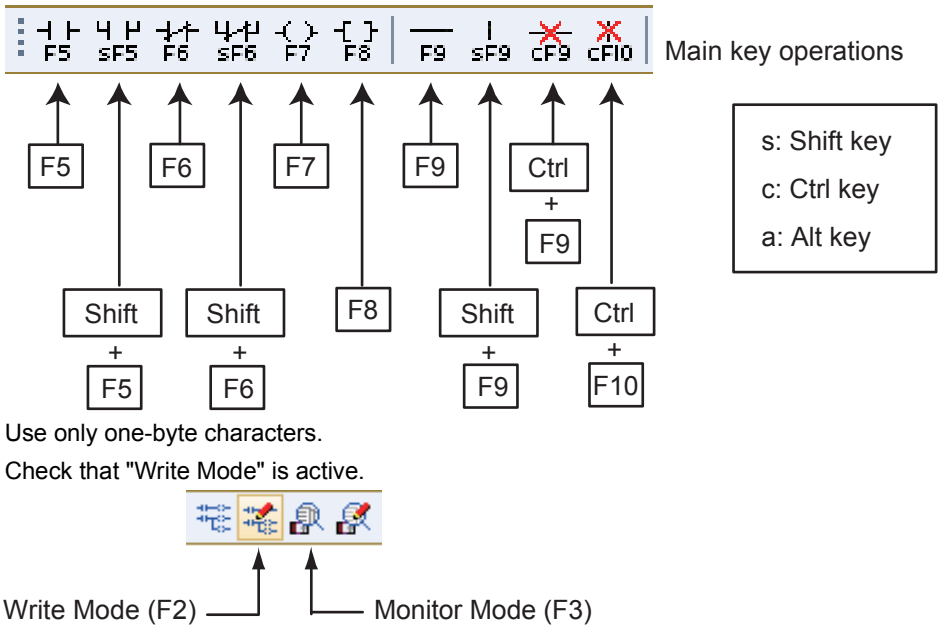


1. This section describes how to create a ladder program such as the one shown on the left.

- Use only one-byte characters.
  Two-byte characters cannot be used.

Point

The following figure shows the buttons on the toolbar. The character below each ladder symbol indicates each function key.



- Use only one-byte characters.
- Check that "Write Mode" is active.

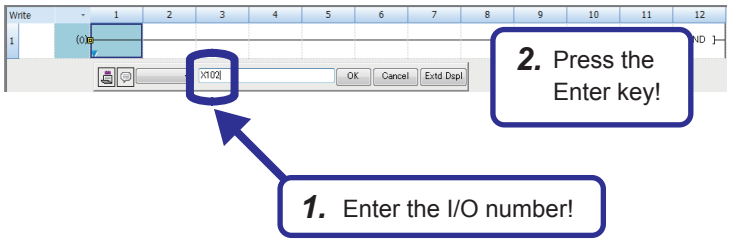
Point

How to input contacts and coils

Users can create ladders with the function keys and tool buttons. To input a contact or coil, specify a position where a contact or coil is to be input with the cursor and enter a device and label.
Users can switch a normally open contact and normally closed contact with the "/" key.
If an added ladder is in contact with the right rail or is an output device (Y, DY), the ladder is recognized as a coil. If not, it is recognized as a contact.

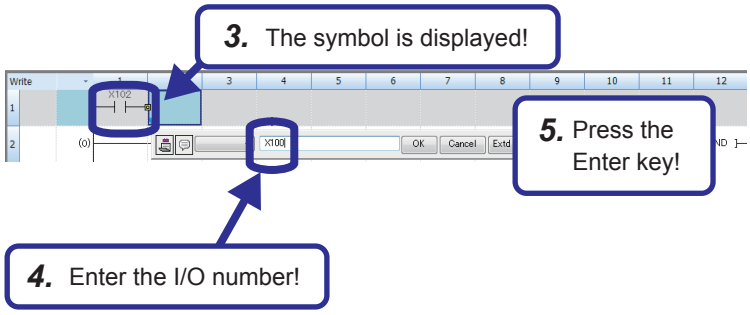
2.4.1 Creating a ladder program by entering devices and labels

Operating procedure



1. Move the cursor to the position where a ladder is added, and enter "X102". (When entering of the number starts, the ladder input window appears.)
To cancel an incorrect entry, press the Esc key.

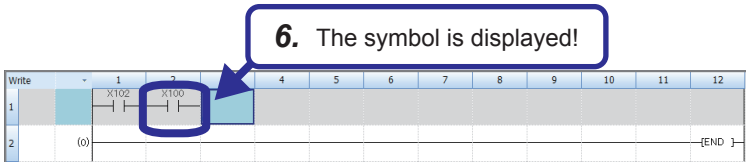
2. To confirm the entry, press the Enter key.
- Clicking the [OK] button also confirms the entry.
- Clicking the [Cancel] button also cancels the entry.



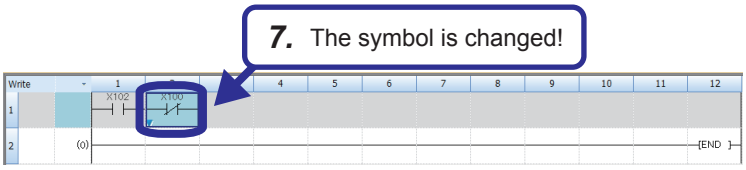
3. The added symbol (X102) is displayed.

4. Move the cursor to the next position and enter "X100".

5. Press the Enter key.



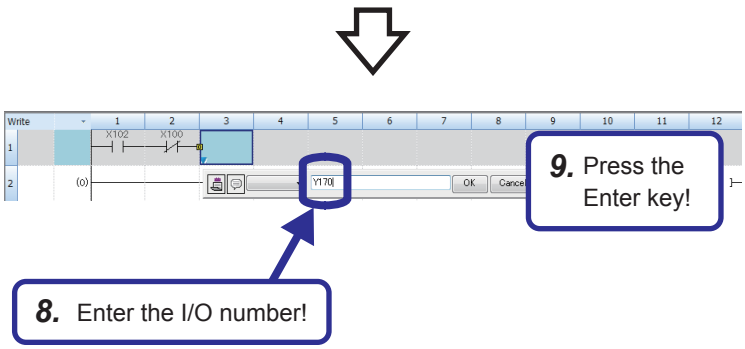
6. The added symbol (X100) is displayed.



7. Select the symbol and press the "/" key to switch the symbol with (X100).

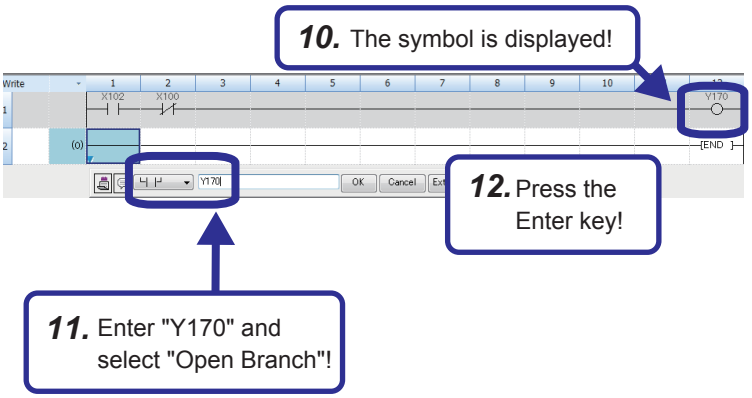
(To the next page)

(From the previous page)



8. Move the cursor to the next position and enter "Y170".

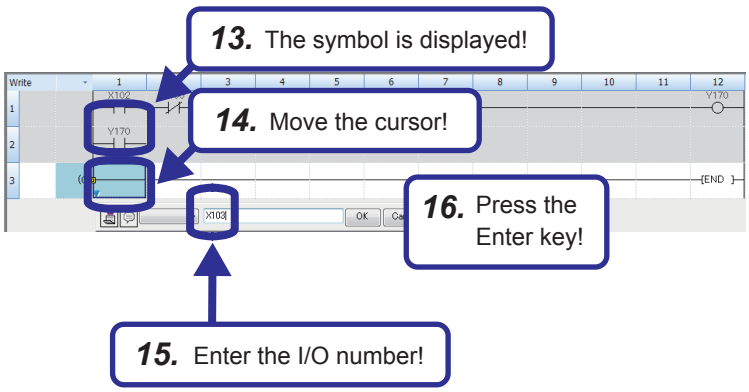
9. Press the Enter key.



10. The added symbol (—(Y170)—) is displayed.

11. Move the cursor to the next position, enter "Y170", and select "Open Branch".

12. Press the Enter key.



13. The added symbol (Y170 ─┤ ├─) is displayed.

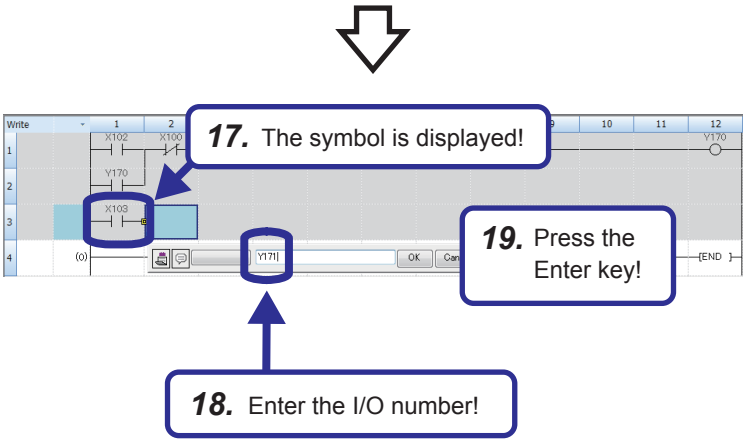
14. Move the cursor to the ladder under Y170 ─┤ ├─.

15. Enter "X103".

16. Press the Enter key.

(To the next page)

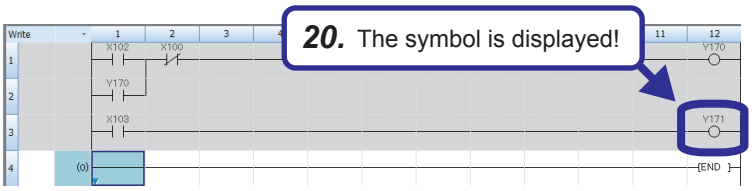
(From the previous page)



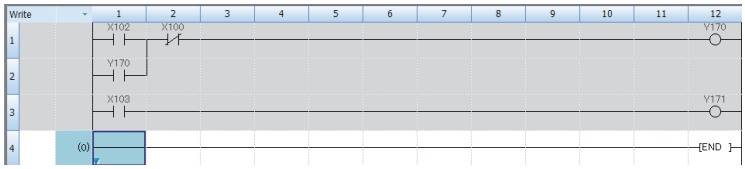
17. The added symbol (X103) is displayed.

18. Move the cursor to the next position and enter "Y171".

19. Press the Enter key.



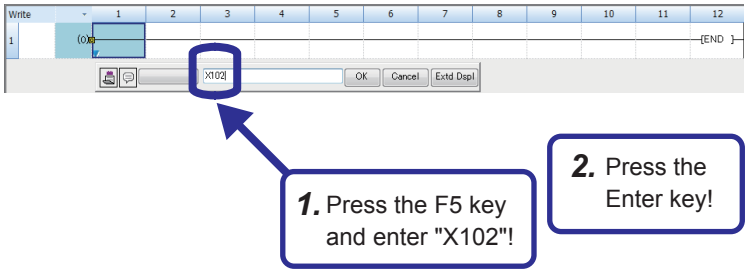
20. The added symbol (Y171) is displayed.



21. Creating a ladder program is completed.

2.4.2 Creating a ladder program with function keys

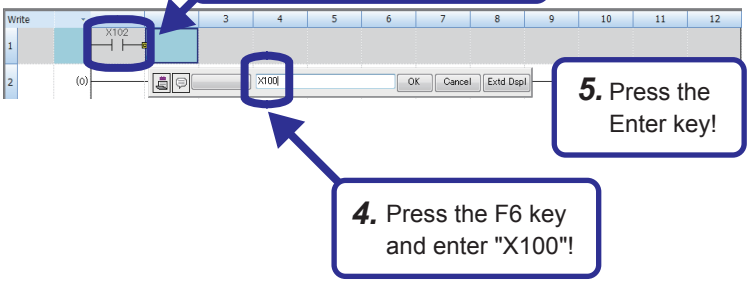
Operating procedure



1. Press the F5 key to open the ladder input window, and enter "X102".
To cancel an incorrect entry, press the Esc key.

2. To confirm the entry, press the Enter key.

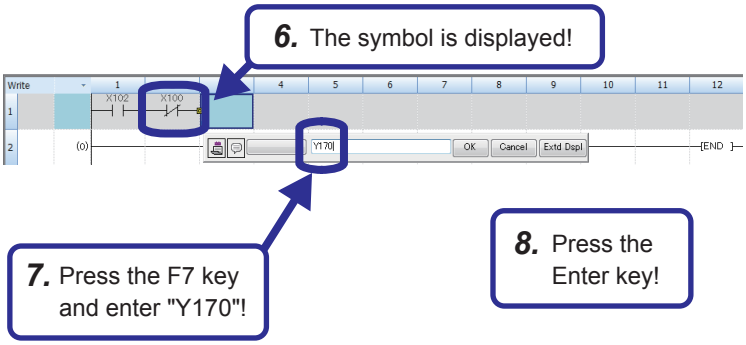
- Clicking the [OK] button also confirms the entry.
- Clicking the [Cancel] button also cancels the entry.



3. The added symbol (X102) is displayed.

4. Press the F6 key and enter "X100".

5. Press the Enter key after entering the device number.



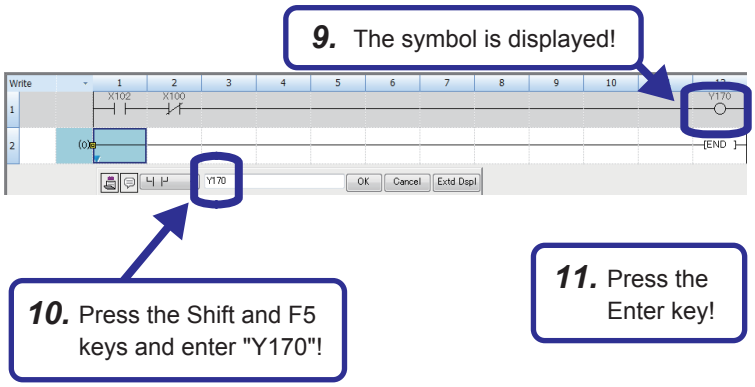
6. The added symbol (X100) is displayed.

7. Press the F7 key and enter "Y170".

8. Press the Enter key after entering the device number.

(To the next page)

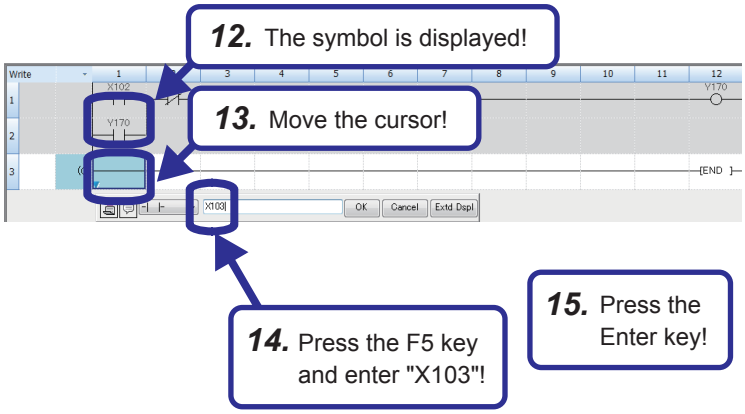
(From the previous page)



9. The added symbol (—(Y170)—) is displayed.

10. Press the [Shift] key and the [F5] key and enter "Y170".

11. Press the [Enter] key after entering the device number.

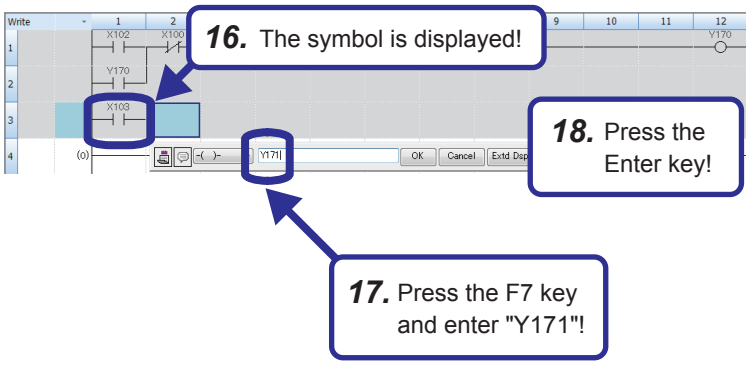


12. The added symbol (Y170) is displayed.

13. Move the cursor to the ladder under Y170.

14. Press the [F5] key and enter "X103".

15. Press the [Enter] key after entering the device number.



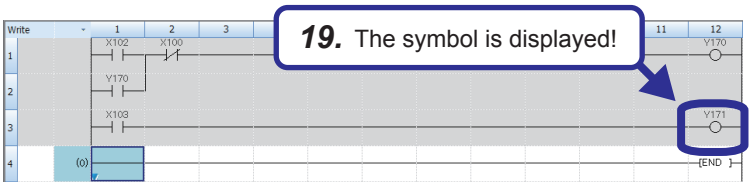
16. The added symbol (X103) is displayed.

17. Press the [F7] key and enter "Y171".

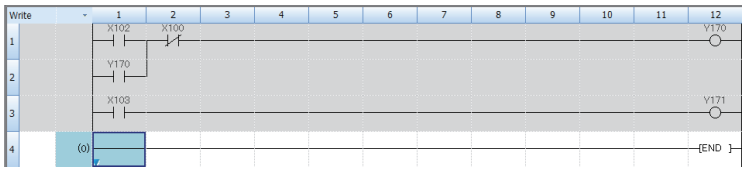
18. Press the [Enter] key after entering the device number.

(To the next page)

(From the previous page)



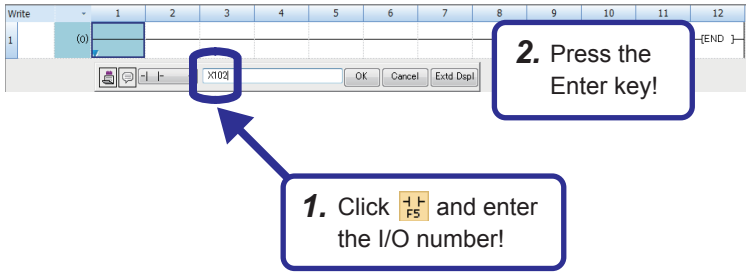
19. The added symbol (—(Y171)—) is displayed.



20. Creating a ladder program is completed.

2.4.3 Creating a ladder program with tool buttons

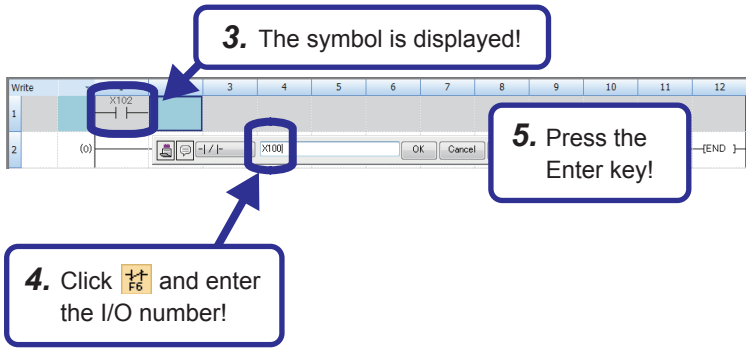
Operating procedure



1. Click on the toolbar to open the ladder input window, and enter "X102".
To cancel an incorrect entry, press the Esc key.

2. To confirm the entry, press the Enter key.

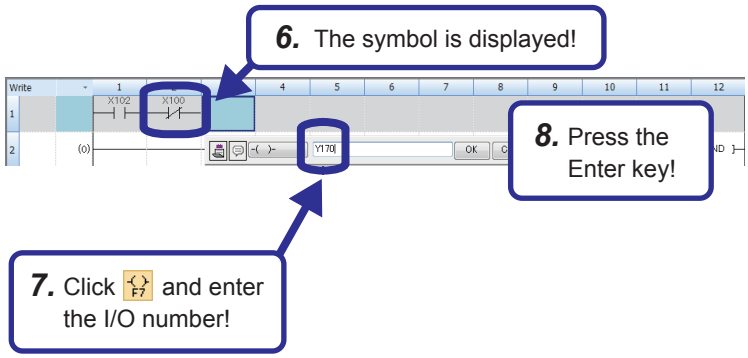
- Clicking the [OK] button also confirms the entry.
- Clicking the [Cancel] button also cancels the entry.



3. The added symbol (X102) is displayed.

4. Click on the toolbar and enter "X100".

5. Press the Enter key.



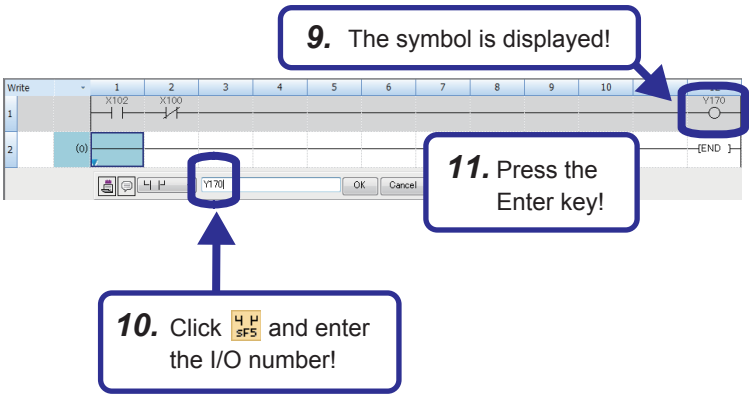
6. The added symbol (X100) is displayed.

7. Click on the toolbar and enter "Y170".

8. Press the Enter key.

(To the next page)

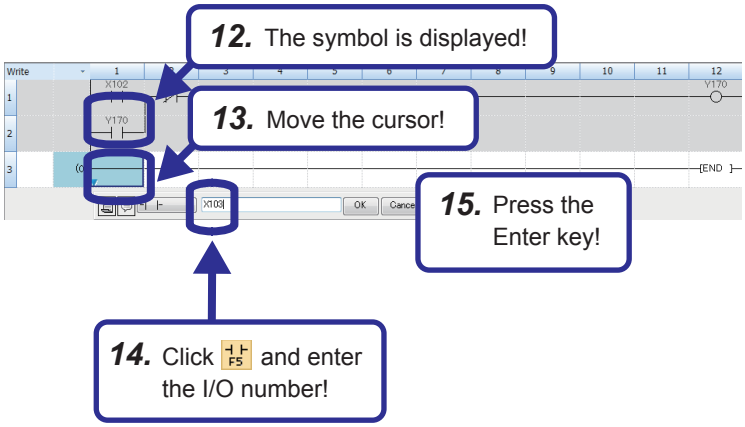
(From the previous page)



9. The added symbol (—(Y170)—) is displayed.

10. Click on the toolbar and enter "Y170".

11. Press the Enter key.

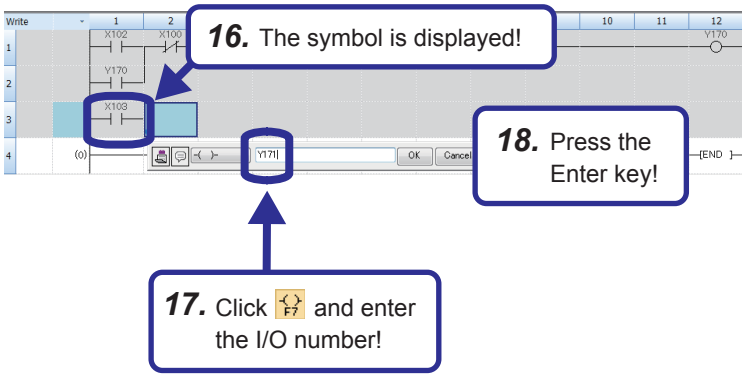


12. The added symbol (Y170) is displayed.

13. Move the cursor to the ladder under Y170.

14. Click on the toolbar and enter "X103".

15. Press the Enter key.



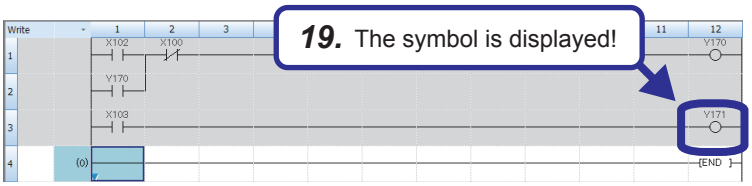
16. The added symbol (X103) is displayed.

17. Click on the toolbar and enter "Y171".

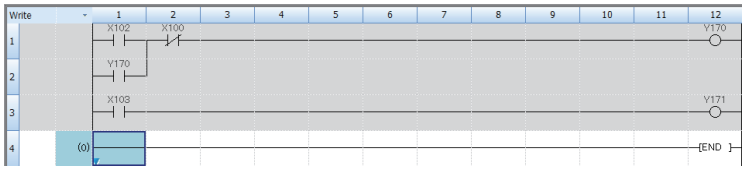
18. Press the Enter key.

(To the next page)

(From the previous page)



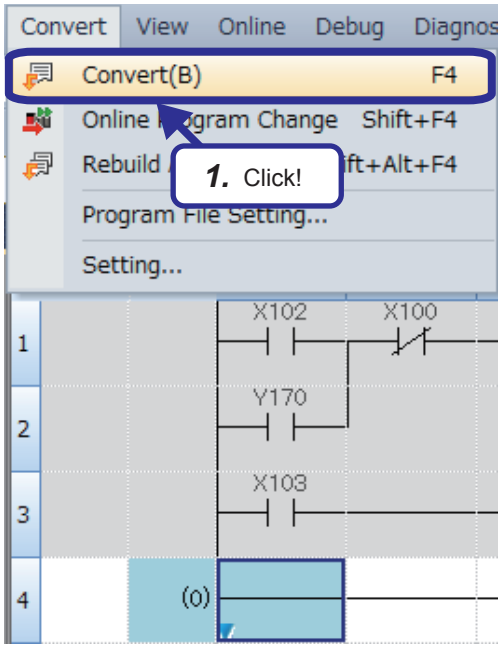
19. The added symbol (—(Y171)—) is displayed.



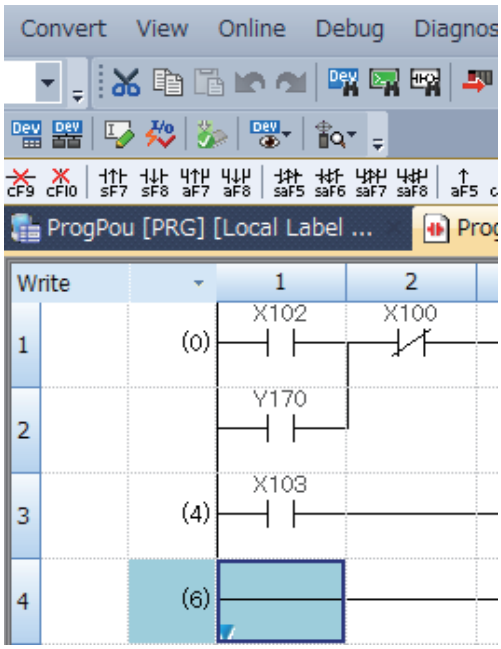
20. Creating a ladder program is completed.

2.5 Converting a Created Ladder Program

Operating procedure



1. Click [Convert] → [Convert] (F4) from the menu.



2. The ladder program is converted. When the conversion processing is completed and the input ladder blocks are determined, the color of those ladder blocks changes from gray to white.

When an error has occurred during conversion, the cursor is moved to the position where the error has occurred. Check the ladder.

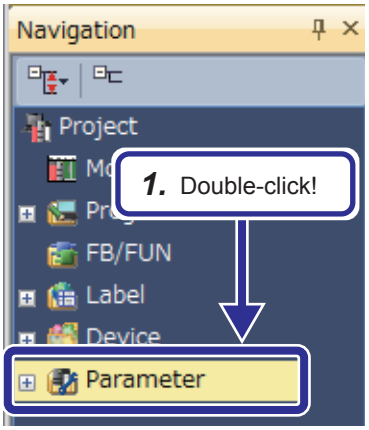
2.6 Reading/Writing Data from/to the Programmable Controller CPU

I/O assignment with the parameter setting

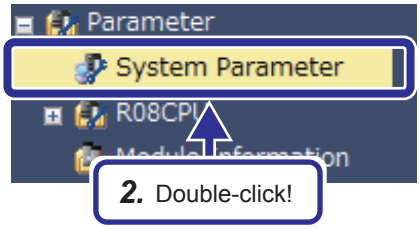
This section describes an example of the I/O assignment setting of parameters.
In this practice, this setting is not configured.

Point

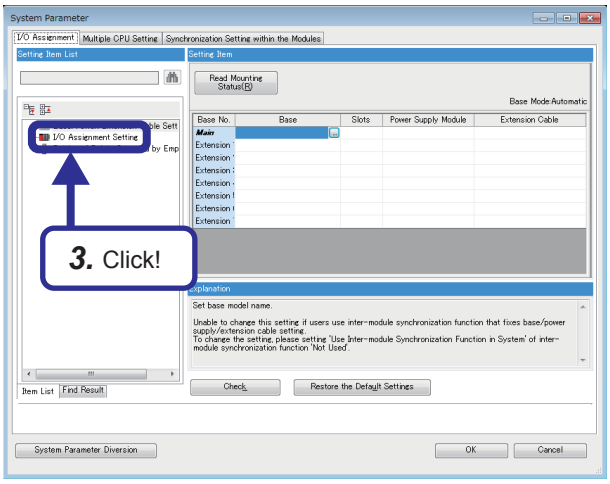
This section describes the I/O assignment setting of parameters.



1. Double-click "Parameter" in the "Project" view.



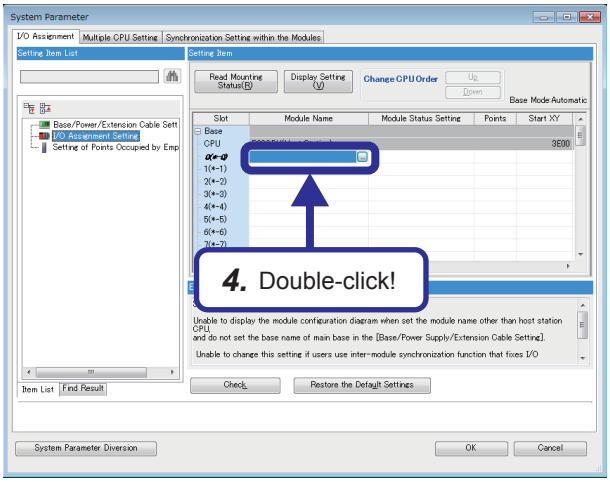
2. Double-click "System Parameter".



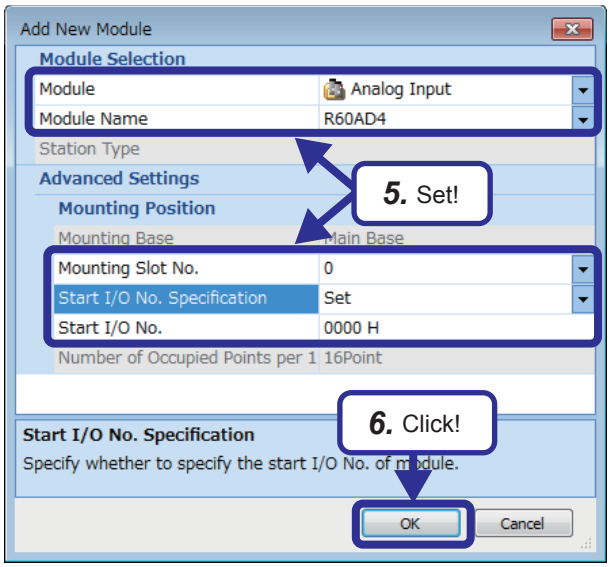
3. The "System Parameter" dialog box appears. Click "I/O Assignment Setting" in "Setting Item List".

(To the next page)

(From the previous page)



4. Double-click a row in "Module Name".



5. Set the following items on the "Add New Module" dialog box.

- "Module"
- "Module Name"
- "Mounting Slot No."
- "Start I/O No. Specification"
- "Start I/O No."

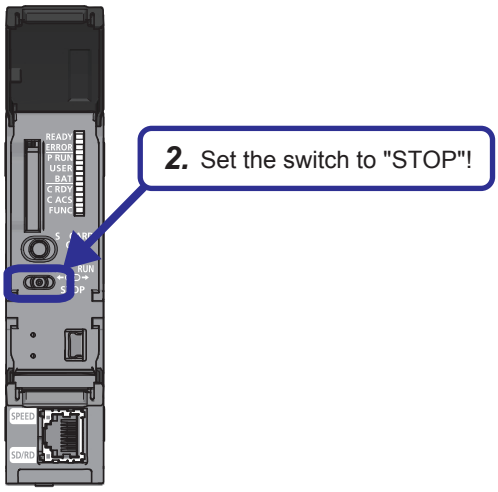
* The left figure is a setting example.

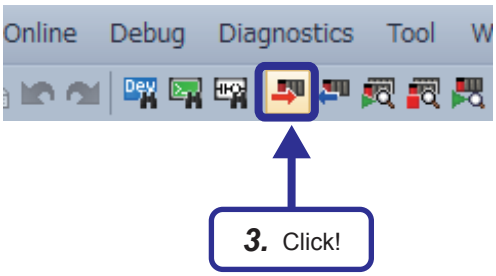
6. Click the [OK] button.

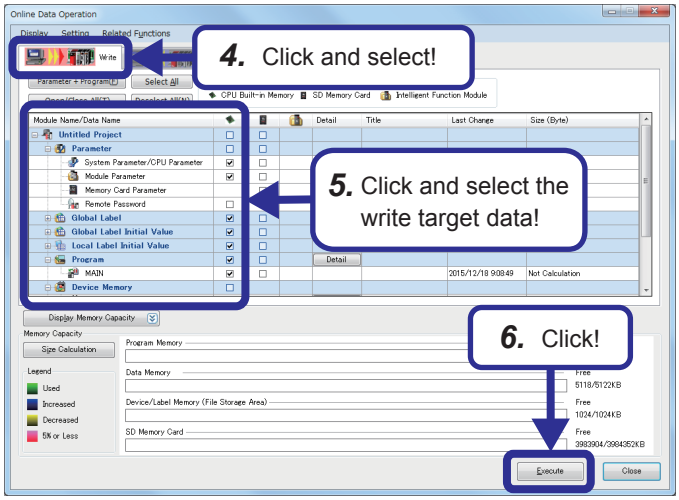
2.6.1 Writing data to the CPU module

Before writing data to the CPU module, initialize the memory.
For details, refer to Section 2.3.5.

Operating procedure







(To the next page)

1. Prior to this operation, create a ladder program (sequence program).

2. Set the RUN/STOP/RESET switch to the STOP position.

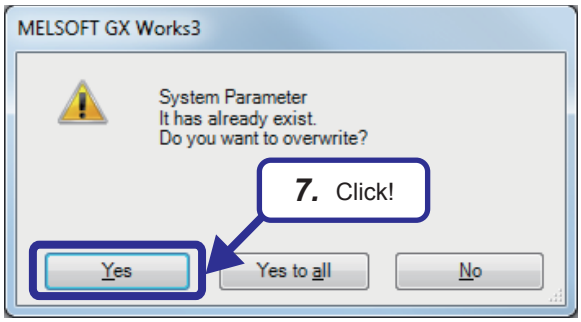
3. Click on the toolbar, or click [Online] → [Write to PLC] from the menu.

4. Select the [Write] tab in the "Online Data Operation" window.

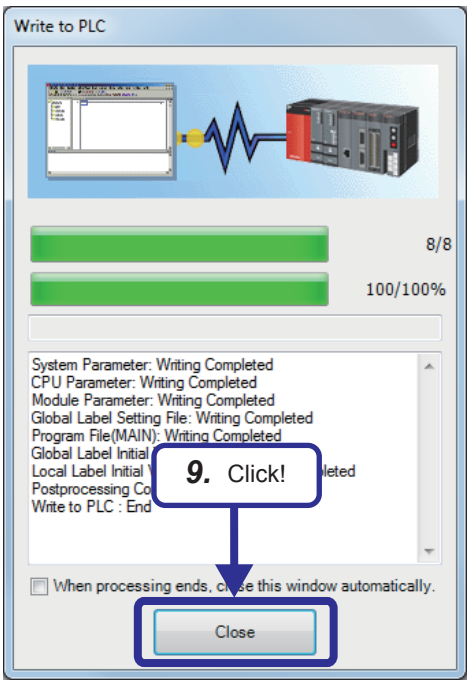
5. Select files and parameters to be written as shown left.

6. After selecting files and parameters, click the [Execute] button.

(From the previous page)

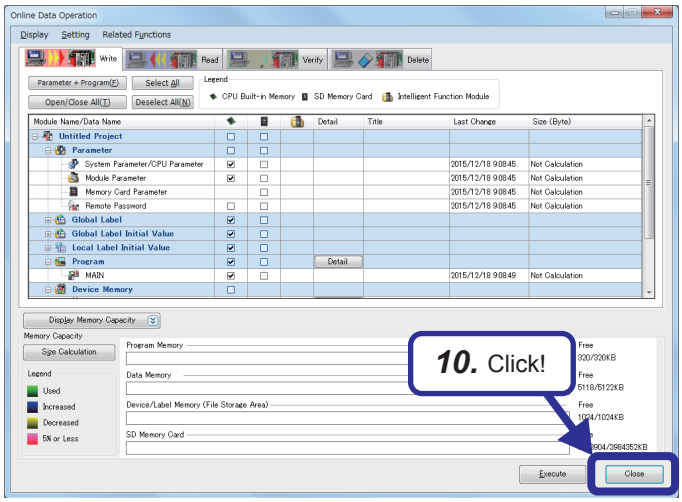


7. If parameters or files have already existed in the CPU module, the confirmation window for overwriting the data appears. Click the [Yes] button.



8. The dialog box indicating that writing is in progress appears.

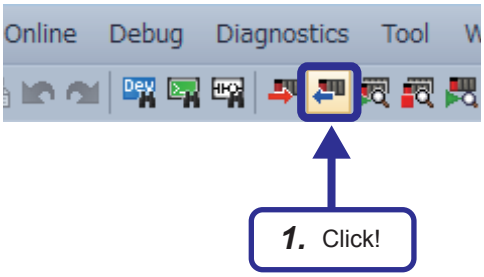
9. When writing the data is completed, the message "Completed" is displayed. Click the [Close] button.



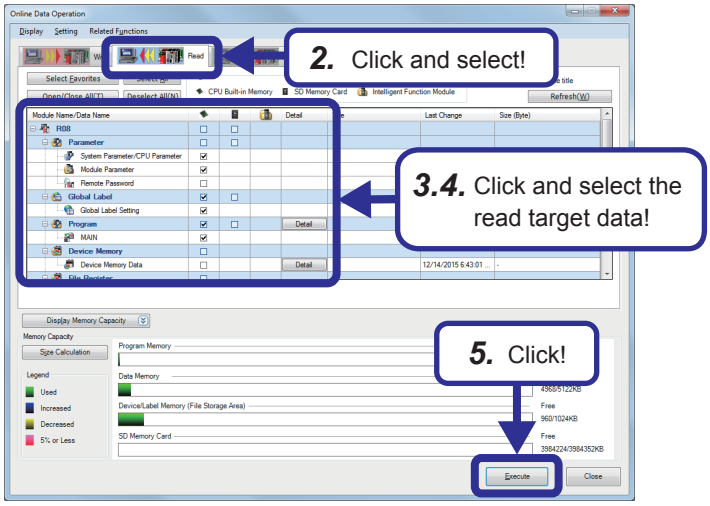
10. Click the [Close] button to close the dialog box.

2.6.2 Reading data from the CPU module

Operating procedure



1. Click on the toolbar, or click [Online] → [Read from PLC] from the menu.

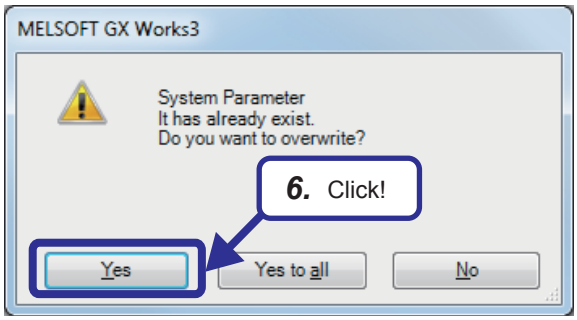


2. Select the [Read] tab in the "Online Data Operation" window.

3. Select files and parameters to be read and the destination where read data is to be stored.

4. Click the [Detail] button to set details such as a read range.

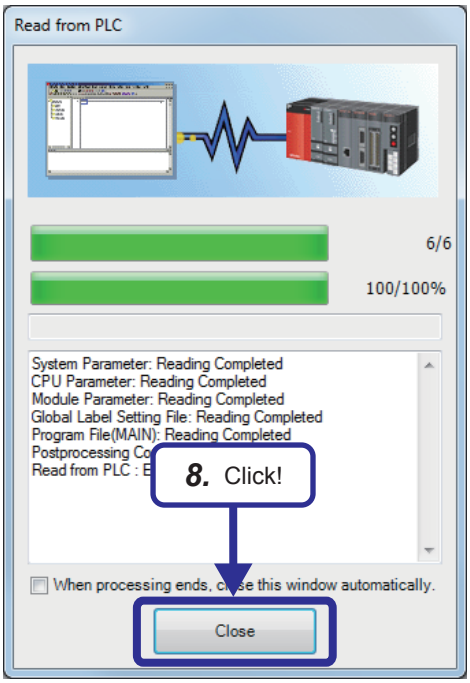
5. After selecting files and parameters, click the [Execute] button.



6. If parameters or files have already existed, the confirmation window for overwriting the data appears. Click the [Yes] button.

(To the next page)

(From the previous page)

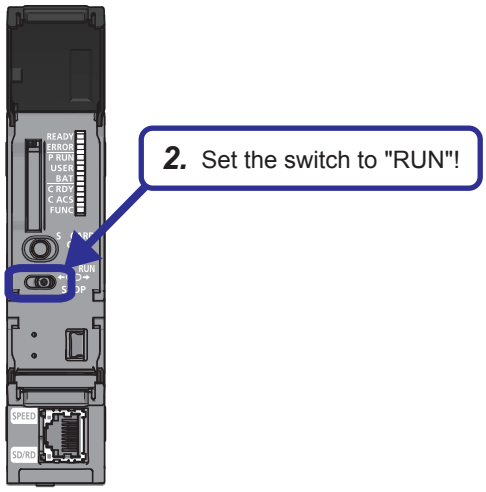


7. The dialog box indicating that reading is in progress appears.

8. When reading the data is completed, the message "Completed" is displayed. Click the [Close] button.

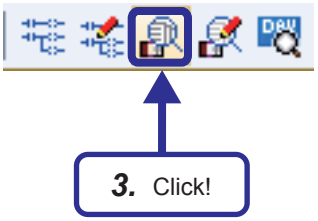
2.7 Monitoring the Ladder

Operating procedure

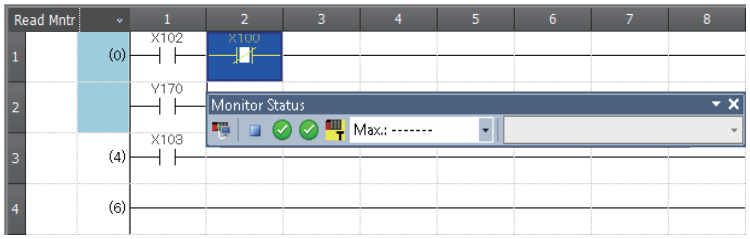


1. Prior to this operation, write a ladder program (sequence program) to the programmable controller CPU.

2. Set the RUN/STOP/RESET switch of the CPU module to the "RESET" position once (for about one second), return the switch to the "STOP" position, and then set the switch to the "RUN" position again.



3. Click on the toolbar, or click [Online] → [Monitor] → [Monitor Mode] from the menu.



4. To stop monitoring, set the mode other than the monitor mode.

Operation practice

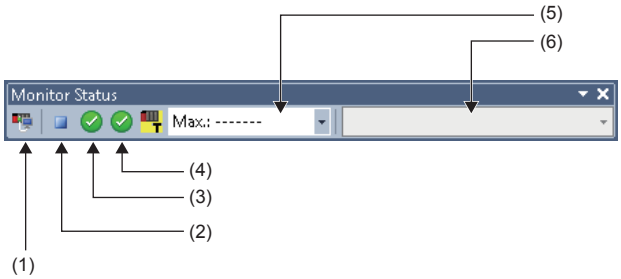
❶ Check that turning on the switch X102 turns on the LED indicator Y170, and that the LED indicator Y170 remains on even after the switch X102 turns off.

❷ Check that turning on the switch X100 turns off the LED indicator Y170, and that the LED indicator Y170 remains off even after the switch X100 turns off.

❸ Turning on the switch X103 turns on the LED indicator Y171.

Monitoring on the monitor status bar

In the monitor mode, the following "Monitor Status" dialog box appears regardless of whether the operation status is monitoring in progress or not.



(1) Connection status

The connection status with the CPU module is displayed.

(2) CPU module operating status

The operating status of the CPU module in accordance with the RUN/STOP/RESET switch of the CPU module or the remote operation by the engineering tool is displayed.

(3) ERROR LED status

The ERROR LED status of the CPU module is displayed.

Clicking the icon opens the "Module Diagnostics" window.

(☞ Page 2 - 40 Diagnosing the Programmable Controller CPU)

(4) USER LED status

The USER LED status of the CPU module is displayed.

Clicking the icon opens the "Module Diagnostics" window.

(☞ Page 2 - 40 Diagnosing the Programmable Controller CPU)

(5) Scan time details

The scan time details are displayed. Select the value to be displayed from the drop-down list (current value, maximum value, or minimum value).

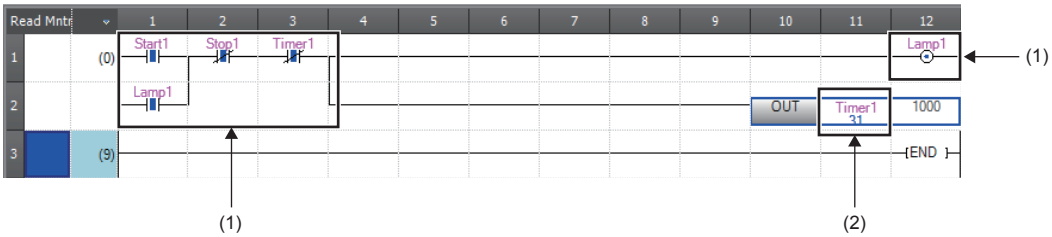
(6) Monitor target selection

Specify the monitor target FB instance when monitoring a FB program.

*1 For the module diagnostics, refer to Section 2.8.

Monitoring on the ladder editor

The following figure shows how the ladder status is displayed on the ladder editor.



(1) The on/off states of contacts and coils are displayed.

(2) The current value of the word/double word type data is displayed.

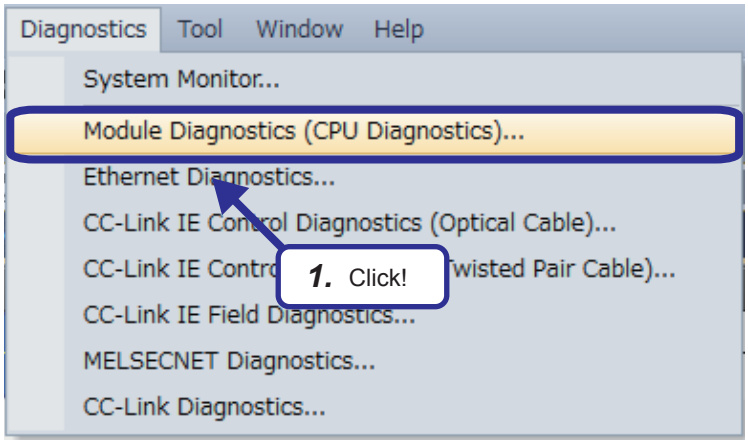
■On/off state display

The on/off states are displayed on the editor as follows:

*1 Only comparison instructions that are equivalent to contacts and the instructions that are equivalent to coils are supported.
Comparison instructions equivalent to contacts: 16-bit binary data comparison, 32-bit binary data comparison, floating-point data comparison, 64-bit floating-point data comparison
Instructions equivalent to coils: SET, RST, PLS, PLF, SFT, SFTP, MC, FF, DELTA, DELTAP

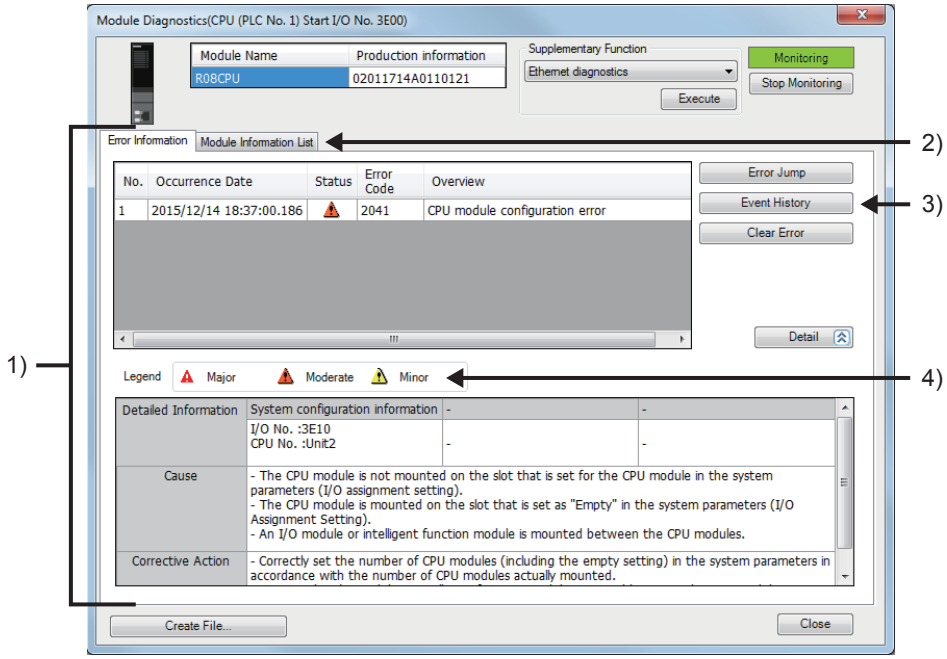
2.8 Diagnosing the Programmable Controller CPU

Operating procedure

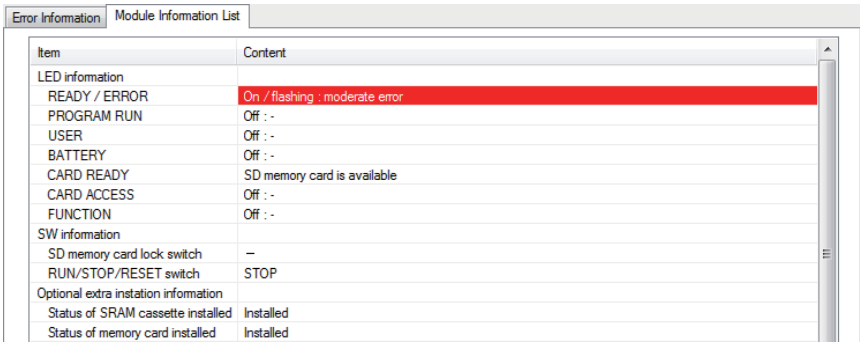


1. Click [Diagnostics] → [Module Diagnostics (CPU Diagnostics)] from the menu.

2. The "Module Diagnostics" window appears.



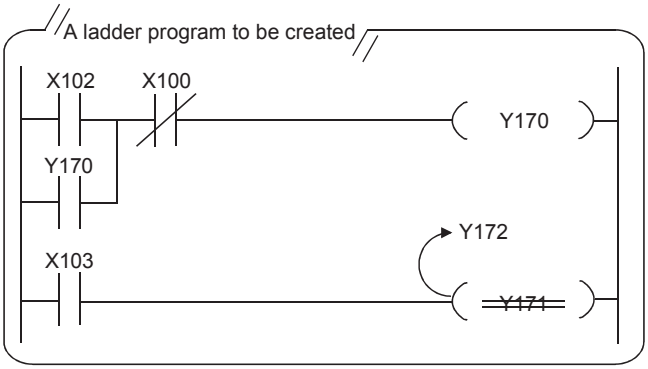
For Q series modules, "-" is displayed in the rows of "Occurrence Date", "Status", and "Overview".

	Item	Description
1)	Error Information	Select this tab to display the error information of the current programmable controller CPU.
2)	Module Information List	Select this tab to display the status information of the programmable controller CPU.  Error Information / Module Information List Item / Content LED information READY / ERROR: On / flashing : moderate error PROGRAM RUN: Off : - USER: Off : - BATTERY: Off : - CARD READY: SD memory card is available CARD ACCESS: Off : - FUNCTION: Off : - SW information SD memory card lock switch: – RUN/STOP/RESET switch: STOP Optional extra instation information Status of SRAM cassette installed: Installed Status of memory card installed: Installed
3)	Event History	Click [Event History] to display the error information, operation history, and system information history of the module.
4)	Legend	Displays the example of icons displayed on the window.

2.9 Editing a Ladder Program

2.9.1 Modifying a part of a ladder program

Operating procedure



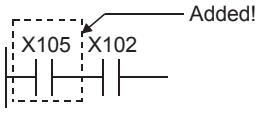
This section describes how to modify a part of the ladder program shown on the left. (OUT Y171 → OUT Y172)

- Use only one-byte characters.
  Two-byte characters cannot be used.

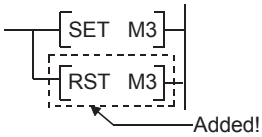
1. Check that "Overwrite" is displayed at the bottom right of the window.

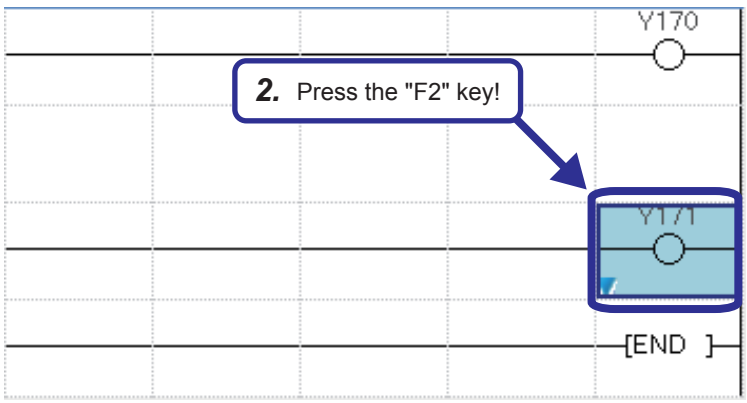
When "Insert" is displayed, press the Insert key to switch to "Overwrite".
When "Insert" is displayed, contacts or coils are added.

<When changing X102 with X105 is attempted>



<When changing SET with RST is attempted>

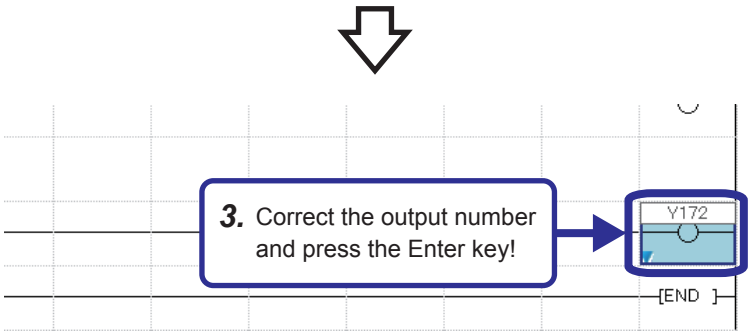




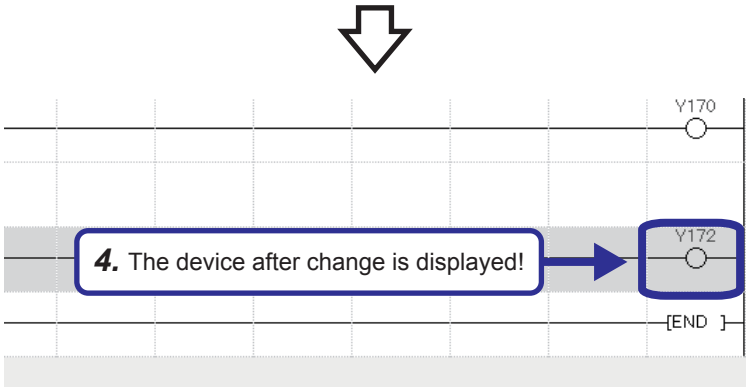
2. Click the position to be modified, and press the "F2" key.

(To the next page)

(From the previous page)



3. The device can be modified. Modify the device to "Y172", and press the [Enter] key.



4. The ladder program after the modification is displayed.
To change only the device number, click the F2 key.

5. Click [Convert] → [Convert] ([F4]) from the menu to convert the ladder program after the modification.

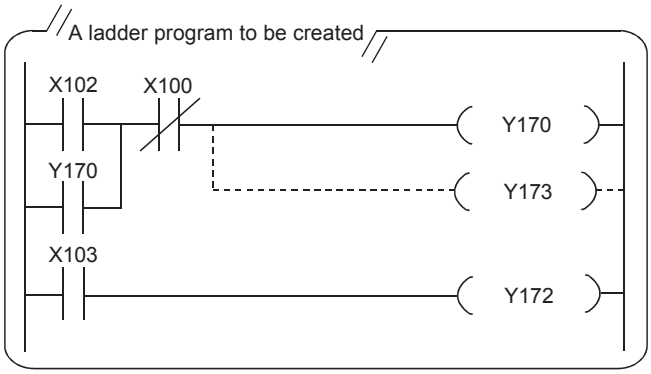
Point

How to input contacts and coils

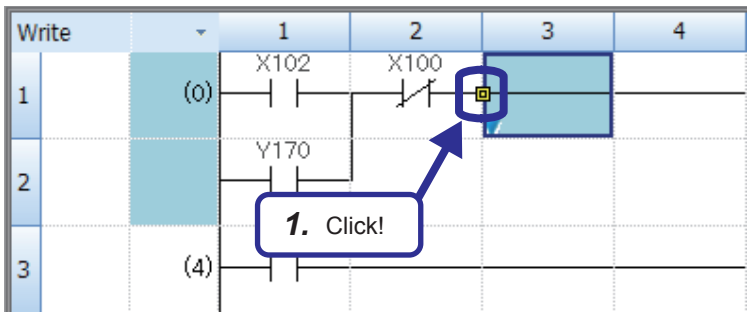
To input a contact or coil, specify a position where a contact or coil is to be input with the cursor and enter a device and label.
If an added ladder is in contact with the right rail or is an output device (Y, DY), the ladder is recognized as a coil. If not, it is recognized as a contact.

2.9.2 Drawing a line

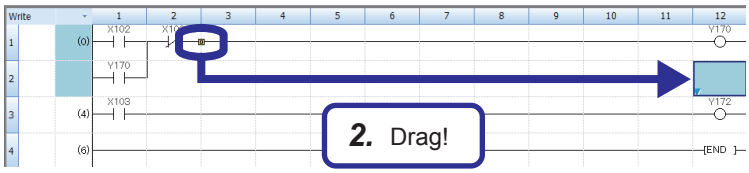
Operating procedure



This section describes how to draw a line in the ladder program shown on the left.

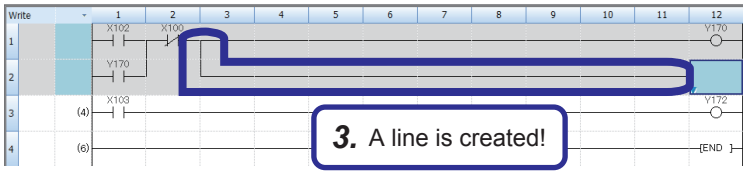


1. Move the mouse pointer close to the exiting line, and click the displayed icon.



2. Drag the icon from the position to the end position.

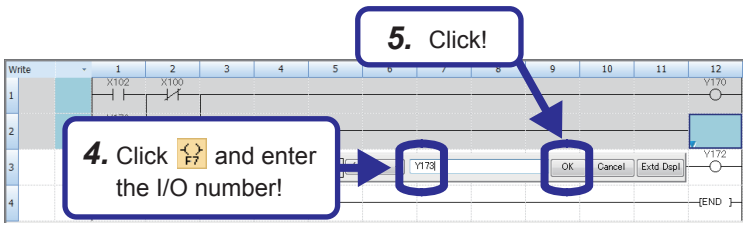
A vertical line is drawn on the left side of the cursor.



3. Release the left button of the mouse to create a line.

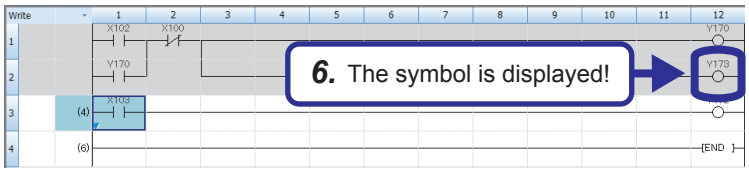
(To the next page)

(From the previous page)



4. Click on the toolbar and enter "Y173".

5. Click the [OK] button.



6. The added symbol (—(Y173)—) is displayed.

7. Click [Convert] → [Convert] (F4) from the menu to convert the ladder program.

Point

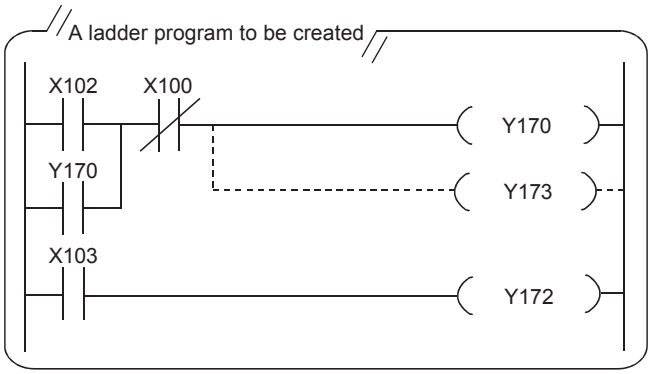
Adding or deleting a line with the key operation

In GX Works3, lines can be added or deleted with the Ctrl key + ←, →, ↑, or ↓.

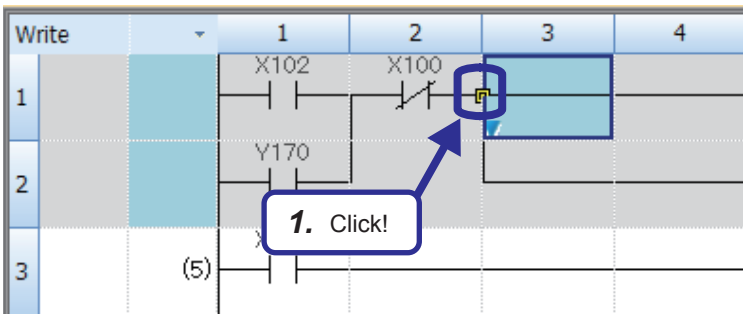
Users can draw a horizontal line from the cursor position to the position of the next contact, coil, or line by pressing Ctrl + Shift + ← or →.

2.9.3 Deleting a line

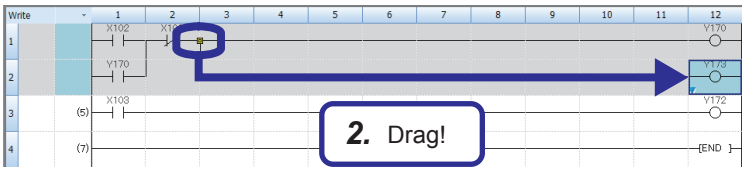
Operating procedure



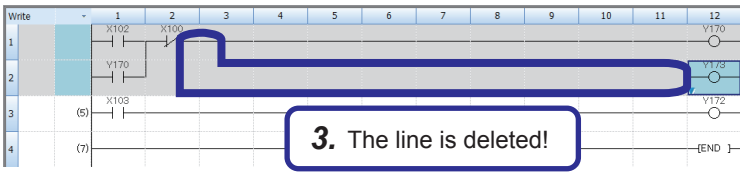
This section describes how to delete a line in the ladder program shown on the left.



1. Move the mouse pointer close to the exiting line, and click the displayed icon.

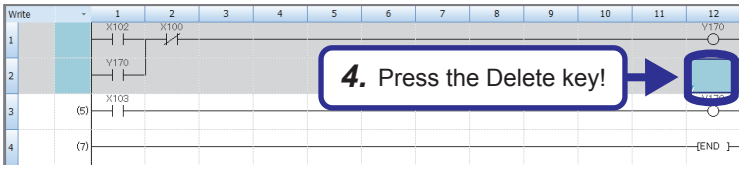


2. Drag the icon along the line to be deleted.



3. Release the left button of the mouse to delete the line.

The line connected to "END" cannot be deleted.



4. Press the Delete key to delete (–(Y173)–).

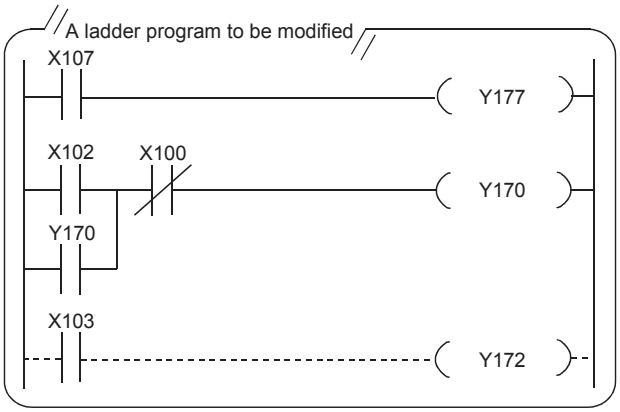
5. Click [Convert] → [Convert] (F4) from the menu to convert the ladder program.

Point

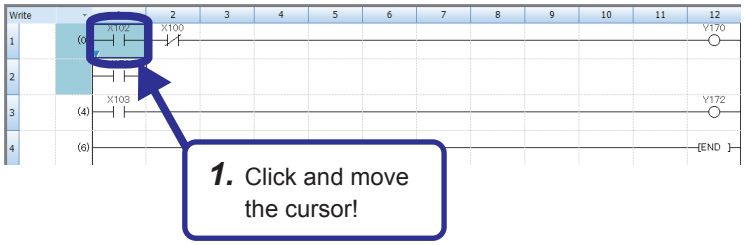
Adding or deleting a line with the key operation
In GX Works3, lines can be added or deleted with the Ctrl key + ←, →, ↑, or ↓.
Users can draw a horizontal line from the cursor position to the position of the next contact, coil, or line by pressing Ctrl + Shift + ← or →.

2.9.4 Inserting a row

Operating procedure

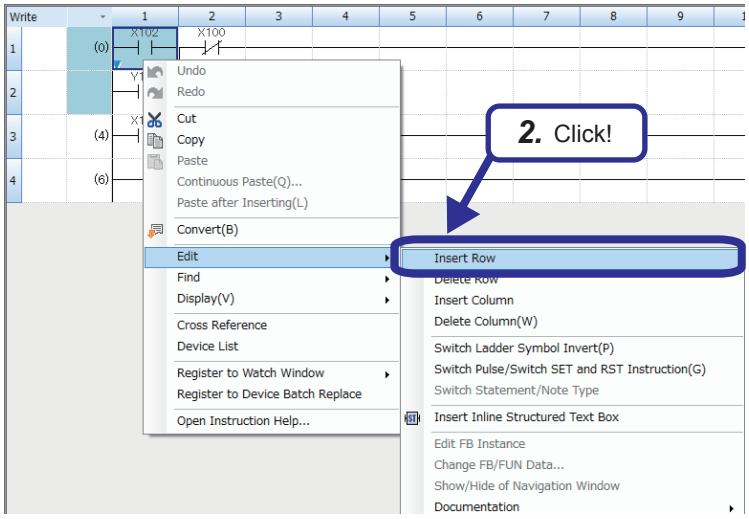


This section describes how to insert a row in the ladder program shown on the left.



1. Click and move the cursor on the row (desired position on the row) where a new row is inserted above.

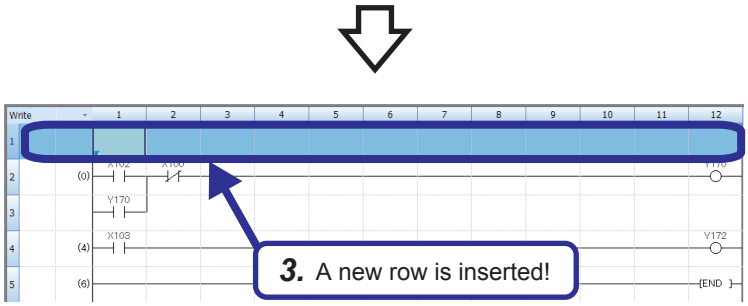
A new row is inserted above the row.



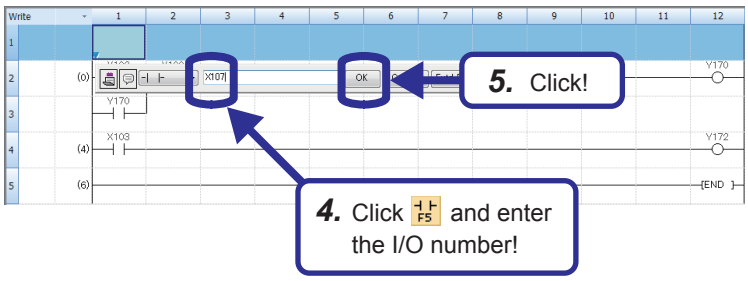
2. Right-click on the ladder editor, and click [Edit] → [Insert Row] (Shift + Insert) from the menu.

(To the next page)

(From the previous page)

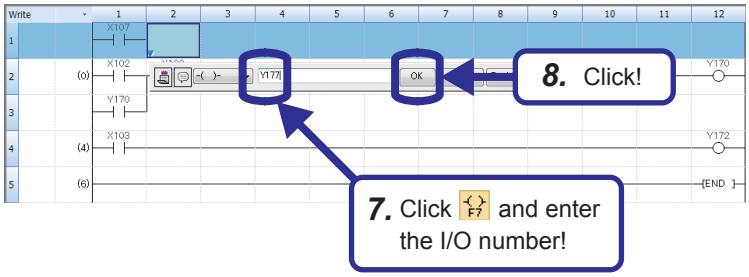


3. A new row is inserted above the row.



4. Click on the toolbar to open the ladder input window, and enter "X107".

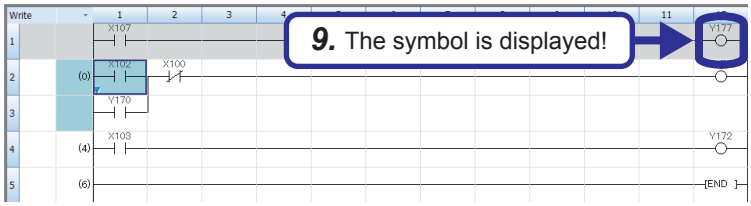
5. Click the [OK] button to confirm the entry.



6. The added symbol (X107) is displayed.

7. Click on the toolbar and enter "Y177".

8. Click the [OK] button.

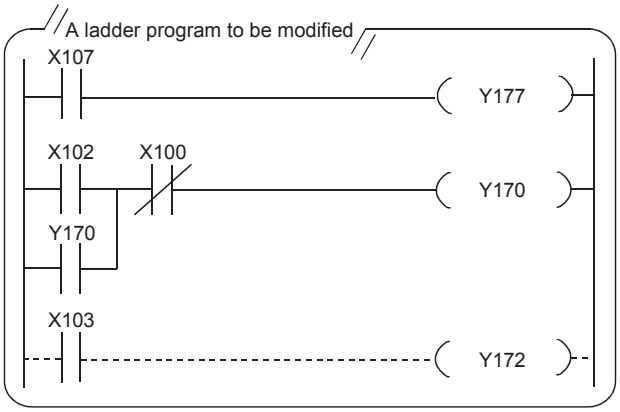


9. The added symbol (Y177) is displayed.

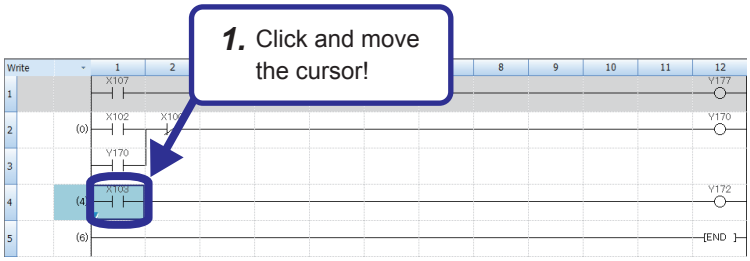
10. Click [Convert] → [Convert] (F4) from the menu to convert the ladder program.

2.9.5 Deleting a row

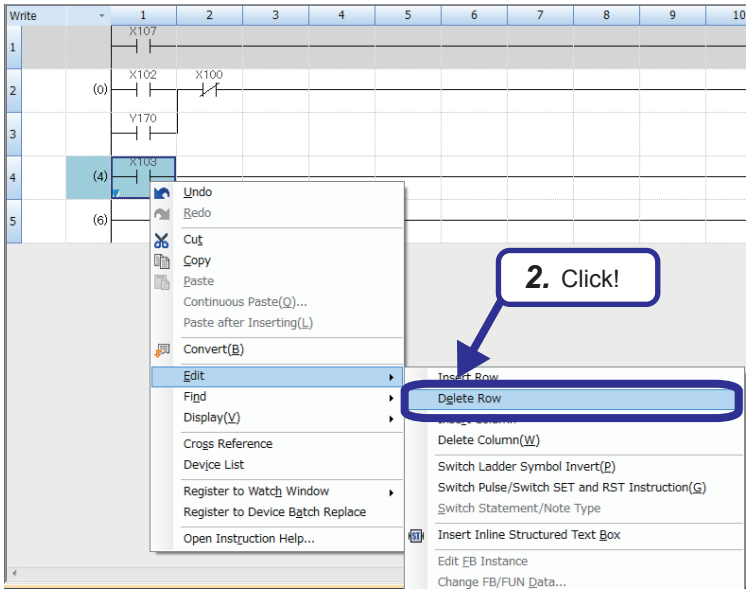
Operating procedure



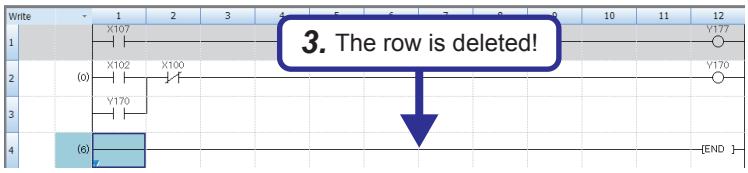
This section describes how to delete a row in the ladder program shown on the left.



1. Click and move the cursor on the row (desired position on the row) to be deleted.



2. Right-click on the ladder editor, and click [Edit] → [Delete Row] (Shift + Delete) from the right-click menu.

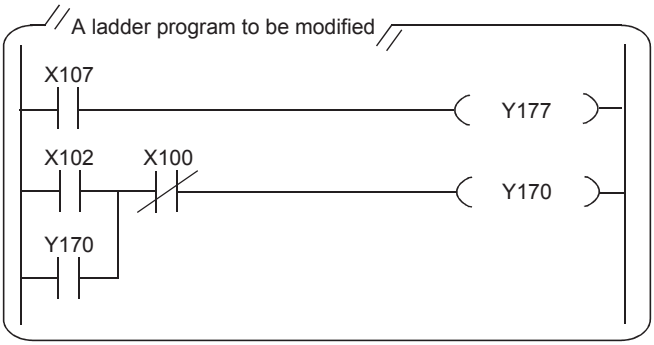


3. The row is deleted.

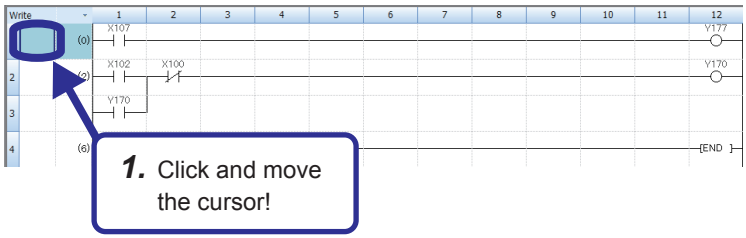
4. Click [Convert] → [Convert] (F4) from the menu to convert the ladder program.

2.9.6 Cutting or copying a ladder

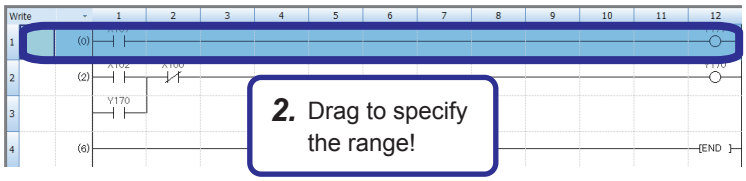
Operating procedure



This section describes how to cut or copy a part of the ladder program shown on the left and paste the cut part or the copy of the part to any desired location in the ladder.

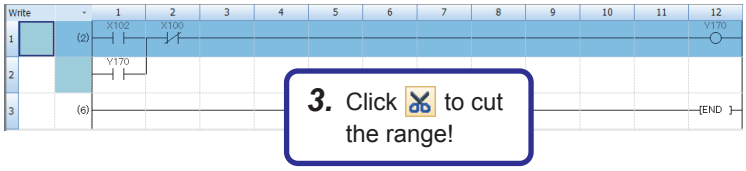


1. Click and move the cursor to the position where a part of the ladder program is to be cut.



2. Drag the mouse to specify the cutting range. The color of the specified range is highlighted.

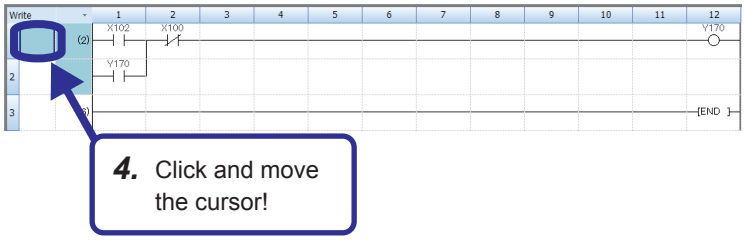
To easily specify the range in units of ladder blocks, click the position where a step number is displayed and drag the mouse vertically.



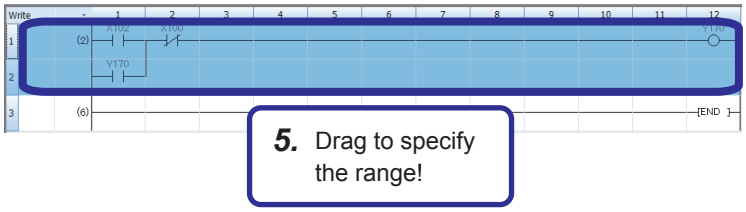
3. Click on the toolbar, or select [Edit] → [Cut] (Ctrl + X) to cut the ladder in the specified range.

(To the next page)

(From the previous page)

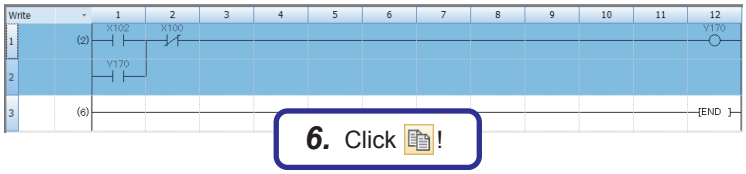


4. Click and the move the cursor to the position where a part of the ladder program is to be copied.

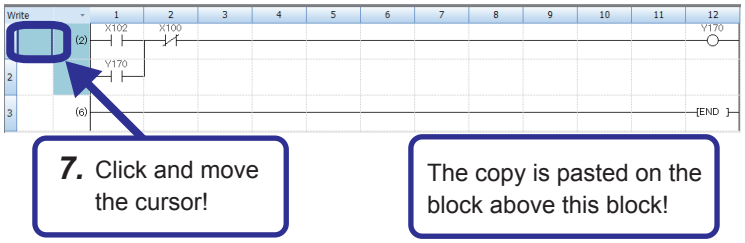


5. Drag the mouse to specify the copy range. The color of the specified range is highlighted.

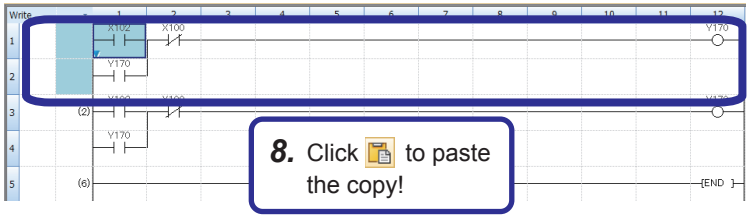
To easily specify the range in units of ladder blocks, click the position where a step number is displayed and drag the mouse vertically.



6. Click on the toolbar, or select [Edit] → [Copy] (Ctrl + C).



7. Click and move the cursor to a position on the ladder block (any position on the block) where the copy is to be pasted.

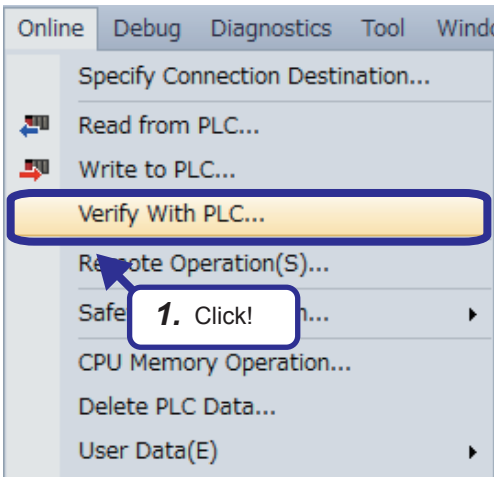


8. Click on the toolbar, or select [Edit] → [Paste] (Ctrl + V) from the menu.

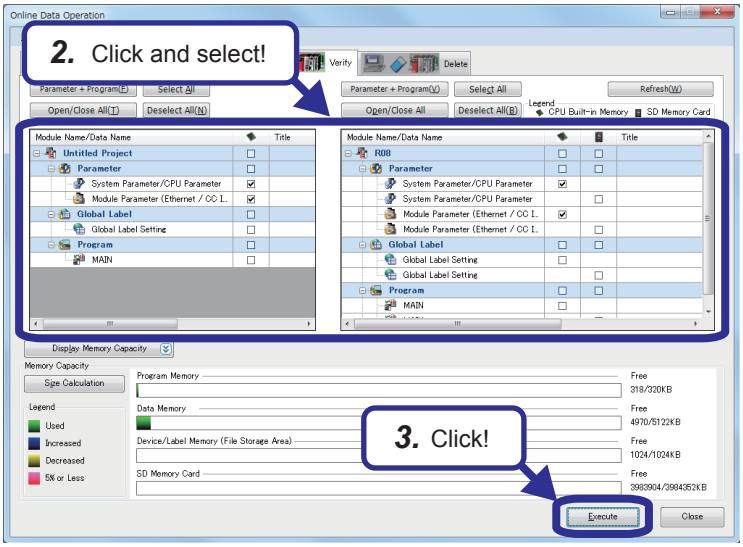
2.10 Verifying Data

This section describes how to verify the currently-opened project and the data stored in the CPU module.
Perform this operation to check whether the projects are identical or to check changes in a program.

Operating procedure

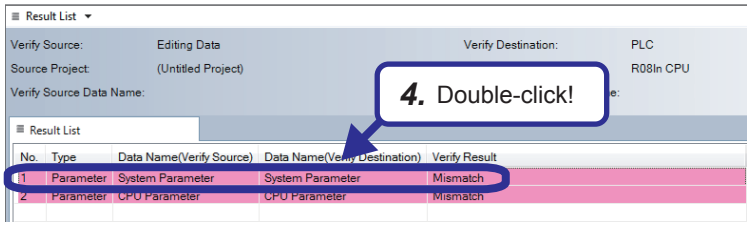


1. Click [Online] → [Verify with PLC] from the menu.



2. The "Online Data Operation" dialog box appears. Select data to be verified.

3. Click the [Execute] button.

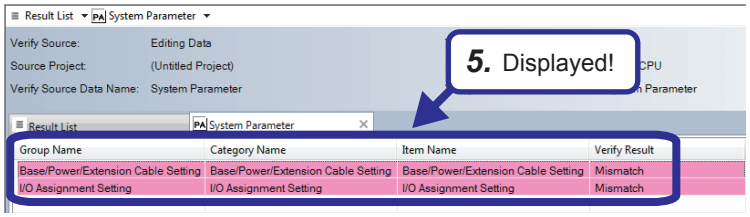


4. Verification results are displayed in the "Result List" window.

To check details of data, double-click the row of the data.

(To the next page)

(From the previous page)

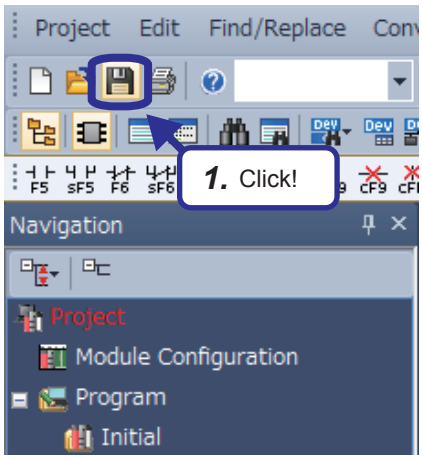


5. Detailed results are displayed.

2.11 Saving a Created Ladder Program

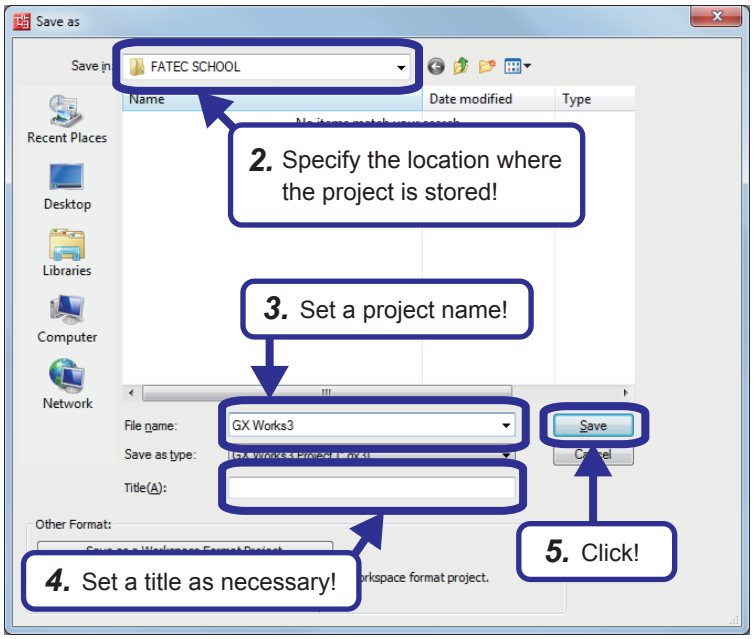
2.11.1 Saving a program in the single file format

Operating procedure



1. Click on the toolbar, or select [Project] → [Save] (Ctrl + S) from the menu.

If the project is overwritten, the saving operation is completed in this step.



2. Specify the location where the project is stored.

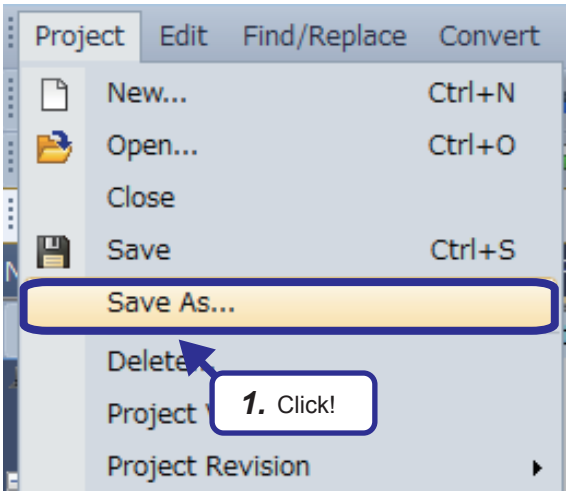
3. Set a project name.

4. Set a title as necessary.

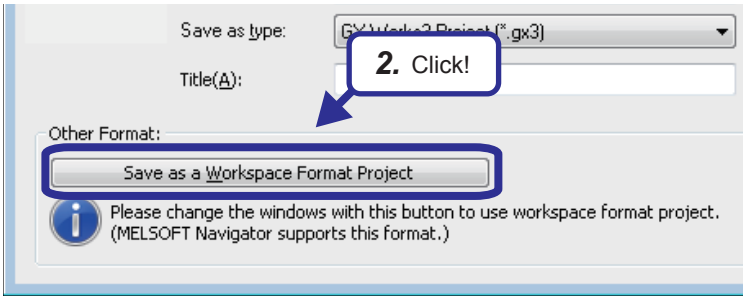
5. After setting each item, click the [Save] button. The saving operation is completed.

2.11.2 Saving a program in the workspace format

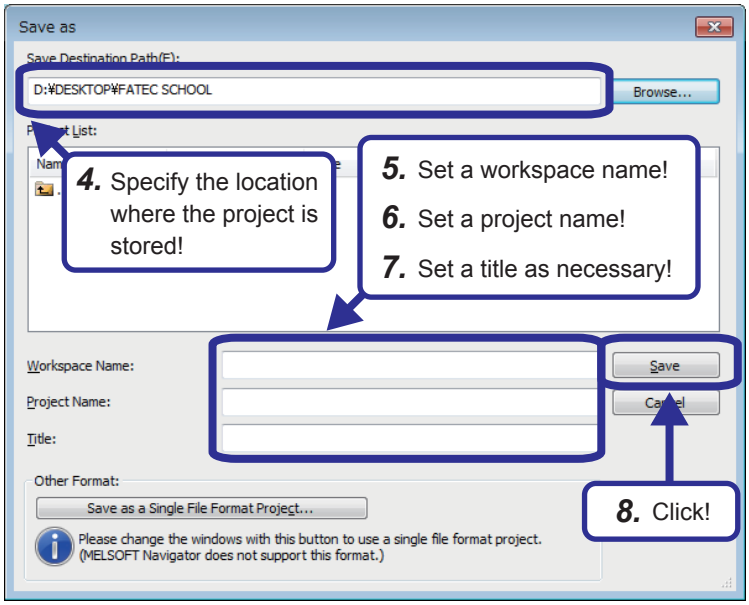
Operating procedure



1. Click [Project] → [Save as] from the menu.



2. Click the [Save as a Workspace Format Project] button at the bottom left in the "Save as" dialog box.



3. The dialog box for saving the project in the workspace format appears.

4. Specify the location where the project is stored.

5. Set a workspace name.

6. Set a project name.

7. Set a title as necessary.

8. After setting each item, click the [Save] button. The saving operation is completed.

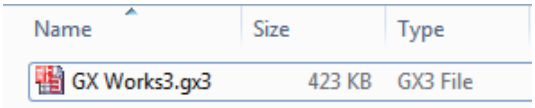
Point

Single file format

A single file format is for handling a project as a single file.

By saving a project in the single file format, users can manage the project without considering folder and file structures. Users can easily perform operations such as changing a project name, copying or pasting a project, or sending/receiving data in Explorer.

<Single file format project (*.gx3)>



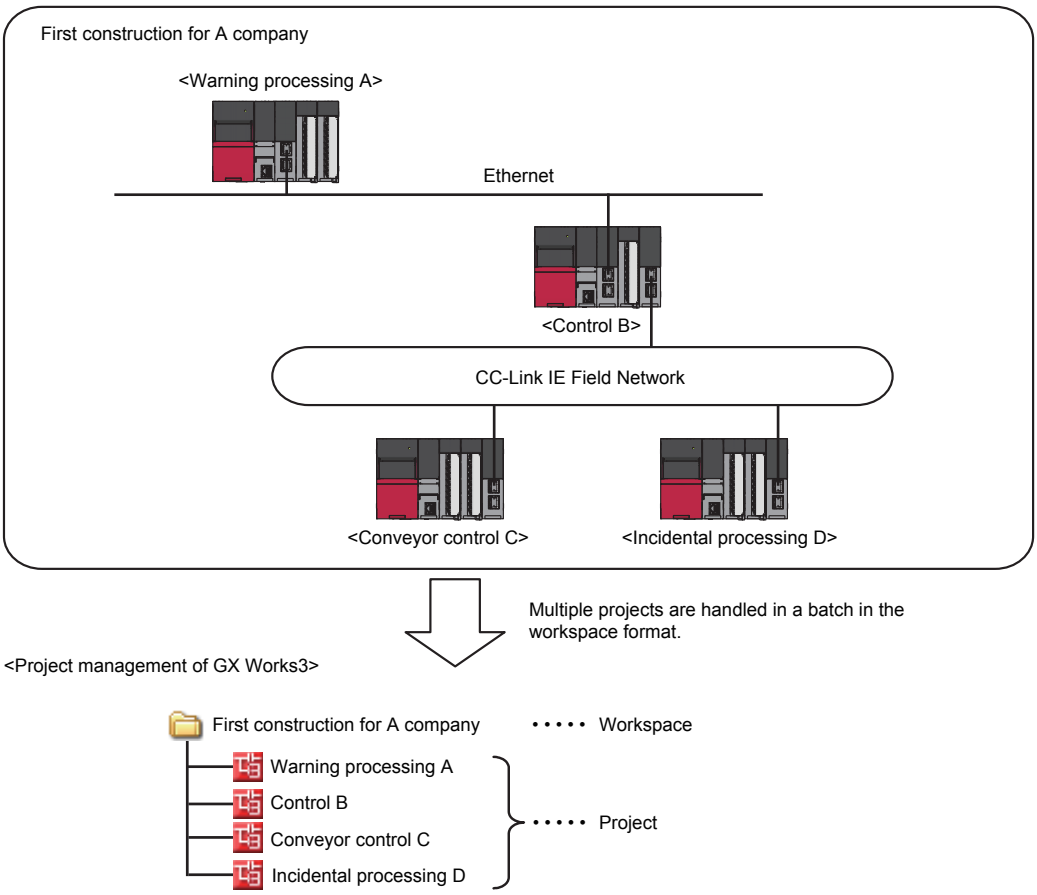
Workspace format

A workspace format is for handling multiple projects in a batch.

To build a system consisting of multiple CPU modules, a project needs to be created for each CPU module. However, when users create and save projects of the system in the workspace format, they can manage the projects in a workspace.

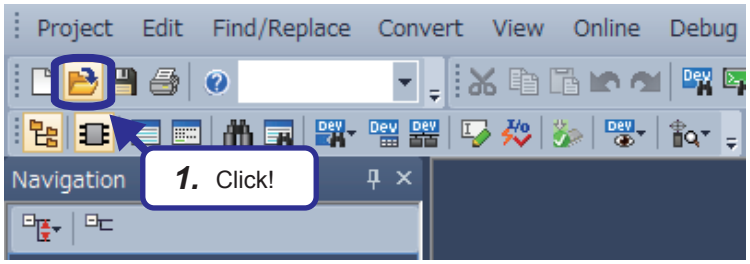
When using MELSOFT Navigator, save projects in the workspace format.

<System configuration example>

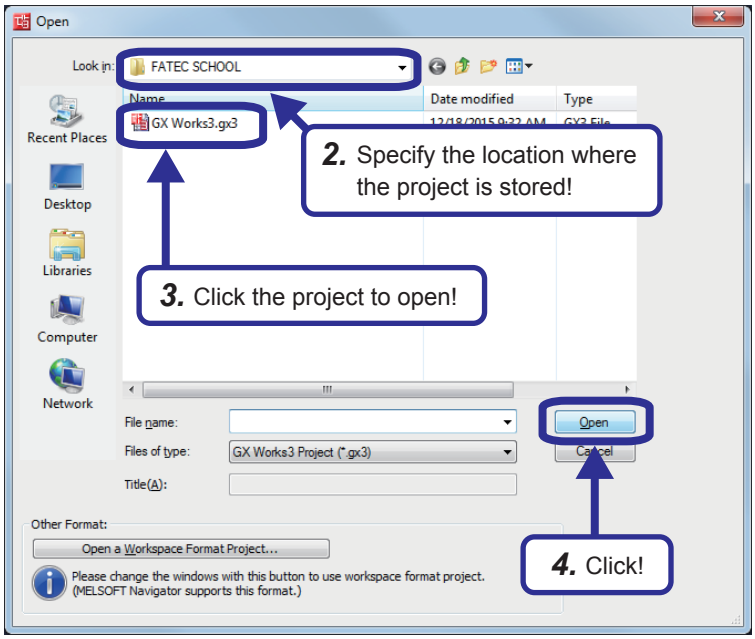


2.12 Opening a Saved Project

Operating procedure



1. Click on the toolbar, or click [Project] → [Open] (Ctrl + O) from the menu.

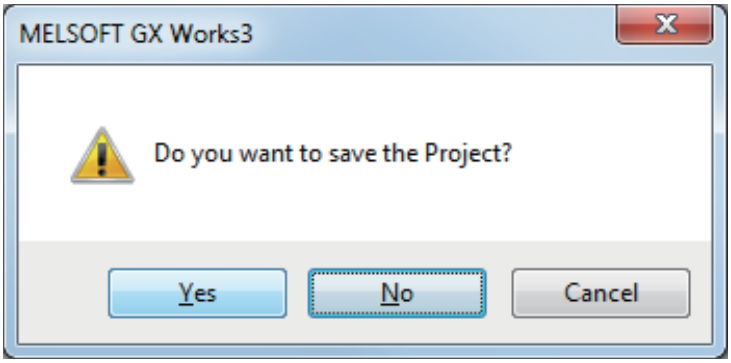


2. Specify the location where the project is stored.

3. Click the project to open.

4. After setting each item, click the [Open] button.

If another project is open and has not been saved, the following dialog box appears.

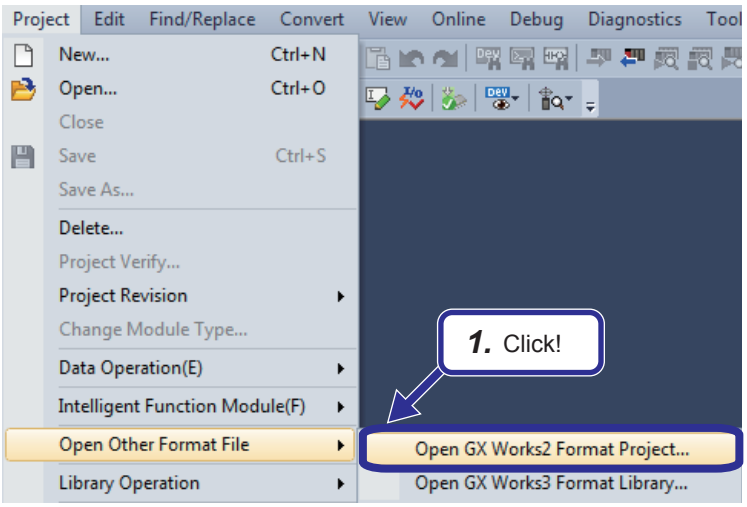


[Yes] • • • • • The project is saved and closed.
[No] • • • • • The project is not saved but closed.
[Cancel] • • • The project is not closed.

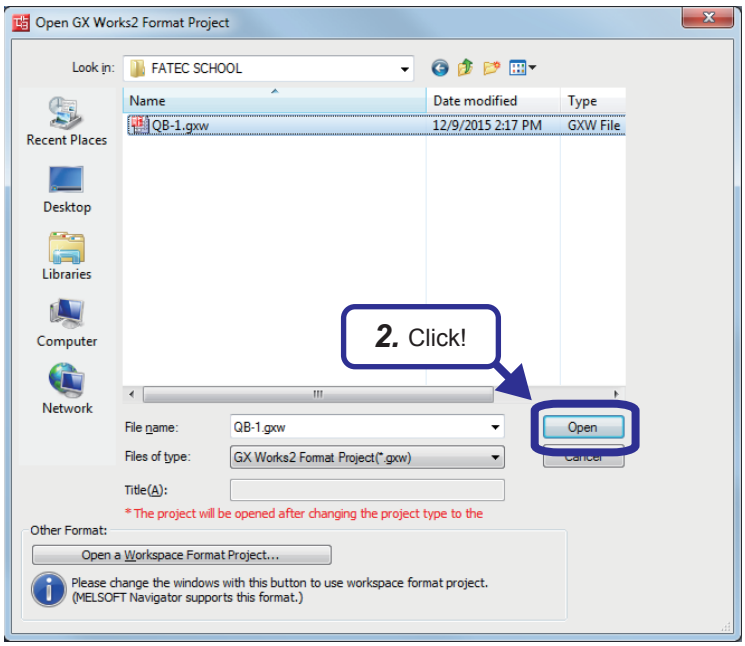
2.13 Opening a Project in Another Format

Open a project created with GX Works2 by changing the module type with GX Works3.

Operating procedure



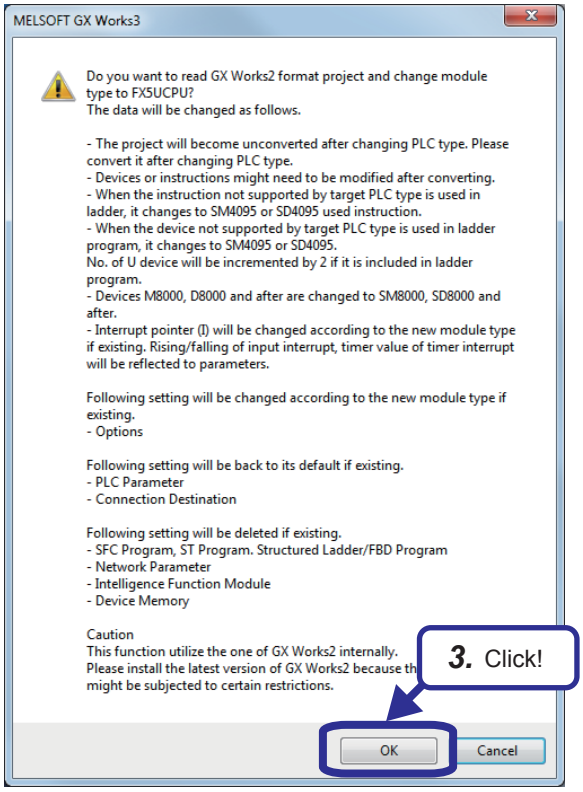
1. Click [Project] → [Open Other Format File] → [Open GX Works2 Format Project] from the menu.



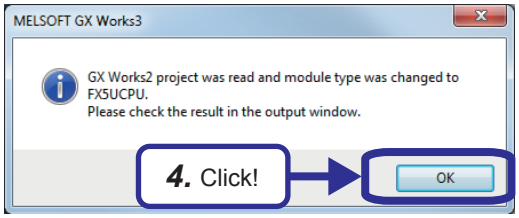
2. Specify a project in the "Open GX Works2 Format Project" dialog box, and click the [Open] button.

(To the next page)

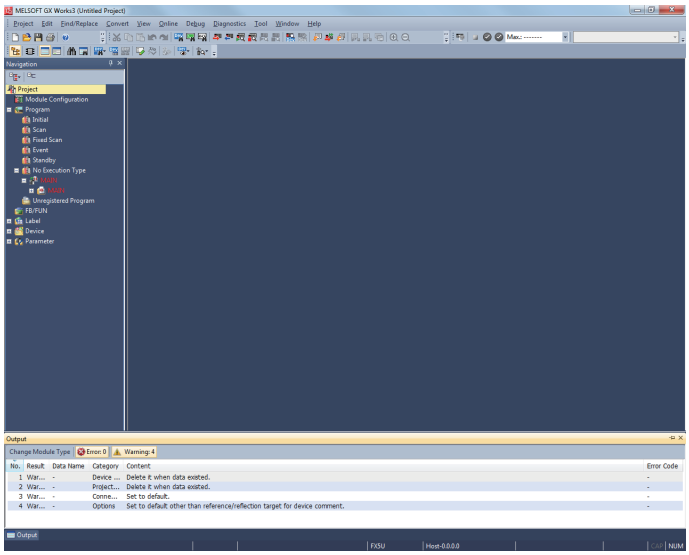
(From the previous page)



3. The message dialog box shown on the left appears. Click the [OK] button.
Changes in the project data due to the module type change are displayed on the output window.



4. When the module type change is completed, the message window shown on the left appears. Click the [OK] button.



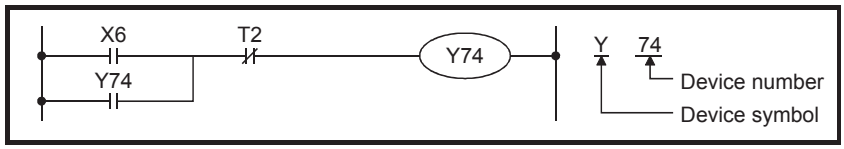
5. The project created with GX Works2 is read.

MEMO

3 DEVICES AND PARAMETERS OF A PROGRAMMABLE CONTROLLER

3.1 Devices

A device is an imaginary element for a program in the programmable controller CPU, as well as components (such as contacts and coils) of a program.



Type		Description	Remarks
X	Input	Provides the programmable controller with commands and/or data using external devices, such as push buttons, transfer switches, limit switches, and digital switches.	• Bit device • Mainly handles on/off signals.
Y	Output	Outputs control results to solenoids, electromagnetic switches, signal lights, and digital indicators.	
M	Internal relay	An auxiliary relay inside a programmable controller that cannot output signals directly to external devices	
L	Latch relay	An auxiliary relay inside a programmable controller that cannot output signals directly to external devices. Data in this device is held at power-off.	
B	Link relay	An internal relay for data link that cannot output signals directly to external devices. The areas not assigned in the link initial information setting can be used as the internal relay.	
F	Annunciator	A device used for detecting failures. Create a failure detection program in advance and turn on the device while the programmable controller is running to store a numerical value in the special register (SD).	
V	Edge relay	An internal relay that stores operation results (on/off information) from the beginning of a ladder block	
SM	Special relay	An internal relay that stores the CPU module status	
SB	Link special relay	An internal relay for data link that indicates the communication status or an error	
FX	Function input	An internal relay that loads the on/off data specified by a subroutine call instruction with argument in a subroutine program	
FY	Function output	An internal relay that passes operation results (on/off data) in a subroutine program to a subroutine program call source	

Type		Description	Remarks
T(ST)	Timer	Four types of up-timing timers are provided: low-speed timer, high-speed timer, low-speed retentive timer, and high-speed retentive timer	• Word device • Mainly handles data. • One word consists of 16 bits. • A bit of a device can be specified by "device number.*". (* = 0 to F (hexadecimal))
C	Counter	Two types of up-timing counters are provided: counters used in sequence programs and counters used in interrupt sequence programs	
D	Data register	Memory that stores data in a programmable controller	
W	Link register	A data register for data link	
R/ZR	File register	A register for extending data registers, which uses the standard RAM or memory card	
SD	Special register	A register that stores the CPU module status	
SW	Link special register	A data register for data link that stores the communication status or error definition	
FD	Function register	A register used for passing data between the subroutine call source and the subroutine program	
U3En\G U3En\HG	CPU buffer memory access device	A device that accesses the memory used by the built-in functions of the CPU module, such as the Ethernet function and the function for writing/reading data between CPU modules in the multiple CPU system	
Z	Index register	A register used for indexing devices (X, Y, M, L, B, F, T, C, D, W, R, K, H, and P)	
RD	Refresh data register	A device provided to be used as a refresh destination of the buffer memory	
N	Nesting	Shows the nesting (nesting structure) of the master control.	
P	Pointer	Points the jump destination of the branch instructions (CJ, SCJ, CALL and JMP).	
I	Interrupt pointer	When an interrupt factor occurs, this device points a jump destination to the interrupt program corresponding to the interrupt factor.	
J	Network No. specification device	Use this device when specifying a network number with the data link instruction.	
LT(LST)	Long timer	An up-timing timer that holds the current value in 32 bits	• Word device • Mainly handles data. • One word consists of 32 bits. A bit of a device can be specified by "device number.*". (* = 0 to F (hexadecimal))
LC	Long counter	An up-timing counter that holds the number of times that an input condition turns on in a program in 32 bits	
LZ	Long index register	A register used for indexing devices (X, Y, M, L, B, F, T, C, D, W, R, K, H, and P) with 32 bits	
U	I/O No. specification device	A device used to specify an I/O number with the intelligent function module dedicated instruction	• Word device • Mainly handles data. • One word consists of 16 bits. • A bit of a device can be specified by "device number.*". (* = 0 to F (hexadecimal))
K	Decimal constant	A device used to specify the following; set value of a timer/counter, pointer number, interrupt pointer number, the number of digits of a bit device, and values of a basic/application instruction.	
H	Hexadecimal constant	A device used to specify values of a basic/application instruction	
E	Real constant	A device used to specify real numbers to an instruction	
"String"	Character string constant	A device used to specify character strings to an instruction	
Jn\X Jn\Y Jn\B Jn\SB	Link direct device	A device that can directly access a link device of a network module (The refresh parameter setting is not required.)	• Bit device • Mainly handles on/off signals.
Jn\W Jn\SW			• Word device • Mainly handles data. • One word consists of 16 bits.
Un\G	Module access device	A device that accesses the buffer memory of an intelligent function module	

3.2 Parameters

The parameters are basic setting values applied to a programmable controller to control objects as desired.
The parameters are classified into three types: system parameters, CPU parameters, and memory card parameters.

System parameters

The following is the list of system parameters.

Item			Parameter No.
I/O Assignment	Base/Power/Extension Cable Setting	Base/Power Supply Module/Extension Cable model name setting	0203H
	I/O Assignment Setting	Slot	0201H
		Module Name	0203H
		Module Type/Points/Start XY/Module Status Setting	0200H
		Control PLC Setting	0202H
	Setting of Points Occupied by Empty Slot		0100H
Multiple CPU Setting	Number of CPU modules		0301H
	Communication Setting between CPU	Refresh Area Setting	0303H
		CPU Buffer Memory Setting (Refresh (At the END))	0304H
		CPU Buffer Memory Setting (Refresh (At I45 Exe.))	0308H
		PLC Unit Data	0309H
		Fixed Scan Communication Function	-
		Fixed Scan Communication Area Setting	0307H
	Fixed Scan Communication Setting	Fixed Scan Interval Setting of Fixed Scan Communication	0306H
		Fixed Scan Communication Function and Inter-module Synchronization Function	0306H
	Operation Mode Setting	Stop Setting	0302H
		Synchronous Startup Setting	030AH
	Other PLC Control Module Setting	I/O Setting Outside Group	0305H
Synchronization Setting within the Modules	Use Inter-module Synchronization Function in System		-
	Select Synchronous Target Unit between Unit		0101H
	Synchronous Fixed Scan Interval Setting within the Modules		0101H
	Synchronous Master Setting within the Modules		0102H

CPU parameters

The following is the list of CPU parameters.

Item			Parameter No.
Name Setting	Title Setting		3100H
	Comment Setting		3101H
Operation Related Setting	Timer Limit Setting		3200H
	RUN-PAUSE Contact Setting		3201H
	Remote Reset Setting		3202H
	Output Mode Setting of STOP to RUN		3203H
	Module Synchronous Setting		3207H
	Clock Related Setting		3209H
Interrupt Settings	Fixed Scan Interval Setting		3A00H
	Fixed Scan Execution Mode Setting		3A00H
	Interrupt Enable Setting in Executing Instruction		3A00H
	Block No. Save/Recovery Setting		3A00H
	Interrupt Priority Setting from Module		3A01H
Service Processing Setting	Device/Label Access Service Processing Setting		3B00H
File Setting	File Register Setting		3300H
	Initial Value Setting		3301H
	File Setting for Device Data Storage		3303H
Memory/Device Setting	Device/Label Memory Area Setting	Extended SRAM Cassette	3404H
		Device/Label Memory Area Capacity Setting	3400H
		Device Points	3401H
		Local Device	3405H
		Latch Range Setting	3407H
		Latch Type Setting of Latch Type Label	3408H
	Index Register Setting		3402H
	Refresh Memory Setting		3403H
	Device Latch Interval Setting		3406H
	Pointer Setting		340BH
	Internal Buffer Capacity Setting		340AH
RAS Setting	Scan Time Monitoring Time (WDT) Setting		3500H
	Constant Scan Setting		3503H
	Error Detection Setting		3501H
	CPU Module Operation Setting at Error Detected		3501H
	LED Display Setting		3502H
	Event History Setting		3504H
Program Setting	Program Setting	Program Name	3700H
		Execution Type	3700H
		Type (Fixed Scan)	3700H
		Type (Event)	3701H
		Detail Setting Information	-
		Refresh Group Setting	3700H
		Device/File Use or not	3700H
	FB/FUN File Setting		3702H
Refresh Setting between Multiple CPU	Refresh Setting (At the END)		3901H
	Refresh Setting (At I45 Exe.)		3902H
Routing Setting	Routing Setting		3800H

Memory card parameters

The following is the list of memory card parameters.

Item	Parameter No.
Boot Setting	2000H
Setting of File/Data Use or Not in Memory Card	2010H

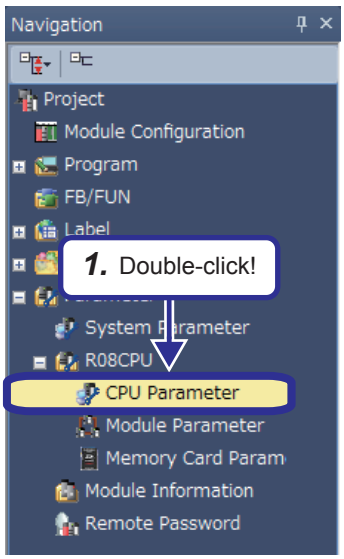
- When GX Works3 starts, the preset values are set in parameters. These values are called default values (initial values).
- The programmable controller can run with default values, however, modify them within a specified range as necessary.

■Operation example: Changing the operation mode when an error occurs

When an operation error occurs, the status of the RCPU changes to STOP with the default value, however, changing the parameters allows the RCPU to keep the RUN state.

(1) Operation error example

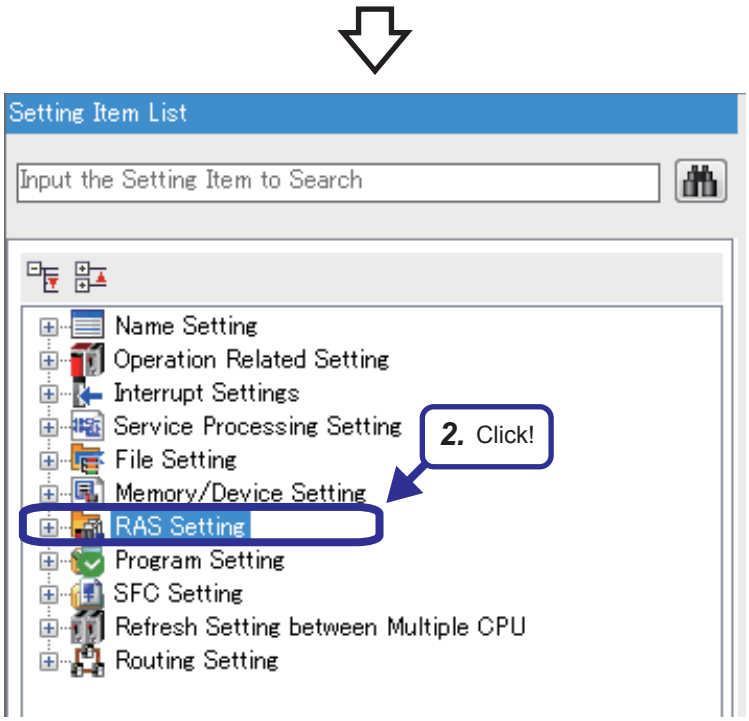
In the division instruction, the processing to divide a value by 0 is executed.



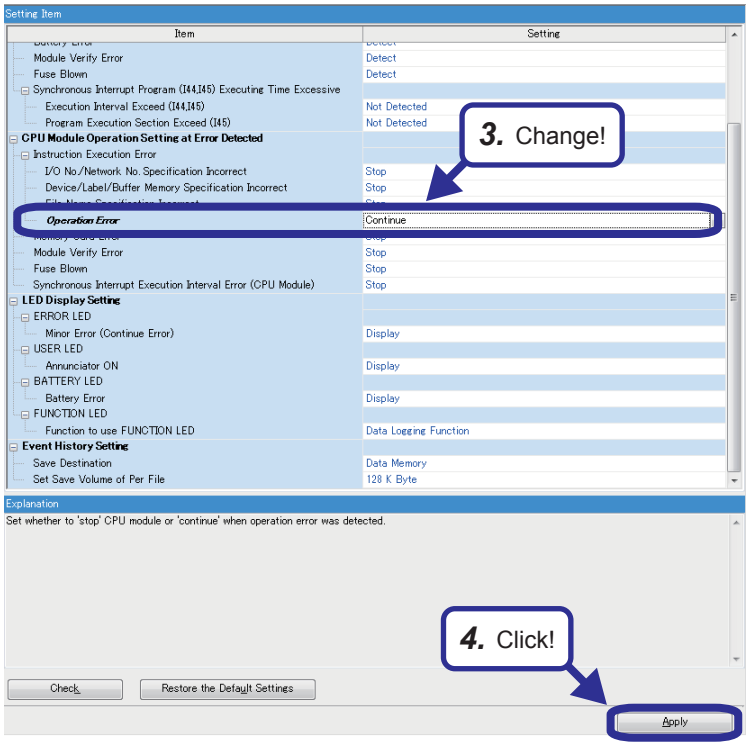
1. Double-click "CPU Parameter" in the "Navigation" window.

(To the next page)

(From the previous page)



2. The "CPU Parameter setting" dialog box appears. Click "RAS Setting" in "Setting Item List".



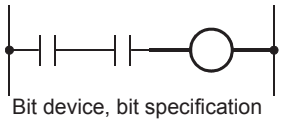
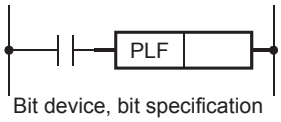
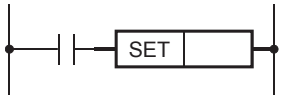
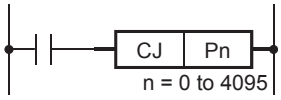
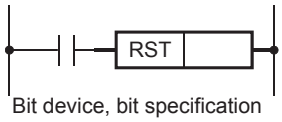
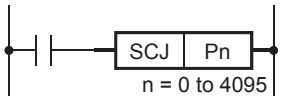
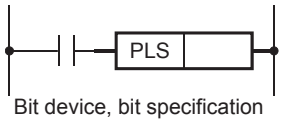
3. Change the setting of "Operation Error" in "CPU Module Operation Setting at Error Detected" to "Continue".

4. Click the [Apply] button.

4 SEQUENCE INSTRUCTIONS AND BASIC INSTRUCTIONS -PART 1-

4.1 Instructions Described in This Chapter

This chapter describes the following sequence instructions and basic instructions.

Instruction symbol	Function	Ladder (device to be used)	Instruction symbol	Function	Ladder (device to be used)
OUT	Coil output	 Bit device, bit specification of word device	PLF	Pulf (Generating pulses for one program cycle at the falling edge of the input signal)	 Bit device, bit specification of word device
SET	Setting devices	 Bit device, bit specification of word device	CJ	Conditional jump (non-delay execution)	 n = 0 to 4095 Pointer
RST	Resetting devices	 Bit device, bit specification of word device	SCJ	Conditional jump (executed after one scan)	 n = 0 to 4095 Pointer
PLS	Pulse (Generating pulses for one program cycle at the rising edge of the input signal)	 Bit device, bit specification of word device			

4.1.1 Instructions not described in this chapter -Part 1-

The following table lists the instructions not described in this chapter. These instructions are introduced in "Introduction: PLC Course" and supported by conventional MELSEC-A series.

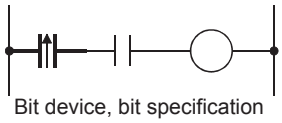
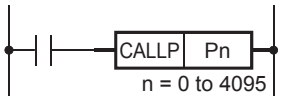
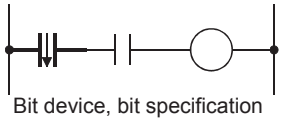
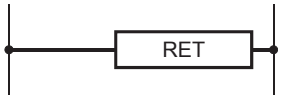
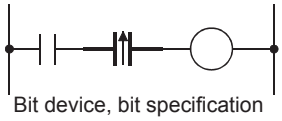
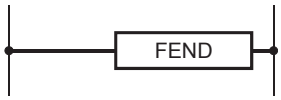
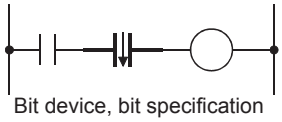
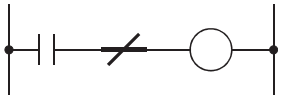
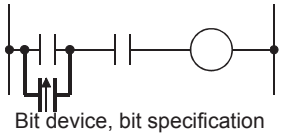
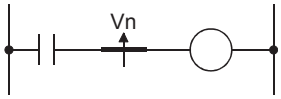
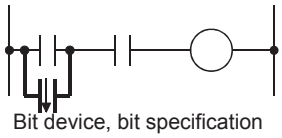
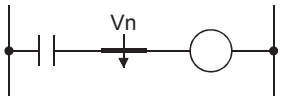
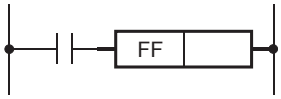
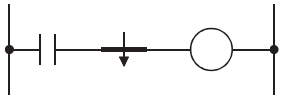
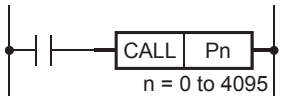
For details, refer to the MELSEC iQ-R Programming Manual (Instructions, Standard Functions/Function Blocks).

Instruction symbol	Function	Ladder (device to be used)	Instruction symbol	Function	Ladder (device to be used)
LD	Starting a logical operation (Starting a normally open contact operation)	Bit device, bit specification of word device	MRD	Intermediate branching	
LDI	Starting a NOT operation (Starting a normally closed contact operation)	Bit device, bit specification of word device	MPP	Terminating branching	
AND	Logical AND operation (Series connection of normally open contacts)	Bit device, bit specification of word device	NOP	No operation	For a space or deleting a program
ANI	Logical AND inverse operation (Series connection of normally closed contacts)	Bit device, bit specification of word device	END	END processing for terminating a program	Always used at the end of a program
OR	Logical OR operation (Parallel connection of normally open contacts)	Bit device, bit specification of word device	STOP	Stopping an operation	STOP
ORI	Logical OR inverse operation (Parallel connection of normally closed contacts)	Bit device, bit specification of word device	SFT	1-bit shifting of devices	SFT Bit device, bit specification of word device
ANB	AND operation between logical blocks (Series connection of blocks)		SFTP	1-bit shifting of devices (Pulse operation)	SFTP Bit device, bit specification of word device
ORB	OR operation between logical blocks (Parallel connection of blocks)		NOPLF	No operation For inserting a page break at print out	NOPLF
MPS	Starting branching		PAGE	No operation Recognized as the step 0 on Page n	PAGE n

4.1.2 Instructions not described in this chapter -Part 2-

The following table lists the instructions for the MELSEC Q and iQ-R series. The instructions are not supported by conventional MELSEC-A series.

For details, refer to the MELSEC iQ-R Programming Manual (Instructions, Standard Functions/Function Blocks).

Instruction symbol	Function	Ladder (device to be used)	Instruction symbol	Function	Ladder (device to be used)
LDP	Starting a rising edge pulse operation	 Bit device, bit specification of word device	CALLP	Calling a subroutine program (Pulse operation)	 n = 0 to 4095 Pointer
LDF	Starting a falling edge pulse operation	 Bit device, bit specification of word device	RET	Returning from a subroutine program	
ANDP	Series connection of rising edge pulses	 Bit device, bit specification of word device	FEND	Ending the main routine program	
ANDF	Series connection of falling edge pulses	 Bit device, bit specification of word device	INV	Inverting operation results	 Bit device, bit specification of word device
ORP	Parallel connection of rising edge pulses	 Bit device, bit specification of word device	EGP	Operation result rising edge pulse conversion (recorded with Vn)	 Bit device, bit specification of word device
ORF	Parallel connection of falling edge pulses	 Bit device, bit specification of word device	EGF	Operation result falling edge pulse conversion (recorded with Vn)	 Bit device, bit specification of word device
MEP	Operation result rising edge pulse conversion	Bit device, bit specification of word device	FF	Inverting device outputs	 Bit device, bit specification of word device
MEF	Operation result falling edge pulse conversion	 Bit device, bit specification of word device	DELTA	Converting the direct output into a pulse	
CALL	Calling a subroutine program	 n = 0 to 4095 Pointer	DELTAP	Converting the direct output into a pulse	

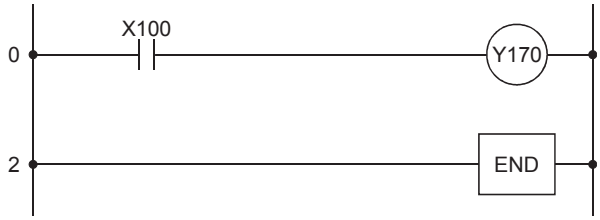
4.2 Differences Between [OUT] and [SET]/[RST]

Point

This section describes the OUT and SET/RST instructions and the operation of a self-holding ladder.

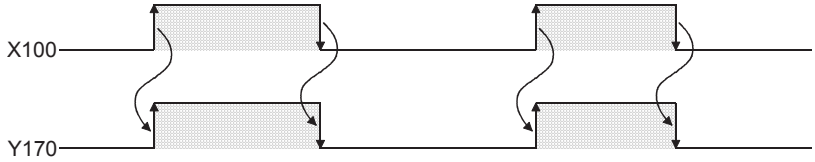
4.2.1 [OUT] (Coil output)

Project name	RB-1
Program name	MAIN



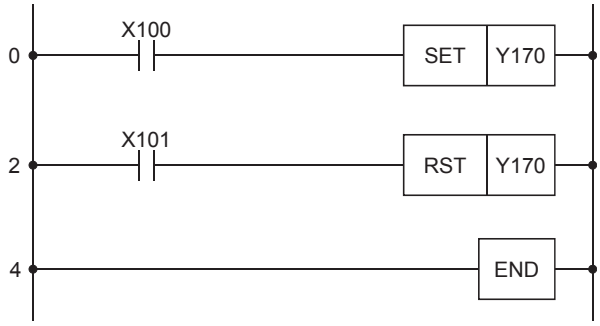
The OUT instruction turns on a specified device when the input condition turns on, and turns off the device when the condition turns off.

■Timing chart



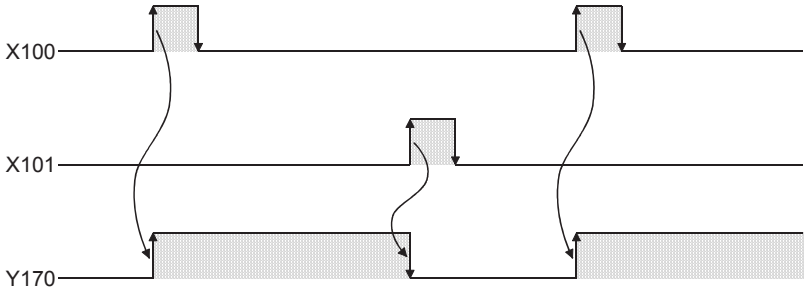
4.2.2 [SET]/[RST](Setting/resetting devices)

Project name	RB-2
Program name	MAIN



The SET instruction turns on a specified device when the input condition turns on, and holds the on state of the device even though the condition turns off.
To turn off the device, use the RST instruction.

■Timing chart

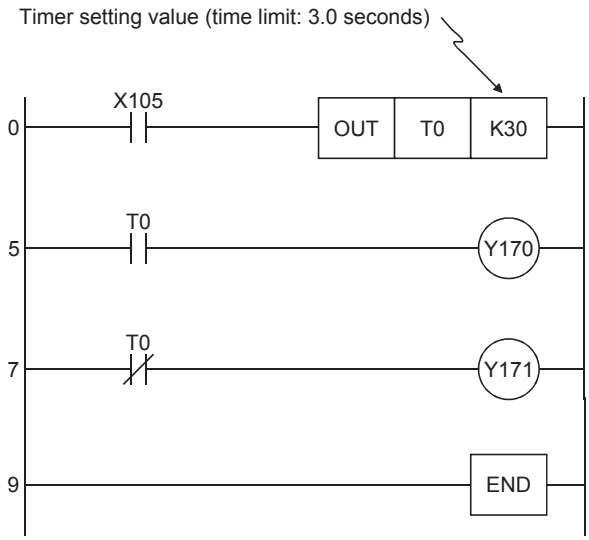


4.3 Measuring Timers (Timer, High-speed Timer, Retentive Timer)

Point

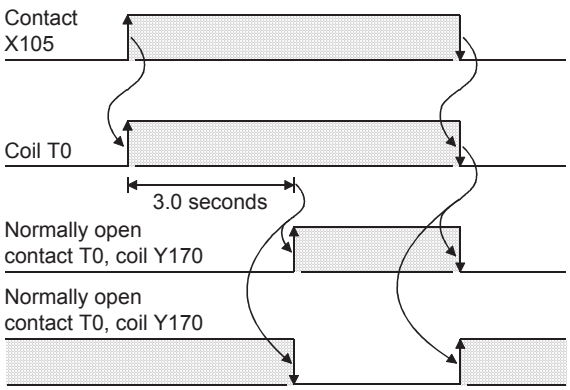
- This section describes how to input a timer.
- This section describes the parameter setting for using a retentive timer.
- This section describes the operation differences of various timers.

Project name	RB-3
Program name	MAIN



*OUT T is a 4-step instruction.

■Timing chart



- The operation of the timer contact delays by a set time after the coil is energized. (On delay timer)
- The setting range of a timer value is K1 to K32767.
  Low-speed (100ms) timer: 0.1 to 3276.7 seconds
  High-speed (10ms) timer: 0.01 to 327.67 seconds
- When the value set to a timer is 0, it is turned on (timeout) by the execution of the instruction.

The following four types of timers are available.

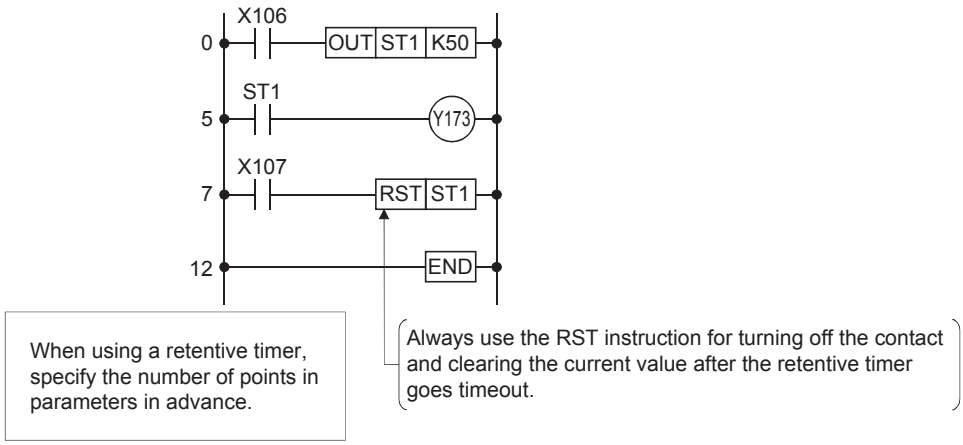
Type	Timer No. (initial value)
Low-speed timer •••••••• Counts time in increments of 100ms.	Initial value T0 to T2047 (2048 timers)
High-speed timer •••••••• Counts time in increments of 10ms.	
Low-speed retentive timer ••••• Integrates time in increments of 100ms.	Initial value: 0 (The value can be changed with parameters.)
High-speed retentive timer •••• Integrates time in increments of 10ms.	

- Change the output instruction (OUT) to <^{H}T0> to select a high-speed timer or high-speed retentive timer.
- To use retentive timers, set the number of device points used for retentive timers in the device setting of the CPU parameter.

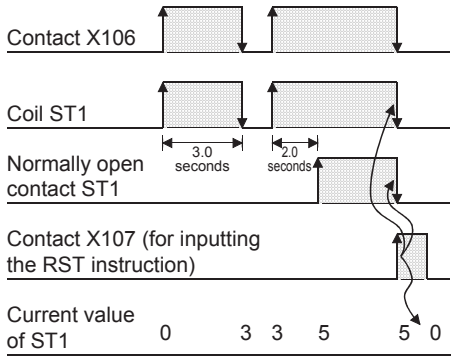
How to use retentive timers

When an input condition turns on, the coil turns on and the value of a retentive timer starts increasing. When the current value becomes equal to a set value, the retentive timer goes timeout and the contact turns on. When the input condition turns off during the addition, the coil turns off but the current value is held. When the input condition turns on again, the coil turns on and the current value is used in the integration to continuously increase the value of the timer.

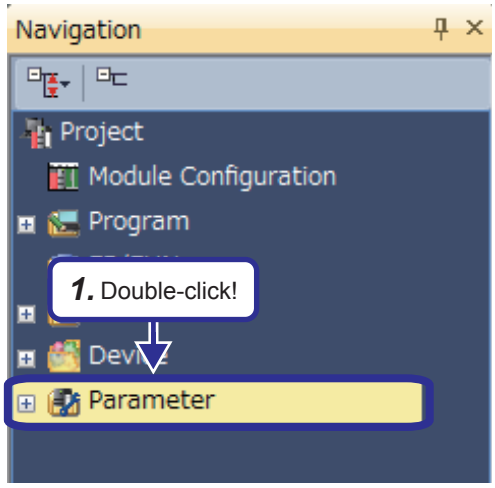
Project name	Retentive timer
Program name	MAIN



■Timing chart



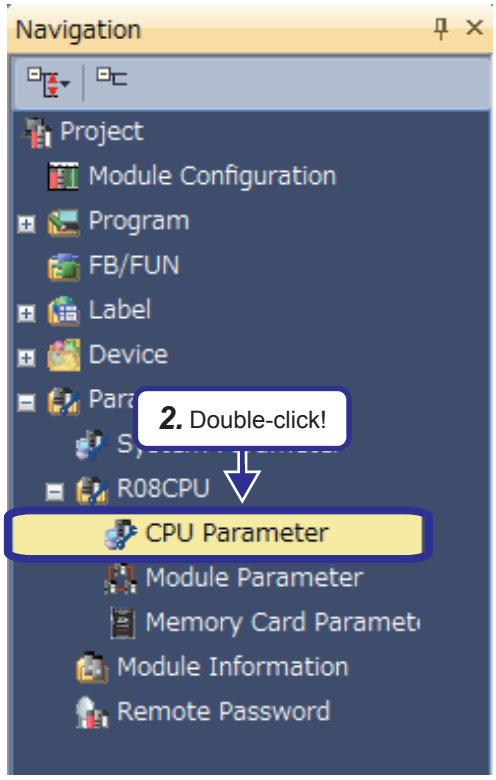
The following describes the operation of when the retentive timer is set to ST0 to ST31.



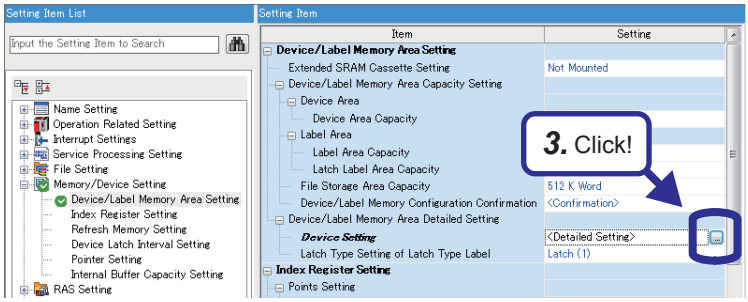
1. Double-click "Parameter" in the "Project" view.

(To the next page)

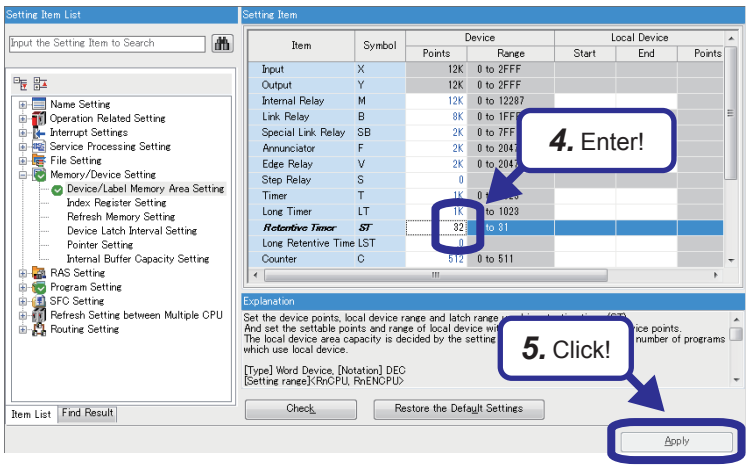
(From the previous page)



2. Double-click "CPU Parameter".



3. Click "Memory/Device Setting" in Setting Item List to display Setting Item.
Select "Device Setting" in "Device/Label Memory Area Detailed Setting" to display the reference button at the right edge. Click this button.



4. Click "Points" of "Device" for "Retentive Timer" and enter "32".

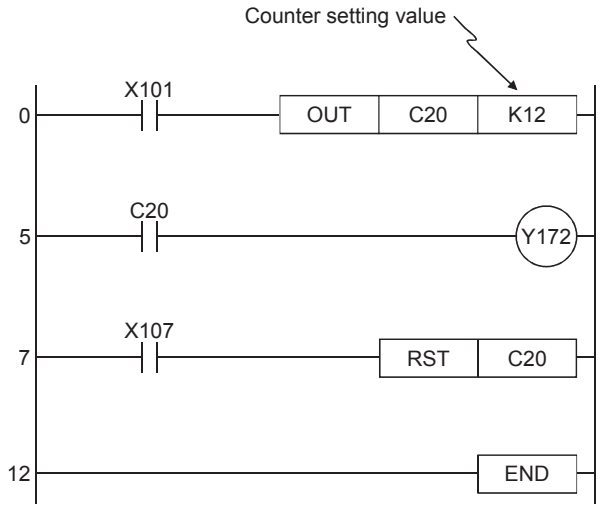
5. After the setting is completed, click the [Apply] button.

4.4 Counting with a Counter

Point

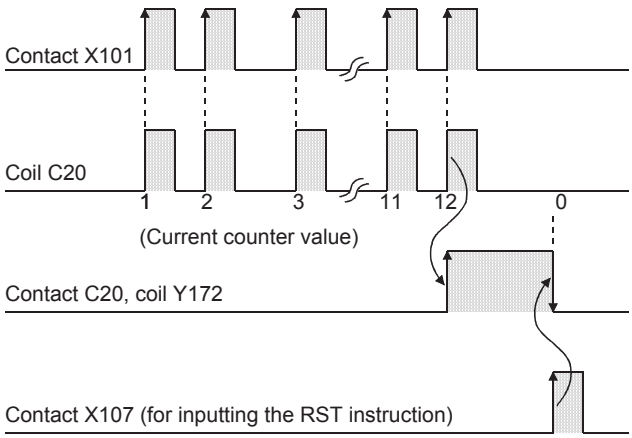
- This section describes how to input a counter.
- This section describes the words "rise (rising edge)" and "fall (falling edge)".

Project name	RB-4
Program name	MAIN



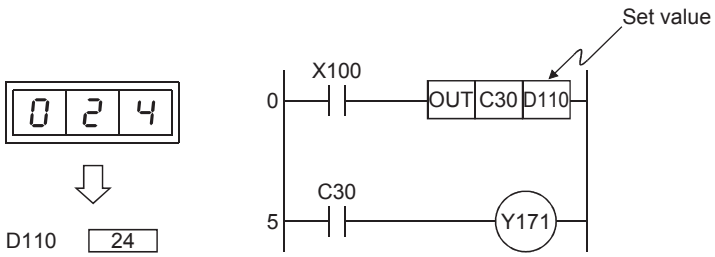
*OUT C is a 4-step instruction.

■Timing chart



- A counter counts at the rising edge of an input signal.
- After counting is up, the counter does not count at the rising edges of the subsequent input signals.
- Once counting is up, the contact status and the current value (count value of the counter) do not change until the RST instruction is executed.
- Executing the RST instruction before counting is up clears the counter value to 0.
- The setting range of a counter value is K0 and K32767. (K0 turns on (starts counting) at execution of the instruction.)

A setting value can be directly specified with K or indirectly specified with D (data register).



- The counter C30 counts up when the number of times that the input signal X100 turns on becomes equal to the value (such as 24) specified in the data register D110.
- The indirect specification is useful for using a value specified with an external device as the setting value of a counter.

Point

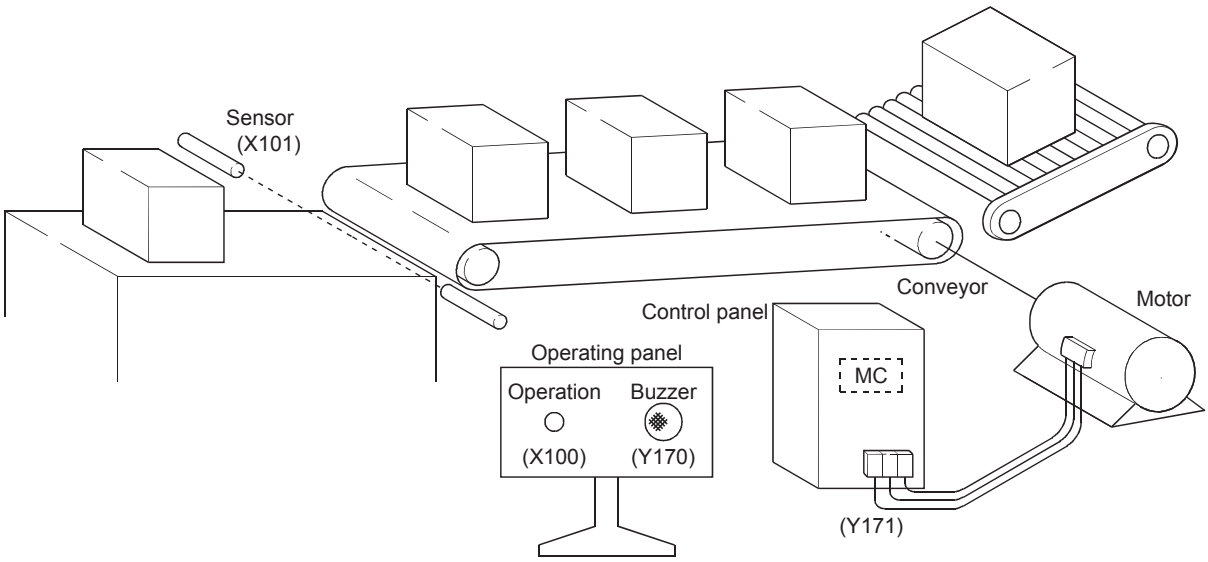
A setting value of a timer can also be indirectly specified with the data register (D) in the same way as the one for counters.

■Ladder example

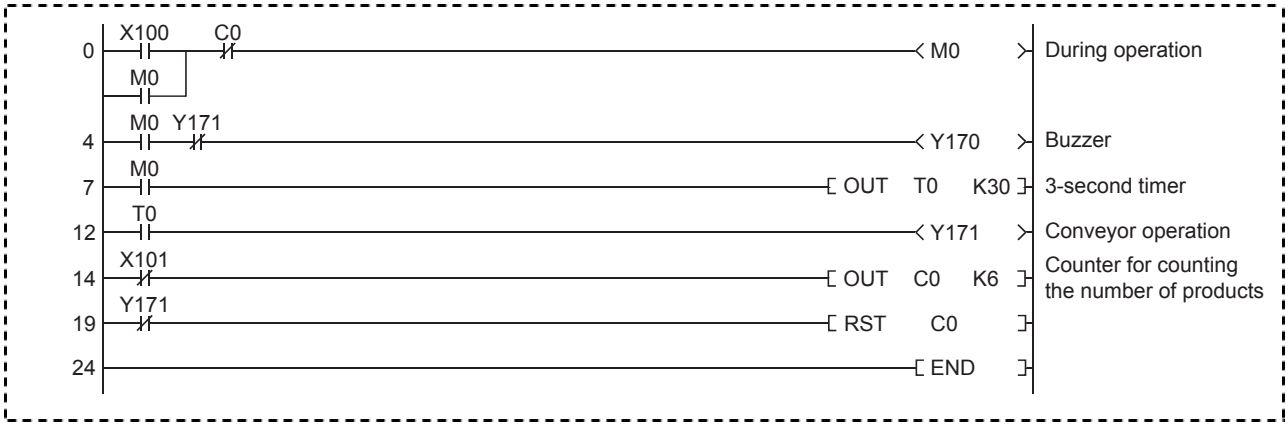
When the execution command switch (X100) of the conveyor turns on, the buzzer (Y170) beeps for three seconds and the conveyor starts to operate (Y171).

The conveyor automatically stops when the sensor (X101) detects that six products have passed through.

Project name	REX1
Program name	MAIN



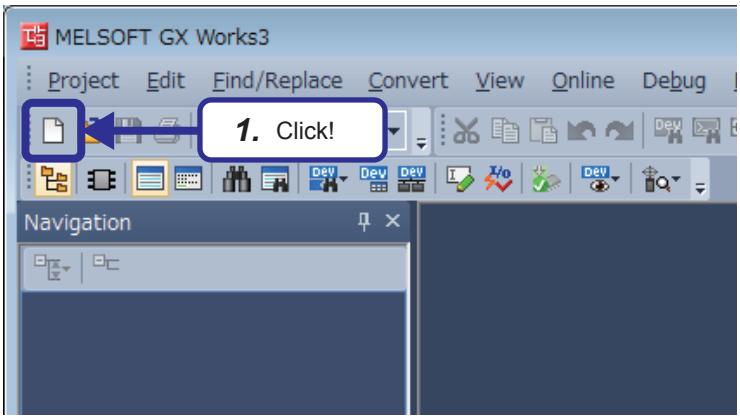
Create the following ladder program and check that it operates properly.



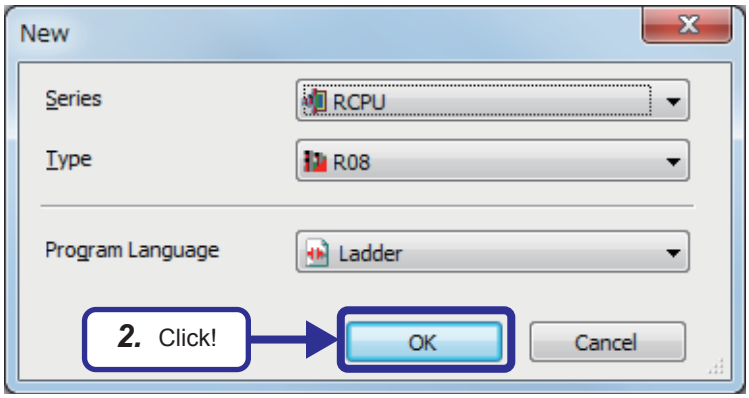
Operating procedure

■Creating a new project

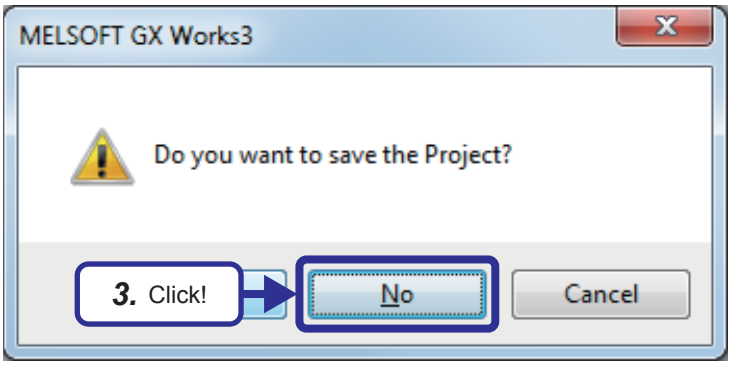
Refer to section 2.2.2 and create a new project.



1. Click on the toolbar.



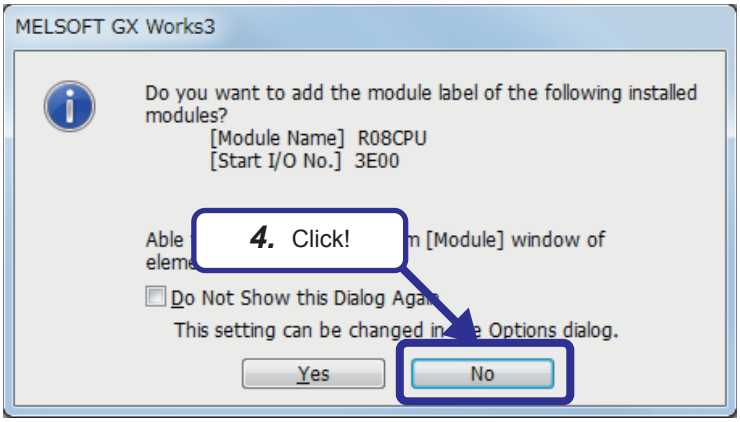
2. The "New" dialog box appears.
Set the RCPU for the series, the R08 for the type, and "Ladder" for "Program Language".
Then, click the [OK] button.



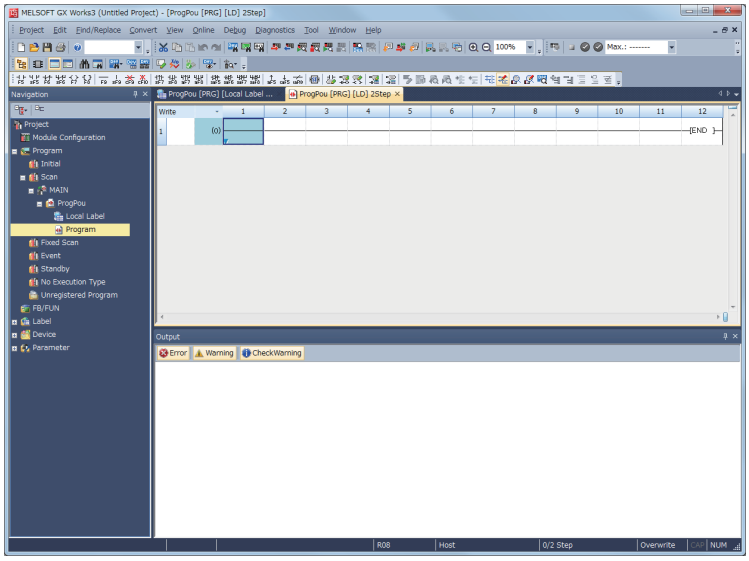
3. If a project that is being created exists, the confirmation dialog box for saving the project appears.
Click the [No] button.

(To the next page)

(From the previous page)



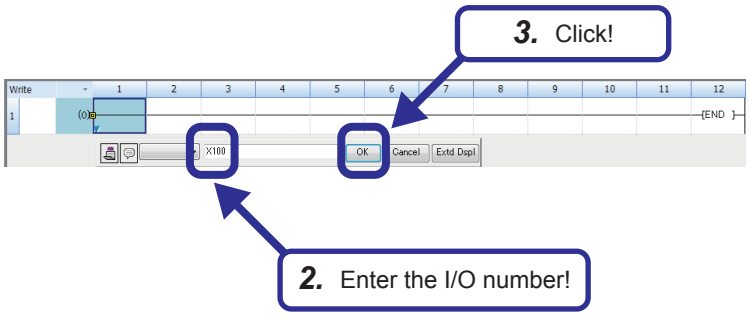
4. The confirmation dialog box for adding module labels appears.
Click the [No] button.



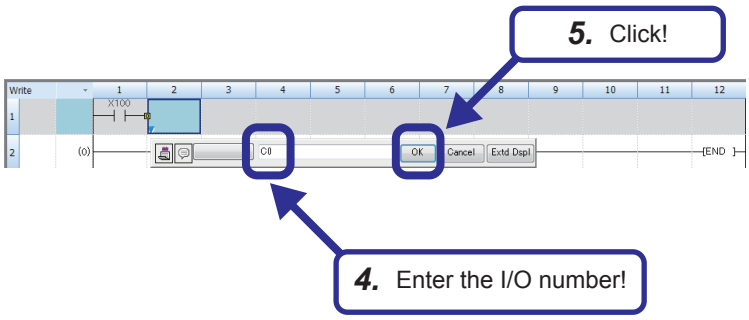
5. The window for creating a new project appears.

■Creating a program

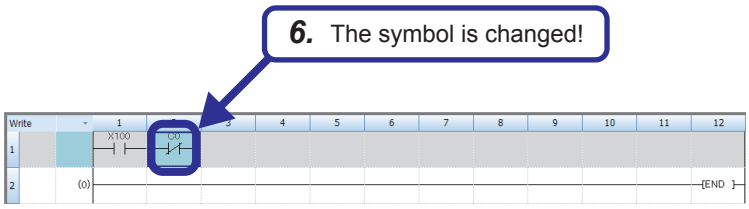
The following describes the procedure of entering devices and labels to create a ladder.



1. Move the cursor to the insertion position.
2. Enter "X100" with a keyboard.
   (When entering of the number starts, the ladder input window appears.)
3. After entering the device number, click the [OK] button.



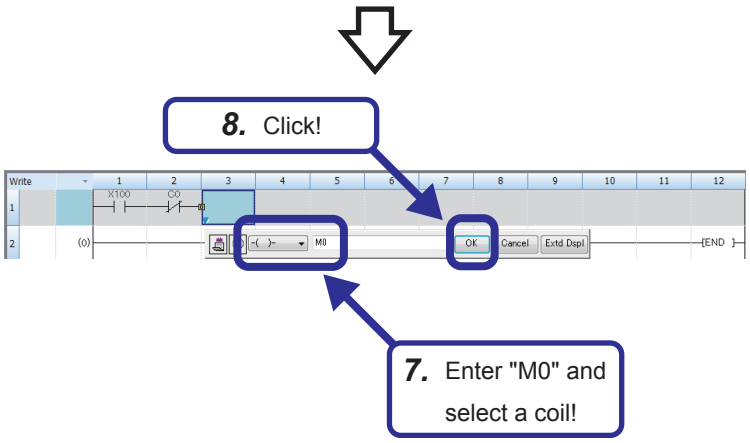
4. Move the cursor to the next insertion position and enter "C0" with the keyboard.
5. After entering the device number, click the [OK] button.



6. Select the input ladder and press the "/" key to switch the symbol.

(To the next page)

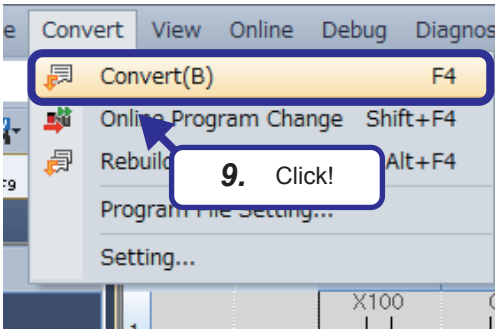
(From the previous page)



7. Move the cursor to the next insertion position, enter "M0" with the keyboard, and select a coil.

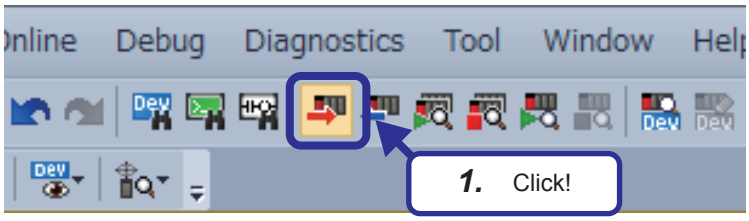
8. After entering the device number, click the [OK] button.

•
•
•
•



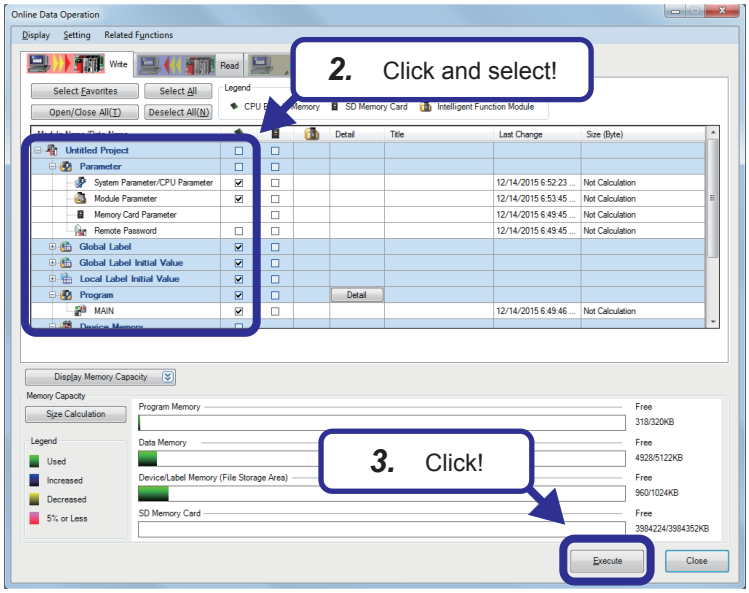
9. When creating the ladder is completed, click [Convert] - [Convert] from the menu.

■Writing data to the programmable controller



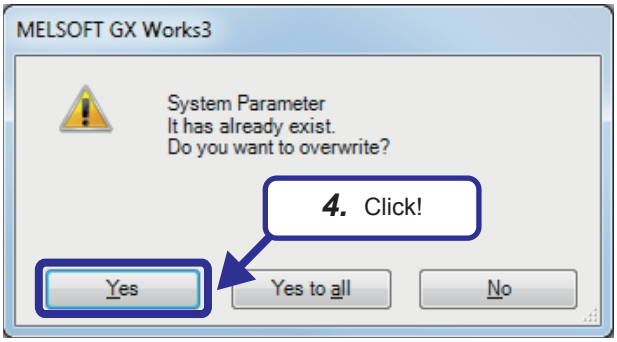
1. Write the data to the programmable controller. (Set the RUN/STOP/RESET switch of the CPU module to STOP.)

Click on the toolbar.
The "Online Data Operation" dialog box appears.



2. Select System Parameter/CPU Parameter, Module Parameter, and Program.

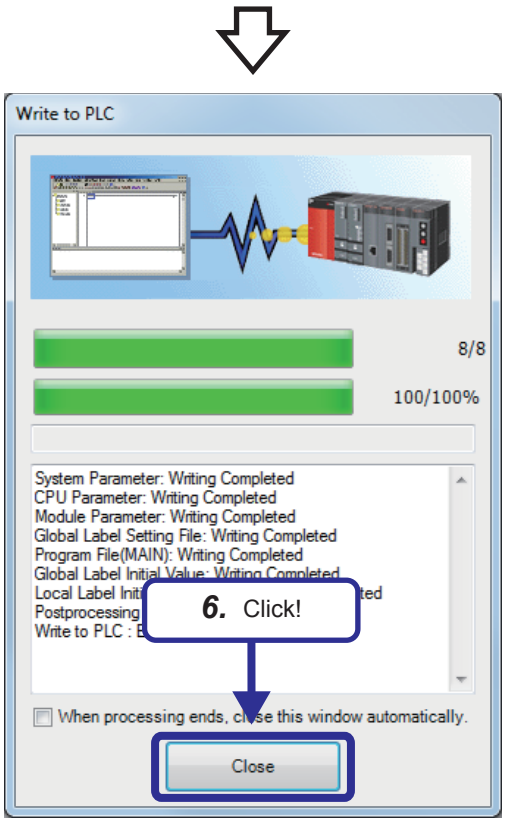
3. Click the [Execute] button.



4. If parameters have been already written, the confirmation dialog box for overwriting the parameters appears.
Click the [Yes] button.

(To the next page)

(From the previous page)



5. The "Write to PLC" dialog box appears.

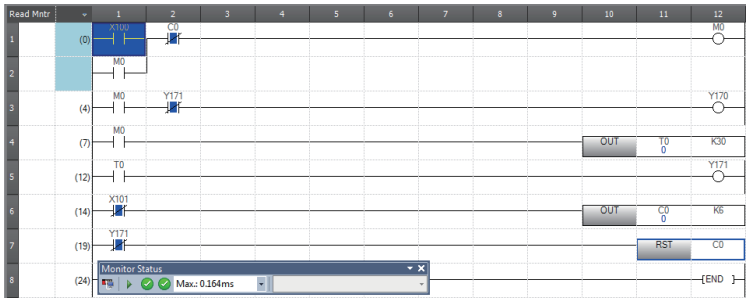
6. When writing the data is completed, the message "Completed" is displayed. Click the [Close] button.

■Monitoring the ladder

Monitor the ladder.

(Hold the RUN/STOP/RESET switch of the CPU module at the RESET position for one second or longer, and set the switch to RUN.)

1. Click on the toolbar or press the F3 key.



2. The ladder (write) window is switched to the ladder monitor window.

Operation practice

❶ Turning on X100 turns on Y170 and starts T0 at the same time.

❷ The timer T0 goes timeout in three seconds, and Y170 turns off and Y171 turns on at the same time.

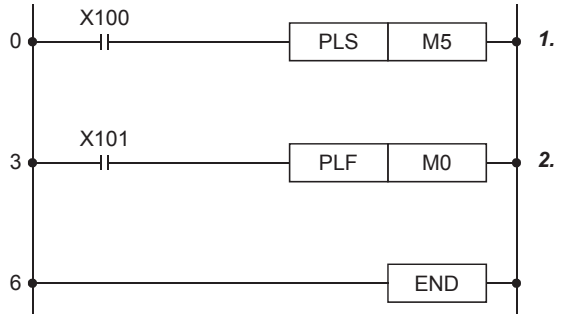
❸ Every time X101 turns on or off, the counter C0 counts the number of operations and Y171 turns off when counting is up (X101 turns on six times).

4.5 [PLS] (Turning on a Specified Device for One Scan at the Rising Edge of an Input Condition) [PLF] (Turning on a Specified Device for One Scan at the Falling Edge of an Input Condition)

Point

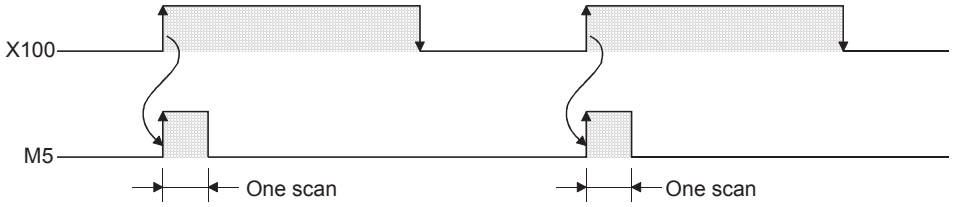
- This section describes the concept of one scan.
- This section describes the operation timing of the PLS/PLF instruction.

Project name	RB-5
Program name	MAIN



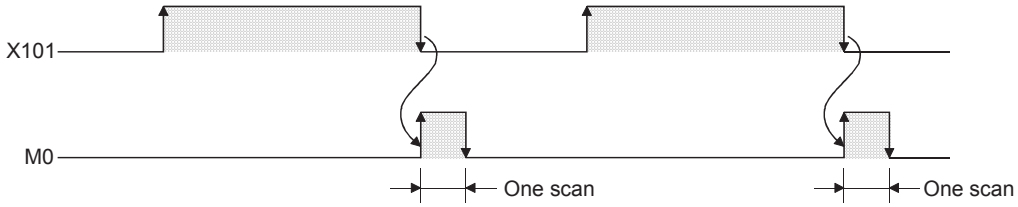
1. The PLS instruction turns on a specified device only for one scan at the rising edge of the commanded condition.

■Timing chart



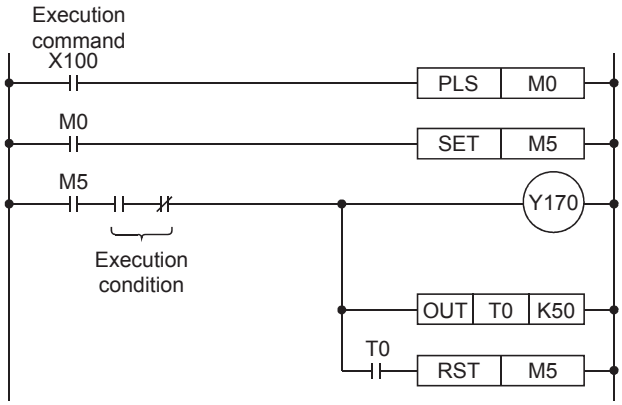
2. The PLF instruction turns on a specified device only for one scan at the falling edge of the commanded condition.

■Timing chart

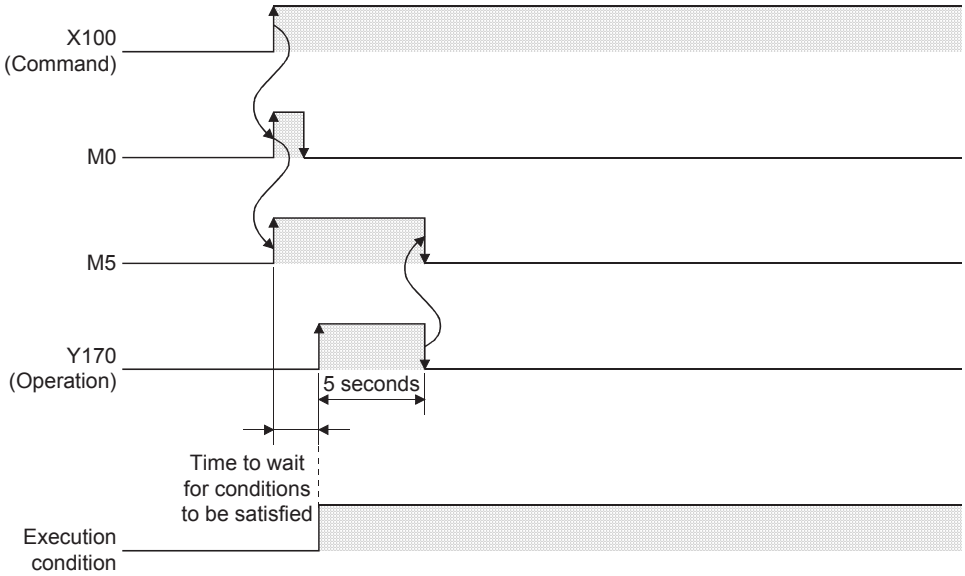


Application

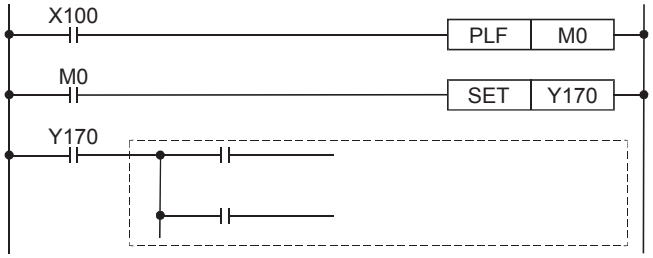
- These instructions can be used in a standby program that waits for an operation condition.

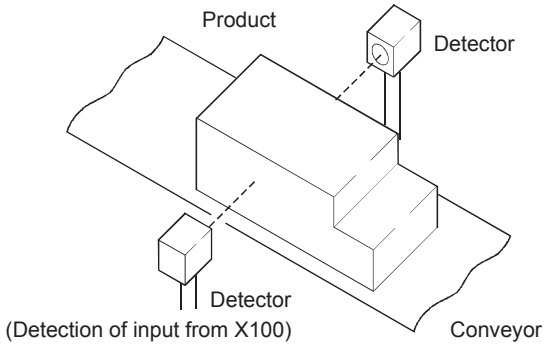


■Timing chart

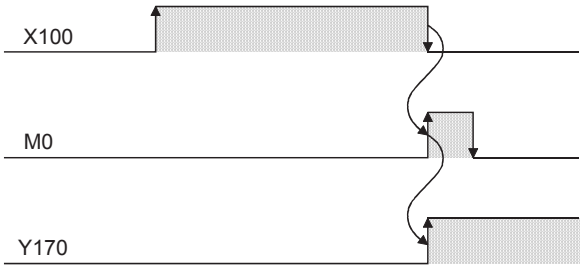


- These instructions can be used in a program that detects the passage of moving objects.
  The program detects that products have passed through and starts the next processing for the products.





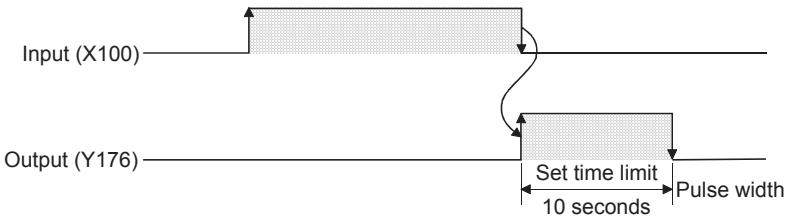
■Timing chart



Useful application of the PLS/PLF instructions (Part 1)

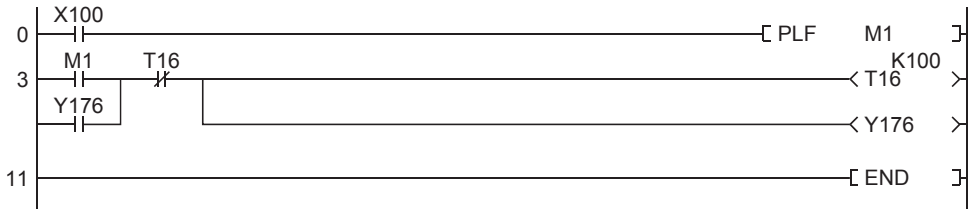
These instructions can be used in a program that executes an output operation for a set period of time at timing when an input signal turns on.

■Timing chart



■Program example

Project name	RB-6
Program name	MAIN

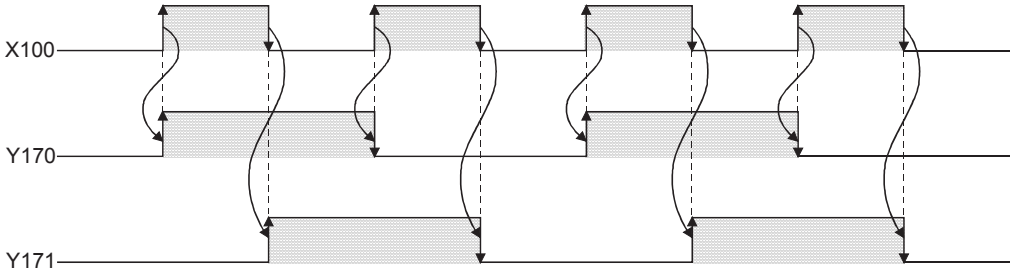


Useful application of the PLS/PLF instructions (Part 2)

These instructions can be used in a program that executes a repetitive operation such as switching the on/off status every time a push button switch (snap switch) is pressed.

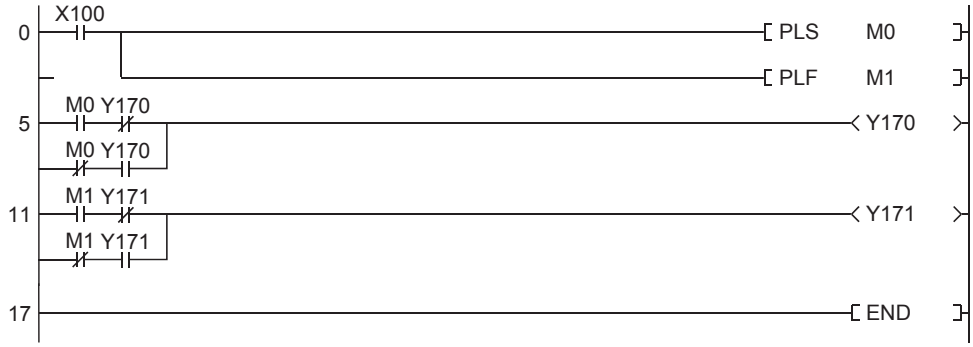
(If the PLS instruction is used in the program, the program is executed at the rising edge caused when the push button switch is pressed. If the PLF instruction is used, the program is executed at the falling edge caused when the switch is released.)

■Timing chart



■Program example

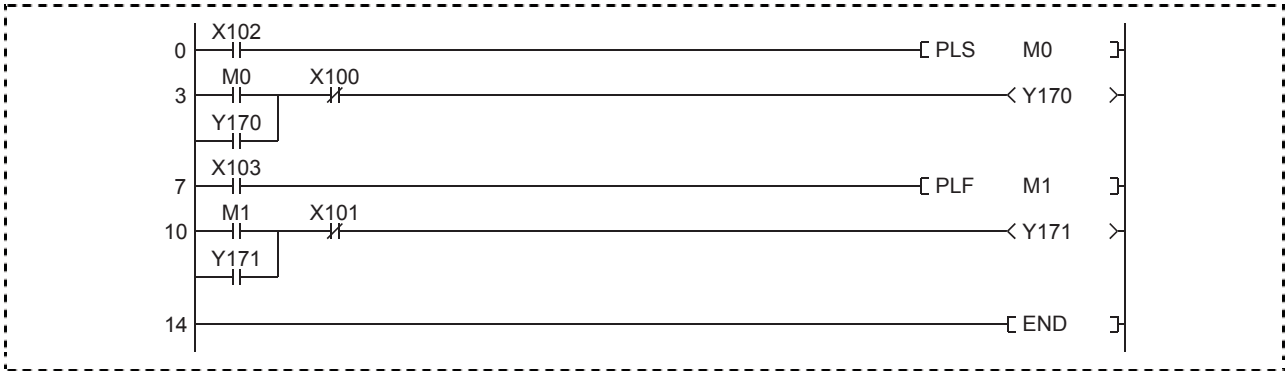
Project name	RB-7
Program name	MAIN



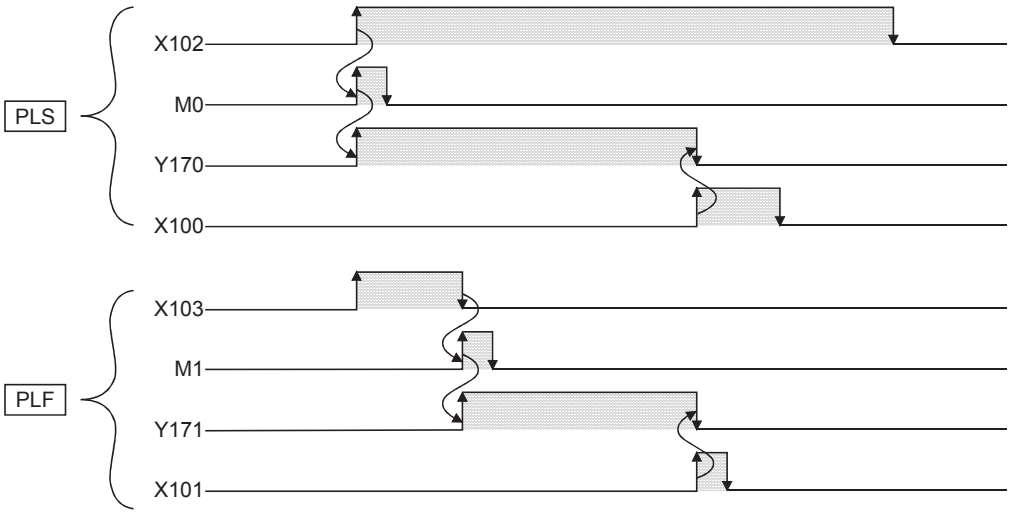
■Ladder example

Create the following ladder program and check that it operates properly.

Project name	REX2
Program name	MAIN

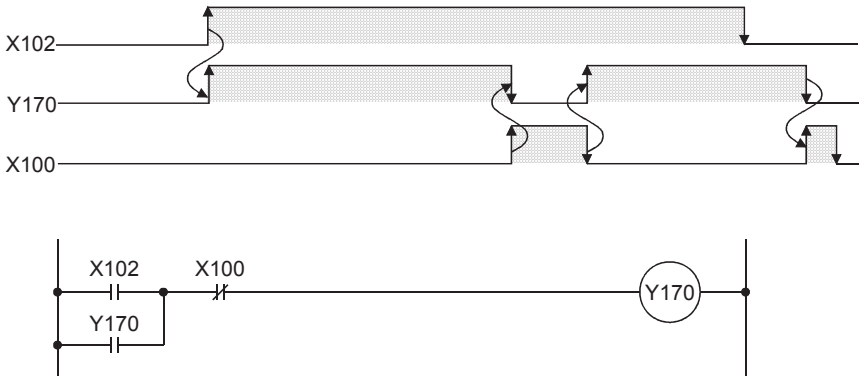


■Timing chart



Point

The following shows the timing chart of a self-holding ladder created with the OUT instruction. Compare this timing chart with the one of the self-holding ladder created with the PLS instruction.



Operating procedure

For the procedures of the following operations, refer to Section 4.4.

■Creating a new project

■Creating a program

■Writing data to the programmable controller

■Monitoring the ladder

Operation practice

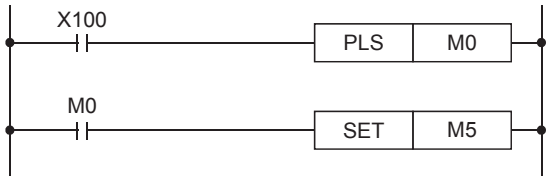
- Turning on X102 turns on Y170, and turning on X100 turns off Y170. (Even when X102 remains on, turning on X100 turns off Y170.)
- Turning off X103 turns on Y171, and turning on X101 turns off Y171.

Related exercise ---- Exercise 3

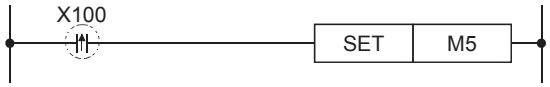
Point

The RCPU does not require input pulse processing because it uses derivation contacts (|↑|/|↓|).

[For the A/AnSCPU module]



[For the RCPU module]



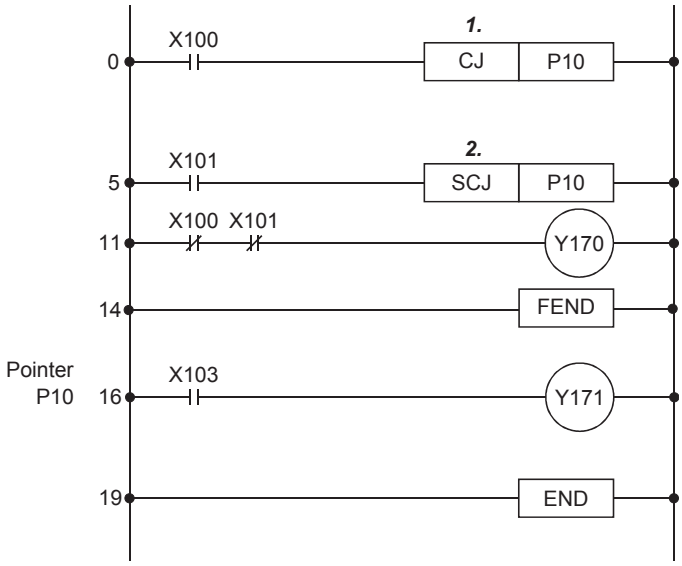
Applicable instructions are LDP, LDF, ANDP, ANDF, ORP, and ORF.

4.6 [CJ] (Conditional Jump of the Non-Delay Execution Type) [SCJ] (Conditional Jump Executed After One Scan)

Point

- This section describes that the programmable controller executes processing in every scan.
- This section describes how to use a pointer.

Project name	RB-10
Program name	MAIN

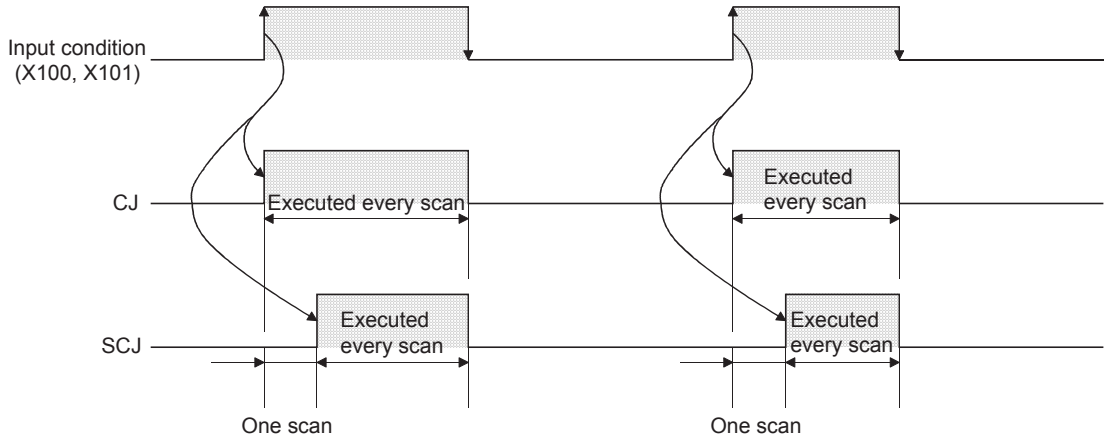


1. When the input condition is on, the CJ instruction instantaneously skips the processing and jumps to a specified destination (pointer number) in the same program file, and the subsequent processing is executed.
When the input condition is off, the instruction does not skip any processing.

2. When the input condition is on, the SCJ instruction executes the processing in the scan without skipping any processing.
From the next scan, the instruction skips the processing and jumps to a specified destination (pointer number) in the same program file, and the subsequent processing is executed.
When the input condition is off, the instruction does not skip any processing.

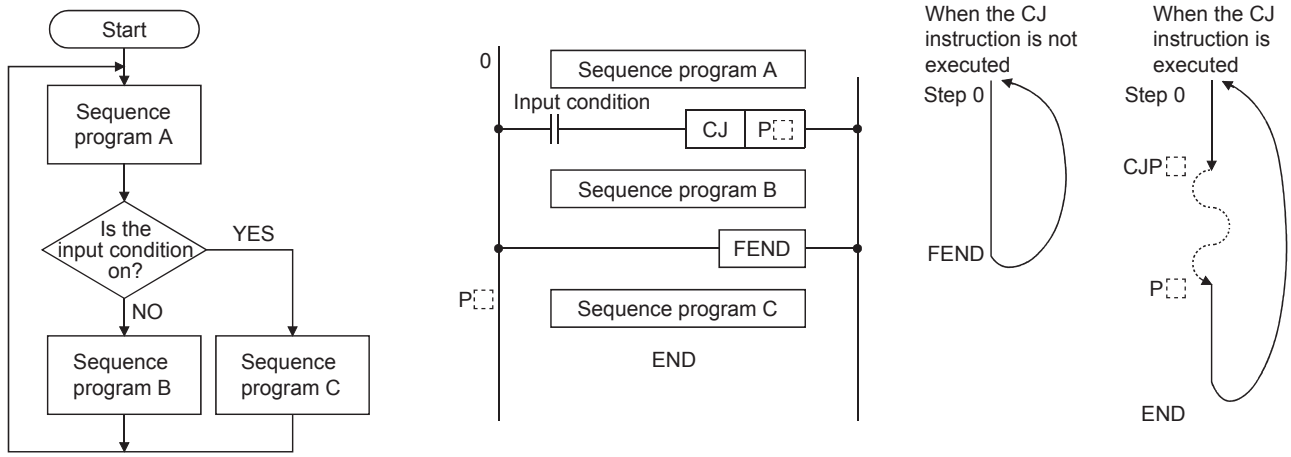
Use the SCJ instruction when some operations need be executed before skipping the program.
For example, use the instruction when an output needs to be turned on or reset in advance.

■Timing chart

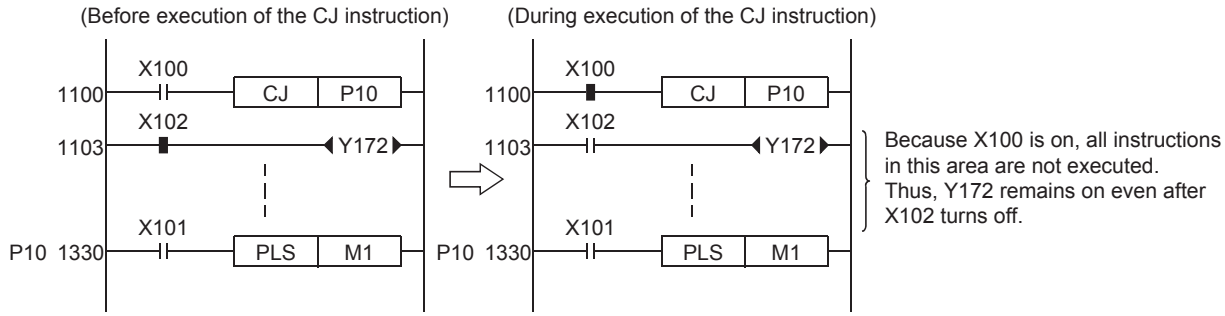


Precautions

- Both the CJ and SCJ instructions can use P0 to P4095 as pointer numbers.
- Use the FEND instruction as shown below to execute the processing with the CJ or SCJ instruction for every program block.



- The status of ladders skipped by the CJ instruction is the one before execution of the CJ instruction.



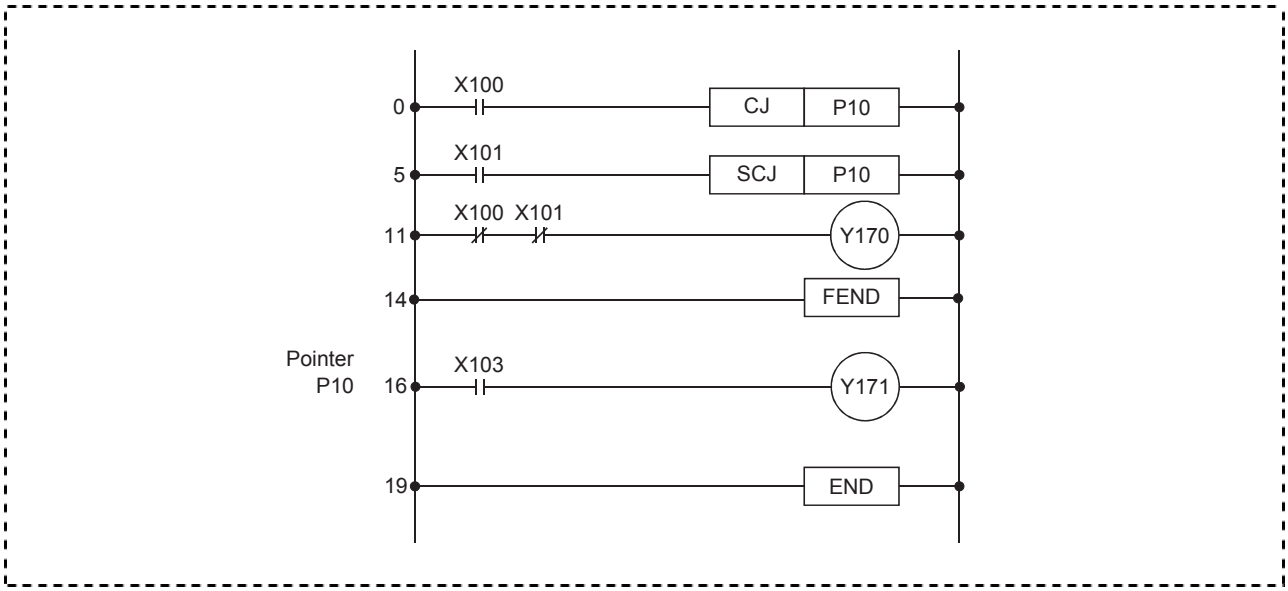
- If the CJ, SCJ, or JMP instruction is used to skip the timer with its coil that is on, the timer does not correctly measure the time.

Operand	Bit		Word			Double word		Indirect specification	Constant			Others (P)	Number of basic steps
	X, Y, M, L, SM, F, B, SB, FX, FY	J□\□	T, ST, C, D, W, SD,SW, FD, R, ZR, RD	U□\G□, J□\□, U3E□\(H)G□	Z	LT, LST, LC	LZ		K, H	E	$		
(P)	—	—	—	—	—	—	—	—	—	—	—	○	4

■Ladder example

Create the following ladder program with GX Works3 and write it to the CPU module of the demonstration machine. Then check the differences between the CJ instruction and the SCJ instruction.

Project name	REX4
Program name	MAIN



Operating procedure

For the procedures of the following operations, refer to Section 4.4.

■Creating a new project

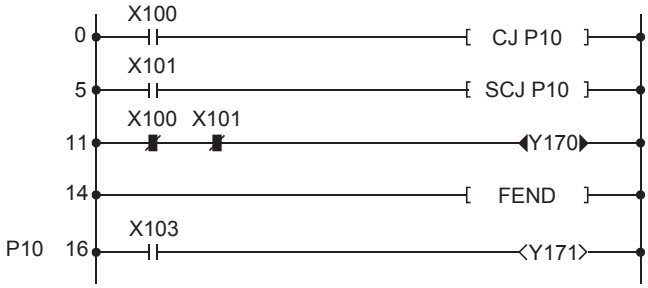
■Creating a program

■Writing data to the programmable controller

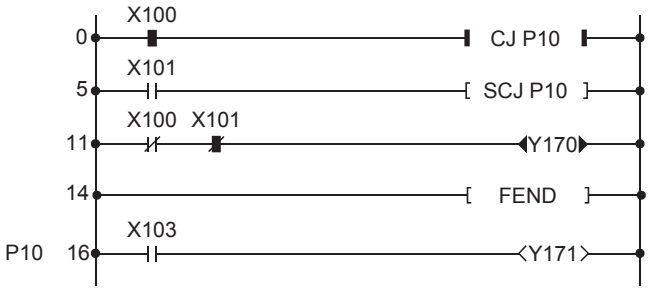
■Monitoring the ladder

Operation practice

(1) When X100 and X101 are off, the CJ and SCJ instructions are not executed.
Thus, Y170 is on.
[Before execution of the CJ/SCJ instruction]

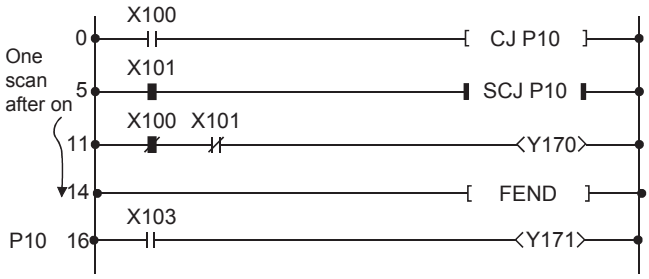


(2) When X100 turns on, the CJ instruction is executed and the processing is skipped and jumped to P10.
Thus, Y170 remains on.
[Execution of the CJ instruction] First scan and later

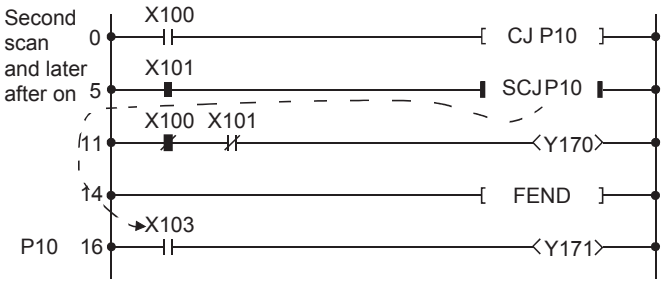


(3) When X100 turns off and X101 turns on, the SCJ instruction is executed and the processing is skipped and jumped to P10 in the second scan and later.
Thus, Y170 turns off.

[Execution of the SCJ instruction] First scan

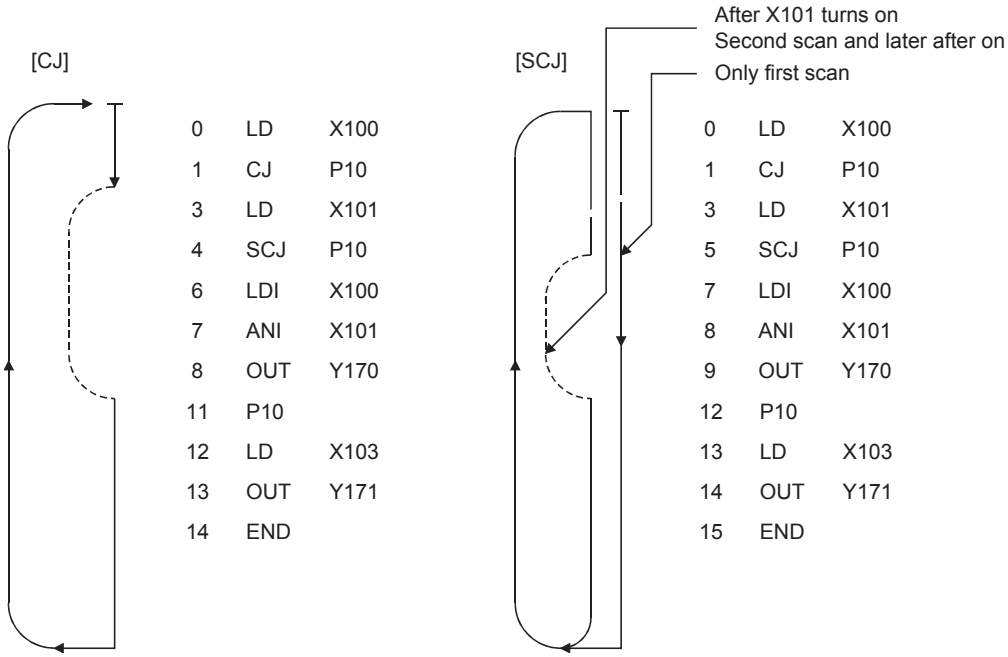


[Execution of the SCJ instruction] Second scan and later



(4) Y171 turns on or off when the CJ or SCJ instruction is executed.

- The following describes the differences between the CJ and SCJ instructions.



4.7 Exercise

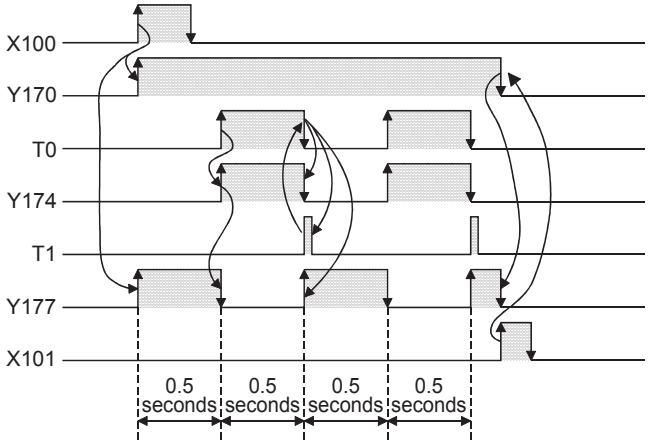
4.7.1 Exercise 1

Project name	RTEST1
Program name	MAIN

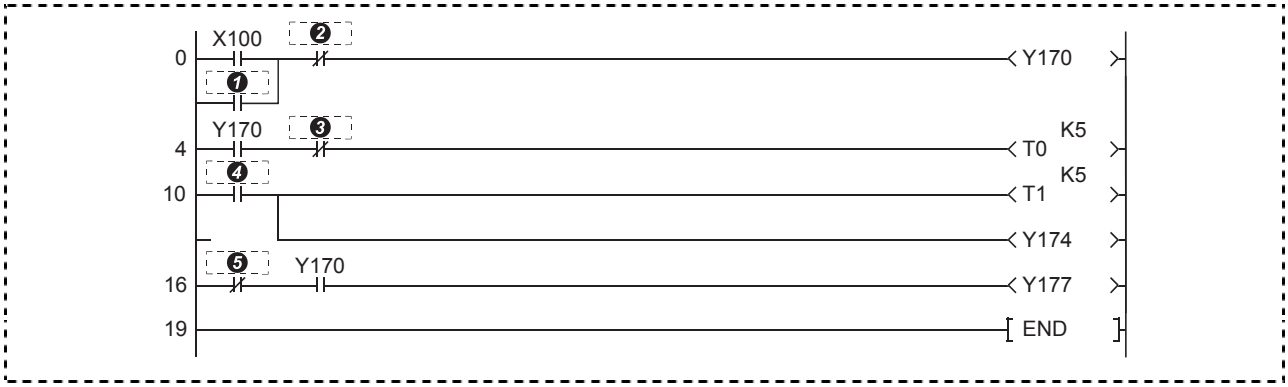
LD to NOP

When X100 turns on, Y170 is self-held, and Y174 and Y177 alternately flicker every 0.5 seconds.
When X101 turns on, Y170 turns off and flickering of Y174 and Y177 also stops.

■Timing chart



Fill in the blanks ( ) in the following program and create the program with GX Works3. Then, check the operation with the demonstration machine.
(For answers, refer to Page 4-34.)



❶ _____
❷ _____
❸ _____
❹ _____
❺ _____

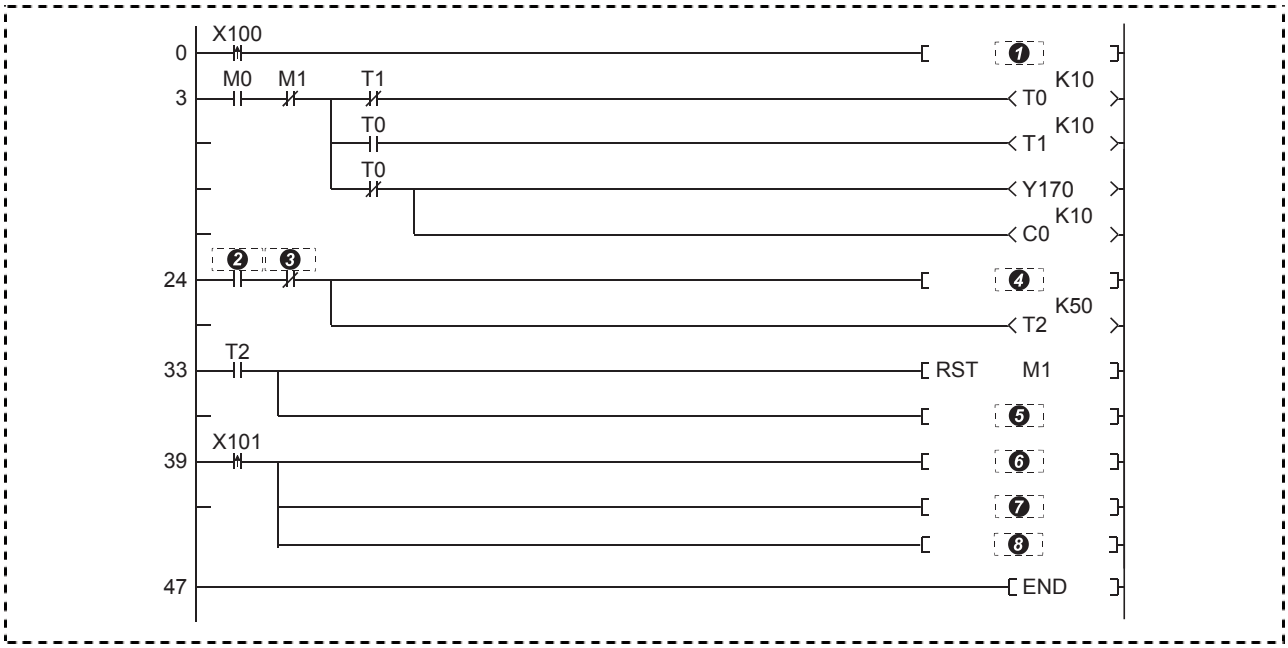
4.7.2 Exercise 2

Project name	RTEST2
Program name	MAIN

SET, RST

When X100 turns on, Y170 flickers at intervals of a one second. When Y170 flickers 10 times, it stops flickering for five seconds and restarts flickering.
Turning on X101 can stop flickering of Y170.

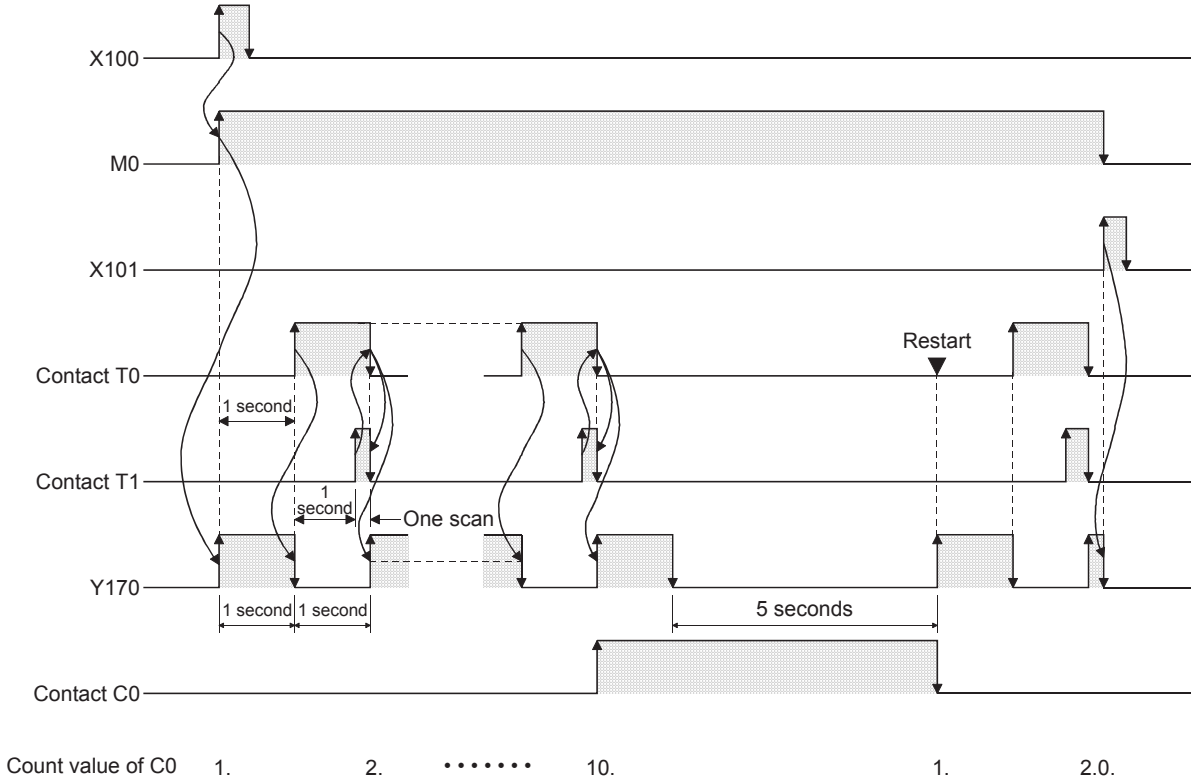
Fill in the blanks ( ) in the following program and create the program with GX Works3. Then, check the operation with the demonstration machine.
(For answers, refer to Page 4-34.)



❶ _____
❷ _____
❸ _____
❹ _____
❺ _____
❻ _____
❼ _____
❽ _____

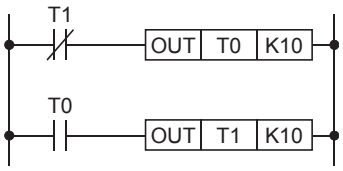
Hints

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

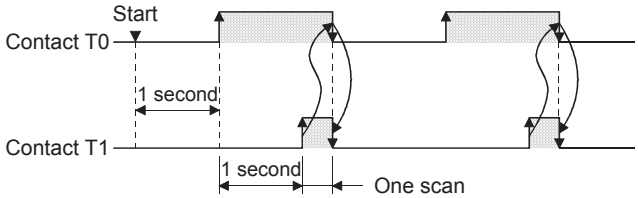


(2) The following shows the basic flicker ladder and its timing chart.

[Ladder]

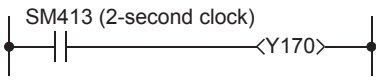


[Timing chart]

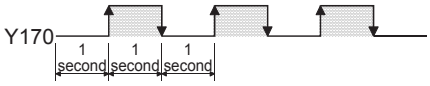


Point

• The flicker ladder can be created with a special relay that generates a clock as shown below.



[Timing chart]



Although the left ladder uses SM413 (2 second clock), the following special relay areas can also be used.

SM409 (0.01 second clock)
SM410 (0.1 second clock)
SM411 (0.2 second clock)
SM412 (1 second clock)
SM414 (2n second clock)
SM415 (2n ms clock)

The operation starts with the clock off when the system is powered on or the CPU module is reset.

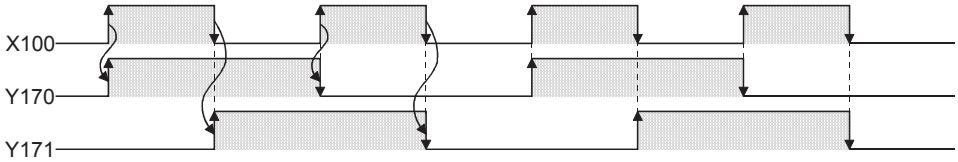
4.7.3 Exercise 3

Project name	RTEST3
Program name	MAIN

PLS, PLF

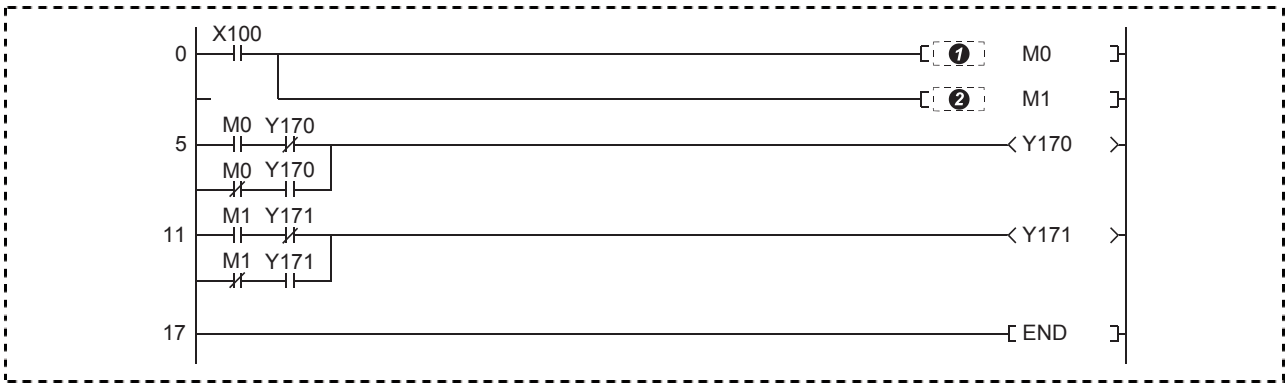
Y170 repeatedly turns on and off every time X100 turns on, and Y171 repeatedly turns on and off every time X100 turns off.

■Timing chart



Fill in the blanks ( ) in the following program and create the program with GX Works3. Then, check the operation with the demonstration machine.

(For answers, refer to Page 4-34.)



❶ _____

❷ _____

Answers for the exercises in Chapter 4

Exercise		Answer
1	❶	Y170
	❷	X101
	❸	T1
	❹	T0
	❺	Y174
2	❶	SET M0
	❷	C0
	❸	Y170
	❹	SET M1
	❺	RST C0
	❻	RST M0
	❼	RST C0
	❽	RST M1
3	❶	PLS
	❷	PLF

5 BASIC INSTRUCTIONS -PART 2-

5.1 Notation of Values (Data)

Point

- This section describes decimal, binary, and hexadecimal notations.
- This section describes a method of interconversion.

The programmable controller CPU converts all information into on or off signals (logical 1 or 0) to store and process them. Thus, the programmable controller executes numerical operations using the numerical values stored as logical 1 or 0 (binary numbers = BIN).
In daily life, decimal values are commonly used. Thus, the decimal-to-binary conversion or the binary-to-decimal conversion are required when values are read (monitored) or written from/to the programmable controller. The engineering tool and some instructions have the functions for those conversions.
This section describes how values (data) are expressed in decimal, binary, hexadecimal or binary-coded decimal notation (BCD), and how to convert values.

Decimal

- A decimal value consists of ten symbols, 0, 1, 2, 3, 4, 5, 6, 7, 8, and 9, which represent the order and size (amount). After a digit reaches 9, an increment resets it to 0, causing an increment of the next digit to the left.
- The following shows how a decimal value (in this case 153) is represented.

$$153 = 100 + 50 + 3$$
$$= 1 \times 100 + 5 \times 10 + 3 \times 1$$
$$= 1 \times 10^2 + 5 \times 10^1 + 3 \times 10^0$$

Decimal symbol (0 to 9)

"Power of digit"

"Power of digit" can be expressed as follows.

$_n$: Digit number (0, 1, 2 ...)

10: Decimal value

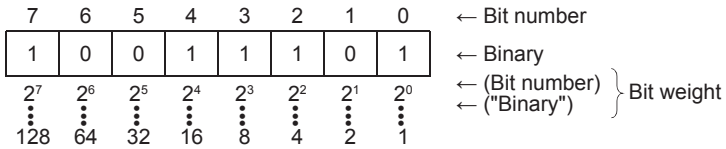
- In the MELSEC iQ-R series programmable controller, the symbol "K" is used to represent a value in decimal.

Binary (BIN)

- A binary value consists of two symbols, 0 and 1, which represent the order and size (amount). After a digit reaches 1, an increment resets it to 0, causing an increment of the next digit to the left. One digit of 0 or 1 is called a bit.

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

- The following example describes how to convert a binary value into a decimal value.
  "10011101"
  The following figure shows the binary value with bit numbers and binary bit weights.



  The binary value is broken as follows.
  = 1 × 128 + 0 × 64 + 0 × 32 + 1 × 16 + 1 × 8 + 1 × 4 + 0 × 2 + 1 × 1
  = 128 + 16 + 8 + 4 + 1
  = 157
  A binary value can be converted into a decimal value by the addition of the weight of each bit whose code is 1.

Hexadecimal

- A hexadecimal value consists of 16 symbols, 0 to 9 and A to F, which represent the order and size (amount). After a digit reaches F, an increment resets it to 0, causing an increment of the next digit to the left.

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
⋮	⋮	⋮
19101	4A9D	0100 1010 1001 1101

3	2	1	0	← Digit number
4	A	9	D	← Hexadecimal

$= (4) \times 16^3 + (A) \times 16^2 + (9) \times 16^1 + (D) \times 16^0$

$= 4 \times 4096 + 10 \times 256 + 9 \times 16 + 13 \times 1$

$= 19101$

"Power of digit"

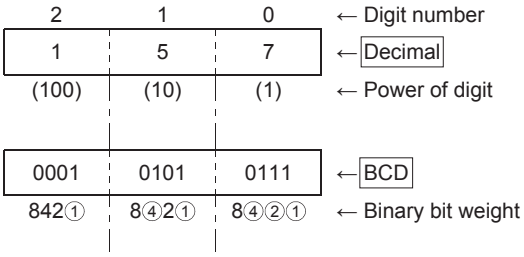
n •••• Digit number

16 •••••• Hexadecimal

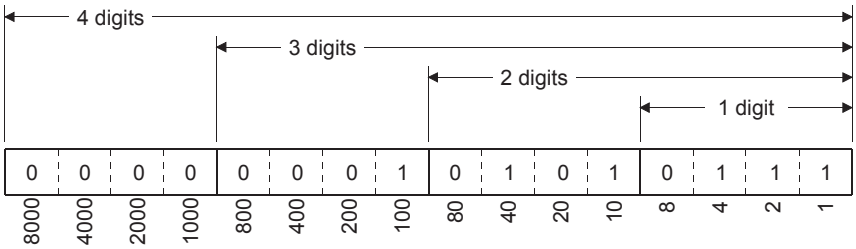
- Four bits of a binary value are equivalent to one digit of a hexadecimal value.
- In the MELSEC iQ-R series programmable controller, the symbol "H" is used to represent a value in hexadecimal.
- Hexadecimal values are used to represent the following device numbers.
  - Input and output (X, Y)
  - Function input and output (FX, FY)
  - Link relay (B)
  - Link register (W)
  - Link special relay (SB)
  - Link special register (SW)
  - Link direct device (Jn\X, Jn\Y, Jn\B, Jn\SB, Jn\W, Jn\SW)

Binary-coded decimal (BCD)

- The binary-coded decimal system uses a binary value to represent each digit of a decimal value.
  The decimal value 157, for example, is expressed as follows.



- In BCD, decimal values of 0 to 9999 (the maximum 4-digit value) can be represented with 16 bits.
  The following figure shows the weight of each bit in BCD.



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

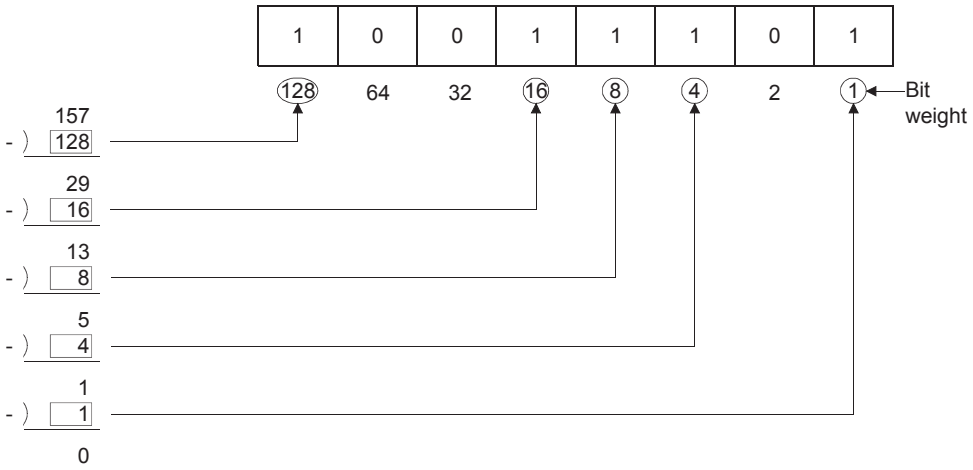
0	1	2	3	4	5	6	7	8	9
1(0) 2(0) 4(0) 8(0) COM	(1) (0) (0) (0)	(0) (1) (0) (0)	(1) (1) (0) (0)	(0) (0) (1) (0)	(1) (0) (1) (0)	(0) (1) (1) (0)	(1) (1) (1) (0)	(0) (0) (0) (1)	(1) (0) (0) (1)

BCD code digital switch

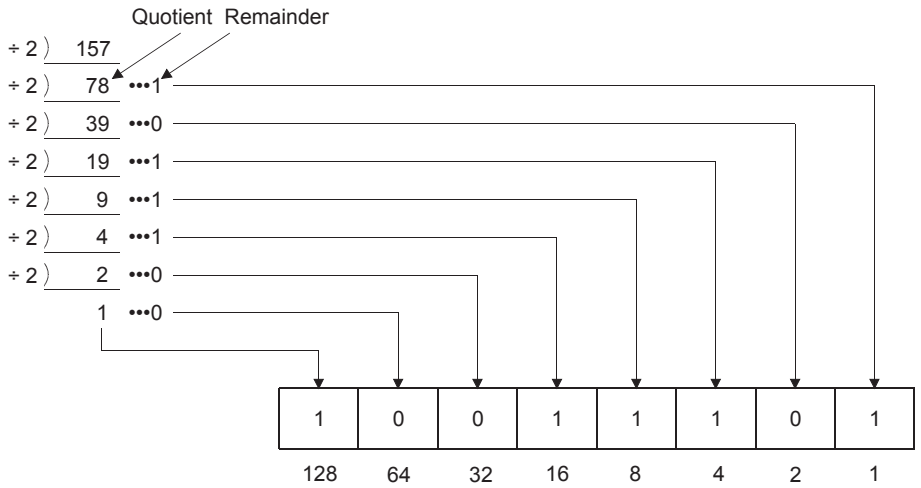
How to convert a decimal value into a binary value

Example) When the decimal value 157 is converted into a binary value

1)



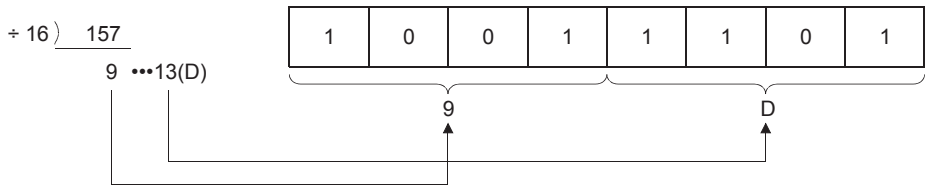
2)



How to convert a decimal value into a hexadecimal value

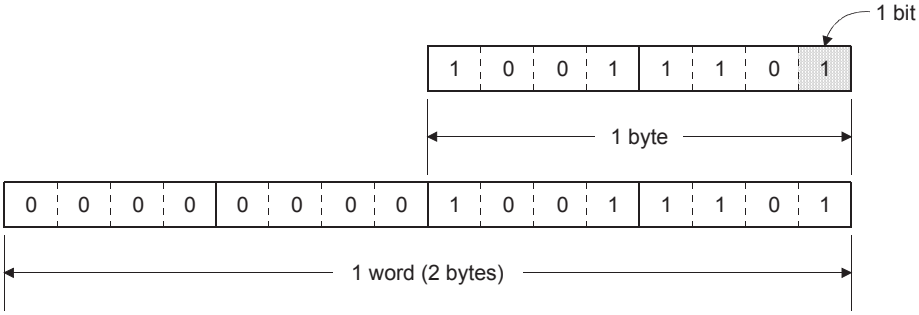
Example) When the decimal value 157 is converted into a hexadecimal value

1)

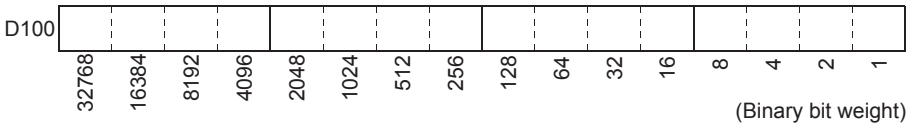


Numerical values used by the MELSEC iQ-R series programmable controller

- Usually, 8 bits are called one byte, and 16 bits (two bytes) are called one word.



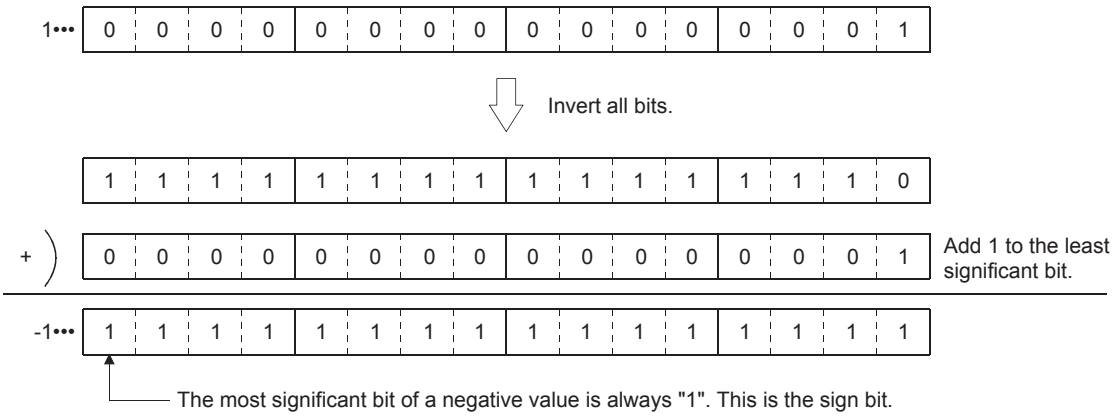
- Registers of each word device in the MELSEC iQ-R series programmable controller consist of 16 bits.
  - Data register (D)
  - Current value of a timer (T)
  - Current value of a counter (C)
  - File register (R)
  - Link register (W)



- Values in the following two ranges can be processed in 16 bits (one word).
  1) 0 to 65535
  2) -32768 to 0 to +32767
- The MELSEC iQ-R series programmable controller uses the range 2).
  A negative value uses the 2's complement against a positive number (1 to +32767).
- In the 2's complement, each binary bit is inverted, and 1 is added to the least significant bit.

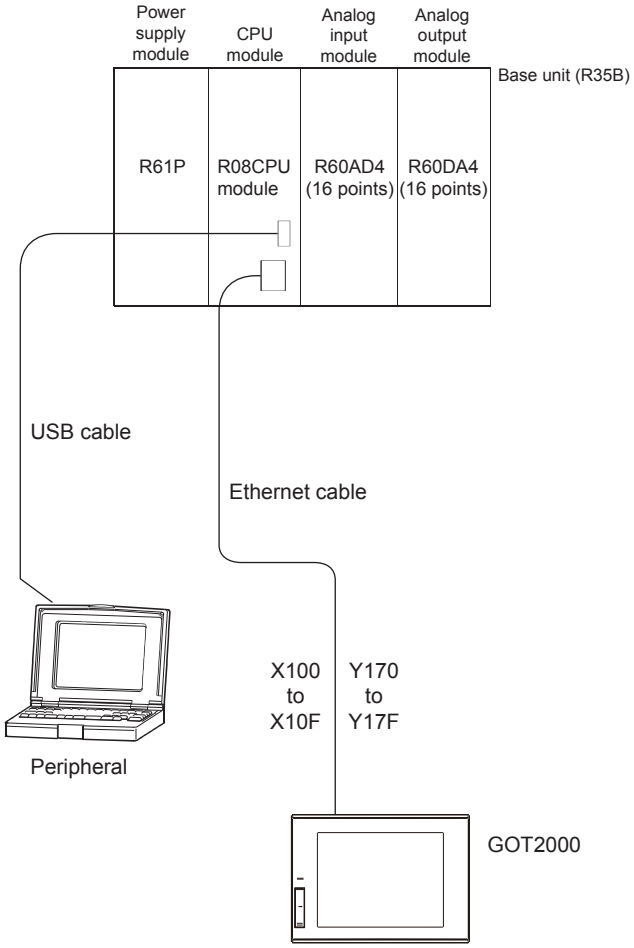
Ex.

How to calculate the 2's complement against 1

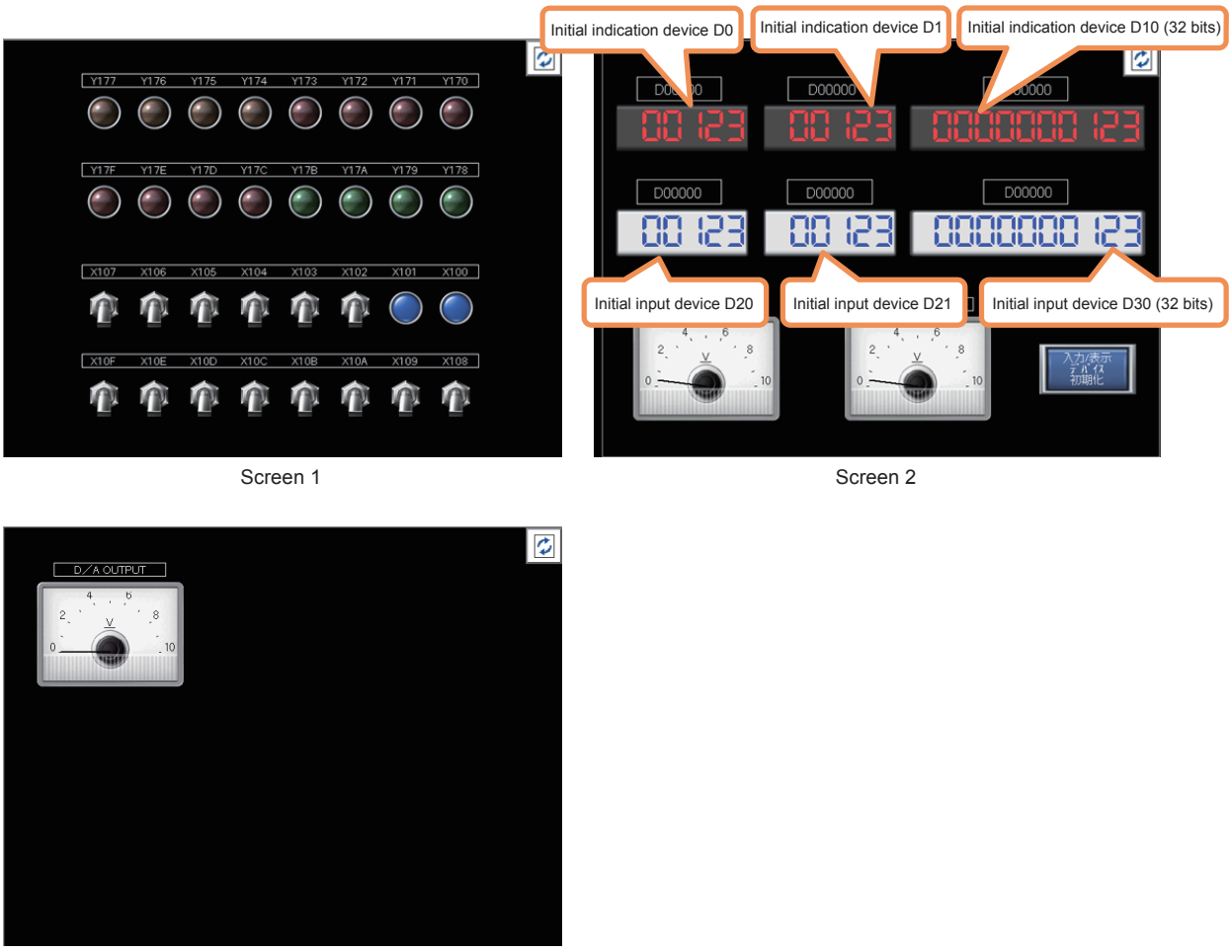


Binary-coded decimal (BCD)	Binary (BIN)	Decimal (K)	Hexadecimal (H)
00000000 00000000	00000000 00000000	0	0000
00000000 00000001	00000000 00000001	1	0001
00000000 00000010	00000000 00000010	2	0002
00000000 00000011	00000000 00000011	3	0003
00000000 00000100	00000000 00000100	4	0004
00000000 00000101	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

System configuration and I/O numbers of the demonstration machine



■GOT screen display



Screen 1

Screen 2

Screen 3

Upper section: The indication device can be changed.
Lower section: Data is displayed.

Upper section: The input device can be changed.
Lower section: The input data can be set and displayed.

- Touching switches the screen.
- The initial value is automatically set to a device number in the upper section (trigger action function).
- Touching the "Initialize Input/Indication Device" button also initializes the device number.

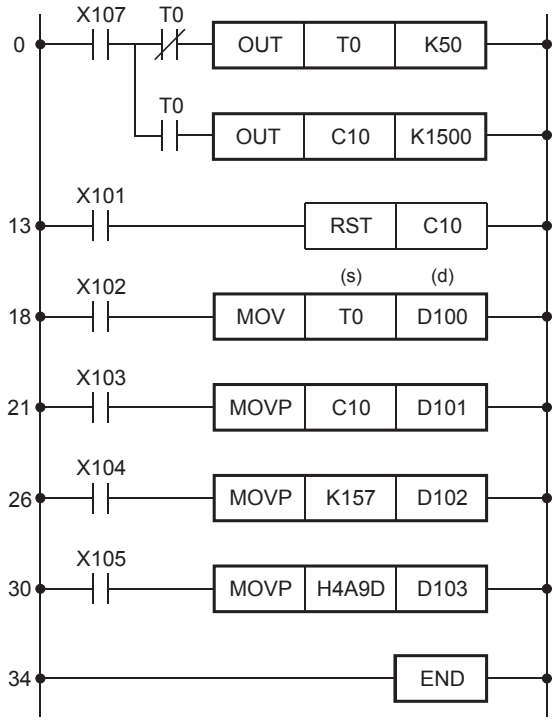
5.2 Transfer Instructions

5.2.1 [MOV(P)] (Transferring 16-bit data)

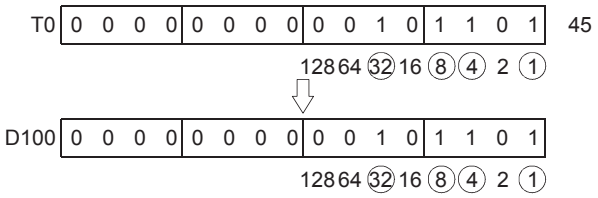
Point

- This section describes that data at the (s) side remains with the instruction for transferring data from the (s) side to the (d) side.
- This section describes the operation differences between the instructions with P and the one without P.

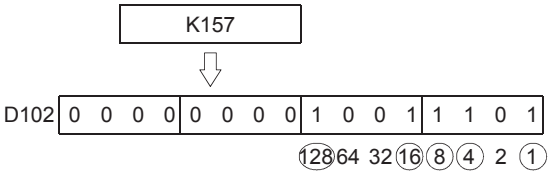
Project name	RB-11
Program name	MAIN



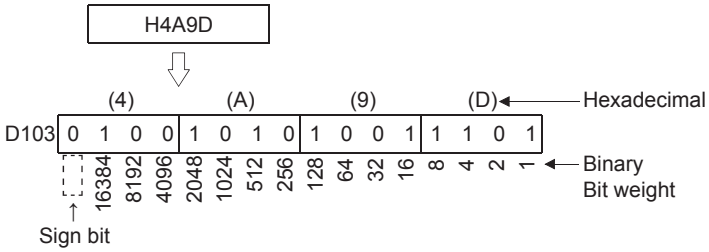
- When the input condition turns on, the current value of the timer T0 (source) is transferred into the data register D100 (destination).
- The current value of T0 in binary is transferred into D100 as it is. (Data conversion is not performed.)



- When the input condition turns on, the decimal number 157 is transferred into the data register D102. The decimal number (K) is automatically converted into a binary value, transferred to the data register D102, and stored there in binary.

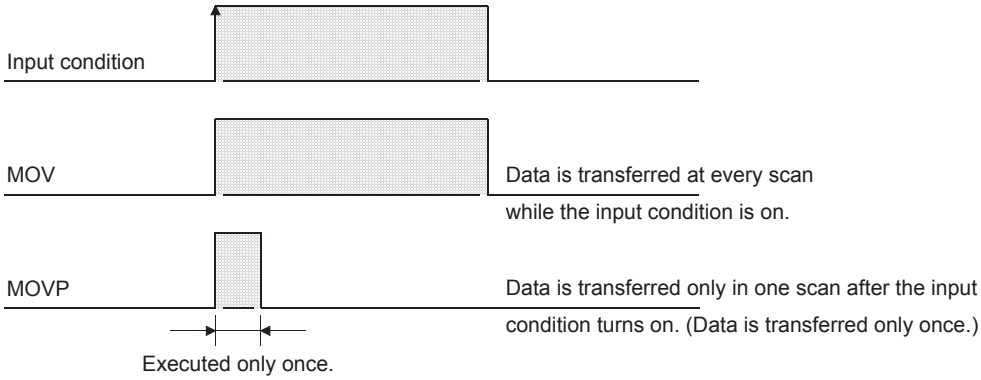


- When the input condition turns on, the hexadecimal value 4A9D is converted into a binary value and transferred into the data register D103.

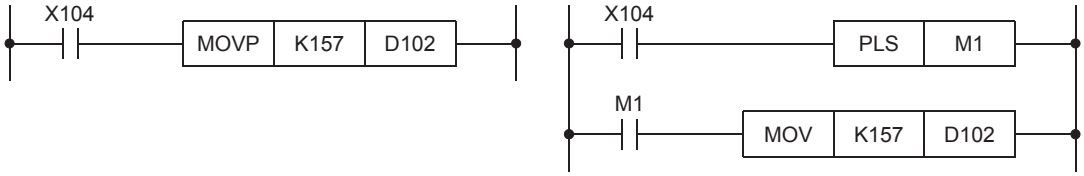


Differences between MOV and MOVP

The P in the MOVP instruction stands for a pulse.



- Use the MOV instruction to read changing data all the time.
  Use the MOVP instruction to instantaneously transfer data such as when setting data or reading data at the occurrence of an error.
- Both of the following ladder programs function similarly.



Operand	Bit		Word			Double word		Indirect specification	Constant			Others	Number of basic steps
	X, Y, M, L, SM, F, B, SB, FX, FY	J□\□	T, ST, C, D, W, SD,SW, FD, R, ZR, RD	U□\G□, J□\□, U3E□\(H)G□	Z	LT, LST, LC	LZ		K, H	E	$		
(s)	○	—	○	○	○	—	—	○	○	—	—	—	*1
(d)	○	—	○	○	○	—	—	○	—	—	—	—	

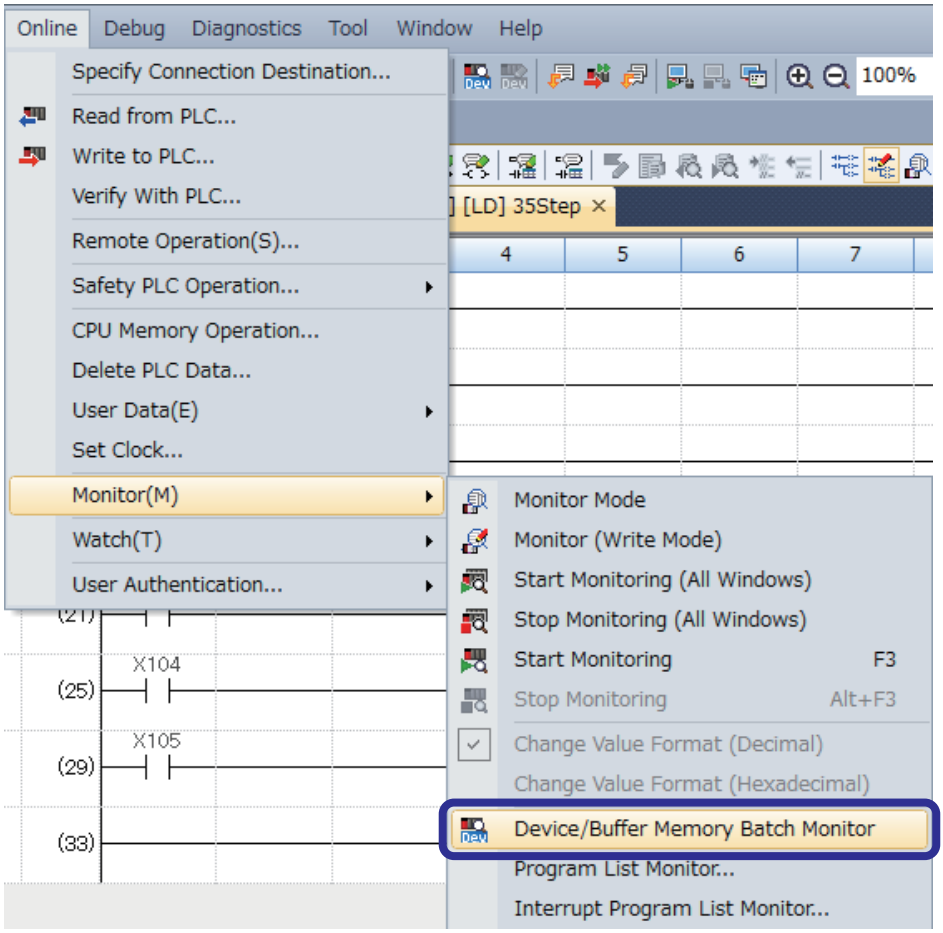
*1 The number of steps varies depending on the devices to be used.

Items to be checked

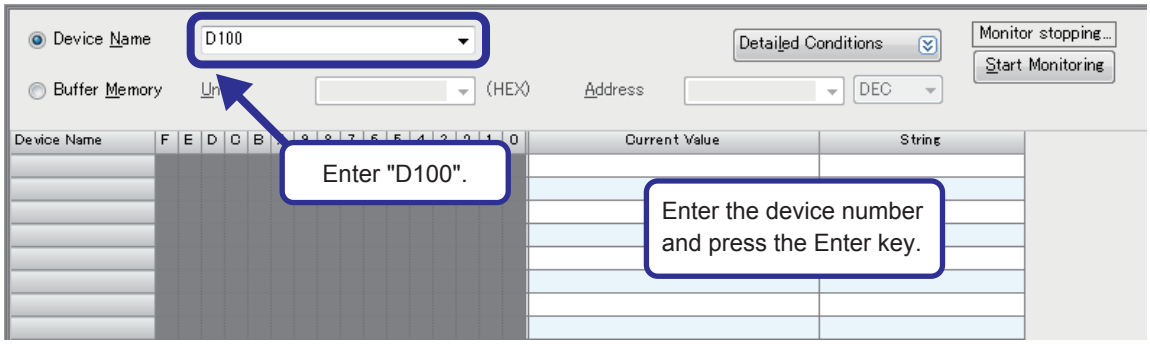
CPU module: RUN

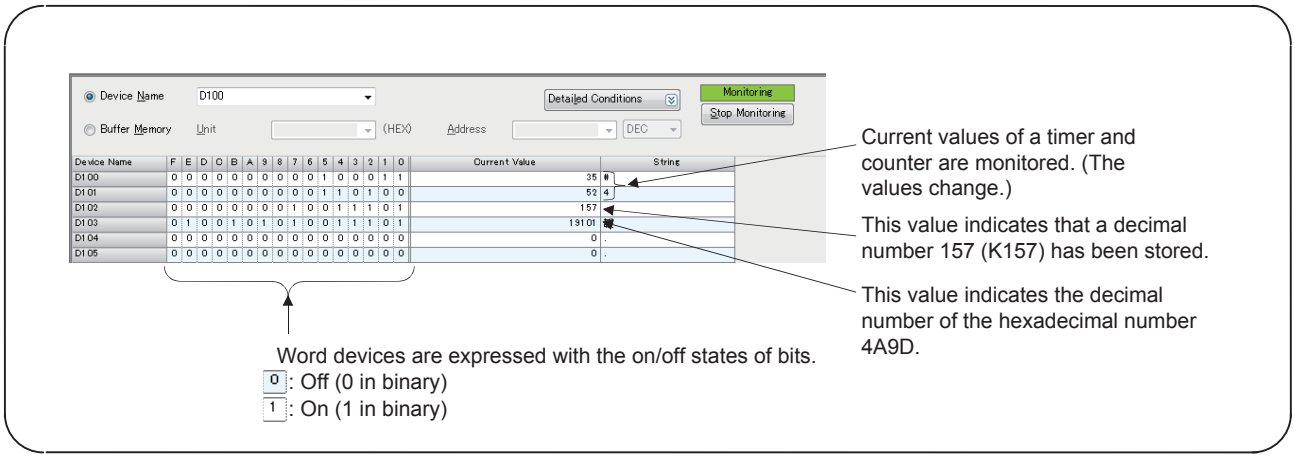
Inputs X102, X103, X104, X105, and X107: On

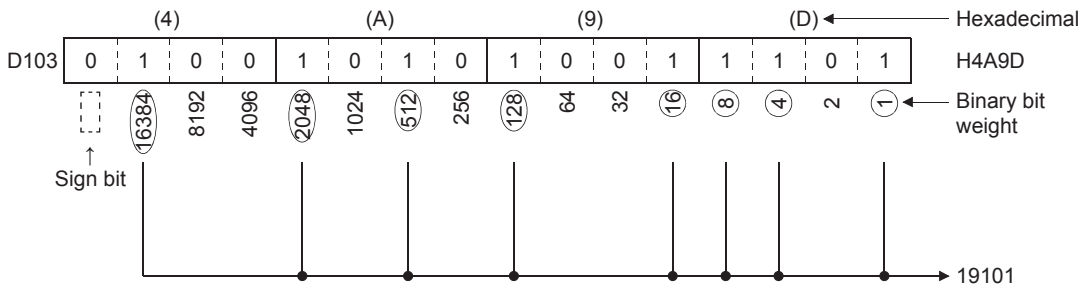
- Monitor the values in the data register (D100 to D103).
  After writing data to the programmable controller, click [Online] → [Monitor] → [Device/Buffer Memory Batch Monitor].
  The "Device/Buffer Memory Batch Monitor" dialog box appears.



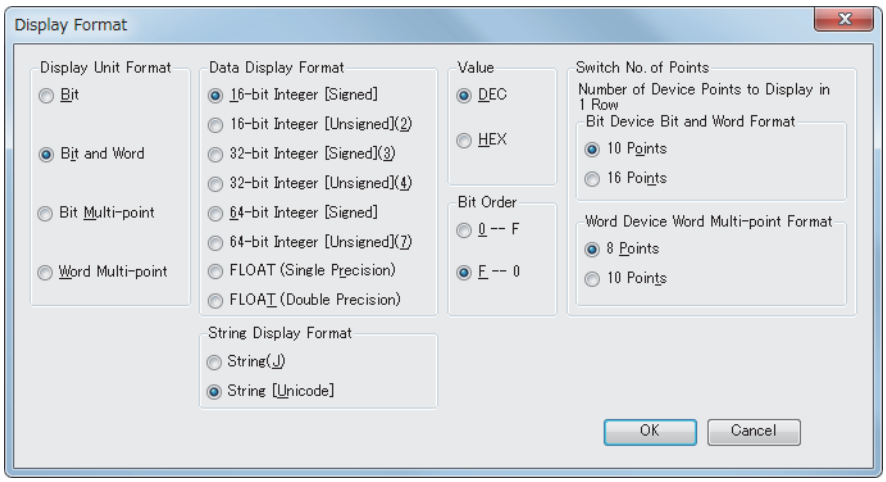
- Enter "D100" in "Device Name" and press the Enter key.



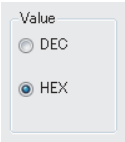

Device Name
D100
Detailed Conditions
Monitoring
Stop Monitoring
Buffer Memory
Unit
(HEX)
Address
DEC
Device Name
F E D C B A 9 8 7 6 5 4 3 2 1 0
Current Value
String
D100 0 0 0 0 0 0 0 0 0 0 1 0 0 0 1 1 35 #
D101 0 0 0 0 0 0 0 0 0 0 1 1 0 1 0 0 52 4
D102 0 0 0 0 0 0 0 0 1 0 0 1 1 1 0 1 157
D103 0 1 0 0 1 0 1 0 1 0 0 1 1 1 0 1 19101
D104 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
D105 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
Current values of a timer and counter are monitored. (The values change.)
This value indicates that a decimal number 157 (K157) has been stored.
This value indicates the decimal number of the hexadecimal number 4A9D.
Word devices are expressed with the on/off states of bits.
0 : Off (0 in binary)
1 : On (1 in binary)


(4)
(A)
(9)
(D)
Hexadecimal
D103
0 1 0 0 1 0 1 0 1 0 0 1 1 1 0 1
H4A9D
16384 8192 4096 2048 1024 512 256 128 64 32 16 8 4 2 1
Binary bit weight
Sign bit
19101

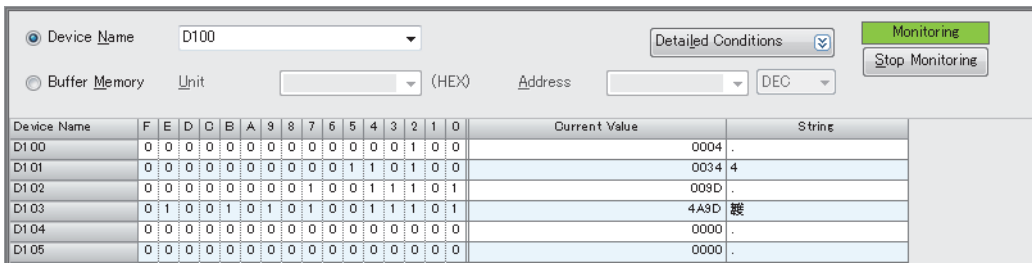
- Click on the toolbar or select [View] → [Display Format Detailed Setting] from the menu.
  The "Display Format" dialog box appears.



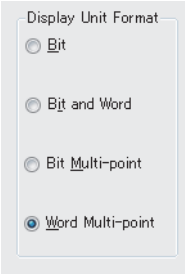
- Change the display of the numerical values being monitored to the hexadecimal notation.
  Select "HEX" for "Value" in the "Device Format" dialog box.



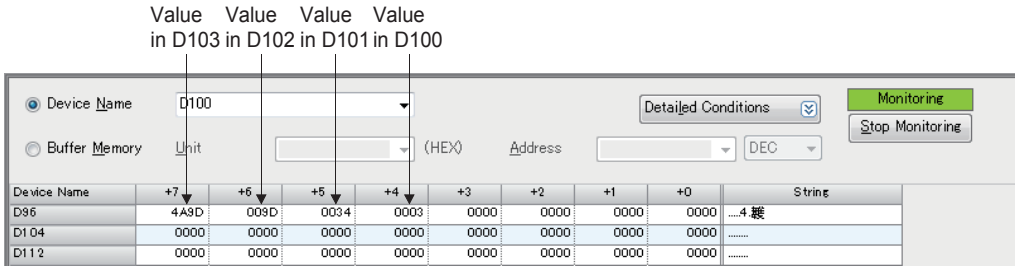
["Device/Buffer Memory Batch Monitor" window]



- Change the display of the numerical values being monitored in a multi-point format.
  Select "Word Multi-point" for "Display Unit Format" in the "Device Format" dialog box.



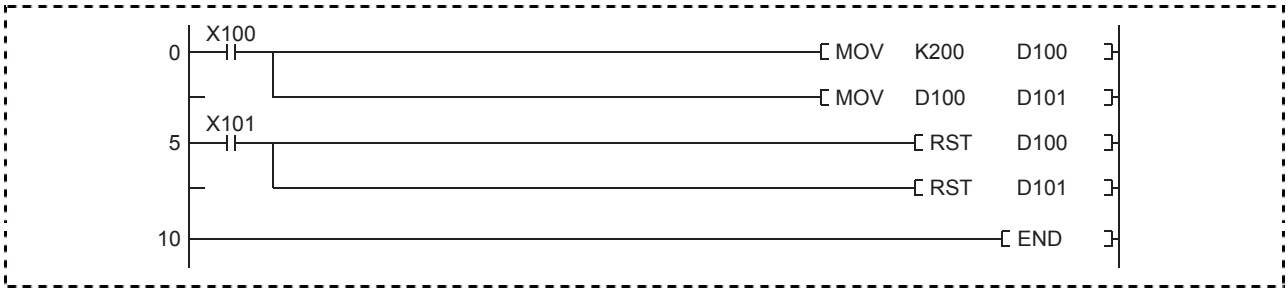
["Device/Buffer Memory Batch Monitor" window]



■Ladder example

Create the following ladder program with GX Works3 and write it to the CPU module of the demonstration machine. Then, check the execution of the MOV instruction.

Project name	REX7
Program name	MAIN



Operating procedure

For the procedures of the following operations, refer to Section 4.4.

■Creating a new project

■Creating a program

■Writing data to the programmable controller

■Monitoring the ladder

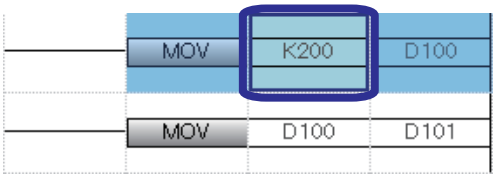
■How to modify the transfer instruction

To modify the transfer instruction, follow the procedure below.

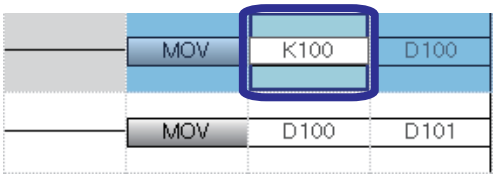
Ex.

Change the transfer data K200 of [MOV K200 D100] to K100.

1. Select the instruction to be modified.



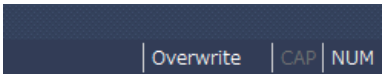
2. Press the F2 key and modify K200 to K100.



3. Press the Enter key.

(All data in -[ ]- can be modified with the above operation method.

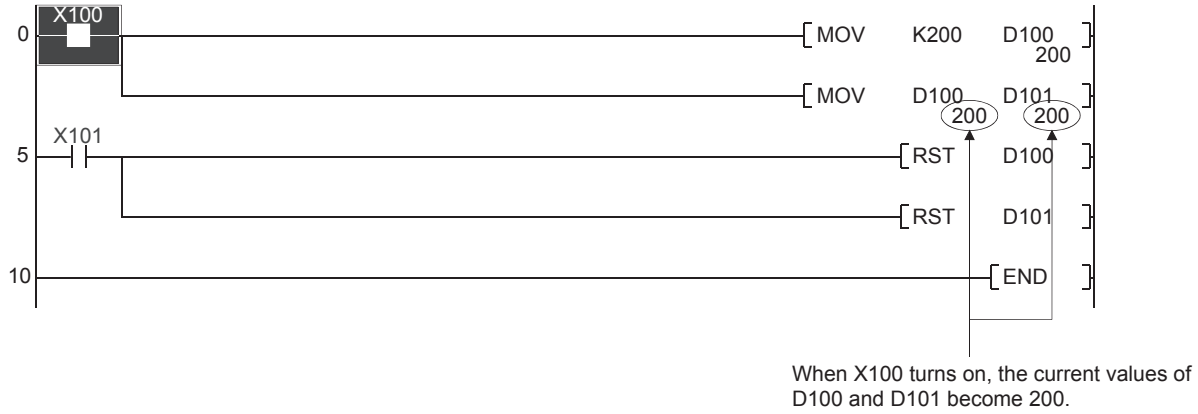
When "Insert" is displayed, however, press the Insert key to change it to "Overwrite" before the modification.)



4. When the modification is completed, click [Convert] → [Convert] from the menu.

■Operation practice

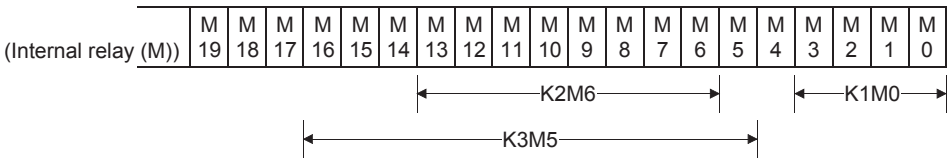
Check that turning on X100 of the operation panel of the demonstration machine changes the values of D100 and D101 to 200 on the monitor window.



■Related exercise ---- Exercise 3

K1M0

- A word device D (data register), T (current value of a timer), or C (current value of a counter) consists of 16 bits (one word), and data is basically transferred in one device.
- With 16 bit devices (such as X, Y, and M), data of the same size as a word device can be handled. The device numbers allocated to the bit devices must be in consecutive order.
- Bit devices can process data in units of four points.
- Other bit devices can also process data in the same way.



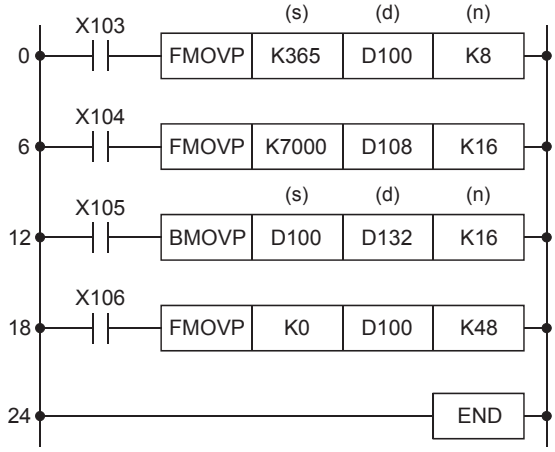
- As long as the device numbers of four bit device areas are in consecutive order, any bit device can be specified as the start device.

5.2.2 [FMOV(P)] (Transferring the same data in a batch) [BMOV(P)] (Transferring block data in a batch)

Point

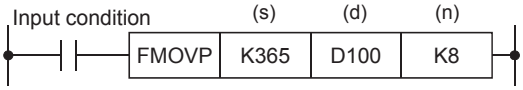
- This section describes transfer instructions that handle multi-point data.
- This section describes how to use batch monitoring.

Project name	RB-14
Program name	MAIN



Operation explanation

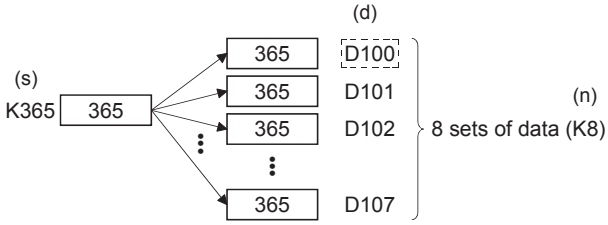
■FMOV



- When the input condition turns on, the FMOV instruction transfers data in the device specified by (s) to the (n) points of device areas starting from the device specified by (d).

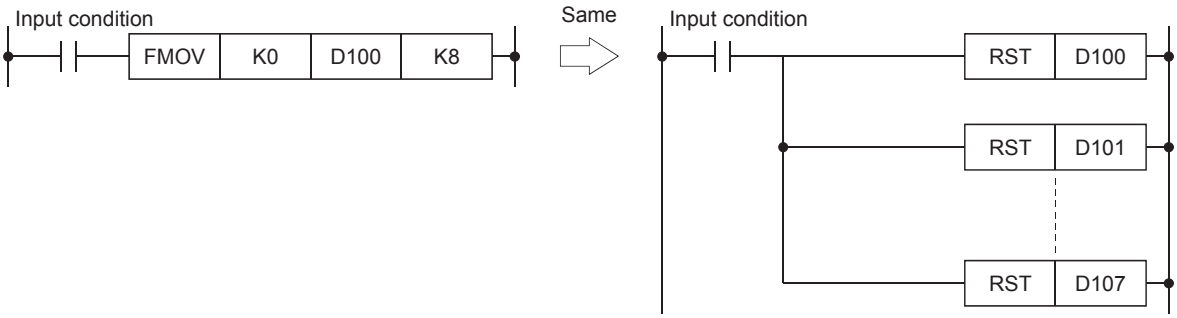
Ex.

The following figure shows the operation of when X103 turns on and the FMOV instruction is executed.



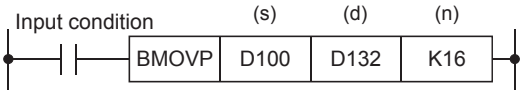
- The FMOV instruction is useful for clearing many data sets in a batch.

Ex.



The FMOV instruction substitutes the multiple RST instructions as shown above.

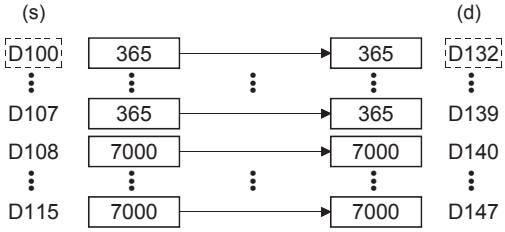
■BMOV



- When the input condition turns on, the BMOV instruction batch-transfers the (n) points of data stored starting from the device specified by (s) to the areas starting from the device specified by (d).

Ex.

The following figure shows the operation of when X105 turns on and the BMOV instruction is executed.



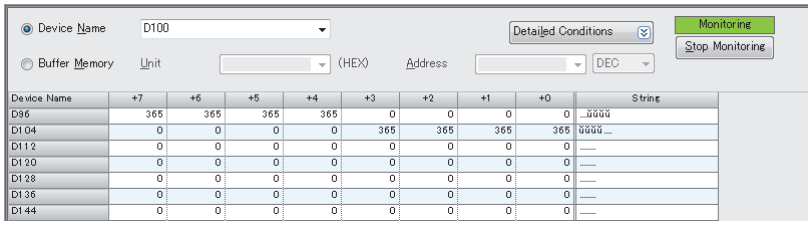
- The BMOV instruction is useful for the following applications.
  - Saving logging data in files
  - Saving important data (such as automatic operation data and measurement data) into the latch areas, for example the data register set for backing up data at power-off in parameter, to prevent data loss at an unintended power failure

Operand	Bit		Word			Double word		Indirect specification	Constant			Others	Number of basic steps
	X, Y, M, L, SM, F, B, SB, FX, FY	J□\□	T, ST, C, D, W, SD,SW, FD, R, ZR, RD	U□\G□, J□\□, U3E□\(H)G□	Z	LT, LST, LC	LZ		K, H	E	$		
(s)	○	—	○	○	—	○	—	○	—	—	—	—	4
(d)	○	—	○	○	—	○	—	○	—	—	—	—	
(n)	○	—	○	○	○	—	—	○	○	—	—	—	

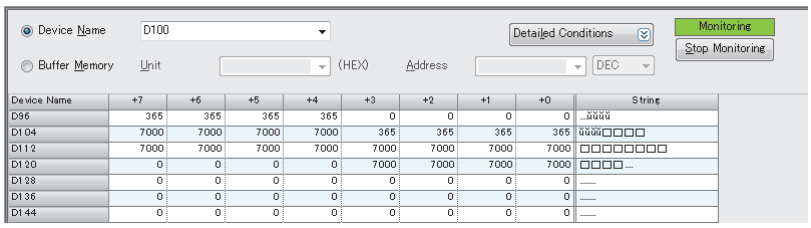
■Operation practice

- Write the program on the previous page to the CPU module, and set the operating status of the CPU module to RUN.
- Follow the procedure below to execute the device batch monitor. Values in D100 to D147 can be monitored.
- After writing data to the programmable controller, click [Online] → [Monitor] → [Device/Buffer Memory Batch Monitor].
  Enter "D100" in "Device Name" of the "Device/Buffer Memory Batch Monitor" dialog box and press the Enter key.
- Click "Display Format Detailed Setting" ( ) to display the "Display Format" dialog box.
  Select "Word Multi-point" for "Display Unit Format".
  → Click the [OK] button.

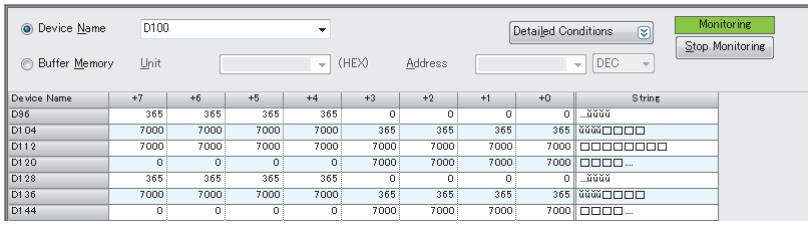
■Monitor window



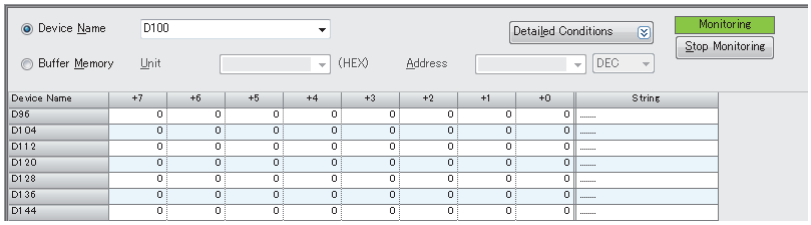
1. Turn on X103.
The numerical value 365 is batch-transferred to the eight consecutive device areas D100 to D107.



2. Turn on X104.
The numeric value 7000 is batch-transferred to 16 consecutive device areas D108 to D123.



3. Turn on X105.
The values in 16 consecutive device areas D100 to D115 are batch-transferred to 16 consecutive device areas D132 to D147.

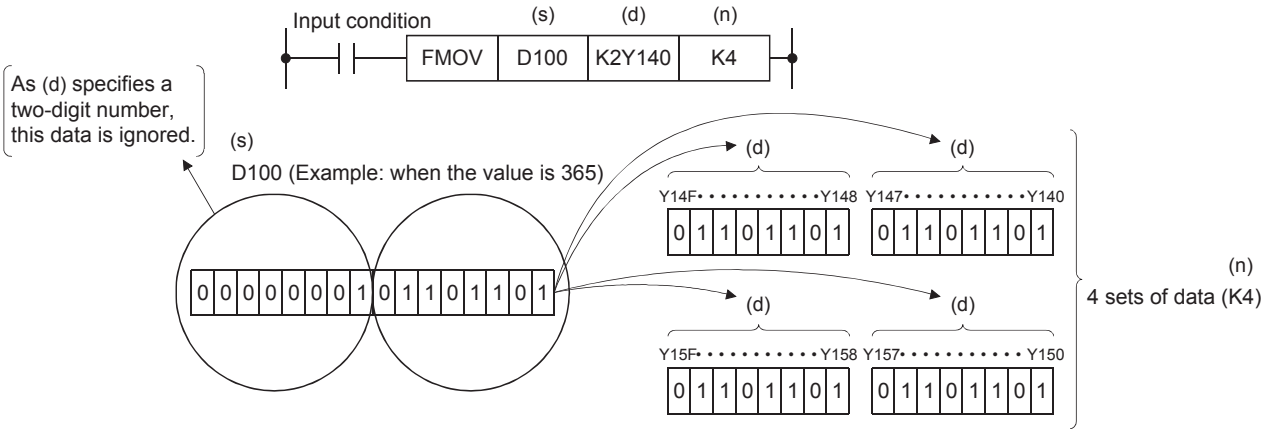


4. Turn on X106.
The value "0" is batch-transferred to all the 48 consecutive device areas D100 to D147.
This processing clears all the 48 register areas to 0.

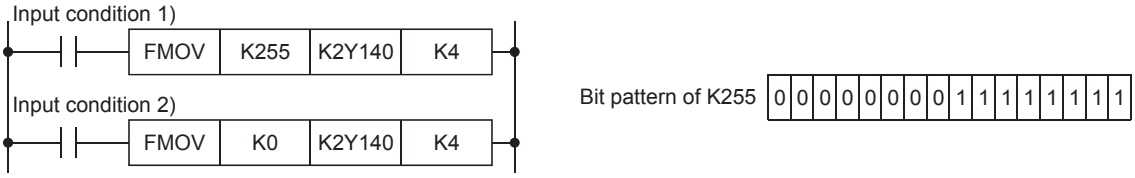
Reference

- If (d) is a bit device, the operation becomes as follows.

■FMOV instruction



- Among the device areas Y140 to Y15F, the device areas where "1" is stored output.
- In the following program, turning on the input condition 1) turns on all the outputs Y140 to Y15F, or turning on the input condition 2) turns off them.



- In units of four bits,

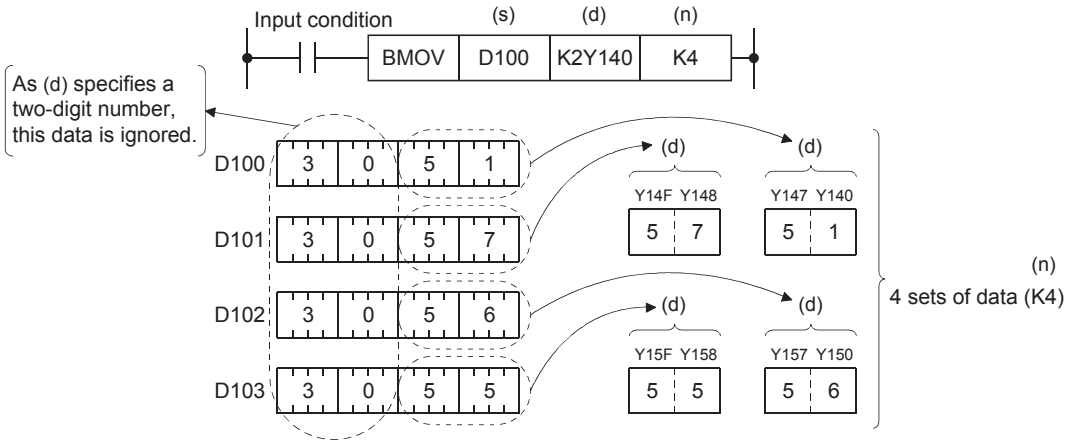
  To turn off 16 bit device areas or less → MOV instruction Example | MOV | K0 | K4M0 |

  To turn off 32 bit device areas or less → DMOV instruction Example | DMOV | K0 | K8M0 |

  To turn off more than 32 bit device areas → FMOV instruction Example | FMOV | K0 | K4M0 | K4 |

  (The FMOV instruction turns off 64 bit device areas.)

■BMOV instruction

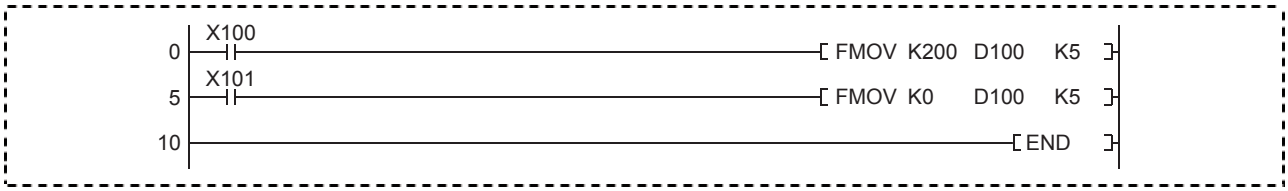


- In the example above, product codes (16 bits) are stored in the device areas D100 to D103. The BMOV instruction is useful for displaying and monitoring only the lower two digits of the codes representing their types.

■Ladder example

Create the following ladder program with GX Works3 and write it to the CPU module of the demonstration machine. Then, check the execution of the FMOV instruction.

Project name	REX9
Program name	MAIN



Operating procedure

For the procedures of the following operations, refer to Section 4.4.

■Creating a new project

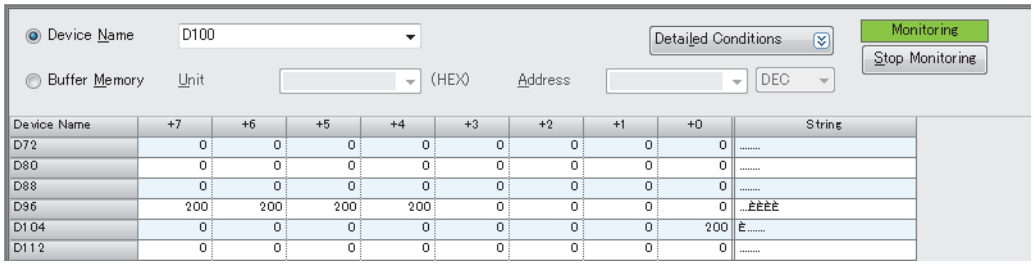
■Creating a program

■Writing data to the programmable controller

■Monitoring the ladder

■Operation practice

Check that turning on X100 on the operation panel of the demonstration machine changes the values of D100 to D104 to 200 on the batch monitor window. The data is cleared when X101 turns on.



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

- Value: DEC •••••••••••••••••••• This setting displays values in decimal.
- Value: HEX •••••••••••••••••••• This setting displays values in hexadecimal.
- Monitor Format: Bit Multi-point •••••••••• This setting displays values in binary.

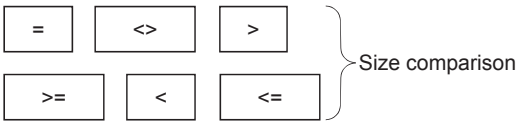
■Related exercise ---- Exercise 4

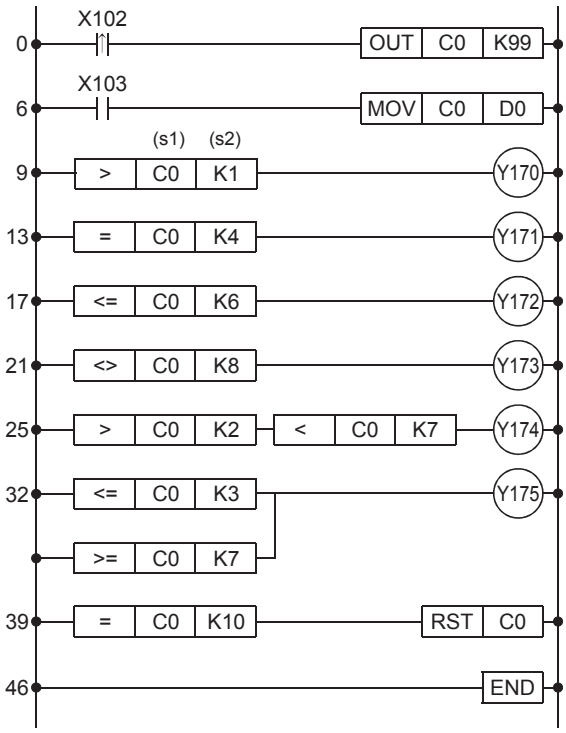
5.3 Comparison Operation Instructions

Point

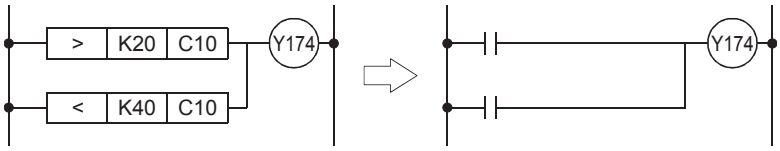
This section describes how to compare numerical values.

Project name	RB-15
Program name	MAIN





- The comparison instruction compares the value in the source 1 (s1) and that in the source 2 (s2), and brings the devices in the continuity state when conditions are satisfied.
- The instruction can be regarded as one normally open contact (⊣⊢) because it goes in the continuity state only when conditions are satisfied.



- [= | (s1) | (s2)] ••••• The ladder goes in the continuity state when the value in the source 1 is equal to that in the source 2.
- [< | (s1) | (s2)] ••••• The ladder goes in the continuity state when the value in the source 1 is smaller than that in the source 2.
- [> | (s1) | (s2)] ••••• The ladder goes in the continuity state when the value in the source 1 is larger than that in the source 2.
- [<= | (s1) | (s2)] ••••• The ladder goes in the continuity state when the value in the source 1 is equal to or smaller than that in the source 2.
- [>= | (s1) | (s2)] ••••• The ladder goes in the continuity state when the value in the source 1 is equal to or larger than that in the source 2.
- [<> | (s1) | (s2)] ••••• The ladder goes in the continuity state when the value in the source 1 are not equal to that in the source 2.

■Operation practice

- Write the program to the CPU module.
- C0 counts the number of times that X102 turns on.
- Turning on X103 displays the current value of C0 in the initial indication device (D0).
- Check that the device areas Y170 to Y175 turn on as follows.

[The range in which Y170 to Y175 turn on]

C0	Y175	Y174	Y173	Y172	Y171	Y170
0	●	—	●	●	—	—
1	●	—	●	●	—	—
2	●	—	●	●	—	●
3	●	●	●	●	—	●
4	—	●	●	●	●	●
5	—	●	●	●	—	●
6	—	●	●	●	—	●
7	●	—	●	—	—	●
8	●	—	—	—	—	●
9	●	—	●		—	●
10	RST C0					

●: On —: Off

- The counter is designed to be reset every time the value reaches 10.
- In this way, the comparison instructions not only compare data but also specify the range. These instructions are commonly used in the program to determine the acceptances of products.

Operand	Bit		Word			Double word		Indirect specification	Constant			Others	Number of basic steps
	X, Y, M, L, SM, F, B, SB, FX, FY	J□\□	T, ST, C, D, W, SD,SW, FD, R, ZR, RD	U□\G□, J□\□, U3E□\(H)G□	Z	LT, LST, LC	LZ		K, H	E	$		
(s1)	○	○	○	○	○	○*1	○*1	○	○	—	—	—	3
(s2)	○	○	○	○	○	○*1	○*1	○	○	—	—	—	

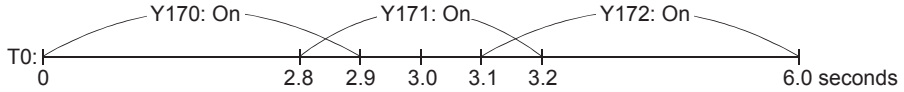
*1 For 32-bit binary data

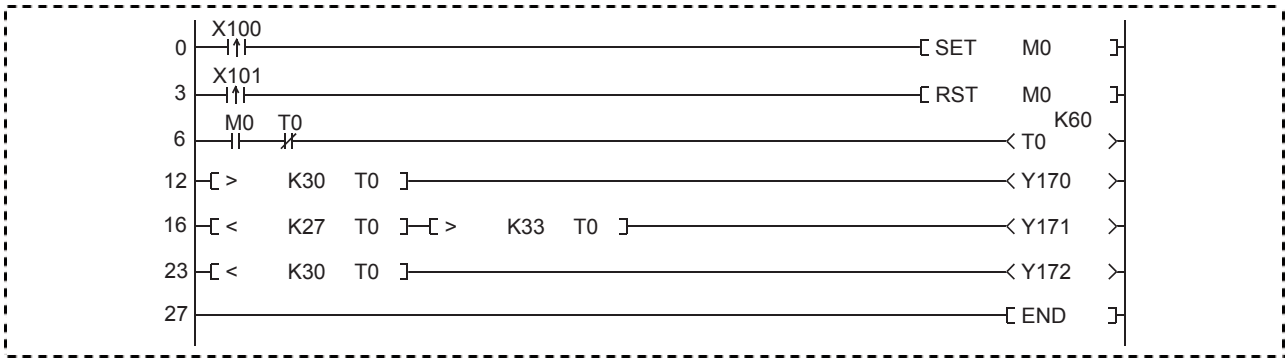
■Ladder example

Open the following project and write it to the CPU module of the demonstration machine. Then, check the execution of the > and < instructions.

Project name	REX10
Program name	MAIN

0.0 second ≤ T0 < 3 seconds → Y170: On
2.7 seconds < T0 < 3.3 seconds → Y171: On
3.0 seconds < T0 ≤ 6.0 seconds → Y172: On



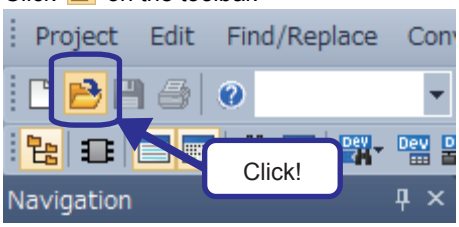


Operating procedure

■Opening a project

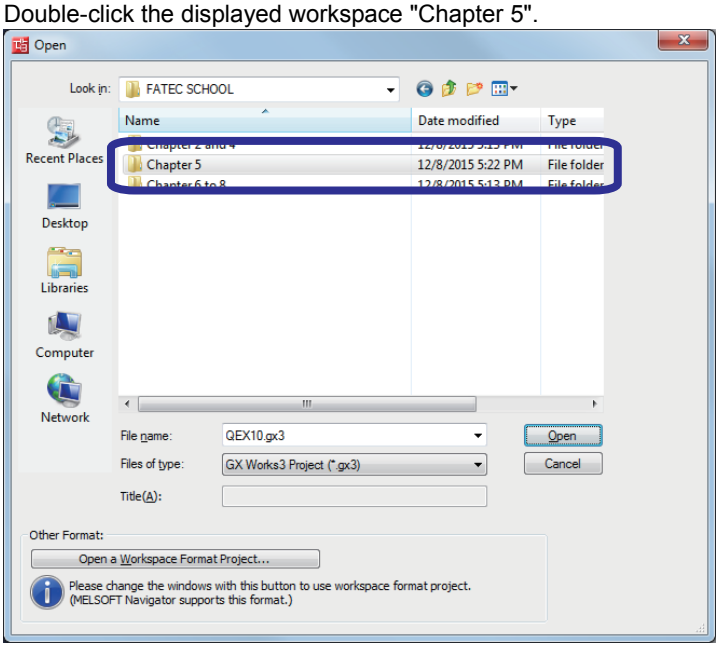
1. Open project data.

Click on the toolbar.

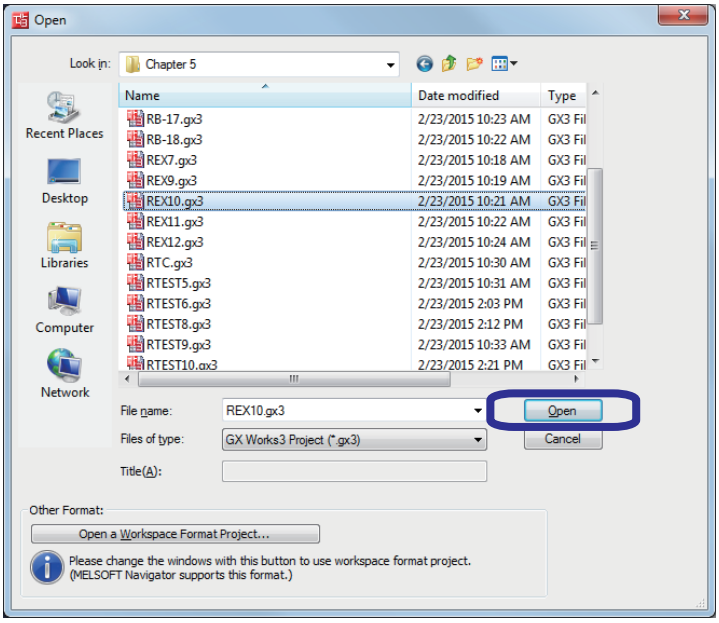


2. The "Open" dialog box appears. Specify the save destination.

Double-click the displayed workspace "Chapter 5".



3. Click "REX10" and click the [Open] button.



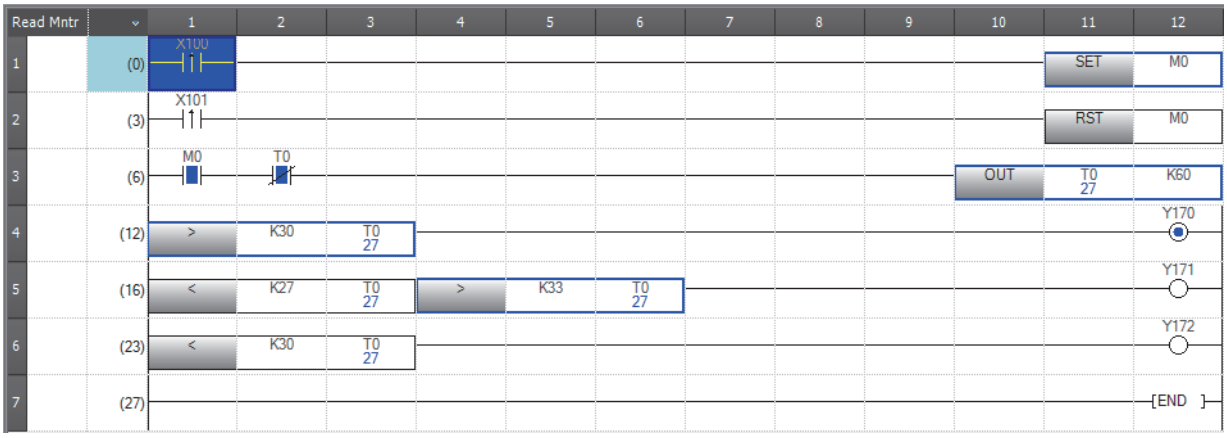
For the procedures of the following operations, refer to Section 4.4.

■Writing data to the programmable controller

■Monitoring the ladder

■Operation practice

Turn on X100 and check the operation of the program.



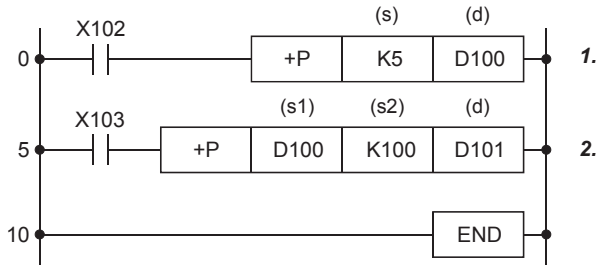
5.4 Arithmetic Operation Instructions

5.4.1 [+(P)] (Addition of 16-bit binary data) [-(P)] (Subtraction of 16-bit binary data)

Point

- This section describes addition or subtraction.
- This section describes the differences between the instructions with P and the one without P.

Project name	RB-16
Program name	MAIN



1. Every time the input condition turns on, the value in the device specified in (s) is added to the value in the device specified in (d), and the result is stored in the device specified in (d).

	(d)		(s)		(d)
	D100 [ ]	+	(5)	→	D100 [ ]
(Input condition)					
First: ON	0 (example)	+	5	→	5
Second: ON	5	+	5	→	10
Third: ON	10	+	5	→	15

The value in D100 changes.

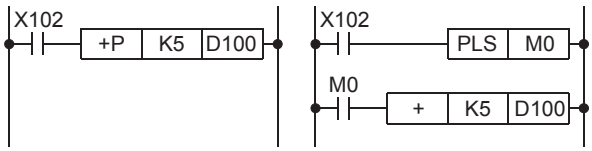
2. When the input condition turns on, the value in the device specified in (s1) is added to the value in the device specified in (s2), and the result is stored in the device specified in (d).

	(s1)		(s2)		(d)
	D100 [ ]	+	(100)	→	D101 [ ]
(Input condition)					
ON	15 (example)	+	100	→	115

The value in D100 is not changed by the addition.

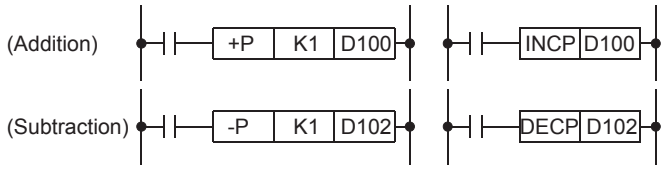
Precautions

- Always use [+P] or [-P] as the addition or subtraction instructions.
- When + or - is used, an addition or subtraction operation is executed at every scan. To use + or -, convert operands into pulse in advance.

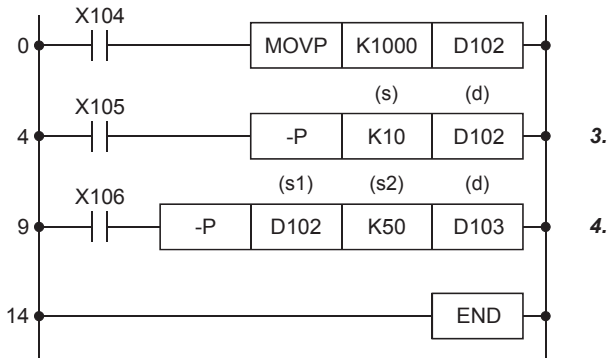


Point

The following two instructions work on the same principle in the addition or subtraction processing.



Project name	RB-17
Program name	MAIN



3. Every time the input condition turns on, the value in the device specified in (s) is subtracted from the value in the device specified in (d), and the result is stored in the device specified in (d).

	(d)		(s)		(d)
	D102 [ ]	-	(10)	→	D102 [ ]
(Input condition)					
First: ON	1000 (example)	-	10	→	990
Second: ON	990	-	10	→	980
Third: ON	980	-	10	→	970

The value in D102 changes.

4. When the input condition turns on, the value in the device specified in (S2) is subtracted from the value in the device specified in (S1), and the result is stored in the device specified in (d).

	(s1)		(s2)		(d)
	D102 [ ]	-	(50)	→	D103 [ ]
(Input condition)					
ON	970 (example)	-	50	→	920

The value in D102 is not changed by the subtraction.

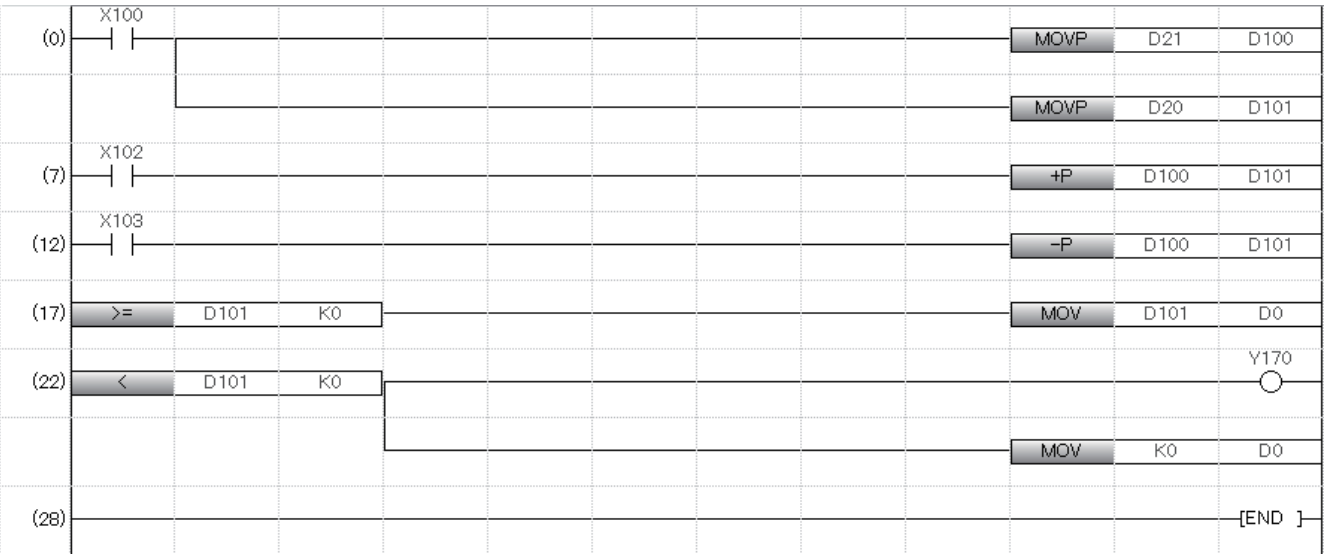
Operand	Bit		Word			Double word		Indirect specification	Constant			Others	Number of basic steps
	X, Y, M, L, SM, F, B, SB, FX, FY	J□\□	T, ST, C, D, W, SD,SW, FD, R, ZR, RD	U□\G□, J□\□, U3E□\(H)G□	Z	LT, LST, LC	LZ		K, H	E	$		
(s1)	○	—	○	○	○	—	—	○	○	—	—	—	3 or 4*1
(s2)	○	—	○	○	○	—	—	○	○	—	—	—	
(d)	○	—	○	○	○	—	—	○	—	—	—	—	

*1 The number of basic steps is four for [ ] (s1) (s2) (d).

■Ladder example

Create the following ladder program with GX Works3 and write it to the CPU module of the demonstration machine. Then, check the execution of the + and - instructions.

Project name	REX11
Program name	MAIN



Operating procedure

For the procedures of the following operations, refer to Section 4.4.

■Creating a new project

■Creating a program

■Writing data to the programmable controller

■Monitoring the ladder

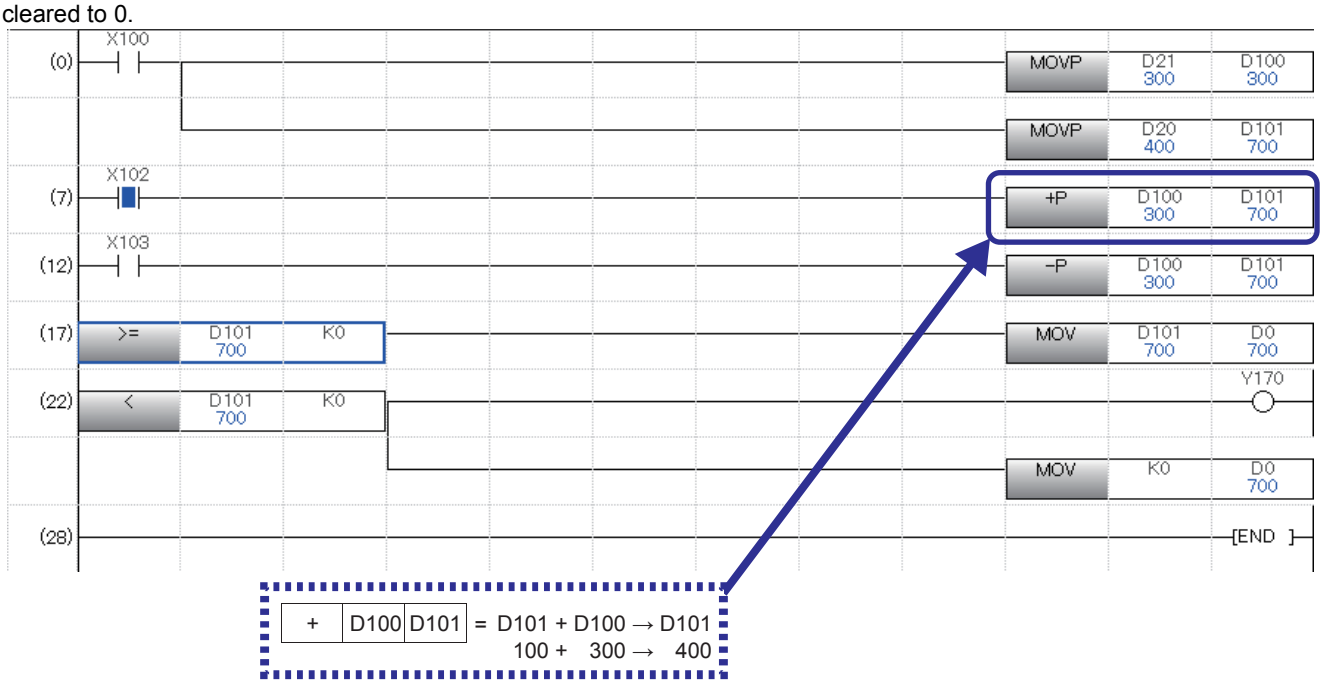
■Operation practice

❶ When X100 turns on, the value in D21 is stored in D100 and the value in D20 is stored in D101.

❷ Turn on X102. The value in D100 is added to the value in D101.

❸ Turn on X103. The value in D100 is subtracted from the value in D101.

❹ The initial indication device D0 displays the calculation result. When the result is a negative value, Y170 turns on and D0 is cleared to 0.

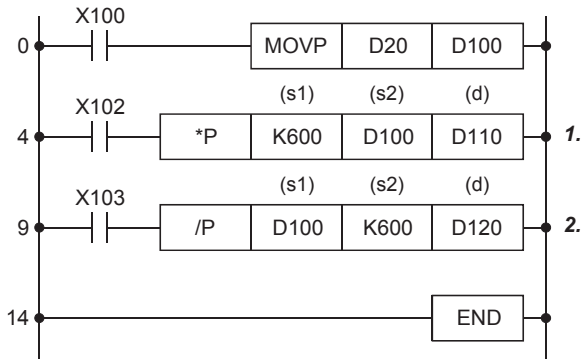


5.4.2 [*(P)] (Multiplication of 16-bit binary data) [/(P)] (Division of 16-bit binary data)

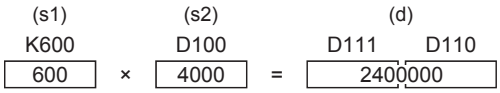
Point

- This section describes multiplication or division.
- This section describes the concept of two words.

Project name	RB-18
Program name	MAIN



1. When the input condition turns on, the value in the device specified in (s1) is multiplied by the value in the device specified in (s2), and the result is stored in the device specified in (d).

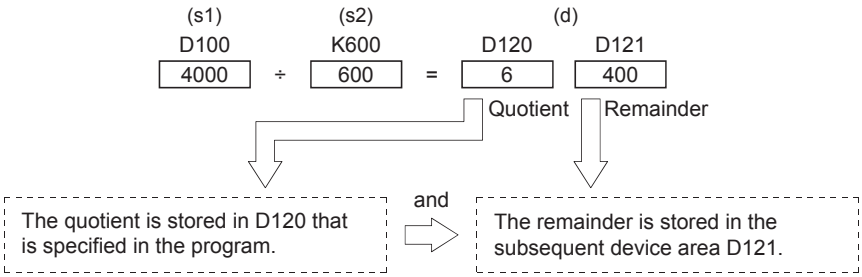


To store the result of 16-bit data × 16-bit data, a space of 16 bits (1 word) is not enough. Thus, the D110 specified in the program and the subsequent device area D111 are used for storing the result.

Since these device areas are used as a 32-bit (2-word) register for storing the result, the left-most bit of D110 (b15) is regarded as a part of the data, not as a sign bit.

When programming a ladder using the operation result of the *(P) instruction, always use 32-bit instructions (such as the DMOV instruction and the DMOVP instruction).

2. When the input condition turns on, the value in the device specified in (s1) is divided by the value in the device specified in (s2), and the result is stored in the device specified in (d). Values after the decimal point of the division result are ignored.



When a bit device is specified in (d), the quotient is stored but the remainder is not stored.
The following shows examples of processing of negative values.

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

The following shows examples of dividing a value by 0 or dividing 0 by a value.

Example
0 ÷ 0
1 ÷ 0
} Error "OPERATION ERROR"
0 ÷ 1 Quotient and remainder = 0

■Operation practice

❶ Write the program to the CPU module and set the operating status of the CPU module to RUN.
❷ Turn on X100 and store the value of the initial input device D20 in D100.
❸ Turn on X102. The multiplication of 600 × D100 is executed.
❹ Turn on X103. The division of D100 ÷ 600 is executed.

Operand	Bit		Word			Double word		Indirect specification	Constant			Others	Number of basic steps
	X, Y, M, L, SM, F, B, SB, FX, FY	J□\□	T, ST, C, D, W, SD,SW, FD, R, ZR, RD	U□\G□, J□\□, U3E□\(H)G□	Z	LT, LST, LC	LZ		K, H	E	$		
(s1)	○	—	○	○	○	—	—	○	○	—	—	—	Multiplication instruction: 3 or 4*1 Division instruction: 4
(s2)	○	—	○	○	○	—	—	○	○	—	—	—	
(d)	○	—	○	○	○	—	—	○	—	—	—	—	

*1 The number of steps in a multiplication instruction varies depending on the devices to be used.

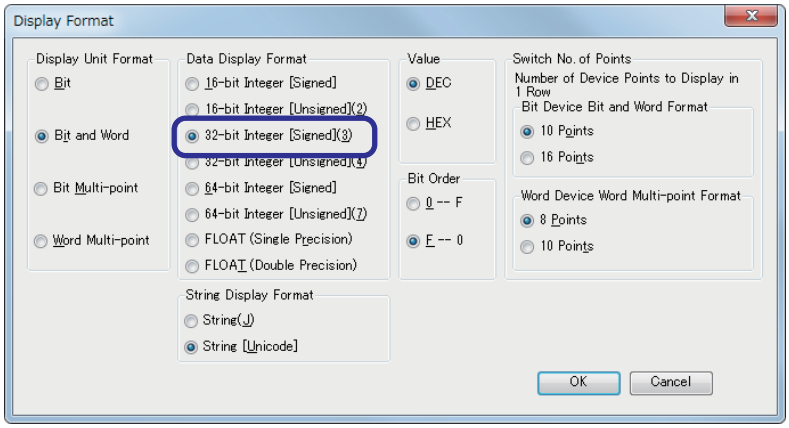
Point

How to monitor 32-bit integer data

When the operation result of the multiplication instruction is outside the range of 0 to 32767, the result cannot be properly displayed even though the value is regarded as a 16-bit integer and the values in the lower registers are monitored in ladder.

To monitor the values properly, follow the procedure below.

- Click [View] → [Display Format Detailed Setting] from the menu to display the "Display Format" dialog box. Then, set "32-bit Integer [Signed]" for "Data Display Format".
  Click the [OK] button.



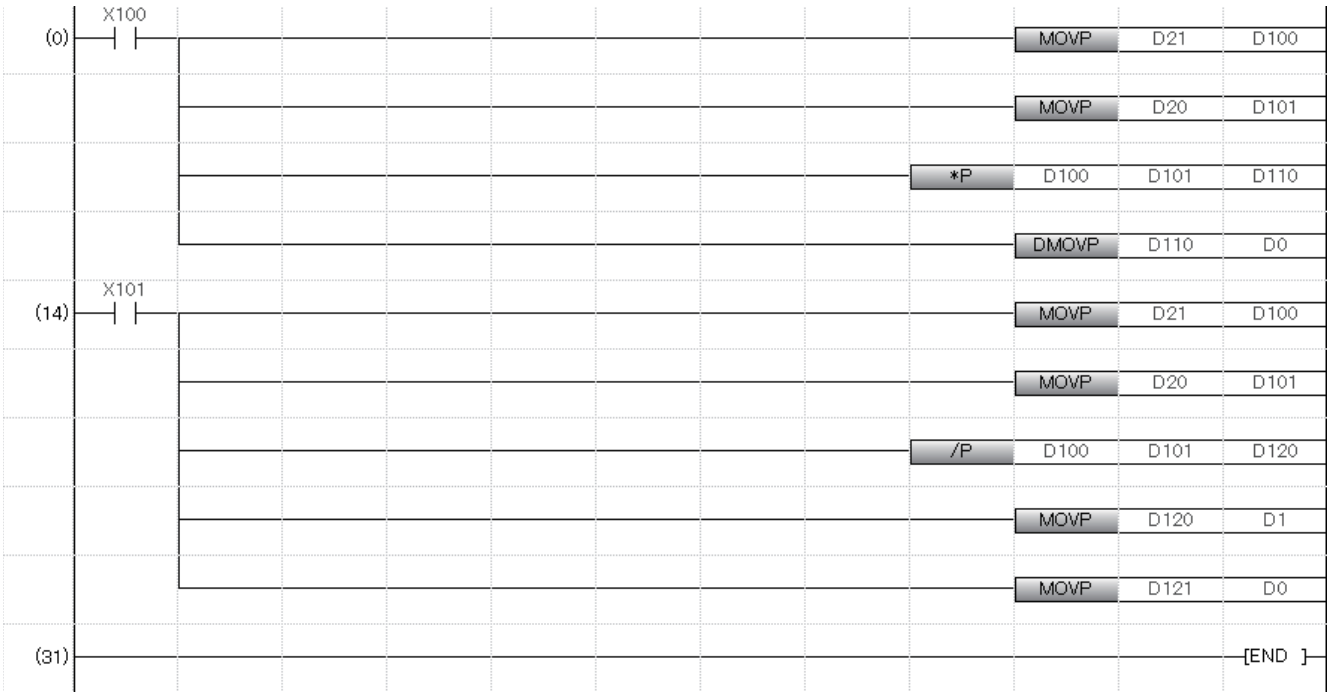
- Values are monitored properly.

Device Name	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0	Current Value
D100	0	0	0	0	1	1	1	1	1	0	1	0	0	0	0	0	4000
D101	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
D102	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
D103	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
D104	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
D105	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
D106	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
D107	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
D108	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
D109	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
D110	1	0	0	1	1	1	1	1	0	0	0	0	0	0	0	0	2400000

■Ladder example

Create the following ladder program with GX Works3 and write it to the CPU module of the demonstration machine. Then, check the execution of the * and / instructions.

Project name	REX12
Program name	MAIN



Operating procedure

For the procedures of the following operations, refer to Section 4.4.

■Creating a new project

■Creating a program

■Writing data to the programmable controller

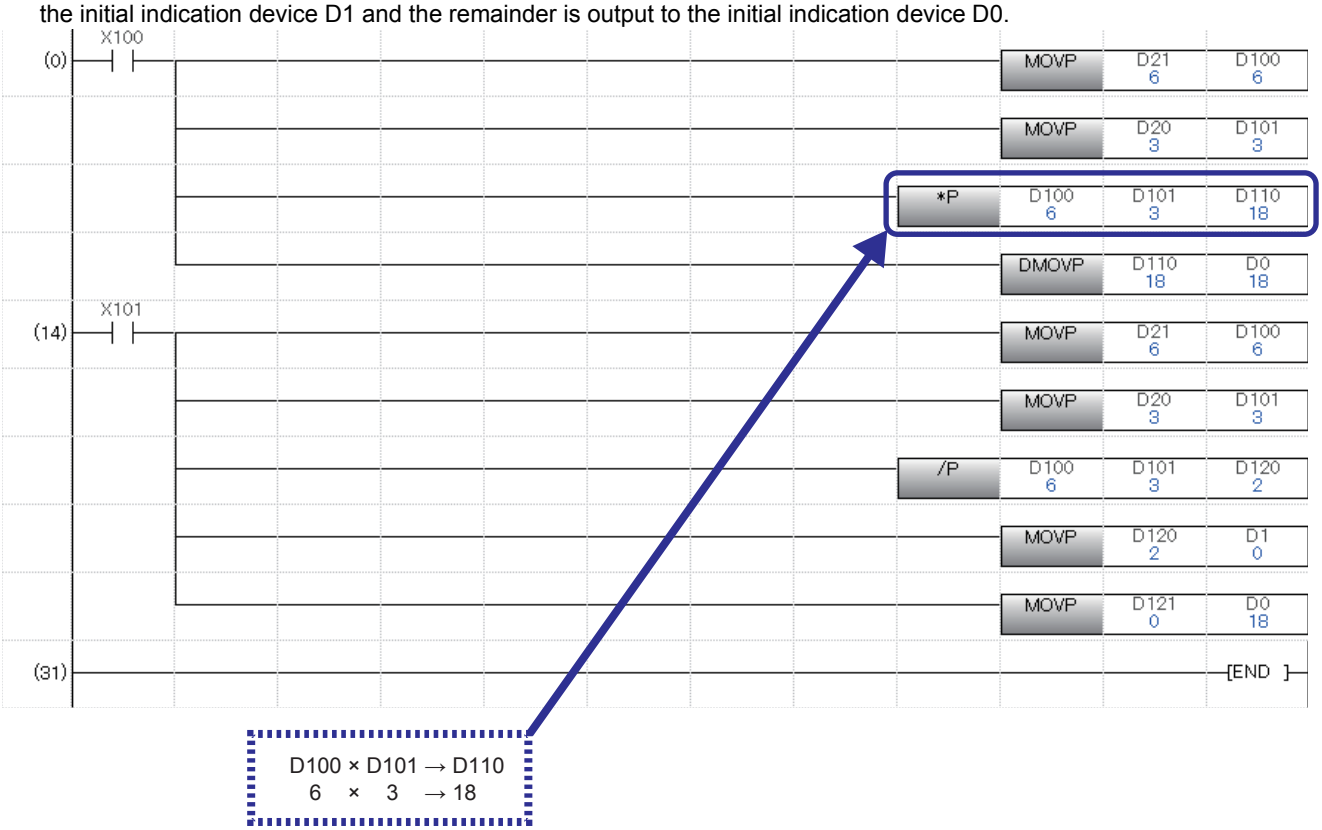
After writing data, touch the lower sections of the initial input device D20 and D21 to enter numerical values.

■Monitoring the ladder

■Operation practice

❶ When X100 turns on, the values in the initial input device D20 and D21 are multiplied, and the result is output to the initial indication device D0.

❷ When X101 turns on, the value in the initial input device D21 is divided by the value in D20, and the quotient is output to the initial indication device D1 and the remainder is output to the initial indication device D0.



5.4.3 32-bit data instructions and their necessities

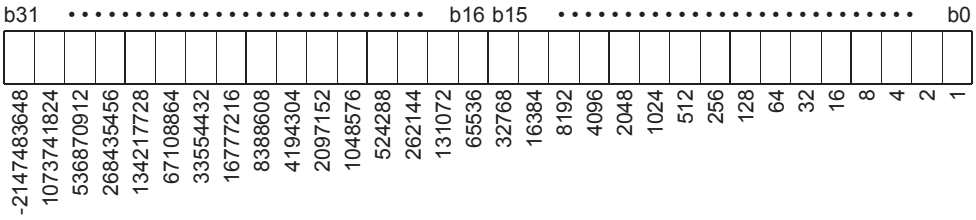
Point

- This section describes the concept of two words.
- This section describes the differences between a one-word instruction and two-word instruction.

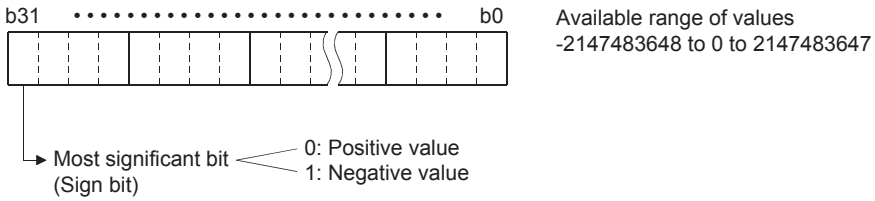
- The unit of the data memory of the MELSEC iQ-R series programmable controller is one word that consists of 16 bits. Thus, data is typically processed in units of one word at the transfer processing, comparison, and arithmetic operation.
- The MELSEC iQ-R series programmable controller can process data in units of two words (32 bits). In that case, "D" is added at the beginning of each instruction to indicate that the instruction processes two-word data. The following shows examples.

Instruction	1 word 16 bits	2 words 32 bits
Transfer	MOV(P)	DMOV(P)
Comparison	<, >, <=, >=, =, <>	D<, D>, D<=, D>=, D=, D<>
Arithmetic operation	+(P)	D+(P)
	-(P)	D-(P)
	*(P)	D*(P)
	/(P)	D/(P)
Available range of numerical values	-32768 to 32767	-2147483648 to 2147483647
Available range of digit specification	K1 to K4	K1 to K8

- The following shows the weights of 32 bits.

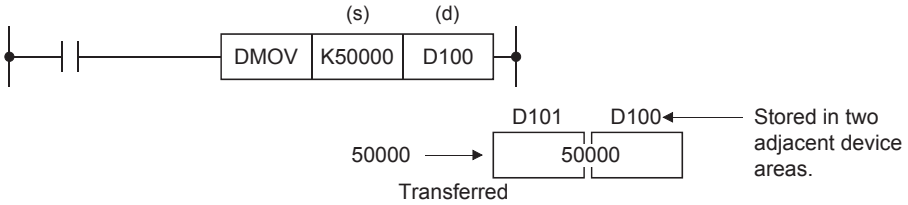


As the case of 16-bit data processing, the programmable controller takes the 2's complement in 32-bit data processing. Thus, the most significant bit b31 (b15 for 16-bit data) is processed as a sign bit.

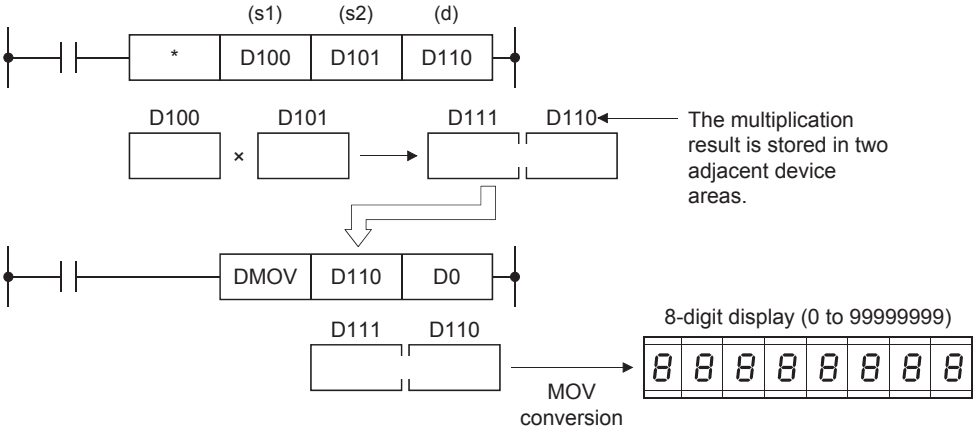


• Whether data is processed as two-word (32-bit) data or not depends on the size of the data.
In the following cases, use two-word instructions.

(1) When the data size exceeds the range (-32768 to 32767) in which data can be processed as one word

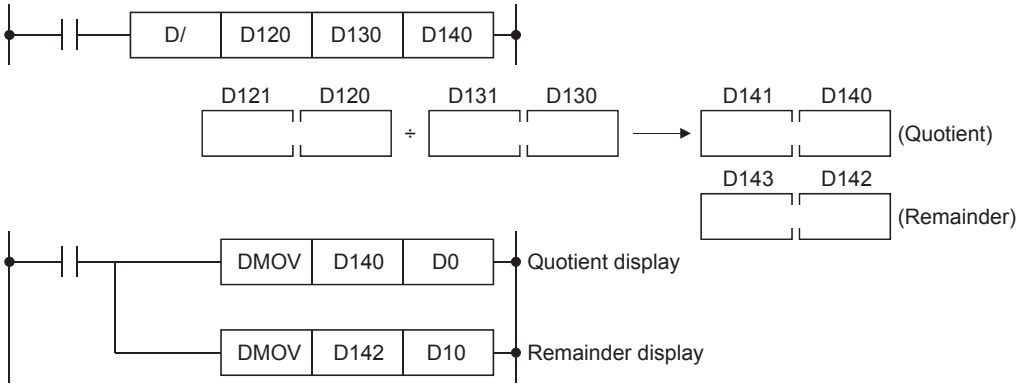


(2) When the result of the 16-bit multiplication instruction (one-word instruction) is transferred



*1 The result of the 32-bit data multiplication will be 64-bit data.

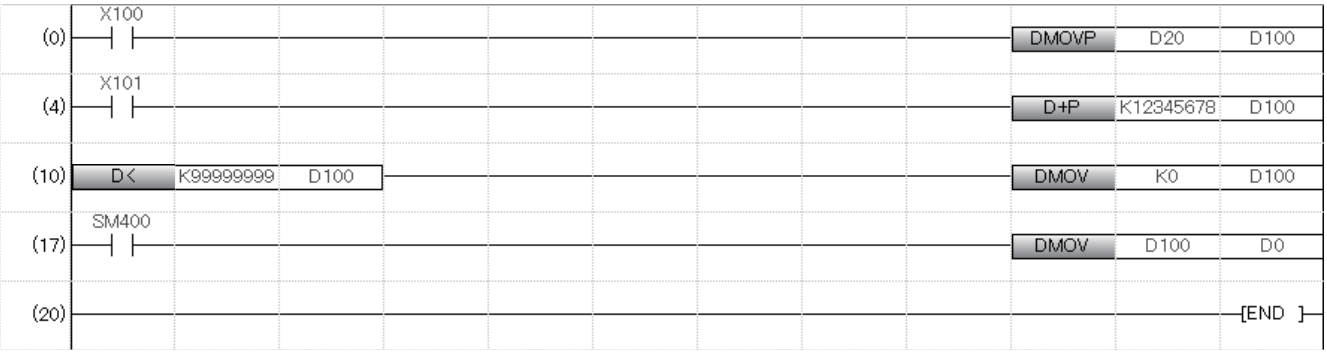
(3) When the result of the 32-bit division instruction is used



■Ladder example

Create the following ladder program with GX Works3 and write it to the CPU module of the demonstration machine. Then, check the two-word instructions.

Project name	2word
Program name	MAIN



Operating procedure

For the procedures of the following operations, refer to Section 4.4.

■Creating a new project

■Creating a program

■Writing data to the programmable controller

■Monitoring the ladder

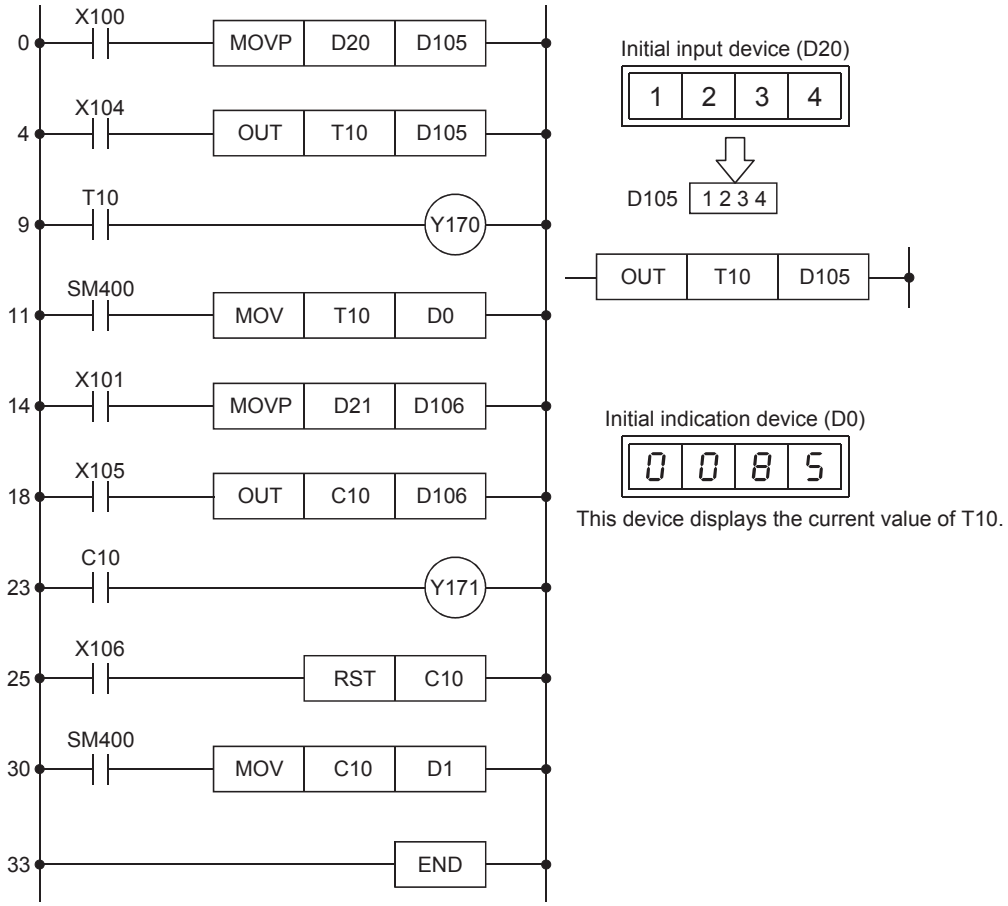
5.5 External Setting of Timer/Counter Values and External Display of Current Values

Point

This section describes how to indirectly specify a setting value of a timer/counter.

A setting value of a timer or counter can be directly specified with K (decimal constant) or indirectly specified with D (data register). In the program shown below, the setting value can be changed from an external device.

Project name	RTC
Program name	MAIN



After reading the program to GX Works3, write it to the programmable controller to check the operation.

Operating procedure

For the procedure of creating a project, refer to Section 5.3.
For the procedures of the operations after creating a project, refer to Section 4.4.

■Creating a new project

■Creating a program

■Writing data to the programmable controller

■Monitoring the ladder

■Operation practice

(1) External setting of timer values and display of current values

- Set the value of a timer in the initial input device (D20) and turn on the switch X100.
- When the switch X104 turns on, Y170 turns on after the time specified with the initial input device (D20).
  (For example, Y170 turns on in 123.4 seconds when | 1 | 2 | 3 | 4 | is set to the timer.)
- The initial indication device (D0) displays the current value of the timer T10.

(2) External setting of counter values and display of current values

- Set the value of a counter in the initial input device (D21) and turn on the switch X101.
- Turn on and off the switch X105 repeatedly. When X105 has turned on for the number of times specified with the initial input device (D21) (counting is up), Y171 turns on.
- The initial indication device (D1) displays the current value of the counter C10 (the number of times that X105 has turned on).
- Turning on the switch X106 clears the counter C10 to 0. When the contact C10 is already on (counting up has already been completed), the contact is released.

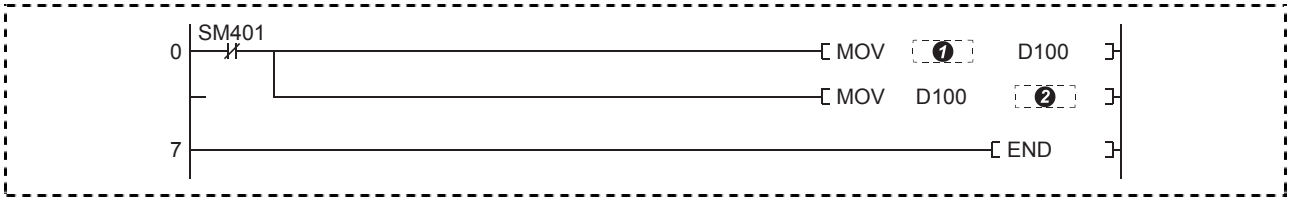
5.6 Exercise

5.6.1 [Exercise 1] MOV-1

Project name	RTEST5
Program name	MAIN

Transfer the eight input points of X100 to X107 into D100 once and output them to Y170 to Y177.
(For example, Y170 turns on when X100 turns on.)
X100 → Y170, X101 → Y171, X102 → Y172, X103 → Y173, X104 → Y174, X105 → Y175, X106 → Y176, X107 → Y177

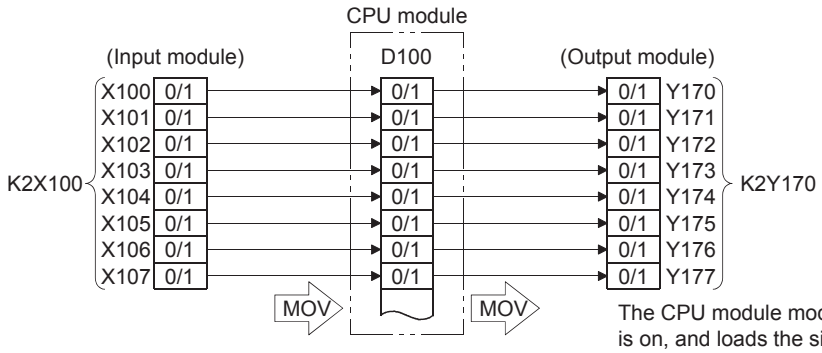
Fill in the blanks ( ) in the following program and create the program with GX Works3. Then, check the operation with the demonstration machine.
(For answers, refer to Page 5-46.)



❶ _____

❷ _____

■Hint



The CPU module module loads the input signal as "1" when it is on, and loads the signal as "0" when it is off.
The output module turns on when the CPU module module outputs "1", or turns off when the CPU module module outputs "0".

Point

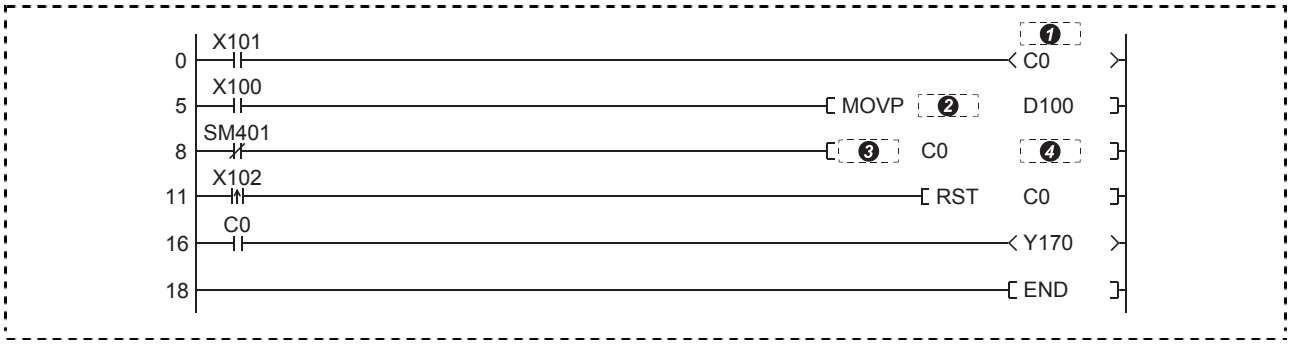
The following is a program created with sequence instructions and no MOV instructions.

5.6.2 [Exercise 2] MOV-2

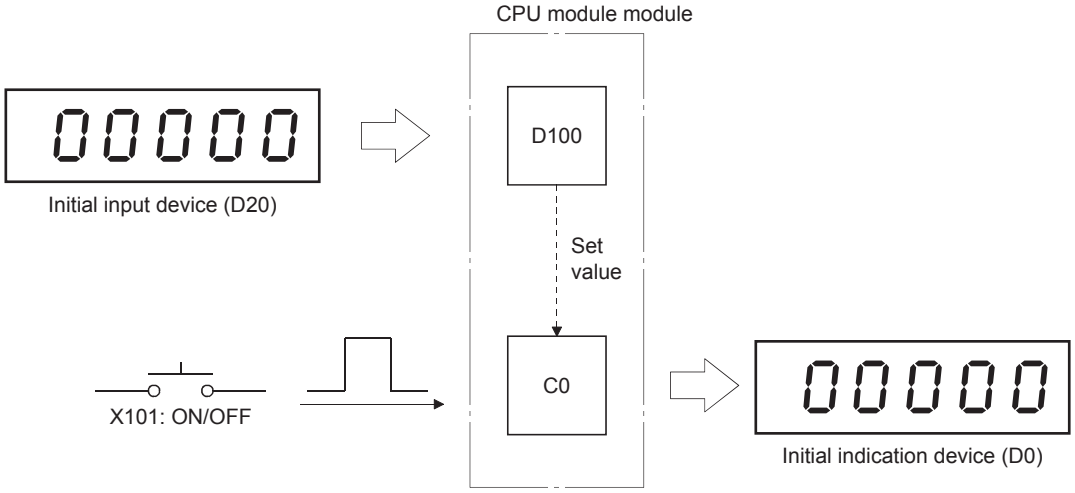
Project name	RTEST6
Program name	MAIN

Output the number of times that X101 has turned on to the initial indication device D0. As a precondition, the value set to the counter (C0) can be input with the initial input device (D20) and the setting will be available with turning on of X100.

Fill in the blanks ( ) in the following program and create the program with GX Works3. Then, check the operation with the demonstration machine.
(For answers, refer to Page 5-46.)



■Hint



❶ _____
❷ _____
❸ _____
❹ _____

5.6.3 [Exercise 3] Comparison instruction

Project name	RTEST8
Program name	MAIN

Use two initial input devices to perform the operation processing of (A - B) and display the result on the initial indication device (D0).

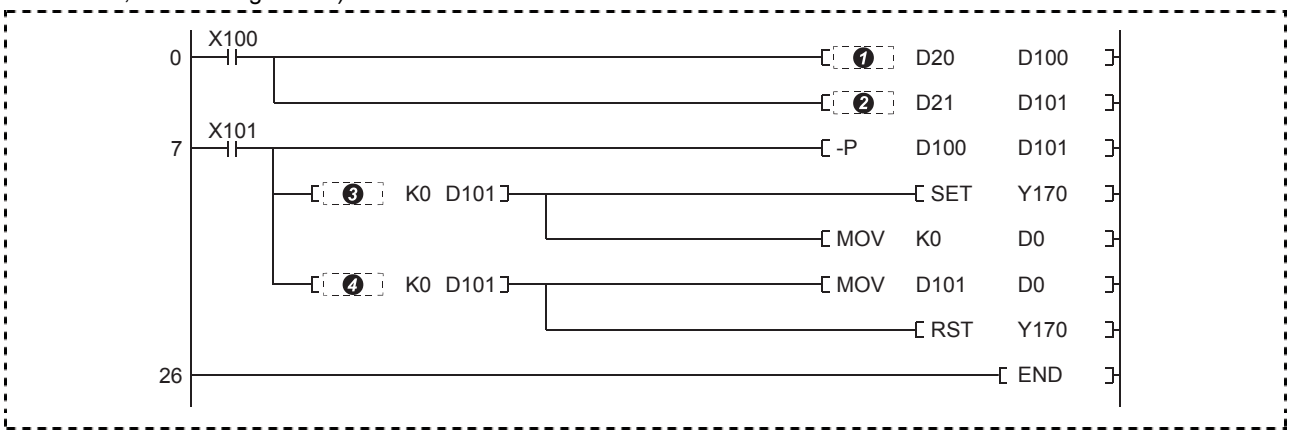
Initial input device (D21) — A

Initial input device (D20) — B

Initial indication device (D0)

The initial indication device (D0) indicates the result of the calculation of A - B. When the result is a negative value, the device indicates 0 and the LED of Y170 turns on.

Fill in the blanks ( ) in the following program and check the operation of the program with the demonstration machine. (For answers, refer to Page 5-46.)



■Hint

The operation result is always output from the CPU module in binary.

-	D100	D101	= D101 - D100 → D101

❶ _____

❷ _____

❸ _____

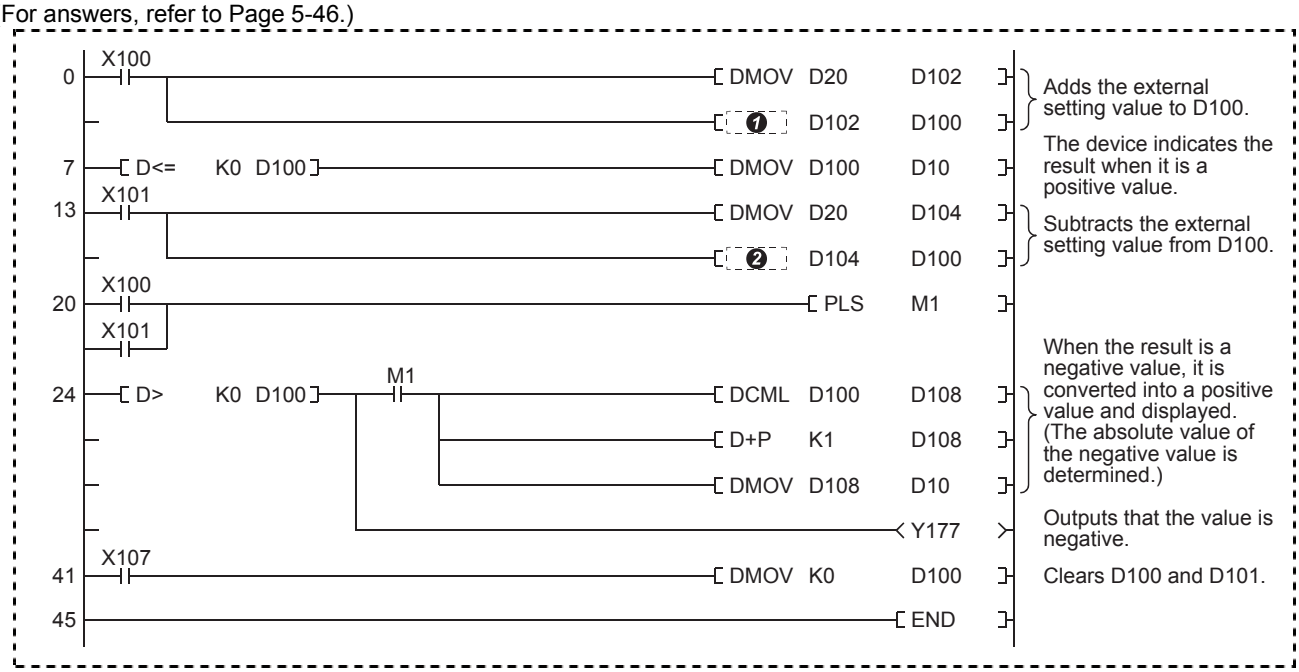
❹ _____

5.6.4 [Exercise 4] +, -

Project name	RTEST9
Program name	MAIN

Load the value specified by the initial input device (D20) into D103 and D102 (32-bit data) when X100 turns on, add each of them to D101 and D100, and display the results in the initial indication device (D10).
Load the value specified by the initial input device (D20) into D105 and D104 when X101 turns on, subtract each of them from D101 and D100 and display the results. When the result is a negative value, Y177 turns on, the 2's complement is taken from the result to obtain and display an absolute value.

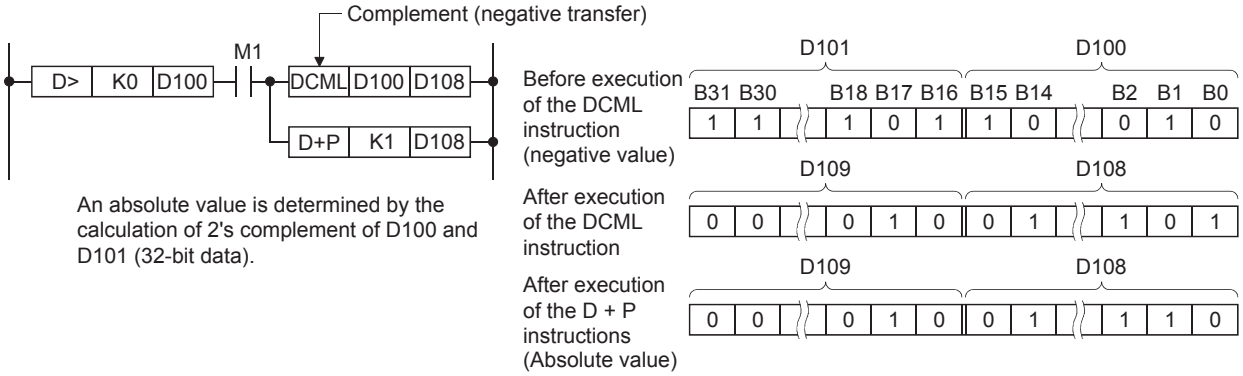
Fill in the blanks ( ) in the following program and check the operation of the program with the demonstration machine. (For answers, refer to Page 5-46.)



❶ _____

❷ _____

■Reference



Point

The CML instruction inverts the bit patterns of (s) and transfers the data into (d) when the input condition turns on.

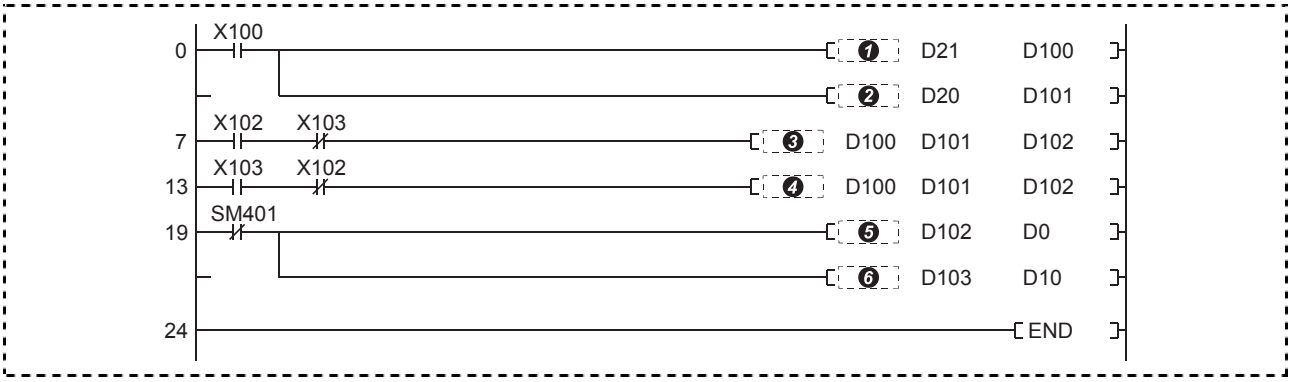
5.6.5 [Exercise 5] *, /

Project name	RTEST10
Program name	MAIN

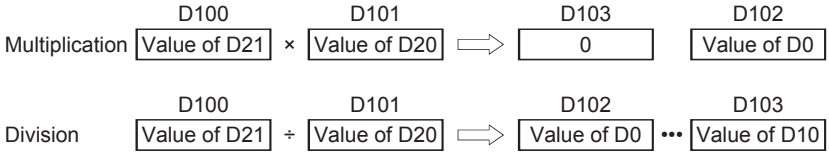
Multiplication and division data can be set when X100 turns on. The values specified by the initial input device D20 to D21 are multiplied when X102 turns on or are divided when X103 turns on. Create a program that outputs the result of the multiplication or the quotient of the division to the initial indication device (D10) and the remainder of the division to the initial indication device (D0).

(D21) × (D20) → (D0)
(D21) ÷ (D20) → (D0) •••••• (D10)
Two-digit numerical values are stored in D20 and D21.

Fill in the blanks ( ) in the following program and create the program with GX Works3. Then, check the operation with the demonstration machine.
(For answers, refer to Page 5-46.)



■Hint



❶ _____
❷ _____
❸ _____
❹ _____
❺ _____
❻ _____

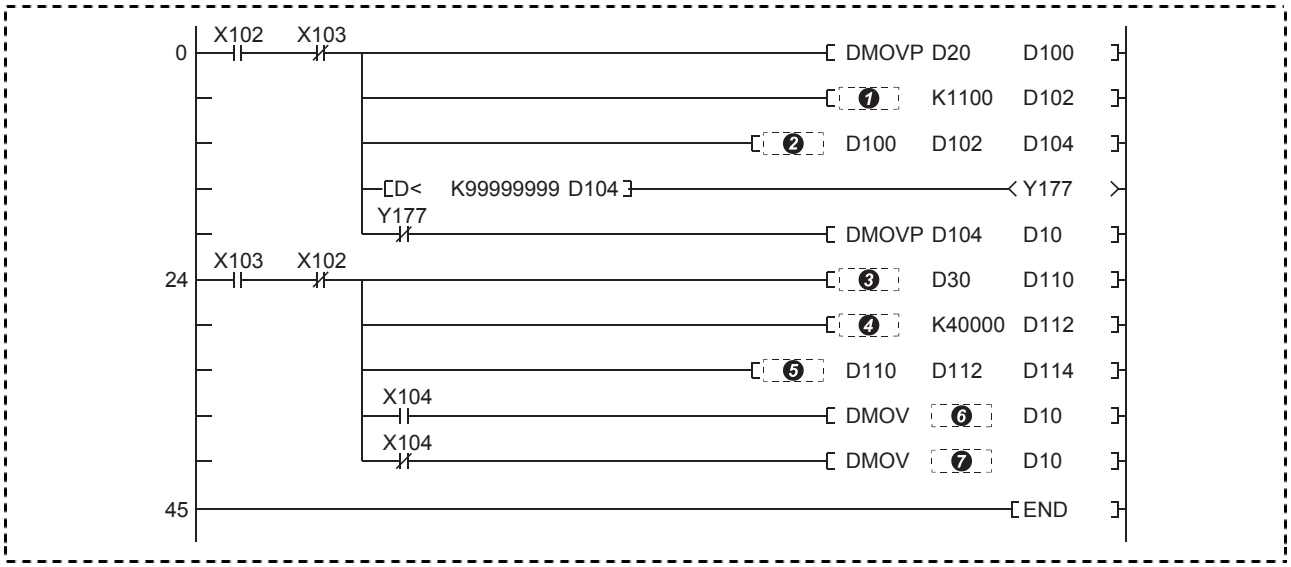
5.6.6 [Exercise 6] D*, D/

Project name	RTEST11
Program name	MAIN

Multiply the value set by the 5-digit initial input device (D20) by 1100 when X102 turns on. When the result is 99999999 or smaller, display the value in the 10-digit initial indication device (D10).
Divide the value set by the 8-digit initial input device (D30) by 40000 when X103 turns on. When X104 is on, display the quotient in the initial indication device (D10). When X104 is off, display the remainder in the 10-digit initial indication device (D10).

(D20) × 1100 → (D10)
(D30) ÷ 40000 → Quotient (D10) •••••• X104: On
Remainder (D10) •••••• X104: Off

Fill in the blanks ( ) in the following program and create the program with GX Works3. Then, check the operation with the demonstration machine.
(For answers, refer to Page 5-46.)



❶ _____
❷ _____
❸ _____
❹ _____
❺ _____
❻ _____
❼ _____

Answers for the exercises in Chapter 5

Exercise		Answer
1	❶	K2X100
	❷	K2Y170
2	❶	D100
	❷	D20
	❸	MOV
	❹	D0
3	❶	MOVP
	❷	MOVP
	❸	>
	❹	<=
4	❶	D+P
	❷	D-P
5	❶	MOVP
	❷	MOVP
	❸	*P
	❹	/P
	❺	MOV
	❻	MOV
6	❶	DMOVP
	❷	D*P
	❸	DMOVP
	❹	DMOVP
	❺	D/P
	❻	D114
	❼	D116

6 HOW TO USE OTHER FUNCTIONS

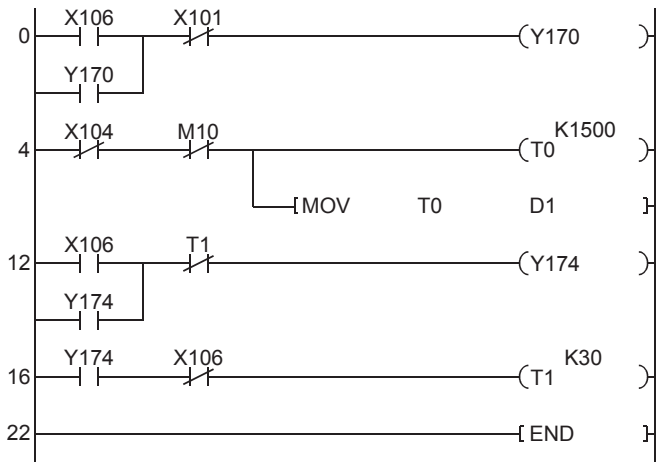
6.1 Online Test Function

Point

This section describes how to change the status of devices forcibly.

As a preparation, follow the procedure below.

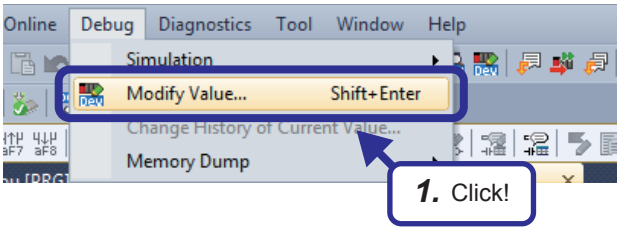
Project name	REX14
Program name	MAIN



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

1. Read the REX14 project with GX Works3.
2. Write the parameters and programs of the read project to the CPU module.
   (The CPU module is in the STOP state.)
3. Set GX Works3 to monitor mode.
4. Check the program displayed in the window.

6.1.1 Forced on/off of the device (Y)



Set the CPU module to the STOP state before this operation.

1. Select the "Y170" cell on the ladder editor and click [Debug] → [Modify Value] from the menu.
Clicking the menu forcibly turns on or off "Y170".

Checking with the demonstration machine

Check that clicking the menu switches the on/off status of Y170 and the LED of Y170 on the demonstration machine also turns on and off depending on this operation.

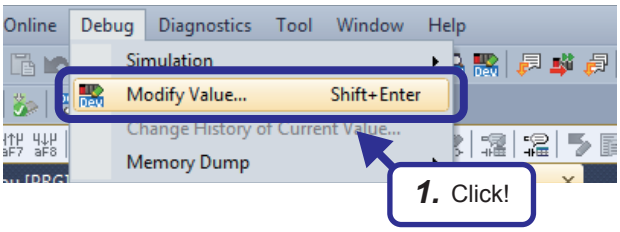
Precautions

When the CPU module is in the RUN state, operation results of the program are displayed preferentially. Thus, set the CPU module to the STOP state before checking with the demonstration machine.

Point

Setting and resetting of contacts, changing current values of word devices, and forced output can also be performed with the test function during ladder monitoring with GX Works3.
Double-clicking a contact (pressing the Enter key) while holding the Shift key in the ladder monitor window of GX Works3 forcibly switches the on/off status of the contact.
For word devices, this operation registers the change target devices on the watch window and changes the current values.

6.1.2 Setting/resetting of the device (M)



Set the CPU module to the RUN state before this operation.

1. Select the "M10" cell on the ladder editor and click [Debug] → [Modify Value] from the menu.
Clicking the menu sets or resets "M10".

Checking with the demonstration machine

Turn off X104 and check the following.

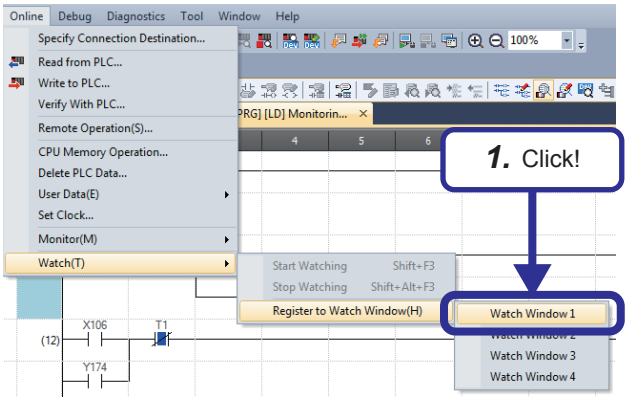
❶ When M10 is set, M10 goes in the non-continuity state and the current value of the timer T0 is cleared to 0.
Check that the display of the initial indication device (D1) stops.

❷ When M10 is reset, M10 goes in the continuity state and the timer T0 starts counting from 0. The counted value increases by 10 every second.
Check that the value in the initial indication device (D1) increases by 10 every second.

Point

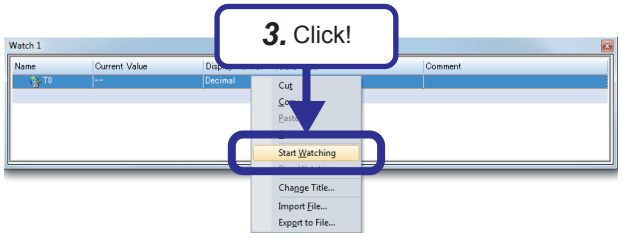
With the same procedure, bit devices other than the internal relay (M) also can be set or reset forcibly.

6.1.3 Current value change of the device (T)



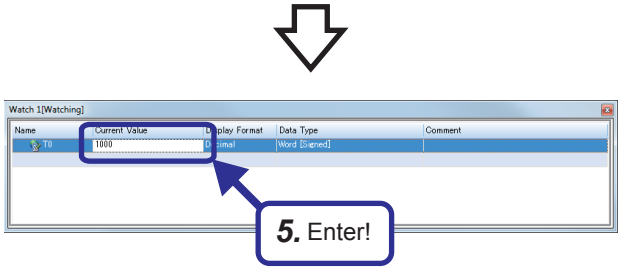
Set the CPU module to the RUN state before this operation.

1. Select the "T0" cell on the ladder editor and click [Online] → [Watch] → [Register to Watch Window] → [Watch Window 1].



2. "T0" is registered in the "Watch 1" window.

3. Right-click "T0" in the "Watch 1" window and click [Start Watching].



4. Watching of "Watch 1" starts.

5. Enter "1000" in "Current Value".

Checking with the demonstration machine

After entering the current value, press the Enter key and check that the value in the initial indication device (D1) changes to 1000.

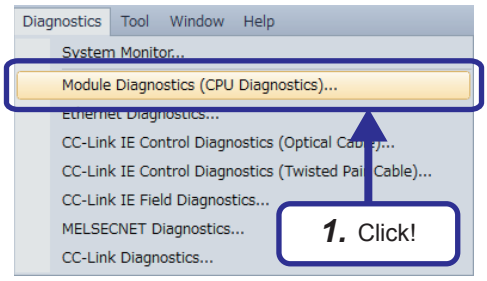
Point

With the same procedure, the current values of word devices other than the timer (T) can also be changed.

6.1.4 Reading error steps

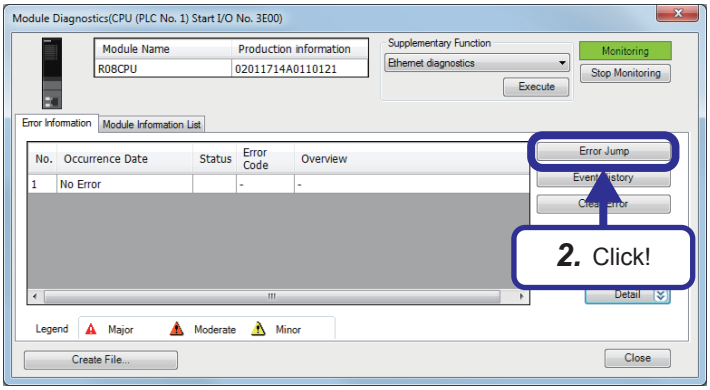
Point

This section describes how to check errors.



Set the CPU module to the RUN state before this operation.

1. Click [Diagnostics] → [Module Diagnostics (CPU Diagnostics)] from the menu.



2. The "Module Diagnostics" dialog box appears. Click the [Error Jump] button to jump to a selected error item.

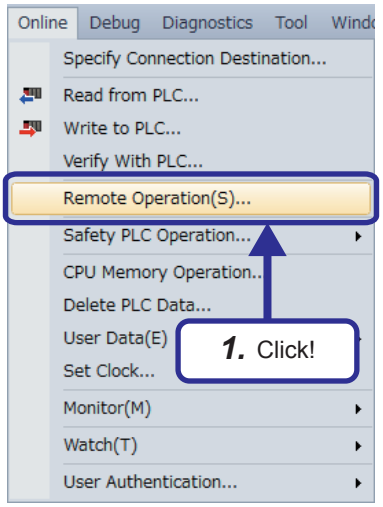
- When an error has been detected, the corresponding error code and overview are displayed.
- When no error has been detected, a message "No Error" is displayed.

6.1.5 Remote RUN/STOP

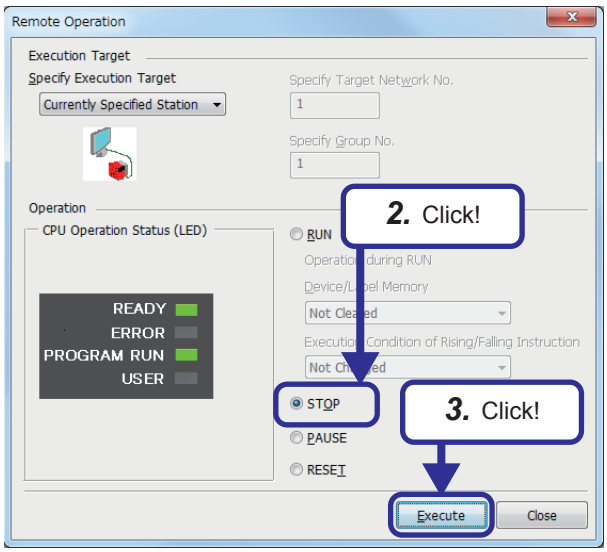
Point

This section describes the method of remote operation with the software.

Set the CPU module to the RUN state before this operation.

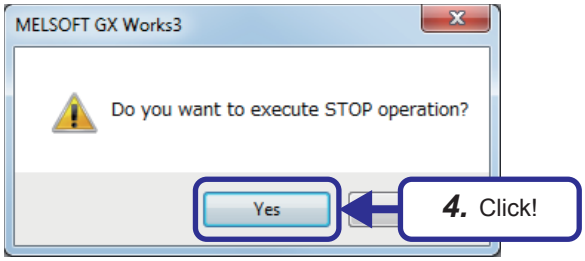


1. Click [Online] → [Remote Operation] from the menu.



2. The "Remote Operation" dialog box appears. Select "STOP" in "Operation".

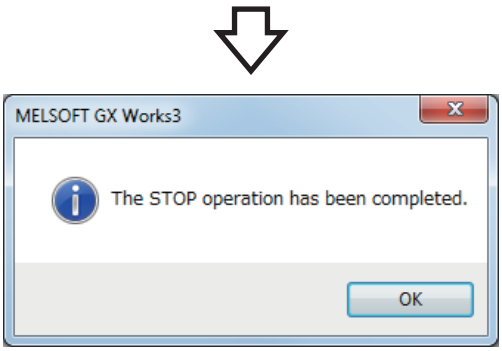
3. After the setting is completed, click the [Execute] button.



4. The message "Do you want to execute STOP operation?" appears. Click the [Yes] button.

(To the next page)

(From the previous page)



The CPU module is set to the STOP state.

5. Select "RUN" in the dialog box of step 2, and perform the steps 2 to 4 again.

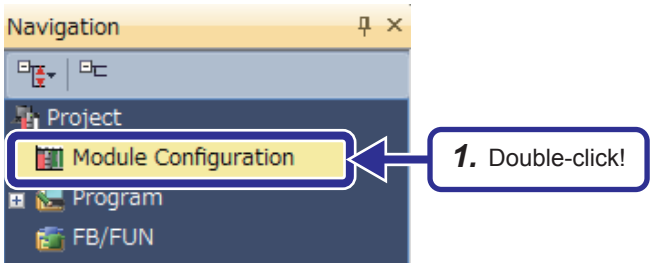
The CPU module that was in the STOP state is set to the RUN state again.

6.2 Creating the Module Configuration

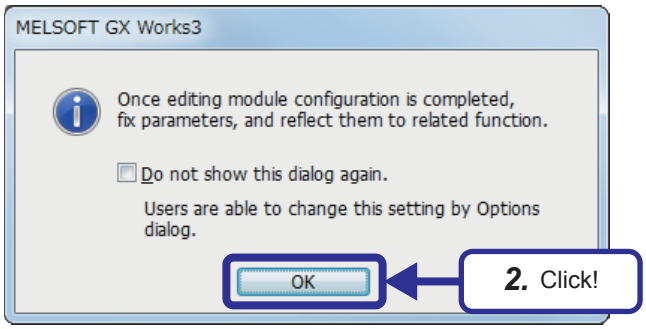
Arrange program elements (objects) in the "Module Configuration" window so that the same configuration as that of the demonstration machine is created.
(For how to create a project, refer to Section 2.2.2.)

The configuration that can be created in the "Module Configuration" window is the one to be managed with the CPU module of the project.

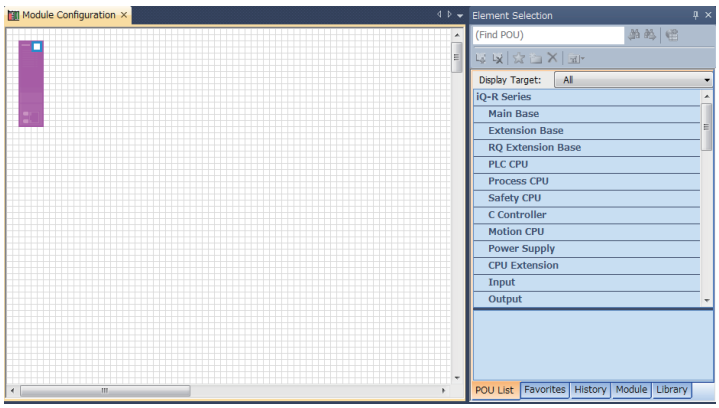
Operating procedure



1. Double-click "Module Configuration" in the "Project" view.



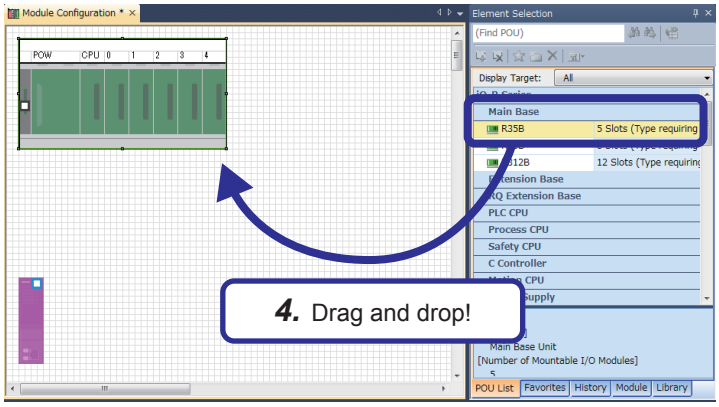
2. The message dialog box shown on the left appears. Click the [OK] button.



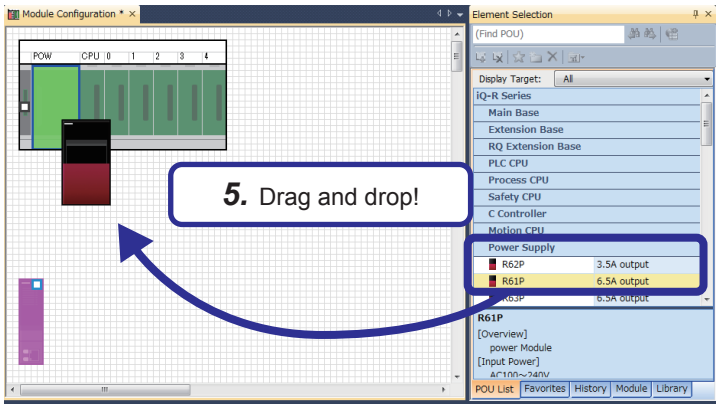
3. The "Module Configuration" window appears.

(To the next page)

(From the previous page)

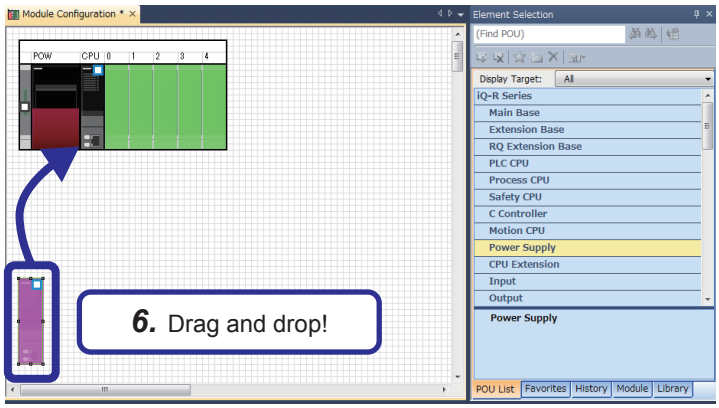


4. Select "R35B" from "Main Base" on the "Element Selection" window, and drag and drop it to the "Module Configuration" window.



5. Select "R61P" from "Power Supply" on the "Element Selection" window, and drag and drop it to the base unit arranged in ***4***.

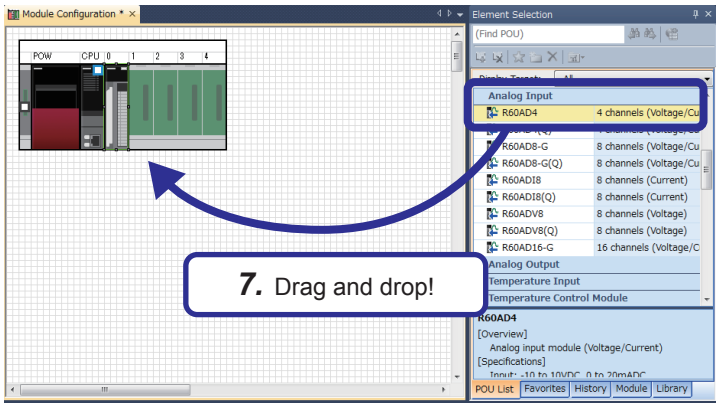
(While the power supply module is being dragged and dropped, the slot where the power supply module can be arranged is highlighted.)



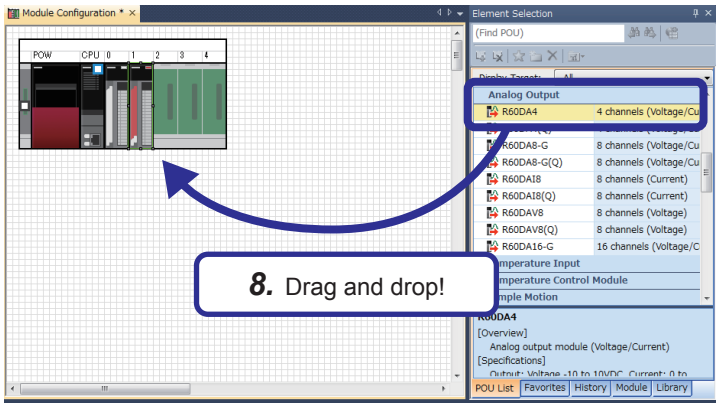
6. Select "R08CPU" that has already been arranged when the "Module Configuration" window appeared, and drag and drop it to the CPU slot on the base unit.

(To the next page)

(From the previous page)



7. Select "R60AD4" from "Analog Input" on the "Element Selection" window, and drag and drop it to the slot No.0 on the base unit.



8. Select "R60DA4" from "Analog Output" on the "Element Selection" window, and drag and drop it to the slot No.1 on the base unit.

Arranging of program elements (objects) in the "Module Configuration" window is completed in the same configuration as the one of the demonstration machine.

6.3 Device Batch Replacement

Point

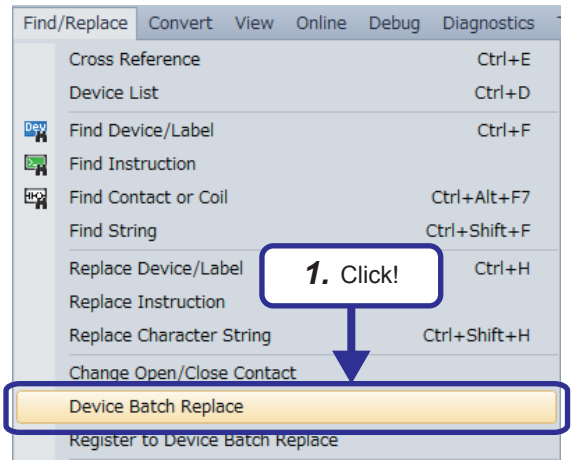
This section describes how to change devices in a program in a batch.

6.3.1 Replacing device numbers in a batch

Replace Y140 to Y17F (64 outputs) with Y120 to Y15F (64 outputs) in a batch.

Point

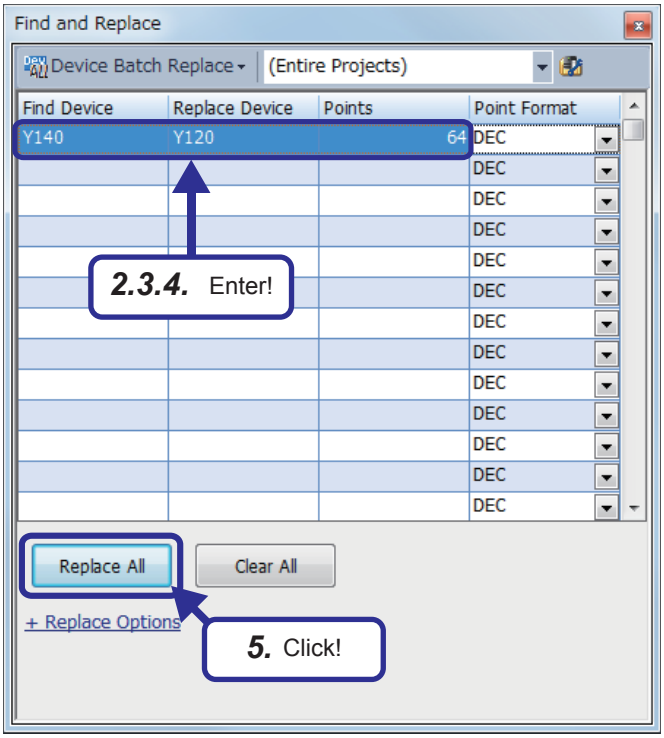
Users can also replace device numbers in a batch by specifying the number of device points in "Replace Device/Label".



(To the next page)

1. Check that "Write Mode" is active and click [Find/Replace] → [Device Batch Replace] from the menu.

(From the previous page)



2. The "Find and Replace" dialog box appears. Click "Find Device" and enter "Y140".

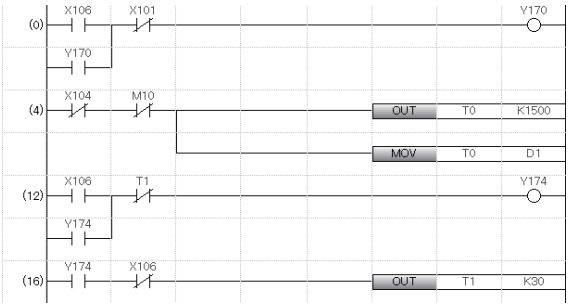
3. Click "Replace Device" and enter "Y120".

4. Click "Points" and enter "64".

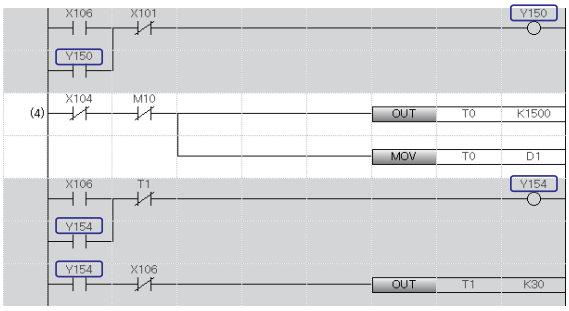
5. After the setting is completed, click the [Replace All] button.

6. Check that device numbers have been changed.

(Before)



(After)

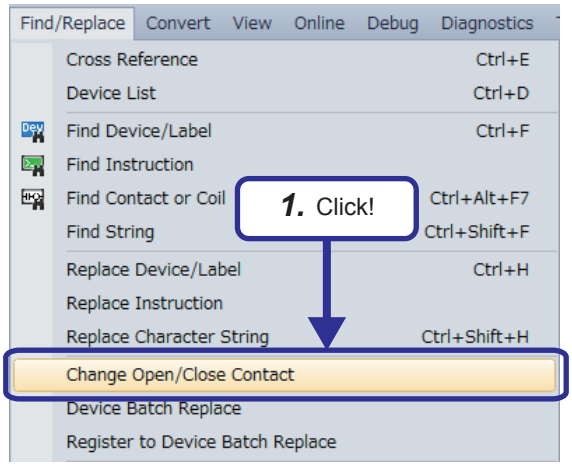


6.3.2 Changing normally open contacts ↔ normally closed contacts of specified devices in a batch

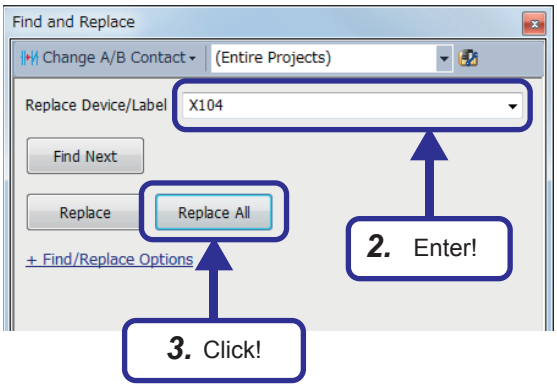
This section describes how to change normally open contacts of specified devices to normally closed contacts and vice versa in a batch.

Point

When changing normally open contacts ↔ normally closed contacts in a specified area of a program, select the area and press the "/" key.



1. Check that "Write Mode" is active and click [Find/Replace] → [Change Open/Close Contact] from the menu.



2. The "Find and Replace" dialog box appears. Click "Replace Device/Label" and enter "X104" in the list box.

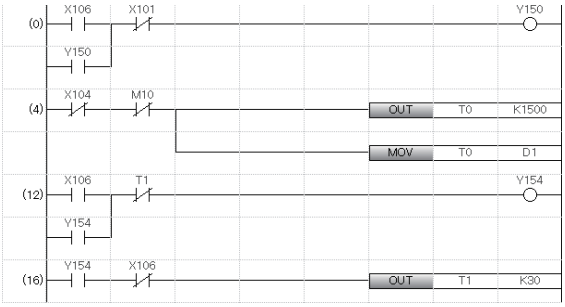
3. After the setting is completed, click the [Replace All] button.

(To the next page)

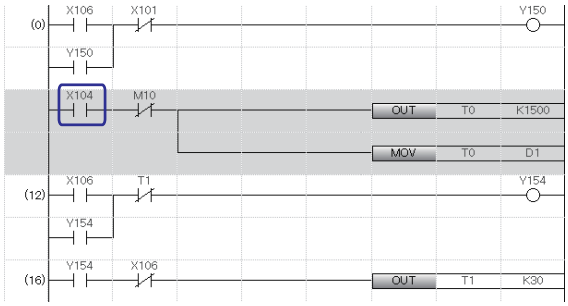
(From the previous page)

4. Check that the normally closed contact is changed to a normally open contact.

(Before)



(After)



Precautions

Before performing the exercise in Section 6.4 after this operation, do not forget to write the program in the personal computer to the CPU module.

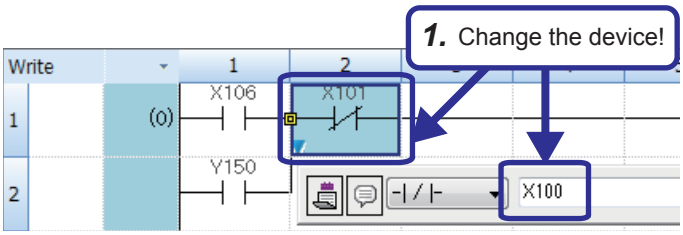
For how to write a program, refer to Section 2.5.

6.4 Online Change

Point

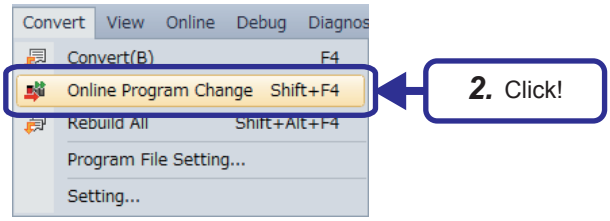
This section describes how to change a program while the CPU module is in the RUN state.

This function allows users to write a program even while the CPU module is in the RUN state.

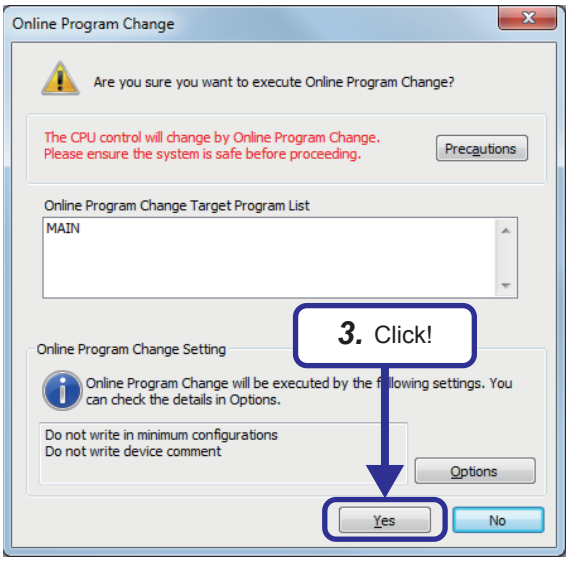


Set the CPU module to the RUN state before this operation.

1. Change the ladder.
(In this example, change "X101" to "X100".)



2. After the change, click [Convert] → [Online Program Change] from the menu.
Or, press Shift + F4.



3. The message "CAUTION" appears. Click the [Yes] button to accept the change.

4. Online change is completed.

Precautions

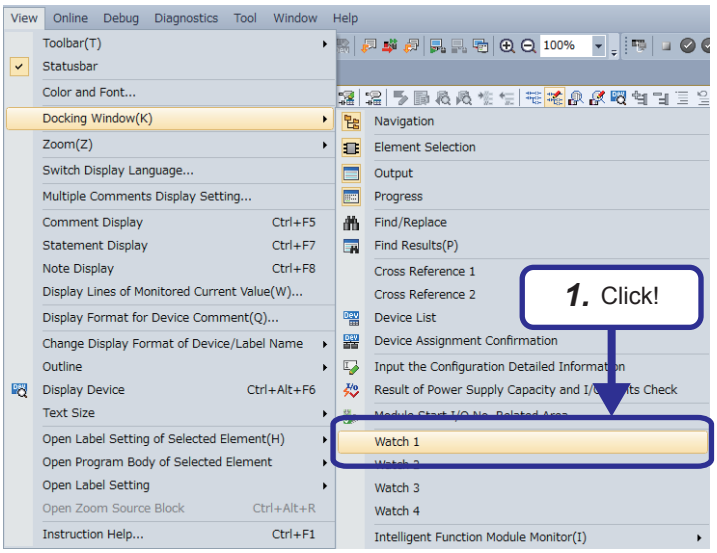
Online change cannot be executed when the program in the CPU module and the program before the modification in GX Works3 do not match. Thus, when whether the programs match or not is unclear, verify them before the modification with GX Works3, and execute the online change.

6.5 Watch Window

Point

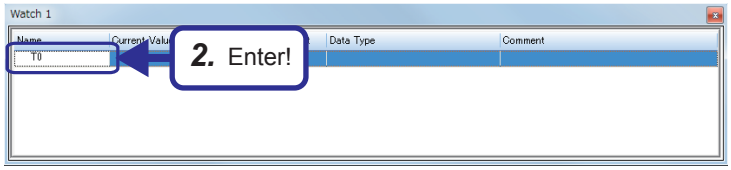
This section describes the "Watch" window where devices can be checked at once.

This section describes how to register multiple devices or labels in one window and to monitor them at the same time.

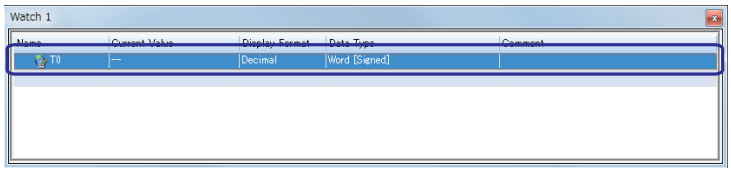


1. Click [View] → [Docking Window] → one of [Watch 1] to [Watch 4] from the menu.

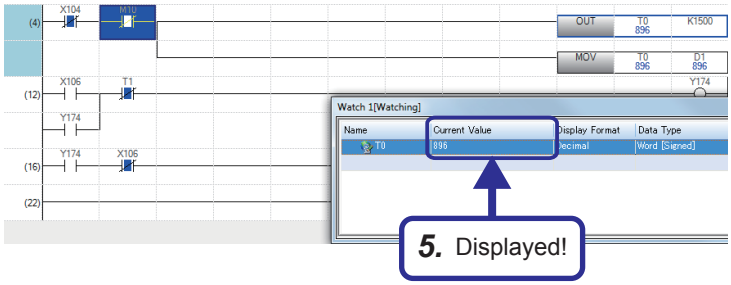
* In this example, select [Watch 1].



2. The "Watch 1" window appears. Select a row to be edited, and click "Name" and enter "T0".



3. The device or label is registered.



4. Click [Online] → [Watch] → [Start Watching] from the menu.

5. The current value of the registered device or label is displayed in the window.

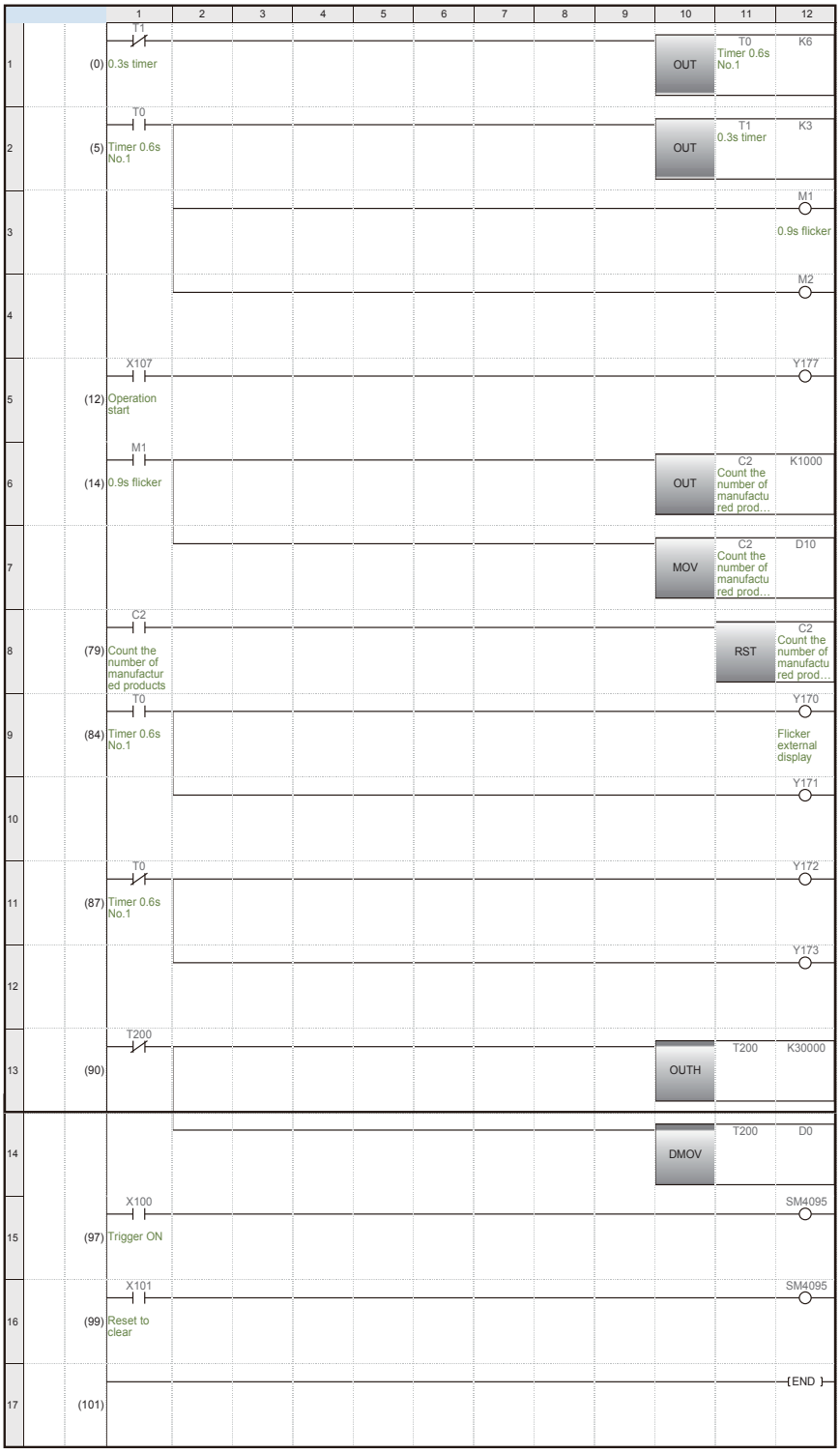
6.6 How to Create Comments

Point

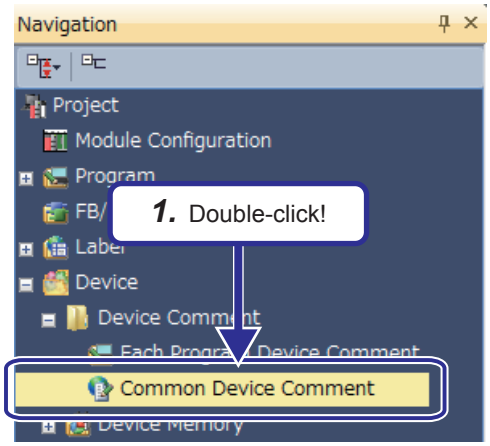
This section describes how to create comments (device comments, statements, and notes) in a program.

Project name	REX15
Program name	MAIN

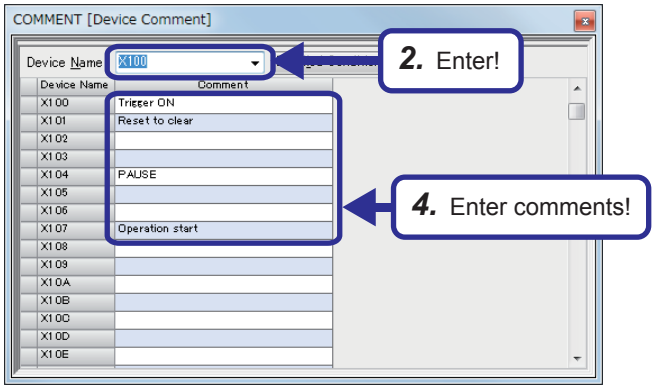
Example of a printed ladder program with comments



Creating comments



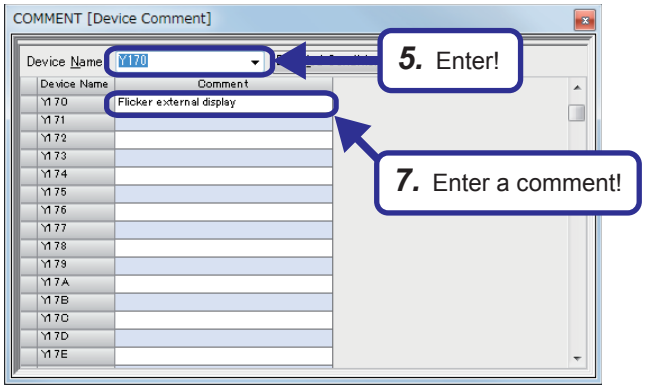
1. Click [Device] → [Device Comment] in the "Project" view and double-click [Common Device Comment] to display the "Device Comment" window.



2. Click "Device Name" and enter "X100" in the list box.

3. Press the Enter key.

4. Click each of comment areas and enter comments as shown on the left.



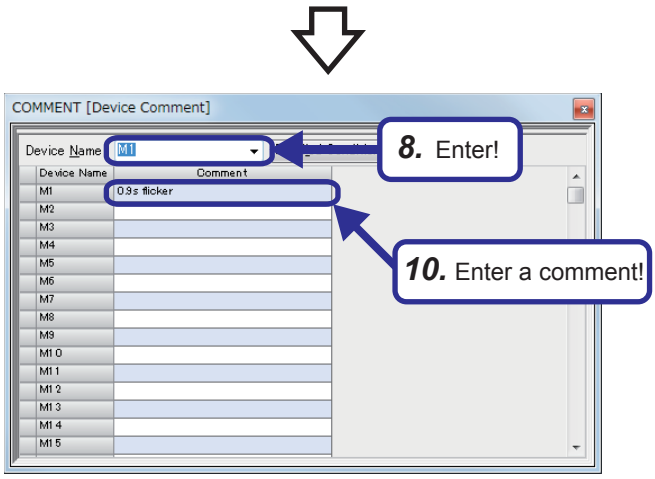
5. Click "Device Name" and enter "Y170" in the list box.

6. Press the Enter key.

7. Click a comment area and enter a comment as shown on the left.

(To the next page)

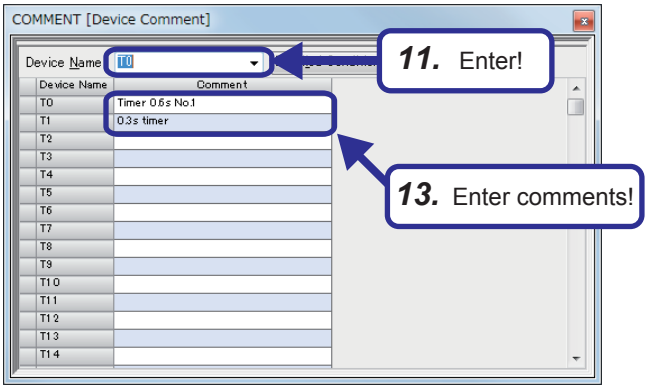
(From the previous page)



8. Click "Device Name" and enter "M1" in the list box.

9. Press the Enter key.

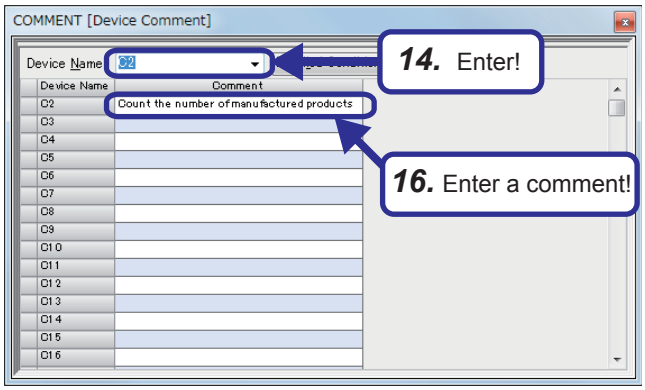
10. Click a comment area and enter a comment as shown on the left.



11. Click "Device Name" and enter "T0" in the list box.

12. Press the Enter key.

13. Click each of comment areas and enter comments as shown on the left.



14. Click "Device Name" and enter "C2" in the list box.

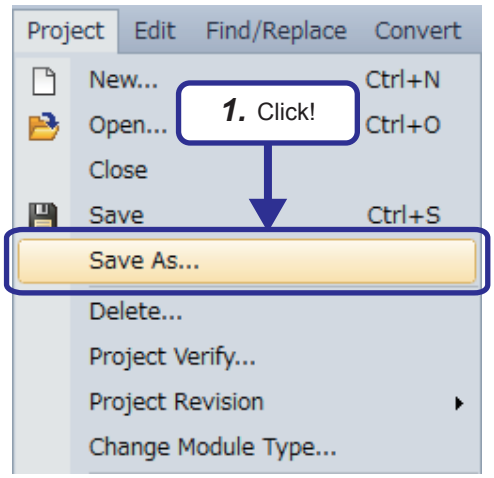
15. Press the Enter key.

16. Click a comment area and enter a comment as shown on the left.

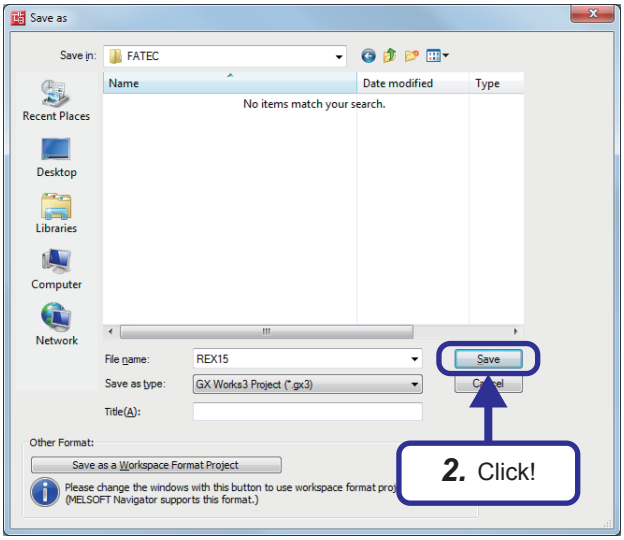
Point

Comments are used for indicating the function or application of each device. Up to 1024 characters can be entered in a comment.

Saving comments

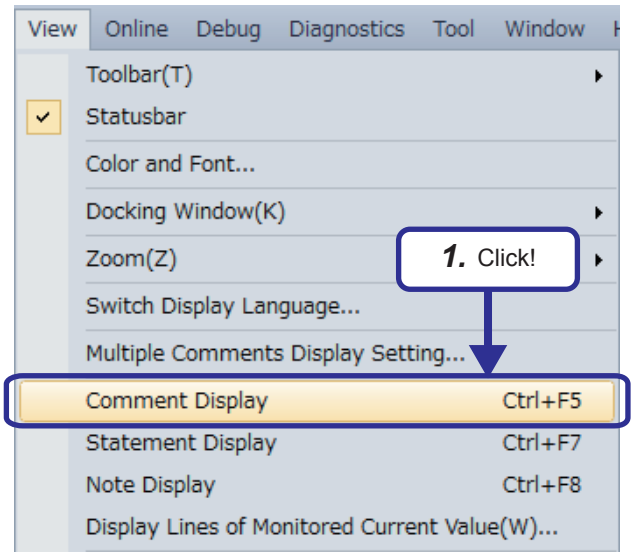


1. Click [Project] → [Save As] from the menu.



2. The "Save as" dialog box appears. Specify the save destination and a project name and click the [Save] button.

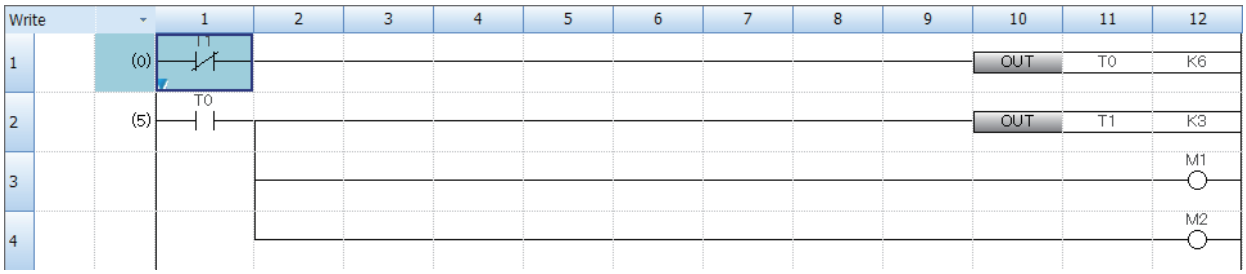
Displaying a ladder with comments in windows of GX Works3



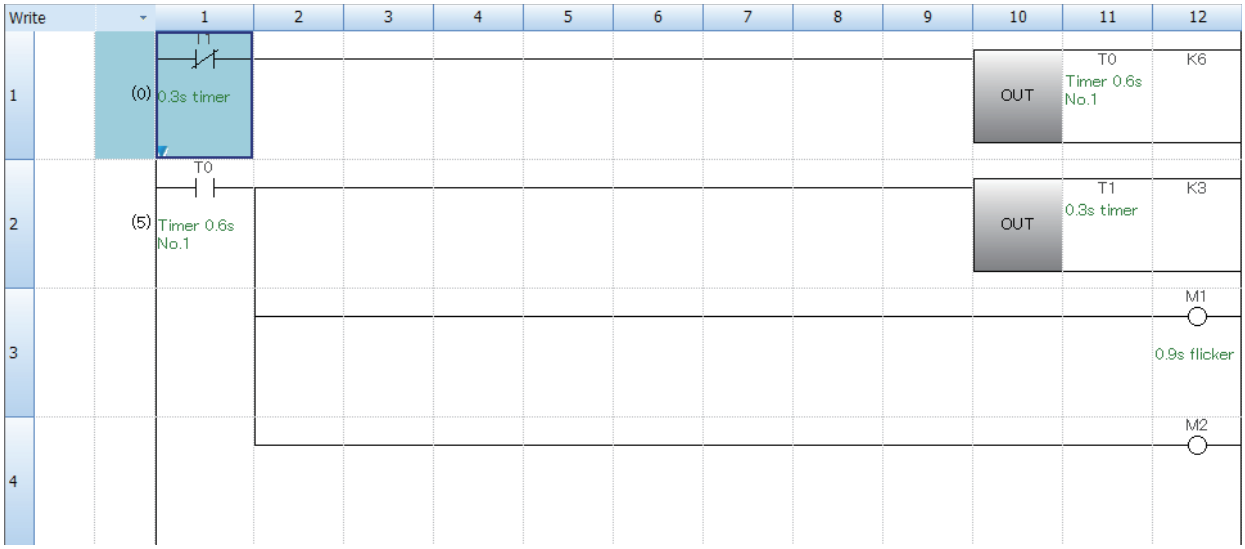
1. Click [View] → [Comment Display] from the menu.

2. The ladder program is displayed with comments.

(When comments are not displayed)



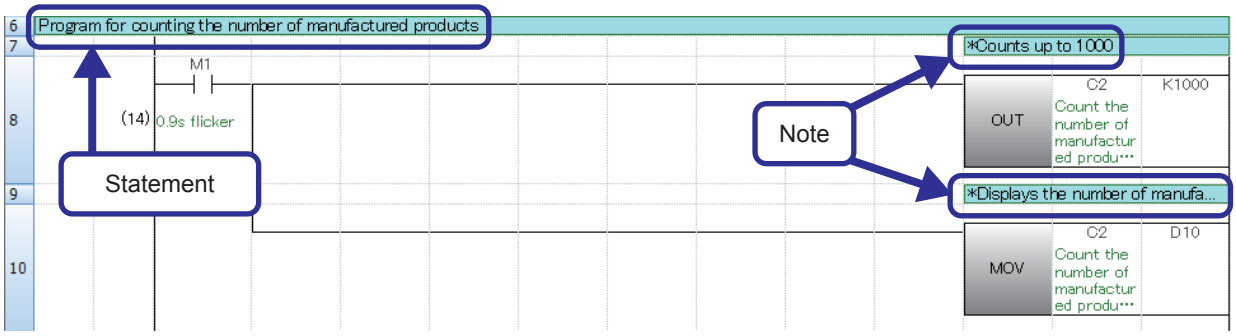
(When comments are displayed)



Point

As well as device comments, statements and notes can be created in a ladder.

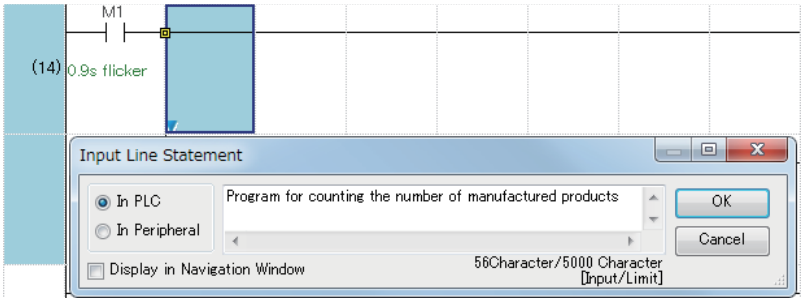
- Statement: A comment that describes the function or application of a ladder block
- Note: A comment that describes the function or application of an output or instruction



- Creating statements

  Click and double-click a ladder block where a comment is to be created.

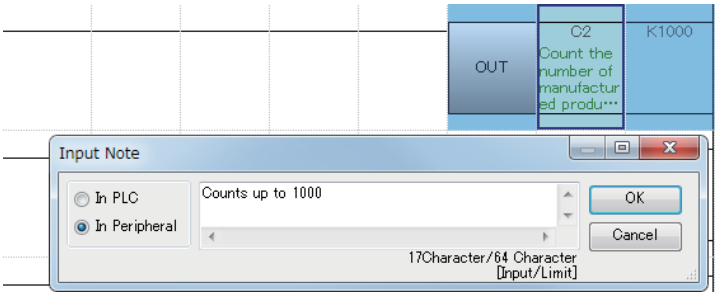
  The "Input Line Statement" dialog box appears. Enter a comment and click the [OK] button.



- Creating notes

  Click and double-click an output or instruction where a comment is to be created.

  The "Input Note" dialog box appears. Enter a comment and click the [OK] button.



- Statements and notes are classified into two categories: "In PLC" and "In Peripheral".

Category	Type	Description
In PLC	• Line statement • P statement • I statement • Note	Statements and notes can be stored in a CPU module. This type of comments uses the following number of steps. (Assumed that only one-byte characters are entered. Values after the decimal point are rounded up.) • 2 + Number of characters ÷ 2 (steps)
In Peripheral	• Line statement • P statement • I statement • Note	Statements and notes cannot be stored in a CPU module. (Only position information is stored.) Statements and notes need to be stored in a peripheral. One line consumes one step. A text that has been entered is automatically preceded by an asterisk "*".

MEMO

7 NEW FUNCTIONS OF MELSEC iQ-R/GX Works3

7.1 Features of MELSEC iQ-R

Productivity

■Newly-developed high-speed system bus that greatly shortens takt time

The newly-developed high-speed system bus (40 times as fast as our conventional products) greatly speeds up the data communication among multiple CPU modules and the large-capacity data communication with network modules. This feature maximizes the performance and functions of the MELSEC iQ-R series.

■Multiple CPU system to realize a high-accuracy motion control

The cycle of data exchange between a programmable controller CPU and a Motion CPU has been speeded up (approximately 4 times as fast as our conventional products), realizing a high-accuracy motion control.

■Synchronization function to realize high-accuracy processing

The inter-module synchronization function operates intelligent function modules and I/O modules in synchronization with the program execution timing of a programmable controller CPU or a Motion CPU, realizing a high-accuracy control of systems and devices.

In addition, the CC-Link IE Field Network or SSCNET III/H synchronous communication synchronizes the operation timing of nodes on the network. This feature reduces variations caused by the network transmission delay time, allowing users to establish a stable system.

Engineering

Users can reduce development costs by intuitively programming with GX Works3.

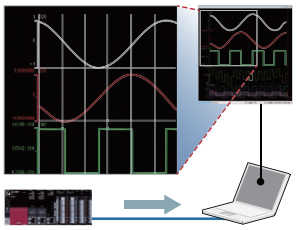
For details on GX Works3, refer to Page 7 - 5 Functions of GX Works3.

Maintenance

The MELSEC iQ-R products are equipped with the preventive maintenance to prevent troubles from occurring and various maintenance functions for quickly recovering the system at occurrence of troubles to shorten downtime, improve the productivity, and maintain the quality of manufactured products.

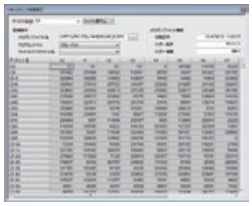
■Collecting production information of production processes

(1) Users can monitor values in specified devices in real time at desired intervals or timing. (CPU module)

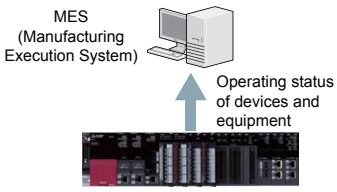


Realtime monitor

(2) When an error has occurred in a system, users can save device data in a batch and check the status at occurrence of the error with the data on the device monitor window. (CPU module)

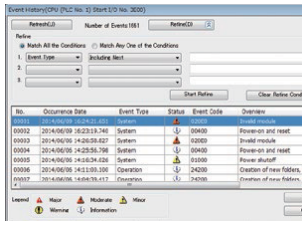


(3) Because data can be directly written to the database in an upper system, users can collect data such as the operating status of devices and equipment for improvement activities before occurrence of troubles. (MES interface module)



■Operation/error information history to solve troubles quickly

Users can check and save the history of events such as writing of programs, occurrence of errors, and power-off in a list. This feature enables users to quickly detect troubles caused by operation mistakes.



List of event history data

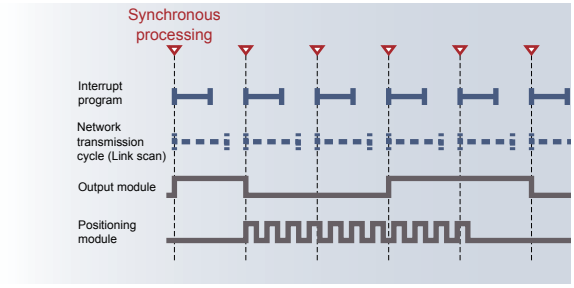
Quality

■Improving the reliability of a production system

MELSEC iQ-R series products have passed our strict quality evaluation tests implemented in various industrial scenes, such as EMC (ElectroMagnetic Compatibility) tests, LSI tests, temperature tests, vibration tests, and HALT tests.
QR codes are used to manage the quality information at the time of manufacturing and to offer high-quality products to our customers.

■Improved quality of products to be manufactured

- With the inter-module synchronization function, users can synchronize the execution of an interrupt program and the network transmission cycle (link scan).
- This function reduces variations of data communication (network transmission delay time) between a programmable controller and devices on a network, improving the quality of products to be manufactured.



Connectivity

With SLMP[*1], users can perform seamless data communication from the production control level of an entire automation system to the device level such as sensors, without considering layers of the network.

■Seamless information linkage

With SLMP, users can access the production control system, programmable controllers, and other devices seamlessly in an identical manner without considering layers and boundaries of the networks. Users can easily monitor devices and collect data from anywhere.

■Simple connection to external devices with MELSOFT Library

With the predefined protocol support function of GX Works3, only selecting a protocol to be used and data to be sent or received enables the simple communication with external devices, such as vision sensors and temperature controllers. Users do not need to create programs for communication, reducing the man-hour for developing programs.

*1 SLMP (Seamless Message Protocol): Simple, common, client-server type protocol that enables users to perform the data communication without considering layers and boundaries of networks among Ethernet products and CC-Link IE-compatible devices

Security

The MELSEC iQ-R series products are equipped with strong security functions such as the security key authentication to protect programs and an IP filter to prevent unauthorized accesses to control system.

■Security authentication to protect project data

The security key authentication function locks programs so that they cannot be opened in the personal computer where no security key has been registered.

■IP filter function

The IP filter function registers IP addresses of devices that can access the CPU module to prevent accesses from devices other than the registered ones. This function reduces risks of programs being hacked by an outsider, unauthorized modifications, or others.

Compatibility

Properties such as MELSEC-Q series programs used in the existing system and various modules can be utilized for the MELSEC iQ-R series.

■Utilizable program properties

MELSEC-Q series programs can be converted[*1] into the ones for the MELSEC iQ-R series and utilized.
Stored program properties can be effectively used to reduce the man-hours for developing programs and to shorten the development period.

*1 Part of the programs cannot be converted. For details, refer to the GX Works3 Operating Manual.

■Utilizable modules

With dedicated extension base units, users can use the MELSEC-Q series modules in the MELSEC iQ-R series system.
(For details on the Q series modules that can be used in the MELSEC iQ-R system, refer to the Module Configuration Manual.)
Users can reduce costs required for spare parts or others and introduce the high-performance MELSEC iQ-R series.

7.2 Differences Between the MELSEC-Q Series and the MELSEC iQ-R Series

For differences between the MELSEC-Q series and the MELSEC iQ-R series, refer to the TECHNICAL BULLETIN No. FA-A-0171.

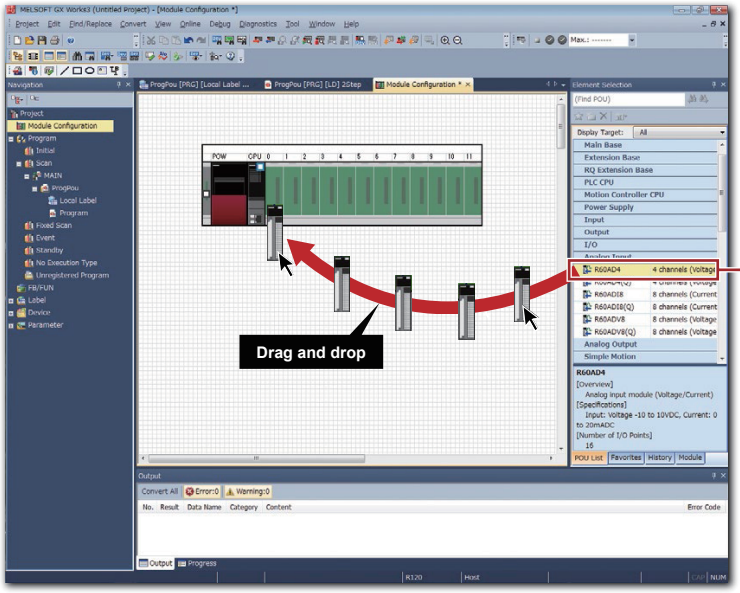
7.3 Functions of GX Works3

GX Works3, an engineering tool, has the functions that facilitate users to create projects (system configuration, programming) and perform maintenance (debugging, diagnostics, and management).

System design

■Simple system design

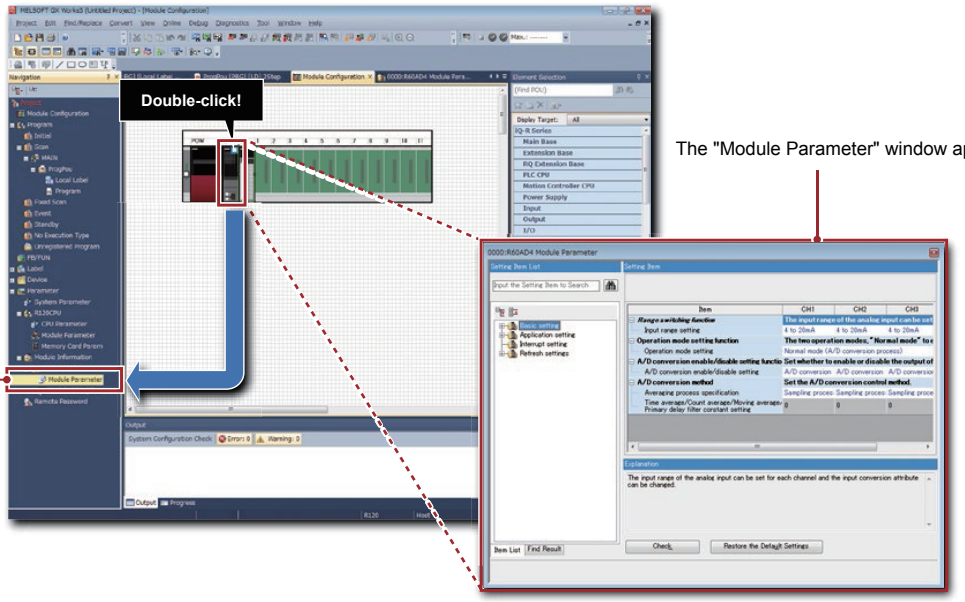
Creation of a project starts from system design. GX Works3 helps users to easily design a system. Users can create a module configuration only by selecting program elements and dragging and dropping them into the "Module Configuration" window of GX Works3.



Add modules by dragging and dropping them from the "Element Selection" window.

■Easy creation of module parameters

Module parameters can be automatically created in the creation of a module configuration. Users can create module parameters of a project only by double-clicking a module on the "Module Configuration" window. Related parameters are displayed as the work window and parameters can be set.



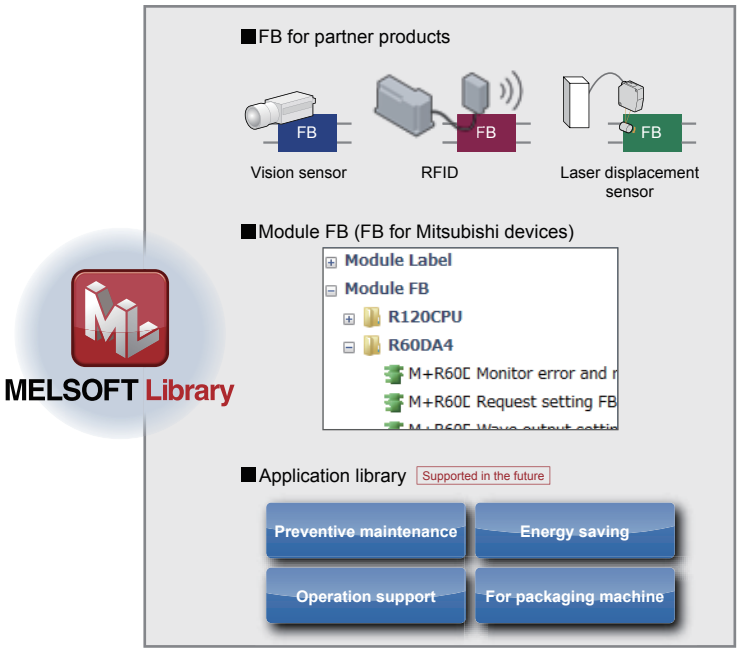
The "Module Parameter" window appears.

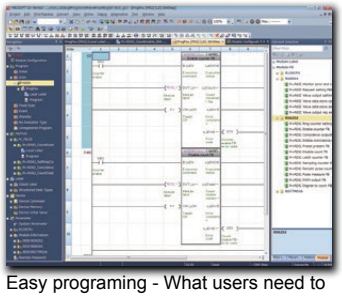
"Module Parameter" is added in the "Navigation" window.

Programming

■MELSOFT Library prepared to reduce man-hours

A variety of libraries (FBs for partner products/module FBs/application libraries or others) are available in MELSOFT Library. Using FBs reduces the man-hours for developing programs.

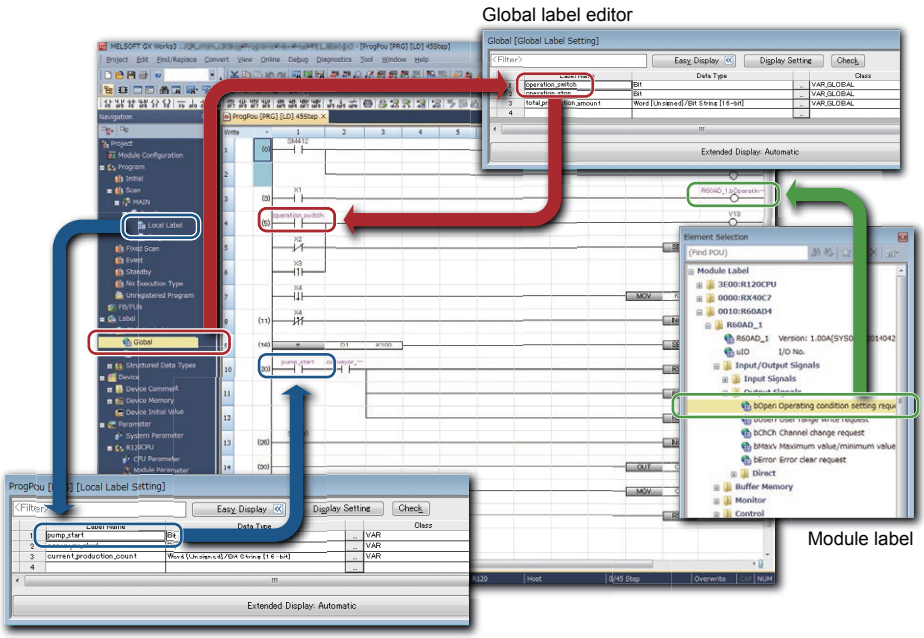




Easy programing - What users need to do is to select parts!

Labels prepared to reduce loads

GX Works3 allows users to use global labels, local labels, and module labels. Global labels can be shared and used in multiple programs and other MELSOFT software applications. Local labels can be used in a program and FB where the labels are registered. Module labels have information about I/O signals and the buffer memory areas of each intelligent function module. Thus, users can create a program without considering I/O addresses and buffer memory addresses.



Maintenance

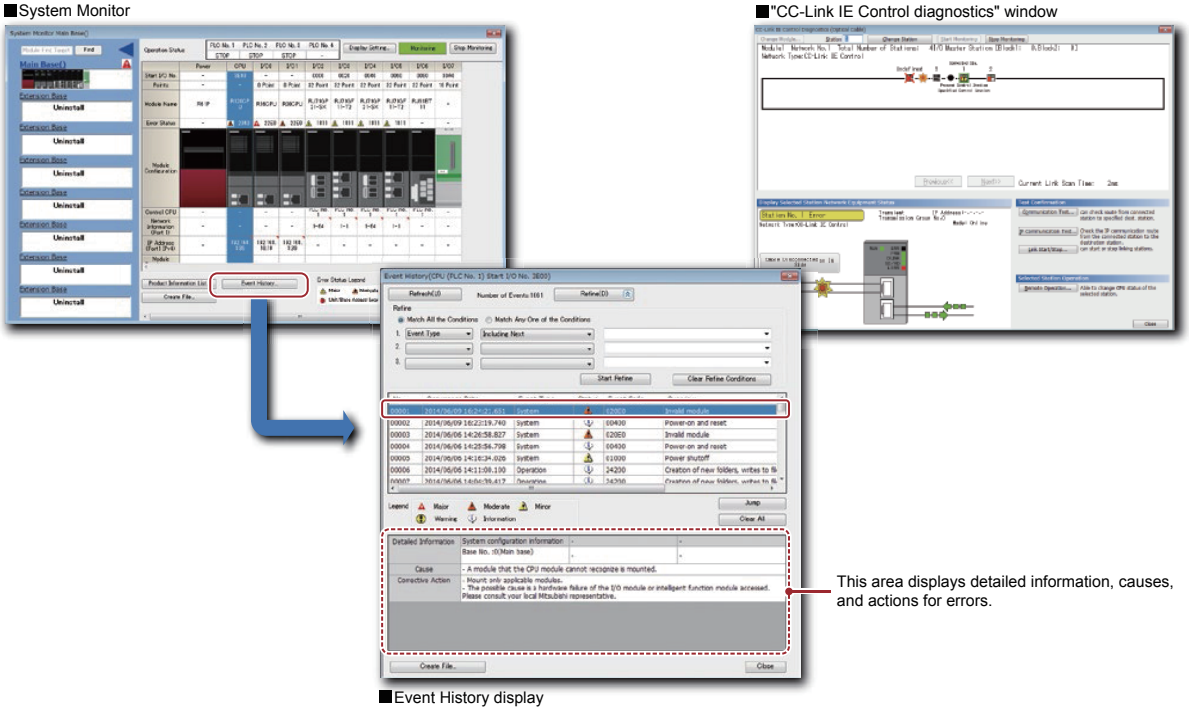
■Diagnostic function

Users can easily identify faulty areas with the diagnostic function of GX Works3. Users can check a module configuration and error status in the system with the system monitor. Users can check errors that have occurred and operations performed in each module in chronological order with the event history display.

Because faulty areas on the network are graphically displayed in the various network diagnostics, downtime can be shortened.

■System Monitor

■"CC-Link IE Control diagnostics" window



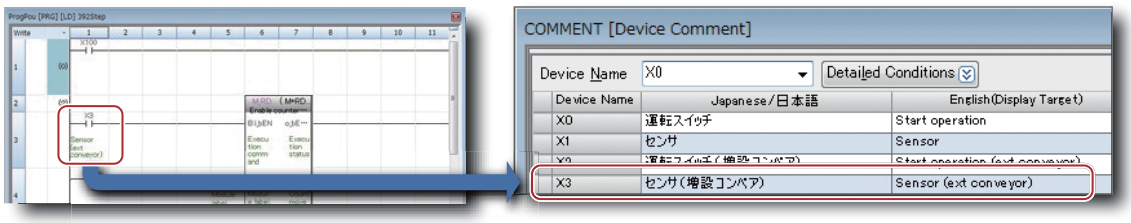
■Event History display

■Language switching

Users can switch the language of the menu display or others in GX Works3. Users can create comments in each language and easily switch the display. Thus, when foreign engineers perform maintenance, they can easily understand programs only by switching the language of comments to their native language, helping their operations.

The display language can be easily switched in one package.

The comment language can be switched.



MEMO

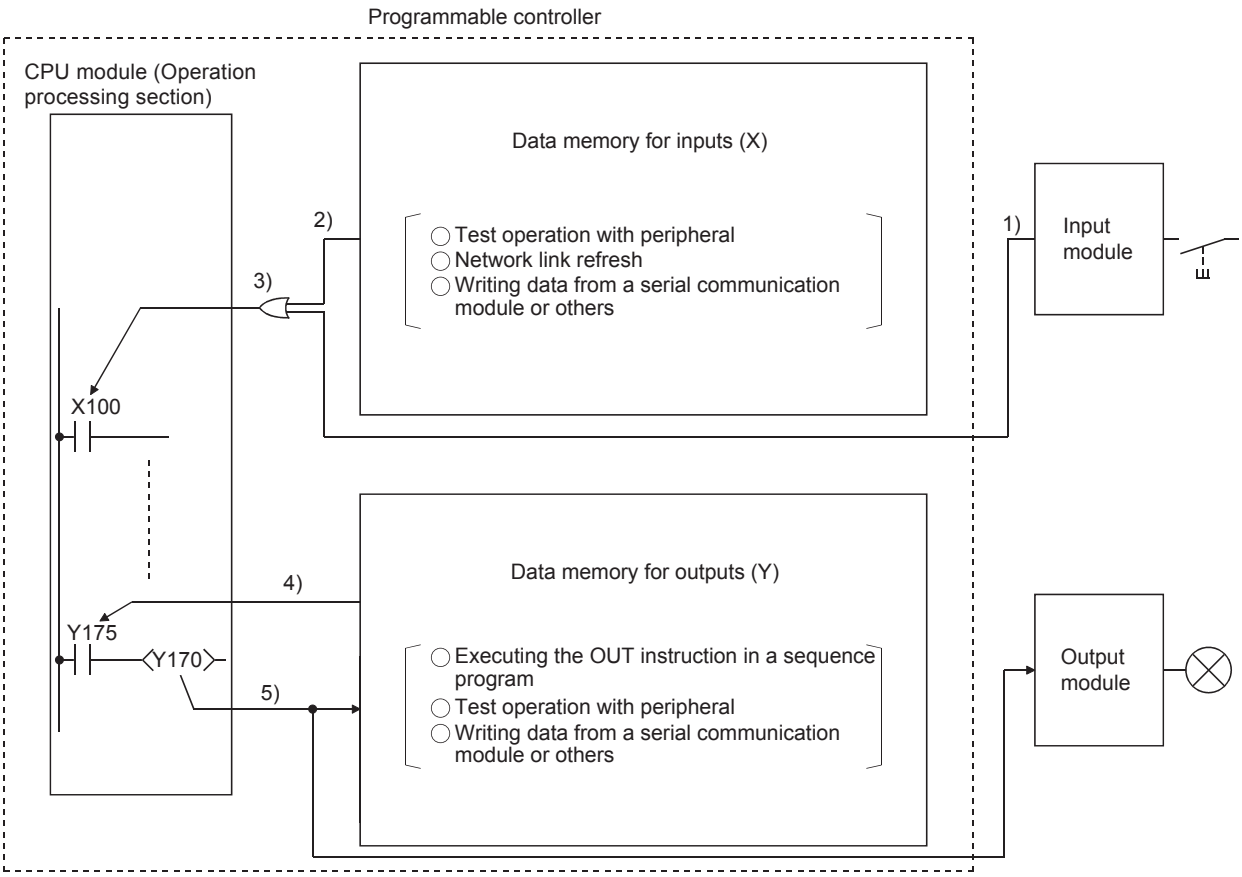
APPENDICES

Appendix 1 I/O Control Mode

The CPU module supports two types of I/O control modes: direct mode and refresh mode.

Appendix 1.1 Direct mode

In the direct mode, input signals are loaded into a programmable controller every time they are input and used as input information.
Operation results of a program are output to the data memory for outputs and an output module.
The following figure shows the flow of I/O information in the direct mode.



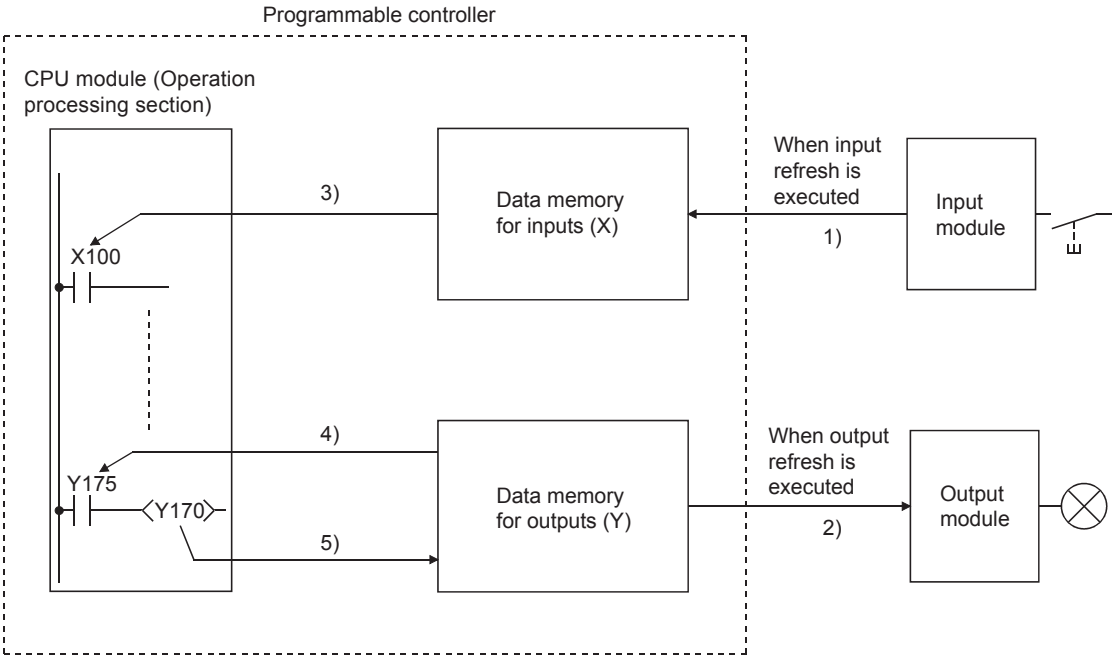
- When an input contact instruction is executed
  An OR operation is executed on the input information 1) of the input module and the input information 2) in the data memory.
  The result is used as input information 3) and the sequence program is executed.
- When an output contact instruction is executed
  The output information 4) is read from the data memory and the sequence program is executed.
- When the output OUT instruction is executed
  The operation result of the sequence program 5) is output to the output module, and stored in the data memory for outputs (Y).
- When the QCPU inputs and outputs 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 caused in an input module are loaded in a batch into the data memory for inputs in a programmable controller CPU before the execution of program every scan. The data in the data memory for inputs is used for executing an operation.

Operation results of an output (Y) program are stored in the data memory for outputs. After the END instruction is executed, the data in the data memory for outputs is output in a batch to an output module.

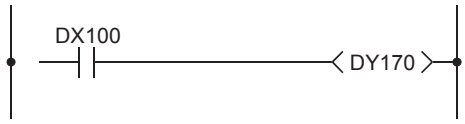
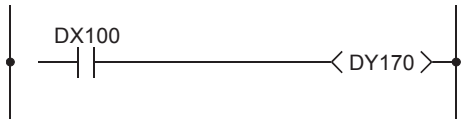
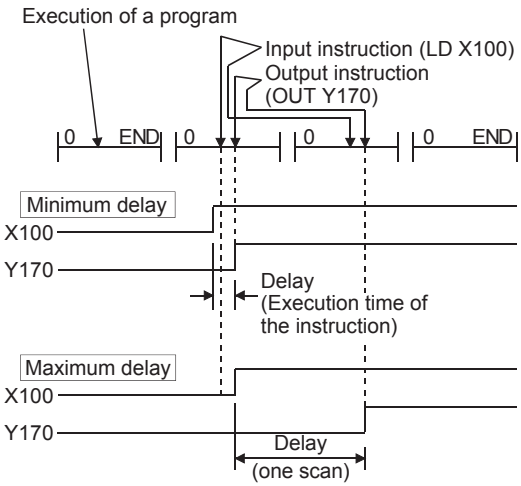
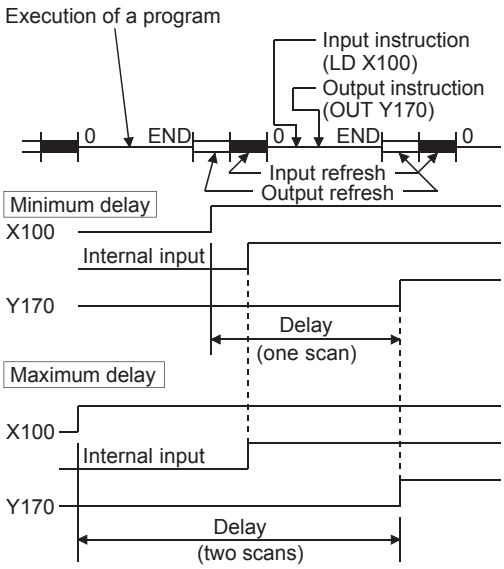
The following figure shows the flow of I/O information in the refresh mode.



- Input refresh
  Input information is read from the input module in a batch 1) before the execution of the step 0, and stored in the data memory for inputs (X).
- Output refresh
  The data in the data memory for outputs (Y) 2) is output to the output module in a batch before the execution of the step 0.
- When an input contact instruction is executed
  Input information is read from the data memory for inputs (X) 3), and the sequence program is executed.
- When an output contact instruction is executed
  Output information is read from the data memory for outputs (Y) 4), and the sequence program is executed.
- When the output OUT instruction is executed
  The operation result of the sequence program 5) is stored in the data memory for outputs (Y).

Appendix 1.3 Comparisons between direct mode and refresh mode

The following table shows the differences between the direct mode and the refresh mode using a ladder program in which the output Y170 turns on when the input X100 turns on as an example.

Item	Direct mode	Refresh mode
1. Ladder example		
2. Response lag from when the input signal turns on to when the output signal turns on	 • The delay time ranges from zero (only execution time of the instruction) to one scan. • The delay time is zero to one scan.	 • The delay time ranges from one to two scans. • The delay time is one to two scans.
3. Execution time of the I/O instruction	• The direct mode needs the time longer than the one for the refresh mode because the programmable controller accesses I/O modules.	• The refresh mode needs the time shorter than the one for the direct mode because the programmable controller accesses the data memory.
4. Scan time	• When the execution of the I/O instructions delays, the scan time becomes longer. • The actual scan time is the execution time of the program.	• When the I/O instructions are executed quickly, the scan time becomes shorter. • The actual scan time is the total of the execution time of a program, input transfer time, and output transfer time.

A

Appendix 2 List of Special Relay Areas

Special relay (SM) is an internal relay whose application is fixed in the programmable controller. Thus, it cannot be used in the same way as other internal relay areas used in a sequence program.
However, users can turn on or off the special relay as needed to control the CPU module or remote I/O module.

The following table lists the items in the list.
For details on special relay areas, refer to the MELSEC iQ-R CPU Module User's Manual (Application).

Item	Description
No.	Special relay number
Name	Special relay name
Data stored	Data stored in the special relay and its meaning
Details	Detailed description of the data stored
Set by (setting timing)	Set side of data (system or user) and timing when data is set by the system <Set by> • S: System • U: User (program, engineering tool, GOT, or other testing operations from external device) • U/S: User and system <Set timing> • Every END: Data is set every time END processing is performed. • Initial: Data is set when initial processing is performed (e.g. powering on the system, changing the operating status from STOP to RUN). • Status change: Data is set when the status is changed. • Error: Data is set when an error occurs. • Instruction execution: Data is set when an instruction is executed. • Request: Data is set when requested by a user (using the special relay). • Writing: Data is set when a user performs a writing operation. • During END: Data is set when END processing is performed. • Power-on to RUN or STOP to RUN: Data is set when the operating status changes from power-on to RUN or from STOP to RUN.

Point

Do not change the data set by the system in a program or by a device test. Doing so may result in system down or communication failure.

Appendix 3 List of Special Register Areas

Special register (SD) is an internal register whose application is fixed in the programmable controller. Thus, it cannot be used in the same way as other internal registers used in a sequence program. However, users can write data in special register areas as needed to control the CPU module or remote I/O module.
The data is stored in special register areas as binary values if not specified.

The following table lists the items in the list.
For details on special register areas, refer to the MELSEC iQ-R CPU Module User's Manual (Application).

Item	Description
No.	Special register number
Name	Special register name
Data stored	Data stored in the special register
Details	Detailed description of the data stored
Set by (setting timing)	Set side of data (system or user) and timing when data is set by the system <Set by> • S: System • U: User (program, engineering tool, GOT, or other testing operations from external device) • U/S: User and system <Set timing> • Every END: Data is set every time END processing is performed. • Initial: Data is set when initial processing is performed (e.g. powering on the system, changing the operating status from STOP to RUN). • Status change: Data is set when the status is changed. • Error: Data is set when an error occurs. • Instruction execution: Data is set when an instruction is executed. • Request: Data is set when requested by a user (using the special relay). • Switch change: Data is set when the switch of the CPU module is changed. • Card insertion/removal: Data is set when an SD memory card is inserted or removed. • Writing: Data is set when a user performs a writing operation. • During END: Data is set when END processing is performed.

Point

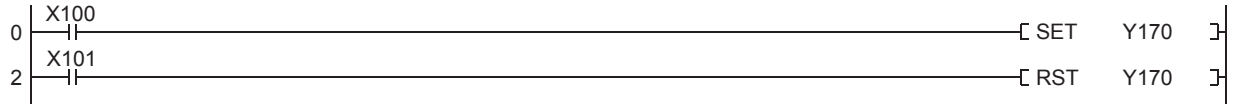
Do not change the data set by the system in a program or by a device test. Doing so may result in system down or communication failure.

A

Appendix 4 Program Examples

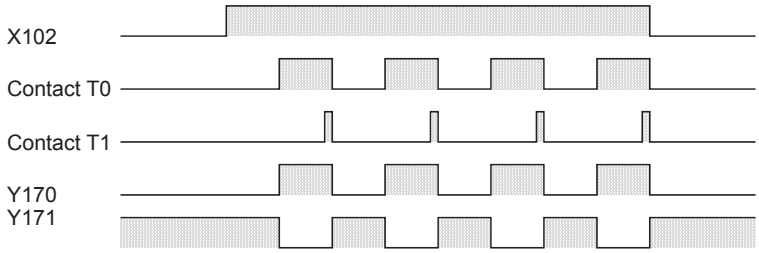
Appendix 4.1 Flip-flop ladder

1. When X100 turns on, Y170 turns on. When X101 turns on, Y170 turns off.



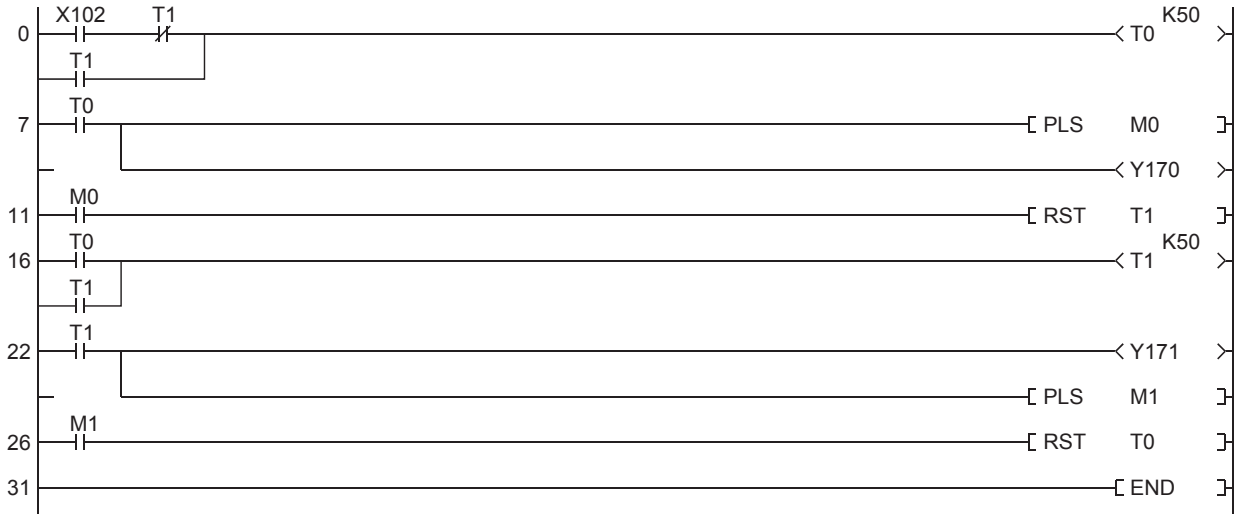
2. When X102 turns on, Y171 turns off if Y170 is on, or turns on if Y170 is off. This flip-flop operation is repeated.

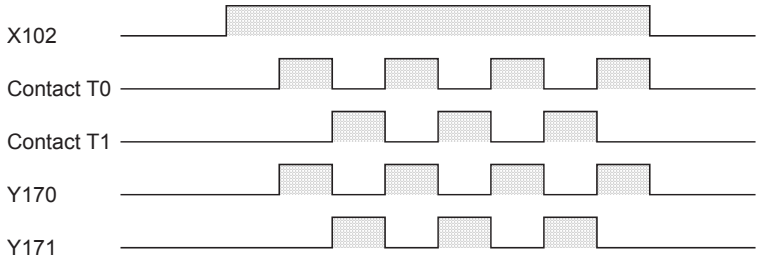
Project name	RA-16
Program name	MAIN



3. When X102 turns on, the flip-flop operation starts. In this operation, Y170 turns on if the timer T0 is on, and Y171 turns on if the timer T1 is on (Cycle: 10 seconds).

Project name	RA-17
Program name	MAIN

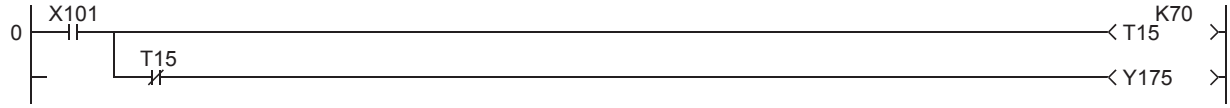


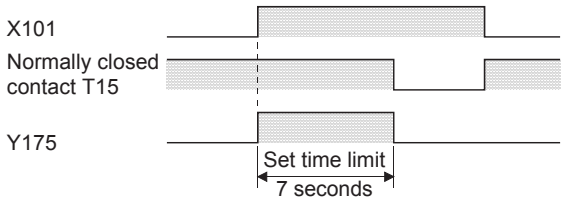


A

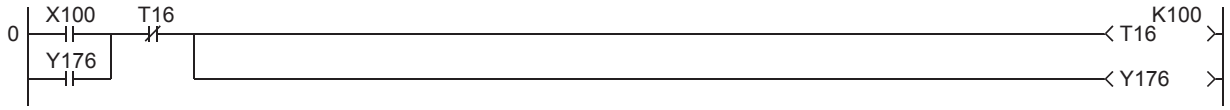
Appendix 4.2 One-shot ladder

1. Output starts and continues for a certain period of time after the input X101 turns on.
(The input ON time must be longer than the set time limit.)

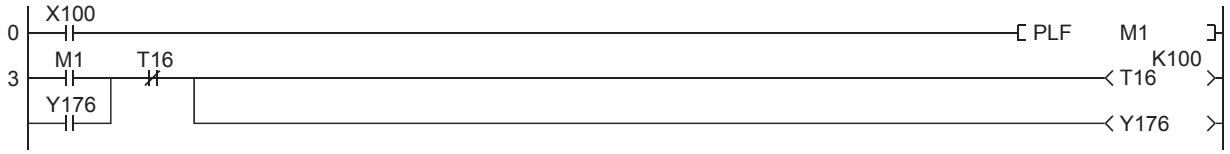


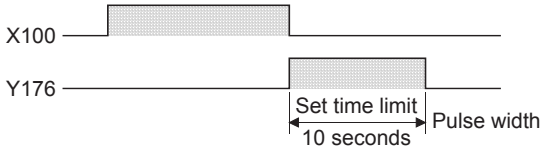


2. When the input X100 turns on momentarily, Y176 turns on for a certain period of time.



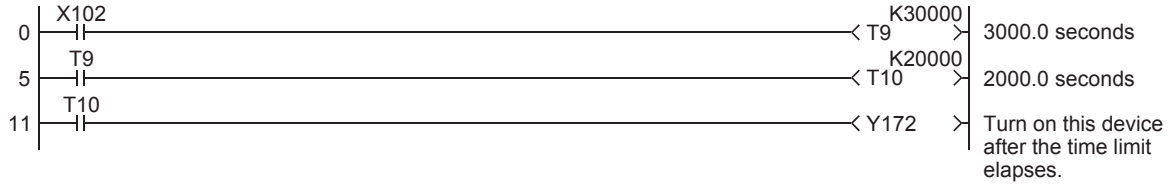
3. When the input X100 turns off, output starts and continues for a certain period of time.

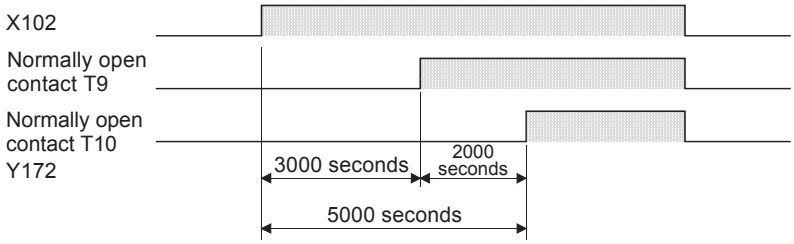




Appendix 4.3 Long-time timer

1. Arrange timers in series to obtain necessary time.

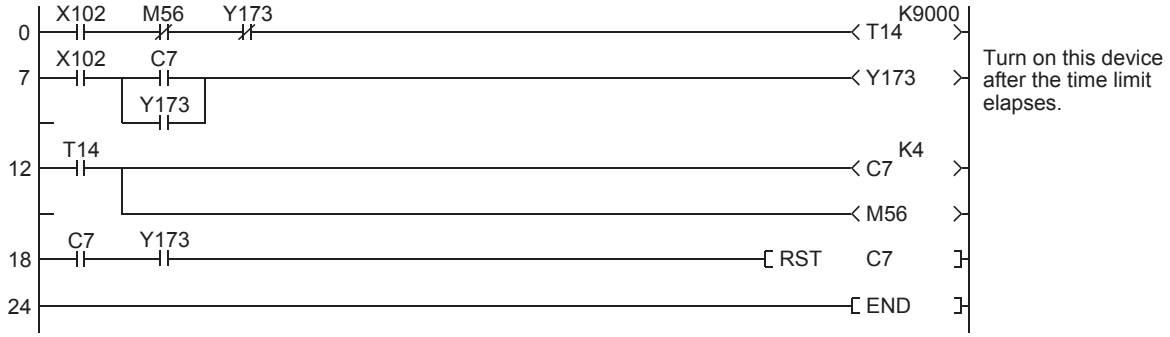


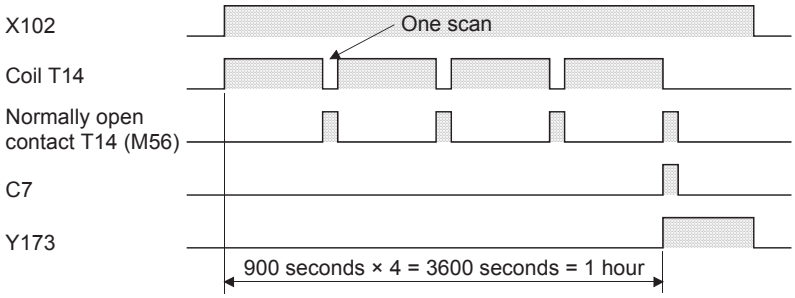


2. Use timers and counters to obtain necessary time.

Time limit of a timer × Set value of a counter = Long-time timer (Note that the accuracy of timers is accumulated.)

Project name	RA-18
Program name	MAIN





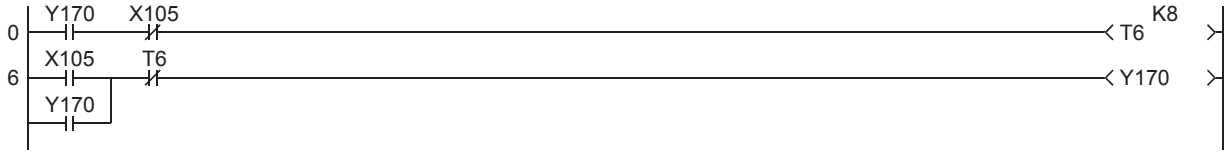
* Obtain necessary time by counting the number of timeouts of the timer T14 with the counter C7.
M56 resets T14 after a timeout. With C7, the output Y173 holds its ON state when counting is up. Y173 resets T14 and stops the subsequent time counting.

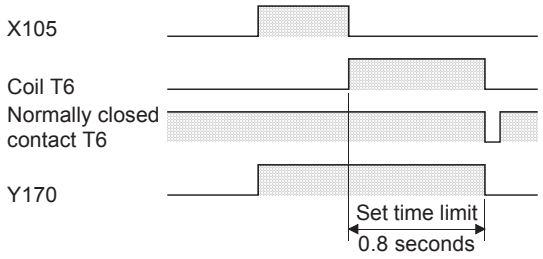
A

Appendix 4.4 Off delay timer

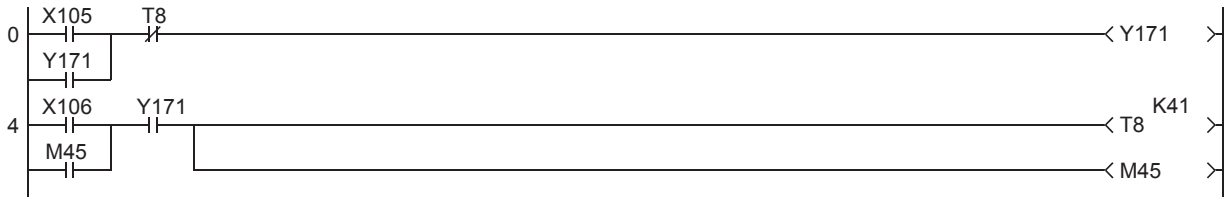
Off delay timers are not provided for the MELSEC iQ-R series. Create off delay timers as follows.

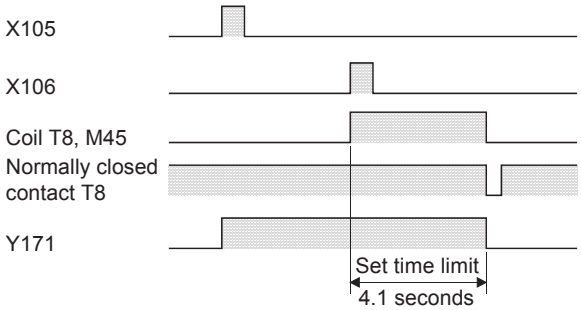
1. When X105 turns off, the timer T6 starts operating.





2. Turning on X105 momentarily sets the operation ready.
When X106 turns on momentarily, the timer T8 starts operating.



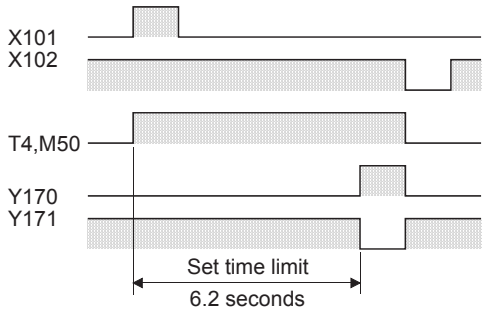


* The above ladder operates as an off delay timer by momentarily turning on the inputs X105 and X106. M45 is equivalent to a momentary contact of T8.

Appendix 4.5 On delay timer (momentary input)

An on delay timer of a programmable controller operates easily with a continuous input. The internal relay (M) is used with a momentary input.

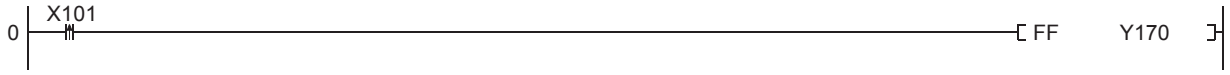
Project name	RA-19
Program name	MAIN



* The above ladder operates as an on delay timer by momentarily turning on the inputs X101 and X102.

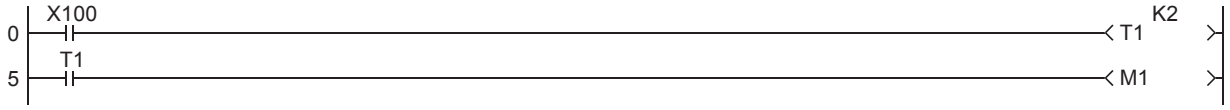
Appendix 4.6 On/off repeat ladder

In an on/off repeat ladder, Y170 turns on when X100 turns on for the first time, and turns off when X100 turns on again.



Appendix 4.7 Preventing chattering inputs

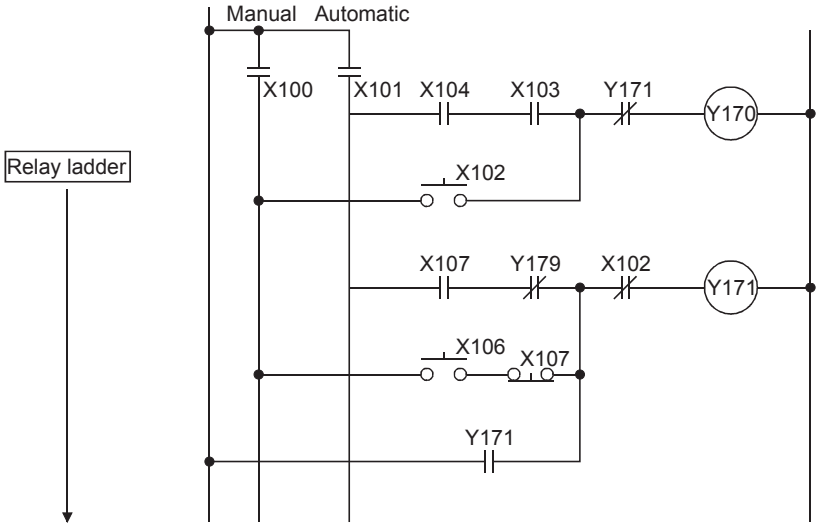
Set a timer so that it starts operating when the input remains on for 0.2 seconds.



M1 turns on when X100 remains on for 0.2 seconds or longer. To prevent chattering inputs, use M1 instead of X100.

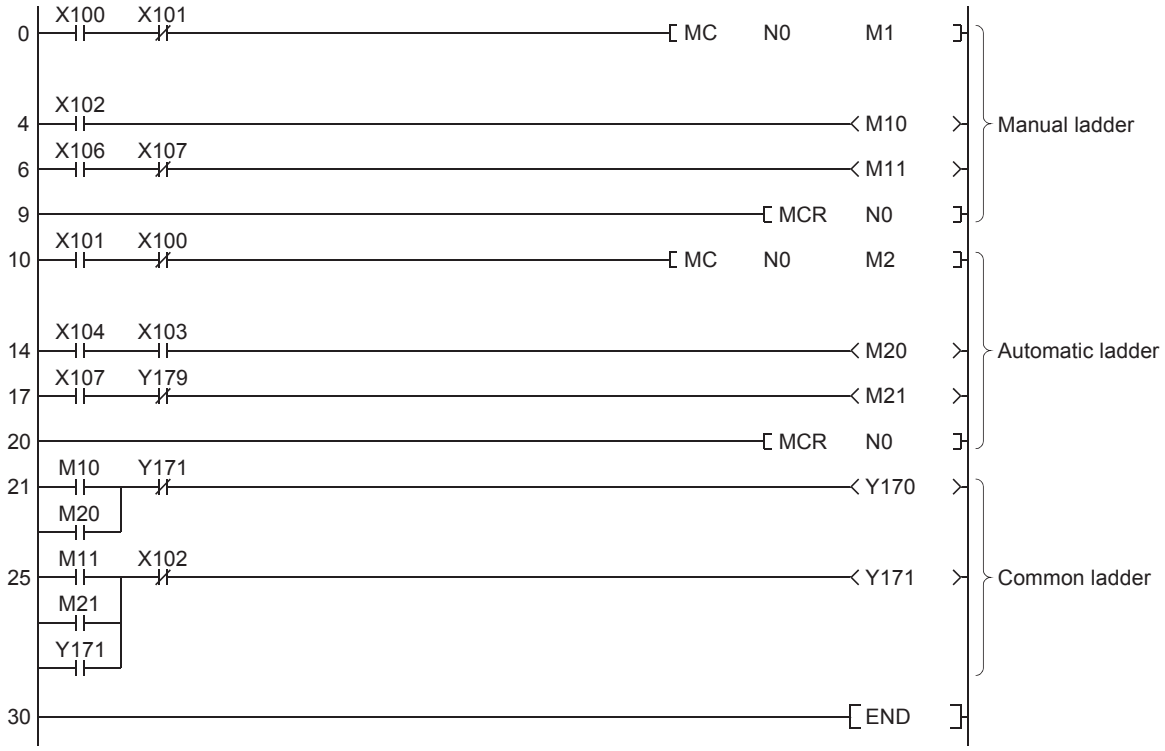
Appendix 4.8 Ladder with common lines

The following ladder cannot be used as a program for a programmable controller. Use the master control instructions (MC, MCR) in the program.



Project name	RA-1
Program name	MAIN

Sequence program with master control instructions



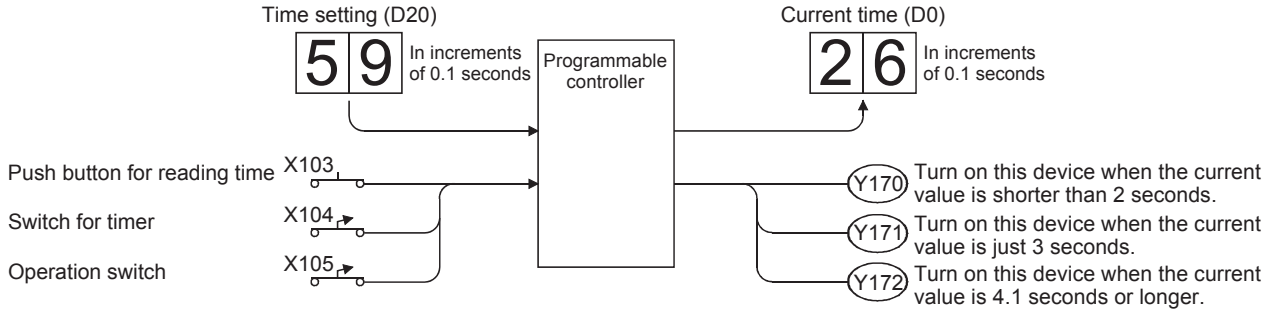
Precautions

GX Works3 displays the on/off state of a master control on the title tag on the monitor window.

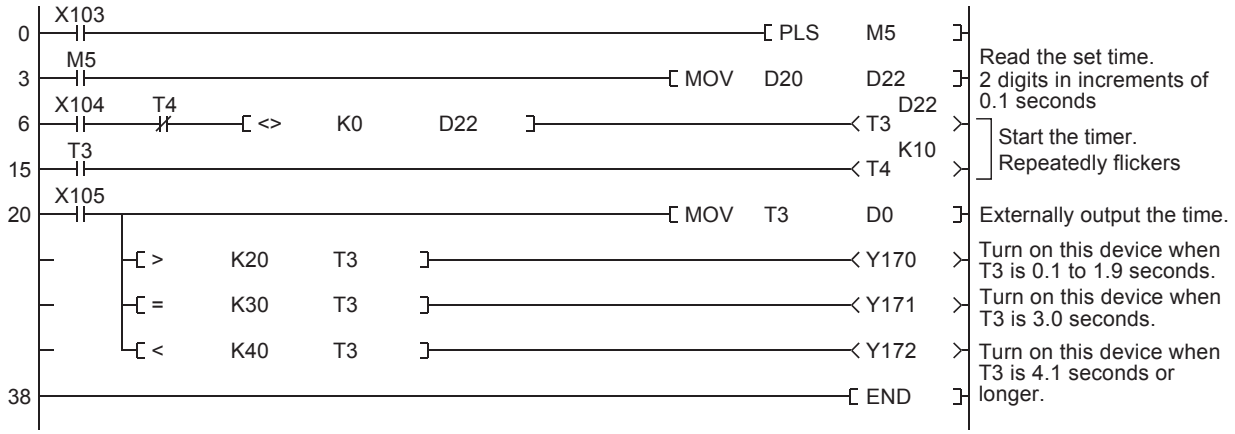
Appendix 4.9 Time control program

Set the time (two digits) with an input device to turn on outputs Y170 to Y172 with a specified time limit and display the elapsed time.

This program repeats this operation.



Project name	RA-2
Program name	MAIN

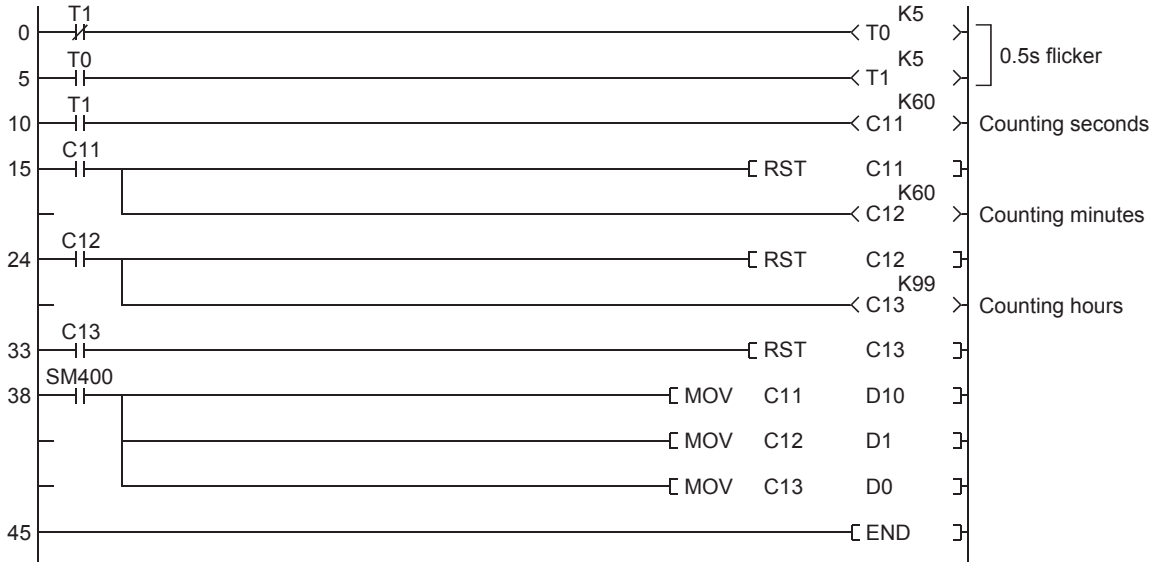


A

Appendix 4.10 Clock ladder

Output time information, such as hour, minute, and second to the initial indication device.

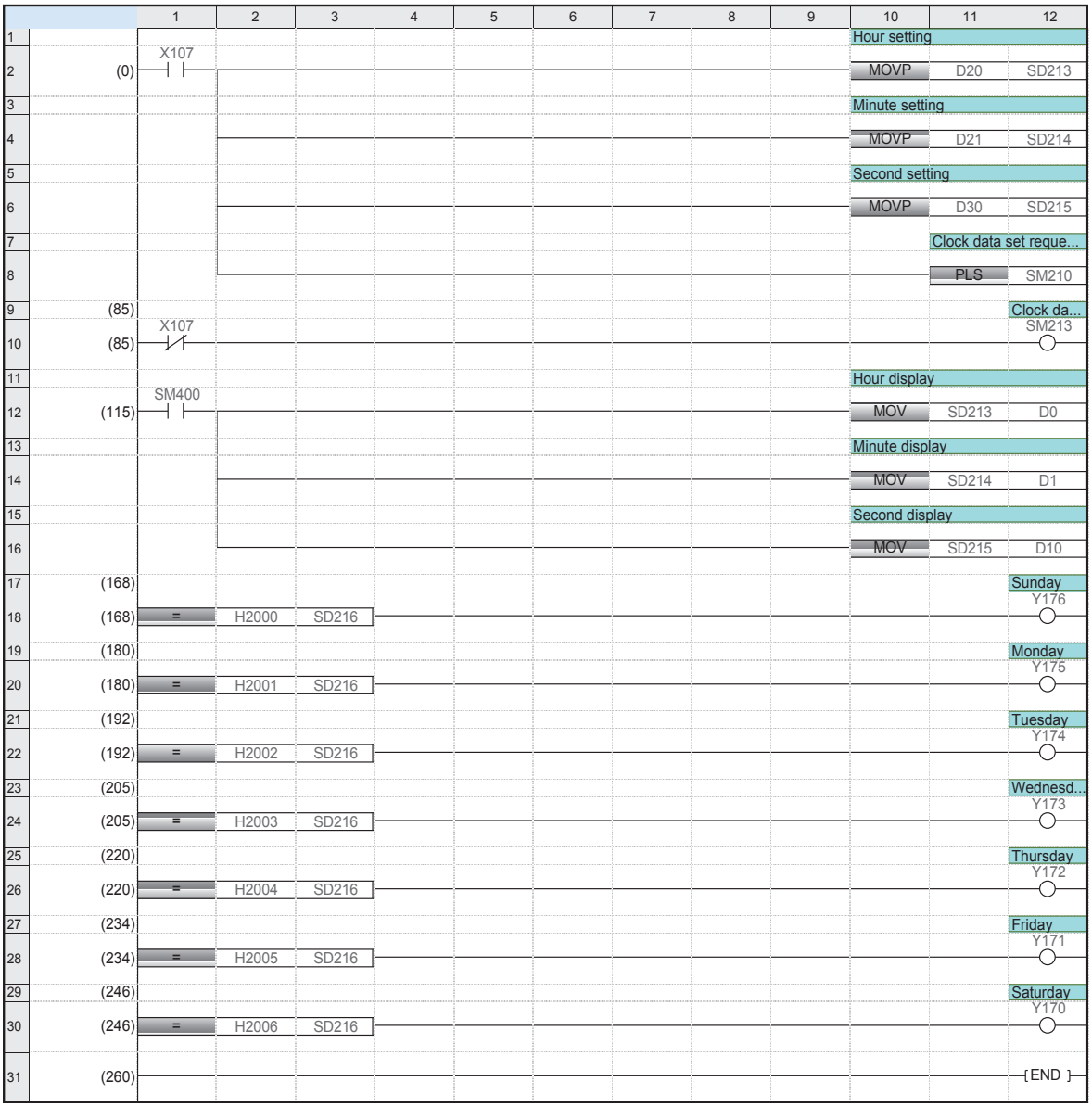
Project name	RA-3
Program name	MAIN



Clock function (supplement)

With the following ladder, the time set with GX Works3 is displayed on the MELSEC iQ-R series demonstration machine.

Project name	REX13

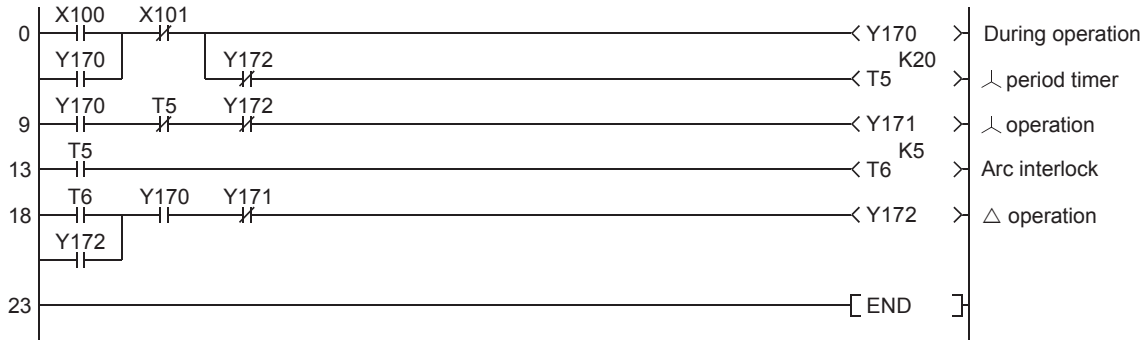


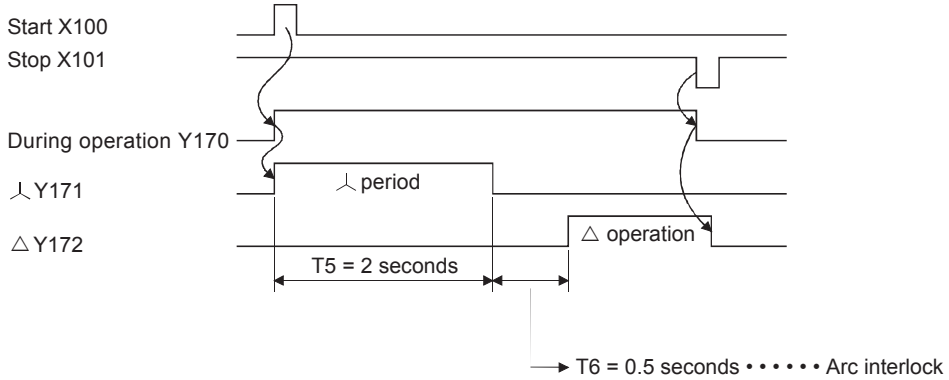
A

Appendix 4.11 Star-delta starting of an electric motor

Turning on the start switch starts the ⅄ operation. After the ⅄ operation time has elapsed, the Δ operation starts through the arc interlock state.

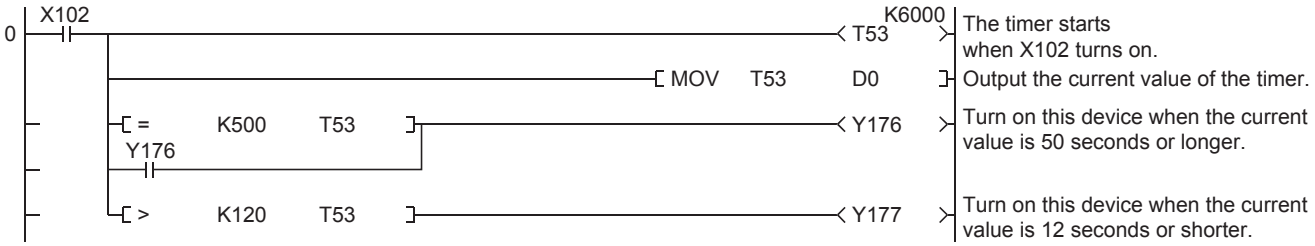
Project name	RA-20
Program name	MAIN

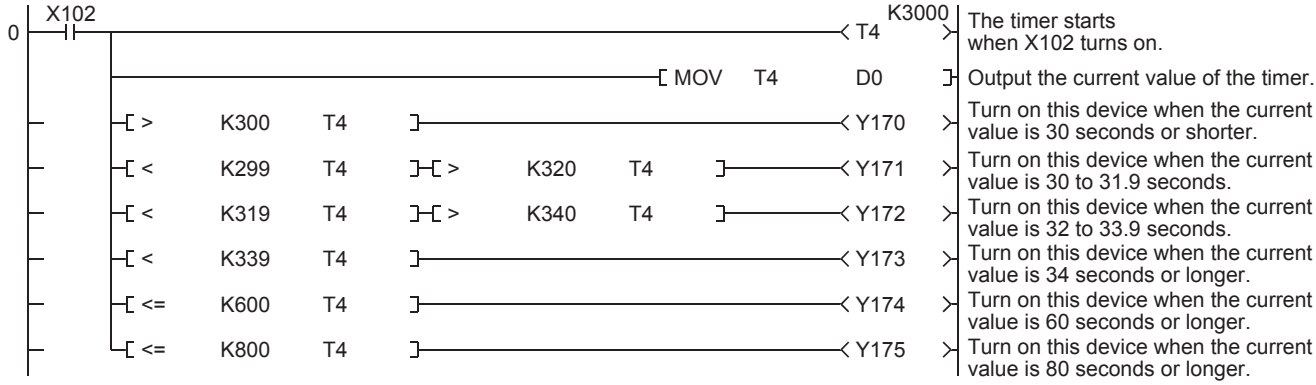




Appendix 4.12 Displaying the elapsed time and outputting before time limit

With the following ladder, the initial indication device displays the elapsed time of a timer and indicates that the elapsed time has reached a set time limit.
This ladder can be applied to counters.





When X102 turns on, the operation starts. When X102 turns off, the operation stops.

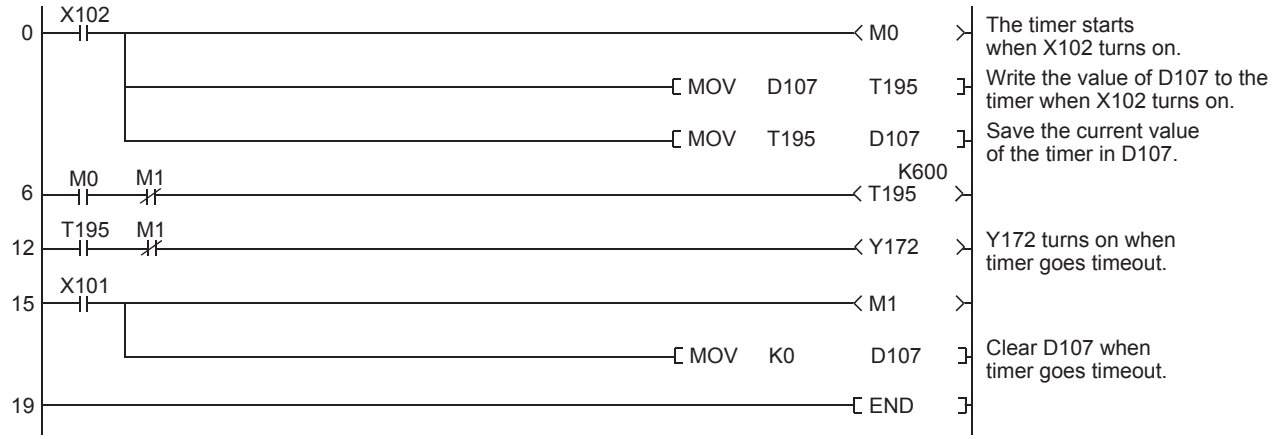
A

Appendix 4.13 Retentive timer

The input X102 repeatedly turns on and off. The ON time of X102 is integrated, and Y172 turns on with the integrated value n.

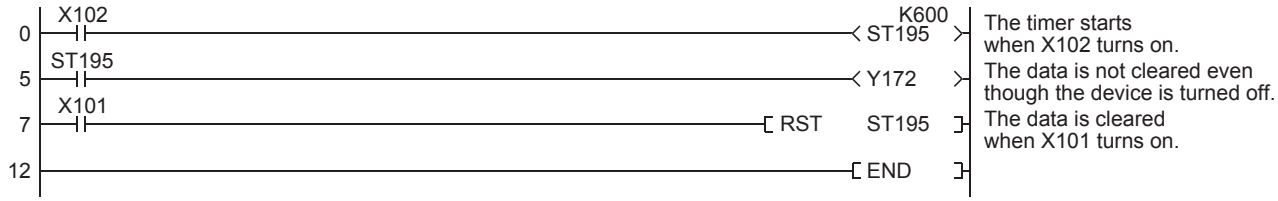
1. In the following ladder, the ON time is integrated without a retentive timer.

Project name	RA-21
Program name	MAIN



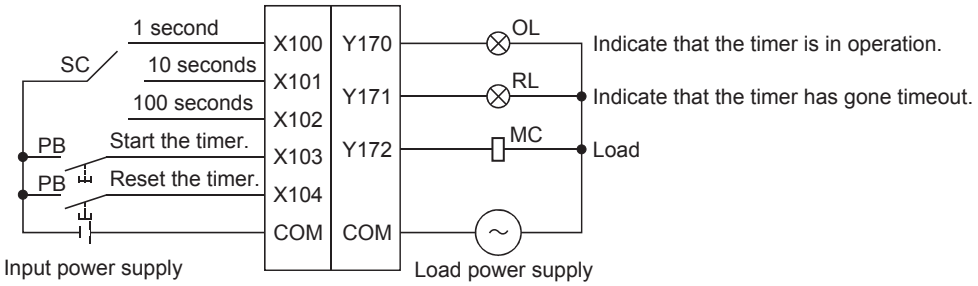
2. In the following ladder, a retentive timer has been assigned in the device setting of the CPU parameter.
Retentive timer (ST): 224 points (ST0 to ST223)

Project name	RA-8
Program name	MAIN

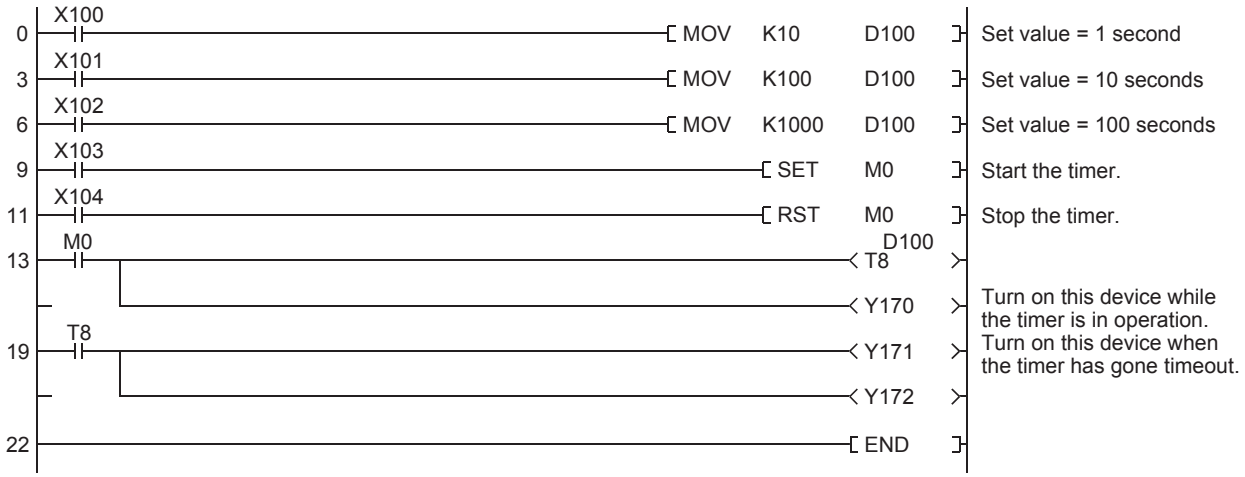


Appendix 4.14 Switching timer setting values with external switches

The three time limits, 1 second, 10 seconds, and 100 seconds, of a timer can be switched with external switches.
To start or reset the timer, use a button on the GOT screen.



Project name	RA-22
Program name	MAIN



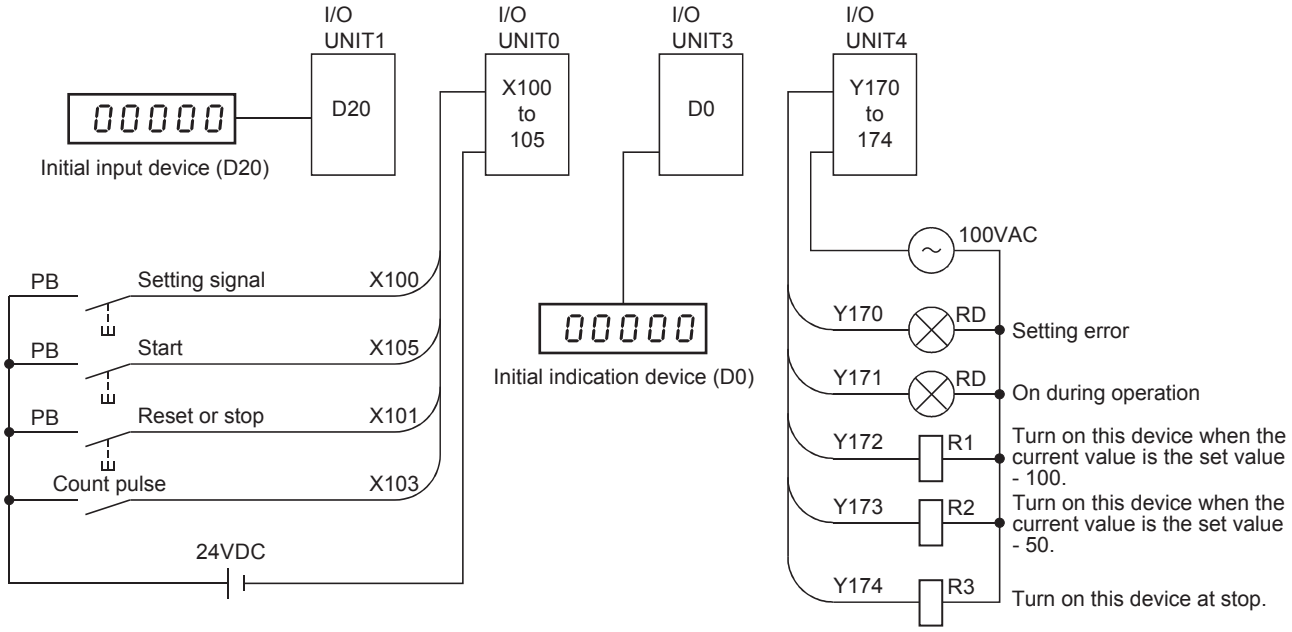
A

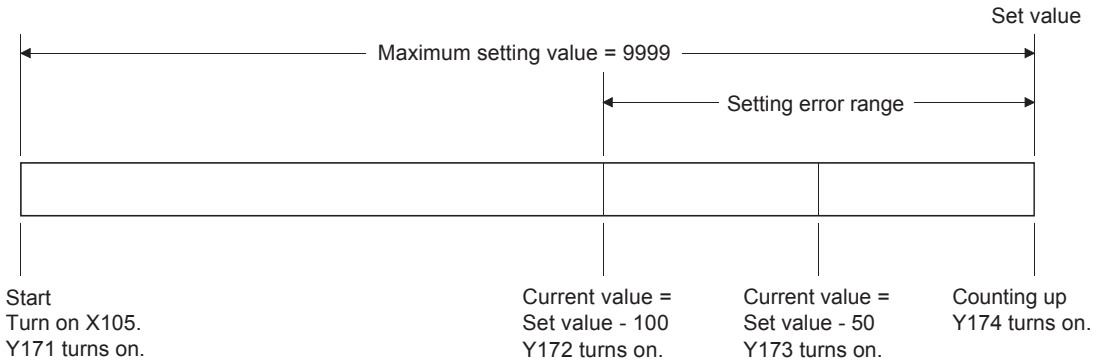
Appendix 4.15 Setting a counter with external switches

Configure the remote setting of a counter with the initial input device of the GOT, and display the current value of a counter in four digits.

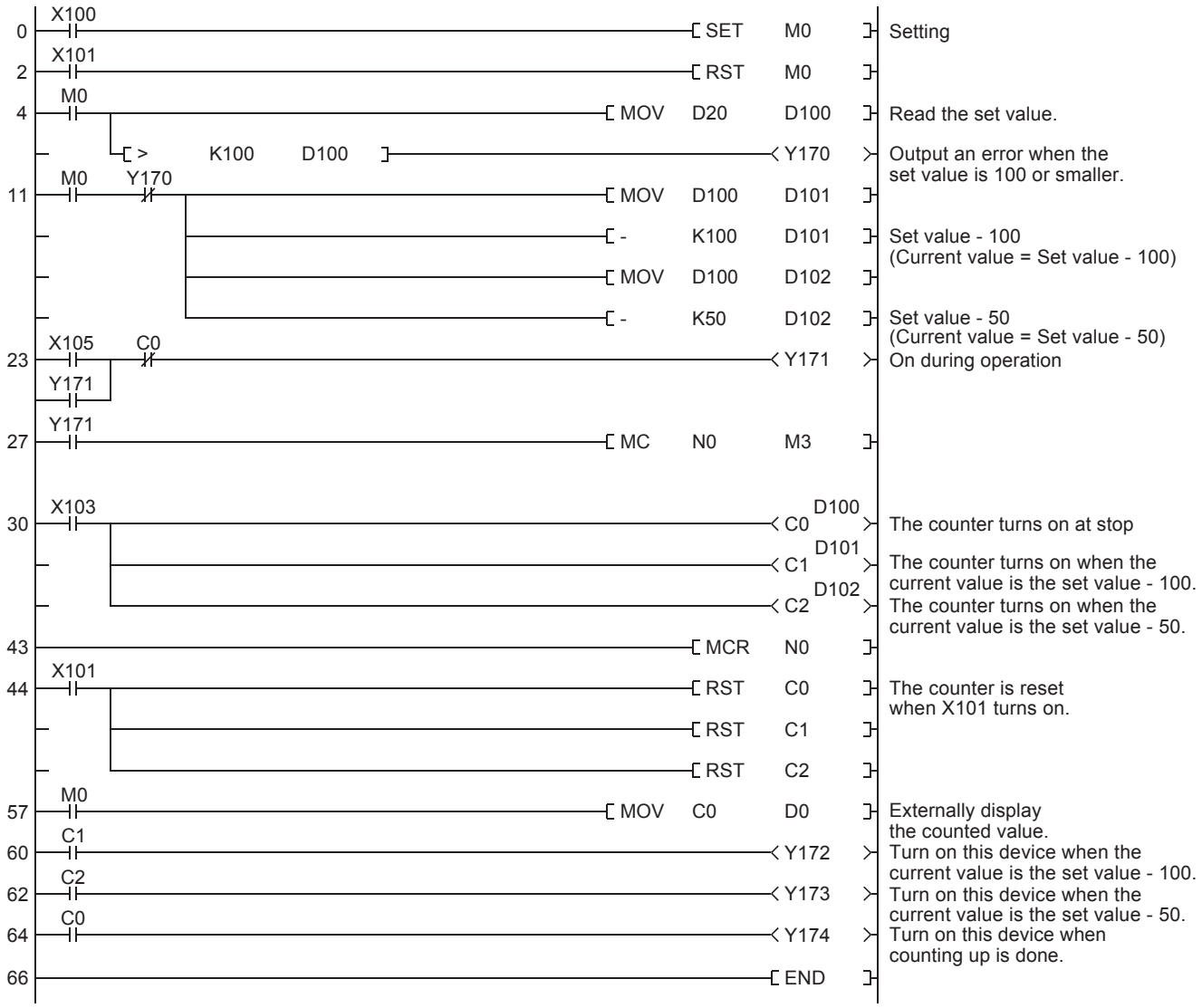
An output turns on when the current value is the set value - 100 or 50, or when counting is up.

When the value set to the counter is smaller than 100, a setting error is displayed.





Project name	RA-4
Program name	MAIN



Precautions

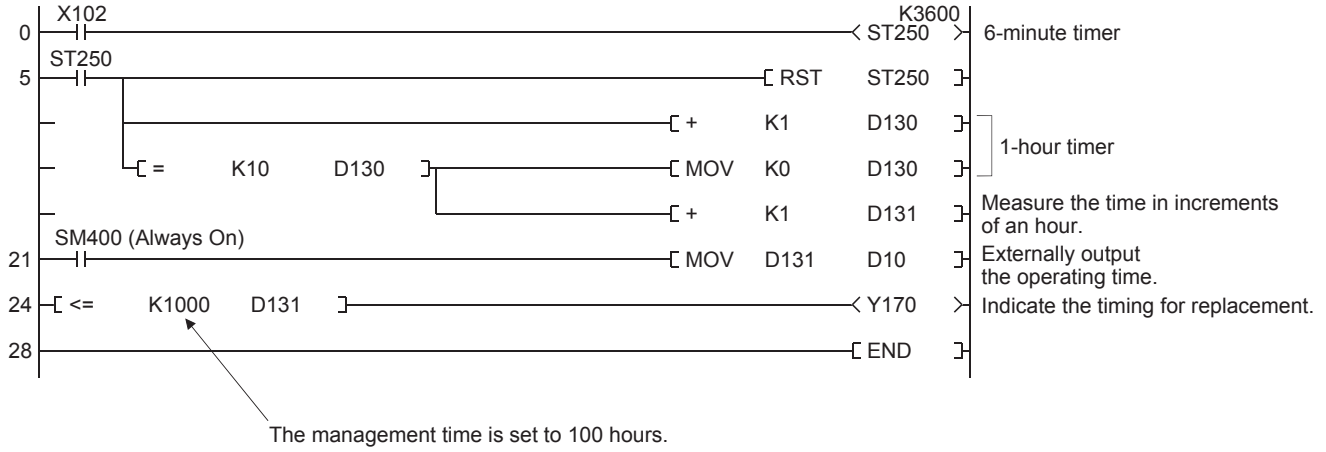
GX Works3 displays the on/off state of a master control on the title tag on the monitor window.

A

Appendix 4.16 Measuring the operating time

Set the operating time of a control target and use this program for maintenance, such as part replacement or lubrication. Take measures to hold data of the timer (ST) and data register (D) at power-off. The following ladder operates as an operating time meter using D131 (in increments of an hour) for externally displaying the time.

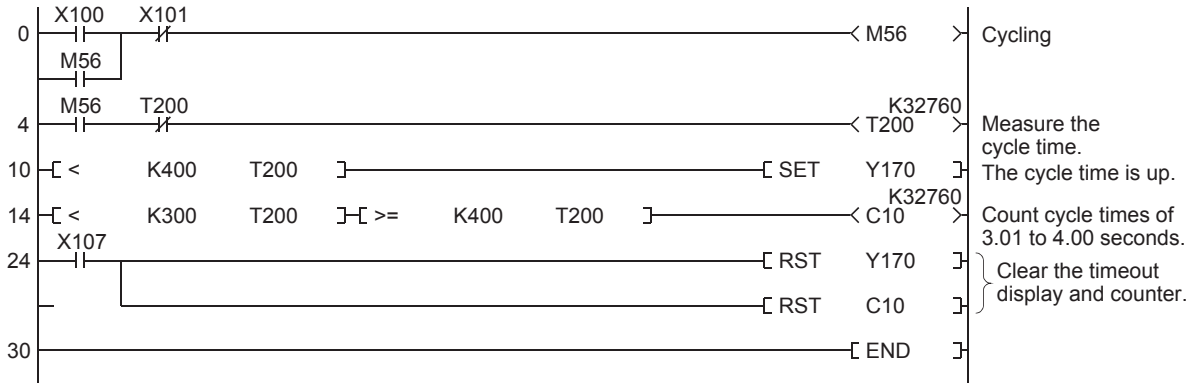
Project name	RA-23
Program name	MAIN



Appendix 4.17 Measuring the cycle time

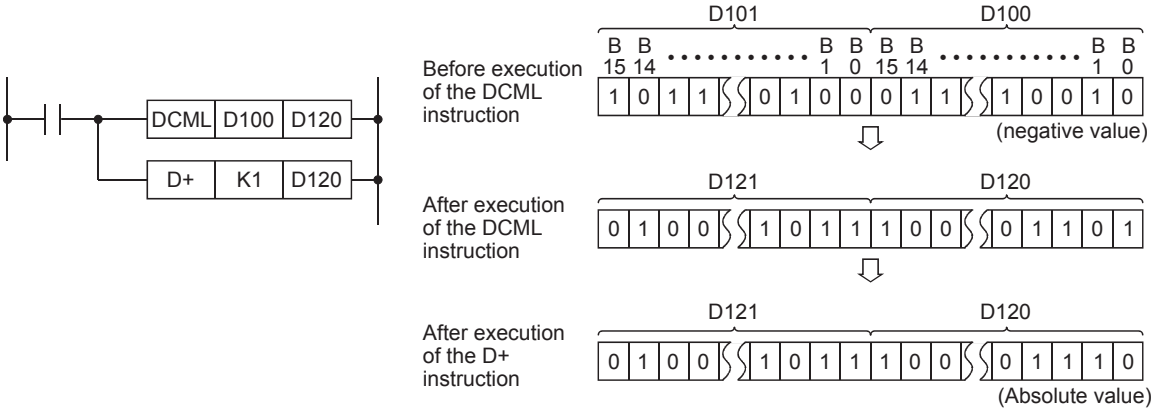
Measure the time from when a control target starts operating until when it completes operating for indicating cycle timeout or managing variation of the time.
With the following ladder, the cycle timeout is indicated, and <, >, and = instructions are used to determine the state of T200. Depending on the determination, the counter operates to measure the deviation of the time.

Project name	RA-24
Program name	MAIN



Appendix 4.18 Application example of (D)CML(P)

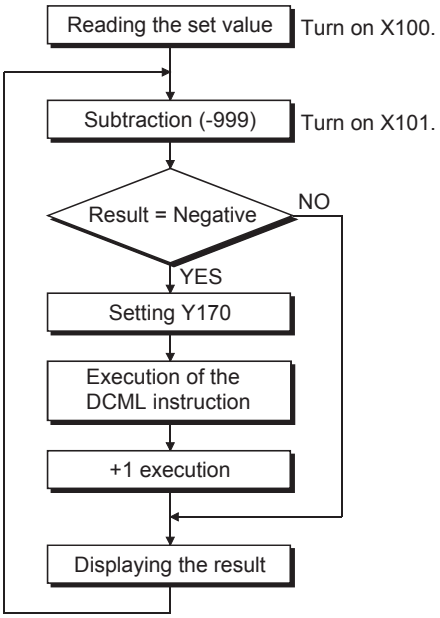
The following ladder is used to obtain the absolute value of a negative value of -32768 or smaller (-2147483648 at a minimum, 32-bit data).

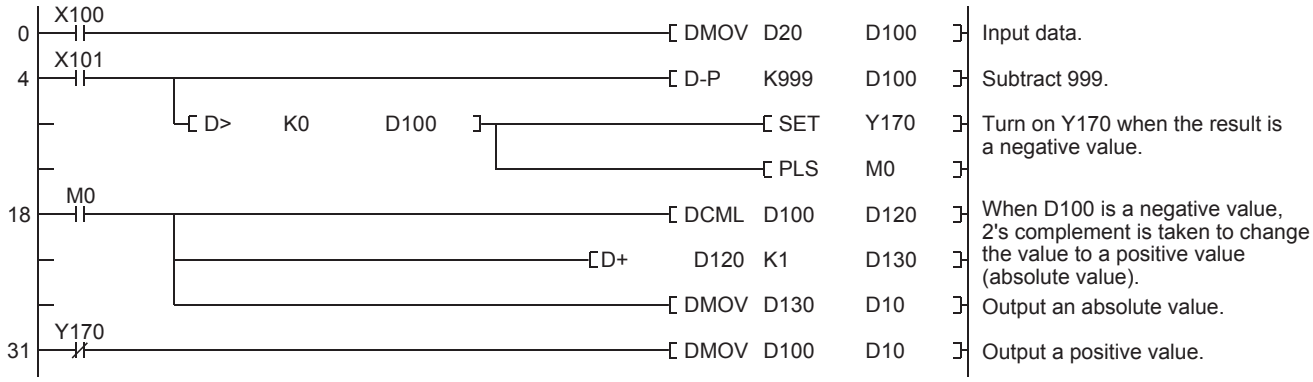


(Example)

Every time when X101 turns on, 999 is subtracted from a set value and the result is displayed.

When the result is a negative value, the output Y170 turns on and its absolute value is displayed.

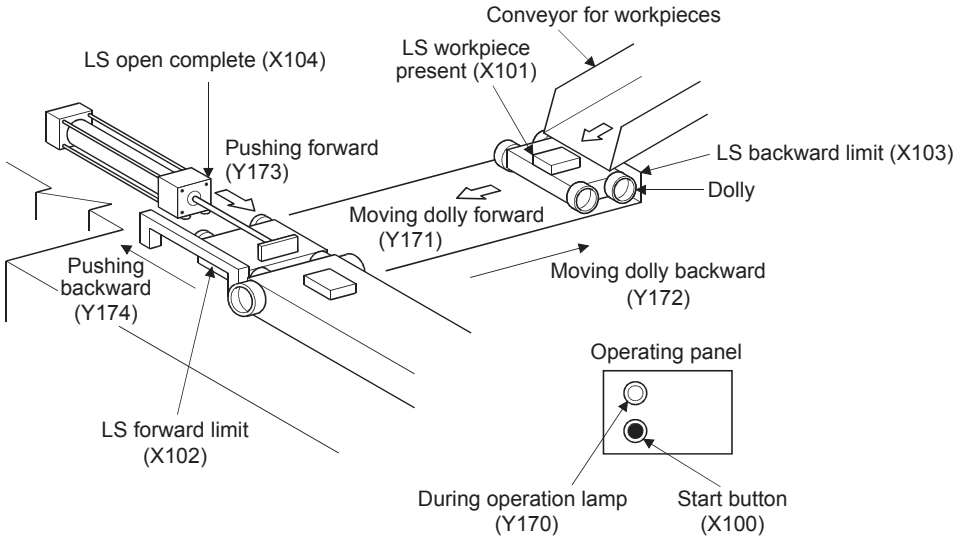


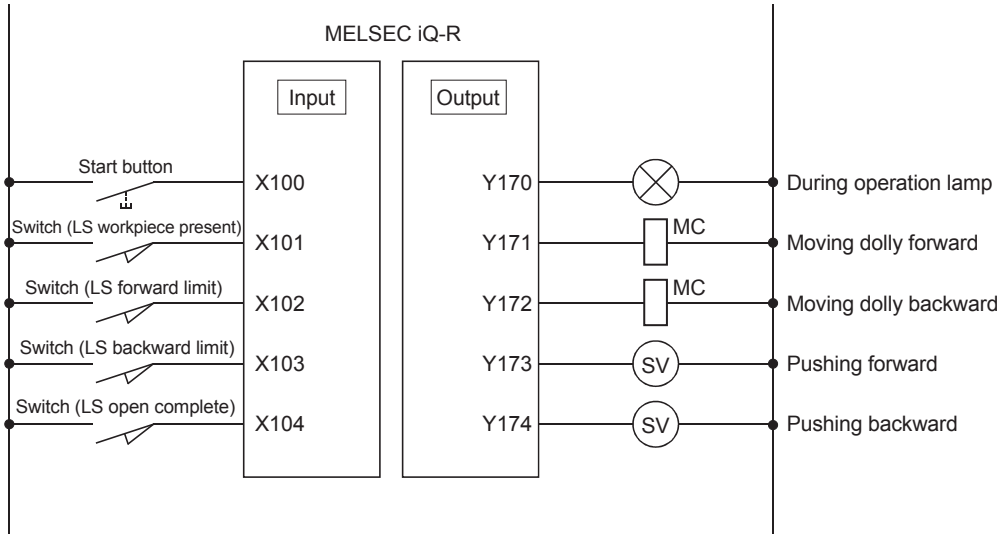


A

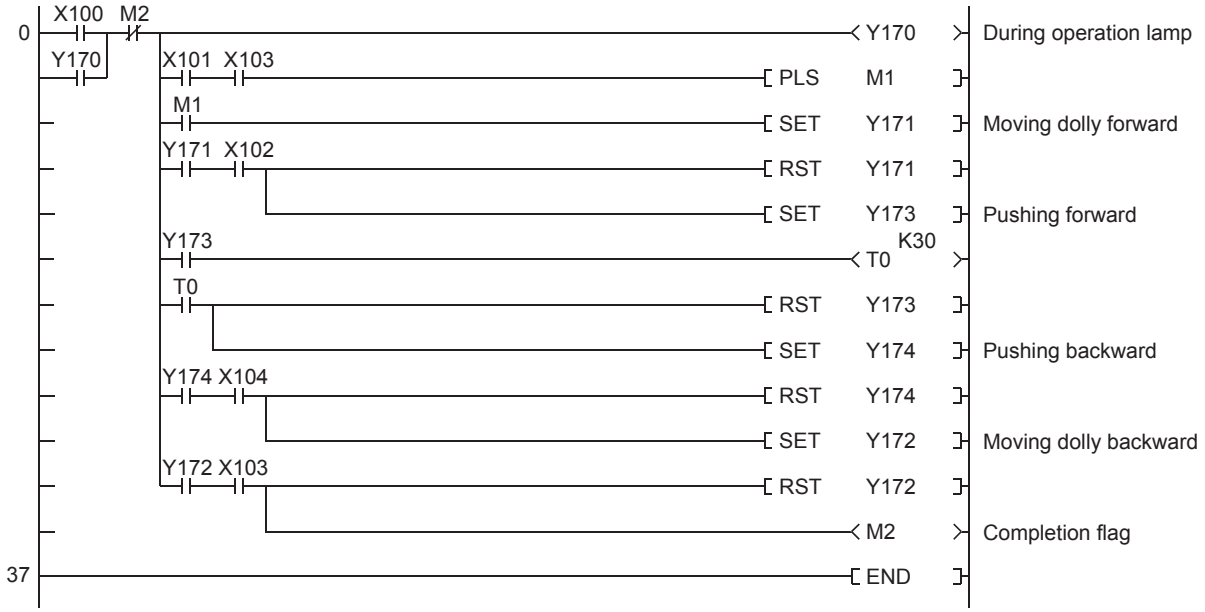
Appendix 4.19 Dolly line control

The following figure shows an example of the sequence control using a dolly to convey a workpiece (material).
Operations in one cycle are as follows; When a workpiece is set on a dolly, the dolly moves forward. When it reaches the forward limit position, the arm pushes the workpiece onto another conveyor. Then, the dolly moves backward and reaches its backward limit position.

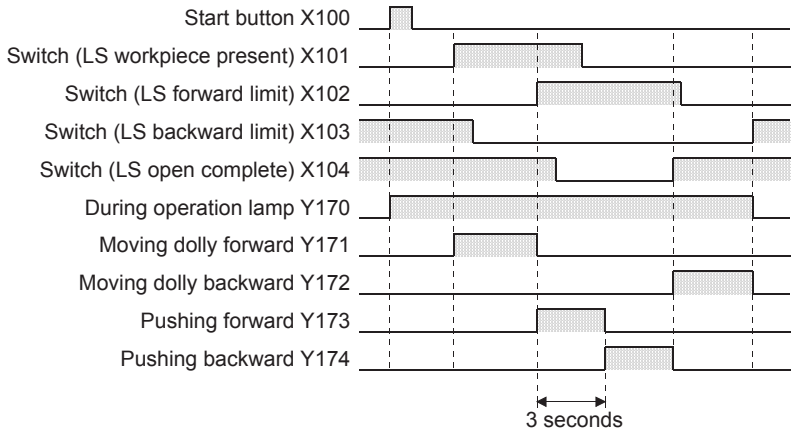




Project name	RA-10
Program name	MAIN



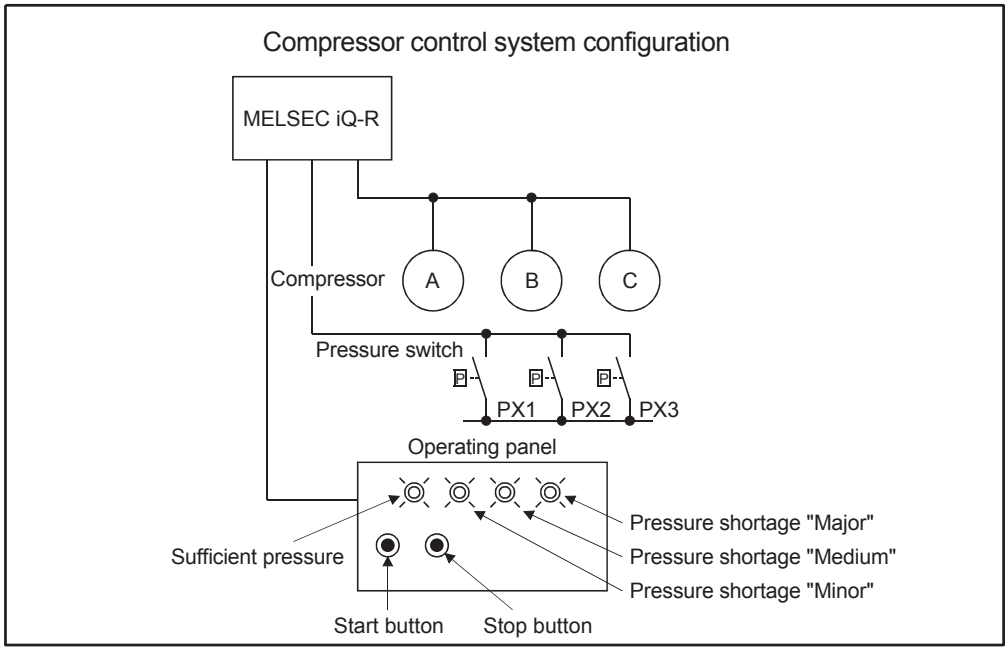
Timing chart

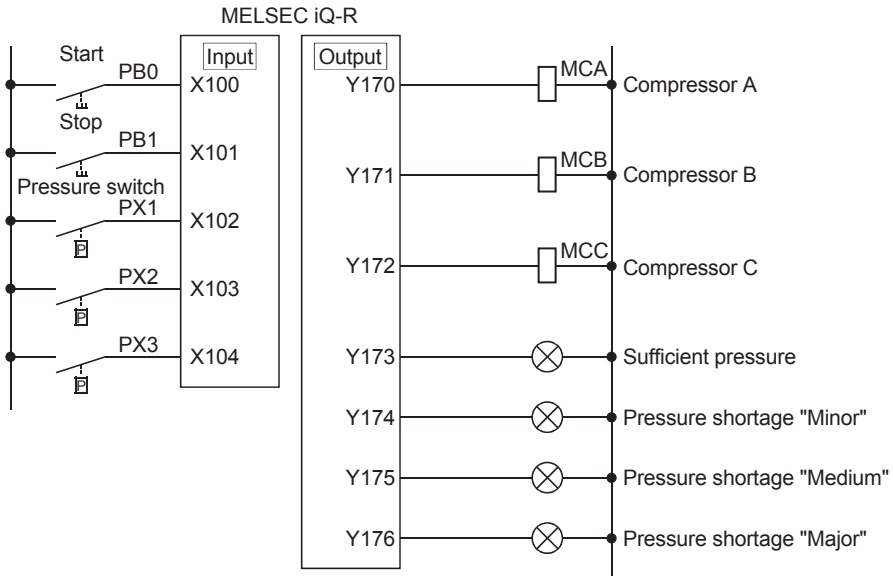


A

Appendix 4.20 Compressor sequential operation with ring counters

The following figure shows a pressure control system with three compressors.
The lack of pressure is detected by three pressure switches. The number of compressors to be operated depends on a pressure shortage level detected. Compressors keep operating until a sufficient pressure is obtained. To equalize the number of operating times of compressors, the sequential control is performed.





■Operation explanation

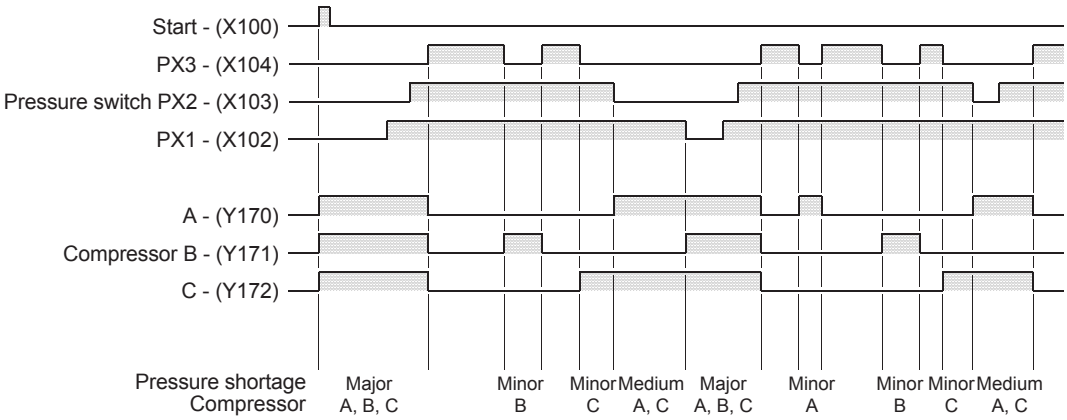
(1) The basic operation of this system is as follows; When the start switch (X100) turns on, three compressors are activated because the pressure switches (X102, X103, and X104) are all off. When a sufficient pressure is obtained (X102, X103, and X104 turn on), all the three compressors will stop.
When the "Minor" pressure shortage occurs (X104 turns off) while all compressors have stopped, one compressor starts and continues operating until a sufficient pressure is obtained.
One compressor to be activated changes in order of A, B, and C every time when a pressure shortage occurs.
To stop the compressor, turn on the stop switch (X101).

(2) If one compressor cannot supply a sufficient pressure and the "Moderate" pressure shortage occurs (X103 turns off), the second compressor is also activated. When the compressor A has been operating, this second compressor will be the compressor C. When the compressor B has been operating, the second compressor will be the compressor A. When the compressor C has operating, and the second compressor will be the compressor B.

(3) If two compressors cannot supply a sufficient pressure and the "Major" pressure shortage occurs (X102 turns off), the third compressor is also activated.
If the "Major" pressure shortage suddenly occurs while one compressor is operating in the basic operation, the other two compressors are simultaneously activated.

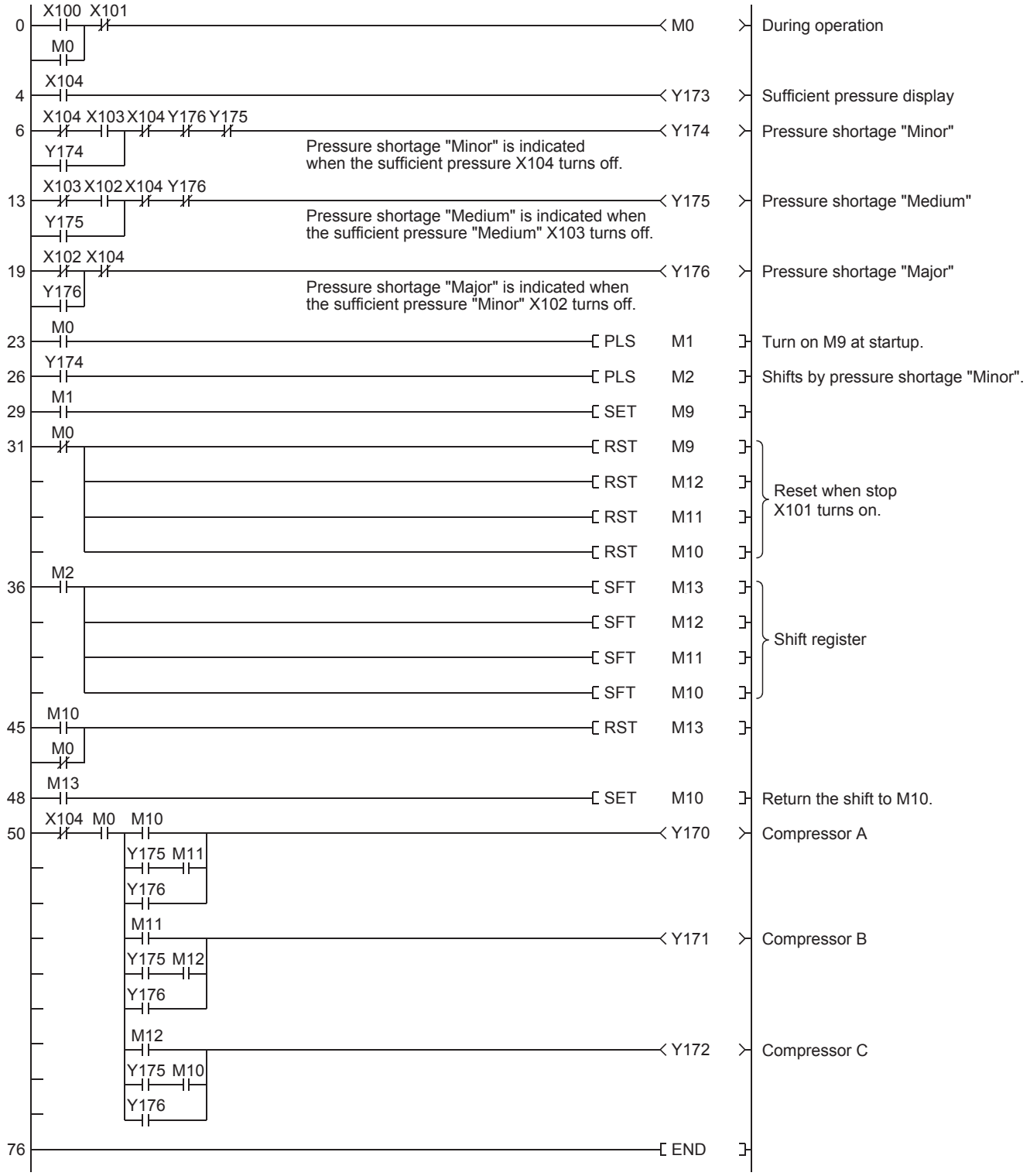
(4) While two or more compressors are operating, they continuously operates until a sufficient pressure is obtained. When a sufficient pressure is obtained (X104 turns on), they will simultaneously stop.

Timing chart

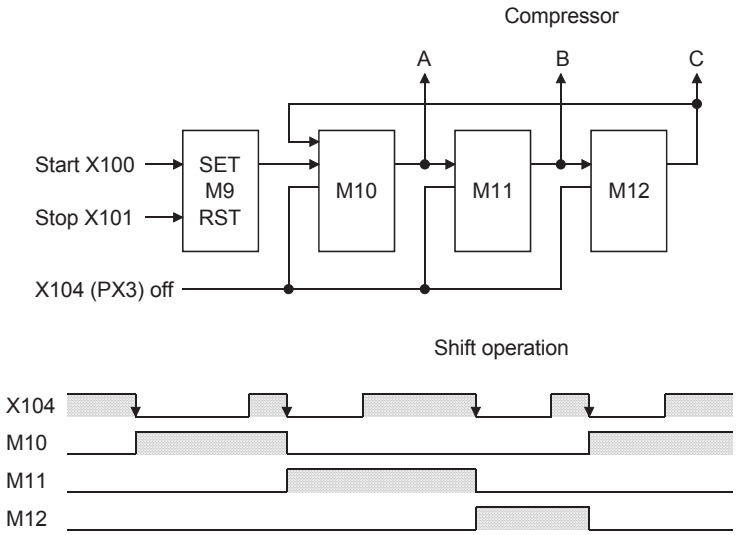


A

Project name	RA-11
Program name	MAIN



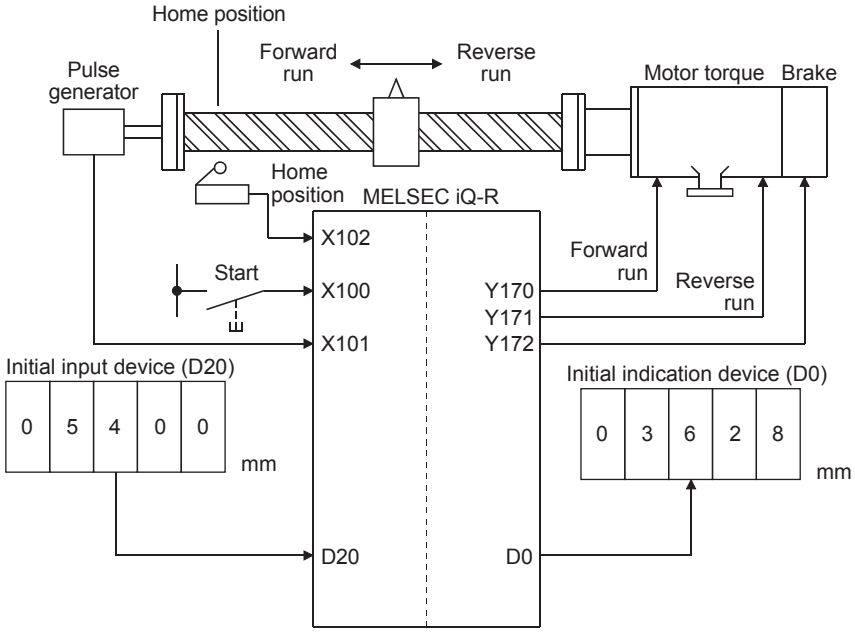
In the basic operation, one compressor is activated when the pressure shortage occurs. The sequential control is performed to equalize the number of operating times of three compressors. This system uses three-step ring counters (ring-type shift registers) of M10 to M12.
When the pressure shortage occurs (X104 turns off), a shift signal is generated.



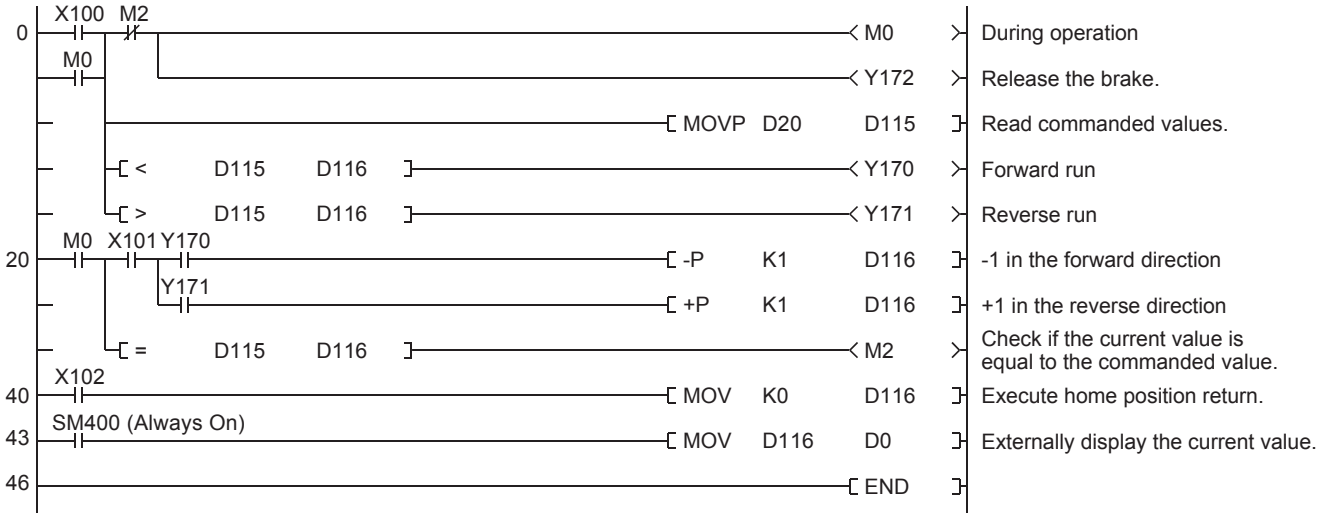
A

Appendix 4.21 Application example to a positioning control

The following figure shows an example of a positioning system with a motor, a brake, and a pulse generator that outputs pulses for each unit.
Set a command value with the initial input device D20. When a positioning operation starts, the commanded value and the current value are compared to determine the run direction (forward or reverse). One is subtracted from the value in the register D116 in the forward direction, and one is added to the value in the register D116 in the reverse direction. When the current value is equal to the commanded value, the positioning operation is completed. The current position is displayed as a numerical value of the initial indication device D0.

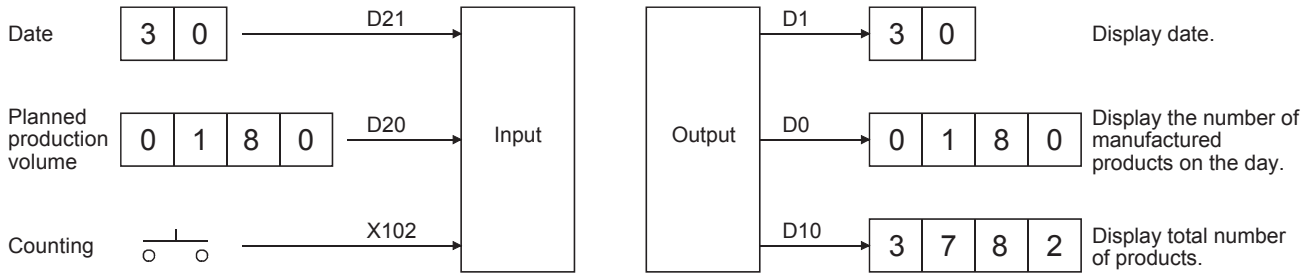


Project name	RA-26
Program name	MAIN



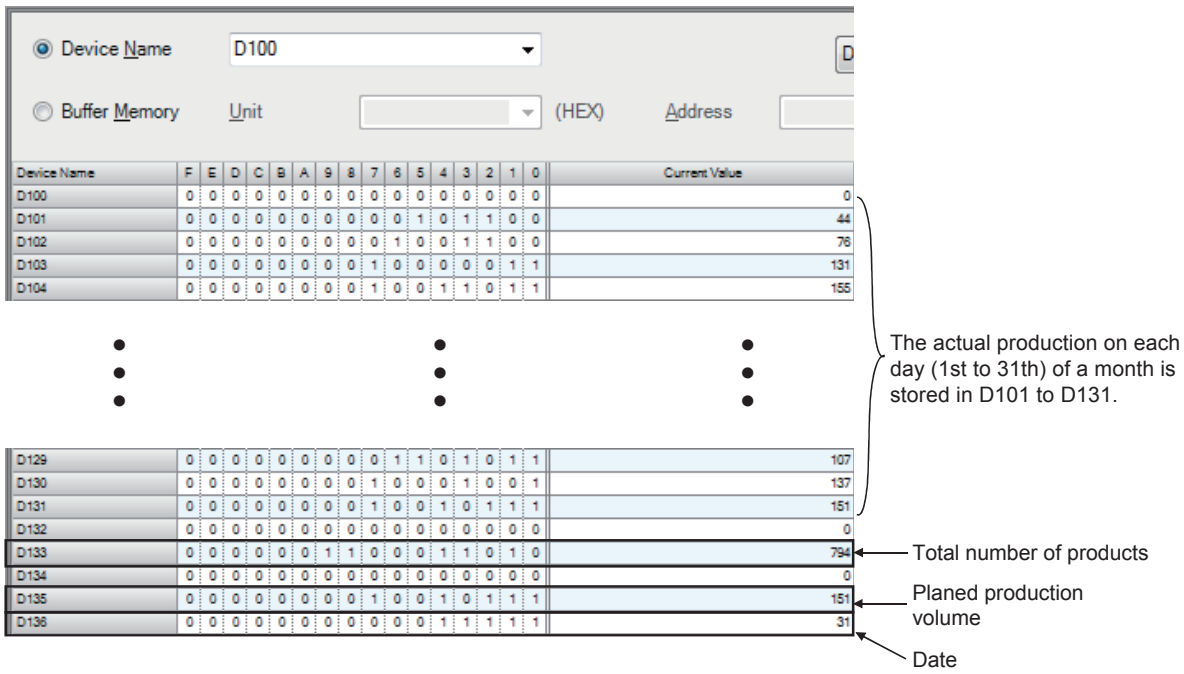
Appendix 4.22 Application example using the index register (Z)

1. The number of products manufactured on a day is counted every day of a month, and the actual production on each day is stored in the data register (D101 to D131) corresponding to the date.
2. Set the planned production volume on a day with the initial input device (D20). When the number of manufactured products reaches this value, the production stops.
3. Set a date with the initial input device (D21).
4. The total number of manufactured products in a month and the number of products manufactured on a day are output to external display devices.



C5 counts the number of products manufactured on the day.
C6 counts the total number of products manufactured in a month.
Enter a date in the index register (Z). The data register corresponding to the date is indirectly specified with D100Z0.
When Z0 is 30, "100 + 30" is stored in D100Z0 and D130 is specified.

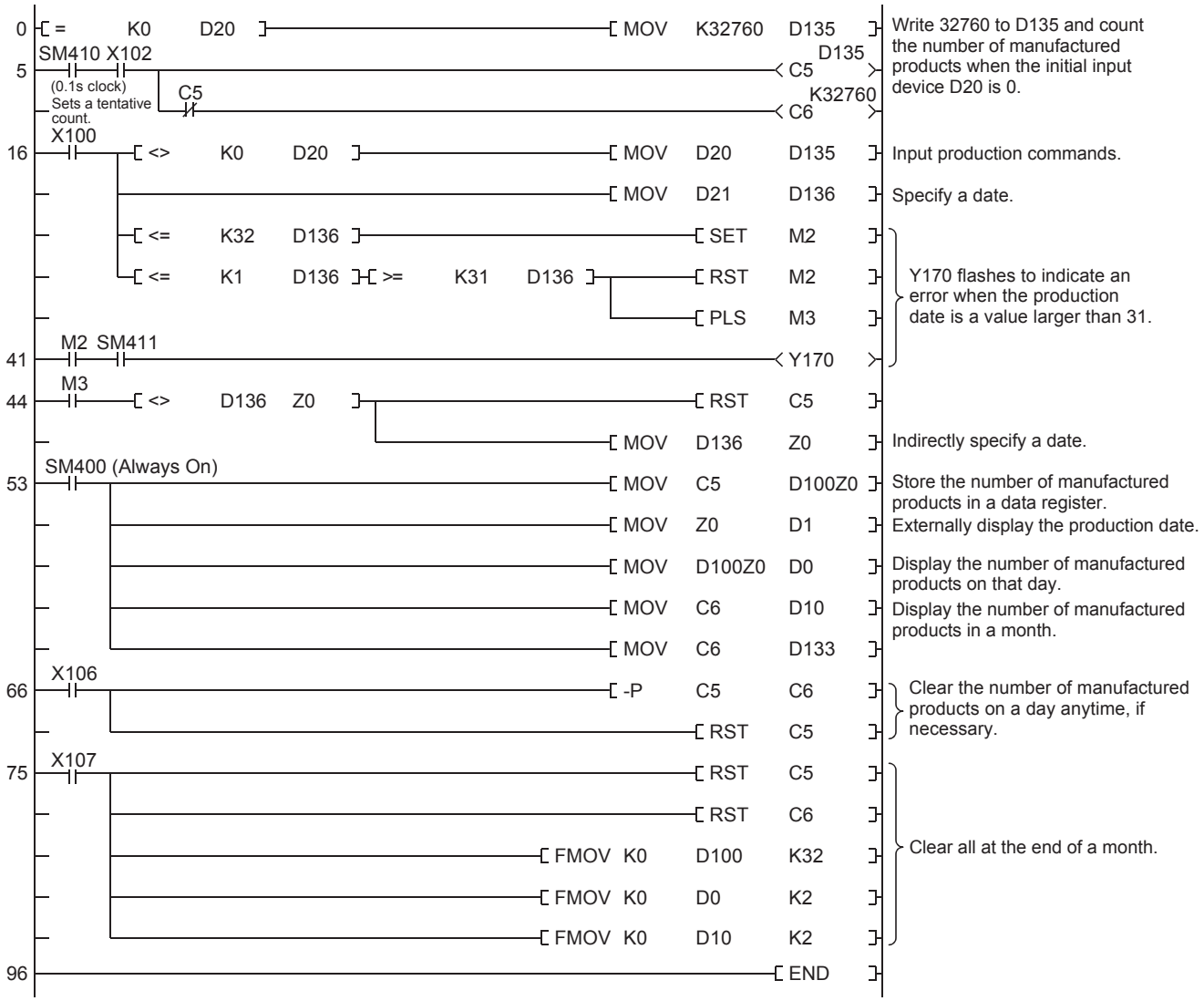
["Device/Buffer Memory Batch Monitor" window]



The number of products manufactured on a day (1st to 31st) is stored in D101 to D131. Read the values and use them as production data as required.

A

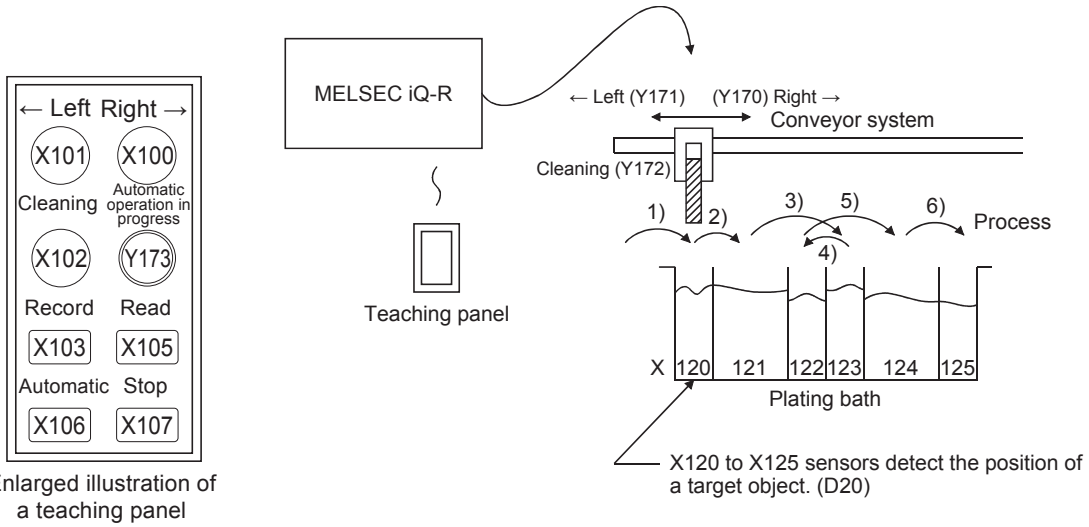
Project name	RA-7
Program name	MAIN

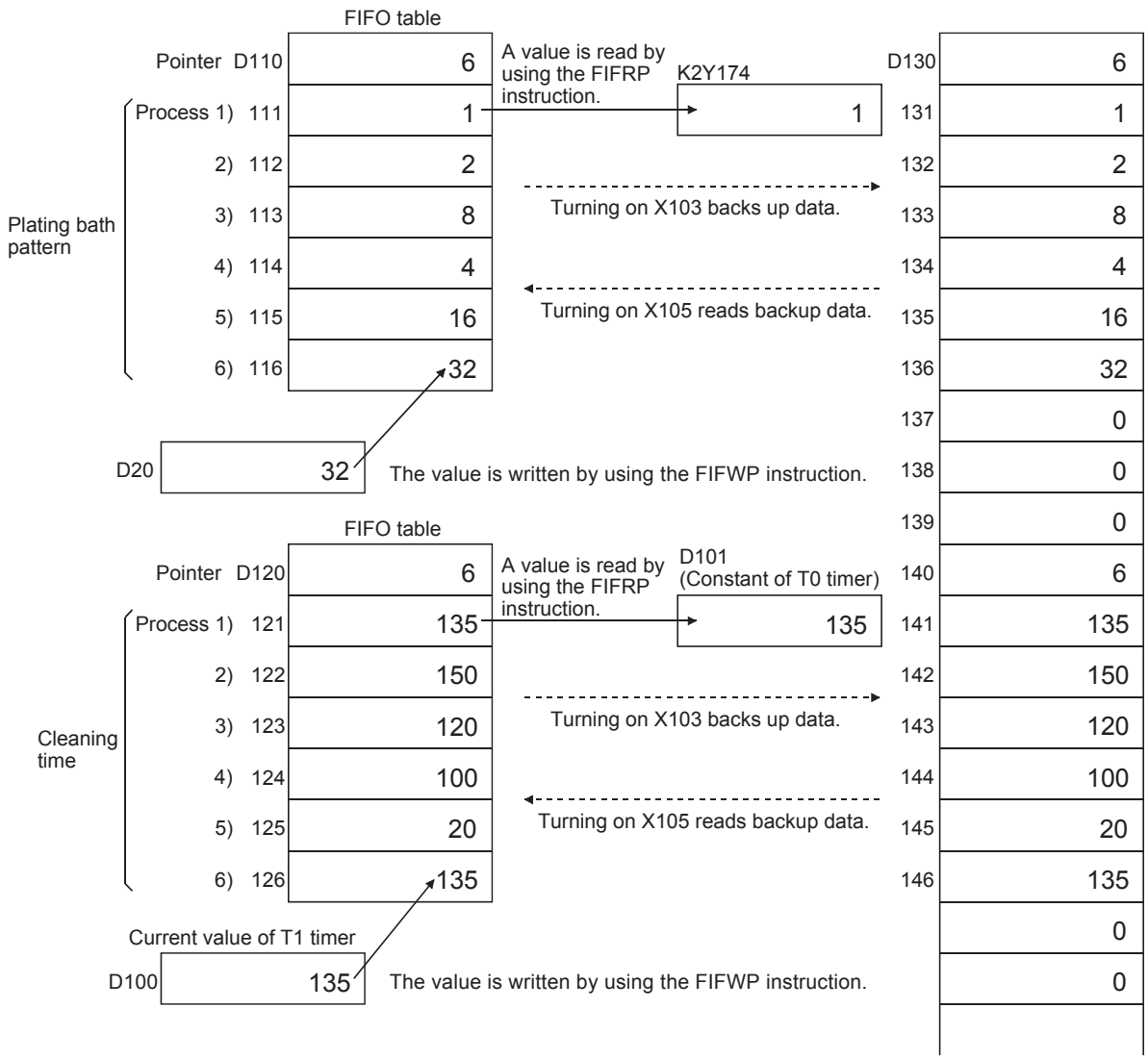


FMOV	K0	D100	K32	Simultaneously transfer data 0 to D100 to D131.
FMOV	K0	D0	K2	Simultaneously transfer data 0 to D0 and D1.
FMOV	K0	D10	K2	Simultaneously transfer data 0 to D10 and D11.

Appendix 4.23 Application example of FIFO instructions

In this system, processes and processing time of manual plating work are recorded to perform the processes with automatic operations.





A

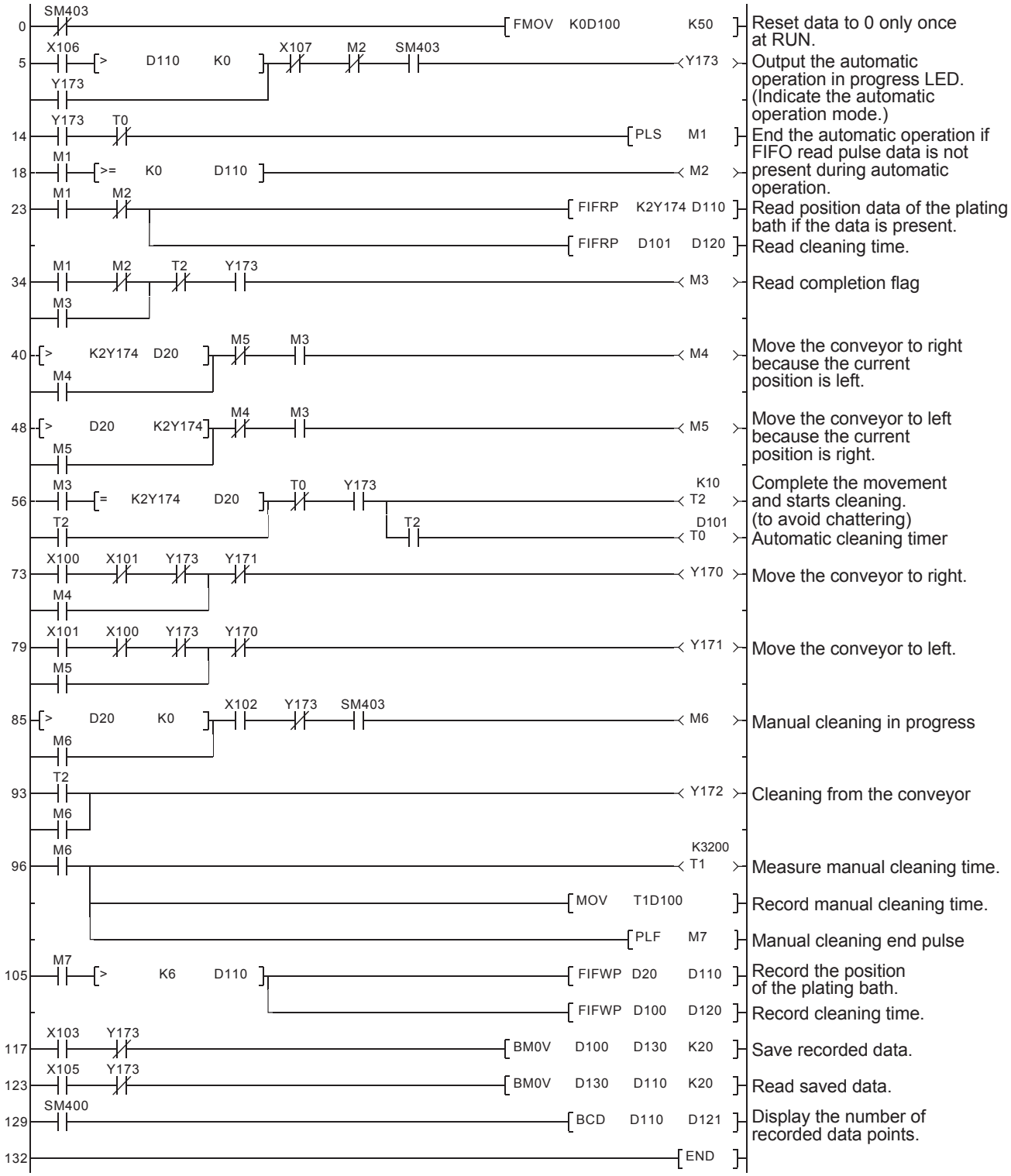
Operation pattern of switching manual work to automatic operation

Teaching panel	Cleaning machine	Plating bath
X100 = Manual right movement button X101 = Manual left movement button X102 = Manual cleaning button X103 = Recording data button X105 = Reading data button X106 = Automatic operation button X107 = Operation stop button Y173 = Automatic operation in progress LED	Y170 = Conveyor, Moving to right Y171 = Conveyor, Moving to left Y172 = Conveyor, Cleaning output	X120 = Plating bath 1 (D20 = K1) X121 = Plating bath 2 (D20 = K2) X122 = Plating bath 3 (D20 = K4) X123 = Plating bath 4 (D20 = K8) X124 = Plating bath 5 (D20 = K16) X125 = Plating bath 6 (D20 = K32)
Start moving to right (X100 = ON)	Moving to right (Y170 = ON) 1)	Standby position (D20 = 0)
Stop moving to right (X100 = OFF)	Stop moving (Y170 = OFF)	Plating bath 1 (D20 = 1)
Start cleaning (X102 = ON)	Cleaning (Y172 = ON)	
Finish cleaning (X102 = OFF)	Stop cleaning (Y172 = OFF) A	
Start moving to right (X100 = ON)	Moving to right (Y170 = ON) 2)	
Stop moving to right (X100 = OFF)	Stop moving (Y170 = OFF)	Plating bath 2 (D20 = 2)
Start cleaning (X102 = ON)	Cleaning (Y172 = ON)	
Finish cleaning (X102 = OFF)	Stop cleaning (Y172 = OFF)	
Start moving to right (X100 = ON)	Moving to right (Y170 = ON) 3)	
Stop moving to right (X100 = OFF)	Stop moving (Y170 = OFF)	Plating bath 4 (D20 = 8)
Start cleaning (X102 = ON)	Cleaning (Y172 = ON)	
Finish cleaning (X102 = OFF)	Stop cleaning (Y172 = OFF)	
Start moving to left (X101 = ON)	Moving to left (Y171 = ON) 4)	
Stop moving to left (X101 = OFF)	Stop moving (Y171 = OFF)	Plating bath 3 (D20 = 4)
Start cleaning (X102 = ON)	Cleaning (Y172 = ON)	
Finish cleaning (X102 = OFF)	Stop cleaning (Y172 = OFF)	
Start moving to right (X100 = ON)	Moving to right (Y170 = ON) 5)	
Stop moving to right (X100 = OFF)	Stop moving (Y170 = OFF)	Plating bath 5 (D20 = 16)
Start cleaning (X102 = ON)	Cleaning (Y172 = ON)	
Finish cleaning (X102 = OFF)	Stop cleaning (Y172 = OFF)	
Start moving to right (X100 = ON)	Moving to right (Y170 = ON) 6)	
Stop moving to right (X100 = OFF)	Stop moving (Y170 = OFF)	Plating bath 6 (D20 = 32)
Start cleaning (X102 = ON)	Cleaning (Y172 = ON)	
Finish cleaning (X102 = OFF)	Stop cleaning (Y172 = OFF)	
Start automatic operation (X106 = ON → OFF)	Moving to left (Y171 = ON)	Plating bath 1 (D20 = 1)
Automatic operation in progress LED (Y173 = ON)	(The automatic operation starts.)	

A

The same operations are automatically executed from (A).

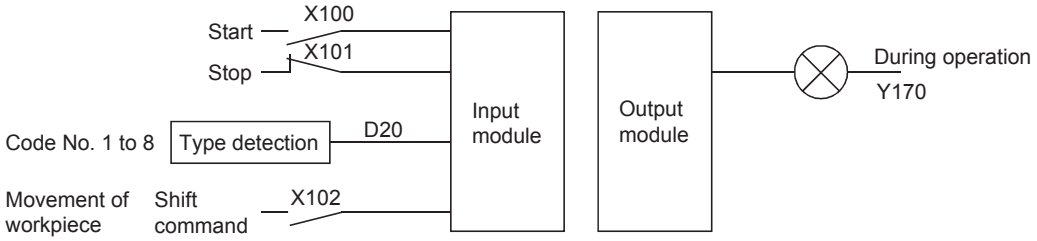
Project name	RA-9
Program name	MAIN



A

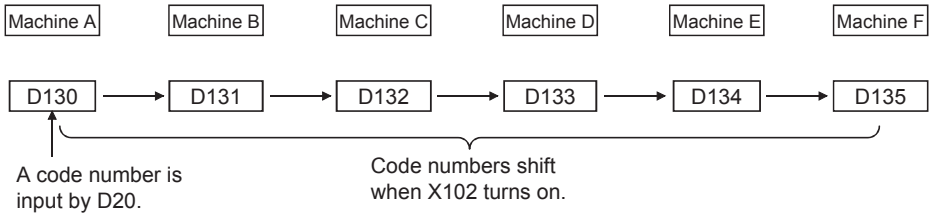
Appendix 4.24 Application example of data shifting

When a workpiece is conveyed, its code number is also shifted. Data is read from the data register for processing machines, and the processing corresponding to the code number is performed.

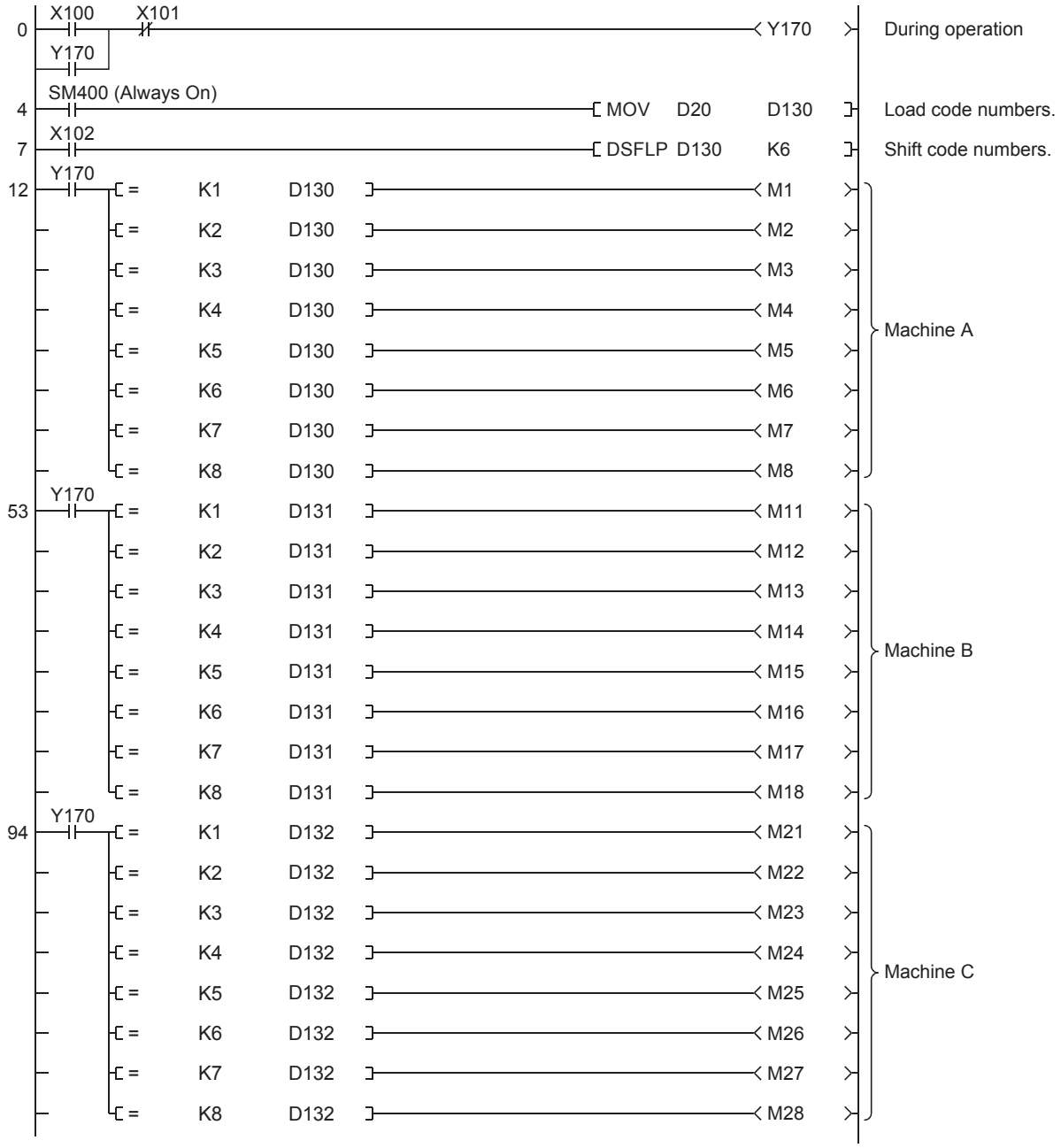


Machine	Data register	Code 1	Code 2	Code 3	Code 4	Code 5	Code 6	Code 7	Code 8
A	D130	M1	M2	M3	M4	M5	M6	M7	M8
B	D131	M11	M12	M13	M14	M15	M16	M17	M18
C	D132	M21	M22	M23	M24	M25	M26	M27	M28
D	D133	M31	M32	M33	M34	M35	M36	M37	M38
E	D134	M41	M42	M43	M44	M45	M46	M47	M48
F	D135	M51	M52	M53	M54	M55	M56	M57	M58

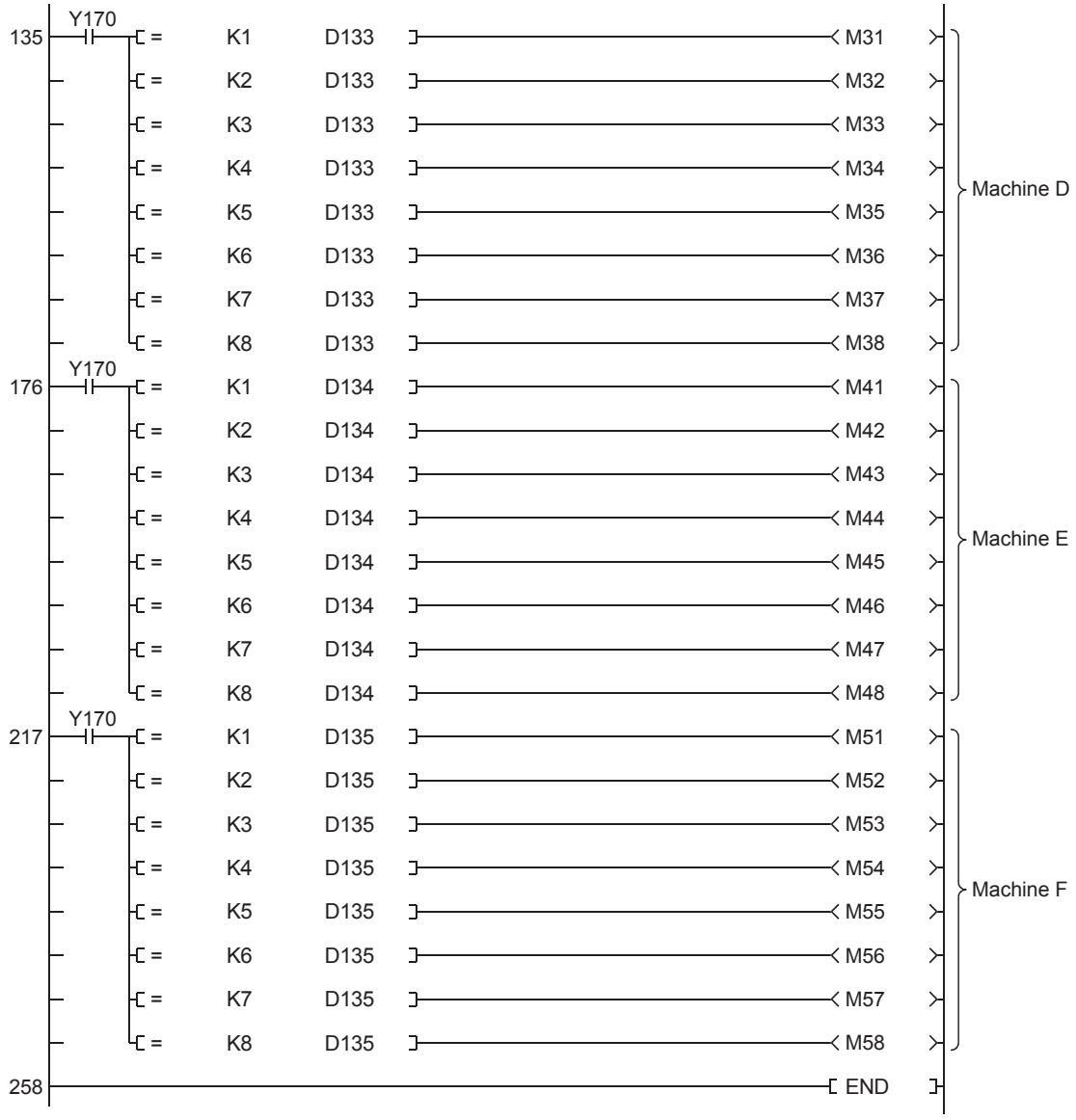
A code number is stored in the data register, and the internal relay (M) corresponding to the code number turns on and processing is executed.



Project name	RA-12
Program name	MAIN



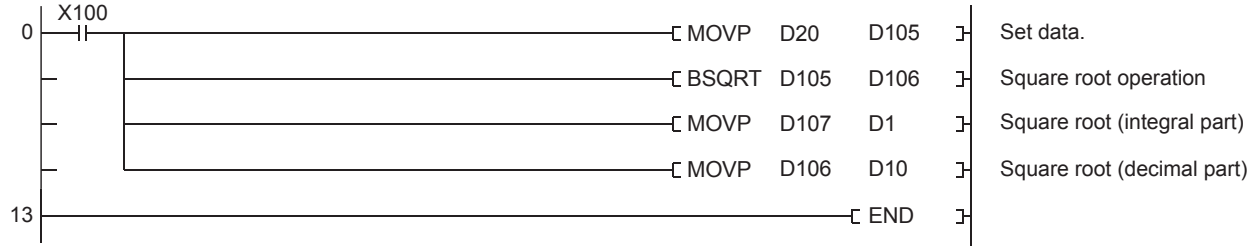
A


135
Y170
= K1 D133 M31
= K2 D133 M32
= K3 D133 M33
= K4 D133 M34
= K5 D133 M35
= K6 D133 M36
= K7 D133 M37
= K8 D133 M38
Machine D
176
Y170
= K1 D134 M41
= K2 D134 M42
= K3 D134 M43
= K4 D134 M44
= K5 D134 M45
= K6 D134 M46
= K7 D134 M47
= K8 D134 M48
Machine E
217
Y170
= K1 D135 M51
= K2 D135 M52
= K3 D135 M53
= K4 D135 M54
= K5 D135 M55
= K6 D135 M56
= K7 D135 M57
= K8 D135 M58
Machine F
258
END

Appendix 4.25 Program example: Square root operations

Project name	RA-14
Program name	MAIN

The square root operation of the value stored in D105 is determined, and a result are stored in D106 and D107.



An operation result is stored as follows.

$$\frac{\sqrt{D105}}{\substack{0\text{ to }9999\\(\text{BCD value})}} = \frac{\overset{\boxed{\text{Integral part}}}{D106}}{\substack{0\text{ to }9999\\(\text{BCD value})}} \cdot \frac{\overset{\boxed{\text{Decimal part}}}{D107}}{\substack{0\text{ to }9999\\(\text{BCD value})}}$$

..... A value whose 5th decimal place is rounded off.
Thus, the value has an error of ±1 at the 4th decimal place.

Point

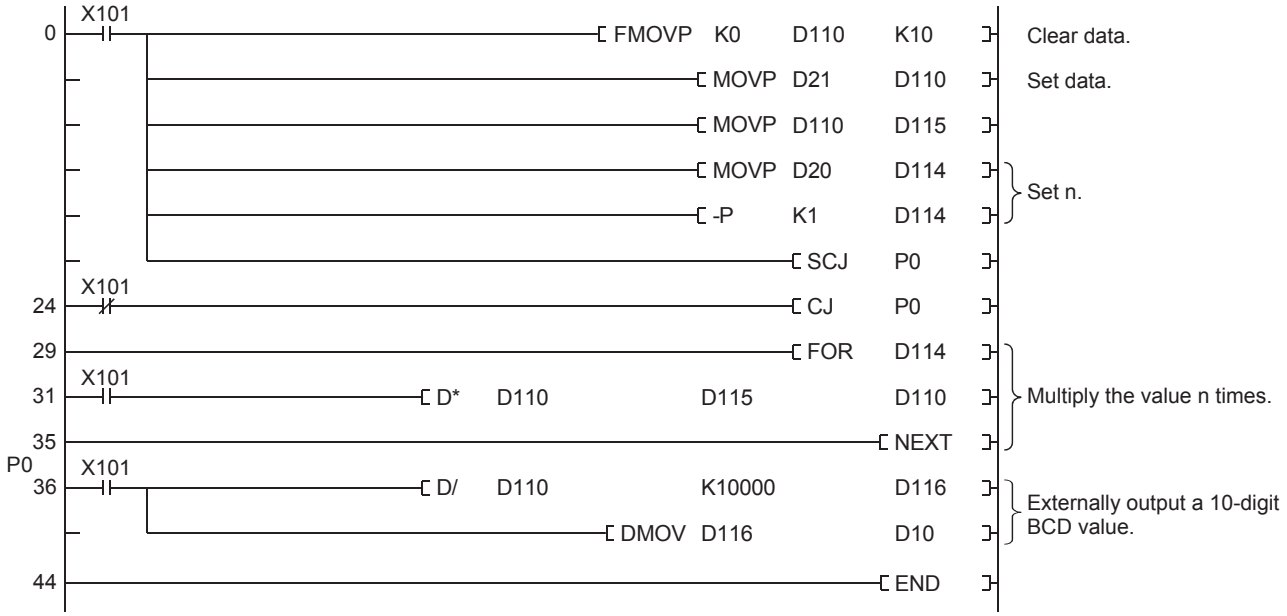
The RCPU provides square root operation instructions for values in a real number (floating point) data format.

A

Appendix 4.26 Program example: Multiplication with the nth power

Project name	RA-15
Program name	MAIN

A value stored in D110 is multiplied n times (n: value stored in D114), and a result is stored in D110.



Precautions

If a value stored in D110 is larger than 2147483647, an error (OPERATION ERROR) occurs.

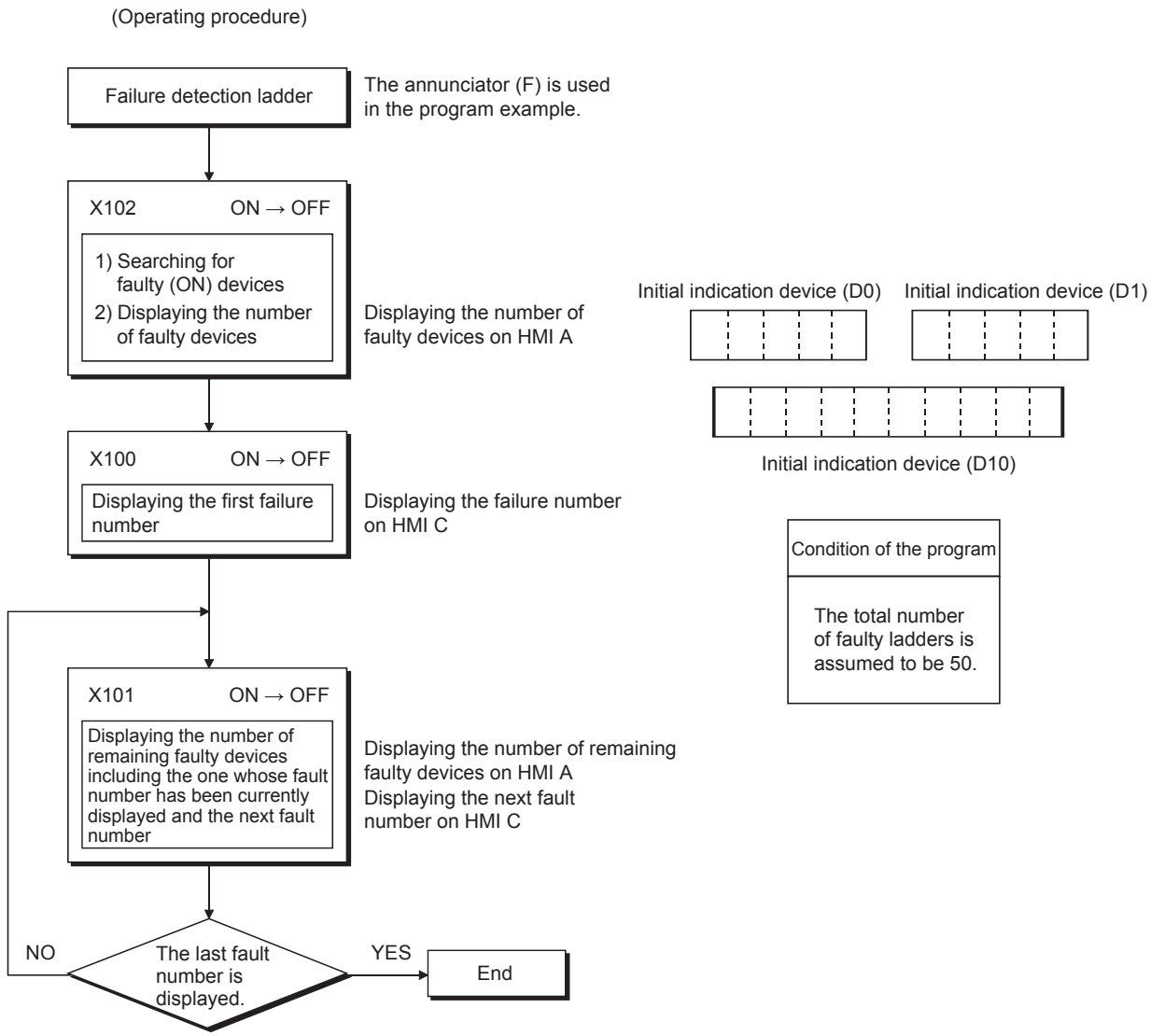
Appendix 4.27 Displaying the number of failures and failure number in a failure detection program

The failure detection program displays the number of devices that is on among the bit devices (such as X, M, and F) used consecutively and their device numbers one by one.

[Application example]

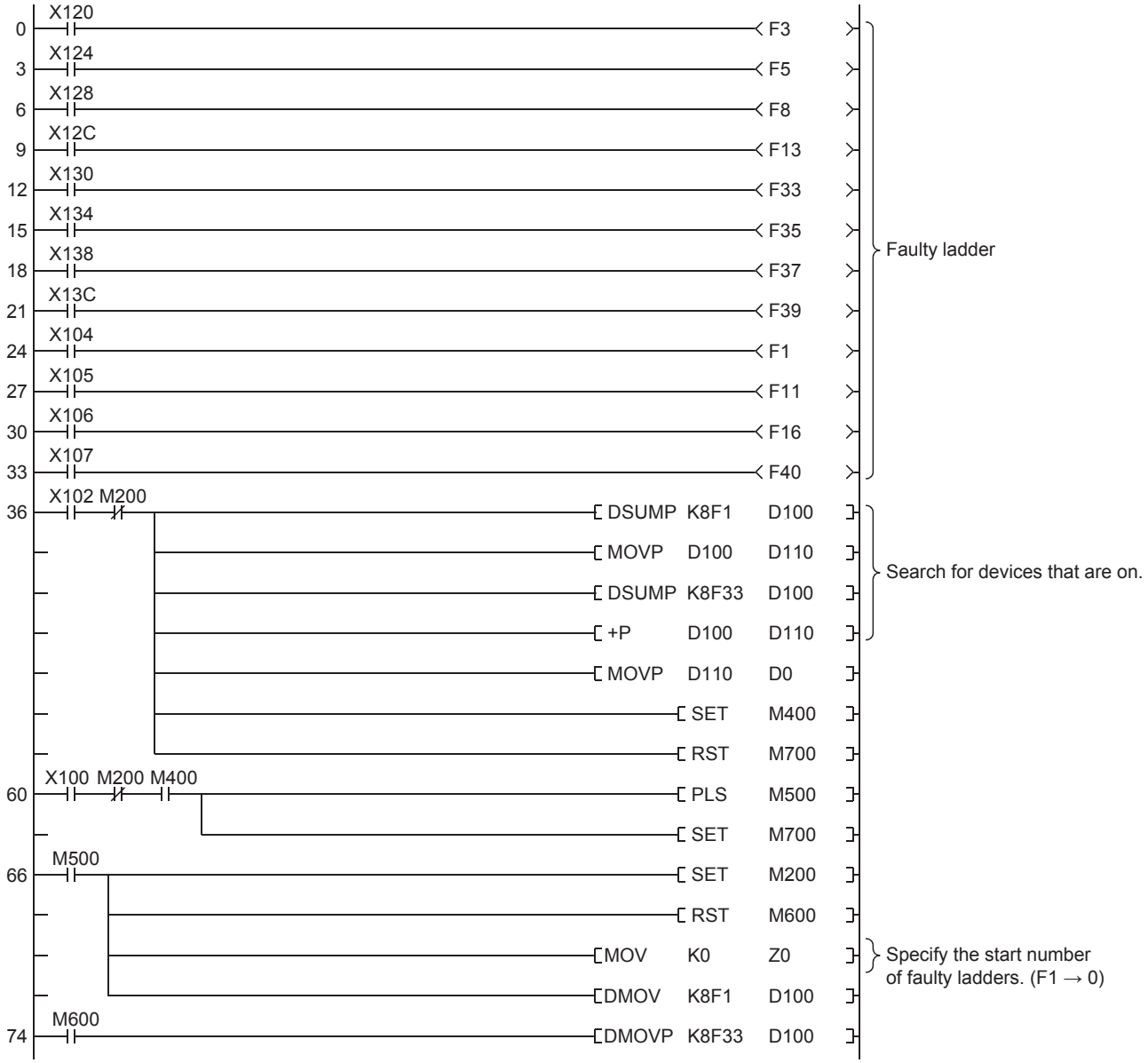
When the internal relay (M) or the annunciator (F) is used as an output device of a failure detection program, use the following program to obtain failure numbers of multiple failures.

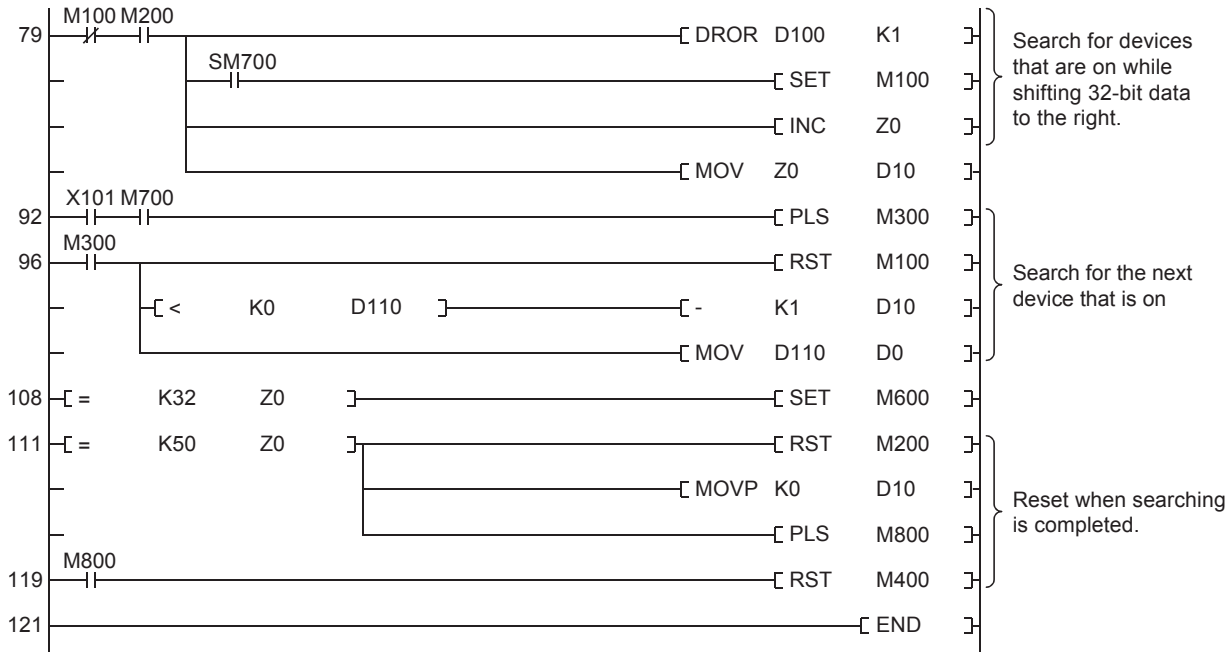
[Sequence program flow]



A

Project name	RA-31
Program name	MAIN

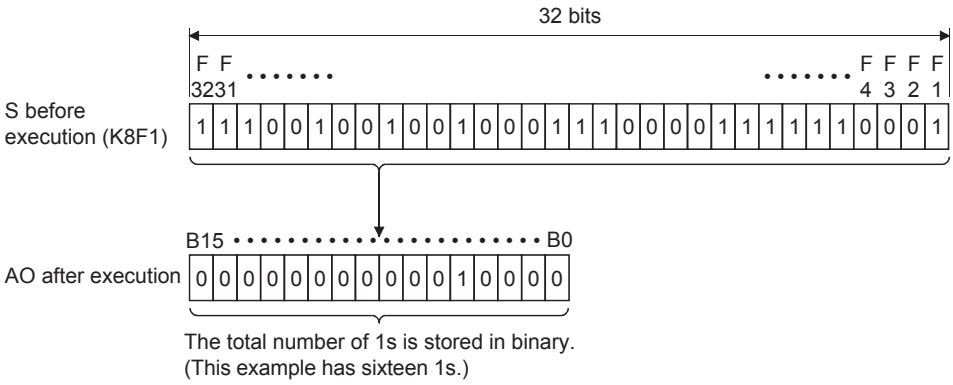




1. Search for devices that are on.

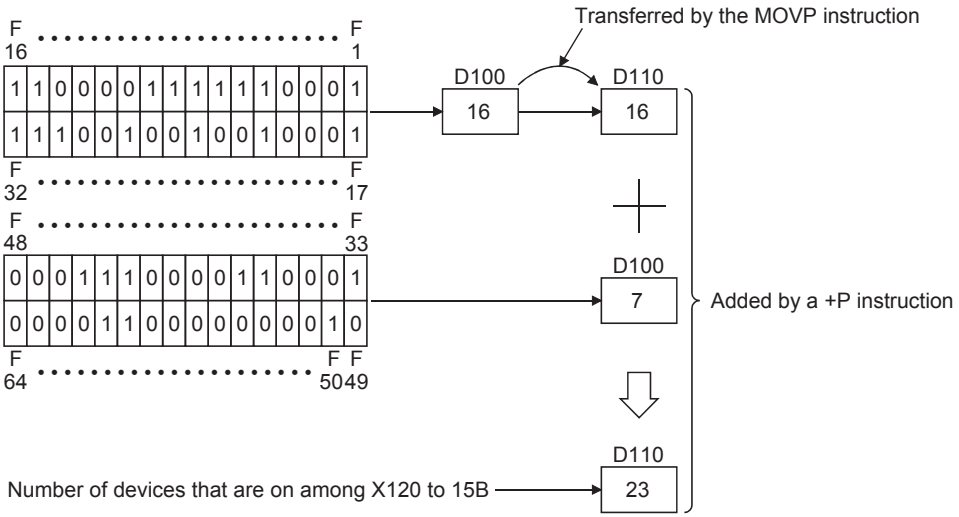
DSUMP	K8F1	D100

DSUMP	K8F33	D100



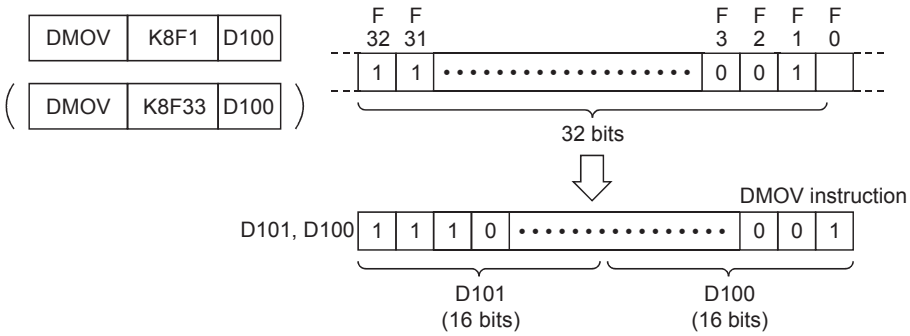
When X102 turns on, the number of bits that are on among F1 to F64 is stored in D110 and displayed.

A



2. Search for devices that are on while shifting 32-bit data to the right.

DROR	D100	K1



DROR	D100	K1

(1) When X100 turns on, the above shift data (D100 and D101) is set and data is shifted to the right by one bit at every scan until the first on bit is detected.
Shifting of data stops in the scan in which the first on bit has been detected (SM700 turns on), and the total number of shifts (corresponding to device numbers) is displayed.

(2) Every time when X101 turns on, the next on bit is detected, and its device number is displayed. At the same time, one is subtracted from the number of bits that are on, and the remaining number of on bits is displayed.

Appendix 5 Memory and Files to be Handled by the CPU Module

File types and storage memory

The following table lists file types and storage destination memory types.

◎: Can be stored (required for operation), ○: Can be stored, ×: Cannot be stored

File type		CPU built-in memory			SD memory card	File name and extension
		Program memory	Device/label memory	Data memory		
		Drive 0	Drive 3	Drive 4	Drive 2	
Program		◎*4	×	◎*4	○	ANY_STRING.PRG
FB program		○*4	×	○*4	○	ANY_STRING.PFB
CPU parameter		×	×	◎	○	CPU.PRM
System parameter		×	×	◎	○	SYSTEM.PRM
Module parameter		×	×	○	○	UNIT.PRM
Module extension parameter		×	×	○	○	• UEXmmmnn.PRM*1 • UEXmmm00.PPR*5
Memory card parameter		×	×	×	○	MEMCARD.PRM
Device comment		×	×	○	○	ANY_STRING.DCM
Initial device value		×	×	○	○	ANY_STRING.DID
Global label setting file		×	×	○	○	GLBLINF.IFG
Initial label value file	Initial global label value file	×	×	○	○	GLBLINF.LID
	Initial local label value file	×	×	○	○	PROGRAM_NAME.LID
File register		×	○	×	○*3	ANY_STRING.QDR
Event history		×	×	○	○	EVENT.LOG
Device data storage file		×	×	○	○*3	DEVSTORE.QST
General-purpose data		×	×	○	○	ANY_STRING.CSV/BIN
Data logging setting file	Common setting file	×	×	×	○	LOGCOM.LCS
	Individual setting file	×	×	○	○	LOGnn.LIS*2
Remote password		×	×	○	○	00000001.SYP

*1 mmm indicates a value calculated by dividing the module I/O No. by 10H (3 digits in hexadecimal). For the CPU module, it will be 3FFH. Also, nn is the serial number (2-digit hexadecimal number) of a module extension parameter of each module.

*2 nn corresponds to the setting number and is 01 through 10.

*3 Can be stored but cannot operate as a function.

*4 When a program or a FB (function) program is stored in the built-in memory of the CPU module, it is divided tor the program memory and the data memory.

*5 Module extension parameter for the protocol setting, storing protocol setting information in the predefined protocol support function

A

Memory capacity

The following table lists the memory capacity of each memory.

Item			R04CPU	R08CPU	R16CPU	R32CPU	R120CPU
Memory capacity	Program capacity		40K steps (160K bytes)	80K steps (320K bytes)	160K steps (640K bytes)	320K steps (1280K bytes)	1200K steps (4800K bytes)
	Program memory		160K bytes	320K bytes	640K bytes	1280K bytes	4800K bytes
	SD memory card		Differs depending on the SD memory card used. (SD/SDHC memory card: 32G bytes maximum)				
	Device/label memory	Total	400K bytes	1188K bytes	1720K bytes	2316K bytes	3380K bytes
		Device area*1	80K bytes				
		Label area*1	60K bytes	80K bytes	100K bytes	180K bytes	220K bytes
		Latch label area*1	4K bytes			8K bytes	
		File storage area*1	256K bytes	1024K bytes	1536K bytes	2048K bytes	3072K bytes
	Data memory		2M bytes	5M bytes	10M bytes	20M bytes	40M bytes
	CPU buffer memory		1072K bytes (536K words) (including the fixed scan communication area (24K words))				
	Refresh memory		2048K bytes*2				

*1 The capacity of device area, label area, latch label area, and file storage area can be changed in parameter. The capacity of the device/label memory can be increased by inserting an extended SRAM cassette.

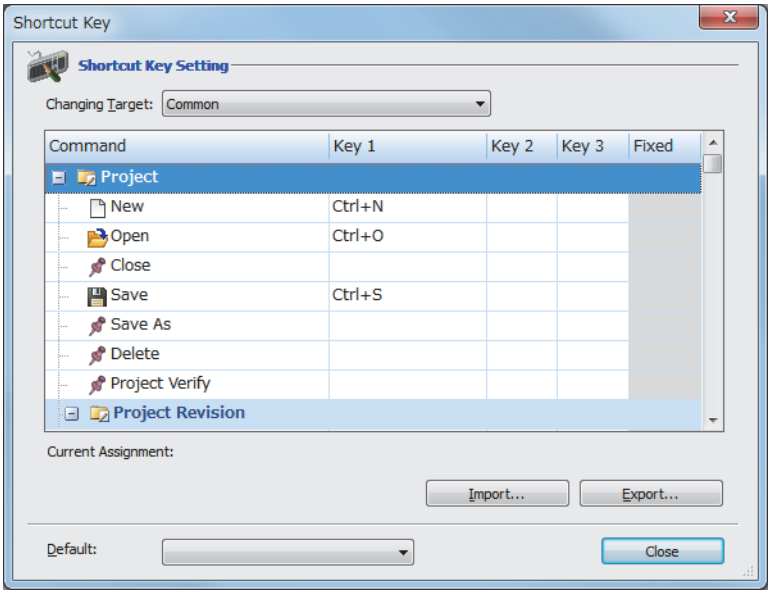
*2 This is the total capacity of the device area and module label area.

Appendix 6 Checking and Setting Shortcut Keys

Shortcut keys of each function can be checked and set on the "Shortcut Key" window.
Up to three keys can be assigned to one command.

Window

[Tool] → [Shortcut Key]



Operating procedure

1. Double-click a command cell to which a shortcut key is to be set.
2. Press keys to be assigned with a keyboard.
3. Click the [Close] button.

■Applying the default setting

Select a format in the drop-down list of "Default" to apply the setting of shortcut keys.
Select one of the following formats.

- Change to GX Works3 Format: Restores the initial setting.
- Change to GPPA Format: Changes the key assignment of all commands to that of GPPA.
- Change to GPPW Format: Changes the key assignment of all commands to that of GX Developer.
- Change to MEDOC: Changes the key assignment of all commands to that of MELSEC MEDOC.

Point

To share the shortcut key setting with other personal computers, import an exported file (*.gks).
The setting file exported from GX Works2 can also be imported.

■Buttons in the window

Import...

Click this button to import a saved shortcut key setting file (*.gks).

Export...

Click this button to save set shortcut keys as a shortcut key setting file (*.gks).

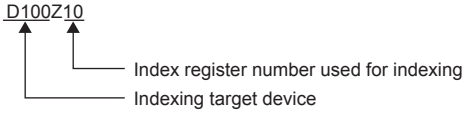
A

Appendix 7 Index Modification

Specify the device number using the index register. The device number to be used is "Device number of device targeted for modification" + "Contents of index register".

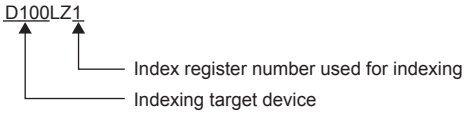
16-bit index modification

The device number is modified using the index register (Z). The modification range for the device in the case of the 16-bit index modification is -32768 to 32767.



32-bit index modification

The device number is modified using the long index register (LZ). The modification range for the device in the case of the 32-bit index modification is -2147483648 to 2147483647.



Devices for which index modification can be performed

The following table lists the devices that can be targeted for index modification.

Item	Device
16-bit index modification	X, Y, M, L, B, F, SB, V, T*1, LT*1, ST*1, LST*1, C*1, LC*1, D, W, SW, SM, SD, Jn\X, Jn\Y, Jn\B, Jn\SB, Jn\W, Jn\SW, Un\G, U3En\G, U3En\HG, R, ZR, RD, P*3, I*3, J, U, K, H
32-bit index modification	M, B, SB, T*1, LT*1, ST*1, LST*1, C*1, LC*1, D, W, SW, Jn\B*2, Jn\W*2, Un\G*2, U3En\G*2, U3En\HG*2, R, ZR, RD, K, H

*1 Can be used for the contact, coil and current value.
*2 For network numbers and the specification source of I/O numbers, 32-bit-based index modification cannot be used.
*3 When it is used as an interrupt pointer, index modification cannot be performed.

Combination of index modification

This section describes the combination of index modification

■Order of device specification and index modification

According to the priority order shown below, the device specification (digit specification, bit specification, indirect specification) and index modification can be applied. However, some word devices may not follow the priority order shown below.

Order of priority	When the device targeted for the device specification and index modification is a bit device	When the device targeted for the device specification and index modification is a word device
High ↑ ↓ Low	1: Index modification 2: Digit specification	1: Index modification 2: Indirect specification 3: Bit specification

■Specification method combined with device specification

The device targeted for specification is modified in order of: 1st modification, 2nd modification and then 3rd modification. Besides, the following contents can be used only for the device for which the 1st modification can be applied. (For example, index modification + digit specification is impossible for the function input (FX).)

Device targeted for specification	1st modification	2nd modification	3rd modification	Example
Bit device	Index modification	Indirect specification	—	K4M100Z2
Word device	Index modification	Bit specification	—	D10Z2.0
	Index modification	Indirect specification	—	@D10Z2
	Bit specification	Index modification	—	D10.8Z2
	Indirect specification	Bit specification	—	@D10.8
	Index modification	Indirect specification	Bit specification	@D10Z2.8
	Indirect specification	Bit specification	Index modification	@D10.8Z2

Precautions

This section describes the precautions on using index modification.

■Index modification between the FOR and NEXT instructions

Between the FOR instruction and the NEXT instruction, pulse output is provided through the edge relay (V). However, pulse output by the PLS, PLF, or pulse conversion (□P) instruction is not available

■Index modification by the CALL instruction

In the CALL instruction, pulse output is provided through the edge relay (V). However, pulse output by the PLS, PLF, or pulse conversion (□P) instruction is not available

■Device range check for index modification

For details on the device range check performed when index modification is used, refer to the following.

📖 MELSEC iQ-R Programming Manual (Instructions, Standard Functions/Function Blocks)

■Change of the index modification range (16-bit ←→ 32-bit modification)

To change the index modification range for switching from 16 bit to 32 bit, the user must:

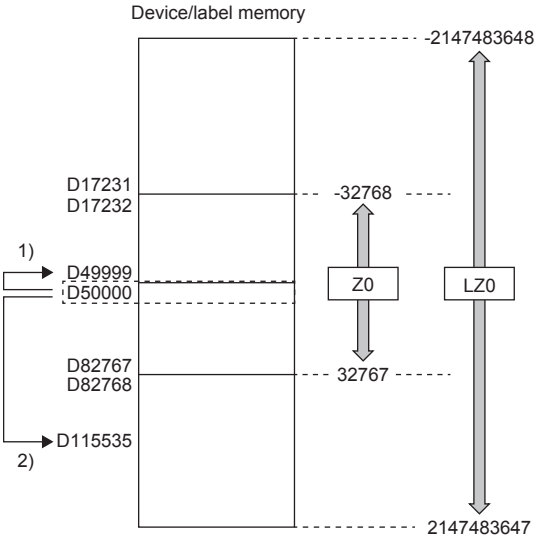
- Review the index modification block(s) within the program.
- For 32-bit-based index modification with ZZ expression, because the specified index register (Zn) and the immediately following index register (Zn+1) are used, caution must be taken to prevent duplicated index registers from being used.
- Review the number of points of the index register (Z) and that of the long index register (LZ), which are specified in "Index Register Setting".

When values are stored in the index register

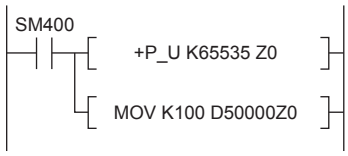
For 16-bit-based index modification using the index register (Z), the range is -32768 to 32767. Therefore, when values within the range from 32768 to 65535 are stored in the index register (Z) for an instruction which processes unsigned data, the instruction does not work in design because the range of the index modification will be -32768 to 32767. For the range of values larger than or equal to 32768, the long index register (LZ) must be used so that 32-bit-based index modification can be applied.

Ex.

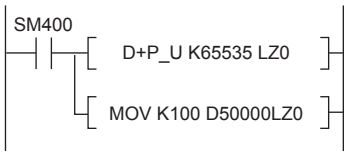
Operation for Index modification



1) Unintended indexing operation



2) Normal indexing operation



(1) When the value 65535 is stored in the index register (Z), D50000(-1) to D49999 are accessed because the value is turned into -1 when index modification is applied.

(2) When a value larger than or equal to 32768 is used for index modification, the value must be stored in the long index register (LZ). In doing so, the value 65535 is used as such for index modification using the long index register (LZ) and D50000 (65535) to D115535 become accessible.

Appendix 8 FB (Function Block)

Appendix 8.1 FB

FB is an abbreviation for a function block. Users can convert frequently-used ladder blocks into FBs and utilize them in a sequence program.

Use of FBs improves the efficiency of program development and reduce mistakes in programming, improving the quality of a program.

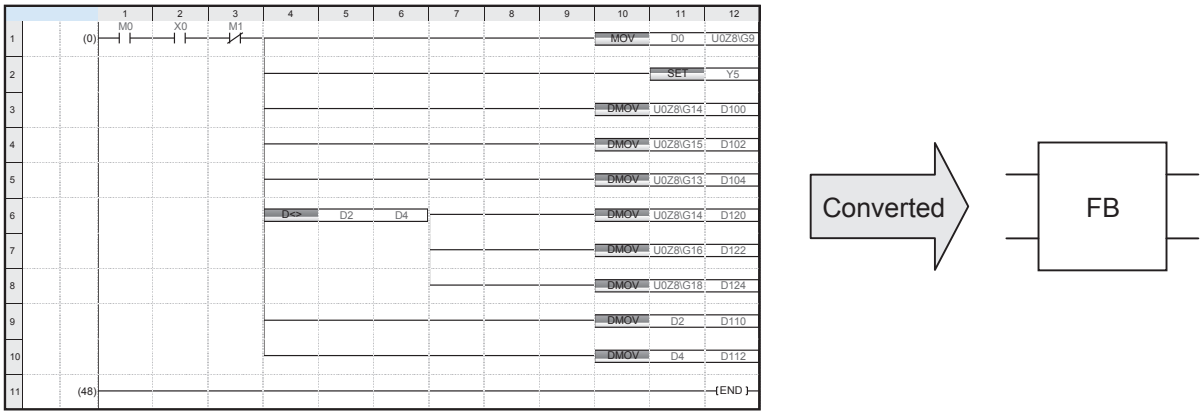


Figure APP 8.1 Converting a sequence program into an FB

A

Appendix 8.1.1 FB conversion

This section describes a flow in which ladder blocks are converted into an FB.

Labels (global labels and local labels) to be registered on the label editor and module labels (global labels) dedicated for a module are prepared.

- Global label: A label that can be used in all programs in a project
- Local label: A label that is used in each program
- Module label: A label where I/O signals and buffer memory areas of a module in use have already been defined. Use of module labels allows users to do programming without considering module internal addresses.

For details on label types, classes, and data types, refer to the following.

📖 MELSEC iQ-R Programming Manual (Program Design)

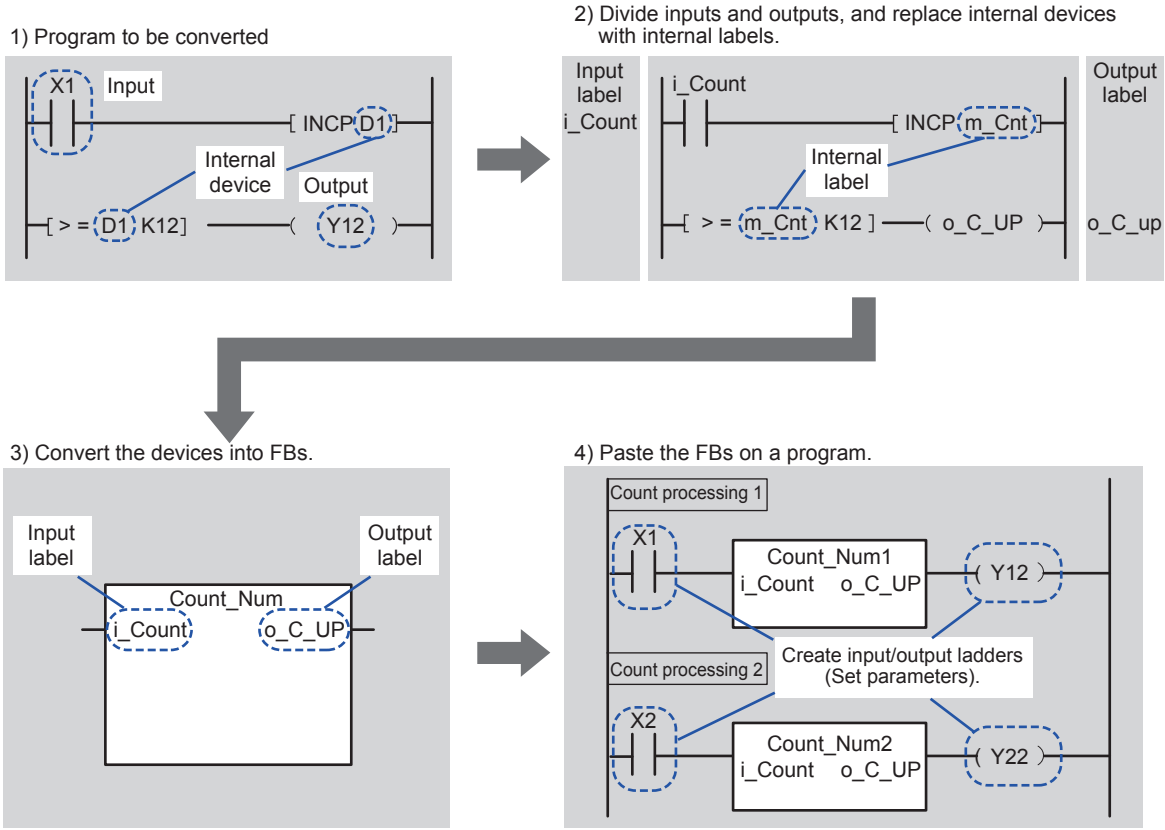


Figure APP 8.1.1 Flow of FB conversion

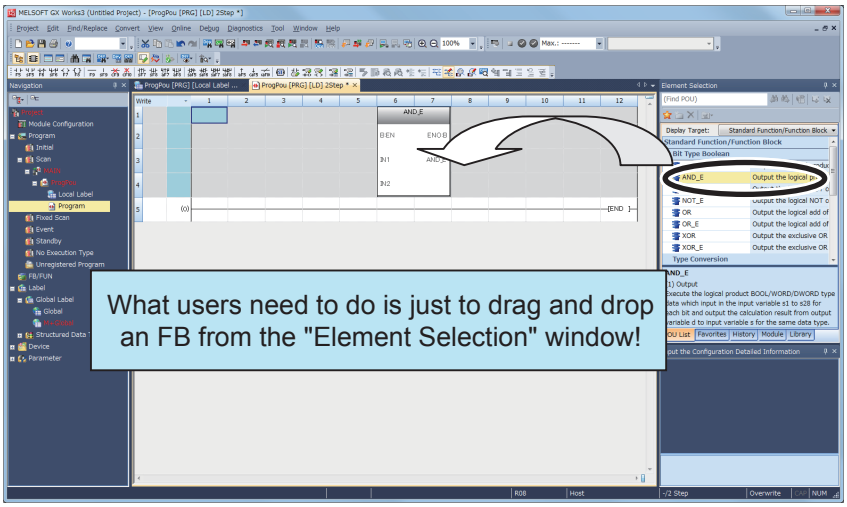
Appendix 8.1.2 Advantages of using FBs

This section describes advantages of using FBs in the creation of a program.

(1) Easy programming

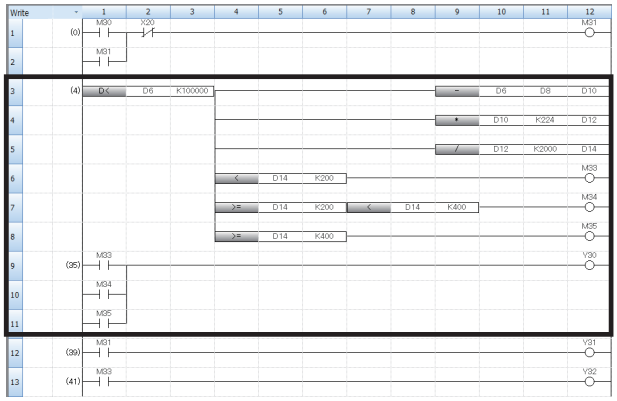
Users can create a sequence program by simply pasting FBs.

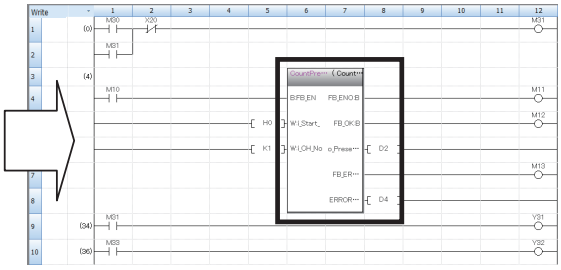
This advantage significantly reduces the man-hours for developing a program. (Using an FB library provided by Mitsubishi Electric Corporation allows users to create programs more easily.)



(2) Easy to read!

Using FBs in a sequence program improves the visibility of a program that only contains "boxes" (FBs), inputs, and outputs.



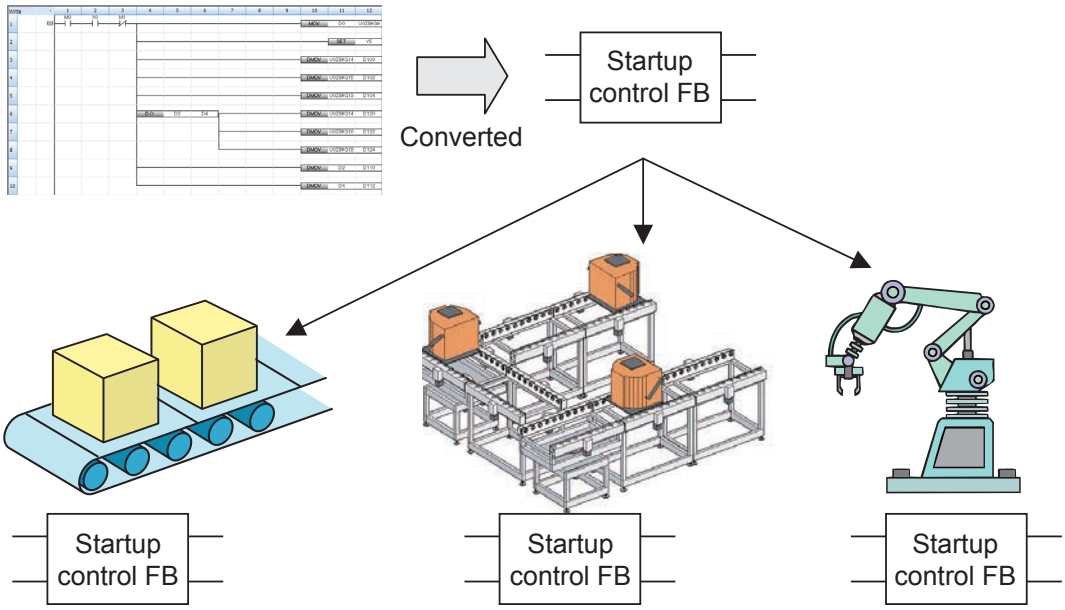


A

(3) Utilizable programs

By converting standard programs into FBs, users can utilize the programs many times.

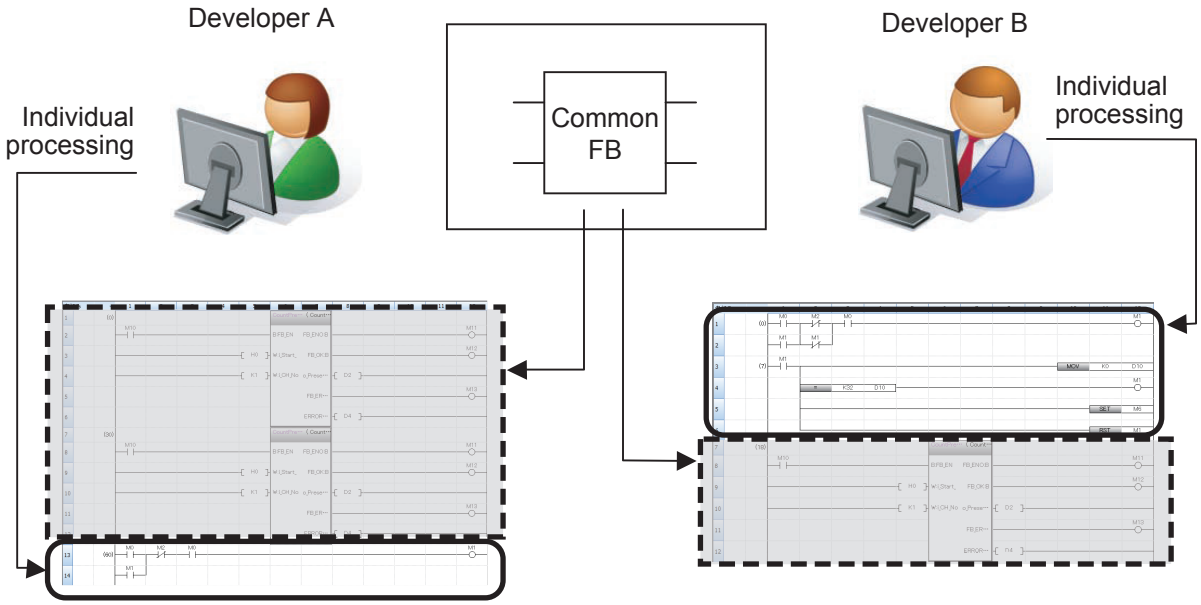
Operations, such as copying a sequence program and modifying devices, which have often been required in the past, will be unnecessary.



(4) Improved quality of programs

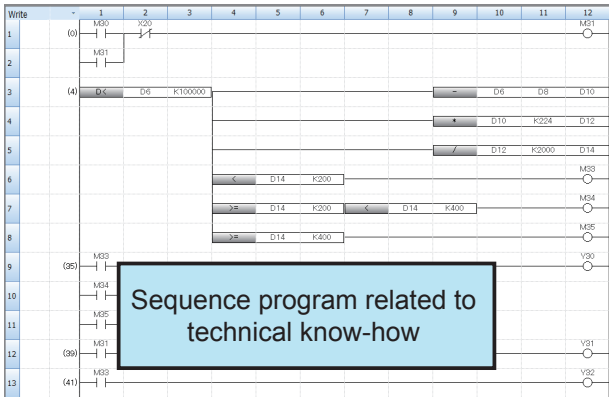
By converting standard programs into FBs and utilizing them, users can develop uniformly-high quality programs without depending on the technological skills that program developers have.

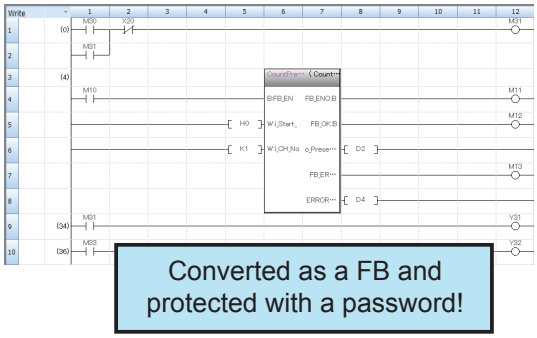
Although each of developers A and B creates a sequence program for a different device, they can use the same FBs for common processing and the quality of their programs is uniformed.



(5) Properties can be protected!

By converting sequence programs into FBs and setting passwords on them, users can protect their technical know-how.

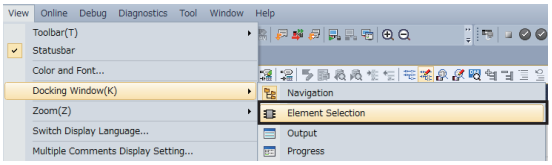




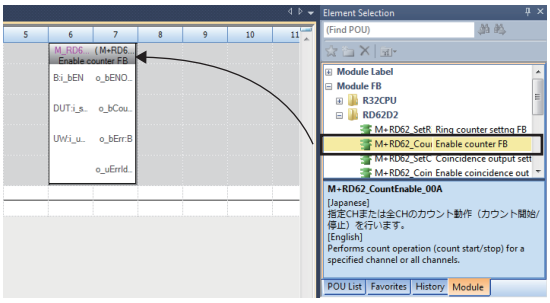
How to insert an FB

The following describes the procedure of inserting an FB.

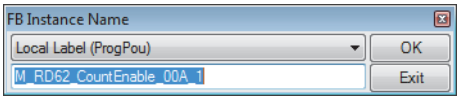
Operating procedure



1. Click [View] ⇨ [Docking Window] ⇨ [Element Selection] from the menu.



2. Select an FB from the "Element Selection" window, and drag and drop it to the desired position on the ladder editor.

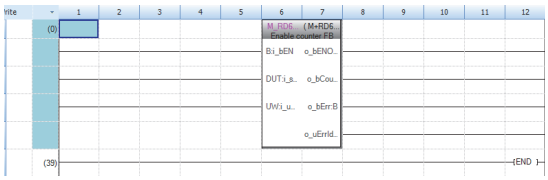


3. The "FB Instance Name" window appears. Select the target label (global label or local label), and enter an instance name.

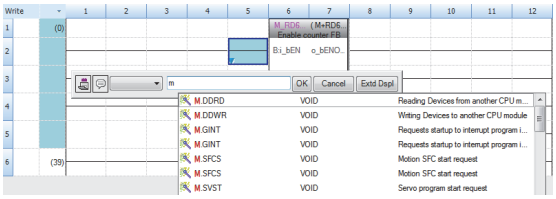
(To the next page)

A

(From the previous page)



4. Select [Convert] ⇨ [Convert] on the menu bar. The ladder is converted, and the rungs are connected to the input and output labels of the FB instance.



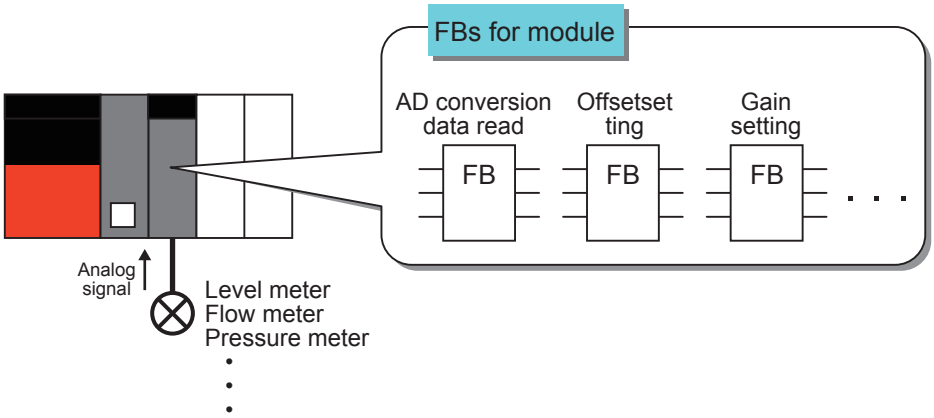
5. Add the input and output parts of the inserted FB to complete the program.

Appendix 8.1.3 FB library

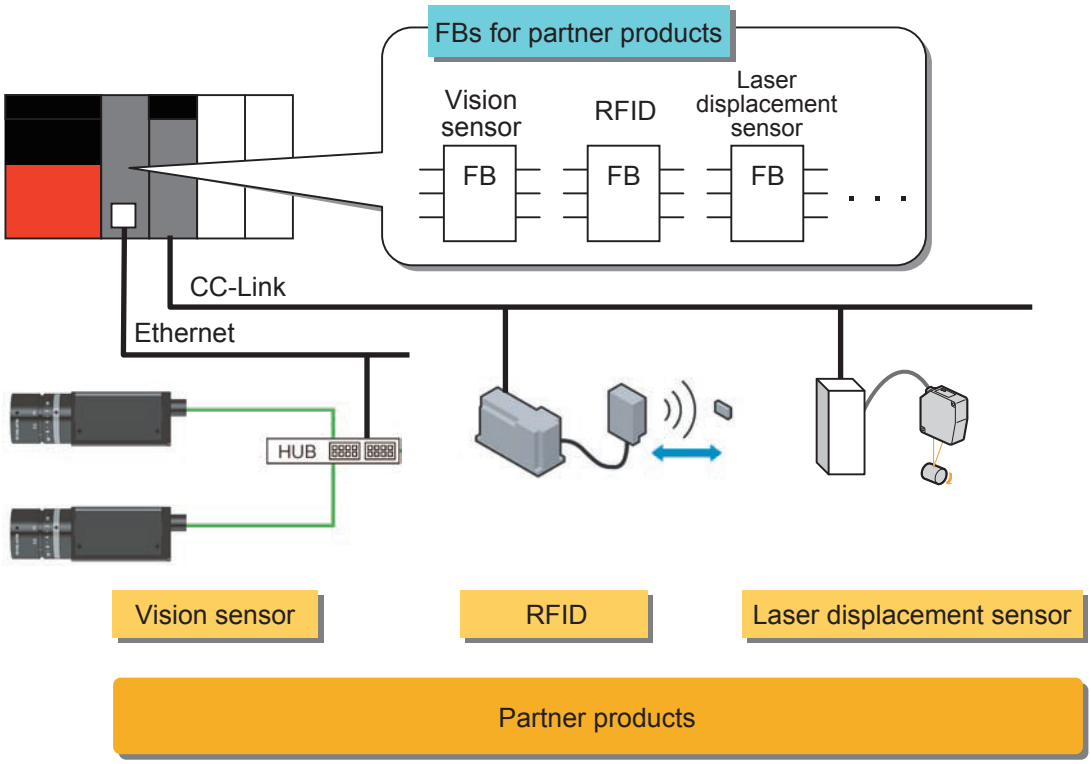
An FB library is a collection of FBs available in GX Works3.

By using the FB library, users can easily create programs for the MELSEC-Q/L series modules and partner products.

■Example: Programmable controller modules



■Example: Partner products



(1) FB library lineup

"FBs for programmable controller modules" or "FBs for partner products" are provided as FB libraries.

(2) How to get an FB library

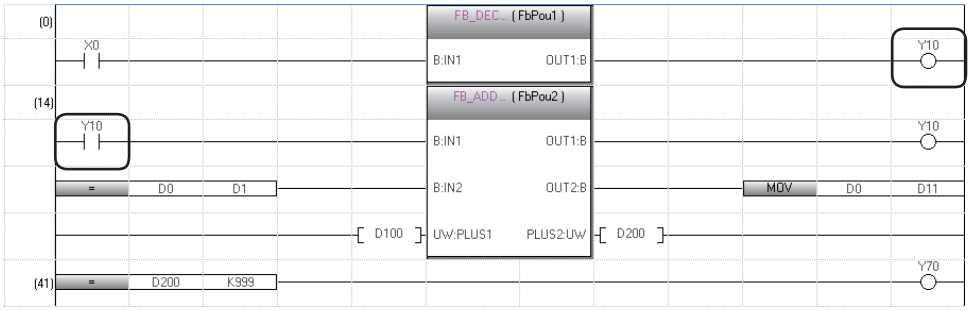
For how to get an FB library, please consult your local Mitsubishi representative.

A

Appendix 8.1.4 Precautions for using FBs

Before using FBs, read the following precautions.

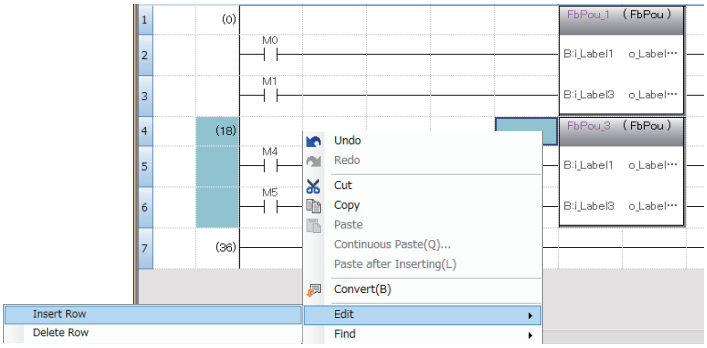
- Only one FB can be pasted on one ladder block.
  An output of an FB instance cannot be directly connected to an input of another FB instance.
  To connect FBs each other, create a coil that receives outputs of an FB, and connect a contact of the coil to an input of another FB.



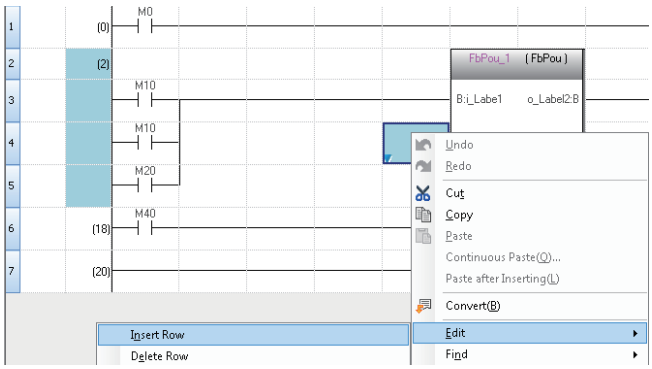
- When the label setting of an FB has been changed, convert the program or all programs.

Point

- To insert an FB instance between FB instances, select [Edit] → [Insert Row] and add an empty row and insert the FB instance there.



- Ladder blocks can be created in parallel at an input part of an FB instance.
  To add a ladder between parallel ladder blocks, select the second row of the input part, and select [Edit] → [Insert Row] to add an empty row and create a ladder as shown below.



Appendix 8.2 Creating a program using FBs

For the operating procedure of using FBs, refer to Appendix 8.1.2 Advantages of using FBs.

Creating a program

Create the program described in the previous section using FBs.

A

MEMO

Mitsubishi Programmable Controllers Training Manual
MELSEC iQ-R Series Basic Course (for GX Works3)

MODEL	
MODEL CODE	
SH(NA)081898ENG-A (1602) MEE	

MITSUBISHI ELECTRIC CORPORATION

HEAD OFFICE : TOKYO BUILDING, 2-7-3 MARUNOUCHI, CHIYODA-KU, TOKYO 100-8310, JAPAN
NAGOYA WORKS : 1-14 , YADA-MINAMI 5-CHOME , HIGASHI-KU, NAGOYA , JAPAN

When exported from Japan, this manual does not require application to the Ministry of Economy, Trade and Industry for service transaction permission.

Specifications subject to change without notice.