

TOSHIBA

**PROGRAMMABLE CONTROLLER
EX-SERIES**

EX100

USER'S MANUAL V2.1

Important information

Misuse of this equipment can result in property damage or human injury. Because controlled system applications vary widely, you should satisfy yourself as to the acceptability of this equipment for your intended purpose. In no event will Toshiba Corporation be responsible or liable for either indirect or consequential damage or injury that may result from the use of this equipment. No patent liability is assumed by Toshiba Corporation with respect to the use of information, illustrations, circuits, equipment, or examples of applications in this publication.

Toshiba Corporation reserves the right to make changes and improvements to this publication and/or related products at any time without notice. No obligation shall be incurred, except as noted in this publication.

This publication is protected by copyright and contains proprietary material. No part of this book may be reproduced, stored in a retrieval system, or transmitted in any form or by any means — electrical, mechanical, by photocopying, recording or otherwise — without obtaining prior written permission from Toshiba.

Copyright 1988 by Toshiba Corporation
Tokyo, Japan

Published Oct. 1988, 2nd edition May, 1990

Inside this manual

This manual has been prepared for first-time users of the EX100 Programmable Controller (referred to in this manual as the EX100) to enable a full understanding of the configuration of the equipment, and to enable the user to obtain the maximum benefits of the equipment.

Sections 1 and 2 outline the EX100 configuration. To fully understand the EX100, it is important to read these chapters carefully. Sections 3 and 4 describe the hardware used in designing external circuits and panels. Sections 5 to 9 are mainly concerned with software. Sections 10 and 11 describe the maintenance procedure for the EX100, to ensure safe operation and long service life.

This manual has been edited for EX100 CPU version 2.1 (V2.1). The enhanced functions of this version are listed below.

Functions	Enhanced		Conventional	
	PU11A	PU12A	PU11	PU12
Software version	V2.1	V2.1	V1. X	V1. X
"Day of the week" in calendar	N / A	Yes	N / A	No
Communication priority mode	Yes	Yes	No	No
EEPROM Write <u>instruction</u>	Yes	Yes	No	No
EEPROM Read <u>instruction</u>	Yes	Yes	No	No
Calendar initializing instruction	N / A	Yes	N / A	No
Data output to special modules	Yes	Yes	No	No
Data input from special modules	Yes	Yes	No	No

Details of each function are explained in Section 9.

Inside each section

The contents of this manual are as follows:

Section 1 Introduction

Introduces the features of the EX100, the names of its components, and describes handling precautions.

Section 2 System configuration

Describes the EX100 input and output configuration, and the equipment that constitutes the EX100.

Section 3 Specifications

Contains the external dimensions of the EX100 and input and output specifications.

Section 4 Installation and Wiring

Describes installation procedures and the wiring method.

Section 5 Operating the EX100

Describe the configuration of the internal memory and the method of operating the EX100.

Section 6 Input and Output Allocation

Explains the assignment of the input and output numbers.

Section 7 Instructions

Describes the various instructions of the EX100 in detail.

Section 8 Basic Programming Procedures

Describes the procedures for starting the EX100, for executing a program, and other operating procedures.

Section 9 Special Functions

Describes the unique special functions of the EX100 and their use.

Section 10 Maintenance

Describes the precautions and maintenance procedures for ensuring reliable operation of the EX100.

Section 11 Troubleshooting

Lists the causes of typical problems and the items of the EX100's diagnostic check.

Appendices

List the internal current consumption of each module, the execution time of each instruction, and module part numbers.

Note and caution symbols

Users of this manual should pay special attention to information preceded by the following symbols.



NOTE

Calls the reader's attention to information considered important for full understanding of programming procedures and / or operation of the equipment.



CAUTION

Calls the reader's attention to conditions or practices that could damage the equipment or render it temporarily inoperative.

Related manuals

The following related manuals are available for the EX100.

Graphic Programmer (GP) Operation Manual
Handy Programmer (HP) Operation Manual
Miniprogrammer (MP) Operation Manual
EX100 Computer Link User's Manual
Motion Control Module User's Manual

Terminology

The following is a list of abbreviations and acronyms used in this manual.

AWG	American Wire Gage
ASCII	American Standard Code for Information Interchange
CPU	Central Processing Unit
EEPROM	Electrically Erasable Programmable Read Only Memory
EPROM	Erasable Programmable Read Only Memory
H	hexadecimal (when it appears in front of an alphanumeric string)
I / O	Input / Output
LCD	Liquid Crystal Display
LED	Light Emitting Diode
ms	millisecond
NEMA	National Electrical Manufacturers' Association
PC	Programmable Controller
PROM	Programmable Read Only Memory
RAM	Random Access Memory
ROM	Read Only Memory
μs	microsecond
Vac	ac voltage
Vdc	dc voltage

1	Introduction	
1.1	Introducing the EX100	7
1.2	EX100 features	8
1.3	Important items	9
2	System Configuration	
2.1	System configuration	11
2.2	Names and functions of the individual components	12
2.3	Input and output configuration	13
2.4	EX100 modules	13
2.4.1	The rack	13
2.4.2	The power supply module	14
2.4.3	The CPU module	15
2.4.4	I / O modules	17
2.4.5	Data link modules	18
2.5	Configuration of the data link system	19
2.6	Peripheral devices	20
2.7	Options	22
3	Specifications	
3.1	General specifications	23
3.2	External dimensions	23
3.3	Functional specifications	24
3.4	I / O specifications	25
4	Installation and Wiring	
4.1	Operating environment	47
4.2	Installing the unit	47
4.3	Mounting the modules	48
4.4	Connecting the expansion unit	49
4.5	Grounding	49
4.5.1	Check points for grounding	50
4.5.2	Grounding methods according to the installation	50
4.6	Wiring the power supply	52
4.7	Wiring the modules and application precautions	53
4.7.1	Module layout and I / O wiring	53
4.7.2	Dimensions of the terminal block	54
4.7.3	Application precautions for input modules	54
4.7.4	Application precautions for transistor output modules	57
4.7.5	Application precautions for triac output modules	58
4.7.6	Application precautions for relay output modules	58
4.7.7	Application precautions for analog input modules	59
4.7.8	Application precautions for analog output modules	59

5	Operating the EX100	
5.1	EEPROM operation	61
5.2	Memory settings	62
5.3	Operation modes	64
5.4	Scanning	66
6	Programming	
6.1	Devices and registers	69
6.2	I / O allocation	75
6.3	Setting the retentive memory area	82
6.4	Program configuration and execution	83
7	Instructions	
7.1	List of instructions	85
7.2	Basic ladder functions	92
7.3	Data transfer instructions	105
7.4	Arithmetic operations	125
7.5	Logical operations	141
7.6	Data conversion instructions	152
7.7	Special functions	158
7.8	Special ladder functions	169
8	Basic Programming Procedures	
8.1	System design overview	181
8.2	Basic programming procedures	182
9	Special Functions	
9.1	The clock-calendar function	185
9.2	Forced operation (Auto RUN-F)	186
9.3	The write-protect function	187
9.4	The hold function	187
9.5	The EEPROM read / write functions	188
9.6	Communication priority mode	189
9.7	Data input / output functions for special modules	190
10	Maintenance and Checks	
10.1	Daily checks	191
10.2	Periodic checks	192
10.3	Spare parts to keep in stock	193
10.4	Removing and installing the optional battery	193
11	Troubleshooting	
11.1	Troubleshooting procedure	195
11.2	List of items for diagnostic check	198
Appendix A	Module current consumption	201
Appendix B	Instruction execution times	202
Appendix C	List of models and types	203

1.1 Introducing the EX100

The EX100 is a compact, high-performance programmable controller with a range of 28 to 480 input and output points.

I / O points — The EX100 can support up to 15 I / O modules in the maximum configuration. Therefore, if 16-point modules (terminal block type) are installed in each slot, the EX100 can control up to 240 points. If 32-point modules (connector type) are installed in each slot, it can control up to 480 points.

Memory capacity — Program memory capacity can be selected in either 4K steps or 3K steps by switch settings. If the 3K-step mode is selected, 1K words of data are stored in EERPOM. (Program is stored in EEPROM)

Control functions — In addition to the basic relay ladder functions, the EX100 provides functions such as data operations, arithmetic operations, various functions, etc. Furthermore, its analog control functions, positioning functions and data communication functions allow its application to wide scope of control systems.

Construction — The EX100's sturdy, compact, modular construction make it an ideal choice for all industrial applications.

Series compatibility — EX100 programs are compatible with other members of the EX Series: the EX200B, EX250 and EX500. Peripheral equipment can also be shared.

1. Introduction

- 1.2 EX100 features**
- Battery-free** — The EX100 CPU has a standard built-in EEPROM, permitting operation without need of a battery. If the memory setting is 3K mode, the variable data can be written into and / or read from the EEPROM, providing completely maintenance-free back-up operation.
 - High speed execution** — Quick response owing to high speed program execution of 0.9 μ s per contact instruction and immediate I / O update instructions.
 - On-line program changing** — On-line (during RUN) program changes are possible, providing efficient program debugging and simulation.
 - Clock-calendar** — The EX100 is also available with the clock-calendar function (year, month, day, day of the week, hours, minutes, seconds), which is a powerful tool for performing scheduled operations and batch processing.
 - Efficient data link network** — The EX100 can communicate with a computer via the RS-485 interface, with other EX series PCs and remote I / O stations via TOSLINE-30, allowing use of an efficient network environment.
 - Write-protect function** — The operation control switch on the EX100 is a key switch, which can set to the protect position. This ensures that the program is not affected by unauthorized operations. It is also possible to protect a program by a special program ID.
 - Built-in 24 Vdc power supply** — The EX100 is equipped with a 24 Vdc power supply for external devices as well as for the EX100 I / O modules. (24 Vdc, $\pm 10\%$ - 0.5 A)

1.3 Important items

Although the EX100 is designed to withstand severe environmental conditions, it is necessary to observe the following installation, wiring, and use guidelines during operation and storage.

Handling and operation

- Because the EX100 has an EEPROM as standard memory, it does not require a battery. When power to the EX100 is turned on, the program stored in the EEPROM is transferred to the RAM for execution. Therefore, if the program in the RAM is changed, it is necessary to write the updated program into the EEPROM BEFORE turning the power off. Otherwise, the updated program will be lost, and the old program will be retained.

Environment

- Ambient temperature and humidity
During operation: 0° to 55°C (32° to 131°F), 20% to 90% RH
During storage: -20° to 75°C (-4° to 167°F), 20% to 90% RH
- Because the EX100 is sensitive electronics equipment, it is necessary to avoid abrupt changes in temperature and humidity.
- Do not install the EX100 in locations subject to excessive shocks or vibrations.

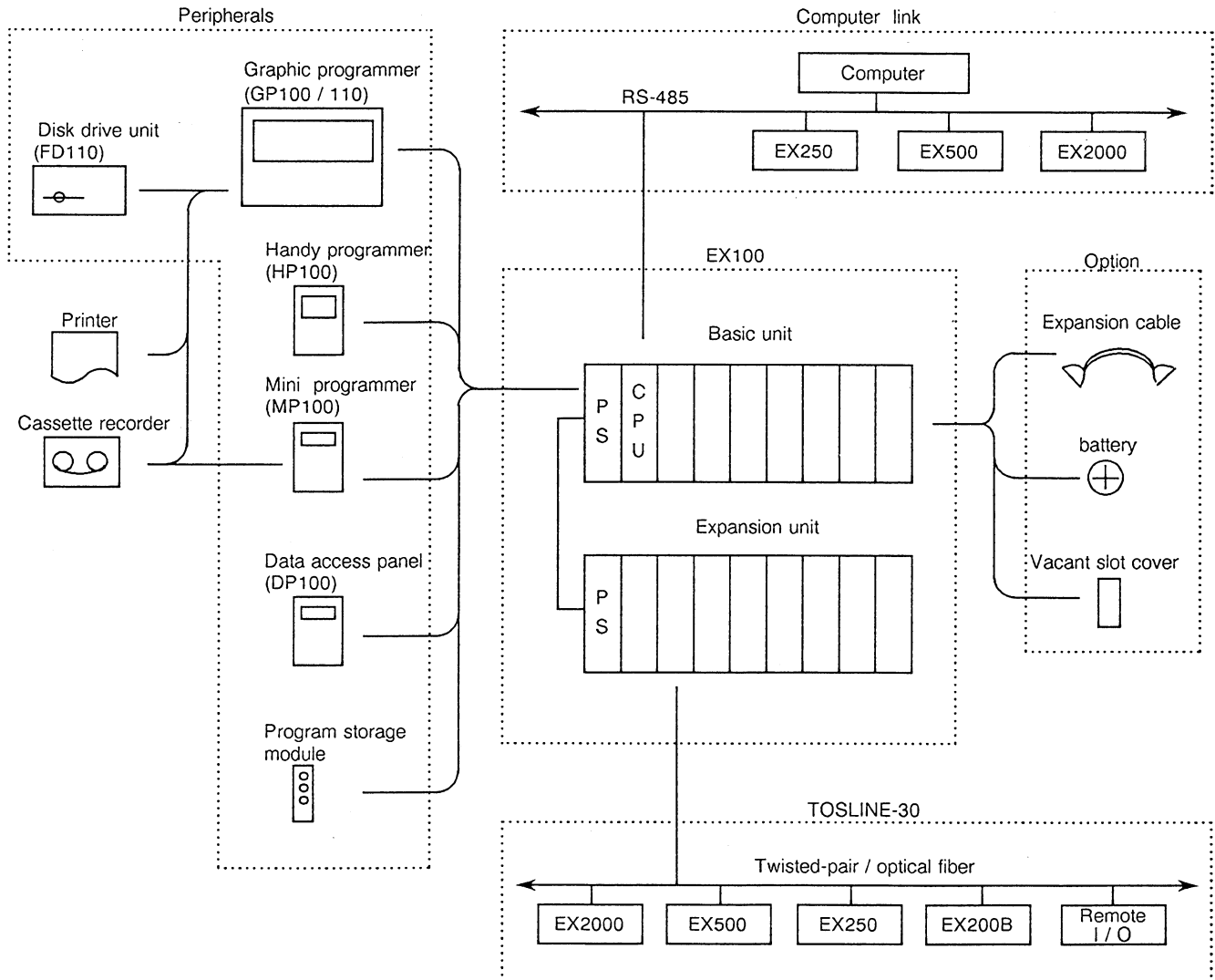
Installation and wiring

- When wiring the EX100 and installing it on a control panel, be sure that fragments of wire or other metal scraps do not fall into the ventilation hole on top of the unit.
- Proper grounding is very important for safety and for the EX100 to operate reliably. Be sure to ground the unit correctly, by referring to Section 4 of this manual.
- Be sure that power is turned off when connecting or disconnecting the expansion cable. Also, be sure that power is turned off when mounting or removing an I / O module.
- When using the expansion unit, supply power simultaneously to the basic and expansion units.

2. System Configuration

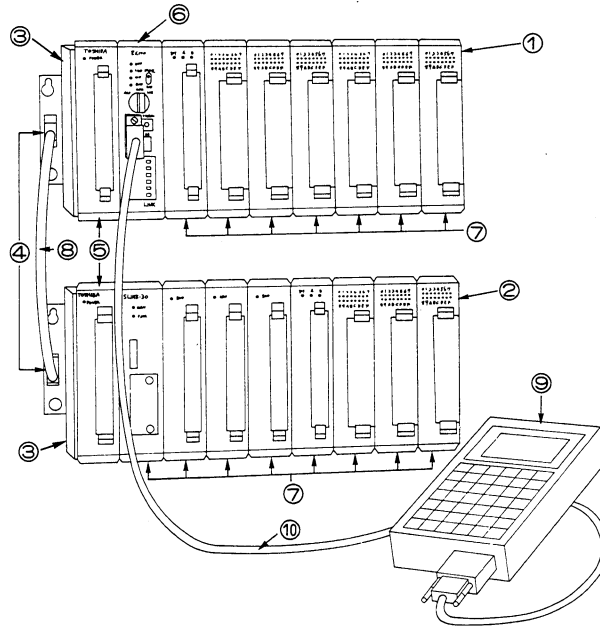
2.1 System configuration

The following figure shows the EX100 system configuration.



2. System Configuration

2.2 Names and functions of individual components



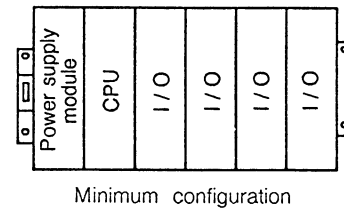
- ① **Basic unit** — The basic unit consists of the rack, the power supply module, the CPU, and I / O modules. The basic unit is the basic construction of the EX100.
- ② **Expansion unit** — If the number of I / O slots is not sufficient, the expansion unit is used to accommodate additional I / O modules as well as the power supply module. The expansion unit is connected directly to the basic unit.
- ③ **Rack** — The rack holds modules, such as the power supply, the CPU and I / O modules.
- ④ **The expansion connector** — The expansion connector is the socket for the cable connecting the expansion unit to the basic unit.
- ⑤ **The power supply module** — The power supply module supplies 5 Vdc power for the CPU and I / O modules from the external power, such as 120 Vac or 240 Vac. This module is mounted in the extreme left slot of the rack.
- ⑥ **The CPU module** — The CPU module is the “brain” of the EX100 and is mounted next to the power supply module on the rack. This module reads the input status, solves the user program, and controls the output status.
- ⑦ **I / O modules** — Input modules convert external input signals into a format that can be read by the CPU module. Output modules convert signals from the CPU module into a level suitable for external output.
- ⑧ **Expansion cable** — The expansion cable connects the expansion unit to the basic unit.
- ⑨ **Programmer** — The programmer is used to write the program into the CPU module and to monitor the program execution status.
- ⑩ **Programmer cable** — The programmer cable connects the programmer to the CPU module. This cable can be connected and disconnected even when power is on.

2. System Configuration

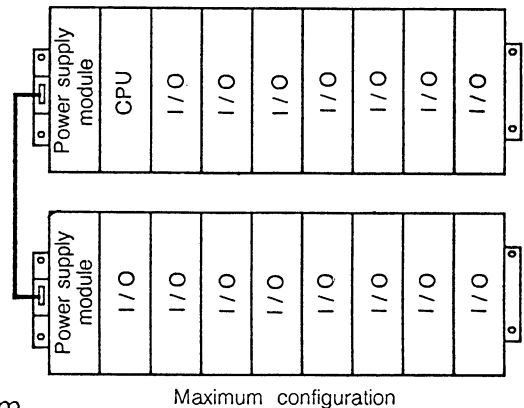
2.3 Input and output configuration

The rack of the EX100 is available in two sizes. The smaller one can accommodate a total of six modules, including the power supply module, and the larger one can accommodate a total of nine modules. Furthermore, there are two types of each rack, the unexpandable type and the expandable type, making a total of four kinds. The expandable rack is equipped with an expansion connector on the left side. In the expansion configuration, two expandable racks are used: one for the basic unit, and one for the expansion unit.

The minimum configuration of the EX100 is a six-slot rack for six modules. This unit can accommodate four I / O modules in addition to the CPU module and the power supply module.



The maximum configuration of the EX100 is two nine-slot racks. This configuration can accommodate up to 15 I / O modules in addition to the CPU module and the two power supply modules. If 32-point I / O modules are mounted in all slots, it is possible to control a maximum of 480 points.



2.4 EX100 modules 2.4.1 The rack

The EX100 consists of the rack, the power supply module, the CPU module, and various I / O modules, as described below. As mentioned in Section 2.3, the rack is available in four types.

Part No.	Slots		Remarks
EX10-UBA1	6 slots	1 × power supply, 1 × CPU, 4 × I / O	Dedicated to the basic unit (Not expandable)
EX10-UBA2	9 slots	1 × power supply, 1 × CPU, 7 × I / O	
EX10-UBB1	6 slots	1 × power supply, 1 × CPU, 4 × I / O (Basic)	For either the basic unit or the expansion unit (Expandable)
		1 × power supply, 5 × I / O (Expansion)	
EX10-UBB2	9 slots	1 × power supply, 1 × CPU, 7 × I / O (Basic)	
		1 × power supply, 1 × CPU, 8 × I / O (Expansion)	



NOTE

The unexpandable type does not have an expansion connector.

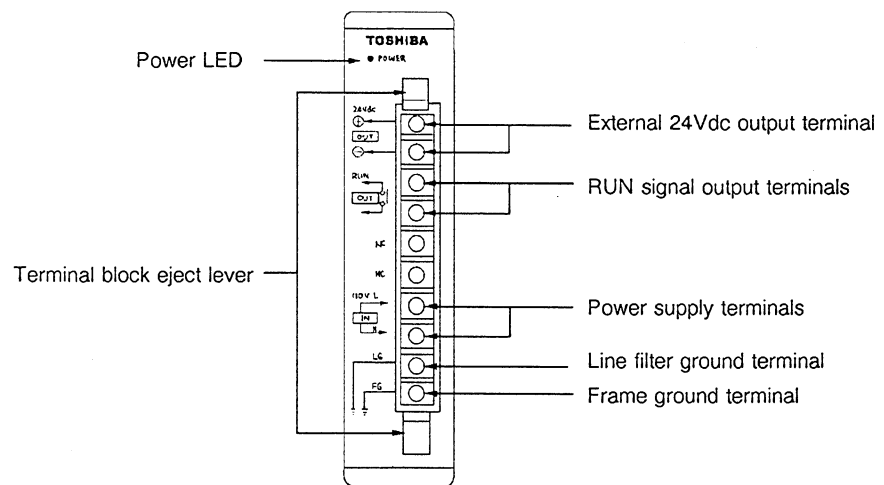
2. System Configuration

2.4.2

The power supply module

The power supply module is mounted in the slot at the extreme left of the rack. The following three types of power supply modules are available, depending on the voltage required.

Part No.	Power Supply Voltage	Output Rating
EX10-MPS51	100 Vac to 120 Vac (+ 10 / - 15%)	Power supply for internal control, 5 V-2.5 A (max.)
EX10-MPS61	200 Vac to 240 Vac (+ 10 / - 15%)	External power supply 24 V ($\pm 10\%$) - 0.5 A (max.)
EX10-MPS31	24 Vdc (+ 20 / - 15%)	Total for internal and external power supply: 15 W or less



External 24Vdc output terminals

These terminals supply 24 Vdc ($\pm 10\%$, 0.5 A max.) to external devices, such as sensors, as well as to the relay output module. 15 W or less, including 5 V for internal power supply. (See Appendix A)

RUN signal output terminals

Built-in NO contact that turns on (contact closes) when the EX100 is in operation, i.e., in the RUN state

240 Vac (+ 10%) / 24 Vdc (+ 20%), 2 A (max.)

These terminals can also be used on the expansion unit.

Power supply terminals

These terminals are connected to the power supply line. See Section 4.6.

Line filter and frame ground terminals

These are grounding terminals. See Section 4.5.

2. System Configuration

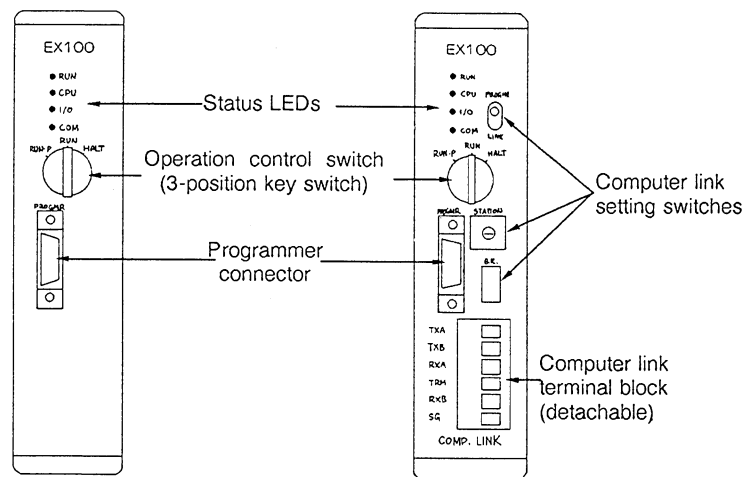
2.4.3 The CPU module

One CPU module is mounted on the basic unit of the EX100. This module must be mounted in the slot next to the power supply module. Two types of CPU modules are available: the standard and the enhanced versions.

Part No.	Function	
EX10-MPU11A	Standard	Standard functions
EX10-MPU12A	Enhanced	Incorporates computer link function and clock-calendar function in addition to the standard functions.

Standard version

Enhanced version



Status LED display

Displays the status of the EX100

- RUN
- CPU
- I / O
- COM

RUN	Lit	In operation (the RUN state)
	Blinking	In the HOLD state
	Not lit	In the HALT state or error state
CPU	Lit	CPU normal
	Blinking	Program abnormal
	Not lit	CPU abnormal
I / O	Lit	I / O normal
	Not lit	I / O abnormal
COM	Blinks during communication with peripherals.	



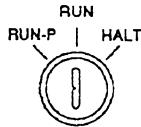
NOTE

If an error occurs in the EX100, this display gives an important clue concerning the nature of the problem. See Section 11 for details.

2. System Configuration

Operation control switch

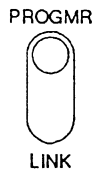
Used to control the operation status of the EX100



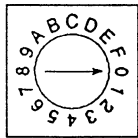
HALT	Program execution stopped. This is the position during normal programming
RUN	Program execution. HALT or RUN can be selected from the programmer. Programming and writing into the EEPROM are possible.
RUN-P	Program execution. HALT or RUN can be selected from the programmer. Programming and writing into the EEPROM are inhibited.

For details of the key switches and operating the EX100, see Section 5.3.

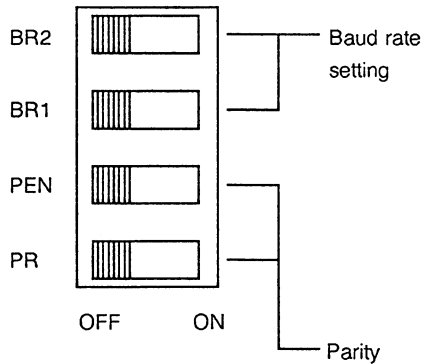
Computer link setting switch (Available only with the enhanced version)



PROGMR	Programmer can be used, computer link cannot be used.
LINK	Computer link can be used, programmer cannot be used.



Use this switch to set the station number for the computer link mode. (0 to 15)



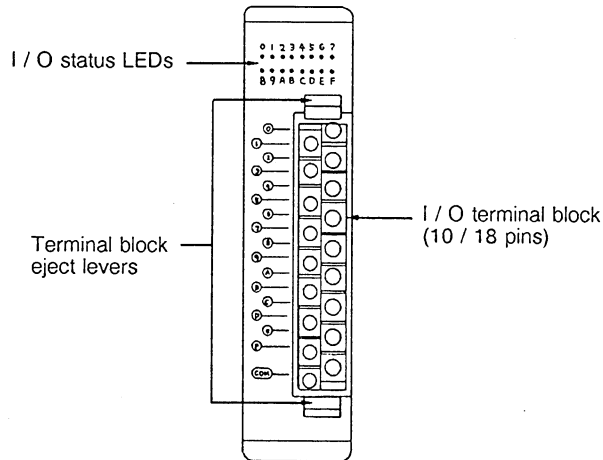
BR2	BR1	Baud rate
OFF	OFF	9600
OFF	ON	4800
ON	OFF	2400
ON	ON	1200

PEN	PR	Parity
OFF	—	None
ON	OFF	ODD
	ON	EVEN

2. System Configuration

2.4.4 I / O modules

As listed below, various I / O modules are available for the EX100, allowing it to be used for a wide variety of applications. The I / O modules can be mounted in any of the slots on the rack in any sequence, although there are recommended configurations to minimize the risk of signal interference.



Part No.	Type	Specifications
EX10•MDI31	dc / ac input	16 points (16 pts common), 12 to 24 Vdc / Vac
EX10•MDI32	dc input	32 points (8 pts common), 24 Vdc
EX10•MIN51	ac input	16 points (16 pts common), 100 to 120 Vac
EX10•MIN61		16 points (16 pts common), 200 to 240 Vac
EX10•MRO61	Relay output	12 points (4 pts common), 240 Vac, +10% / 24 Vdc, +20% (max.), 2 A / pt, 4 A / 4 pts common (max.)
EX10•MRO62		8 points (isolated), 240 Vac, +10% / 24 Vdc, +20% (max.) 2 A / points (max.)
EX10•MDO31	Transistor output	16 points (16 points common), 5 to 24 Vdc, 1 A / point (max.), 1.2 A / 4 points (max.)
EX10•MDO32		32 points (8 points common), 5 to 24 Vdc, 0.1 A / point (max.)
EX10•MAC61	Triac output	12 points (4 points common), 100 to 240 Vac, 0.5 A / point (max.), 0.6 A / SSR (max.)

2. System Configuration

Part No.	Type	Specifications
EX10-MAI21	Analog input	4 channels, 1 to 5 V / 4 to 20 mA, 8-bit resolution
EX10-MAI22		4 channels, 1 to 5 V / 4 to 20 mA, 12-bit resolution
EX10-MAI31		4 channels, 0 to +10 V, 8-bit resolution
EX10-MAI32		4 channels, -10 to 10 V, 12-bit resolution
EX10-MAO31	Analog output	2 channels, 1 to 5 V / 4 to 20 mA / 0 to 10 V, 8-bit resolution
EX10-MAO22		2 channels, 1 to 5 V / 4 to 20 mA, 12-bit resolution
EX10-MAO32		2 channels, -10 to +10 V, 12-bit resolution
EX10-MPI21	Pulse input	1 ch (two phase and zero marker), 5 / 12 V, 100 kpps (max), 24-bit counter
EX10-MMC11	Motion control	1 axis, 200 kpps (max.), 5 to 24 Vdc, $\pm 999,999$ pulses, 64 points data



NOTE For detailed specifications of the I / O modules, see Section 3.4.

2.4.5 Data link modules

The following data link modules are available for the EX100, allowing it to communicate with other EX Series PCs (EX100, EX200B, EX250, EX500 or EX2000) and remote I / O stations.

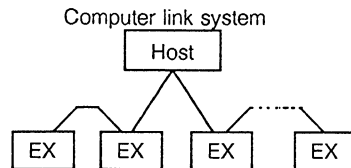
Part No.	Type	Specifications
EX10-MLK11	TOSLINE-30 Twisted-pair	8 / 16 / 32 words cyclic transmission, 187.5 kbps, 1 km max.
EX10-MLK12	TOSLINE-30 Optical fiber	8 / 16 / 32 words cyclic transmission, 375 kbps, 2 km max.

2. System configuration

2.5 Configuration of the data link system

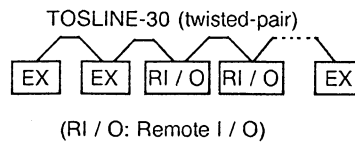
The EX100 supports two types of data link systems, the computer link and TOSLINE-30.

The computer link is a data transmission function between the host computer and the EX100, using the standard RS-485 interface. The data in the EX100 can be read and written by creating a simple communication program on the computer. (Communication via the RS-232C is also possible using a conversion adapter.)

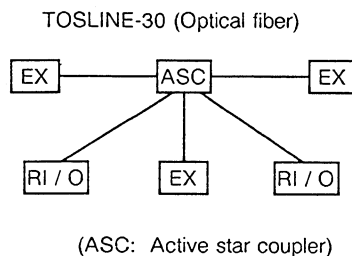


Interface	Conforms to RS-485
Transmission system	Half-duplex 4-wire system
Synchronization system	Start-stop system
Topology	Party line (multi-drop)
Transmission speed	1200 / 2400 / 4800 / 9600 bps
Transmission distance	1 km maximum
No. of stations	16 maximum

TOSLINE-30 is an N:N data link system dedicated to the EX Series, and can simultaneously provide a link between PCs and a remote I / O system. Using TOSLINE-30, data can be exchanged with several EX100 units, or other members of the EX Series PC, by the similar way as that of normal input and output.



Topology	Party line (multi-drop)
Transmission speed	187.5 kbps
Transmission distance	1 km maximum (total)
No. of stations	17 maximum
Transmission capacity	8 / 16 / 32 words (cyclic)
Response speed	25 ms / 32 words
Checking method	Inverted double transmission



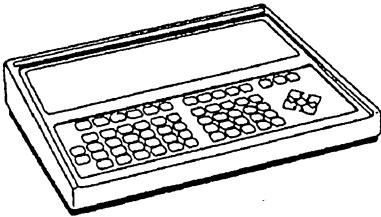
Topology	Star
Transmission speed	375 kbps
Transmission distance	2 km maximum (stn-stn)
No. of stations	16 maximum
Transmission capacity	8 / 16 / 32 words (cyclic)
Response speed	19.2 ms / 32 words
Checking method	Inverted double transmission



Refer to the separate manual for details of the data link system.

2. System configuration

2.6 Peripheral devices



The following peripheral devices are available for the EX100.

The graphic programmer (GP)

The graphic programmer is a multi-function programmer with a large full-dot LCD. In addition to the stand-alone programming function, it has interfaces for an external disk drive, printer, and cassette tape recorder, enabling it to fully support program design for the EX100 system.

The following five GP models are available for the EX100.

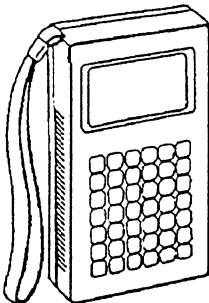
	Enhanced			Conventional	
	GP110	GP110AP1	GP110AP2	GP100	GP100AP
Power voltage	100 to 240 Vac			100 to 120 Vac	
LCD with back light	Yes			No	
Printer I / F	No	Yes		No	Yes
Disk drive I / F	No	Yes		No	
Cassette I / F	Yes				
Stand-alone function	No	Yes		No	Yes
EX2000 support	No		Yes	No	



NOTE

GP100 and GP100AP have the following functional limitations when used with the EX100.

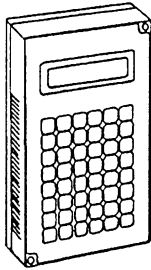
- (1) EEPROM Write command is not supported.
Therefore, the special relay R62E should be used to write the program into EEPROM. (see 5.1)
- (2) "SP", "OPT" and "i" settings are not available in the manual I / O allocation. (see 6.2)
- (3) Immediate input (FUN096) and immediate output (FUN097) instructions cannot be programmed.



The handy programmer (HP)

The handy programmer is a compact, hand-held programmer, that can be used to program the EX100 using ladder diagrams.

2. System configuration

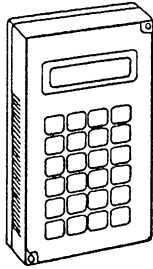


The miniprogrammer (MP)

The miniprogrammer is a compact, hand-held programmer. It is useful for making minor changes to data and to programs. It is also equipped with an interface for a cassette tape recorder.

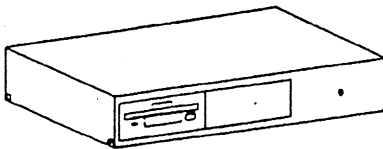


NOTE The MP has the same functional limitations as the GP100 / 100AP. (See previous page)



The data access panel (DP)

The data access panel is available for monitoring and changing the data of the EX100, e.g., timer / counter presets, register values, etc. It cannot be used to modify the program. The data access panel can also display user-defined ASCII diagnostic messages.



The disk drive unit (FD)

The disk drive unit is used with the GP110AP1 or GP110AP2 for storing and comparing programs.

- 3.5 inch floppy disk, 1 drive
- Record: GP (EX) → FD
- Load: GP (EX) ← FD
- Compare: GP (EX) ↔ FD
- Disk format

The program storage module*

The program storage module is an external memory dedicated to the EX100 program. By using this module, program saving from the EX100 to the module, and program loading from the module to the EX100 can be done without need of a programmer. Because the program storage module has an EEPROM, maintenance-free program storage and rapid saving / loading can be done.

* Under development

2. System configuration

- 2.7 Expansion cable**
Options The expansion cable is used to connect the basic and the expansion units. It is available in the following three lengths.

Part No.	Length
EX10-CAR3	0.3 m
EX10-CAR5	0.5 m
EX10-CAR7	0.7 m

Battery

The EX100's CPU module has an EEPROM for storing the user program permanently. Moreover, the data in the retentive registers and the clock-calendar are backed up by a built-in capacitor (7 days at 25°C).

The optional battery is used when it is necessary to back up the data in the retentive registers and the clock-calendar in excess of the capacitor's back-up period.

Applicable battery

Type: CR2032 (sold on the open market)
Voltage: 3 V
Capacity: 180 mAh
Recommended
replacement period: 1 year

Vacant slot cover

Optional covers are available for covering vacant slot on the rack.

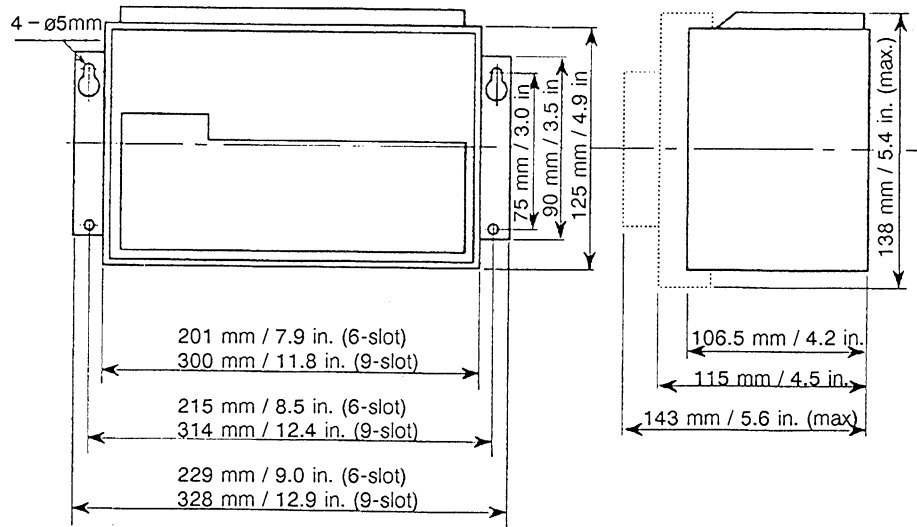
Part number: EX10-ABP1

3. Specifications

3.1 General specifications

Item	Specifications
Power supply voltage	(1) 100 to 120 Vac (+10 / -15%), 50 / 60 Hz (±5%)
	(2) 200 to 240 Vac (+10 / -15%), 50 / 60 Hz (±5%)
	(3) 24 Vdc (+20 / -15%)
Power consumption	50 VA or less (ac power supply) 22 W or less (dc power supply)
Retentive power fault	10 ms or less
Insulation resistance	10 MΩ or more (between power terminals and case)
Withstand voltage	1500 Vac, 1 min. (between power terminals and case)
Ambient temperature	Operating temperature: 0° to 55°C (32° to 131°F) Storage temperature: -20° to 75°C (-4° to 167°F)
Ambient humidity	20 to 90%RH, no condensation
Noise immunity	1000 Vp.p / 1 μs, NEMA ICS3-304
Vibration immunity	16.7 Hz -3 mmp.p (X, Y, and Z directions)
Shock immunity	10 g, 3 times (X, Y, and Z directions)
Weight	6-slot unit with I / Os: 2.8 kg (6.1 lb) 9-slot unit with I / Os: 4.0 kg (8.8 lb)

3.2 External dimensions



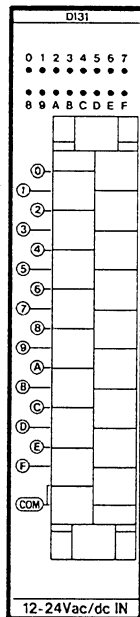
3. Specifications

3.3 Functional specifications

Item		Specifications
Control method		Stored program cyclic scanning system
I / O update		Batch I / O (immediate I / O instruction available)
Program language		Ladder diagram with function block
Memory	Program capacity	3K steps or 4K steps, switch selection. (For 3K steps, 1Kword of data is stored in the EEPROM)
	Memory type	EEPROM (Transferred to the RAM when the power is turned on.)
Instructions		15 basic types, 67 functional types
Execution speed		0.9 μ s / contact instruction
		110 μ s / 16 bit addition
No. of I / O points		Discrete I / O: 480 points Register I / O: 60 registers (1 register = 16 points) Devices and registers share the area.
Internal relays / registers	Data register	1536 registers (1 registers = 16 bits)
	Timer register	120 (0.1 s), 8 (0.01 s) Set value range: 0 to 32767
	Counter register	96, Set value range: 0 to 65535
	Auxiliary relay	960 points / 60 registers (area shared)
	Link relay	512 points / 32 registers (area shared)
	Special relay	Link status, timing clock, special functions, self diagnosis, and others (64 points total)
	Retentive memory	Data registers, timer registers, counter registers, and auxiliary relays can be designated for the retentive memory.
Clock-calendar function (Enhanced CPU only)		Year, month, day, day of the week, hours, minutes, seconds
Data link	Computer link (Enhanced CPU only)	RS-485, 16 stations maximum, 1 km maximum
	Link between PCs	TOSLINE-30, twisted pair / optical fiber cable
	Remote I / O	
Self diagnosis		Memory, I / O bus, program, I / O response, scan time, transmission, and watchdog timer checks
Self monitoring		Error table, scan time
RAM back-up		Built-in capacitor: 7 days / 25°C (77°F)
		Optional battery: 2 years / 25°C (77°F)
Programming tool		GP, HP, MP
Maintenance tool		DP

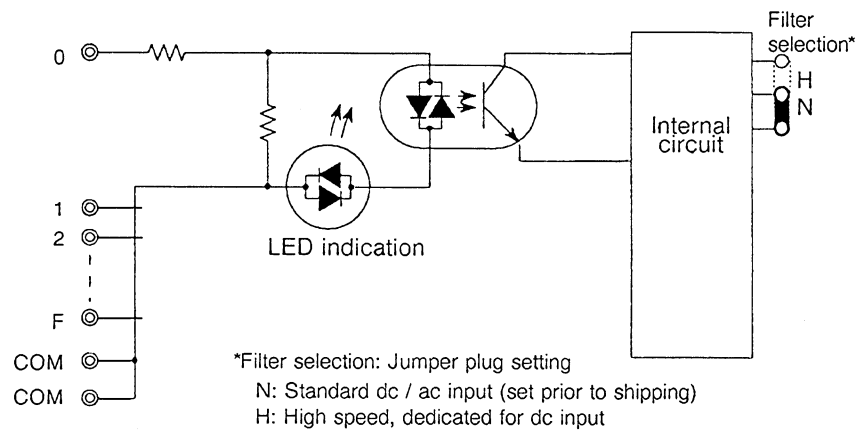
3.4 I / O specifications

16-point DC / AC input

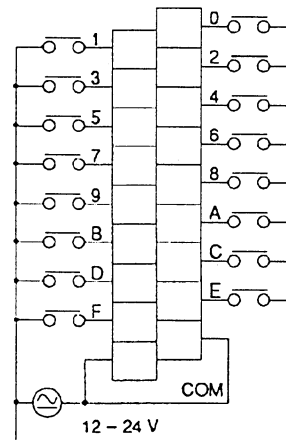


Item		DI31 (EX10-MDI31)
Input voltage range		12 to 24 V, + 10 / - 15%, dc / ac (50-60 Hz)
Minimum ON voltage		9.6 V
Maximum OFF voltage		3.6 V (0.7 mA or less)
Input current		8 mA (24 V) (typ.)
No. of input points		16 points (16 points / common)
On delay	Mode N	10 ms or less (dc) / 20 ms or less (ac)
	Mode H	1.5 ms or less (dc)
OFF delay	Mode N	10 ms or less (dc) / 15 ms or less (ac)
	Mode H	1.5 ms or less (dc)
Withstand voltage		1500 Vac, 1 minute
Current consumption		15 mA (5 Vdc) or less

Circuit diagram

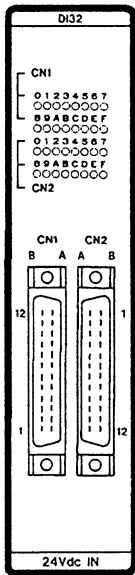


Terminal connection



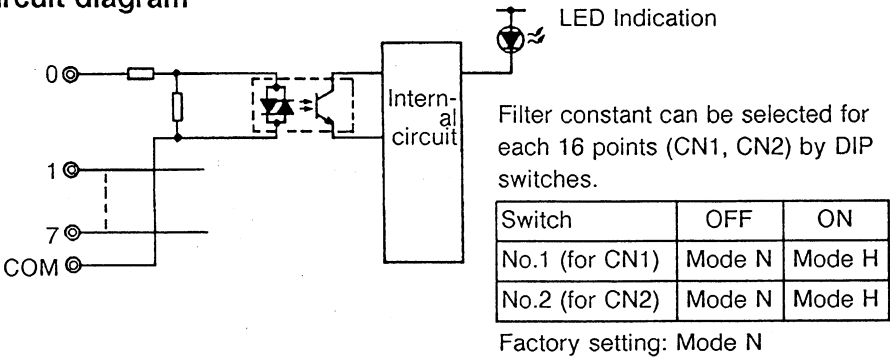
3. Specifications

32-point DC input

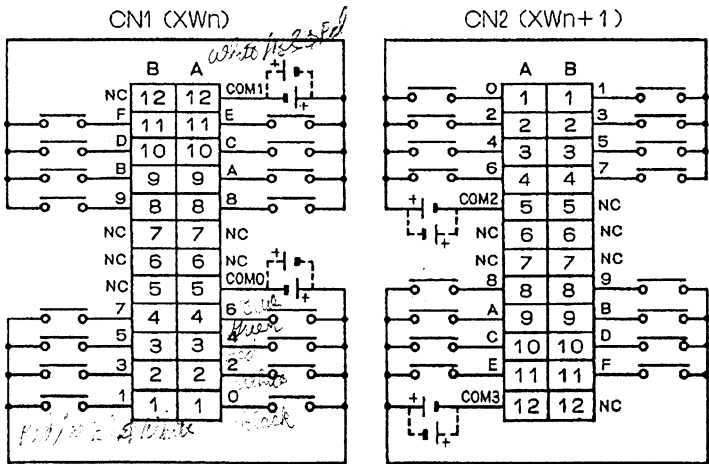


Item		DI32 (EX10-MDI32)
Input voltage		24 Vdc, +10 / -15%
Minimum ON voltage		18.0 V
Maximum OFF voltage		6.0 V
Input current		5 mA (24 Vdc) (typ.)
No. of input points		32 points
ON delay	Mode N	10 ms or less
	Mode H	1.5 ms or less
OFF delay	Mode N	10 ms or less
	Mode H	1.5 ms or less
External connection		2×24-pin connector
Common System	No. of commons	4
	Input points per common	8 points / common
	Common polarity	Non
Withstand voltage		1500 Vac / 1 minute
Current consumption		80 mA (5 Vdc) or less

Circuit diagram



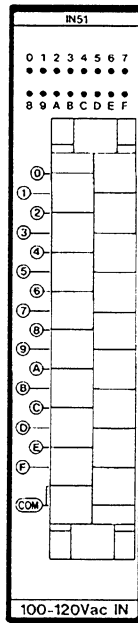
Terminal connection



Cable side connectors (soldering type) are attached as standard.

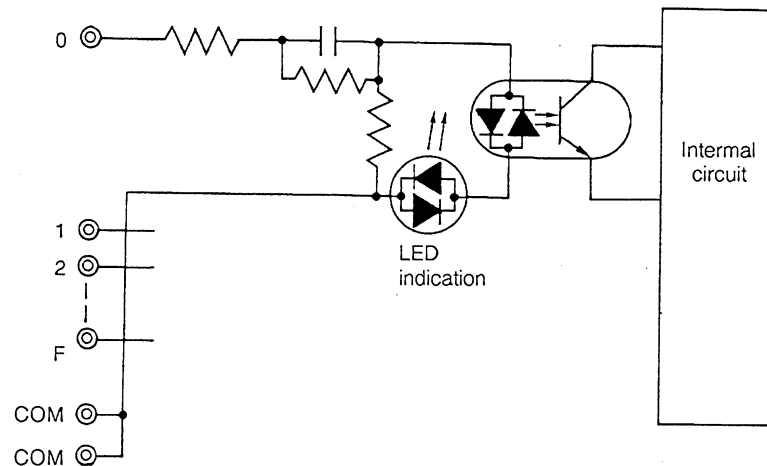
3. Specifications

16-point AC input

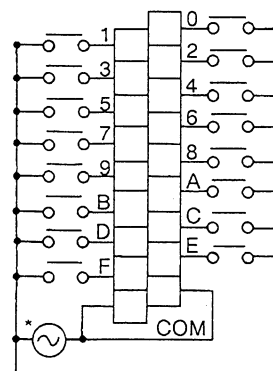


Item	IN51 (EX10-MIN51)	IN61 (EX10-MIN61)
Input voltage range (sine wave)	100 to 120 Vac, + 10 / - 15% (50 to 60 Hz)	200 to 240 Vac, + 10 / - 15% (50 to 60 Hz)
Minimum ON voltage	80 Vac	160 Vac
Maximum OFF voltage	30 Vac (2 mA or less)	60 Vac (2 mA or less)
Input current	7 mA (100 V, 50 Hz) (typ.)	6 mA (200 V / 50 Hz) (typ.)
No. of input points	16 points (16 points / common)	16 points (16 points / common)
ON delay	20 ms or less	20 ms or less
OFF delay	15 ms or less	15 ms or less
Withstand voltage	1500 Vac, 1 minute	1500 Vac, 1 minute
Current consumption	15 mA (5 Vdc) or less	15 mA (5 Vdc) or less

Circuit diagram



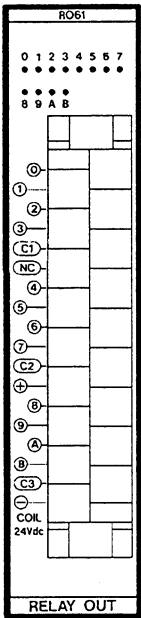
Terminal connection



* IN51: 100-120V (50 / 60Hz)
IN61: 200-240V (50 / 60Hz)

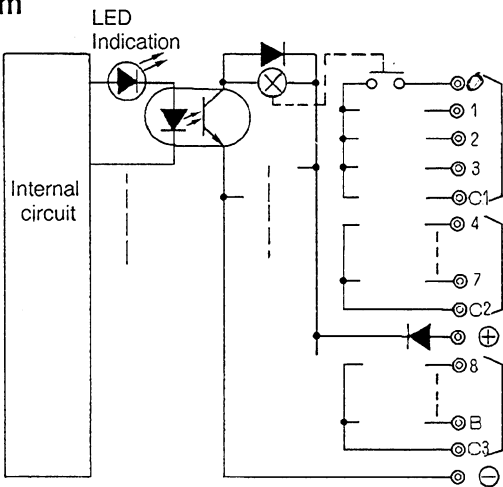
3. Specifications

12-point relay output

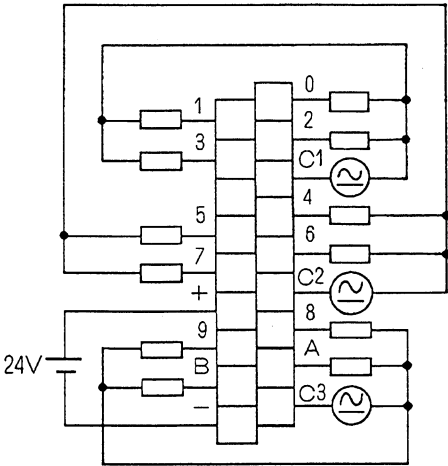


Item	RO61 (EX10-MRO61)
Load power	24 Vdc, + 20% (max.) / 240 Vac, + 10% (max.)
Maximum load	2 A / pt (resistive load), 1 A / pt (inductive load) 4 A / common
Minimum load	50 mW (5 V or more)
No. of output points	12 points (4 points / common)
ON delay	10 ms or less
OFF delay	15 ms or less
Leakage current when OFF	0 mA
Withstand voltage	1500 Vac, 1 minute
Current consumption	50 mA (5 Vdc) or less
External power required	24 Vdc \pm 10% - 140 mA (all pts ON), 10 mA / pt

Circuit diagram



Terminal connection

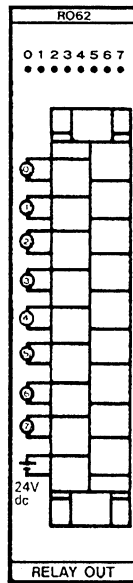


NOTE

Expected relay life: 100,000 operations (electrical)
20 million operations (mechanical)

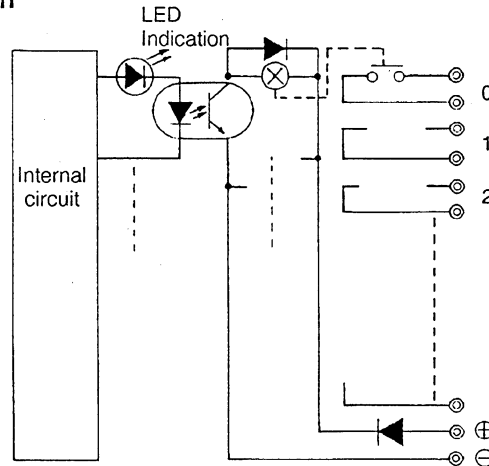
3. Specifications

8-point isolated relay output

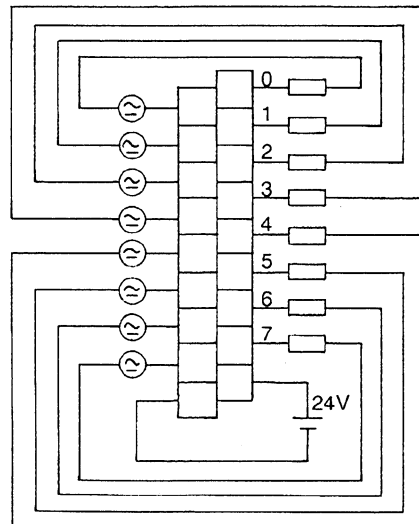


Item	RO62 (EX10-MRO62)
Load power	24 Vdc, +20% (max.) / 240 Vac, +10% (max.)
Maximum load	2 A / pt (resistive load), 1 A / pt (inductive load)
Minimum load	50 mW (5 V or more)
No. of output points	8 points (isolated)
ON delay	10 ms or less
OFF delay	15 ms or less
Leakage current when OFF	0 mA
Withstand voltage	1500 Vac, 1 minute
Current consumption	40 mA (5 Vdc) or less
External power required	24 Vdc \pm 10% - 100 mA (all pts ON), 10 mA / pt

Circuit diagram



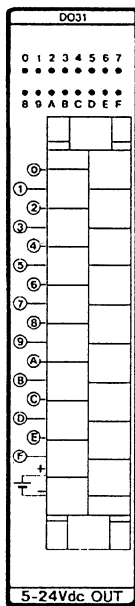
Terminal connection



Expected relay life: 100,000 operations (electrical)
20 million operations (mechanical)

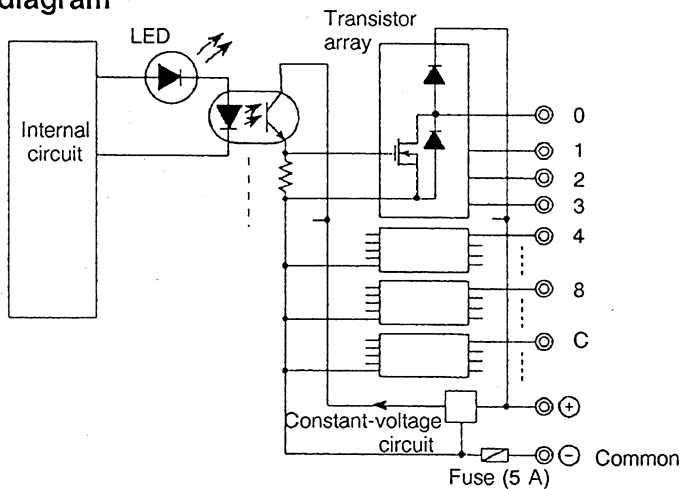
3. Specifications

16-point transistor output

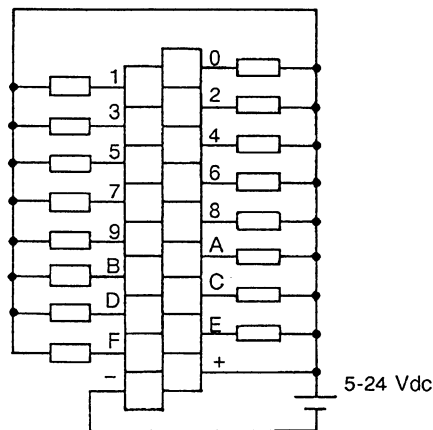


Item	DO31 (EX10-MDO31)
Load power	5 to 24 Vdc, +10 / -5%
ON output current	1 A / point (load power 7 V or more)
	0.3 A / point (load power 7 V or less)
	1.2 A / 4 points (transistor array)
ON resistance	1.5 Ω or less
No. of output points	16 points (16 points / common, Θ common)
ON delay	1 ms or less
OFF delay	1 ms or less
Leakage current when OFF	0.1 mA or less
Withstand voltage	1500 Vac, 1 minute
Current consumption	60 mA (5 Vdc) or less

Circuit diagram

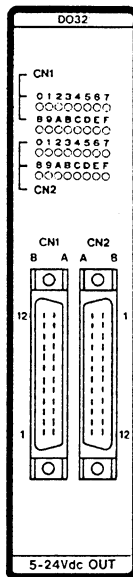


Terminal connection



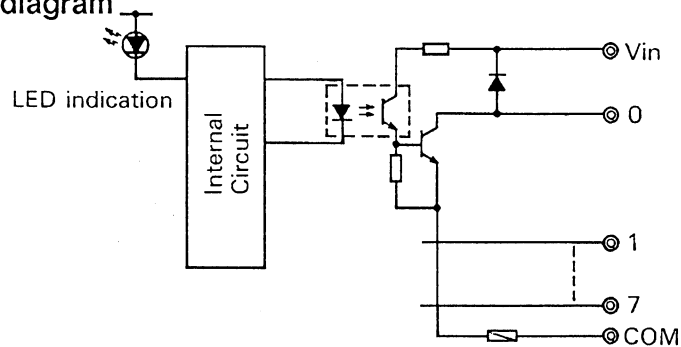
3. Specifications

32-point transistor output

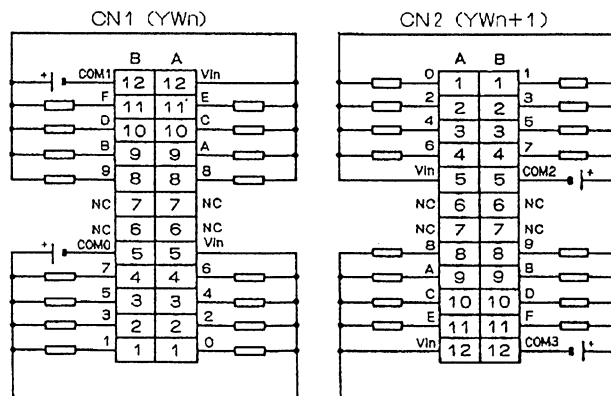


Item		DO32 (EX10-MDO32)
Load power		5 to 24 Vdc, +10 / -5%
ON output current		100 mA / point (load power 24 V)
		20 mA / point (load power 5 V)
		800 mA / common
ON saturated voltage		0.4 V or less
No. of output points		32 points
Output method		current sinking
ON delay		1 ms or less
OFF delay		2 ms or less (typ.)
Leakage current when OFF		0.1 mA or less
External connection		2×24-pin connector
Common system	No. of commons	4
	Output pts per common	8 points per common
	Common polarity	⊖ common
Withstand voltage		1500 Vdc, 1 minute
Built-in fuses		4×2 A / common
Current consumption		250 mA (5 Vdc) or less

Circuit diagram



Terminal connection

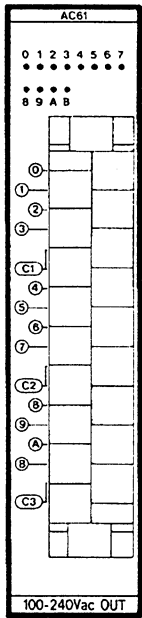


NOTE

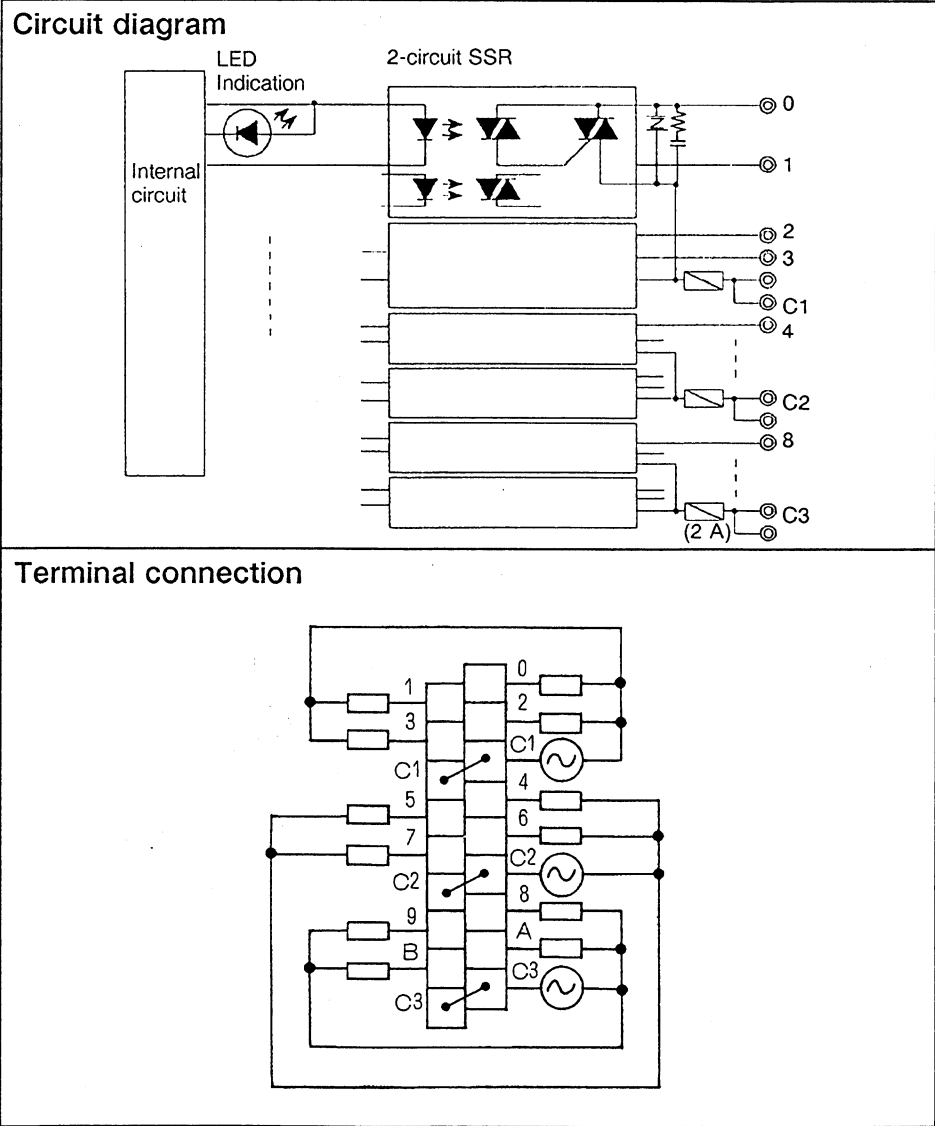
Cable side connectors (soldering type) are attached as standard.

3. Specifications

12-point triac output

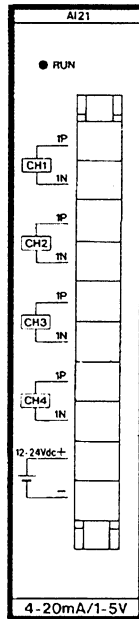


Item	AC61 (EX10-MAC61)
Load power	100 to 240 Vac, + 10 / - 15% (50 to 60 Hz sine wave)
ON output current	0.5 A / point, 0.6 A / SSR
ON saturated voltage	1.5 V or less (0.3 A load)
No. of output points	12 points (4 points / common)
ON delay	1 ms or less
OFF delay	1/2 cycle of load power + 1 ms or less
Leakage current when OFF	1.2 mA (100 Vac) or less, 3 mA (240 Vac) or less
Withstand voltage	1500 Vac, 1 minute
Current consumption	300 mA (5 Vdc) or less (all points ON), 20 mA / pt



3. Specifications

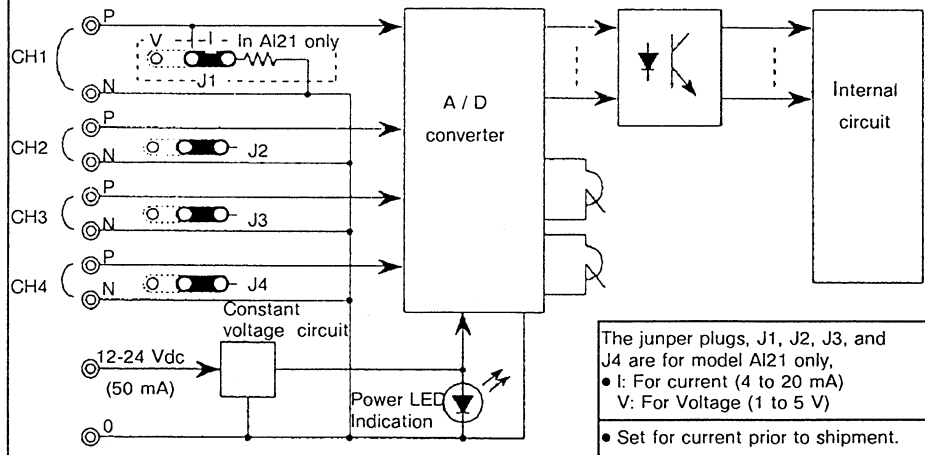
4-channel analog input (8-bit)



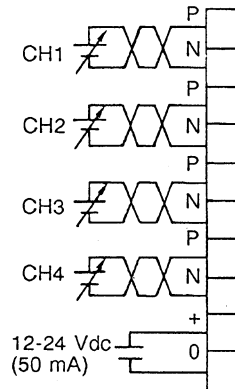
The input range for Model AI21 is set to 4 to 20 mA prior to shipping. Set the jumper plugs to the V position when inputting 1 V to 5 V.

Item	AI21 (EX10-MAI21)	AI31 (EX10-MAI31)
Input range	1 to 5 V or 4 to 20 mA	0 to 10 V
Input impedance	1 to 5 V: 500 k Ω or more 4 to 20 mA: 250 Ω	500 k Ω or more
No. of input points	4 channels, N common	4 channels, N common
Resolution	1 to 5 V: 0 to 250 4 to 20 mA: 0 to 250	0 to 10 V: 0 to 250
Overall accuracy	$\pm 1\%$ (FS)	$\pm 1\%$ (FS)
Conversion cycle	Approx. 1 ms	Approx. 1 ms
Wire breakage detection	Yes, for 4 to 20 mA	—
External power failure detection	Yes	Yes
Withstand voltage	1500 Vac, 1 minute	1500 Vac, 1 minute
Current consumption	50 mA (5 Vdc) or less	50 mA (5 Vdc) or less
External power required	12 to 24 Vdc, $\pm 10\%$ – 50 mA	12 to 24 Vdc, $\pm 10\%$ – 50 mA

Circuit diagram



Terminal connection

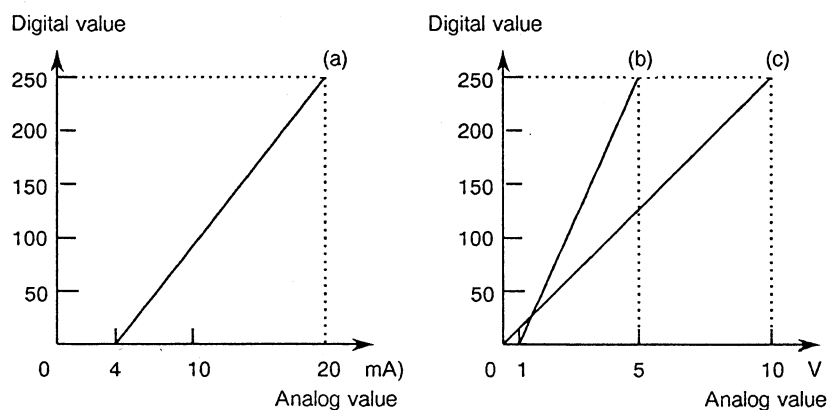


Use shielded twisted-pair cables for analog signals. Be sure to correctly ground the shield.

3. Specifications

4-channel analog input (8-bit) (cont'd)

A / D conversion



(a) 4 to 20 mA: $D = 15.625 A - 62.5$

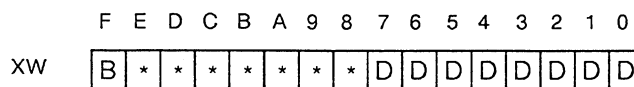
(b) 1 to 5 V: $D = 62.5 A - 62.5$

(c) 0 to 10 V: $D = 25 A$

D: Digital value

A: Analog value

Data format



D: Data bit (8 bits)

0 to 250 (H00 to HFA)

B: Bit for detecting trouble in external wiring

0: Normal

1: Abnormal If all the data bits are 0, the current input cable is open (4 to 20 mA only).
If all the data bits are 1, the external power supply is off.

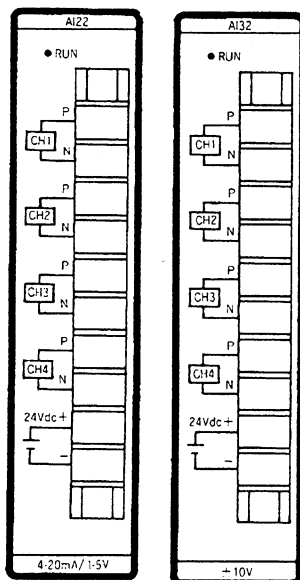
*: Always 0

Register assignment

	F	0
XWn	CH1	
XWn+1	CH2	
XWn+2	CH3	
XWn+3	CH4	

3. Specifications

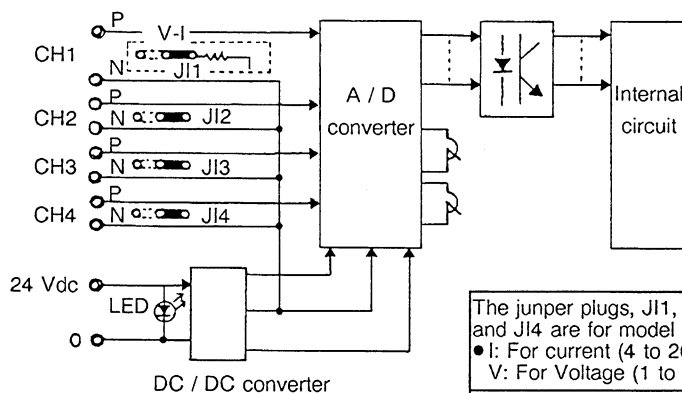
4-channel analog input (12-bit)



The input range for Model AI22 is set to 4 to 20 mA prior to shipping. Set the jumper plugs to the V position when inputting 1V to 5 V

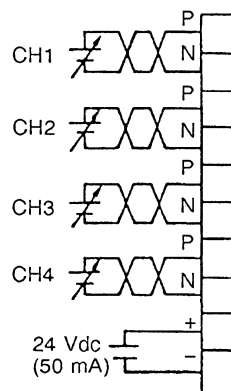
Item	AI22 (EX10-MAI22)	AI32 (EX10-MAI32)
Input range	1 to 5 V or 4 to 20 mA	- 10 to + 10 V
Input impedance	1 to 5 V: 1 M Ω or more, 4 to 20 mA: 250 Ω	1 M Ω or more
No. of input points	4 channels, N common	4 channels, N common
Resolution	1 to 5 V: 0 to 4000 4 to 20 mA: 0 to 4000	- 10 to + 10 V: - 2000 to 2000
Overall accuracy	$\pm 0.5\%$ FS / 25°C $\pm 1\%$ FS / 0 to 55°C	$\pm 0.5\%$ FS / 25°C $\pm 1\%$ FS / 0 to 55°C
Conversion cycle	Approx. 9.6 ms / 4 channels	Approx. 9.6 ms / 4 channels
Wire breakage detection	Yes, for 4 to 20 mA	—
External power failure detection	Yes	No
Withstand voltage	1500 Vac, 1 minute	1500 Vac, 1 minute
Current consumption	50 mA (5 Vdc) or less	50 mA (5 Vdc) or less
External power required	24 Vdc, $\pm 10\%$ - 50 mA	24 Vdc, $\pm 10\%$ - 50 mA

Circuit diagram



The jumper plugs, JI1, JI2, JI3, and JI4 are for model AI22 only.
• I: For current (4 to 20 mA)
• V: For Voltage (1 to 5 V)
• Set for current prior to shipment.

Terminal connection

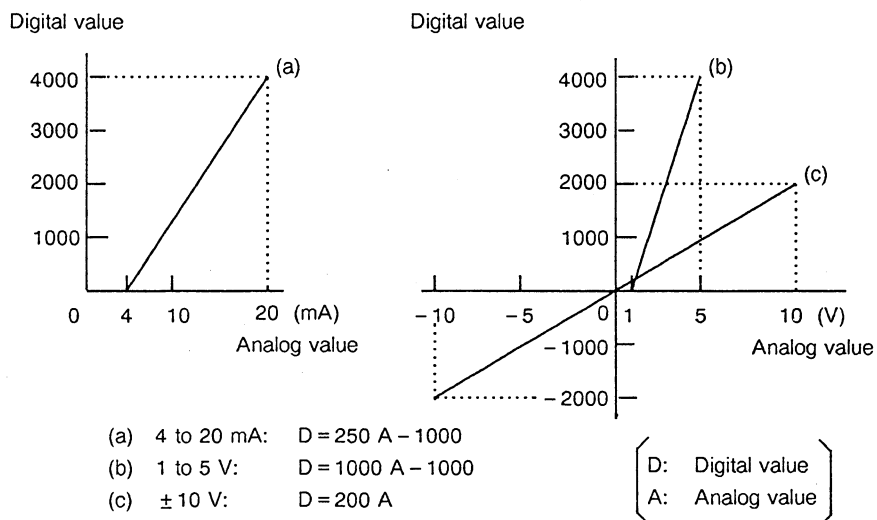


Use shielded twisted-pair cables for analog signals. Be sure to correctly ground the shield.

3. Specifications

4-channel analog input (12-bit) (cont'd)

A / D conversion



Data format

• 4 to 20 mA / 1 to 5 V

	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0
XW	B	*	*	*	D	D	D	D	D	D	D	D	D	D	D	D

D: Data bit (12 bits)

0 to 4000 (H0000 to H0FA0)

B: Bit for detecting trouble in external wiring

0: Normal

1: Abnormal (current input wire is open or external power supply is off)

∗: Always 0

• ± 10 V

	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0
XW	S	S	S	S	S	D	D	D	D	D	D	D	D	D	D	D

S: Sign bit

0: Positive

1: Negative

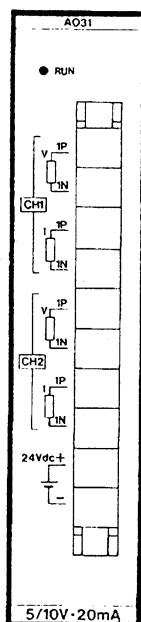
D: Data bit (11 bits)

-2000 to 2000 (HF830 to H07D0)

Two's complement if negative

3. Specifications

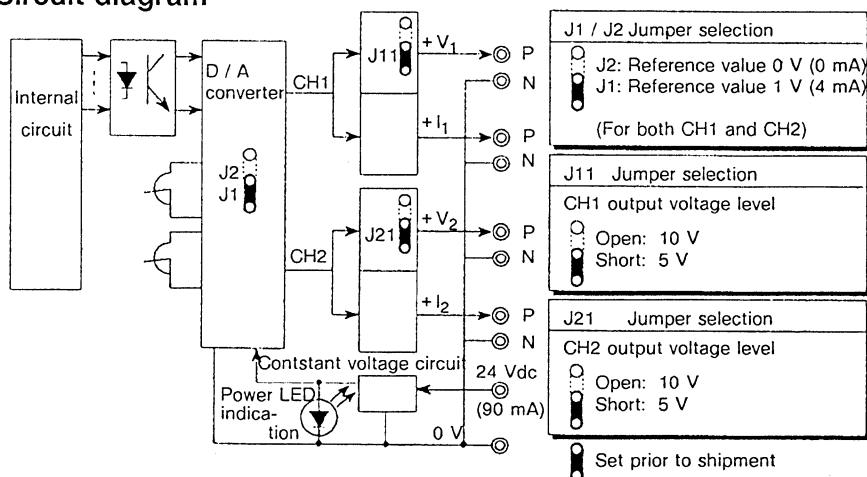
2-channel analog output (8-bit)



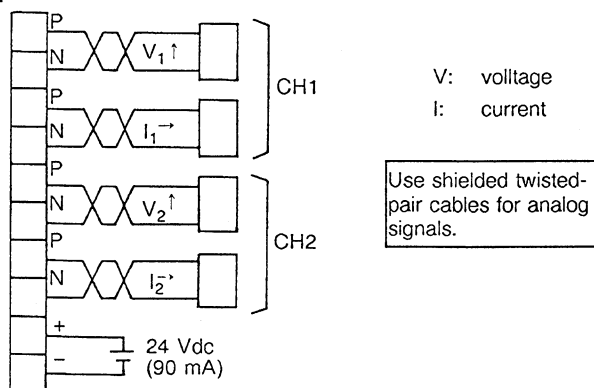
1 to 5 V / 4 to 20 mA is set at pre-shipment inspection. Refer to the circuit configuration for other settings

Item	AO31 (EX10-MAO31)
Output range	0 to 10 V, 1 to 5 V, 4 to 20 mA
Load impedance	5 V full-scale terminal: 5 kΩ or more
	10 V full-scale terminal: 10 kΩ or more
	20 mA full-scale terminal: 600 kΩ or less
No. of output channels	2, voltage and current paired (N common)
Resolution	0 to 250 (full scale)
Overall accuracy	± 1% (FS)
Conversion cycle	Approx. 1 ms
External power failure detection	No
Withstand voltage	1500 Vac, 1 minute
Current consumption	70 mA (5 Vdc) or less
External power required	24 Vdc, ± 10% - 90 mA

Circuit diagram



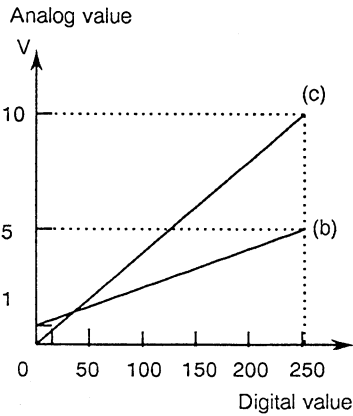
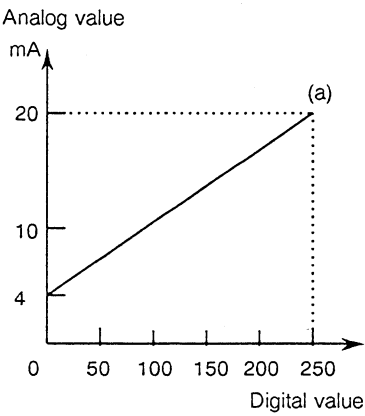
Terminal connection



3. Specifications

2-channel analog
output (8-bit)
(cont'd)

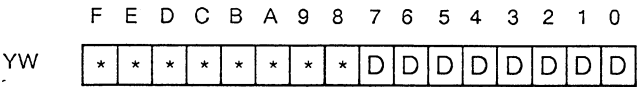
D / A conversion



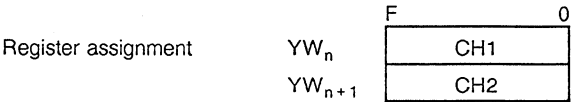
- (a) 4 to 20 mA: $A = 0.064 D + 4$
(b) 1 to 5 V: $A = 0.016 D + 1$
(c) 0 to 10 V: $A = 0.04 D$

A: Analog value
D: Digital value

Data format



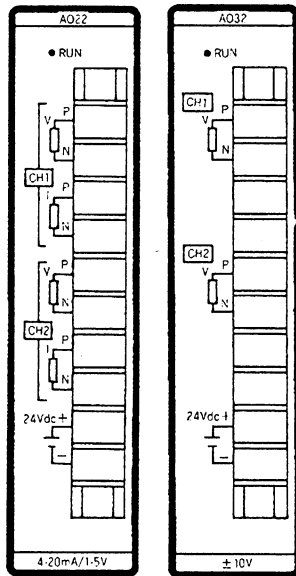
- D : Data bit (8 bits)
0 to 250 (H00 to HFA)
* : Invalid (does not affect D / A conversion.)



If the immediate output instruction (FUN097) is used, two registers (both channels) should be specified as immediate output registers.

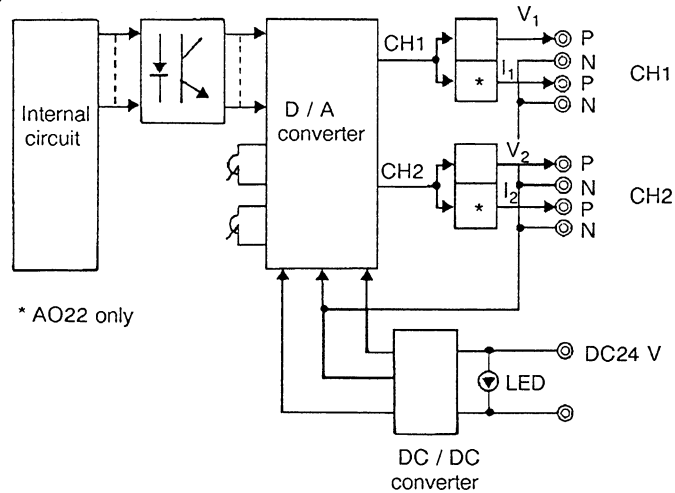
3. Specifications

2-channel analog output (12-bit)

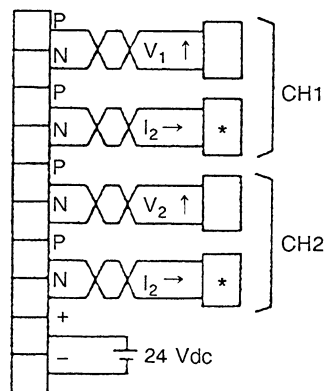


Item	AO22 (EX10-MAO22)	AO32 (EX10-MAO32)
Output range	1 to 5 V or 4 to 20 mA	-10 to 10 V
Load impedance	1 to 5 V: 5 k Ω or more 4 to 20 mA: 600 Ω or less	5 k Ω or more
No. of output channels	2 channels, (N common) (voltage and current paired)	2 channels (N common)
Resolution	1 to 5 V: 0 to 4000 4 to 20 mA: 0 to 4000	-10 to +10 V: -2000 to 2000
Overall accuracy	$\pm 0.5\%$ FS / 25°C $\pm 1\%$ FS / 0 to 55°C	$\pm 0.5\%$ FS / 25°C $\pm 1\%$ FS / 0 to 55°C
Conversion cycle	Approx. 1 ms	Approx. 1 ms
External power failure detection	No	No
Withstand voltage	1500 Vac, 1 minute	1500 Vac, 1 minute
Current consumption	170 mA (5 Vdc) or less	170 mA (5 Vdc) or less
External power required	24 Vdc, $\pm 10\%$ - 90 mA	24 Vdc, $\pm 10\%$ - 90 mA

Circuit diagram



Terminal connection



V: voltage
I: current

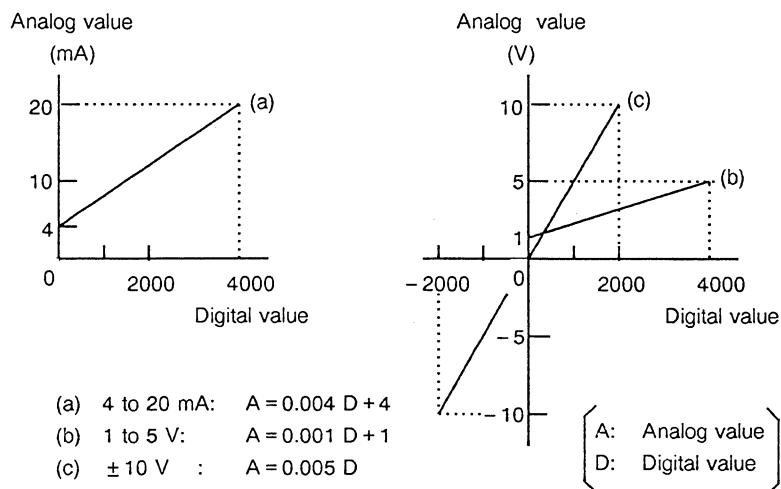
* AO22 only

Use shielded twisted-pair cables for analog signals.

3. Specifications

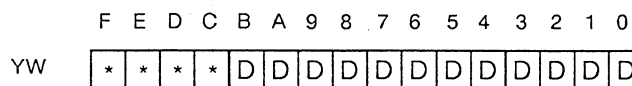
2-channel analog output (12-bit) (cont'd)

D / A conversion



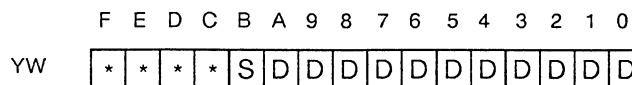
Data format

- 4 to 20 mA / 1 to 5 V



- D: Data bit (12 bits)
 0 to 4000 (H0000 to H0FA0)
 *: Invalid (does not affect D / A conversion)

- ± 10 V



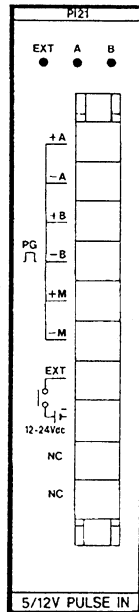
- S: Sign bit
 0: Positive
 1: Negative
 D: Data bit (11 bits)
 -2000 to 2000 (HF830 to H07D0)
 Two's complement if negative



If the immediate output instruction (FUN097) is used, two registers (both channels) should be specified as immediate output registers.

3. Specifications

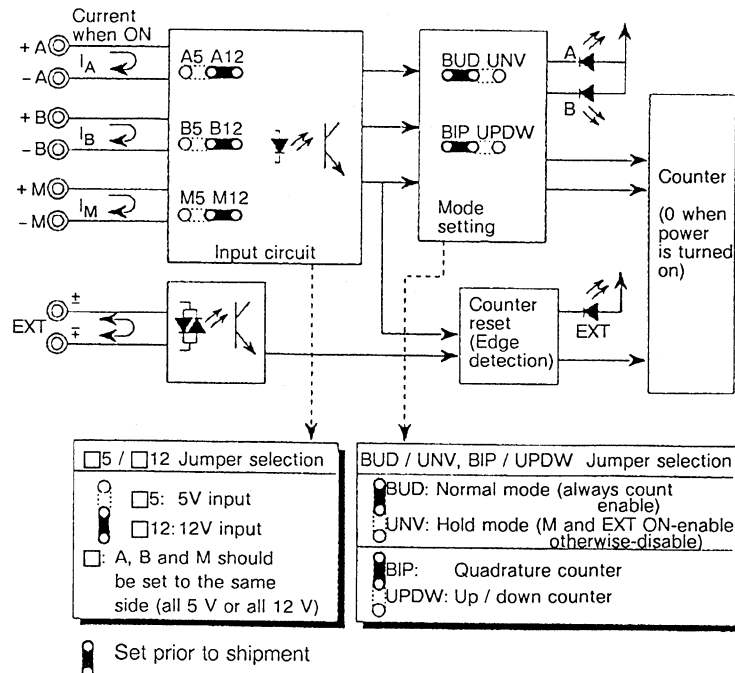
1-channel pulse input



Item		PI 21 (EX10-MPI 21)
Input voltage	A, B, M	12V, +10 / -20% (12V setting) / 5V, +10 / -20% (5V setting)
	EXT	12 to 24 Vdc, +10 / -15%
Minimum ON voltage	A, B, M	9 V (12 V setting) / 3.5 V (5 V setting)
	EXT	9.6 V
Maximum OFF voltage	A, B, M	2 V (12 V setting) / 1 V (5 V setting)
	EXT	3.6 V
Input current	A, B, M	12 V - 7.5 mA (12 V setting) / 5 V - 10 mA (5 V setting)
	EXT	24 V - 10 mA, 12 V - 5 mA
Input channel		1 channel (phase A, B, M, and EXT)
Count speed		100 kpps (max.) (pulse width 4 μ s or more)
Counter		24-bit binary
Pulse mode	Quadrature	Phase A, B (90 degree phase shift), up / down
	Up / down	Phase A: count up / phase B: count down
Counter mode	Normal	Always count enable
	Hold	Both M and EXT ON: count enable Either M or EXT OFF: count value held
Counter reset		Count value is reset to zero at the moment when both M and EXT are turned ON
EXT ON / OFF delay		5 ms or less <i>MORE</i>
Withstand voltage		1500 Vac, 1 minute
Current consumption		80 mA (5 Vdc) or less

The input voltage of A, B, and M are set to 12 V, and the counter mode is set to quadrature prior to shipping.

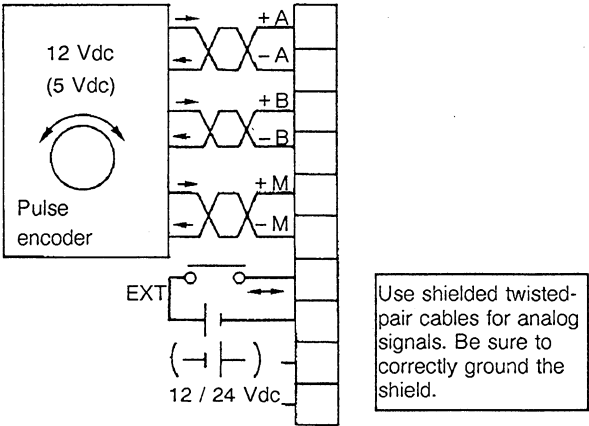
Circuit diagram



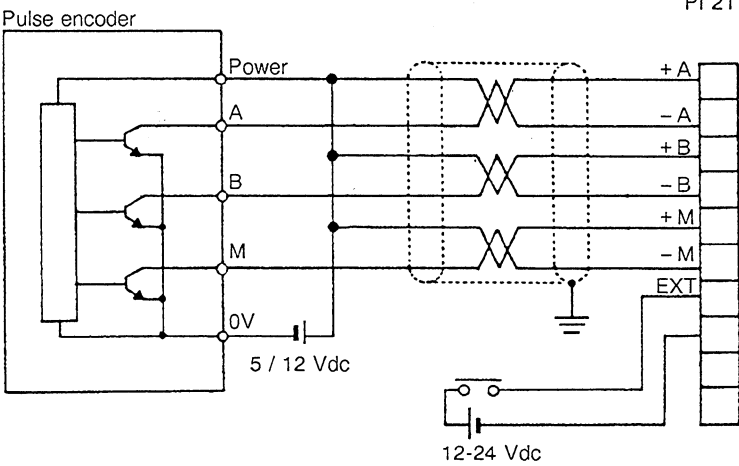
3. Specifications

1-channel pulse input
(cont'd)

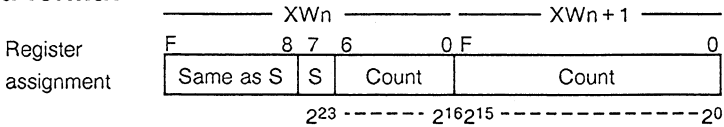
Terminal connection



Wiring example

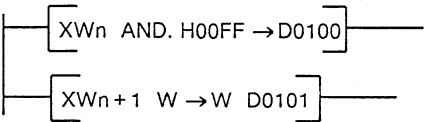


Data format



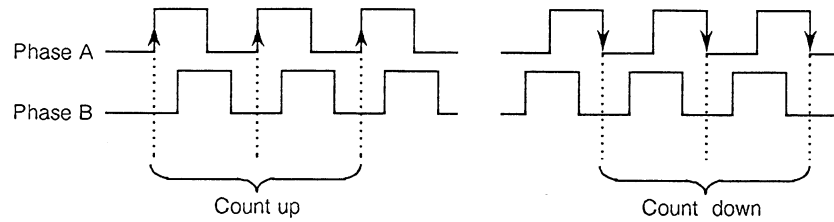
24 bits counter — 0 to 16,777,215 (H00FFFFFF)

NOTE Bits 8 to F of XWn are same as bit 7 of XWn.
Normally, bits 8 to F should be masked by user program as follows.
(e.g. storing the count value into D0100 · D0101)

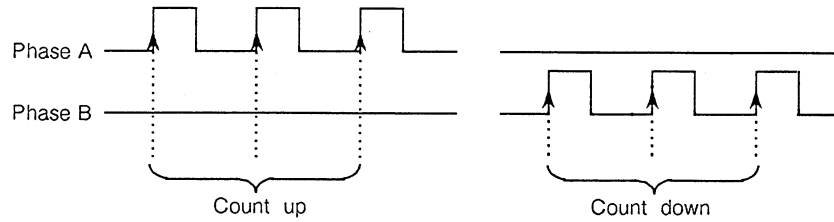


1-channel pulse input (cont'd)

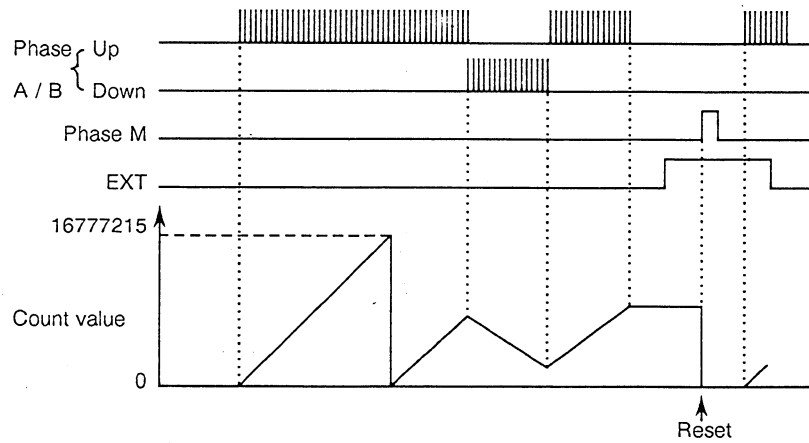
Pulse mode < Quadrature >



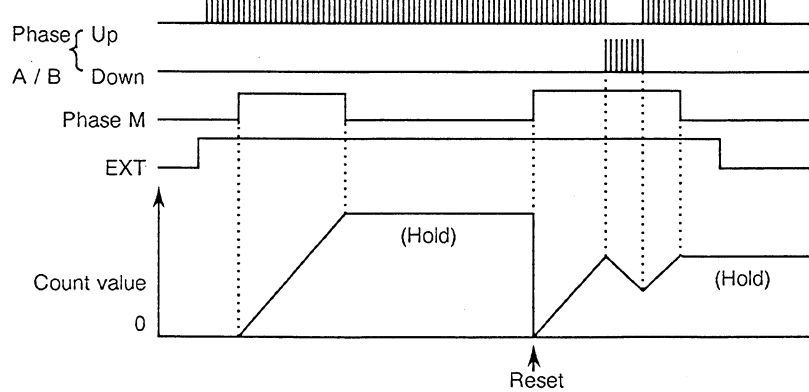
< Up / down >



Counter mode < Normal >



< Hold >

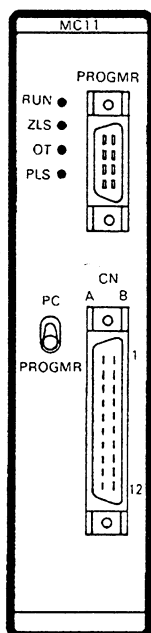


NOTE

If the immediate input instruction (FUN096) is used, two registers should be specified as immediate input registers.

3. Specifications

Motion control



Item		MC11 (EX10-MMC11)	
No. of control axes		1 axis	
Control units		Pulse / inch / mm etc.	
Control range		± 999,999 (units)	
Point data capacity		64 points	
Absolute max. speed		200 kpps	
Operation speed		Origin return speed, max. speed, min. speed	
Acceleration / deceleration system		Automatic trapezoidal / triangular system	
Acceleration / deceleration rate		0 to 26 s	
Backlash compensation		0 to 1000 pulses	
Zero position offset		± 999,999 units	
Dwell time		0 to 99 s	
I / O occupancy points		X + Y 4 W (64 bits)	
Parameter storage		EEPROM	
Input	Input voltage	12 / 24 Vdc (Z-phase: 5 / 12 / 24 Vdc)	
	Input current	10 mA (24 V)	
	ON / OFF voltage	9.6 V / 3.2 V	
	ON / OFF delay	5 ms (Z-phase: 1 ms)	
Output	Pulse output	Mode (switch setting)	(1) CW / CCW, error counter clear (2) Pulse / direction, error counter clear
		Output method	Open collector (5 – 24 Vdc, 50 mA)
		ON / OFF delay	2 μs
	RUN output	Output method	Open collector (5 – 24 Vdc, 50 mA)
		Operation	ON during normal operation
Current consumption	Internal	200 mA – 5 Vdc 400 mA – 5 Vdc (when HP is connected)	
	External	100 mA – 12 / 24 Vdc	

Connector pin arrangement

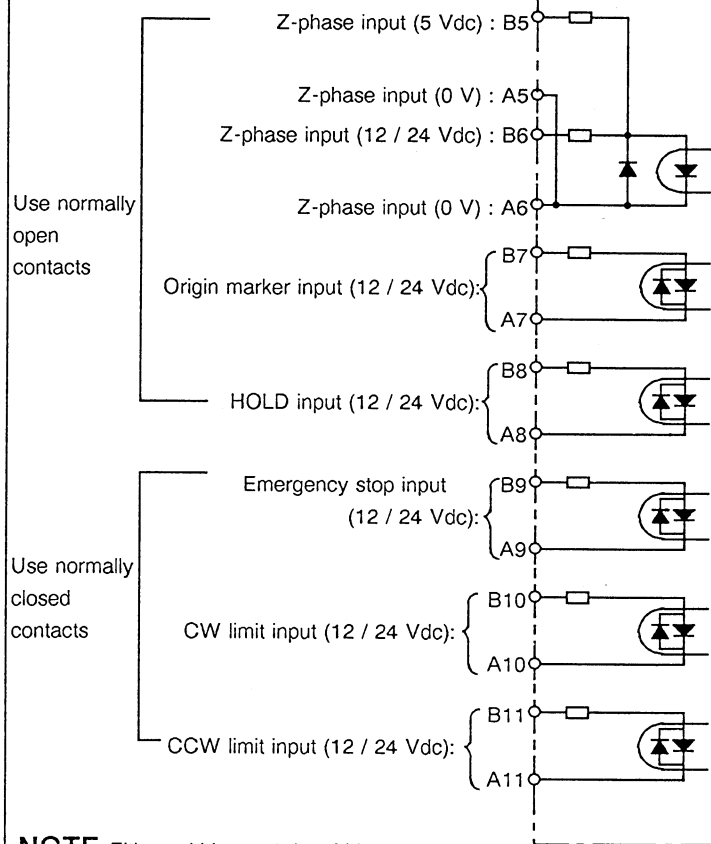
	A	B	
RUN output (0 V)	1	1	RUN output (5 – 24 Vdc)
CW / Pulse output (0 V)	2	2	CW / Pulse output (5 – 24 Vdc)
CCW / Direction output (0 V)	3	3	CCW / Direction output (5 – 24 Vdc)
Error counter clear output (0 V)	4	4	Error counter clear output (5 – 24 Vdc)
Z-phase input (0 V)	5	5	Z-phase input (5 Vdc)
Z-phase input (0 V)	6	6	Z-phase input (12 / 24 Vdc)
Origin marker input (0 V)	7	7	Origin marker input (12 / 24 Vdc)
Hold input	8	8	Hold input (12 / 24 Vdc)
Emergency stop input	9	9	Emergency stop input (12 / 24 Vdc)
CW limit input	10	10	CW limit input (12 / 24 Vdc)
CCW limit input	11	11	CCW limit input (12 / 24 Vdc)
External power (0 V)	12	12	External power (12 / 24 Vdc)



A cable side connector (soldering type) is attached as standard.

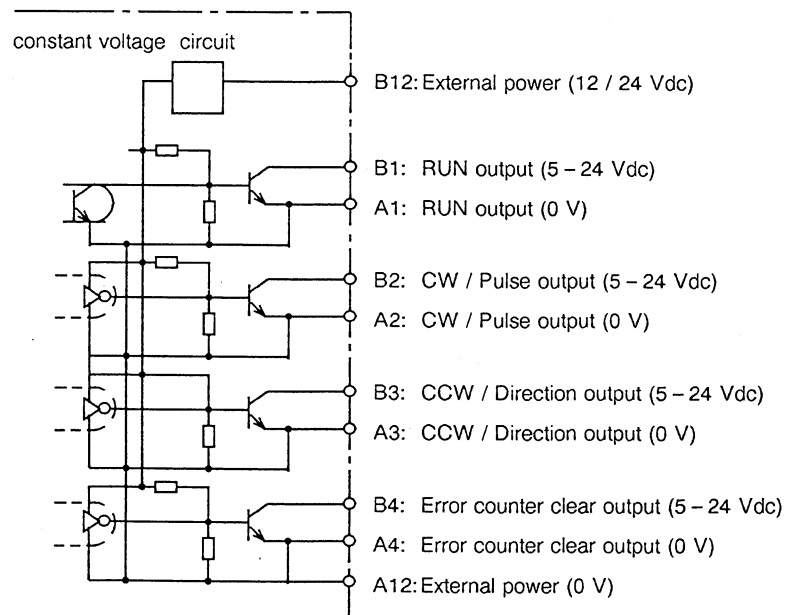
Motion control (cont'd)

Input circuit



NOTE Either 5 Vdc or 12 / 24 Vdc can be used alternatively for the Z-phase input.

Output circuit



4. Installation and Wiring

4.1 Operating environment

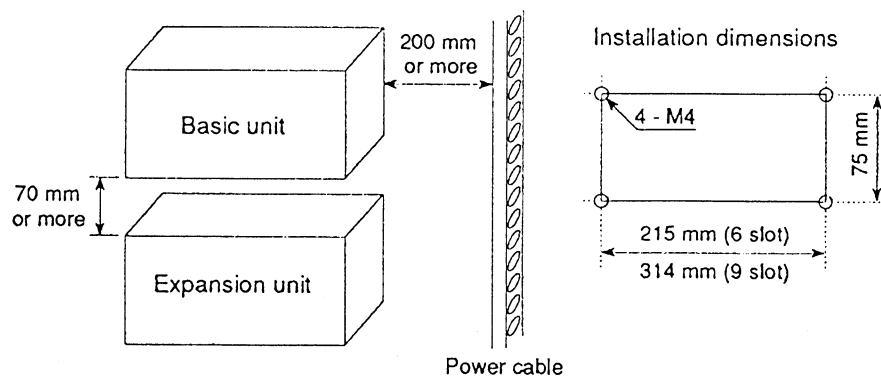
Do not install the EX100 in the following locations:

- Where the ambient temperature drops below 0°C (32°F) or exceeds 55°C (131°F)
- Where the relative humidity drops below 20% or exceeds 90%
- Where there is condensation due to sudden temperature changes
- In locations subject to vibrations that exceed tolerance
- In locations subject to shocks that exceed tolerance
- Where there are corrosive or inflammable gases
- In locations subject to dust, machining debris or other particles
- In locations exposed to direct sunlight

Observe the following precautions when installing enclosures in which the EX100 will be mounted:

- Provide the maximum possible distance between high-voltage or high-power panels. This distance must be at least 200 mm (8 in).
- If installing the enclosures in the vicinity of high-frequency equipment, be sure to correctly ground the enclosures.
- When sharing the channel base with other panels, check for leakage current from the other panels or equipment.

4.2 Installing the unit



4. Installation and Wiring

Installation precautions:

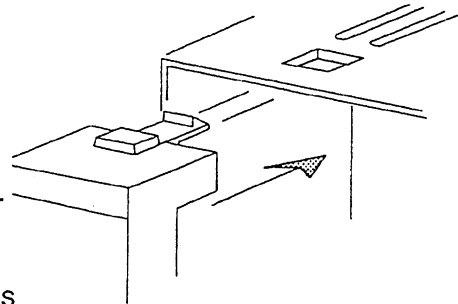
- Because the EX100 is not dust-proof, install it in a dust-proof enclosure.
- Do not install the unit directly above equipment that generates a large amount of heat, such as a heater, transformer, or large-capacity resistor.
- Do not install the unit within 200 mm (8 in) of a high-voltage or power cable.
- Allow at least 70 mm (2.8 in) on all sides of the unit for ventilation.
- For safety during maintenance and operation, install the unit as far as is possible from high-voltage or power equipment. Alternatively, keep the unit separate using a steel plate or similar separator.
- If installing the unit near high-voltage or power equipment, the grounding requires special attention. See Section 4.5.
- Be sure to install the unit vertically. Be sure to install the power supply module in the left slot of the EX100.

4.3 Mounting the modules

The power supply module must be mounted in the slot at the extreme left of the rack. Install the CPU module in the slot next to the power supply module.

The modules are installed as follows:

1. First, install the power supply module at the left end. Then install other modules from the left slot, taking care to securely insert them into the slots of the rack.
2. Insert the modules fully until the front panels of the modules are locked in the rack.

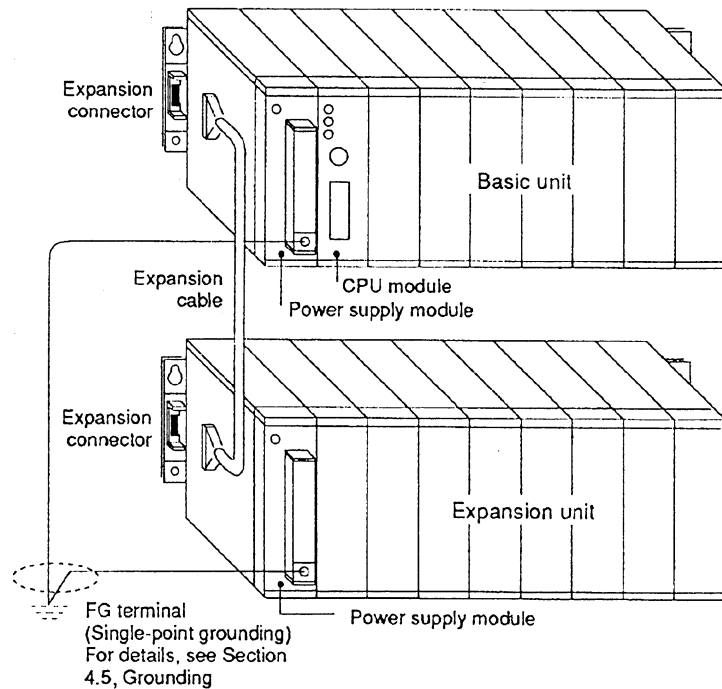


- CAUTION** • For safety, be sure to turn off power to the EX100 before installing or removing a module.
- After installing the modules, secure the unit to keep the modules vertical. Also, be sure that the modules remain vertical if transporting the unit.

4. Installation and Wiring

4.4 Connecting the expansion unit

One expansion unit, consisting of either six or nine slots, can be connected to the basic unit. To connect an expansion unit, it is necessary to use rack EX10*UBB1 or EX10*UBB2, equipped with an expansion connector, for both the basic unit and the expansion unit.



NOTE

- Separate expansion cable from all other cables.
- The expansion cable is available in three lengths, depending on the configuration, 0.3 m, 0.5 m, 0.7 m.

4.5 Grounding

The optimum method for grounding electronic equipment is to ground it separately from other high-power systems, and to ground more than two units of electronic equipment with a single-point ground.

The EX100 has a noise-proof design, sufficient to withstand industrial operating conditions. Although the EX100 itself can resist noise (signal interference) without being grounded, for safety and reliability, grounding is recommended.

4. Installation and Wiring

4.5.1 Check points for grounding

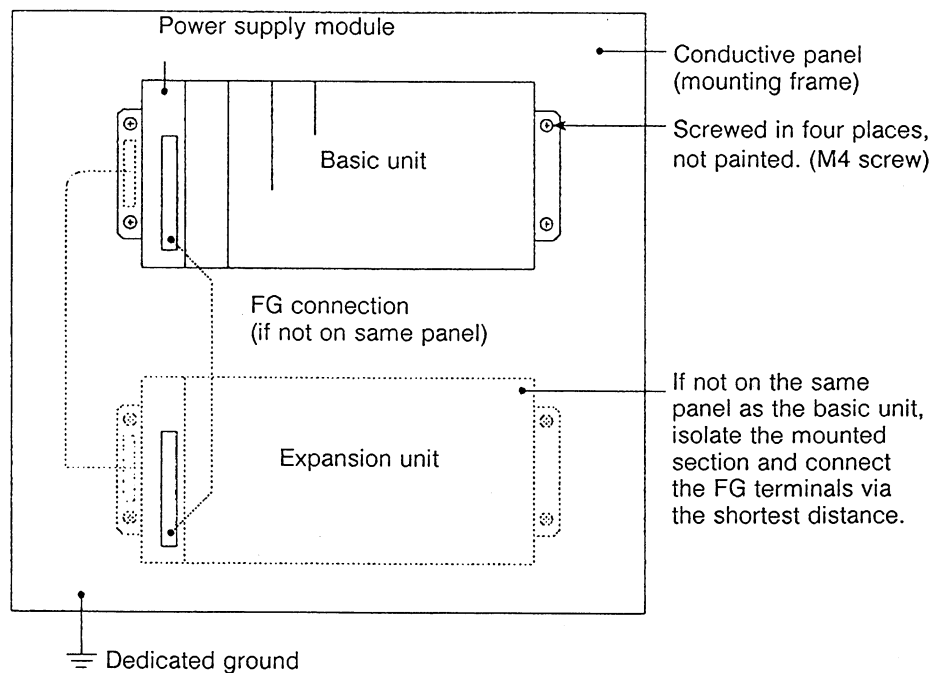
Check the grounding against the following criteria:

1. The case containing the electronic equipment must not become a path for a ground current. A high-frequency current is particularly harmful.
2. Equalize the ground potentials when more than two units of electronic equipment are to be connected. Ground them at a single point.
3. Do not connect the ground of the EX100 to that of high-power systems.
4. Do not use a ground that has an unstable impedance, such as painted screws, or grounds subject to vibration.

4.5.2 Ground methods according to the installation

1. Mounting on a highly conductive frame

Apply the following grounding if the frame on which the EX100 is mounted is itself highly conductive and does not share a ground with other high-power systems.



NOTE

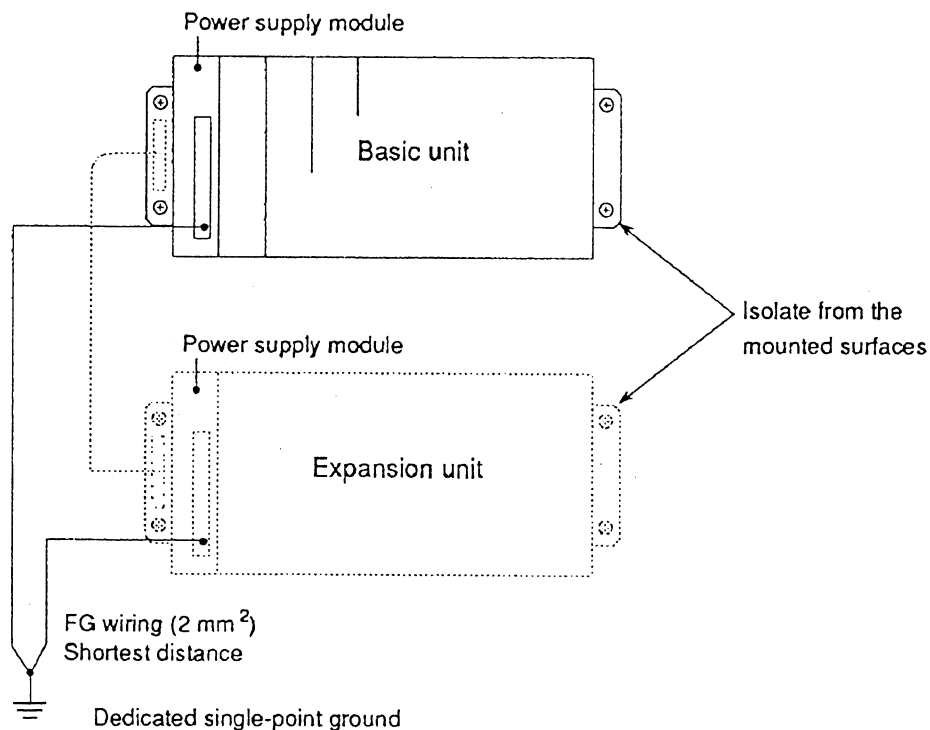


If the frame is not highly conductive, if the ground for the frame is shared with that for high power systems, or if the ground for the frame is not stable, isolate the EX100 from the panel as described below.

4. Installation and Wiring

2. Isolated mounting

If the mounting frame is not highly conductive, isolate the EX100 from the mounting frame, and connect the grounding line as follows.



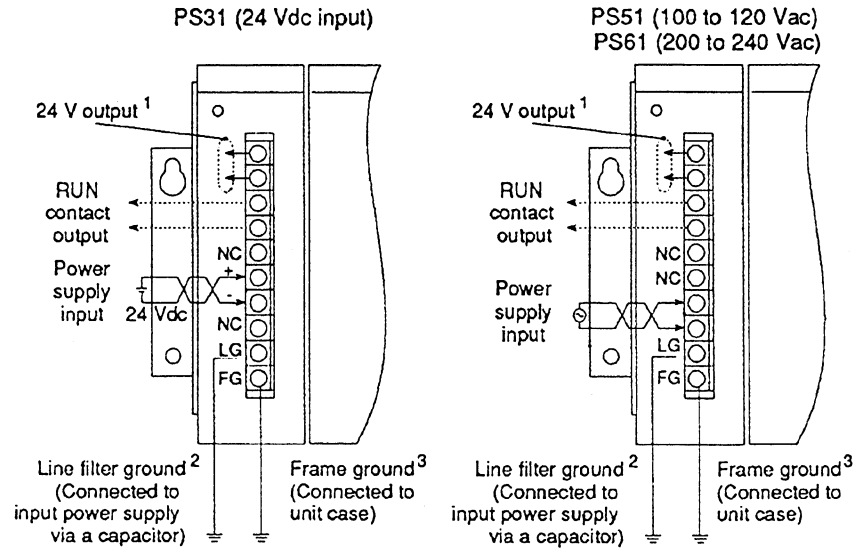
3. Grounding without a ground point

If there is no suitable grounding or no dedicated ground available, isolate the EX100 from the mounting frame in a similar way to that described in 2. above. There will be no problem operating the EX100 as long as the FG terminals of the basic and expansion units are connected. However, for safety, provide single-point grounding to a point, with impedance.*

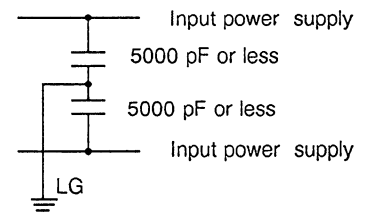
- * Resistance: Ground the frame through 1 W-1 K Ω .
- Inductance: Ground the frame through 2 A-100 μ H

4. Installation and Wiring

- 4.6 Wiring the power supply** Wire the external power supply to the EX100 power supply module. When using the expansion unit, supply the power simultaneously to the basic unit and expansion unit.



- ¹ The total power of the internal 5 V and the 24 V output must be 15 W or less. Connection to another power supply system is not possible.
- ² Normally, the LG and FG terminals are shorted. However, if the leakage current through LG is a problem, or if the ground for the power supply system is provided separately, the LG terminal should be open.
- ³ For details, see Section 4.5, Grounding.

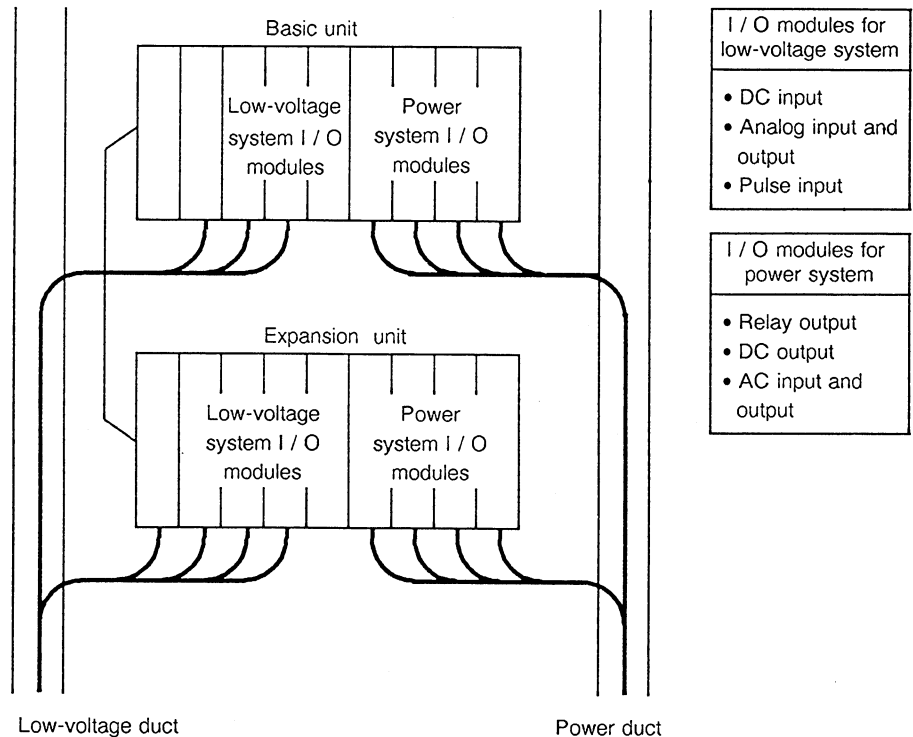


4. Installation and Wiring

4.7 Wiring the I / O modules and application precautions

This section describes precautions for wiring the I / O modules. Precautions for applications are given as a reference for wiring field inputs and outputs.

4.7.1 Module layout and I / O wiring

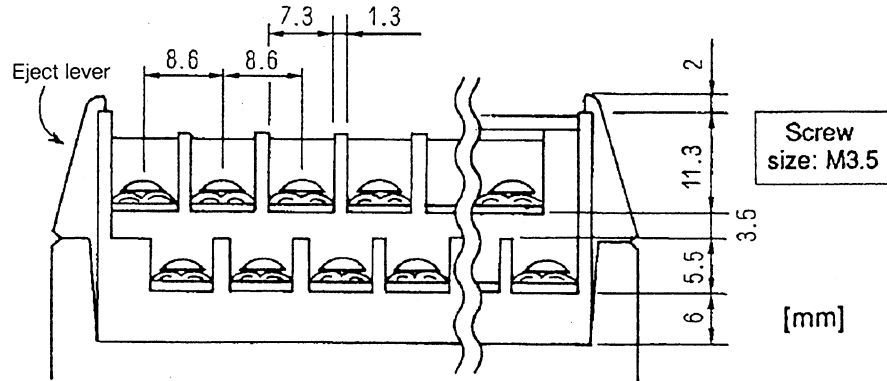


1. To improve the unit's resistance to signal interference, install modules for low-voltage signals toward the left of the unit, and modules for power signals toward the right. Also, separate the wires of each.
2. Allow at least 70 mm (2.8 in) clearance between the units and between other control equipment to allow access for maintenance and ventilation.
3. When installing the unit near high-voltage or high-power equipment, leave at least 200 mm (8 in) clearance, or shield the unit with a steel plate.
4. 18 AWG (0.75 mm²) to 22 AWG (0.3 mm²) wires are recommended for I / O signals.

4. Installation and Wiring

4.7.2 Dimensions of the terminal block

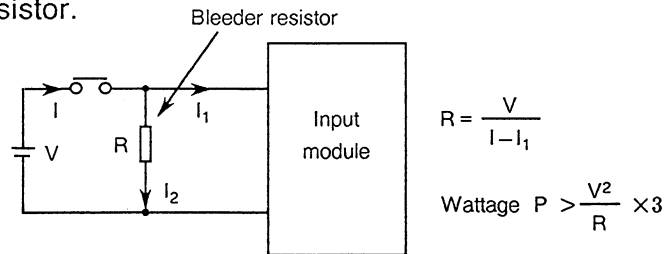
The dimensions of the terminal blocks for the I / O modules (2 levels, 18 pins) are shown below. The sizes of the screws and distance between terminals of both the 18-pin and 10-pin terminal blocks are the same.



Applicable wire: 14 AWG (2mm²) to 22 AWG (0.3mm²)

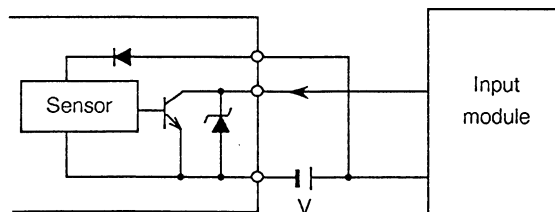
4.7.3 Application precautions for input modules

1. Minimum ON / OFF time of the input signal
The following conditions guarantee correct reading of the ON / OFF state of the input signal:
Input ON time: ON delay time + the time for one scan
Input OFF time: OFF delay time + the time for one scan
The ON and OFF times of the input signals must be longer than these intervals.
2. The reliability of some contacts cannot be guaranteed by the specified input current. In this case, install an external bleeder resistor.



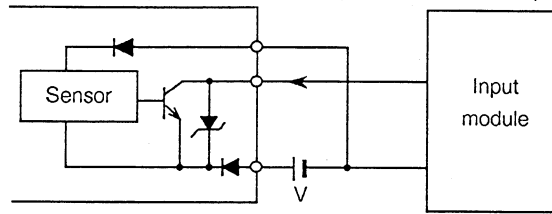
3. An example of connecting transistor output equipment to an input module is shown below:

- For NPN open collector (+ve common)

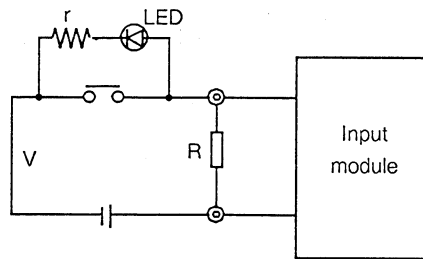


4. Installation and Wiring

- For PNP open collector (-ve common)

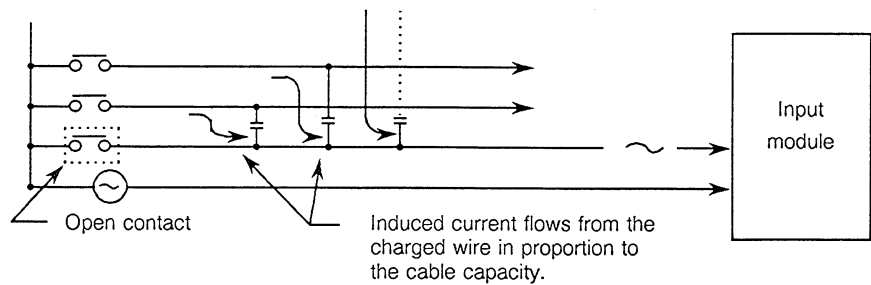


4. When a switch with an LED is used, the input module sometimes cannot recognize that the switch is off due to the current leakage through the LED. In this case, install a bleeder resistor, R , to reduce input impedance.



5. With ac input signals, if the external cable is long or if a multi-core cable is used, an induced current flows from the charged wire to the open wire, in proportion to the capacities of the cables. In this case, sometimes the voltage reaches the level of the ON input even though the contact is open, causing the module to malfunction for no apparent reason. The usual practice when this happens is to reduce input impedance.

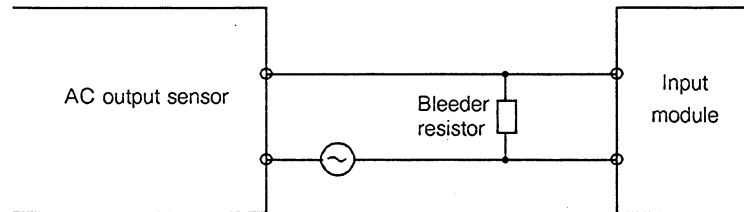
Install a resistor or a resistor and capacitor between the input and common terminals, or use shielded cables.



Such precautions are necessary when dealing with a large number of ac input signals.

4. Installation and Wiring

6. If an ac output sensor is connected, it is sometimes not possible to detect the OFF state due to a leakage current. This problem can be rectified by installing an external bleeder resistor.



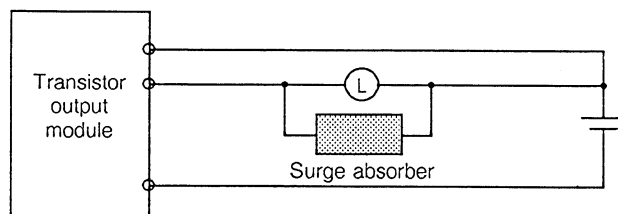
Select a bleeder resistor according to the following criteria:

- The voltage between the input terminals must be lower than the OFF voltage when the sensor is switched off.
- The current must be within the allowable values when the sensor is switched on.
- Calculate the wattage of the bleeder resistor by multiplying the current when the sensor is switched on times three.


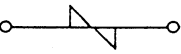

4. Installation and Wiring

4.7.4 Application precautions for transistor output modules

1. Power must be supplied to the internal control circuit of the transistor output module. If power is connected with the polarities reversed, the internal fuse will blow. Be sure polarity is correct.
2. Over current protection
The transistor output module contains the fuse(s). The transistor cannot always be protected against a shorted load. The fuse can, however, protect the transistor module if the pattern inside the module burns, or if the external cable burns.
3. Output surge protection
A relatively large surge occurs if an inductive load is connected to the output. This surge passes through the external wiring and sometimes adversely affects other systems. To eliminate this surge, install a surge-absorbing device in parallel to the inductive load, as shown below.



Select a surge absorber that precisely meets the requirements.

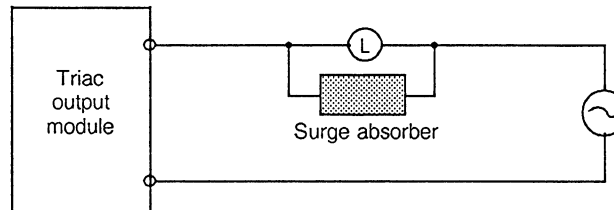
- ① Flywheel diode
(for clamping voltage)  Inverse withstand voltage: At least three times that of the power supply
Forward current: Larger than the load current
- ② Varistor
(for clamping voltage)  The voltage rating is roughly twice the maximum (peak) voltage of the power supply.
- ③ Snubber (CR) circuit (for attenuating high frequencies)  R: 0.5 to 1 Ω per volt coil voltage
C: 0.5 to 1 μF per ampere of coil current (non-polarity capacitor)

4. Installation and Wiring

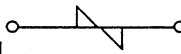

4.7.5 Application precautions for triac output modules

1. Overcurrent protection
The triac module contains one 2 A fuse for every four output points. When the load short circuits, the fuses are designed to blow to protect the triacs. However, if a fuse blows, semiconductor devices may be damaged to some extent. Therefore, when installing this module, short circuits must be avoided. Pay particular attention to the wiring.

2. Output surge protection

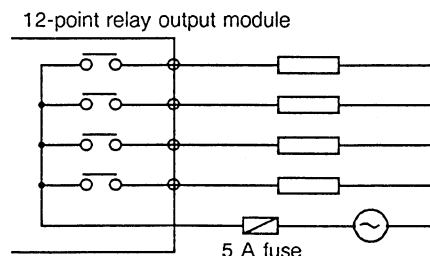


Select a surge absorber that precisely meets the requirements.

- ① Varistor (for clamping voltage)
 The voltage rating is roughly 1.2 times the maximum (peak) voltage of the power supply
- ② Snubber (CR) circuit (for attenuating coil high frequencies)

R: 0.5 to 1 Ω per volt coil voltage
C: 0.5 to 1 μF per ampere of current (non-polarity capacitor)

4.7.6 Application precautions for relay output modules

1. A power supply of 24 Vdc must be supplied to the relay drive circuit of the relay output module. Connect a power supply of 24 Vdc \pm 10% between the positive and negative terminals.
2. The relay output module does not contain a fuse for protection against overload. Be sure to install the appropriate fuses.

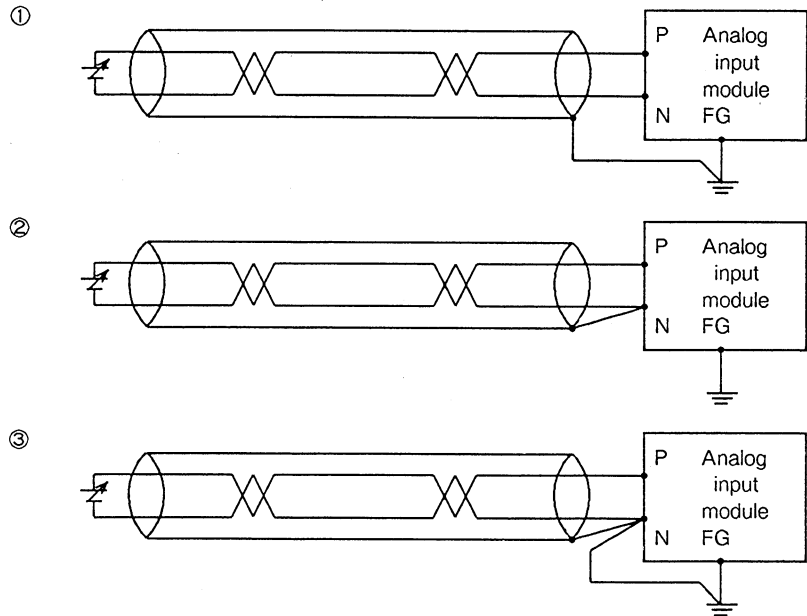


If there is no fuse protecting the module, the patterns inside the module will be burned in the event of a short circuit, overload, etc.

3. Output surge protection
As mentioned in the sections concerning the triac and transistor output modules, it is necessary to install a surge absorbing device for protection where an inductive load is used.

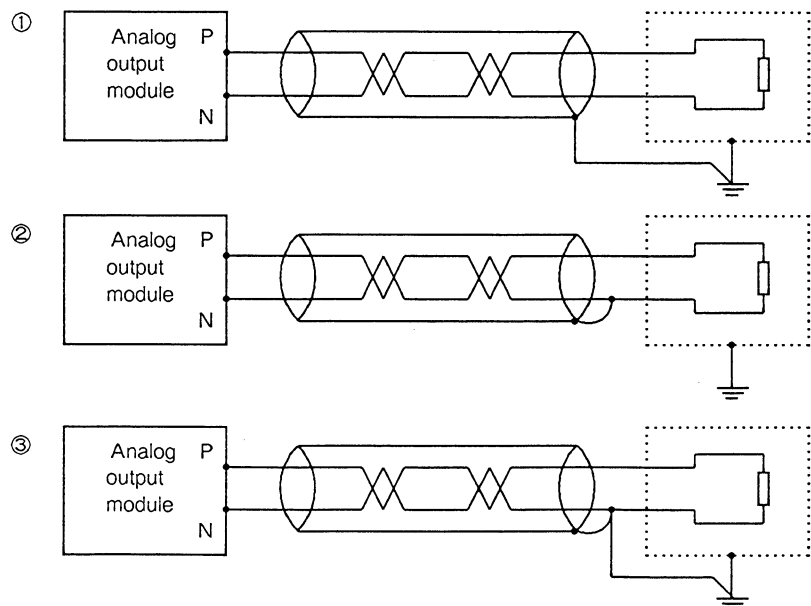
4.7.7 Application precautions for analog input modules

1. An external power supply of $24\text{ Vdc} \pm 10\%$ must be supplied to the analog input module. The wires carrying the power supply should be separated from other wires to prevent signal interference.
2. The shield of the input cable should be grounded as ① below. However, in some cases, ② or ③ will be more effective.



4.7.8 Application precautions for analog output modules

1. An external power supply of $24\text{ Vdc} \pm 10\%$ must be supplied to the analog output module. The wires carrying the power supply should be separated from other wires to prevent signal interference.
2. The shield of the output cable should be grounded as ① below. However, in some cases, ② or ③ will be more effective.

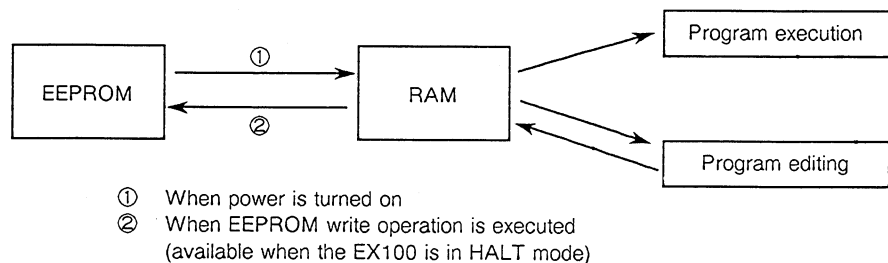


5.1 EEPROM operation

The EX100 is equipped with an EEPROM and a RAM as standard features. The user program is stored in the EEPROM so that the user program can be maintained without need of a battery.

The user program stored in the EEPROM is transferred to the RAM when power is turned on. Subsequent program execution is done based on the RAM contents. Program editing is also done based on the RAM contents.

Therefore, if the program is modified, it is necessary to execute the EEPROM write operation by using the programmer.



There are two methods for doing the EEPROM write operation:

- (1) Enter the EEPROM write command (control 94) from the programmer.

(Key operation using the GP100)
[CNTL] [9] [4] [EXE] [EXE]

- (2) Set special relay R62E to ON by using the programmer.

(Key operation example using GP100)

[MON] [EXE] Set the monitor mode.

[HOME] [←] [↑] Move the cursor to the
auxiliary data monitor area.

[STS] [R] [6] [2] [E] [EXE] ... Register R62E.

[DSET] [1] [EXE] Set R62E to ON.



- (1) If a programmer that does not support the EEPROM write command is used, method (2) should be used.

- (2) The R62E is reset to OFF automatically after EEPROM write is completed.

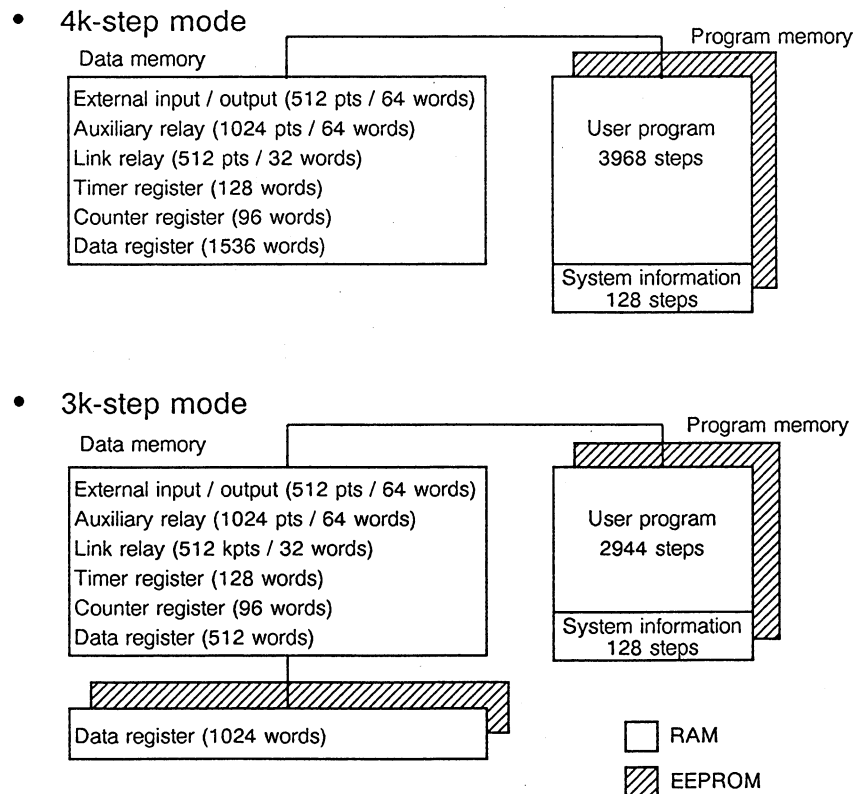
5. Operating the EX100

- 5.2 Memory settings** The internal memory of the EX100 consists of program memory and data memory. Program memory is used to store the user program. Data memory is used to store the ON / OFF status of external I / O and various control data. (see figure below)

As explained previously, the EX100 has an EEPROM for memory back-up. The user can select the EEPROM utilization in either 4K-step mode or the 3K-step mode by using the DIP switch on the CPU module.

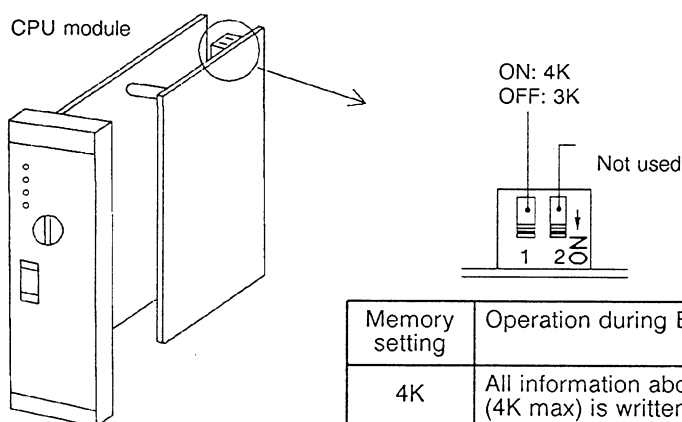
- 4K-step mode: Program memory (4K steps max.) is stored in the EEPROM.
- 3K-step mode: Program memory (3K steps max.) and the contents of 1024 data registers (D0512 to D1535) are stored in the EEPROM.

Internal memory configurations for each mode are shown in the following illustration.



5. Operating the EX100

Select the memory setting by using the DIP switch on the CPU module as follows.



Memory setting	Operation during EEPROM write
4K	All information about the user program (4K max) is written into the EEPROM.
3K	User program (3K max) and the contents of data registers D0512 to D1535 (total 1024 words) are written into the EEPROM.

(This DIP switch is set to 4K before shipment.)

The following table shows the conditions for transferring program (and data) from the EEPROM to the RAM when power is turned on.

Memory setting at power on	Memory setting during previous EEPROM writing		
	4K-step mode		3K-step mode
	3K or less of program	3K or more program	—
4K-step mode	Transfers program normally.	Transfers program normally.	Transfers only program.
3K-step mode	Transfers program and data. ¹	Does not transfer. ²	Transfers program and data normally.

1. The program is properly transferred, but the data in registers D0512 to D1535 is undefined.
2. The operation is inhibited because of a memory setting mismatch. It is necessary to clear the memory.



NOTE

- (1) When the 3K-step mode is selected, data registers D0512 to D1536 should be specified as retentive memory. Otherwise, these data are transferred but then cleared at register initialization. (see 5.4)
- (2) When the 3K-step mode is selected, data stored in the EEPROM can be accessed by the user program. (see 9.5)

5. Operating the EX100

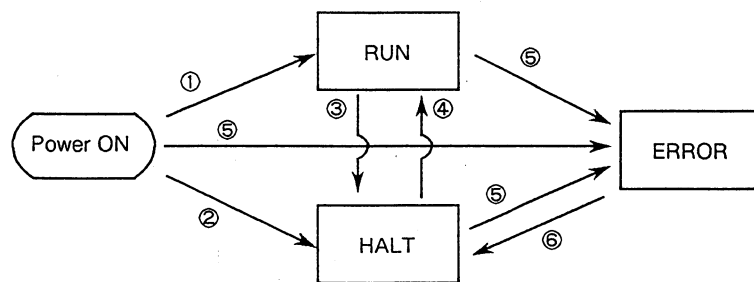
5.3 Operation modes The EX100 has two basic operation modes, the RUN mode and the HALT mode. The EX100 also has the ERROR mode, for use as an abnormal state.

RUN: In the RUN mode, the EX100 reads external signals, executes the user program stored in the RAM, and outputs signals to the external devices according to the user program. It is in the RUN mode that the EX100 performs scans, which is the basic operation of PCs. A program is not normally modified in the RUN mode. However, the EX100 has an online program changing function, which enables program changes when the system is in operation. This enables the program to be modified without turning off external outputs in the RUN mode.

HALT: In this mode, execution of a user program is stopped, and all outputs are switched off. This is the mode in which programming is normally performed. Writing the program into the EEPROM can only be performed when the EX100 is in the HALT mode.

ERROR: The EX100 enters the error mode if internal trouble is detected during self-diagnosis, and normal functioning cannot continue. Program execution stops and all outputs are switched off. To exit from the error mode, enter an Error Reset command from the programmer, or cycle power off and then on again.

The transitional conditions of each mode are shown below.



- ① When the operation control switch is in RUN or RUN-P.
- ② When the operation control switch is in HALT.
- ③ When the operation control switch is moved from RUN or RUN-P to HALT, or when the HALT command is executed from the programmer.
- ④ When the operation control switch is moved from HALT to RUN or RUN-P, or when the RUN or RUN-F commands are executed from the programmer. (enabled when the switch is in RUN or RUN-P)
- ⑤ When an error is detected during self diagnosis.
- ⑥ When the error reset command is executed from the programmer.

5. Operating the EX100

When the operation control switch is in the RUN-P position, program modifications and EEPROM write operation cannot be performed. (An error message will be displayed on the programmer if an attempt is made)

Therefore the user program can be protected from unauthorized operations by setting the switch in the RUN-P position.

The following table lists the available functions relevant to the program modifications in each key position.

Key position	Programmer's command	Operation mode	Available function
HALT	RUN / RUN-F / HALT commands are disabled.	HALT	Programming and EEPROM write are available.
RUN	RUN / RUN-F / HALT commands are enabled.	RUN	On-line program changes are available. EEPROM write is not available.
		HALT	Programming and EEPROM write are available.
RUN-P	RUN / RUN-F / HALT commands are enabled.	RUN	On-line program changes and EEPROM write are not available.
		HALT	Programming and EEPROM write are not available.

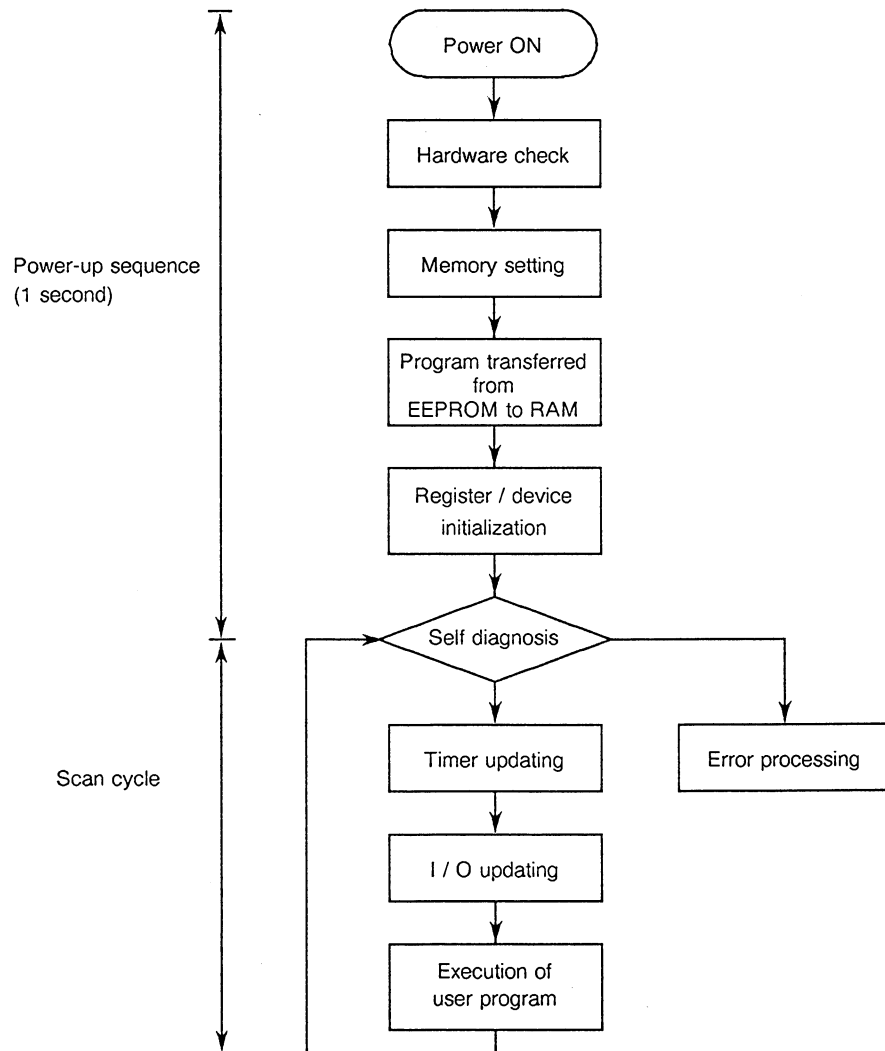
NOTE



- (1) Even when the key position is RUN-P, the contents of data memory in RAM can be changed.
- (2) On-line program changes are disabled when:
 - ① The total number of program control instructions, i.e., END, MCS, MCR, JCS, JCR, are changed.
 - ② Execution order of program control instructions are changed.
- (3) If the operation mode has entered ERROR, writing operations are disabled. Execute the error reset command to return to HALT mode.

5. Operating the EX100

- 5.4 Scanning** The flowchart below shows the internal operations performed by the EX100 from the time power is turned on to the processing for program execution. As the diagram shows, executing a program consists of continuous scanning operations. One scan is a cycle starting with self diagnosis and finishing with the completion of program execution.



Hardware check:
Memory setting:
Program transfer:
Register/device initialization:
Self diagnosis:
Timer updating:
I / O updating:
User program execution:
Error processing:

Checks and initializes the memory, ICs and I / O buses.
Sets a 3K or 4K system according to the DIP switch setting.
Transfers data from the EEPROM to the RAM.
Initializes registers and devices. (See next page.)
Checks for the existence of errors. (See next page.)
Updates timing relays and timer registers.
Updates external I / O registers and link registers.
Executes the user program.
Processes errors detected in the EX.

Initializing registers and devices when power is turned on

Register / device	Retentive data		
	Retentive memory (KEEP AREA TOP)	Forced devices	Forced coils
External input and output (X / Y)	Cannot be set.	Holds only devices specified as forced.	Holds only devices specified as forced coils.
Auxiliary relay (R)	Holds registers specified for retentive memory.	Cannot be set.	Holds only devices specified as forced coils.
Timer (T)	Holds registers specified for retentive memory.	Cannot be set.	Cannot be set.
Counter (C)	Holds registers specified for retentive memory.	Cannot be set.	Cannot be set.
Data register (D)	Holds registers specified for retentive memory.	Cannot be set.	Cannot be set.
Link register (Z)	Cannot be set.	Holds only devices specified as forced.	Holds only devices specified as forced coils.



NOTE When power is turned on, all registers and devices that are not designated for retention are cleared to 0. Retentive devices and registers consist of devices that are forced, and devices and registers specified in the retentive memory.

Self diagnosis

Item	Check method
Program memory	Checks the program using checksum.
Program syntax	Checks for the existence of the END instruction, checks the syntax of JCS / JCR, MCS / MCR, and output operands.
Scan time	Checks program scan time.
I / O	Checks the response from I / O modules.
Illegal instructions	Checks for the existence of illegal instructions.
TOSLINE	Checks the data link modules.
Computer link	Checks the computer link interfaces.
Watchdog timer	Checks the processor operation.

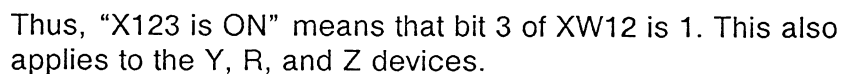
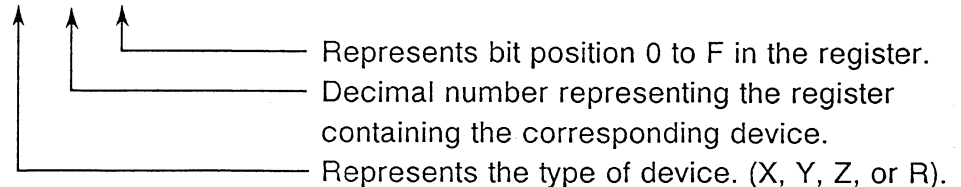
6.1

Devices are divided into four types:

- Registers are divided into seven types:

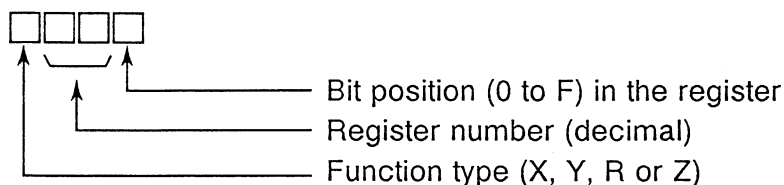
- ## Device and register numbers

Bit position


$$\underline{X} \quad \underline{12} \quad \underline{3}$$


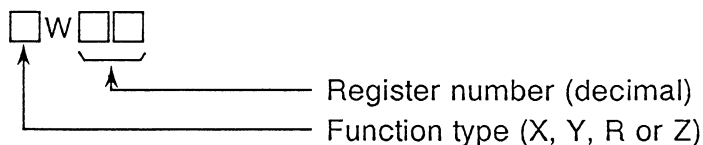
6. Programming

- Addressing devices** • External input, external output, auxiliary relay and link devices.



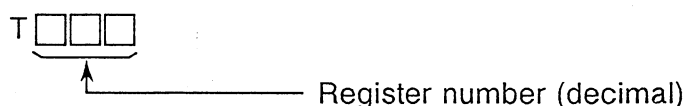
(e.g., X000, Y027, R10A, Z31F, etc.)

- Addressing registers** • External input, external output, auxiliary relay and link registers.

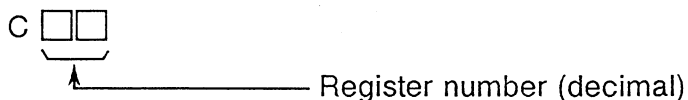


(e.g., XW00, YW02, RW10, ZW31, etc.)

- Timer register



- Counter register

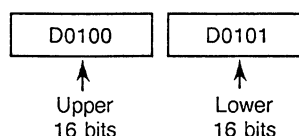


- Data register



- NOTE**
- (1) The available data range in each register is 0 to 65535 (H0000 to HFFFF) except for the timer register. The timer register range is 0 to 32767 (H0000 to H7FFF), because the timer instructions work for double precision internally.

- (2) Double-length (32 bits) data is available in two consecutive registers. (e.g., D0100 and D0101)



In this manual, a double-length register is expressed by D0100 · D0101.

Device / register	Symbol	No. of points	Address
External input device	X	512 points total	X000~X31F
External output device	Y		Y000~Y31F
External input register	XW	64 words total	XW00~XW63
External output register	YW		YW00~YW63
Auxiliary relay device	R	1024 points	R000~R63F
Auxiliary relay register	RW	64 words	RW00~RW63
Data register	D	1536 words	D0000~D1535
Link device	Z	512 points	Z000~Z31F
Link register	ZW	32 words	ZW00~ZW31
Timer register	T	128 words	T000~T127
Counter register	C	96 words	C00~C95



NOTE Registers can be regarded as a group of 16 continuous devices, except for data registers (D), timer registers (T), and counter registers (C). External input registers XW00 to XW31 and external output registers YW00 to YW31 can also be used as devices. However, registers XW32 to XW63 and YW32 to YW63 can only be used as registers.

External input devices (X)

These devices indicate the on / off states of inputs through the input modules. External input devices can be used many times in a program. The function type X is assigned to various input modules.

External output devices (Y)

External output devices store the on / off signals that drive the external devices via the output modules. They can be used for coils in a program. The function type Y is assigned to various output modules.

External input registers (XW)

These are 16-bit registers for storing numerals, such as analog input, pulse input, and values received from the input modules. This function type, XW, is assigned to various input modules. The number of registers is determined by the module.

External output registers (YW)

These 16-bit registers are used for storing numerals, such as analog output and numerical indicators for output via the output modules. This function type, YW, is assigned to various output modules. The number of registers is determined by the module.

6. Programming

Auxiliary relay devices and registers (R / RW)

The auxiliary relay devices, R, are used to store intermediate results of sequences. The auxiliary relay registers, RW, are used to store temporary results of functional instructions. The data in R / RW cannot be output directly to the output modules. It is necessary to move the data to Y / YW.

It is possible to specify these registers in retentive memory to retain their data in the event of a power failure.

The topmost area of the devices, R600 to R63F, is assigned to the special relays, as explained below.

Data registers (D)

Data registers are the same as auxiliary relay registers, RW, except that data registers cannot be used as devices. If the memory setting is the 3K mode, 1K of the register (D0512 to D1535) can be saved in the EEPROM as fixed data. It is possible to specify these registers in retentive memory to retain their data in the event of a power failure.

Link devices and registers (Z / ZW)

Link devices and registers are used for the TOSLINE-30 data link system. Each register is specified either as a TALKER or LISTENER. The data in the TALKER register is sent to other stations in the TOSLINE-30 network, and the LISTENER register receives data from the other stations in the network.

If the TOSLINE-30 is not used, these registers can be used for the same purpose as RW registers.

Timer registers (T)

Timer registers are used for storing the remaining time of timer instructions, such as the on and off delay timers and single-shot timers. These registers cannot be used for storing the results of functional instructions. It is possible to specify these registers in retentive memory to retain their data in the event of a power failure.

T000 to T119: 100 ms timer

T120 to T127: 10 ms timer

Counter registers (C)

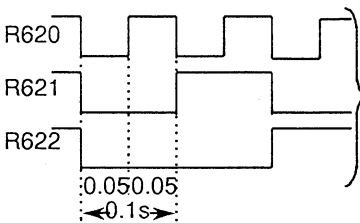
Counter registers are used to store the current count of counter instructions. These registers cannot be used for storing the results of functional instructions. It is possible to specify these registers in retentive memory to retain their data in the event of a power failure.

Special relays

Devices R600 to R63F are assigned to the special relays as listed below. These relays can be used for interlocking in a program.

Device	Name	Comments
R600	Data link normal ZW00	<ul style="list-style-type: none"> • ON if transmission to the corresponding register is normal. • OFF if a transmission error occurs, or if transmission is not used. ON if transmission becomes normal. • This area can be used as a normal auxiliary relay device if transmission is not used. (It is recommended that this area not be used to allow for future system expansion.)
R601	Data link normal ZW01	
R602	Data link normal ZW02	
R603	Data link normal ZW03	
R604	Data link normal ZW04	
R605	Data link normal ZW05	
R606	Data link normal ZW06	
R607	Data link normal ZW07	
R608	Data link normal ZW08	
R609	Data link normal ZW09	
R60A	Data link normal ZW10	
R60B	Data link normal ZW11	
R60C	Data link normal ZW12	
R60D	Data link normal ZW13	
R60E	Data link normal ZW14	
R60F	Data link normal ZW15	
R610	Data link normal ZW16	
R611	Data link normal ZW17	
R612	Data link normal ZW18	
R613	Data link normal ZW19	
R614	Data link normal ZW20	
R615	Data link normal ZW21	
R616	Data link normal ZW22	
R617	Data link normal ZW23	
R618	Data link normal ZW24	
R619	Data link normal ZW25	
R61A	Data link normal ZW26	
R61B	Data link normal ZW27	
R61C	Data link normal ZW28	
R61D	Data link normal ZW29	
R61E	Data link normal ZW30	
R61F	Data link normal ZW31	

6. Programming

Device	Name		Comments
R620	Timing relay 0.1 s		 <p>Constitute a binary counter</p> <p>Note that 0.1 s and 0.2 s timing relays sometimes cannot be read if the scan time is long.</p>
R621	Timing relay 0.2 s		
R622	Timing relay 0.4 s		
R623	Timing relay 0.8 s		
R624	Timing relay 1 s		
R625	Timing relay 2 s		
R626	Timing relay 4 s		Note that 0.1 s and 0.2 s timing relays sometimes cannot be read if the scan time is long.
R627	Timing relay 8 s		
R628	Calendar function flag		Used for the calendar function.
R629	HOLD device		Enters the HOLD state when ON.
R62A	EEPROM write flag		Used for EEPROM write instructions.
R62B	Communication priority mode flag		Used for the communication priority mode
R62C			Reserved by system (cannot be used)
R62D	Auto RUN-F enable flag		Used for the auto RUN-F function.
R62E	Always OFF (EEPROM write device)		Writes to the EEPROM when ON.
R62F	Always ON		Always ON relay.
R630	Self diagnosis error: • ON if an error occurs • Cleared by resetting the error	CPU error	Watchdog timer error.
R631			Reserved by the system. (Cannot be used)
R632			
R633		EEPROM error	EEPROM data is abnormal.
R634		I / O error	I / O bus is abnormal.
R635		I / O reference error	Mounted I / O and assigned I / O mismatch.
R636		Program error	Program data abnormal
R637		Scan time over	Scan time exceeded 200 ms.
R638			Reserved by the system (Cannot be used)
R639			
R63A	Communication port flag		ON = LINK / OFF = PROGMR (OFF if standard CPU)
R63B	Programmer transmission error		ON if communication with programmer is abnormal.
R63C	Automatically reset to OFF when restored.	TOSLINE error	Abnormality found in TOSLINE-30
R63D		Computer link error	Computer link abnormal
R63E	Execution of diagnosis instruction		ON when user-specified diagnosis instruction is executed.
R63F			Reserved by system (Cannot be used)

6.2 I / O allocation

I / O allocation is performed to assign the EX100's registers / devices to the individual I / O modules and TOSLINE-30 modules.

The external input registers / devices (XW / X) are assigned to the input modules.

The external output registers / devices (YW / Y) are assigned to the output modules. The register numbers of the external input and output registers are consecutive. Thus one register number can be assigned for either input or output.

The link registers / devices (ZW / Z) are assigned to the TOSLINE-30 modules independent of the external input and output registers.

The external input and output registers can be regarded as a group of 16 continuous devices. Hence, if a register is assigned to a discrete I / O module, each device in the register is assigned to an external signal.

NOTE See 6.1 for details of registers / devices.



I / O allocation methods

There are two methods used for I / O allocation:

(1) Automatic I / O allocation:

When the automatic I / O allocation command (control 5) is entered from the programmer, the EX100 CPU reads the types of modules mounted, then assigns the registers to the modules according to rules that will be explained later.

(2) Manual I / O allocation

Manual I / O allocation is done by setting the types of modules into the slots one by one by the programmer (system 2 screen). Then register assignment is determined according to rules that will be explained later.

This method is used when it is necessary to program without modules, or when it is necessary to set functions that cannot be performed by automatic I / O allocation.

6. Programming

The module types that are read to the EX100's CPU by automatic I / O allocation are listed in the following table.

Part No.	Description		Module Type
EX10-MDI31	16-point dc / ac input (12 – 24 Vdc / ac)		X 1W
EX10-MDI32	32-point dc input (24 Vdc)		X 2W
EX10-MIN51	16-point ac input (100-120 Vac)		X 1W
EX10-MIN61	16-point ac input (200-240 Vac)		X 1W
EX10-MRO61	12-point relay output (240 Vac / 24 Vdc)		Y 1W
EX10-MRO62	8-point relay output (isolated) (240 Vac / 24 Vdc)		Y 1W
EX10-MDO31	16-point transistor output (5 – 24 Vdc)		Y 1W
EX10-MDO32	32-point transistor output (5 – 24 Vdc)		Y 2W
EX10-MAC61	12-point triac output (100 – 240 Vac)		Y 1W
EX10-MAI21	4 ch analog input (4-20 mA / 1-5 V)		X 4W
EX10-MAI22			X 4W
EX10-MAI31	4 ch analog input (0-10 V)		X 4W
EX10-MAI32	4 ch analog input (± 10 V)		X 4W
EX10-MAO31	2 ch analog output (0-10V / 1-5 V / 4-20 mA)		Y 2W
EX10-MAO22	2 ch analog output (1-5 V / 4-20 mA)		Y 2W
EX10-MAO32	2 ch analog output (± 10 V)		Y 2W
EX10-MPI21	1 ch pulse input		X 2W
EX10-MMC11	1 axis motion control		X + Y 4W
EX10-MLK11	TOSLINE-30 (wire)	8-word setting	Z 8W
		16-word setting	Z 16W
		32-word setting	Z 32W
EX10-MLK12	TOSLINE-30 (optical)	8-word setting	Z 8W
		16-word setting	Z 16W
		32-word setting	Z 32W



(1) The module type is expressed by a combination of the function type (X, Y or Z) and the number of registers assigned.

(2) For the TOSLINE-30, the link registers (ZW) are assigned independent of the external input and output registers. The number of registers assigned is determined by the setting status of the DIP switches on the module. (Transmission capacity)

The following table lists the module types and functions available when using manual I / O allocation.

Function type	No. of assigned registers	Remarks	
X	01, 02, 04, 08	Input	
Y	01, 02, 04, 08	Output	
X + Y	02, 04, 08	Input / output	
Z	08, 16, 32	TOSLINE-30	
Blank	(01)	Vacant slot	
iX	01, 02, 04, 08	Input	Batch I / O update is not executed
iY	01, 02, 04, 08	Output	
iX + Y	02, 04, 08	Input / output	
SP	01, 02, 04, 08, 16, 32	Space (Vacant slot)	
OPT	—	Option	

- (1) Module type is expressed by a combination of the function type and the number of registers assigned.
- (2) A slot containing no module is allocated as blank, and one external output register (YW) is assigned internally.
- (3) iX, iY, iX + Y, SP and OPT are available only when manual I / O allocation is used.
- (4) Batch I / O update is not executed for modules allocated as iX, iY or iX + Y.
I / O update for such modules is executed only when the immediate input or output instructions are executed.
- (5) SP is used for assigning the optional number of registers to a vacant slot. External output registers (YW) are assigned internally.
- (6) OPT does not occupy any registers. OPT can be used for a vacant slot to which no register assigned.

6. Programming

Rules for I / O allocation

- (1) Registers are assigned to modules installed in the EX100 sequentially from left to right.
- (2) If an expansion unit is connected, registers are assigned to it in the same way sequentially after the basic unit's assignments.
- (3) With regard to I / O allocation, a 6-slot unit and a 9-slot unit are allocated in the same manner. This means that three vacant slots are provided internally after the 6-slot unit.
- (4) Vacant slots are allocated as blanks during automatic I / O allocation. One external output register (YW) is assigned internally to the blank setting slot.
- (5) Consecutive register numbers are assigned to input and output modules.
- (6) Link registers (ZW) are assigned to the TOSLINE-30 module.
- (7) The minimum allocation unit is one register, that is 16 bits. Therefore, for example, when one register is assigned to a 12-point module, bits C to F are invalidated.

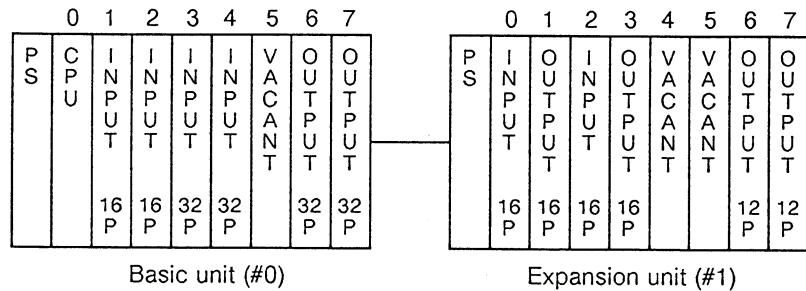


NOTE

Operation (RUN) with vacant slots between allocated modules is possible if the vacant slots have been designated as blank, SP or OPT. However, if the slot settings are other than these three, operation (RUN) is not possible owing to the module-response check. In such a case, the forced operation (RUN-F) can be used to override the module-response check. (See 9.2)

6. Programming

Example 1 (1) When an EX100's modules have been mounted as follows:



(2) Registers are assigned as follows by automatic I / O allocation.

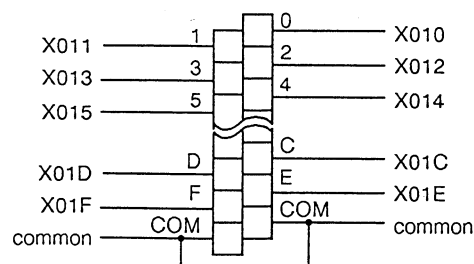
Unit	Slot	Module type	Register assignment
0	0	CPU	—
	1	X 1W	XW00
	2	X 1W	XW01
	3	X 2W	XW02, XW03
	4	X 2W	XW04, XW05
	5	Blank	(YW06)
	6	Y 2W	YW07, YW08
	7	Y 2W	YW09, YW10
1	0	X 1W	XW11
	1	Y 1W	YW12
	2	X 1W	XW13
	3	Y 1W	YW14
	4	Blank	(YW15)
	5	Blank	(YW16)
	6	Y 1W	YW17
	7	Y 1W	YW18

NOTE



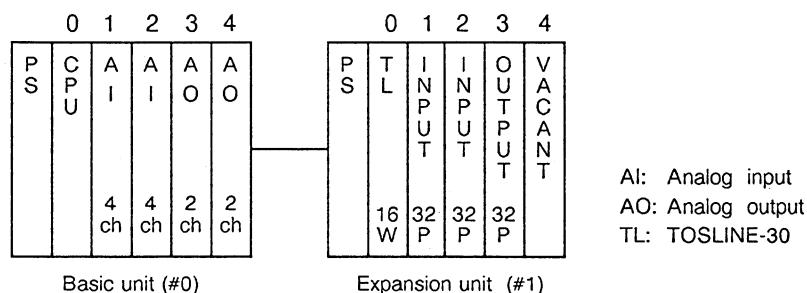
Once a register has been assigned to a module, the devices inside the register are assigned to the module's external signals. Device assignment is determined by a combination of the register number and signal number, which is indicated on the module.

(Example: 16-point input module allocated to XW01)



6. Programming

Example 2 (1) When an EX100's modules have been mounted in the following configuration:



(2) Registers are assigned as follows by automatic I / O allocation.

Unit	Slot	Module type	Register assignment
0	0	CPU	—
	1	X 4W	XW00, XW01, XW02, XW03
	2	X 4W	XW04, XW05, XW06, XW07
	3	Y 2W	YW08, YW09
	4	Y 2W	YW10, YW11
	(5)	Blank	(YW12)
	(6)	Blank	(YW13)
	(7)	Blank	(YW14)
1	0	Z 16W	ZW00 to ZW15
	1	X 2W	XW15, XW16
	2	X 2W	XW17, XW18
	3	Y 2W	YW19, YW20
	4	Blank	(YW21)
	(5)	Blank	(YW22)
	(6)	Blank	(YW23)
	(7)	Blank	(YW24)



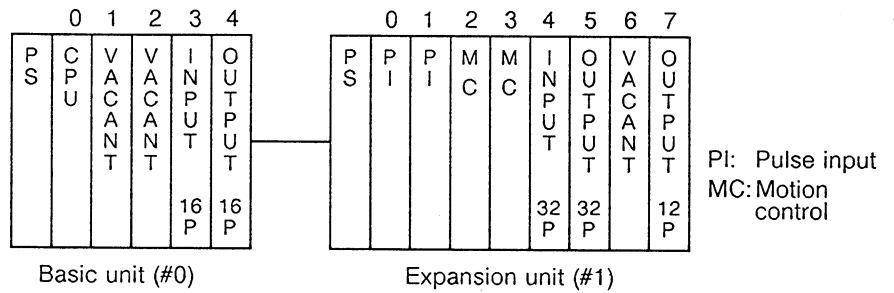
(1) There will be three vacant slots registered internally after a 6-slot unit.

(2) For analog input and output modules, lower registers are assigned to the lower channels indicated on the module.

(3) Up to four TOSLINE-30 modules can be mounted within the capacity of the link registers. (32 registers)

6. Programming

Example 3 (1) When an EX100's modules have been designed to mount as follows:



(2) If following setting has been done by manual I / O allocation, registers are assigned as follows.

Unit	Slot	Module type	Register assignment
0	0	CPU	—
	1	SP 4W	(YW00 to YW03)
	2	SP 1W	(YW04)
	3	iX 1W	XW05
	4	iY 1W	YW06
	(5)	OPT	—
	(6)	OPT	—
	(7)	OPT	—
1	0	X 2W	XW07, XW08
	1	iX 2W	XW09, XW10
	2	X+Y 4W	XW11, XW12, YW13, YW14
	3	X+Y 4W	XW15, XW16, YW17, YW18
	4	X 2W	XW19, XW20
	5	Y 2W	YW21, YW22
	6	SP 2W	(YW23, YW24)
	7	Y 1W	YW25



(1) Vacant slots that have been designated as OPT do not occupy any registers.

(2) I / O update for modules that have been allocated as iX, iY or iX + Y is executed only when the immediate input or output instructions are executed.

6. Programming

6.3 Setting the retentive memory area

Retentive memory area can be specified for the following registers.

- Auxiliary relay devices / registers (R / RW)
- Data registers (D)
- Counter registers (C)
- Timer registers (T)

The contents of registers designated for retention retain the previous data at initialization in the power up sequence. (See 5.4)

To specify registers for the retentive memory area, select the first registers by using the programmer's system information editing function. By this operation, the first specified register to the highest number register is specified as retentive memory area.

For example, if following settings have been entered:

RW ... 16
D ... 0
C ... 50
T ... Does not set

The EX100's retentive memory areas will be designated as follows.

RW16 to RW63
D0000 to D1535
C50 to C95



- (1) Data in the retentive memory are backed up by a built-in capacitor. (Back-up period: 7 days at 25°C)
An optional battery is required for longer back up.
- (2) If the memory setting has been set to the 3K mode, the contents of D0512 to D1535 are stored in the EEPROM. In this case, this area should be specified as retentive memory. Otherwise these data will be transferred from EEPROM to RAM, but then cleared at initialization. (See 5.2)

6.4 Program configuration and execution

The user program comes from the system information and the execution program. (Normally the execution program is simply called "the user program")

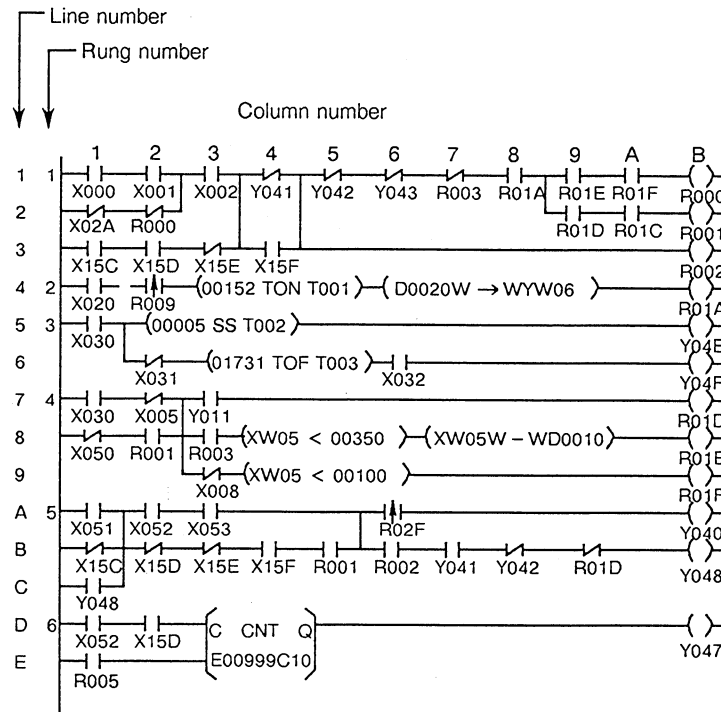
The system information contains program-related data such as program ID, retentive memory areas, I / O allocation, etc. System information has a 128-step capacity.

The execution program stores the user application program, which was written in ladder diagram language. It has a 3968-step capacity when the memory setting is 4K mode. The execution program has a 2944-step capacity when the memory setting is 3K mode.

The execution program is stored in memory in units of one page each.

Each page has the following capacity.

- 14 columns by 11 lines
- 154 steps max. / page
- 32 steps max. / rung

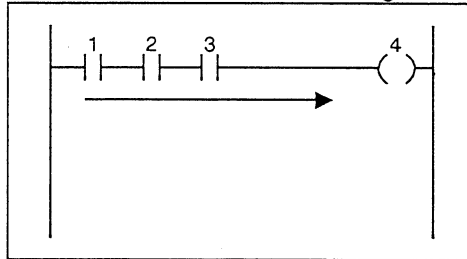


6. Programming

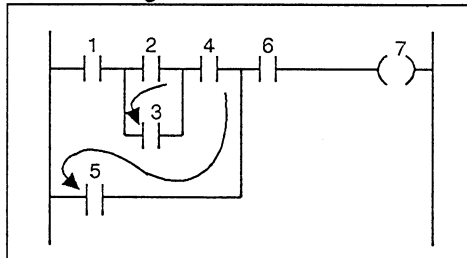
Program execution order:

- Executes sequentially from page 1 to the page having an END instruction.
- On each page, the program executes sequentially from rung 1, rung 2, and so forth.
- Each rung in a ladder diagram is executed according to the following rules.

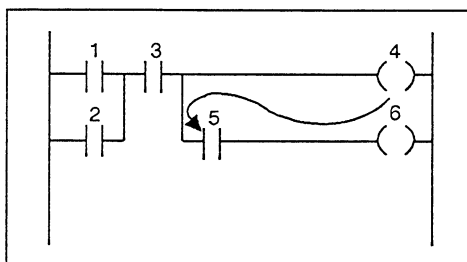
Rule 1: Execution from left to right in a simple line



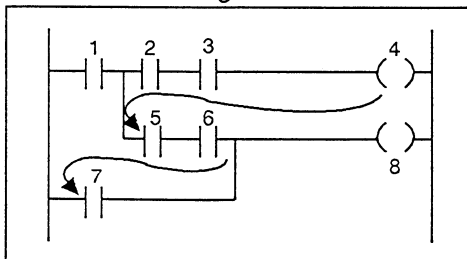
Rule 2: OR logic is executed first



Rule 3: Execution from upper lines to lower lines in branches



Execution according to a combination of rules 2 and 3



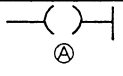
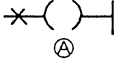


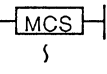
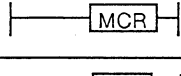
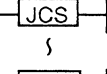
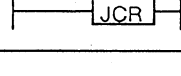
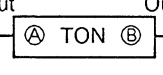
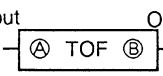
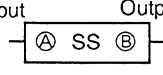
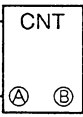

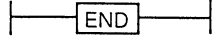


The numbers in the above figures represent program execution order.

7.1 List of instructions

Basic ladder functions

The EX100 has 15 types of basic instructions and 67 types of functional instructions, as listed below. The operations of these instructions will be described in detail later. The tables listing these instructions on the following seven pages are provided as a quick reference.

Instruction	Expression	Description	No. of steps
NO contact		NO contact of device A.	1
NC contact		NC contact of device A.	1
Coil		Relay coil of device A.	1
Forced coil		Forced coil of device A. (The coil is specified as forced.)	1
Transitional contact (rising)	Input  Output	Turns ON output for 1 scan when input changes from OFF to ON.	1
Transitional contact (falling)	Input  Output	Turns ON output for 1 scan when input changes from ON to OFF.	1
Master control	Input  	Turns OFF the power rail between MCS and MCR when MCS is OFF.	MCS, MCR 1 each
Jump control	Input  	Jumps from JCS to JCR when JCS is ON.	JCS, JCR 1 each
ON delay timer	Input  Output	Turns ON output when the time specified by A has elapsed after the input turns ON. (B is a timer register.)	2 / 3
OFF delay timer	Input  Output	Turns OFF output when the time specified by A has elapsed after the input turns OFF. (B is a timer register.)	2 / 3
Single-shot timer	Input  Output	Turns ON output for the time specified by A when the input turns ON. (B is a timer register.)	2 / 3
Counter	Count input  Enable input 	Counts the number of cycles the count input is ON while the enable input is ON and turns output ON when the number of cycles specified by A is reached. (B is a counter register.)	2 / 3
End		Specifies the end of a program.	1

7. Instruction

Data transfer instructions

Instruction (FUN. No.)	Expression	Description	No. of steps
Register transfer (FUN. 000)	$\text{Ⓐ} \quad W \rightarrow W \quad \text{Ⓑ}$	Transfers data from register Ⓐ to register Ⓑ.	3
Constant transfer (FUN. 001)	$\text{Ⓐ} \quad K \rightarrow W \quad \text{Ⓑ}$	Transfers 16-bit constant Ⓐ to register Ⓑ.	3 / 4
Table initialization (FUN. 002)	$\text{Ⓐ} \quad \text{TINZ [nn]} \quad \text{Ⓑ}$	Transfers data from register Ⓐ to every register in the table, size [nn] starting with register Ⓑ.	4
Multiplexer (FUN. 003)	$\text{Ⓐ} \quad T \rightarrow W \text{ [nn]} \quad \text{Ⓑ} \rightarrow \text{Ⓒ}$	Transfers data from the register specified by Ⓑ in the table, size [nn] starting with register Ⓐ, to register Ⓒ.	5
Demulti-plexer (FUN. 004)	$\text{Ⓐ} \quad W \rightarrow T \text{ [nn]} \quad \text{Ⓑ} \rightarrow \text{Ⓒ}$	Transfers data from register Ⓐ to the register specified by Ⓑ in the table, size [nn] starting with register Ⓒ.	5
Table block transfer (FUN. 005)	$\text{Ⓐ} \quad T \rightarrow T \text{ [nn]} \quad \text{Ⓑ}$	Transfers the data of every register in the table, size [nn] starting with Ⓐ, to the registers after Ⓑ.	4

Arithmetic operations

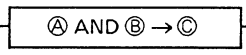
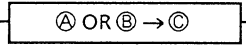
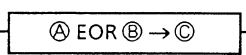
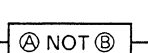
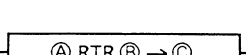
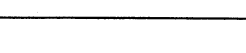
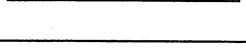
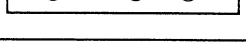
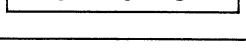
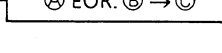
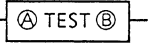
Instruction (FUN. No.)	Expression	Description	No. of steps
Register addition (FUN. 010)	$\text{Ⓐ} + \text{Ⓑ} \rightarrow \text{Ⓒ}$	Adds the data in registers Ⓐ and Ⓑ and stores the result in register Ⓒ.	4
Register subtraction (FUN. 011)	$\text{Ⓐ} - \text{Ⓑ} \rightarrow \text{Ⓒ}$	Subtracts register Ⓑ from register Ⓐ and stores the result in register Ⓒ.	4
Register multiplication (FUN. 012)	$\text{Ⓐ} \times \text{Ⓑ} \rightarrow \text{Ⓒ}$	Multiplies the data in registers Ⓐ and Ⓑ and stores the result in double-length register Ⓒ · Ⓒ + 1.	4
Register division (FUN. 013)	$\text{Ⓐ} / \text{Ⓑ} \rightarrow \text{Ⓒ}$	Divides the data in double-length register Ⓐ · Ⓐ + 1 by register Ⓑ and stores the quotient in Ⓒ and the remainder in Ⓒ + 1.	4
Register comparison (FUN. 014)	$\text{Ⓐ} > \text{Ⓑ}$ — Output	Compares registers Ⓐ and Ⓑ and turns on the output if the result is true.	3
Register comparison (FUN. 015)	$\text{Ⓐ} = \text{Ⓑ}$ — Output	Compares registers Ⓐ and Ⓑ and turns on the output if the result is true.	3
Register comparison (FUN. 016)	$\text{Ⓐ} < \text{Ⓑ}$ — Output	Compares registers Ⓐ and Ⓑ and turns on the output if the result is true.	3

Arithmetic operations (cont'd)

Instruction (FUN. No.)	Expression	Description	No. of steps
Double-length addition (FUN. 017)	$\text{A} \text{ ++ } \text{B} \rightarrow \text{C}$	Adds the data of double-length (32-bit) registers $\text{A} \cdot \text{A} + 1$ and $\text{B} \cdot \text{B} + 1$ and stores the result in $\text{C} \cdot \text{C} + 1$.	4
Double-length subtraction (FUN. 018)	$\text{A} \text{ -- } \text{B} \rightarrow \text{C}$	Subtracts the data of double-length registers $\text{B} \cdot \text{B} + 1$ from $\text{A} \cdot \text{A} + 1$ and stores the result in $\text{C} \cdot \text{C} + 1$.	4
Constant addition (FUN. 020)	$\text{A} \text{ +. } \text{B} \rightarrow \text{C}$	Adds the data in register A and constant B and stores the result in C .	4 / 5
Constant subtraction (FUN. 021)	$\text{A} \text{ -. } \text{B} \rightarrow \text{C}$	Subtracts constant B from the data in register A and stores the result in C .	4 / 5
Constant multiplication (FUN. 022)	$\text{A} \text{ x. } \text{B} \rightarrow \text{C}$	Multiplies constant B with the data in register A and stores the result in double-length register $\text{C} \cdot \text{C} + 1$.	4 / 5
Constant division (FUN. 023)	$\text{A} \text{ /. } \text{B} \rightarrow \text{C}$	Divides the data in double-length register $\text{A} \cdot \text{A} + 1$ by constant B and stores the quotient in C , and the remainder in $\text{C} + 1$.	4 / 5
Constant comparison (FUN. 024)	$\text{A} \text{ >. } \text{B}$ — Output	Compares the data in register A with constant B and turns on the output if the result is true.	3 / 4
Constant comparison (FUN. 025)	$\text{A} \text{ =. } \text{B}$ — Output	Compares the data in register A with constant B and turns on the output if the result is true.	3 / 4
Constant comparison (FUN. 026)	$\text{A} \text{ <. } \text{B}$ — Output	Compares the data in register A with constant B and turns on the output if the result is true.	3 / 4

7. Instruction

Logical operations

Instruction (FUN.No.)	Expression	Description	No. of steps
Register AND (FUN. 030)		ANDs the data of registers \textcircled{A} and \textcircled{B} and stores the result in register \textcircled{C} .	4
Register OR (FUN. 031)		ORs the data of registers \textcircled{A} and \textcircled{B} and stores the result in register \textcircled{C} .	4
Register exclusive OR (FUN. 032)		Exclusive ORs the data in registers \textcircled{A} and \textcircled{B} and stores the result in register \textcircled{C} .	4
Register inversion (FUN. 033)		Inverts each bit of register \textcircled{A} and stores the result in register \textcircled{B} .	3
Right rotation (FUN. 035)		Rotates the bits in register \textcircled{A} to the right by the number specified in register \textcircled{B} and stores the result in register \textcircled{C} .	4
Left rotation (FUN. 036)		Rotates the bits in register \textcircled{A} to the left by the number specified in register \textcircled{B} and stores the result in register \textcircled{C} .	4
Constant AND (FUN. 040)		ANDs the data in register \textcircled{A} and constant \textcircled{B} and stores the result in register \textcircled{C} .	4 / 5
Constant OR (FUN. 041)		ORs the data in register \textcircled{A} and constant \textcircled{B} and stores the result in register \textcircled{C} .	4 / 5
Constant exclusive OR (FUN. 042)		Exclusive ORs the data in register \textcircled{A} and constant \textcircled{B} and stores the result in register \textcircled{C} .	4 / 5
Bit test (FUN. 043)		ANDs the data in register \textcircled{A} and constant \textcircled{B} and turns on the output if the result is other than 0.	3 / 4
Complement (FUN. 046)		Calculates the 2's complement of data in register \textcircled{A} and stores the result in \textcircled{B} .	3

Data conversion instructions

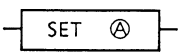
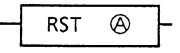
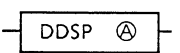
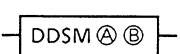
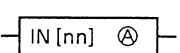
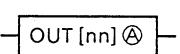
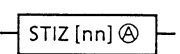
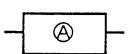
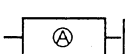








Instruction (FUN. No.)	Expression	Description	No. of steps
Binary conversion (FUN. 050)	$\text{---} \boxed{\text{A} \text{ BIN } \text{B}} \text{---}$	Converts the BCD data in register A into binary, and stores the result in register B.	3
Single-length BCD conversion (FUN. 051)	$\text{---} \boxed{\text{A} \text{ BCD1 } \text{B}} \text{---}$	Converts the binary data in register A into 4-digit BCD code and stores the result in register B.	3
Double-length BCD conversion (FUN. 052)	$\text{---} \boxed{\text{A} \text{ BCD2 } \text{B}} \text{---}$	Converts the binary data in double-length registers A · A + 1 into BCD code and stores the result B · B + 1 · B + 2.	3
Encode (FUN. 053)	$\text{---} \boxed{\text{A} \text{ ENC } \text{B}} \text{---}$	Converts the most significant ON bit position of A into 4-bit data and stores the result in B.	3
Decode (FUN. 054)	$\text{---} \boxed{\text{A} \text{ DEC } \text{B}} \text{---}$	Converts the least significant 4-bit data of A into bit position and stores the result in B.	3
Bit count (FUN. 055)	$\text{---} \boxed{\text{A} \text{ BITC } \text{B}} \text{---}$	Counts the number of bits that are ON in register A and stores the result in register B.	3

7. Instruction

Special functions



Instruction (FUN. No.)	Expression	Description	No. of steps
Upper limit (FUN. 060)	$\text{A UL B} \rightarrow \text{C}$	Sets the data in register A as the upper limit according to register B and stores the result in register C.	4
Lower limit (FUN. 061)	$\text{A LL B} \rightarrow \text{C}$	Sets the data in register A as the lower limit according to register B and stores the result in register C.	4
Maximum value (FUN. 062)	A MAX [nn] B	Locates the maximum value in table size [nn] starting with register A and stores it in register B. Stores the pointer indicating the value in register B + 1.	4
Minimum value (FUN. 063)	A MIN [nn] B	Locates the minimum value in table size [nn] starting with register A and stores it in register B. Stores the pointer indicating the value in register B + 1.	4
Average value (FUN. 064)	A AVE [nn] B	Calculates the average value of table size [nn] starting with register A and stores it in register B.	4
Function generator (FUN. 065)	$\text{A FG [nn] B} \rightarrow \text{C}$	Generates optional functions by registering function parameters.	5
Square root (FUN. 070)	A RT B	Calculates the square root of the data in double-length register A · A + 1 and stores the result in register B.	3
Sine function (FUN. 071)	A SIN B	Divides the data in register A by 100 and obtains its sine. Multiplies the answer by 10000 and stores the result in register B.	3
Arcsine function (FUN. 072)	A ASIN B	Divides the data in register A by 10000 and obtains its arcsine. Multiplies the answer by 100 and stores the result in register B.	3
Cosine function (FUN. 073)	A COS B	Divides the data in register A by 100 and obtains its cosine. Multiplies the answer by 10000 and stores the result in register B.	3
Arccosine function (FUN. 074)	A ACOS B	Divides the data in register A by 10000 and obtains its arccosine. Multiplies the answer by 100 and stores the result in register B.	3

Other functions

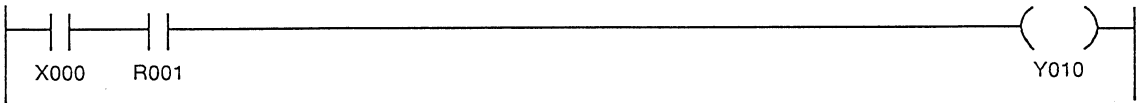
Instruction (FUN. No.)	Expression	Description	No. of steps
Device set (FUN. 080)		Turns on device A.	2
Device reset (FUN. 081)		Turns off device A.	2
Diagnostic display (FUN. 090)		When the input turns ON, the diagnostic code set by A is displayed on a peripheral device.	2 / 3
Diagnostic display with message (FUN. 091)		When the input turns ON, the diagnostic code set by A and the message registered in the registers after B are displayed on a peripheral device.	3 / 4
Immediate input (FUN. 096)		Immediately updates the input data of [nn] registers starting with A.	3
Immediate output (FUN. 097)		Immediately updates the output data of [nn] registers starting with A.	3
Step sequence initialization (FUN. 100)		Initializes the step sequencer beginning with device A.	3
Step sequence input (FUN. 101)		Sets the step sequencer suitable for sequential control.	2
Step sequence output (FUN. 102)			2
Flip-flop (FUN. 110)	Set input  Reset input 	Turns on device A when the set input turns on and turns off device A when the reset input turns on.	2
Up/down counter (FUN. 111)	Up / down Select input  Count input  Enable input 	While the enable input is on, counts up or down the number of cycles the count input turns on, according to the up / down select input. Select input: ON = up, OFF = down	2
Shift register (FUN. 112)	Data input  Shift input  Enable input 	If the shift input is on while the enable input is on, shifts the data in the shift register by one bit. Shift register: [nn] bits of data starting with device A.	3

7. Instructions

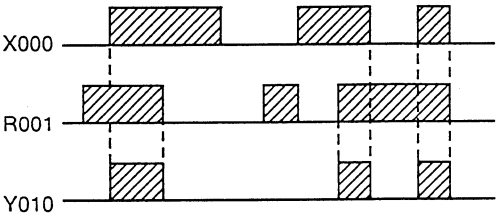
7.2
Basic ladder functions

		NO-Contact															
E x p r e s s i o n	<div><div>Input</div><div>Output</div><div>Ⓐ</div></div>															Number of Steps	
																1	
F u n c t i o n	NO-contact (normally-open contact) of a relay coil corresponding to device number Ⓐ						Input		Operation							Output	
							OFF		Regardless of the content of the device							OFF	
							ON		When the content of the device is OFF							OFF	
									When the content of the device is ON							ON	
O p e r a n d	Symbol	Name	R	X	Y	Z	RW	XW	YW	ZW	D	T	C			Constant	
	Ⓐ	Device number	○	○	○	○											

Sample Program





Operation



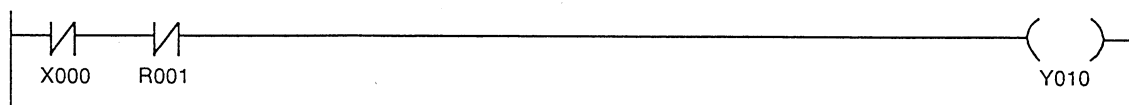
Description

- Coil Y010 turns ON only when the contents of devices X000 and R001 are both ON.

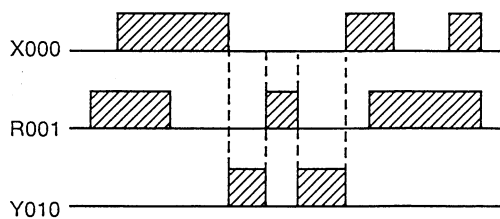
7. Instructions

 NC-Contact															
E x p r e s s i o n	Input  Output Ⓐ	Number of Steps													
		1													
F u n c t i o n	NC-contact (normally-closed contact) of a relay coil corresponding to device number Ⓐ						Input		Operation						Output
							OFF		Regardless of the content of the device						OFF
							ON		When the content of the device is OFF						ON
									When the content of the device is ON						OFF
O p e r a t i o n	Symbol	Name	R	X	Y	Z	RW	XW	YW	ZW	D	T	C		Constant
	Ⓐ	Device number	○	○	○	○									

Sample Program



Operation



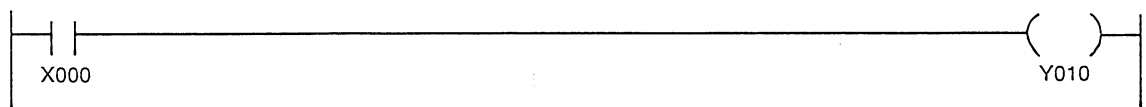
Description

- Coil Y010 turns ON only when the contents of devices X000 and R001 are both OFF.

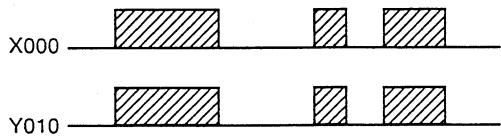
7. Instructions

		Coil													
E x p r e s s i o n	Input														
		Number of Steps													
F u n c t i o n	Relay coil of device number A							Input	Operation						
								OFF	Sets the device to OFF.						
								ON	Sets the device to ON.						
O p e r a n d	Symbol	Name	R	X	Y	Z	RW	XW	YW	ZW	D	T	C		Constant
	A	Device number	○		○	○									

Sample Program



Operation



Description

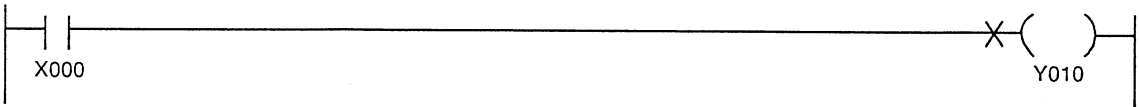
- Y010 is ON when X000 is ON, and it is OFF when X000 is OFF.

Note

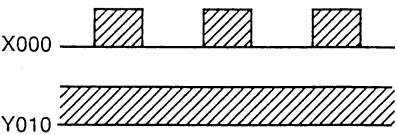
- No other instruction can be written on the right-hand side of a coil.

→X()—		Forced Coil														
E x p r e s s i o n	Input →X()— Ⓐ														Number of Steps	
															1	
F u n c t i o n	Maintains the preceding state regardless of input state.						Input		Operation						Output	
							OFF		Maintains the preceding state							
							ON		Maintains the preceding state							
O p e r a n d	Symbol	Name		R	X	Y	Z	RW	XW	YW	ZW	D	T	C		Constant
	Ⓐ	Device number		○		○	○									

Sample Program



Operation



Description

- A forced coil maintains the preceding state regardless of the input condition.
- In this example, the coil maintains the ON state. If it is then forcibly reset by data setting from the programmer, it maintains the OFF state.

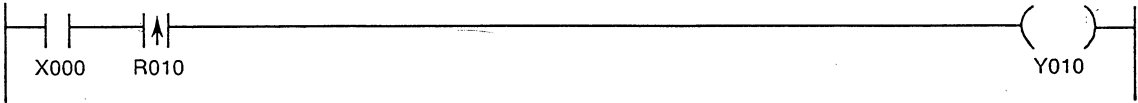
Note

- Forced coil is a debugging function. The state of the forced coil device can be set ON or OFF by the programmer.

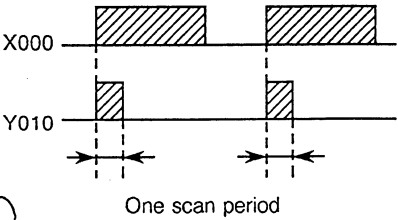
7. Instructions

<div><div></div><div></div></div>		Transitional Contact (Rising Edge)														
Expression	<div>Input<div><div></div><div></div></div>Output</div> <div>Ⓐ</div>															
		Number of Steps														
		1														
Function	Turns output ON for one scan period only when input changes from OFF to ON.	Input		Operation										Output		
		OFF		Regardless of the preceding state.										OFF		
		ON		When input of the preceding scan is OFF										ON		
				When input of the preceding scan is ON										OFF		
Operand	Symbol	Name	R	X	Y	Z	RW	XW	YW	ZW	D	T	C			Constant
	Ⓐ	Device number	○													

Sample Program



Operation


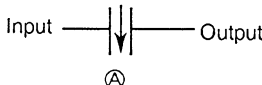


Note

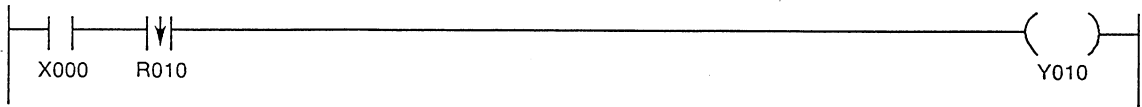
- This instruction creates a pulse signal in response to a change that sets the input ON.
- Device Ⓐ stores the input state of the preceding scan.
- Do not duplicate device Ⓐ.

Description

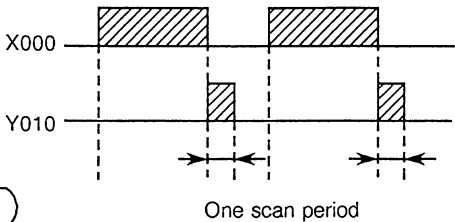
- Output of the transitional contact R010 is turned ON for only one scan period after NO-contact X000 turns ON.

		Transitional Contact (Falling Edge)															
E x p r e s s i o n															Number of Steps		
															1		
F u n c t i o n	Turns output ON for one scan period only when input changes from ON to OFF.							Input		Operation						Output	
								OFF		When input of the preceding scan is OFF						OFF	
										When input of the preceding scan is ON						ON	
								ON		Regardless of the preceding state.						OFF	
O p e r a n d	Symbol	Name	R	X	Y	Z	RW	XW	YW	ZW	D	T	C			Constant	
	Ⓐ	Device number	○														

Sample Program



Operation



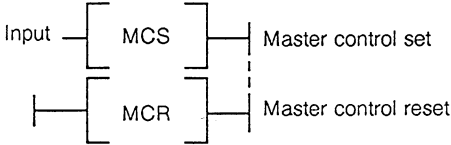
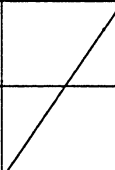
Note

- This instruction creates a pulse signal in response to a change that sets the input from ON to OFF.
- Device Ⓐ stores the input state of the preceding scan.
- Do not duplicate device Ⓐ.

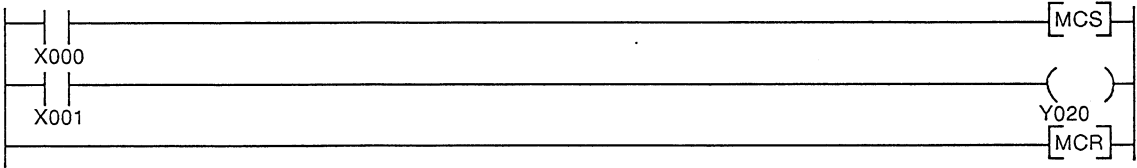
Description

- Output of the transitional contact R010 is turned ON for only one scan period after NO-contact X000 changes from ON to OFF.

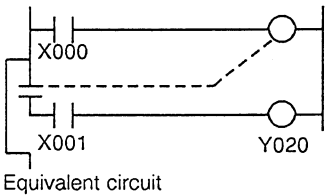
7. Instructions

MCS MCR		Master Control Set / Reset																		
E x p r e s s i o n	<div></div>																Number of Steps			
																	1			
F u n c t i o n	When the MCS input is ON, ordinary operation is performed, when the MCS input is OFF, the left power rail between MCS and MCR goes OFF.							Input		Operation						Output				
								OFF		Sets the left power rail between MCS and MCR to OFF										
								ON		Ordinary operation										
O p e r a n d	Symbol	Name					R	X	Y	Z	RW	XW	YW	ZW	D	T	C			Constant

Sample Program



Operation



Description

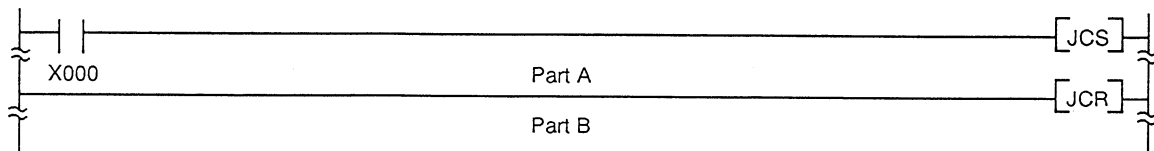
- Ordinary operation is performed when device X000 is ON.
- When the MCS input is OFF, the left power rail between MCS and MCR is OFF and coil Y020 is OFF regardless of the state of device X001.

Note

- MCS and MCR must be used as a pair.
- MCS and MCR cannot be nested. (Two or more MCS instructions cannot be programmed consecutively.)
- No input condition is required for MCR.

JCS JCR		Jump Control Set / Reset															
E x p r e s s i o n	<div><div>Input</div><div><div>JCS</div><div>JCR</div></div><div><div>Jump control set</div><div>Jump control reset</div></div></div>															Number of Steps	
																1	
F u n c t i o n	When the JCS input is ON, instructions between JCS and JCR are skipped at high speed, and execution restarts at the instruction following JCR.						Input		Operation							Output	
							OFF		Non-execution								
							ON		Execution of jump instruction								
O p e r a n d	Symbol	Name		R	X	Y	Z	RW	XW	YW	ZW	D	T	C			Constant

Sample Program



Description

- When NO-contact X000 (input condition) goes to ON, the instructions in part A are not executed and execution restarts at the instructions in part B.
- When NO-contact X000 is OFF, JCS and JCR instructions are ignored, and the instructions in parts A and B are executed.

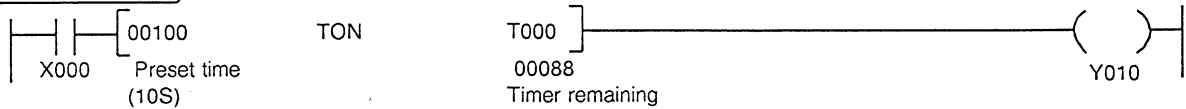
Note

- JCS and JCR must be used as a pair.
- If JCR is located before JCS, an error occurs.
- Two or more JCS instructions cannot be programmed consecutively.
- No input condition is required for JCR.

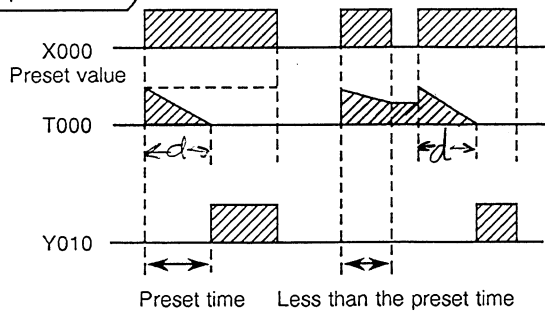
7. Instructions

TON		On-Delay Timer														
E x p r e s s i o n	<div>Input</div> <div><div>Ⓐ</div><div>TON</div><div>Ⓑ</div></div> <div>Timer output</div> <div><div></div><div></div></div> <div>Data displayed</div>														Number of Steps	
															2 / 3	
F u n c t i o n	Output turns ON at the specified time after the input comes ON.							Input	Operation							Output
								OFF	Non execution							OFF
								ON	Time remaining > 0							OFF
O p e r a n d	Symbol	Name	R	X	Y	Z	RW	XW	YW	ZW	D	T	C			Constant
	Ⓐ	Preset time					○	○	○	○	○					0~32767
	Ⓑ	Time remaining										○				

Sample Program



Operation

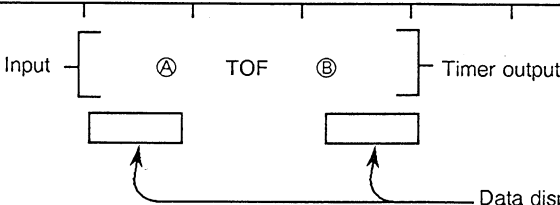


Description

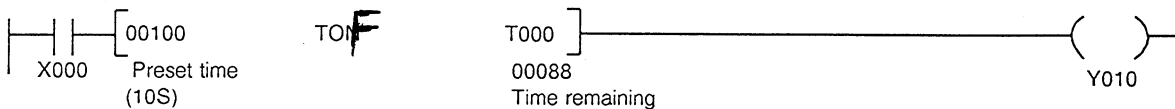
- TON output and relay coil Y010 are turned ON 10 seconds after NO-contact X000 comes ON. The time remaining is stored in T000.
- If X000 goes to OFF in less than the preset time, T000 maintains the current value, and the output remains off.

Note

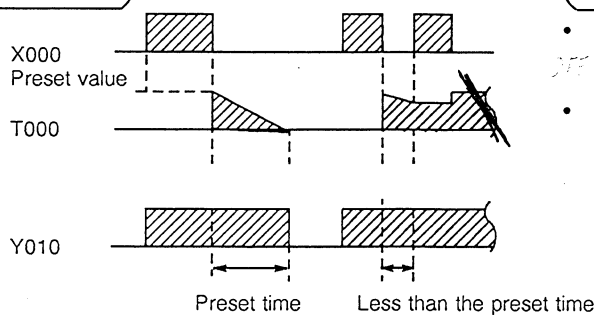
- Time is set in 0.1-second units for T000 to T119 (up to 3276.7 seconds), and in 0.01-second units for T120 to T127 (up to 327.67 seconds).
- After the time remaining reaches 0, it is not updated but remains at 0.
- Either a constant (3 steps) or a register (2 steps) can be used for the preset time Ⓐ

TOF		OFF-Delay Timer															
E x p r e s s i o n																Number of Steps	
																2 / 3	
F u n c t i o n	Sets the output OFF at the specified time after the input goes to OFF.						Input		Operation						Output		
							OFF		Time remaining > 0						ON		
									Time remaining = 0						OFF		
O p e r a n d							ON		Non execution						ON		
	Symbol	Name	R	X	Y	Z	RW	XW	YW	ZW	D	T	C			Constant	
	Ⓐ	Preset time					○	○	○	○	○					0~32767	
	Ⓑ	Time remaining										○					

Sample Program



Operation



Description

- TON output and relay coil Y010 are turned ON 10 seconds after NO-contact X000 goes OFF.
- When NO-contact X000 changes from ON to OFF, the preset time is loaded to T000 and the timer countdown is started. If X000 turns ON again in less than the preset time, the output remains on.

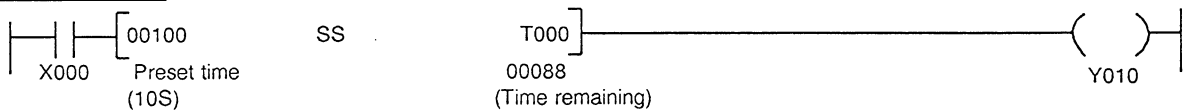
Note

- Time is set in 0.1-second units for T000 to T119 (up to 3276.7 seconds), and in 0.01-second units for T120 to T127 (up to 327.67 seconds).
- Either a constant (3 steps) or a register (2 steps) can be used for the preset time Ⓐ

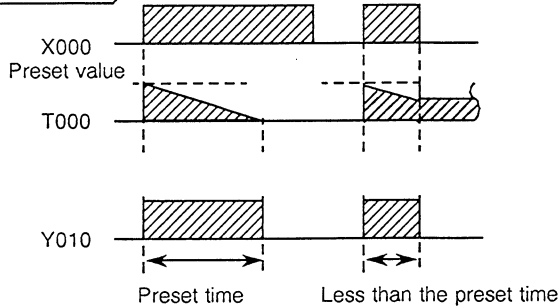
7. Instructions

SS		Single Shot													
Expression	<div>Input<div><div>Ⓐ</div><div>SS</div><div>Ⓑ</div></div>Timer output</div>													Number of Steps	
														2 / 3	
Function	Keeps output ON for the specified time after the input goes to ON.	Input		Operation										Output	
		OFF		Non execution										OFF	
		ON		Time remaining > 0										ON	
				Time remaining = 0										OFF	
Operand	Symbol	Name	R	X	Y	Z	RW	XW	YW	ZW	D	T	C		Constant
	Ⓐ	Preset time					○	○	○	○	○				0~32767
	Ⓑ	Time remaining										○			

Sample Program



Operation



Description

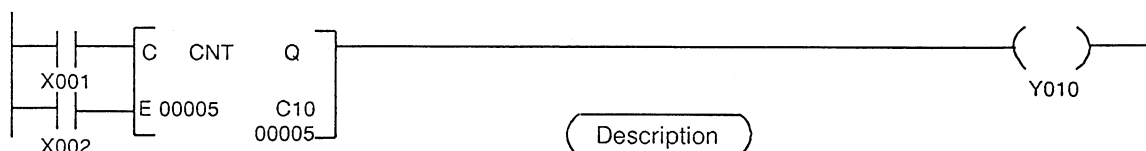
- SS output and relay coil Y010 remain ON 10 seconds after NO-contact X000 goes ON. The time remaining is stored in T000.
- If NO-contact X000 turns OFF within the preset time, the output goes OFF.

Note

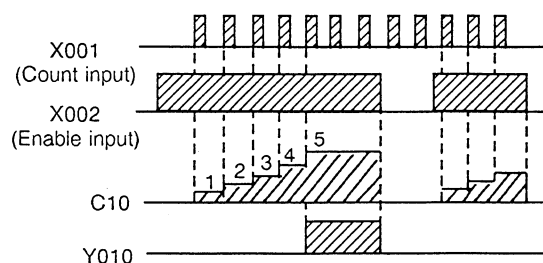
- Time is set in 0.1-second units for T000 to T119 (up to 3276.7 seconds) and in 0.01-second units for T120 to T127 (up to 327.67 seconds).
- After the time remaining reaches 0, it is not updated but remains at 0.
- Either a constant (3 steps) or a register (2 steps) can be used for the preset time Ⓐ.

CNT		Counter														
E x p r e s s i o n	<div><div>Counter input</div><div>Enable input</div><div><div>C</div><div>E</div><div><div>CNT</div><div>Q</div></div><div>Counter output</div></div><div><div>Ⓐ</div><div>Ⓑ</div></div><div>Data displayed</div></div>													Number of Steps		
														2 / 3		
F u n c t i o n	Counts the number of input changes from OFF to ON while the enable input is ON, and sets its output ON when the count Ⓑ reaches the set value Ⓐ.	Enable Input		Operation										Output		
		OFF		No execution, (count) ← 0										OFF		
		ON		Count < set value										OFF		
O p e r a n d			Count = set value										ON			
	Symbol	Name	R	X	Y	Z	RW	XW	YW	ZW	D	T	C			Constant
	Ⓐ	Preset value					○	○	○	○	○					0~65535
	Ⓑ	Count											○			

Sample Program



Operation



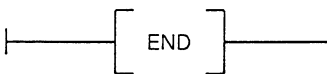
Description

- The counter counts changes of NO-contact X001 (count input) from OFF to ON only while NO-contact X002 (enable input) is ON. The count is stored in C10. When the count reaches the preset value (00005), the counter output is turned ON.
- When the enable input, X002, turns OFF, the count value is reset to zero and the counter output is turned OFF.

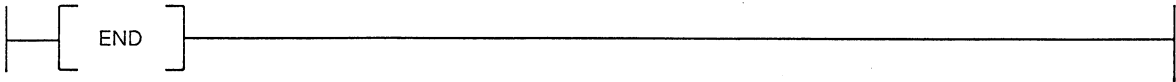
Note

- No pulse contact has to be provided before the counter instruction. (The CNT instruction itself detects an input rise from OFF to ON.)
- Cascade connection of counters enables counting beyond the maximum preset value.
- Either a constant (3 steps) or a register (2 steps) can be used for the preset value.

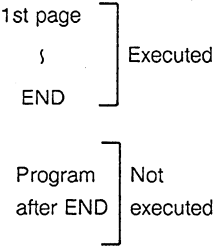
7. Instructions

END		End																
E x p r e s s i o n															Number of Steps			
															1			
F u n c t i o n	Indicates the end of the program.							Condition Input		Operation							Output	
O p e r a n d	Symbol	Name		R	X	Y	Z	RW	XW	YW	ZW	D	T	C			Constant	

Sample Program



Operation



Description

- A program is executed from the first page to the END instruction.
- Program steps can be written after the END instruction, but they are not executed. Those steps are, however, counted in the total number of steps.

Note

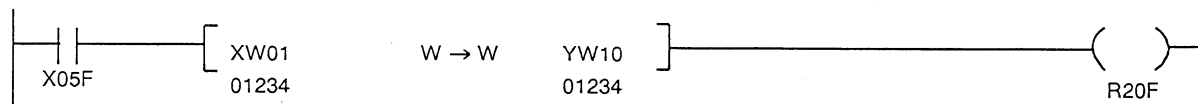
- At least one END instruction must be written in a program. (Two or more END instructions may be written in one program.)

7.3

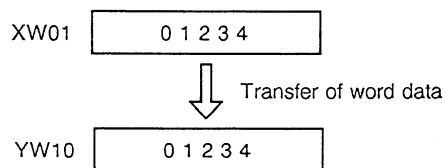
Data transfer instructions

W → W		Register Transfer (FUN000)													
E x p r e s s i o n		Input [Ⓐ] W → W Ⓑ Execution output Data displayed												Number of Steps	
														3	
F u n c t i o n	Stores the register Ⓐ data in register Ⓑ.						Input		Operation						Output
							OFF		No execution						OFF
							ON		Execution						ON
O p e r a n d	Symbol	Name	R	X	Y	Z	RW	XW	YW	ZW	D	T	C		Constant
	Ⓐ	Source register					○	○	○	○	○	○	○		
	Ⓑ	Destination register					○		○	○	○				

Sample Program



Operation



Description

- This instruction stores the contents of register XW01 (01234) in register YW10 and sets its output ON when NO-contact X05F is ON.
- The instruction executes no storage operation and sets its output OFF when NO-contact X05F is OFF.

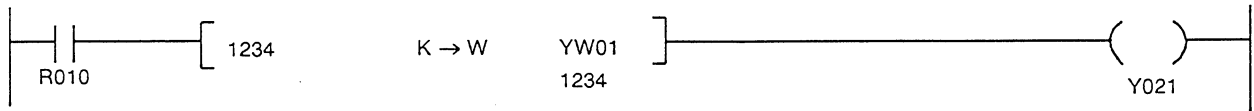
Note

- No constant can be used for the source register Ⓐ.

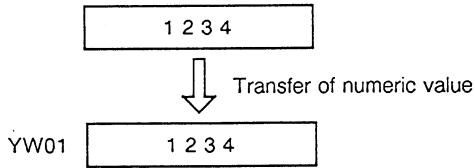
7. Instructions

K → W		Constant Transfer (FUN001)													Number of Steps	
E x p r e s s i o n	Input	Execution output													3 / 4	
		Data displayed														
F u n c t i o n	Stores numeric value ① in register ②.						Input		Operation						Output	
							OFF		No execution						OFF	
							ON		Execution						ON	
O p e r a n d	Symbol	Name	R	X	Y	Z	RW	XW	YW	ZW	D	T	C		Constant	
	①	Source data													0~65535	
	②	Destination register					○		○	○	○					

Sample Program



Operation



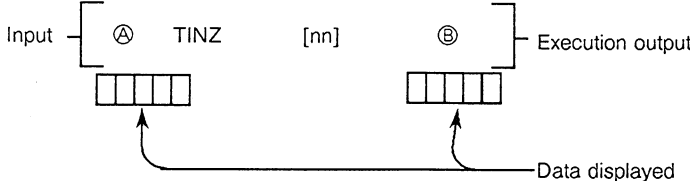
Description

- This instruction stores the constant value 01234 in register YW01 and sets its output ON when NO-contact R010 is ON.
- The instruction does not execute and its output is OFF when NO-contact R010 is OFF.

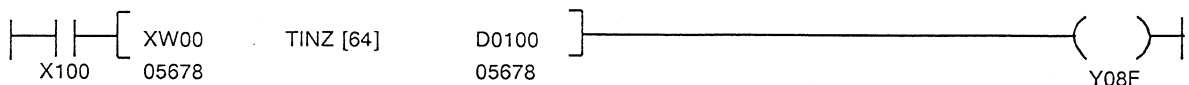
Note

- The number of steps used for this instruction depends on the constant value:
① < 256: 3 steps
① ≥ 256: 4 steps

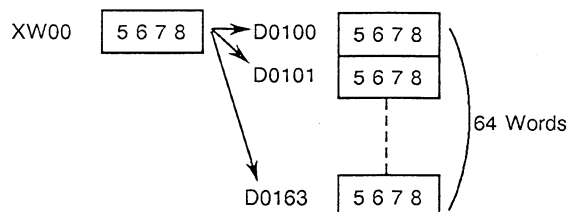
7. Instructions

TINZ		Table Initialization (FUN002)												Number of Steps	
E x p r e s s i o n	Input													4	
F u n c t i o n	Initializes all contents of a table of size [nn] with register ② at its top, using the contents of register ①.						Input		Operation						Output
							OFF		No execution						OFF
							ON		Execution						ON
O p e r a n d	Symbol	Name	R	X	Y	Z	RW	XW	YW	ZW	D	T	C		Constant
	①	Initialization register					○	○	○	○	○	○	○		
	②	Table top register					○		○	○	○				
	nn	Table size													1 ~ 64

Sample Program



Operation



Description

- This instruction transfers the contents of register XW00 (5678) to every register in a table of size 64 with register D0100 at its top and sets the output ON when NO-contact X100 is ON.
- The instruction does not execute any transfer operation, and it sets its output OFF when NO-contact X100 is OFF.

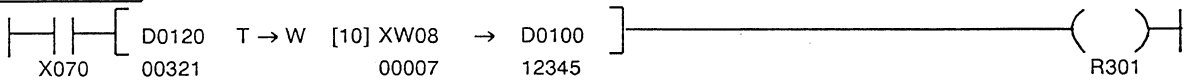
Note

- Table size is up to 64 words (registers) ($1 \leq nn \leq 64$)
- A constant cannot be used for the operand ①.

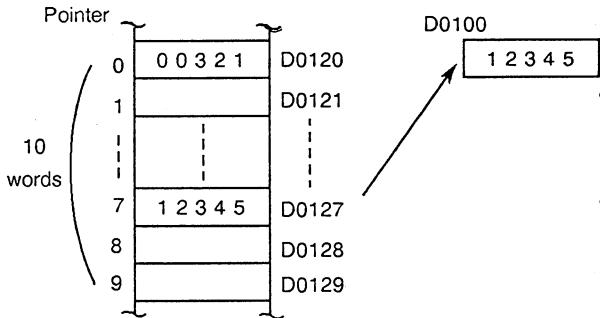
7. Instructions

T → W		Multiplexer (FUN003)												Number of Steps		
E x p r e s s i o n	<div>Input [Ⓐ T → W [nn] Ⓑ → Ⓒ] Outside table frame output</div> <div><div><div></div><div></div><div></div><div></div><div></div></div><div><div></div><div></div><div></div><div></div><div></div></div><div><div></div><div></div><div></div><div></div><div></div></div></div> <div>Data displayed</div>														5	
	F u n c t i o n	Stores the contents of the register specified by Ⓑ in a table of size [nn] with register Ⓐ at the top, into register Ⓒ.							Input	Operation						
OFF									No execution							OFF
ON									Normal execution							OFF
	Outside table frame							ON								
O p e r a n d	Symbol	Name	R	X	Y	Z	RW	XW	YW	ZW	D	T	C			Constant
	Ⓐ	Table top register					<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>			
	Ⓑ	Table pointer					<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>			
	Ⓒ	Transfer destination register					<input type="radio"/>		<input type="radio"/>	<input type="radio"/>	<input type="radio"/>					
	nn	Table size														1 ~ 64

Sample Program



Description



Description

- This instruction stores the contents of register D0127, which is the seventh register (because the content of pointer XW08 is 7) in a table 10 words in size with register D0120 at its top, into register D0100 when NO-contact X070 is ON. It then sets the output OFF because data transfer is executed normally.
- When the contents of XW08 indicate a register outside the table (the contents of XW08 are greater than 9 in the above program sample), the instruction does not execute data transfer but sets the output ON.
- The instruction does not execute any data transfer and sets its output OFF when NO-contact X070 is OFF.

Note

- Table size is up to 64 words (registers) (1 ≤ nn ≤ 64).
- Registers in a table are counted with the top register as 0.
- A table must be within the effective range of register addresses.

7. Instructions

T → W		Multiplexer (FUN003) [EEPROM write instruction (register→EEPROM)]												
Expression											Number of Steps		5	
<div><div>Input</div><div><div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div></div></div></div>														

7. Instructions

Description The EEPROM write instruction is a special mode of FUN003 used to transfer the data of registers directly to the EEPROM. By using this instruction, the variable data can be stored in the EEPROM. The operation mode of FUN003 is selected by changing bits E and F of register ③. (In this instruction, register ③ should be H4000)

Operation When the input comes ON, the data in the source table are transferred to the destination table in the EEPROM (and the RAM).

Table size (Number of data to be transferred):

Specified by nn or register ③ + 1 (Valid range is 1 to 16)

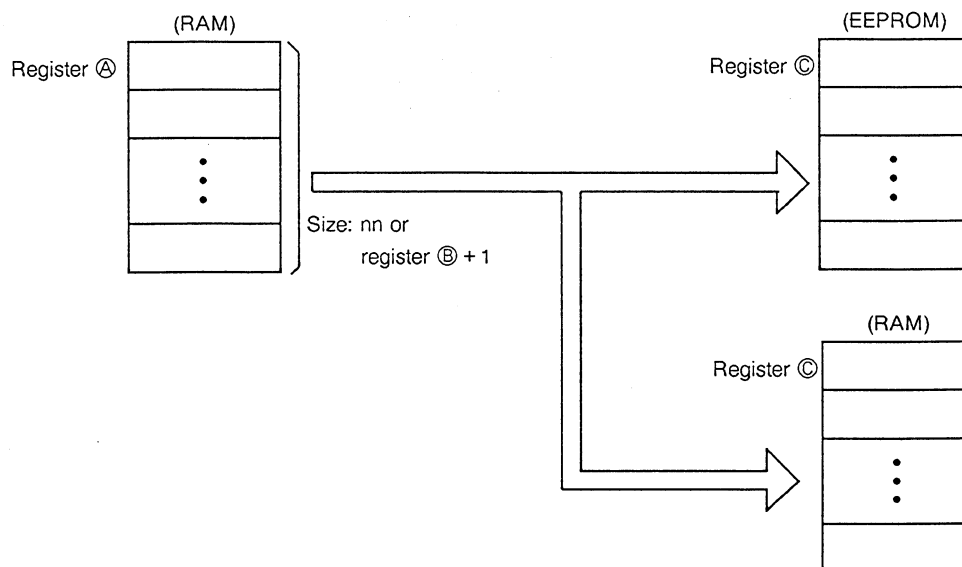
Source table:

Starts with register ①

Destination table:

Starts with register ④ in the EEPROM and the RAM

($D0512 + 16 \times n$ ($n = 0, 1, \dots, 63$)) are valid as register ④)



(1) This instruction is valid only when the EX100 memory setting is 3K mode.

(2) This instruction functions by combining the transitional contact of R62A and FUN003.

Error condition

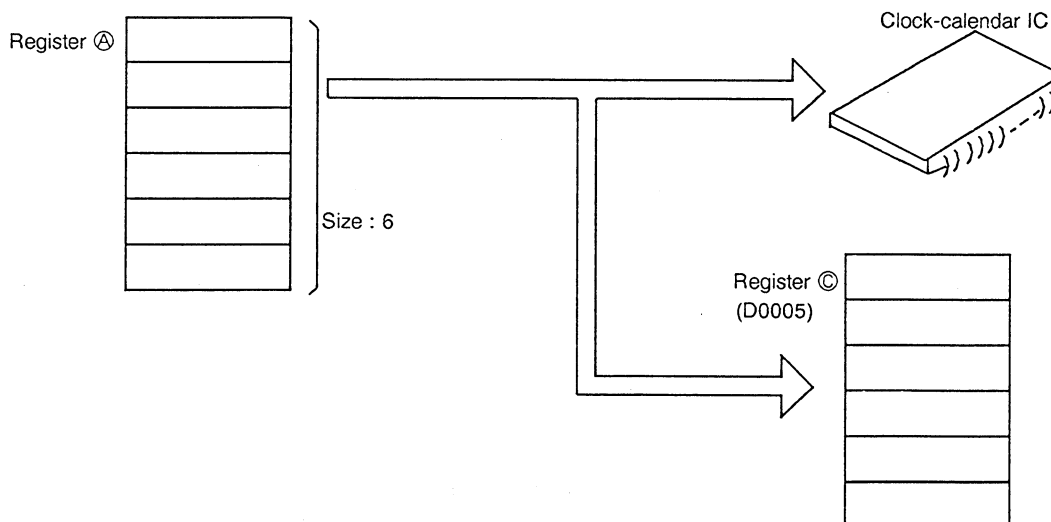
Item	Operation	Error output
Normal operation	Execution	OFF
Register size overflow error in register ④ (Size of register ④ + (nn, or register ⑤ + 1) exceeded the register area.)	No execution	ON
Transfer data number error Excluding nn = 64 (The data of nn or register ⑤ + 1 is 0 or exceeds 16.)	No execution	ON
Register size overflow error in register ⑤ + 1 (Register ⑤ is the highest number register. (nn = 64))	No execution	ON
Setting error of register ⑥ (The number of register ⑥ is not $D512 + 16 \times n$. (n = 0 to 63))	No execution	ON
Memory setting error (4K) (The memory setting of EX100 is not 3K-step mode.)	No execution	ON
Write-protect error (The position of the key switch is set to RUN-P.)	No execution	ON
EEPROM write error (Writing to the EEPROM did not complete normally.) (EEPROM data is undefined)	Execution	ON

7. Instructions

T → W		<div>Multiplexer (FUN003) [Clock-calendar data setting instruction]</div>																									
Expression		<div>Input → <div>Ⓐ T → W [nn] Ⓑ → Ⓒ</div> — Error output</div> <div><div><div></div><div></div><div></div><div></div><div></div><div></div></div><div><div></div><div></div><div></div><div></div><div></div><div></div></div><div><div></div><div></div><div></div><div></div><div></div><div></div></div></div> <div>— Data displayed</div>										Number of Steps	5														
Function		<table><tr><td rowspan="3">Sets the initial values into the clock-calendar data.</td><td>Input</td><td colspan="2">Operation</td><td>Output</td></tr><tr><td>OFF</td><td colspan="2">No execution</td><td>OFF</td></tr><tr><td rowspan="2">ON</td><td>Normal execution</td><td>OFF</td></tr><tr><td>Error</td><td>ON</td></tr></table>												Sets the initial values into the clock-calendar data.	Input	Operation		Output	OFF	No execution		OFF	ON	Normal execution	OFF	Error	ON
Sets the initial values into the clock-calendar data.	Input	Operation		Output																							
	OFF	No execution		OFF																							
	ON	Normal execution	OFF																								
Error		ON																									
Operand																											
<div>Ⓐ: The top register of the source table.</div> <div>Ⓑ: Operation type selecting register for FUN003.</div> <div><div><div>F</div><div>E</div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div></div></div>																											

Description The clock-calendar data setting instruction is a special mode of FUN003 used to set the initial data into the clock-calendar IC. The operation mode of FUN003 is selected by changing bits E and F of register ③. (In this instruction, register ③ should be H4000)

Operation When the input comes ON, the data of 6 registers, starting with register ①, are transferred to the clock-calendar IC as well as to the clock-calendar registers (D0005 to D0010).

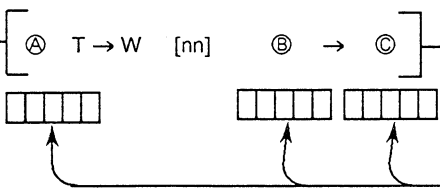
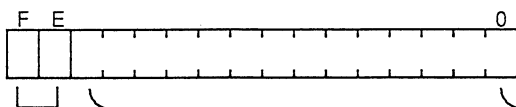


NOTE This instruction is valid only when the clock-calendar function of the PU12A (enhanced type CPU) is used.

Error condition

Item	Operation	Error output
Normal operation	Execution	OFF
Register size overflow error in register ① (Register ① + 6 exceeded the register area.)	No execution	ON
Transfer data number error (The data of nn is not 6.)	No execution	ON
Register setting error in register ④ (Register ④ is not D0005.)	No execution	ON
Calendar unmounted error (The CPU module of the EX100 is not PU12A.)	No execution	ON
Calendar function is not used.	No execution	ON

7. Instructions

T → W		Multiplexer (FUN003) [Data output instruction for special modules]												
Expression		<div>Input — [Ⓐ T → W [nn] Ⓑ → Ⓒ] — Error output</div> <div></div>											Number of Steps	5
Function		Transfers the data of every register in the table starting with Ⓐ to the special module allocated to Ⓒ.												
		Input	Operation								Output			
		OFF	No execution								OFF			
		ON	Normal execution								OFF			
			Error								ON			
Operand		<div>Ⓐ: The top register of the source table.</div> <div>Ⓑ: Operation type selector and destination address.</div> <div></div> <div>The top address of destination in the special module</div> <div>00: Multiplexer instruction (Normal mode)</div> <div>01: EEPROM write instruction / Calendar data setting</div> <div>10: <u>Data output instruction for special modules</u></div> <div>11: Reserved (Do not use)</div> <div>Ⓒ: Output register assigned to the special module (YW only)</div> <div>nn: Number of data to be transferred—use either of the following two methods</div> <div>(1) nn: Specifies the number of data by constant (1 to 63)</div> <div>(2) Register Ⓑ + 1 : If nn is 64, the data in the register next to register Ⓑ specifies the number of data (1 to 128)</div>												
Symbol	Name	R	X	Y	Z	RW	XW	YW	ZW	D	T	C	Constant	
nn	Number of data to be transferred												1 ~ 64	
Ⓐ	The top register of the source table					○	○	○	○	○	○	○		
Ⓑ	Operation type					○	○	○	○	○		○		
Ⓒ	The destination I / O register							○						

Description The data output instruction for the special module is a special mode of FUN003 used to transfer the data of registers to special modules such as the motion control module.
The operation mode of FUN003 is selected by changing bits E and F of register ⑥.

Operation When the input comes ON, the data in the source table are transferred to the destination table in the special module.

Table size (Number of data to be transferred):

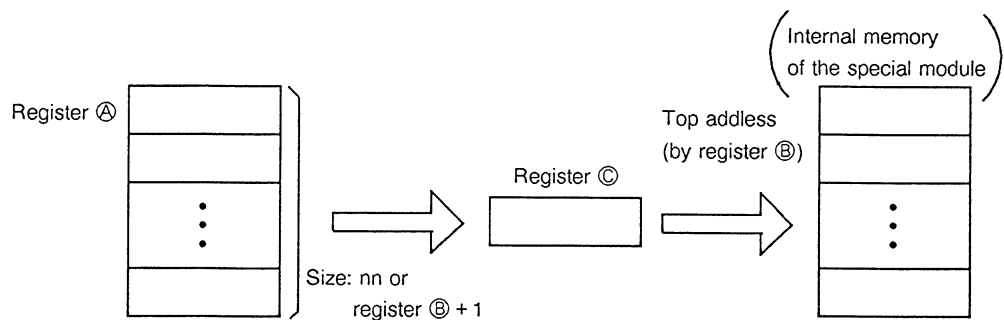
Specified by nn or register ⑥ + 1 (Valid range is 1 to 128)

Source table:

Starts with register ④

Destination table:

Starts with the address specified by bits 0 to D of register ⑥, in the internal memory of the special module that is allocated to register ③.



(1) The data of register ③ will not be changed by executing this instruction.

(2) If the computer link is used at a transmission rate of 9600 bps, the table size is limited to 64 registers or less.

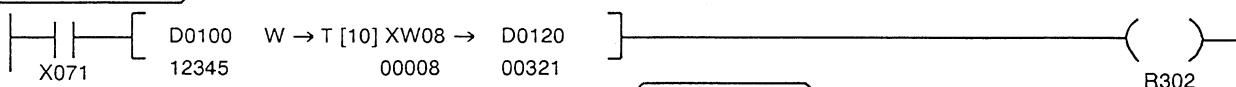
7. Instructions

Error condition

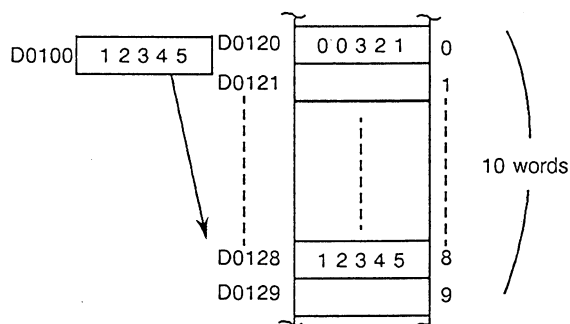
Item	Operation	Error output
Normal operation	Execution	OFF
Register size overflow in register ④ (Register ④ + (nn, or register ⑤ + 1) exceeded the register area.)	No execution	ON
Transfer data number error (The data of nn or register ⑤ + 1 is 0 or exceeds 128.)	No execution	ON
Overflow of the register area of Register ⑤ + 1 (when nn = 64) (Register ⑤ is the highest number register.)	No execution	ON
Internal memory area error in the special I / O module (The value of bits 0 to D of register ⑥ is not in the range of 0 to 350.)	No execution	ON
Internal memory area error in the special I / O module (The value of bits 0 to D of register ⑥ + (nn or register ⑥ + 1) exceeded 350.)	No execution	ON
Output register error (Register ⑦ is not YW.)	No execution	ON
Special I / O module error (Transmission destination I / O is not the special module.)	No execution	ON
I / O request error (The special I / O module is not operative.)	No execution	ON
I / O dismantled error (The special I / O module is not mounted → ERROR mode)	No execution	ON
I / O response error (The operation of the special I / O module is abnormal → ERROR mode)	No execution	ON
I / O parity error (The I / O bus is abnormal → ERROR mode)	No execution	ON

W → T		Demultiplexer (FUN004)												Number of Steps	
E x p r e s s i o n	Input	<div> <div>Ⓐ</div> <div>W → T [nn]</div> <div>Ⓑ</div> <div>→</div> <div>Ⓒ</div> <div>Outside table frame output</div> </div>												5	
		<div> <div>Ⓐ</div> <div>Ⓑ</div> <div>Ⓒ</div> <div>Data displayed</div> </div>													
F u n c t i o n	Stores the contents of register Ⓐ in the register specified by Ⓑ in a table of size [nn] with register Ⓒ at its top.						Input		Operation					Output	
							OFF		No execution					OFF	
							ON		Normal execution					OFF	
O p e r a t i o n									Outside table frame					ON	
	Symbol	Name	R	X	Y	Z	RW	XW	YW	ZW	D	T	C		Constant
	Ⓐ	Source register					○	○	○	○	○	○	○		
	Ⓑ	Table pointer					○	○	○	○	○	○	○		
	Ⓒ	Table top register					○		○	○	○				
	nn	Table size													1 ~ 64

Sample Program



Operation



Description

- This instruction stores the contents of register D0100 (12345) in register D0128, which is the eighth register (because the content of pointer XW08 is 8) in a table 10 words in size with register D0120 at its top, when NO-contact X071 is ON. It sets the output OFF.
- When the contents of pointer register XW08 indicate a register outside the table (the contents of XW08 are greater than 9 in the above program sample), the instruction does not execute data transfer but sets the output ON.
- The instruction does not transfer any data and sets its output OFF when NO-contact X071 is OFF.

Note

- Table size can be up to 64 words (registers) ($1 \leq nn \leq 64$).
- Registers in a table are counted with the top register as 0.
- A table into which data is to be transferred must be within the effective range of register addresses.

7. Instructions

W → T		Demultiplexer (FUN004) [EEPROM read instruction (EEPROM→ Register)]																																																																																		
Expression											Number of Steps		5																																																																							
<div><div>Input</div><div>Ⓐ W → T [nn]</div><div>Ⓑ → Ⓒ</div><div>Error output</div><div>Data displayed</div></div>																																																																																				
Function											Input	Operation		Output																																																																						
Transfers the data stored in the EEPROM to the registers in the RAM.											OFF	No execution		OFF																																																																						
											ON	Normal execution		OFF																																																																						
												Error		ON																																																																						
Operand																																																																																				
<div><div>Ⓐ: The top register of the source table in the EEPROM (D0512 to D1535)</div><div>Ⓑ: Operation type selecting register for FUN004</div><div><div><div>F E</div><div>Not used by this instruction</div><div>0</div></div><div><div>00: Demultiplexer instruction (Normal mode)</div><div>01: <u>EEPROM read instruction</u></div><div>10: Data input instruction for special modules</div><div>11: Reserved (Do not use)</div></div></div><div>(Set H4000 for the value of register Ⓑ.)</div></div>																																																																																				
<div><div>Ⓒ: The top register of the destination table (in the RAM)</div><div>nn: Number of data to be transferred — use either of the following two methods</div><div><div>(1) nn: Specifies the number of data by constant (1 to 16)</div><div>(2) Register Ⓑ + 1: If nn is 64, the data in the register next to register Ⓑ specifies the number of data (1 to 16)</div></div></div>																																																																																				
<div><div>NOTE</div><div>If nn is 64, the maximum register number of Ⓐ is D1472)</div></div>																																																																																				
<table><tr><td>Symbol</td><td>Name</td><td>R</td><td>X</td><td>Y</td><td>Z</td><td>RW</td><td>XW</td><td>YW</td><td>ZW</td><td>D</td><td>T</td><td>C</td><td>Constant</td></tr><tr><td>nn</td><td>Number of data to be transferred</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td>1 ~ 16 (64)</td></tr><tr><td>Ⓐ</td><td>The top register of the source table in the EEPROM</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td>D512~ D1535</td><td></td><td></td><td></td></tr><tr><td>Ⓑ</td><td>Operation type</td><td></td><td></td><td></td><td></td><td>○</td><td>○</td><td>○</td><td>○</td><td>○</td><td>○</td><td>○</td><td></td></tr><tr><td>Ⓒ</td><td>The top register of the destination table</td><td></td><td></td><td></td><td></td><td>○</td><td></td><td>○</td><td>○</td><td>○</td><td></td><td></td><td></td></tr></table>															Symbol	Name	R	X	Y	Z	RW	XW	YW	ZW	D	T	C	Constant	nn	Number of data to be transferred												1 ~ 16 (64)	Ⓐ	The top register of the source table in the EEPROM									D512~ D1535				Ⓑ	Operation type					○	○	○	○	○	○	○		Ⓒ	The top register of the destination table					○		○	○	○			
Symbol	Name	R	X	Y	Z	RW	XW	YW	ZW	D	T	C	Constant																																																																							
nn	Number of data to be transferred												1 ~ 16 (64)																																																																							
Ⓐ	The top register of the source table in the EEPROM									D512~ D1535																																																																										
Ⓑ	Operation type					○	○	○	○	○	○	○																																																																								
Ⓒ	The top register of the destination table					○		○	○	○																																																																										

Description The EEPROM read instruction is a special mode of FUN004 used to transfer data stored in the EEPROM to registers in the RAM. The mode is selected by changing bits E and F of register ③. (In this instruction, register ③ should be H4000)

Operation When the input comes ON, the data in the source table in the EEPROM are transferred to the destination table in the RAM.

Table size (Number of data to be transferred):

Specified by nn or register ③ + 1 (Valid range is 1 to 16)

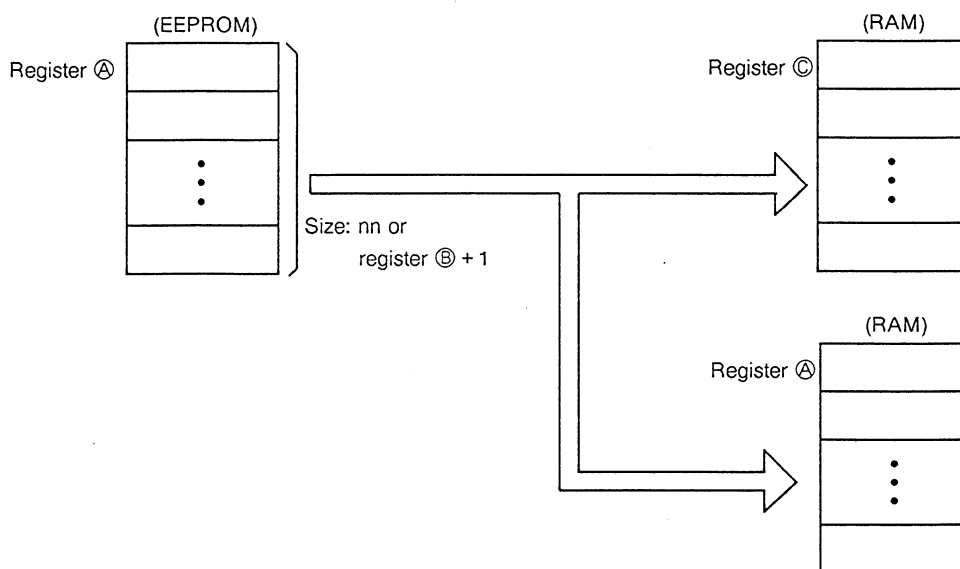
Source table:

Starts with register ① in the EEPROM

(D0512 to D1535 are valid as register ①)

Destination table:

Starts with register ④ and register ① in the RAM.



This instruction is valid only when the EX100 memory setting is 3K mode.

7. Instructions

Error condition

Item	Operation	Error output
Normal operation	Execution	OFF
Register size overflow error in register ④ (Size of register ④ + (nn, or register ③ + 1) exceeded the register area.)	No Execution	ON
Transfer data number error Excluding nn = 64 (The data of nn or register ③ + 1 is 0 or greater than 16.)	No execution	ON
Register size overflow in register ③ + 1 (Register ③ is the highest number register. (nn = 64))	No execution	ON
Register size overflow in register ⑤ (Size of register ⑤ + (nn, or register ③ + 1) exceeded the register area.)	No execution	ON
Memory setting error (4K) (The memory setting of EX100 is not 3K mode.)	No execution	ON

W → T		<div>Demultiplexer (FUN004)</div> <div>[Data input instruction for special modules]</div>														
Expression		<div><div>Input</div><div><div>Ⓐ W → T [nn]</div><div>Ⓑ → Ⓒ</div></div><div>Error output</div><div><div><div></div><div></div><div></div><div></div><div></div></div><div><div></div><div></div><div></div><div></div><div></div></div><div><div></div><div></div><div></div><div></div><div></div></div></div><div>Data displayed</div></div>										Number of Steps		5		
Function		Inputs the data from the special module allocated to Ⓐ into registers starting with Ⓒ										Input	Operation		Output	
												OFF	No execution		OFF	
												ON	Normal execution		OFF	
													Error		ON	
Operand		<div>Ⓐ: The input register assigned to the special module (XW only)</div> <div>Ⓑ: Operation type selector and source address</div> <div><div><div>F</div><div>E</div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div></div></div>														

7. Instructions

Description The data input instruction for the special module is a special mode of FUN004 used to input data from special modules such as the motion control module into the registers.
The operation mode of FUN004 is selected by changing bits E and F of register ③.

Operation When the input comes ON, the data in the source table in the special module are transferred to the destination table.

Table size (Number of data to be transferred):

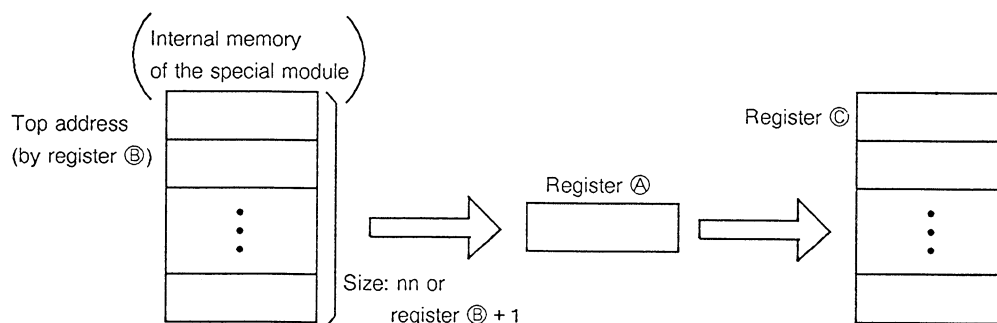
Specified by nn or register ③ + 1 (Valid range is 1 to 128)

Source table:

Starts with the address specified by bits 0 to D of register ③, in the internal memory of the special module allocated to register ④

Destination table:

Starts with register ⑤



(1) The data of register ④ will not be changed by executing this instruction.

(2) If the computer link is used at a transmission rate of 9600 bps, the table size is limited to 64 registers or less.

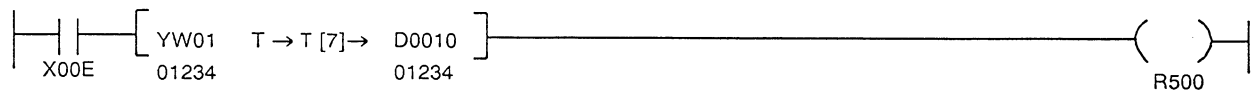
Error condition

Item	Operation	Error output
Normal operation	Execution	OFF
Input register error (Register ④ is not XW.)	No execution	ON
Transfer data number error (The data / content of nn or register ④ + 1 is 0 or exceeds 128.)	No execution	ON
Register size overflow error in Register ④ + 1 (when nn = 64) (Register ④ is the highest number register.)	No execution	ON
Special I / O internal memory area error (The value of bits 0 to D of register ④ is not in the range of 0 to 350.)	No execution	ON
Size-over error in the internal memory of the special I / O module (The value of bits 0 to D of register ④ + (nn or register ④ + 1) exceeded 350.)	No execution	ON
Register size overflow error in register ④ (Register ④ + (nn, or register ④ + 1) exceeded the register area.)	No execution	ON
Special I / O module error (Input destination I / O is not the special module.)	No execution	ON
I / O request error (The special I / O module is not operative.)	No execution	ON
I / O modules dismantled (The special I / O module is not mounted → ERROR mode.)	No execution	ON
I / O response error (The operation of the special I / O module is abnormal → ERROR mode.)	No execution	ON
I / O parity error (The I / O bus is abnormal → ERROR mode.)	No execution	ON

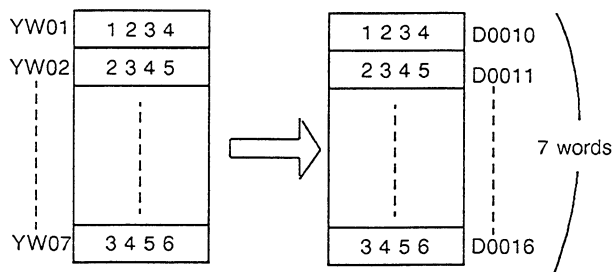
7. Instructions

T → T		Table Transfer (FUN005)													
E X P R E S S I O N	Input	T → T [nn] Execution output													
		Data displayed													
F U N C T I O N	Transfers a block of data in a table of size [nn] with register Ⓐ at the top, into registers beginning with register Ⓑ.														Number of Steps
															4
O P E R A N D	Symbol	Name	R	X	Y	Z	RW	XW	YW	ZW	D	T	C		Constant
	Ⓐ	Top register of source table					○	○	○	○	○	○	○		
	Ⓑ	Top register of destination table					○		○	○	○				
	nn	Table size													1 ~ 64

Sample Program



Operation



Description

- This instruction transfers a block of the contents of registers of 7 words, beginning with YW01, into registers beginning with register D0010, and sets its output ON when NO-contact X00E is ON.
- The instruction does not execute any transfer and sets its output OFF when NO-contact X00E is OFF.

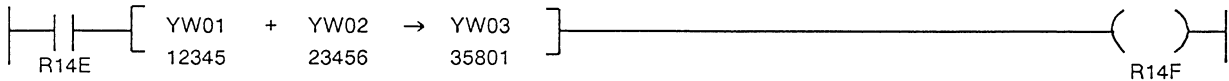
Note

- Table size is up to 64 words (registers) ($1 \leq nn \leq 64$).
- Source and destination tables must be within the effective range of register addresses.
- Source and destination registers can be of the same type or can overlap each other.

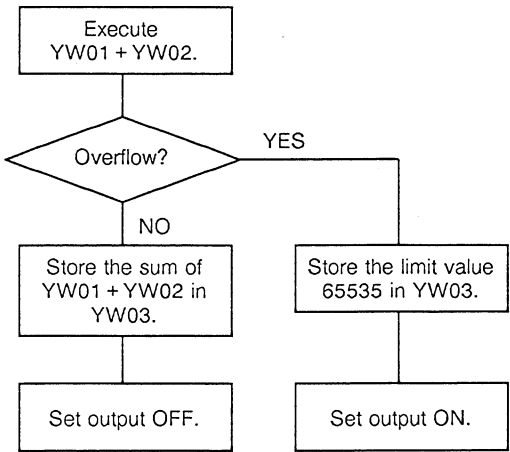
7.4
Arithmetic operations

+		Register Addition (FUN010)														
Expression	<div>Input <div><div>Ⓐ</div><div>+</div><div>Ⓑ</div><div>→</div><div>Ⓒ</div></div> Overflow output</div> <div><div><div></div><div></div><div></div><div></div><div></div></div><div><div></div><div></div><div></div><div></div><div></div></div><div><div></div><div></div><div></div><div></div><div></div></div></div> <div>Data displayed</div>															Number of Steps
																4
Function	Adds the contents of register Ⓑ to those of register Ⓐ and stores the sum in register Ⓒ.							Input	Operation							Output
								OFF	No execution							OFF
								ON	Normal execution							OFF
									Overflow							ON
Operand	Symbol	Name	R	X	Y	Z	RW	XW	YW	ZW	D	T	C			Constant
	Ⓐ	Augend register					<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>			
	Ⓑ	Addend register					<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>			
	Ⓒ	Sum					<input type="radio"/>		<input type="radio"/>	<input type="radio"/>	<input type="radio"/>					

Sample Program



Operation



Description

- This instruction adds the contents of register YW02 (23456) to those of register YW01 (12345) and stores the sum in register YW03 when NO-contact R14E is ON. Its output is set to OFF because no overflow occurs.
- If an overflow error occurs during operation, the instruction stores the limit value (65535) in YW03 and sets its output ON.
- The instruction does not execute any operation and sets its output OFF when NO-contact R14E is OFF.

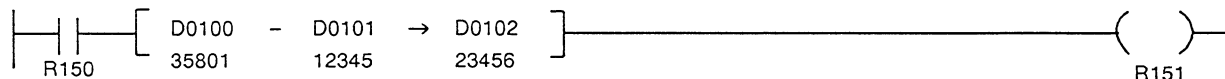
Note

- No constant value can be used because this instruction executes register addition.
- Operands Ⓐ, Ⓑ and Ⓒ can be the same register.

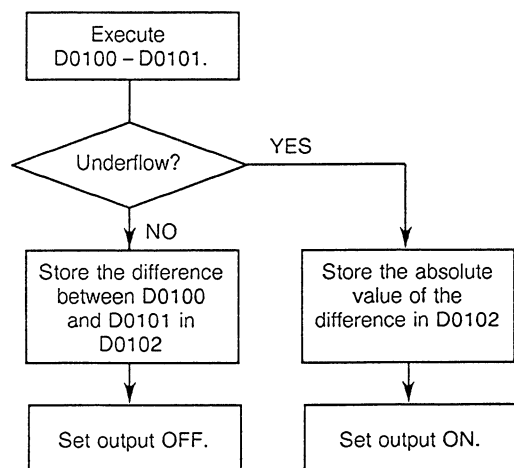
7. Instructions

[illegible]

Sample Program



Operation



Description

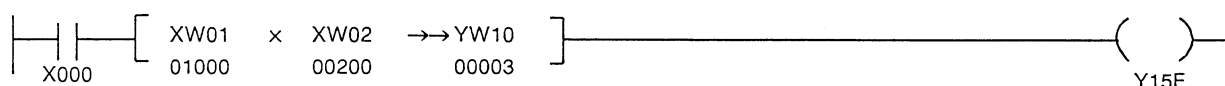
- This instruction subtracts the contents of register D0101 from those of register D0100 and stores the difference in register D0102 when NO-contact R150 is ON. It sets its output OFF if no underflow occurs.
- If an underflow error occurs during operation, the instruction stores the absolute value in D0102 and sets its output ON.
- The instruction does not execute any operation and sets its output OFF when NO-contact R150 is OFF.

Note

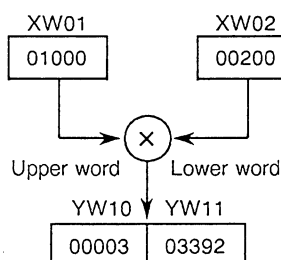
- No constant value can be used because this instruction executes register subtraction.
- Operands Ⓐ, Ⓑ and Ⓒ may be the same register.

×		Register Multiplication (FUN012)															
E x p r e s s i o n	<div>Input</div> <div><div>Ⓐ</div><div>×</div><div>Ⓑ</div><div>→</div><div>Ⓒ</div></div> <div>Execution output</div> <div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div></div> <div>Data displayed</div>														Number of Steps		
															4		
F u n c t i o n	Multiplies the contents of register Ⓐ by those of register Ⓑ and stores the product in double-length register Ⓒ · Ⓒ + 1.							Input		Operation						Output	
								OFF		No execution						OFF	
								ON		Execution						ON	
O p e r a n d	Symbol	Name	R	X	Y	Z	RW	XW	YW	ZW	D	T	C			Constant	
	Ⓐ	Multiplicand register					<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>				
	Ⓑ	Multiplier register					<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>				
	Ⓒ	Product					<input type="radio"/>		<input type="radio"/>	<input type="radio"/>	<input type="radio"/>						

Sample Program



Operation



Description

- This instruction multiplies the contents of register XW01 (1000) by those of register XW02 (200) and stores the products in two consecutive registers, YW10 and YW11, when NO-contact X000 is ON. It stores the upper word in YW10 and the lower word in YW11, and sets its output ON.
- The instruction does not execute any operation and sets its output OFF when NO-contact X000 is OFF.

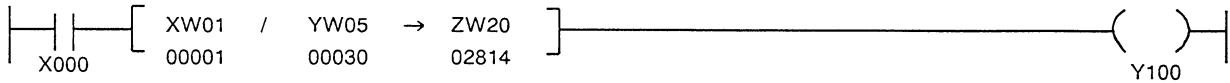
Note

- No constant value can be used because this instruction executes register multiplication.
- Carefully specify register Ⓒ, considering register Ⓒ + 1, so that those registers do not exceed the specifiable register range.
- The result is expressed by (upper word register) x 65536 + (lower word register). In the sample above, 3 x 65536 + 3392 = 200000.

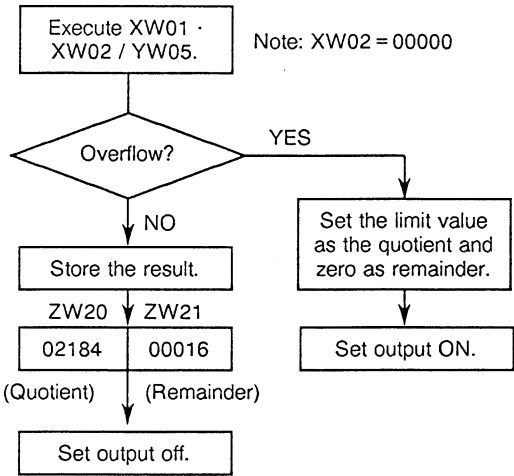
7. Instructions

/		Register Division (FUN013)													
Expression	<div>Input $\left[\begin{array}{ c c c c c } \hline \textcircled{A} & & & & \end{array} \right. / \begin{array}{ c c c c c } \hline \textcircled{B} & & & & \end{array} \rightarrow \begin{array}{ c c c c c } \hline \textcircled{C} & & & & \end{array} \left. \vphantom{\begin{array}{ c c c c c } \hline \textcircled{C} & & & & \end{array}} \right] \text{Overflow output}$</div> <div><div><div><div></div><div></div><div></div><div></div><div></div></div><div><div></div><div></div><div></div><div></div><div></div></div><div><div></div><div></div><div></div><div></div><div></div></div></div><div><div></div><div></div><div></div><div></div><div></div></div><div><div></div><div></div><div></div><div></div><div></div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div><div></div><div></div><div></div><div></div><div></div></div>														

Sample Program



Operation



Description

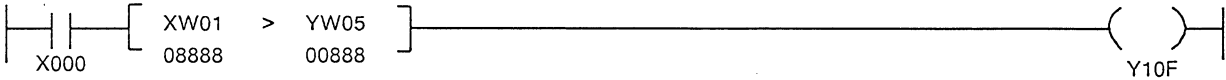
- This instruction divides double-length data consisting of the contents of registers XW01 (00001) and XW02 (00000) by that of register YW05 (30) when NO-contact X000 is ON. If no overflow occurs, it stores the quotient (02184) in register ZW20 and the remainder (16) in register ZW21, then turns the output OFF. If an overflow occurs, it stores the limit value of 65535 as the quotient and zero as the remainder, then turns the output ON.
- The instruction does not execute any operation and turns its output OFF when NO-contact X000 is OFF
- The operation result is as follows:
 $((XW01) \times 65536 + (XW02)) / (YW05) = (ZW20) \text{ remainder } (ZW21)$
For example, $(1 \times 65536 + 00000) / 30 = 2184 \text{ remainder } 16$.

Note

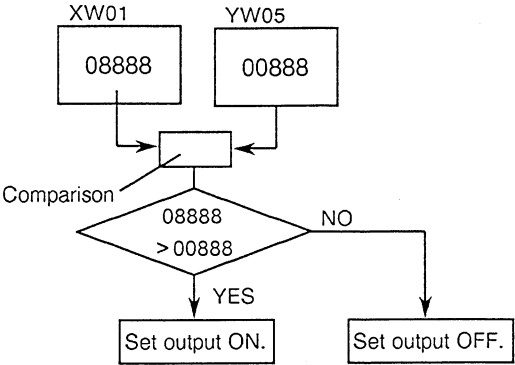
- No constant value can be used because this instruction executes register division.
- Carefully specify register ③, considering register ③ + 1, so that those registers do not exceed the specifiable register range.
- The dividend is the double-length data of registers ① and ① + 1.

>		Greater-Than Comparison (FUN014)															
E x p r e s s i o n	<div>Input <div><div>Ⓐ</div><div>></div><div>Ⓑ</div></div> Decision output</div> <div><div><div></div><div></div><div></div><div></div><div></div></div><div><div></div><div></div><div></div><div></div><div></div></div></div> <div>Data displayed</div>															Number of Steps	
																3	
F u n c t i o n	Compares the contents of register Ⓐ with those of register Ⓑ and sets the output ON if the former is greater than the latter (Ⓐ > Ⓑ).							Input		Operation						Output	
								OFF		No execution						OFF	
								ON		A > B						ON	
										A ≤ B						OFF	
O p e r a n d	Symbol	Name	R	X	Y	Z	RW	XW	YW	ZW	D	T	C			Constant	
	Ⓐ	Value to be compared					<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>				
	Ⓑ	Reference value					<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>				

Sample Program



Operation



Description

- This instruction compares the contents of register XW01 (8888) with the contents of register YW05 (888) when NO-contact X000 is ON.
If XW01 > YW05, it sets the output ON.
If XW01 ≤ YW05, it sets the output OFF.
For the example program, it sets the output ON.
- The instruction does not execute any comparison and sets its output OFF when NO-contact X000 is OFF.

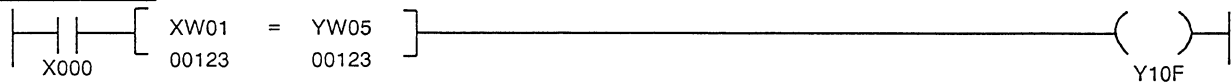
Note

- No constant value can be used because this instruction executes register comparison.
- The value to be compared Ⓐ and the reference value Ⓑ are not altered by comparison.

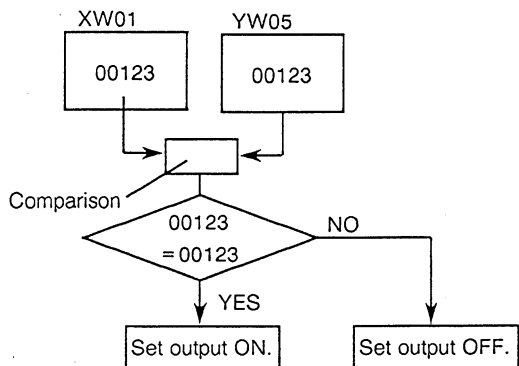
7. Instructions

=		Equal Comparison (FUN015)														
E x p r e s s i o n	<div>Input</div> <div><div>Ⓐ</div><div>=</div><div>Ⓑ</div></div> <div>Decision output</div> <div><div><div></div><div></div><div></div><div></div><div></div><div></div></div><div><div></div><div></div><div></div><div></div><div></div><div></div></div></div> <div>Data displayed</div>															Number of Steps
																3
F u n c t i o n	Compares the contents of register Ⓐ with those of register Ⓑ and sets the output ON if the former equals the latter {Ⓐ = Ⓑ}.						Input		Operation						Output	
							OFF		No execution						OFF	
							ON		A = B						ON	
									A ≠ B						OFF	
O p e r a n d	Symbol	Name	R	X	Y	Z	RW	XW	YW	ZW	D	T	C	T.	C.	Constant
	Ⓐ	Value to be compared					<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>			
	Ⓑ	Reference value					<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>			

Sample Program



Operation



Note

- No constant value can be used because this instruction executes register comparison.
- The value to be compared ① and the reference value ② are not altered by comparison.

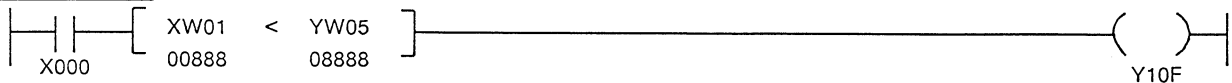
Description

- This instruction compares the contents of register XW01 (00123) with the contents of register YW05 (00123) when NO-contact X000 is ON. If XW01 = YW05, it sets the output ON. For the example program, it sets the output ON.
- The instruction does not execute comparison and sets its output OFF when NO-contact X000 is OFF.

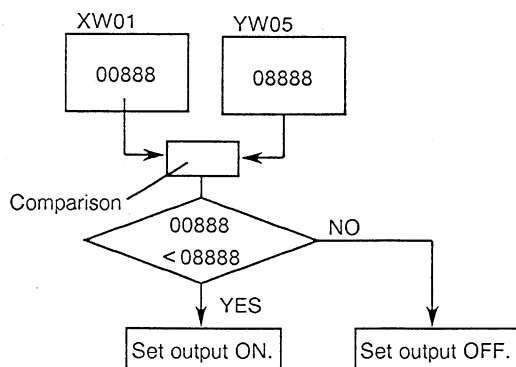
7. Instructions

[illegible]

Sample Program



Operation



Description

- This instruction compares the contents of register XW01 (888) with the contents of register YW05 (8888) when NO-contact X000 is ON.
If $XW01 < YW05$, it sets the output ON.
If $XW01 \geq YW05$, it sets the output OFF.
For the example program, it sets the output ON.
- The instruction does not execute any comparison and sets its output OFF when NO-contact X000 is OFF.

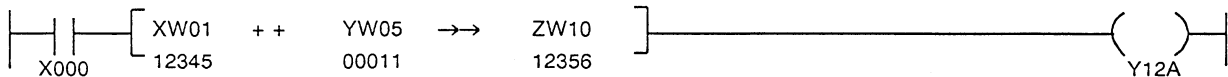
Note

- No constant value can be used because this instruction executes register comparison.
- The value to be compared Ⓐ and the reference value Ⓑ are not altered by comparison.

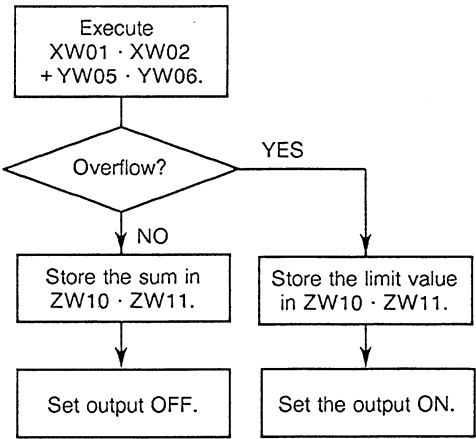
7. Instructions

+ +		Register Double-Length Addition (FUN017)															
E x p r e s s i o n	<div>Input [① + + ② $\rightarrow\rightarrow$ ③] Overflow output</div> <div><div><div></div><div></div><div></div><div></div><div></div></div><div><div></div><div></div><div></div><div></div><div></div></div><div><div></div><div></div><div></div><div></div><div></div></div></div> <div>Data displayed</div>															Number of Steps	
																4	
F u n c t i o n	Adds the contents of registers ② and ② + 1 to those of registers ① and ① + 1 and stores the sum in registers ③ and ③ + 1 .							Input		Operation						Output	
								OFF		No execution						OFF	
								ON		Normal execution						OFF	
										Overflow						ON	
O p e r a n d	Symbol	Name	R	X	Y	Z	RW	XW	YW	ZW	D	T	C			Constant	
	①	Augend register					<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>						
	②	Addend register					<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>						
	③	Sum					<input type="radio"/>		<input type="radio"/>	<input type="radio"/>	<input type="radio"/>						

Sample Program



Operation



Note

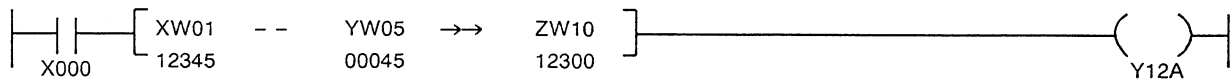
- No numeric value can be used because this instruction executes register addition.
- A and A + 1, B and B + 1, and C and C + 1 are handled as 32-bit, double-length registers.
- A, B, and C may be even or odd registers.

Description

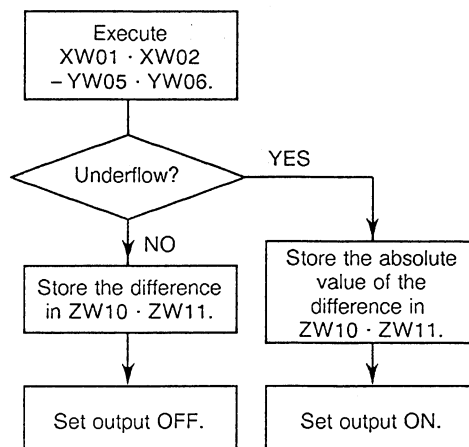
- This instruction adds the contents of two consecutive registers, XW01 and XW02, to those of two other consecutive registers, YW05 and YW06, when NO-contact X000 is ON. If no overflow occurs, it stores the sum in two consecutive registers, ZW10 and ZW11, and sets its output OFF.
- If an overflow occurs during operation, the instruction stores the limit value HFFFFFFF in registers C and C + 1, then sets its output ON.
- The instruction does not execute any operation and sets its output OFF when NO-contact X000 is OFF.

--		Register Double-Length Subtraction											(FUN018)			
E x p r e s s i o n	<div>Input</div> <div><div>Ⓐ</div><div>--</div><div>Ⓑ</div><div>→→</div><div>Ⓒ</div></div> <div>Underflow output</div> <div><div><div></div><div></div><div></div><div></div><div></div><div></div></div><div><div></div><div></div><div></div><div></div><div></div><div></div></div><div><div></div><div></div><div></div><div></div><div></div><div></div></div></div> <div>Data displayed</div>													Number of Steps		
														4		
F u n c t i o n	Subtracts the contents of registers Ⓑ and Ⓑ + 1 from those of registers Ⓐ and Ⓐ + 1 and stores the result in registers Ⓒ and Ⓒ + 1.						Input		Operation						Output	
							OFF		No execution						OFF	
							ON		Normal execution						OFF	
									Underflow						ON	
O p e r a n d	Symbol	Name	R	X	Y	Z	RW	XW	YW	ZW	D	T	C			Constant
	Ⓐ	Minuend register					<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>					
	Ⓑ	Subtrahend register					<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>					
	Ⓒ	Difference					<input type="radio"/>		<input type="radio"/>	<input type="radio"/>	<input type="radio"/>					

Sample Program



Operation



Note

- No numeric value can be used because this instruction executes register subtraction.
- Ⓐ and Ⓐ + 1, Ⓑ and Ⓑ + 1, and Ⓒ and Ⓒ + 1 are handled as 32-bit, double-length registers.
- Ⓐ, Ⓑ, and Ⓒ may be even or odd registers.

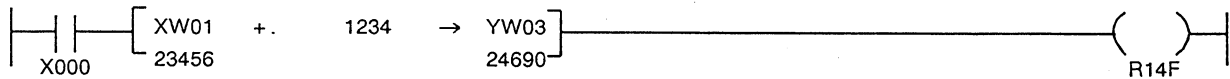
Description

- This instruction subtracts the contents of two consecutive registers, YW05 and YW06 from those of two other consecutive registers, XW01 and XW02, when NO-contact X000 is ON. If no underflow occurs, it stores the difference in two consecutive registers, ZW10 and ZW11, then sets its output OFF.
- If an underflow occurs during operation, the instruction stores the absolute value in registers Ⓒ and Ⓒ + 1, then sets its output ON.
- The instruction does not execute any operation and sets its output OFF when NO-contact X000 is OFF.

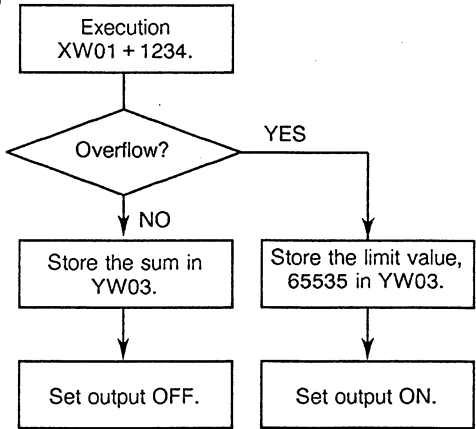
7. Instructions

+.		Constant Addition (FUN020)														Number of Steps	
E x p r e s s i o n	Input	[① +. ② → ③] Overflow output														4 / 5	
		[5 boxes] [5 boxes] [5 boxes]															
		Data displayed															
F u n c t i o n	Adds a numeric value ② to the contents of register ① and stores the sum in register ③.							Input		Operation						Output	
								OFF		No execution						OFF	
								ON		Normal execution						OFF	
O p e r a n d										Overflow						ON	
	Symbol	Name	R	X	Y	Z	RW	XW	YW	ZW	D	T	C			Constant	
	①	Augend register					○	○	○	○	○	○	○				
	②	Addend														0~65535	
	③	Sum					○		○	○	○						

Sample Program



Operation



Description

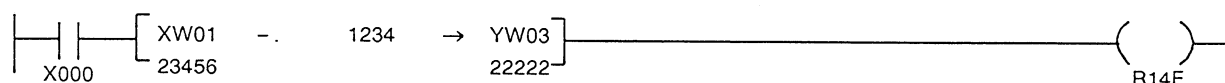
- This instruction adds constant value, 1234, to the contents of register XW01 (23456) and stores the sum in register YW03 when NO-contact X000 is ON. It sets its output OFF if no overflow occurs.
- If an overflow occurs during operation, the instruction stores the limit value, 65535, in register ③, then sets its output ON.
- The instruction does not execute any operation and sets its output OFF when NO-contact X000 is OFF.

Note

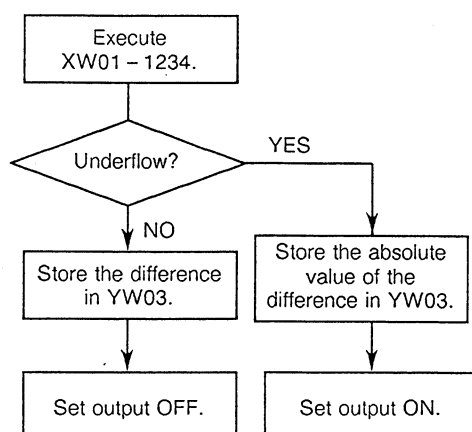
- No register can be used for the addend ② because this instruction executes numeric value addition.
- Augend register ① and sum register ③ may be the same register.
Example: D0100 +.1 → D0100 (increment)
- The number of steps used this instruction depends on the constant value:
 - ② < 256: 4 steps
 - ② ≥ 256: 5 steps

- .		Constant Subtraction (FUN021)											Number of Steps	
E X P R E S S I O N	Input	<div> <div>Ⓐ</div> <div>- .</div> <div>Ⓑ</div> <div>→</div> <div>Ⓒ</div> </div> <div>Underflow output</div> <div> <div>□□□□□</div> <div>□□□□□</div> <div>□□□□□</div> </div> <div>Data displayed</div>											4 / 5	
F U N C T I O N	Subtracts a numeric value Ⓑ from the contents of register Ⓐ and stores the result in register Ⓒ.						Input		Operation					Output
							OFF		No execution					OFF
							ON		Normal execution					OFF
O P E R A N D	Symbol	Name	R	X	Y	Z	RW	XW	YW	ZW	D	T	C	Constant
	Ⓐ	Minuend register					○	○	○	○	○	○	○	
	Ⓑ	Subtrahend												0~65535
	Ⓒ	Difference					○		○	○	○			

Sample Program



Operation



Note

- No register can be used for the subtrahend Ⓑ because this instruction executes numeric value subtraction.
- Minuend register Ⓐ and difference register Ⓒ may be the same register. Example: D0100-. 1 → D0100 (decrement)
- The number of steps used for this instruction depends on the constant value:
 - Ⓑ < 256: 4 steps
 - Ⓑ ≥ 256: 5 steps

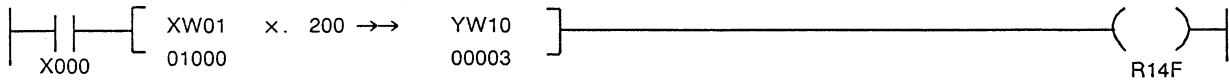
Description

- This instruction subtracts a numeric value, 1234, from the contents of register XW01 (23456) and stores the difference, 22222, in register YW03 when NO-contact X000 is ON. It sets its output OFF if no underflow occurs.
- If an underflow occurs during operation, the instruction stores the absolute value in register Ⓒ, then sets its output ON.
- The instruction does not execute any operation and sets its output OFF when NO-contact X000 is OFF.

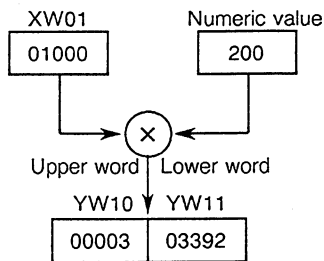
7. Instructions

×.		Constant Multiplication (FUN022)																
E x p r e s s i o n	<div>Input [① ×. ② →→ ③] Execution output</div> <div><div><div></div><div></div><div></div><div></div><div></div></div><div></div><div></div><div></div><div></div><div></div></div> <div>Data displayed</div>																Number of Steps	
																	4 / 5	
F u n c t i o n	Multiplies the contents of register ① by numeric value ② and stores the product in double-length register ③ · ③ + 1.							Input		Operation							Output	
								OFF		No execution							OFF	
								ON		Execution							ON	
O p e r a n d	Symbol	Name	R	X	Y	Z	RW	XW	YW	ZW	D	T	C			Constant		
	①	Multiplicand register					○	○	○	○	○	○	○					
	②	Multiplier														0~65535		
	③	Product					○		○	○	○							

Sample Program



Operation



Description

- This instruction multiplies the contents (1000) of register XW01 by a numeric value, 200, and stores the product in two consecutive registers, YW10 and YW11, when NO-contact X000 is ON.
- It stores the upper word part in YW10 and the lower word part in YW11, then sets its output ON.
- The instruction does not execute any operation and sets the output OFF when NO-contact X000 is OFF.
- The operation result is expressed by (upper word register) x 65536 + (lower word register). In the example above, 3 x 65536 + 3392 = 200000

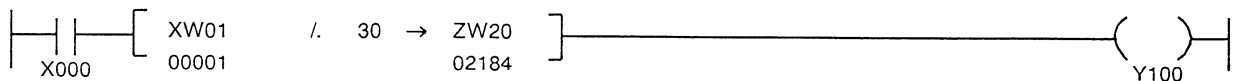
Note

- No register can be used for the multiplier ② because this instruction executes numeric value multiplication.
- Carefully specify register ③, considering register ③ + 1, so that these registers do not exceed the specifiable register range.
- The number of steps used for this instruction depends on the constant value:
 - ② < 256: 4 steps
 - ② ≥ 256: 5 steps

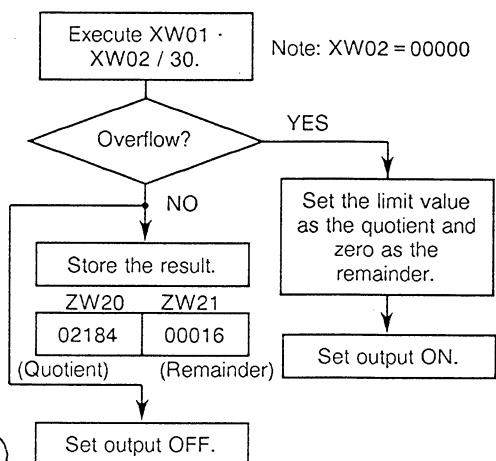
7. Instructions

[illegible]

Sample Program



Operation



Description

- This instruction divides double-length data consisting of the contents of registers XW01 (00001) and the XW02 (00000) by a numeric (30) when NO-contact X000 is ON. If no overflow occurs, it stores the quotient (2184) in registers ZW20 and the remainder (16) in register ZW21, then sets the output OFF.
- If an overflow occurs, it stores the limit value, 65535, as the quotient and zero as the remainder, then sets the output ON.
- The instruction does not execute any operation and sets its output OFF when NO-contact X000 is OFF.
- The operation is as follows:

$$\{ (XW01) \times 65536 + (XW02) \} / 30 = (ZW20) \text{ remainder } (ZW21)$$
 For example, $(1 \times 65536 + 0) / 30 = 2184 \text{ remainder } 16$

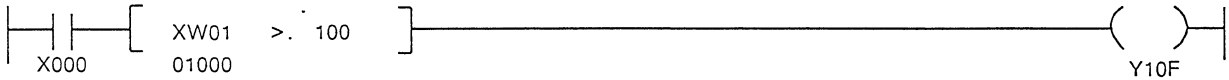
Note

- No register value can be used for the divisor ③ because this instruction executes numeric value division.
- Carefully specify register ④, considering register ④ + 1, so that these registers do not exceed the specifiable register range.
- The dividend is the double-length data of registers ① and ① + 1.
- The number of steps used for this instruction depends on the constant value :
 - ③ < 256: 4 steps
 - ③ ≥ 256: 5 steps

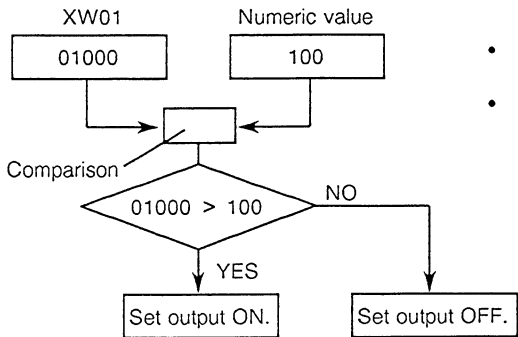
7. Instructions

>.		Consant Comparison Greater-Than (FUN024)															
E x p r e s s i o n	<div>Input <div><div>Ⓐ</div><div>>.</div><div>Ⓑ</div></div> Decision output</div> <div><div><div></div><div></div><div></div><div></div><div></div></div><div>↑</div><div>Data displayed</div></div>														Number of Steps		
															3 / 4		
F u n c t i o n	Compares the contents of register Ⓐ with numeric value Ⓑ and sets the output ON if the former is greater than the latter {Ⓐ > Ⓑ}.						Input		Operation						Output		
							OFF		No execution						OFF		
							ON		A > B						ON		
									A ≤ B						OFF		
O p e r a n d	Symbol	Name		R	X	Y	Z	RW	XW	YW	ZW	D	T	C			Constant
	Ⓐ	Value to be compared						○	○	○	○	○	○	○			
	Ⓑ	Reference value															0~65535

Sample Program



Operation



Description

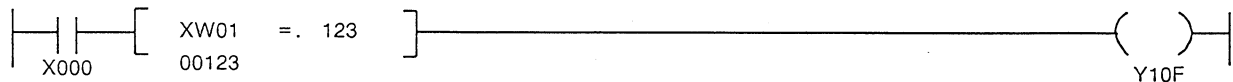
- This instruction compares the contents of register XW01 (1000) with a numeric value (100) when NO contact X000 is ON.
- If XW01 > 100, it sets the output ON. For the example program, it sets the output ON
- The instruction does not execute any comparison and sets its output OFF when NO-contact X000 is OFF.

Note

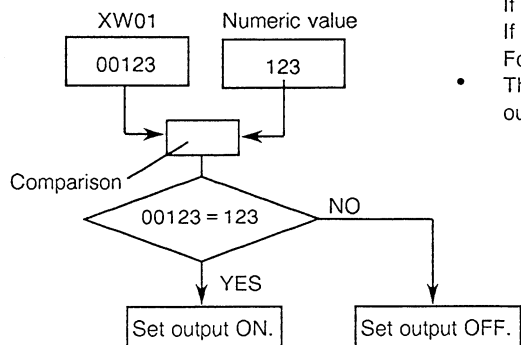
- No register can be used for reference value ② because this instruction executes numeric value comparison.
- The number of steps used for this instruction depends on the constant value:
 - ② < 256: 3 steps
 - ② ≥ 256: 4 steps

= .		Constant Comparison Equal (FUN025)													
E x p r e s s i o n	Input	<div> <div>Ⓐ</div> <div>= .</div> <div>Ⓑ</div> </div> <div>Decision output</div> <div> <div>□ □ □ □ □ □</div> <div>↑</div> <div>Data displayed</div> </div>												Number of Steps	
														3 / 4	
F u n c t i o n	Compares the contents of register Ⓐ with numeric value Ⓑ and sets the output ON if the former equals the latter {Ⓐ = Ⓑ}.						Input		Operation						Output
							OFF		No execution						OFF
							ON		A = B						ON
O p e r a n d	Symbol	Name	R	X	Y	Z	RW	XW	YW	ZW	D	T	C		Constant
	Ⓐ	Value to be compared					○	○	○	○	○	○	○		
	Ⓑ	Reference value													0~65535

Sample Program



Operation



Note

- No register can be used for reference value Ⓑ because this instruction executes numeric value comparison.
- The number of steps used for this instruction depends on the constant value:
 - Ⓑ > 256: 3 steps
 - Ⓑ ≥ 256: 4 steps

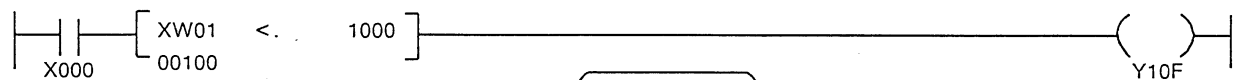
Description

- This instruction compares the contents of register XW01 (00123) with a numeric value (123) when NO-contact X000 is ON.
If XW01 = 123, it sets the output ON.
If XW01 ≠ 123, it sets the output OFF.
For the example program, it sets the output ON.
- The instruction does not execute any comparison and sets its output OFF when NO-contact X000 is OFF.

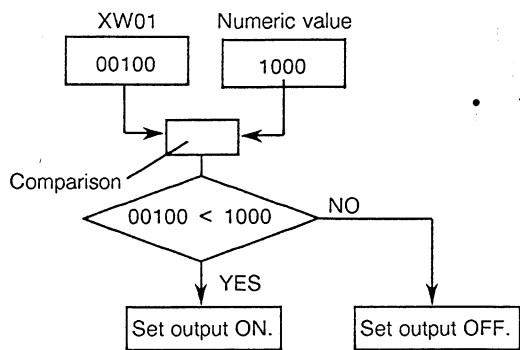
7. Instructions

<.		Constant Comparison Less-Than (FUN026)												Number of Steps	
E x p r e s s i o n	Input	Decision output												3 / 4	
		Data displayed													
F u n c t i o n	Compares the contents of register ① with numeric value ② and sets the output ON if the former is less than the latter {① < ②}.						Input		Operation						Output
							OFF		No execution						OFF
							ON		A < B						ON
O p e r a n d	Symbol	Name	R	X	Y	Z	RW	XW	YW	ZW	D	T	C		Constant
	①	Value to be compared					○	○	○	○	○	○	○		
	②	Reference value													0~65535

Sample Program



Operation



Description

- This instruction compares the contents of register XW01 (100) with a numeric value (1000) when NO-contact X000 is ON.
If XW01 < 1000, it sets the output ON.
If XW01 ≥ 1000, it sets the output OFF.
For the example program, it sets the output ON.
- The instruction does not execute any comparison and sets its output OFF when NO-contact X000 is OFF.

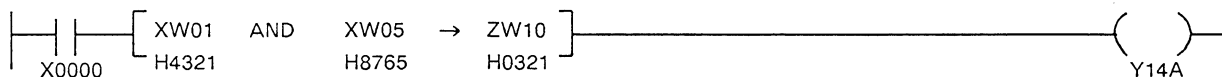
Note

- No register can be used for reference value ② because this instruction executes numeric comparison.
- The number of steps used for this instruction depends on the constant value:
② < 256: 3 steps
② ≥ 256: 4 steps

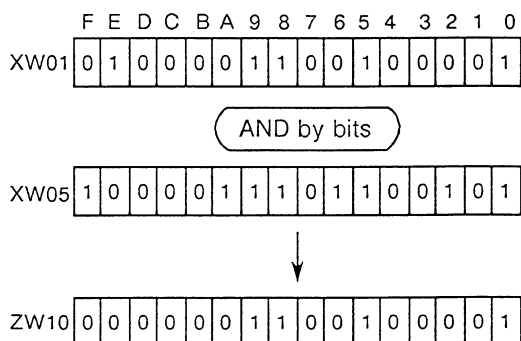
7.5 Logical operations

AND		Register AND (FUN030)												Number of Steps	
E x p r e s s i o n	Input	<div> <div>Ⓐ</div> <div>AND</div> <div>Ⓑ</div> <div>→</div> <div>Ⓒ</div> </div> <div>Execution output</div>												4	
		<div> <div>H</div> <div></div> <div></div> <div></div> <div></div> </div> <div></div> <div> <div>H</div> <div></div> <div></div> <div></div> <div></div> </div> <div></div> <div> <div>H</div> <div></div> <div></div> <div></div> <div></div> </div> <div></div>												Data displayed	
F u n c t i o n	ANDs individual bits of register Ⓐ with those of register Ⓑ and stores the result in register Ⓒ.						Input		Operation						Output
							OFF		No execution						OFF
							ON		Execution						ON
O p e r a n d	Symbol	Name	R	X	Y	Z	RW	XW	YW	ZW	D	T	C		Constant
	Ⓐ	Operation data					○	○	○	○	○	○	○		
	Ⓑ	Operation data					○	○	○	○	○	○	○		
	Ⓒ	AND					○		○	○	○				

Sample Program



Operation



Description

- This instruction ANDs individual bits in the word data of registers XW01 and XW05 and stores the result in register ZW10 when NO-contact X000 is ON, then sets the output ON.
- The instruction does not execute the AND operation and sets its output OFF when NO-contact X000 is OFF.

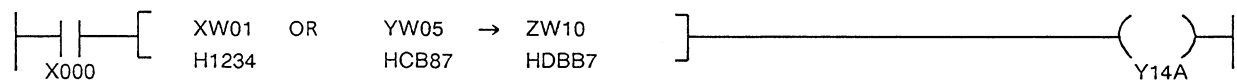
Note

- No numeric value can be used because this instruction executes a register AND calculation.

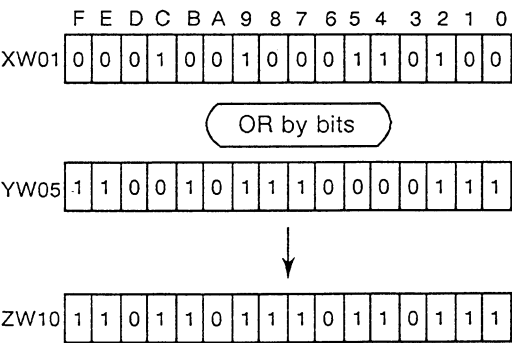
7. Instructions

OR		Register OR (FUN031)													
E x p r e s s i o n	Input	OR													
		Execution output													
F u n c t i o n	H	Data displayed													
		Number of Steps													
O p e r a t i o n	Symbol	Name	R	X	Y	Z	RW	XW	YW	ZW	D	T	C		Constant
O p e r a t i o n	A	Operation data					O	O	O	O	O	O	O		
O p e r a t i o n	B	Operation data					O	O	O	O	O	O	O		
O p e r a t i o n	C	OR					O		O	O	O				
O p e r a t i o n															

Sample Program



Operation



Description

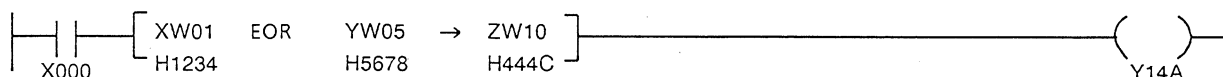
- This instruction ORs individual bits in the word data of registers XW01 and YW05 and stores the result in register ZW10 when NO-contact X000 is ON, then sets the output ON.
- The instruction does not execute any operation and sets output OFF when NO-contact X000 is OFF.

Note

- No numeric value can be used because this instruction executes a register OR calculation.

EOR		Register Exclusive OR (FUN032)															
Expression	<div> Input <div> <div>Ⓐ</div> <div>Ⓑ</div> <div>→</div> <div>Ⓒ</div> </div> <div> <div>EOR</div> <div>Execution output</div> </div> </div> <div> <div>H</div> <div>H</div> <div>H</div> </div> <div> <div>Data displayed</div> </div>	Number of Steps															
		4															
Function	Exclusive ORs individual bits of register Ⓐ with those of register Ⓑ and stores the result in register Ⓒ.							Input		Operation							Output
								OFF		No execution							OFF
								ON		Execution							ON
Operand	Symbol	Name	R	X	Y	Z	RW	XW	YW	ZW	D	T	C				Constant
	Ⓐ	Operation data					○	○	○	○	○	○	○				
	Ⓑ	Operation data					○	○	○	○	○	○	○				
	Ⓒ	Exclusive OR				○	○		○	○	○						

Sample Program



Operation

	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0
XW01	0	0	0	1	0	0	1	0	0	0	1	1	0	1	0	0

Exclusive OR by bits

YW05	0	1	0	1	0	1	1	0	0	1	1	1	1	0	0	0
------	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

ZW10	0	1	0	0	0	1	0	0	0	1	0	0	1	1	0	0
------	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

Description

- This instruction exclusive ORs individual bits in the word data of registers XW01 and YW05 and stores the result in registers ZW10 when NO-contact X000 is ON, then sets the output ON.
- The instruction does not execute any operation and sets its output OFF when NO-contact X000 is OFF.

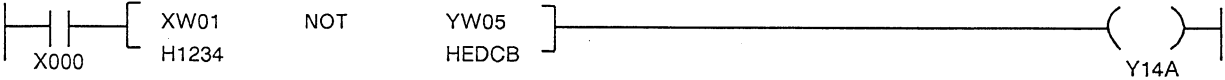
Note

- No numeric value can be used because this instruction executes a register exclusive OR calculation.

7. Instructions

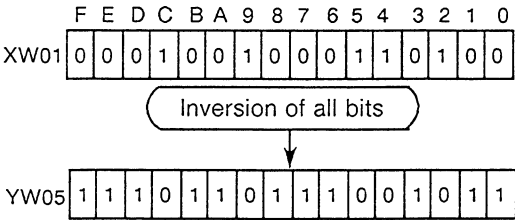
NOT		Inversion (FUN034)															
E x p r e s s i o n	<div>Input<div>Ⓐ</div>NOT<div>Ⓑ</div>Execution output</div> <div>H<div></div><div></div><div></div><div></div><div></div></div> <div>H<div></div><div></div><div></div><div></div><div></div></div> <div>Data displayed</div>															Number of Steps	
																3	
F u n c t i o n	Inverts the contents of register Ⓐ and stores the result in register Ⓑ.						Input		Operation						Output		
							OFF		No execution						OFF		
							ON		Execution						ON		
O p e r a n d	Symbol	Name	R	X	Y	Z	RW	XW	YW	ZW	D	T	C			Constant	
	Ⓐ	Operation data					<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>				
	Ⓑ	Inverted data					<input type="radio"/>		<input type="radio"/>	<input type="radio"/>	<input type="radio"/>						

Sample Program



Y14A

Operation



Description

- This instruction stores the inverted form of all bits of register XW01 in register YW05 and sets the output ON when NO-contact X000 is ON.
- The instruction does not execute any operation and sets its output OFF when NO-contact X000 is OFF.

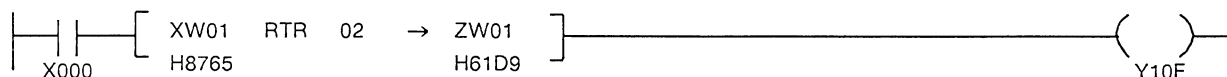
Note

- No numeric value can be used because this instruction executes register inversion.

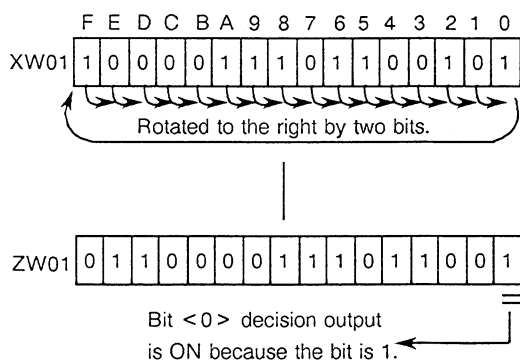
7. Instructions

RTR		Right Rotate (FUN035)												Number of Steps	
E x p r e s s i o n	Input	<div> <div>Ⓐ</div> <div>RTR</div> <div>Ⓑ</div> <div>→</div> <div>Ⓒ</div> </div> <div>Bit <0> decision output</div>												4	
		<div> <div>H</div> <div> <div></div><div></div><div></div><div></div> </div> </div> <div> <div>H</div> <div> <div></div><div></div><div></div><div></div> </div> </div> <div>Data displayed</div>													
F u n c t i o n	Rotates the contents of register Ⓐ to the right by the number of bits specified by Ⓑ and stores the result in register Ⓒ.						Input		Operation						Output
							OFF		No execution						OFF
							ON		Execution		Bit <0> is 1.				ON
											Bit <0> is 0.				OFF
O p e r a n d	Symbol	Name	R	X	Y	Z	RW	XW	YW	ZW	D	T	C		Constant
	Ⓐ	Operation data					○	○	○	○	○	○	○		
	Ⓑ	Number of bit positions rotated													0~15
	Ⓒ	Rotation result					○		○	○	○				

Sample Program



Operation



Description

- This instruction rotates the contents of register XW01 (H8765) to the right by two bits and stores the rotated data in register ZW01 (H61D9) when NO-contact X000 is ON.
- It sets the output ON because bit <0> = 1 in the result. If bit <0> is 0, it sets the output OFF.
- The instruction does not execute any rotation and sets its output OFF when NO-contact X000 is OFF.

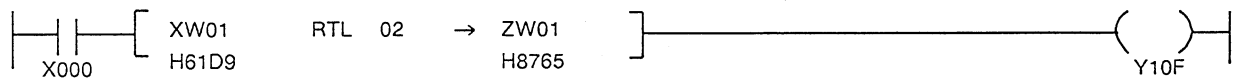
Note

- The number of bit positions rotated, Ⓑ, is specifiable 0 to 15.
- Operation data Ⓐ is not altered by rotation.

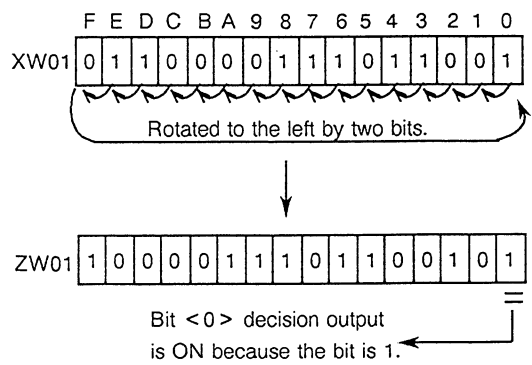
7. Instructions

RTL		Left Rotate (FUN036)															
E x p r e s s i o n	<div>Input [㉔ RTL ㉕ → ㉖] Bit <0> decision output</div> <div>H [] H []</div> <div>↑ ↑</div> <div>Data displayed</div>													Number of Steps			
														4			
F u n c t i o n	Rotates the contents of register ㉔ to the left by the number of bits specified by ㉕ and stores the rotation result in register ㉖.						Input		Operation						Output		
							OFF		No execution						OFF		
							ON		Execution				Bit <0> is 1.		ON		
													Bit <0> is 0.		OFF		
O p e r a n d	Symbol	Name		R	X	Y	Z	RW	XW	YW	ZW	D	T	C			Constant
	㉔	Operation data						○	○	○	○	○	○	○			
	㉕	Number of bit positions rotated															0~15
	㉖	Rotation result						○		○	○	○					

Sample Program



Operation



Description

- This instruction rotates the contents of register XW01 (H61D9) to the left by two bits and stores the rotated data in register ZW01 (H8765) when NO-contact X000 is ON.
- It sets the output ON because bit <0> = 1 in the result.
- If bit <0> is 0, it sets the output OFF.
- The instruction does not execute any rotation and sets its output OFF when NO-contact X000 is OFF.

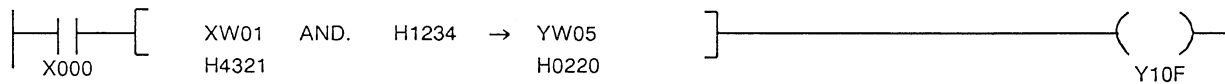
Note

- The number of bit positions rotated ② is specifiable from 0 to 15.
- Operation data ① is not altered by rotation.

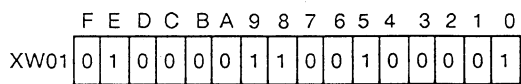
7. Instructions

AND.		Constant AND (FUN040)															
Expression	<div><div>Input</div><div><div>Ⓐ</div><div>AND.</div><div>Ⓑ</div><div>→</div><div>Ⓒ</div></div><div>Execution output</div></div> <div><div>H</div><div><div></div><div></div><div></div><div></div></div><div></div><div><div>H</div><div><div></div><div></div><div></div><div></div></div><div></div></div><div>Data displayed</div></div>															Number of Steps	
	4 / 5																
Function	ANDs individual bits of register Ⓐ with those of numeric value Ⓑ and stores the result in register Ⓒ.							Input		Operation						Output	
								OFF		No execution						OFF	
								ON		Execution						ON	
Parameter	Symbol	Name	R	X	Y	Z	RW	XW	YW	ZW	D	T	C			Constant	
	Ⓐ	Operation data					<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>				
	Ⓑ	Operation data														H0000~HFFFF	
	Ⓒ	AND					<input type="radio"/>		<input type="radio"/>	<input type="radio"/>	<input type="radio"/>						

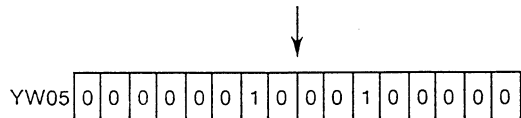
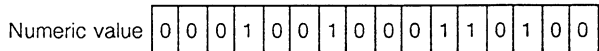
Sample Program



Operation



AND by bits



Description

- This instruction ANDs individual bits in the word data of register XW01 with those of a numeric value (H1234) and stores the result in register YW05 when NO-contact X000 is ON, then sets the output ON.
- The instruction does not execute any operation and sets its output OFF when NO-contact X000 is OFF.

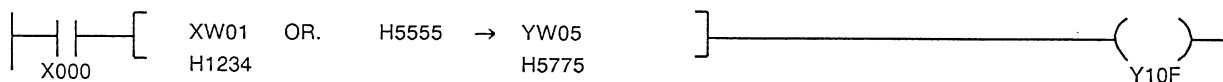
Note

- No register can be used for operation data ③ because this instruction executes a numeric value AND calculation. The number of steps used for this instruction depends on the constant value:
 - ③ < H0100: 4 steps
 - ③ ≥ H0100: 5 steps

7. Instructions

OR.		Constant OR (FUN041)												Number of Steps	
E x p r e s s i o n	Input	<div> <div>Ⓐ</div> <div>OR.</div> <div>Ⓑ</div> <div>→</div> <div>Ⓒ</div> </div> <div>Execution output</div>												4 / 5	
		<div> <div>H</div> <div></div> <div></div> <div></div> <div></div> </div> <div></div> <div> <div>H</div> <div></div> <div></div> <div></div> <div></div> </div> <div>Data displayed</div>													
F u n c t i o n	ORs individual bits of register Ⓐ with those of numeric value Ⓑ and stores the result in register Ⓒ.						Input		Operation						Output
							OFF		No execution						OFF
							ON		Execution						ON
O p e r a n d	Symbol	Name	R	X	Y	Z	RW	XW	YW	ZW	D	T	C		Constant
	Ⓐ	Operation data					○	○	○	○	○	○	○		
	Ⓑ	Operation data													H0000 ~ HFFFF
	Ⓒ	OR					○		○	○	○				

Sample Program



Operation

	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0
XW01	0	0	0	1	0	0	1	0	0	0	1	1	0	1	0	0

OR by bits

Numeric value	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1
---------------	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

YW05	0	1	0	1	0	1	1	1	0	1	1	1	0	1	0	1
------	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

Description

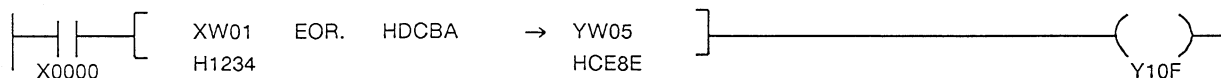
- This instruction ORs individual bits in the word data of register XW01 with those of a numeric value (H5555) and stores the result in register YW05 when NO-contact X000 is ON, then sets the output ON.
- The instruction does not execute any operation and sets output OFF when NO-contact X000 is OFF.

Note

- No register can be used for operation data Ⓑ because this instruction executes a numeric value OR calculation.
- The number of steps used for this instruction depends on the constant value:
 - Ⓑ < H0100: 4 steps
 - Ⓑ ≥ H0100: 5 steps

EOR.		Constant Exclusive OR (FUN042)															
E x p r e s s i o n	<div>Input</div> <div>Ⓐ</div> <div>EOR.</div> <div>Ⓑ</div> <div>→</div> <div>Ⓒ</div> <div>Execution output</div> <div>H</div> <div><div></div><div></div><div></div><div></div></div> <div>H</div> <div><div></div><div></div><div></div><div></div></div> <div>Data displayed</div>															Number of Steps	
	4 / 5																
F u n c t i o n	Exclusive ORs individual bits of register Ⓐ with those of numeric value Ⓑ and stores the result in register Ⓒ.							Input		Operation						Output	
								OFF		No execution						OFF	
								ON		Execution						ON	
O p e r a n d	Symbol	Name	R	X	Y	Z	RW	XW	YW	ZW	D	T	C			Constant	
	Ⓐ	Operation data					<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>				
	Ⓑ	Operation data														H0000~HFFFF	
	Ⓒ	Exclusive OR					<input type="radio"/>		<input type="radio"/>	<input type="radio"/>	<input type="radio"/>						

Sample Program



Operation

	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0
XW01	0	0	0	1	0	0	1	0	0	0	1	1	0	1	0	0

Exclusive OR by bits

Numeric value	1	1	0	1	1	1	0	0	1	0	1	1	1	0	1	0
---------------	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

YW05	1	1	0	0	1	1	1	0	1	0	0	0	1	1	1	0
------	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

Description

- This instruction exclusive ORs individual bits in the word data of register XW01 with those of a numeric value (HDCBA) and stores the result in register YW05 when NO-contact X000 is ON, then sets the output ON.
- The instruction does not execute any operation and sets its output OFF when NO-contact X000 is OFF.

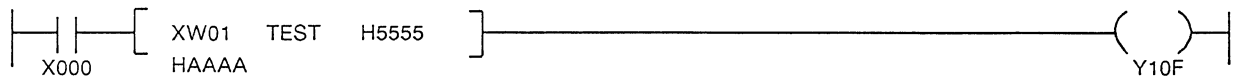
Note

- No register can be used for operation data Ⓑ because this instruction executes a numeric value exclusive OR calculation.
- The number of steps used for this instruction depends on the constant value:
 - Ⓑ < H0100: 4 steps
 - Ⓑ ≥ H0100: 5 steps

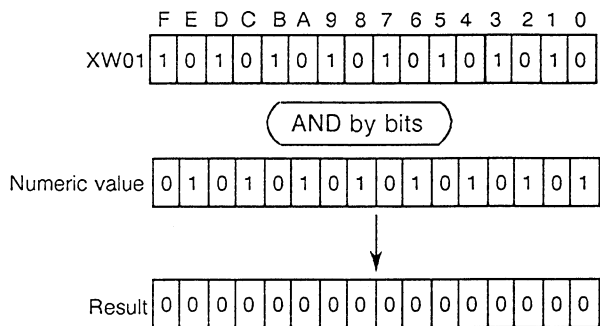
7. Instructions

TEST		Bit Test (FUN043)														Number of Steps						
E x p r e s s i o n	<div>Input [① TEST ②] Decision output</div> <div>H <table><tr><td></td><td></td><td></td><td></td></tr></table></div> <div>↑ Data displayed</div>																				3 / 4	
F u n c t i o n	ANDs individual bits of register ① with those of numeric value ② and sets the output ON if the result is not 0, or sets the output OFF if it is 0.							Input		Operation						Output						
								OFF		No execution						OFF						
								ON		AND result is not 0.						ON						
										AND result is 0.						OFF						
O p e r a n d	Symbol	Name	R	X	Y	Z	RW	XW	YW	ZW	D	T	C			Constant						
	①	Operation data					○	○	○	○	○	○	○									
	②	Test data														H0000 ~ HFFFF						

Sample Program



Operation



Description

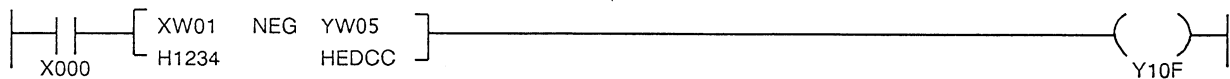
- This instruction ANDs individual bits in the word data of register XW01 with those of a numeric value (H5555) when NO-contact X000 is ON. It sets the output OFF because the result is 0. If the result is not 0, it sets the output ON.
- The instruction does not execute any operation and sets its output OFF when NO-contact X000 is OFF.

Note

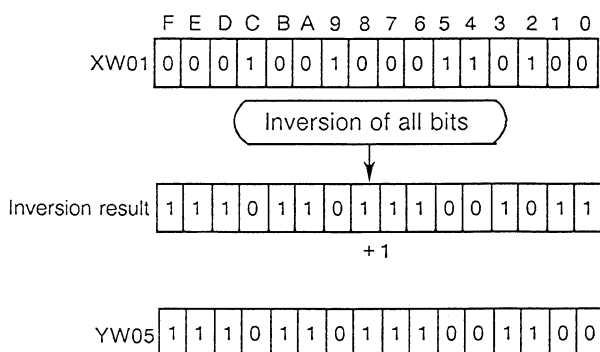
- Only a numeric value can be used for test data ②.
- The number of steps used for this instruction depends on the constant value:
② < H0100: 3 steps
② ≥ H0100: 4 steps

NEG		Two's Complement (FUN046)													
E x p r e s s i o n	Input	<div> <div>Ⓐ</div> <div>NEG</div> <div>Ⓑ</div> </div> <div>Execution output</div> <div> <div>H</div> <div></div> <div></div> <div></div> <div></div> <div></div> </div> <div> <div>H</div> <div></div> <div></div> <div></div> <div></div> <div></div> </div> <div>Data displayed</div>												Number of Steps	
														3	
F u n c t i o n	Calculates the two's complement of the contents of register Ⓐ and stores it in register Ⓑ.							Input	Operation						Output
								OFF	No execution						OFF
								ON	Execution						ON
O p e r a n d	Symbol	Name	R	X	Y	Z	RW	XW	YW	ZW	D	T	C		Constant
	Ⓐ	Operation data					<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>		
	Ⓑ	Two's complement data					<input type="radio"/>		<input type="radio"/>	<input type="radio"/>	<input type="radio"/>				

Sample Program



Operation



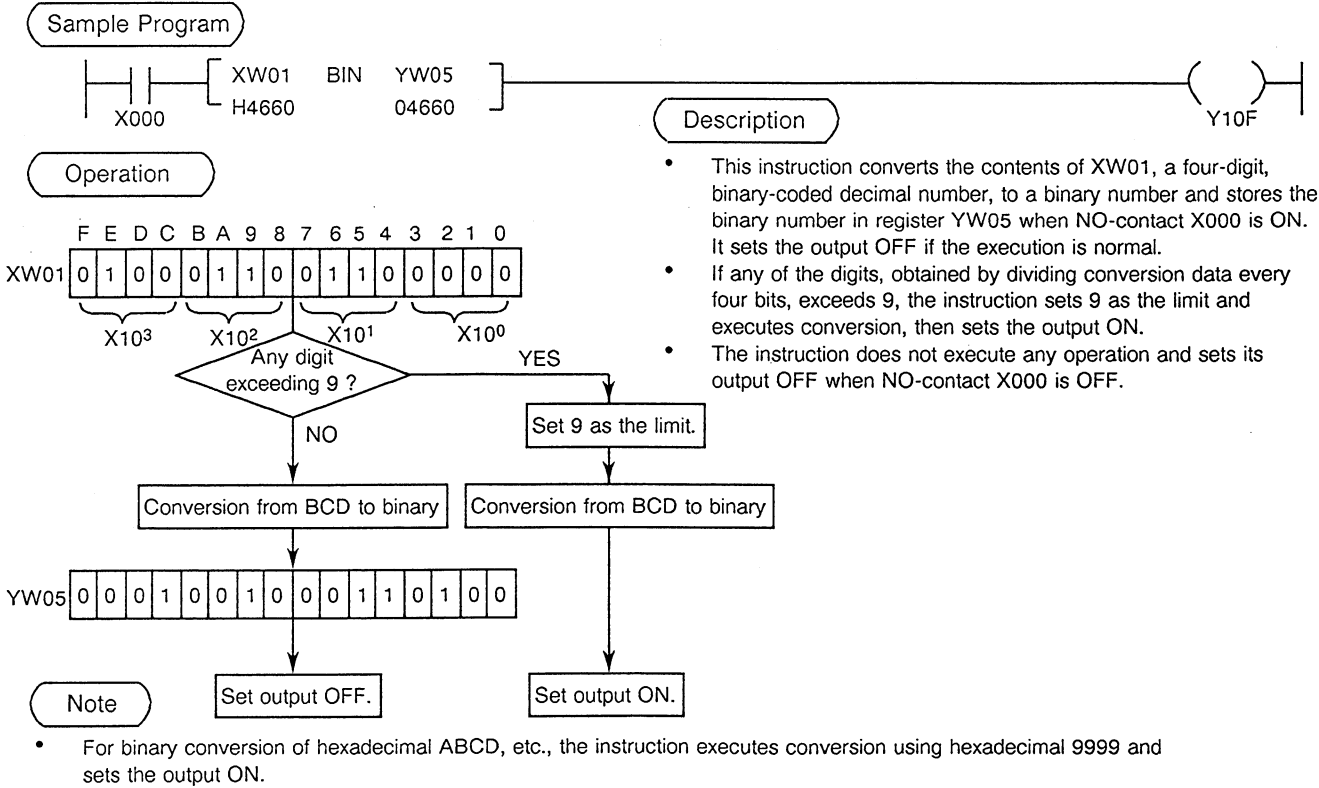
Description

- This instruction calculates the two's complement (inverted data + 1) of register XW01 and stores it in register YW05, then sets the output ON, when NO-contact X000 is ON.
- The instruction does not execute any operation and sets its output OFF when NO-contact X000 is OFF.

7. Instructions

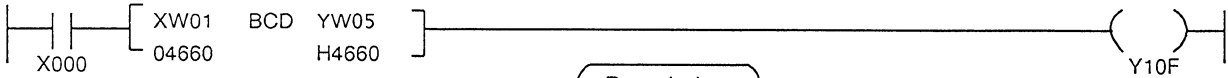
7.6 Data conversion instructions

BIN		Binary Conversion (FUN050)															Number of Steps	
E x p r e s s i o n	Input	BIN															3	
		Error output																
F u n c t i o n	H	Data displayed																
O p e r a n d	Symbol	Operation data															Constant	
		Binary data																
F u n c t i o n	Input	Operation															Output	
		No execution															OFF	
O p e r a n d	ON	Execution															OFF	
																	ON	
O p e r a n d	Symbol	Name															Constant	
		Operation data																
O p e r a n d	Symbol	Name															Constant	
		Binary data																
O p e r a n d	Symbol	Name															Constant	
		Binary data																
O p e r a n d	Symbol	Name															Constant	
		Binary data																

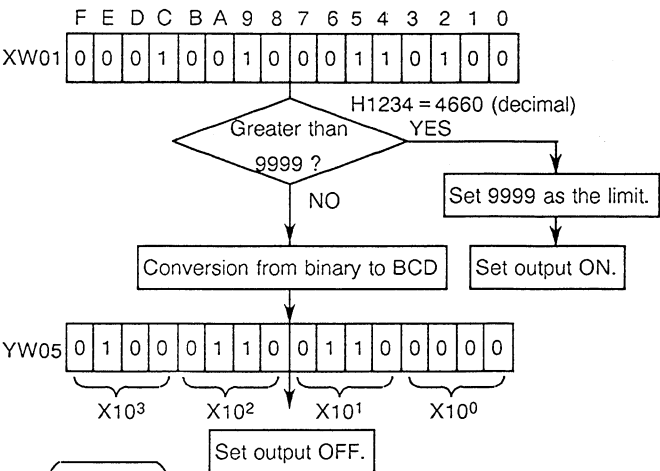


BCD1		BCD Conversion (FUN051)															
Expression	<div><div>Input</div><div><div>Ⓐ</div><div>BCD1</div><div>Ⓑ</div></div><div>Error output</div><div><div></div><div></div><div></div><div></div><div></div><div></div></div><div>H</div><div><div></div><div></div><div></div><div></div><div></div><div></div></div><div>Data displayed</div></div>														Number of Steps		
															3		
Function	Converts the contents of register Ⓐ to four-digit (BCD) data and stores the BCD data in register Ⓑ.							Input		Operation						Output	
								OFF		No execution						OFF	
								ON		Execution				Normal		OFF	
														Error		ON	
Operand	Symbol	Name	R	X	Y	Z	RW	XW	YW	ZW	D	T	C			Constant	
	Ⓐ	Operation data (binary)					○	○	○	○	○	○	○				
	Ⓑ	BCD data					○		○	○	○						

Sample Program



Operation



Note

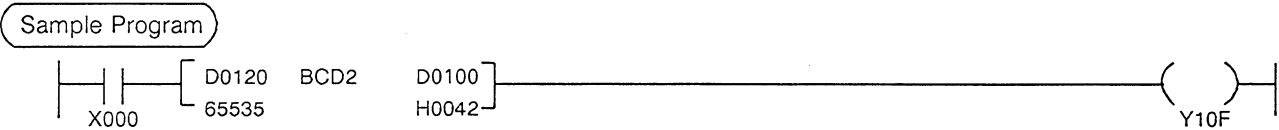
- The instruction's conversion range is from 0 to 9999 (hexadecimal 0 to 270F)
- The instruction is applicable only to single-length BCD conversion. (For double-length data conversion, see FUN52, BCD2.)

Description

- This instruction converts the contents of XW01 to four-digit, binary-coded decimal data and stores it in register YW05 when NO-contact X000 is ON. It sets the output OFF if the execution is normal.
- If data to be converted is greater than 9999, the instruction sets 9999 as the limit and executes conversion, then sets the output ON.
- The instruction does not execute any operation and sets its output OFF when NO-contact X000 is OFF.

7. Instructions

BCD2		Double-Length BCD Conversion (FUN052)																
E x p r e s s i o n	<div>Input [Ⓐ BCD2 Ⓑ] Execution output</div> <div><div><div></div><div></div><div></div><div></div><div></div></div><div>H <div><div></div><div></div><div></div><div></div><div></div></div></div></div> <div>Display of data without sign</div>															Number of Steps		
																3		
F u n c t i o n	Converts double-length data of registers Ⓐ and Ⓐ + 1 to binary-coded decimal (BCD) data of up to 10 digits and stores the BCD data in registers Ⓑ , Ⓑ + 1 and Ⓑ + 2.							Input		Operation							Output	
								OFF		No execution							OFF	
								ON		Execution							ON	
O p e r a n d	Symbol	Name	R	X	Y	Z	RW	XW	YW	ZW	D	T	C			Constant		
	Ⓐ	Operation data (binary)					<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>							
	Ⓑ	BCD data					<input type="radio"/>		<input type="radio"/>	<input type="radio"/>	<input type="radio"/>							



Operation

D0120 (upper word) D0121 (lower word)

HFFFF

HFFFE

BIN → BCD

D0100 (High order) D0101 (Middle) D0102 (Low order)

H0042

H9496

H7294

10¹⁰ 10⁰

Set output ON.

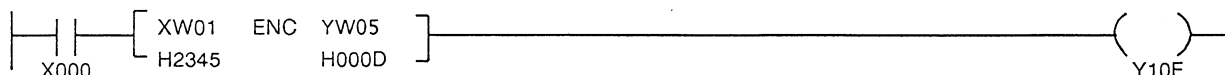
Description

- This instruction converts the double-length binary data of D0120 and D0121 to BCD data of up to 10 digits and stores the BCD data in D0100, D0101, and D0102 when NO-contact X000 is ON. It then sets the output ON.
- The instruction does not execute any operation and sets its output OFF when NO-contact X000 is OFF.

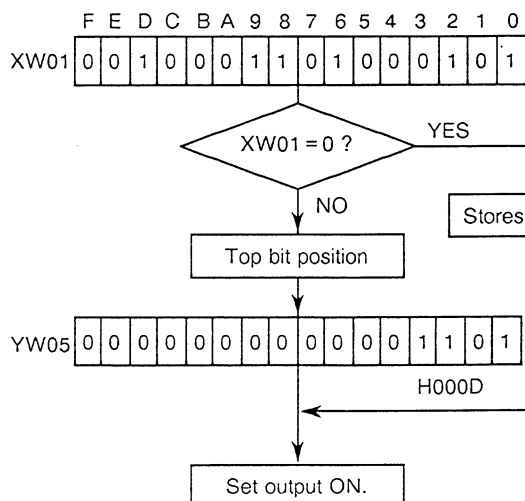
- Note
- The instruction's conversion range is from 0 to 4294967295 (0 to HFFFFFFF) .

ENC		Encode (FUN053)												Number of Steps	
E X P R E S S I O N	Input	ENC												3	
		Execution output													
F U N C T I O N	Input	Operation												Output	
		No execution												OFF	
O P E R A T I O N	Input	Execution												ON	
O P E R A T I O N	Symbol	Name	R	X	Y	Z	RW	XW	YW	ZW	D	T	C	Constant	
	Ⓐ	Encode data					○	○	○	○	○	○	○		
O P E R A T I O N	Ⓑ	Bit position data					○		○	○	○				
O P E R A T I O N															
O P E R A T I O N															

Sample Program



Operation



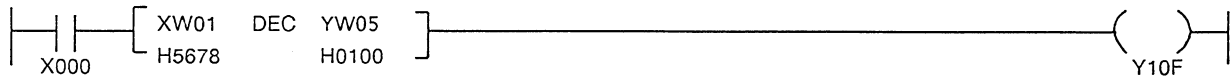
Description

- This instruction finds the bit position of the most significant ON bit in register XW01 and converts the bit position to four bit data and stores it in register YW05 when No-contact X000 is ON, then sets the output ON.
- If register XW01 is 0, the instruction stores HFFFF in YW05 and sets the output ON.
- The instruction does not execute any encode operation and sets its output OFF when NO-contact X000 is OFF.

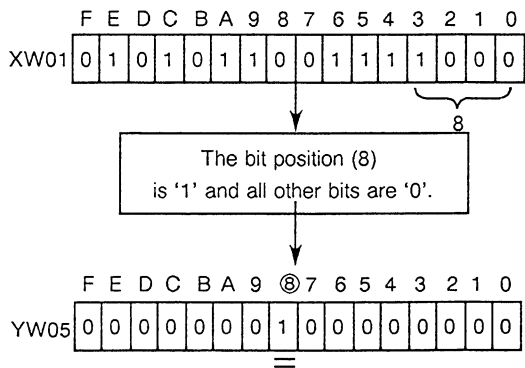
7. Instructions

DEC		Decode (FUN054)														Number of Steps		
E x p r e s s i o n	<div><div>Input</div><div><div>Ⓐ</div><div>DEC</div><div>Ⓑ</div></div><div>Execution output</div><div>H<div><div></div><div></div><div></div><div></div></div></div><div>H<div><div></div><div></div><div></div><div></div></div></div><div>Data displayed</div></div>																3	
	F u n c t i o n	Stores a ' 1 ' in register Ⓑ only at the bit position indicated by the least significant four bits of register Ⓐ and stores '0' at all other bit positions.						Input		Operation						Output		
OFF								No execution						OFF				
ON								Execution						ON				
O p e r a n d	Symbol	Name	R	X	Y	Z	RW	XW	YW	ZW	D	T	C			Constant		
	Ⓐ	Bit position data					<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>					
	Ⓑ	Decode data					<input type="radio"/>		<input type="radio"/>	<input type="radio"/>	<input type="radio"/>							

Sample Program



Operation

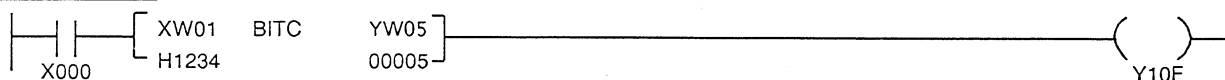


Description

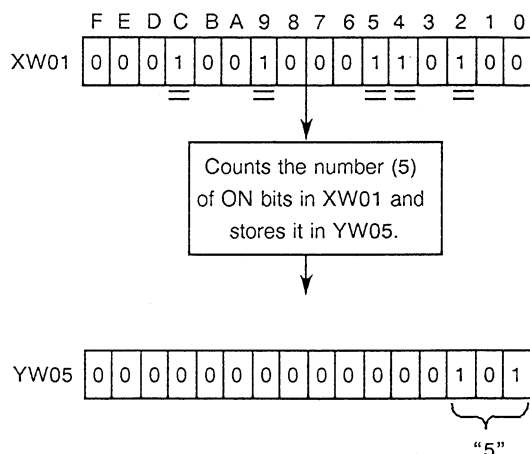
- This instruction stores a '1' in register YW05 at the bit position (8) indicated by the least significant four bits of the contents of XW01, stores '0' at other bit positions when NO-contact X000 is ON, then sets the output ON.
- In the register XW01, only the least significant four bits are valid and all other bits are ignored.
- The instruction does not execute any decode operation and sets its output OFF when NO-contact X000 is OFF.

BITC		Bit Count (FUN055)												Number of Steps	
E x p r e s s i o n	Input	Execution output												3	
		Data displayed													
F u n c t i o n	Counts the number of ON bits in the contents of register ④ and stores the total in register ⑤.												Input	Operation	Output
													OFF	No execution	OFF
													ON	Execution	ON
O p e r a n d	Symbol	Name	R	X	Y	Z	RW	XW	YW	ZW	D	T	C		Constant
	④	Bit count data					○	○	○	○	○	○	○		
	⑤	Total ON bits					○		○	○	○				

Sample Program



Operation



Description

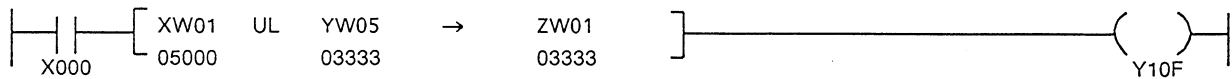
- This instruction counts the number of ON bits in XW01 (5) and stores it in YW05 when NO-contact X000 is ON, then sets the output ON.
- The instruction does not execute any bit count operation and sets its output OFF when NO-contact X000 is OFF.

7. Instructions

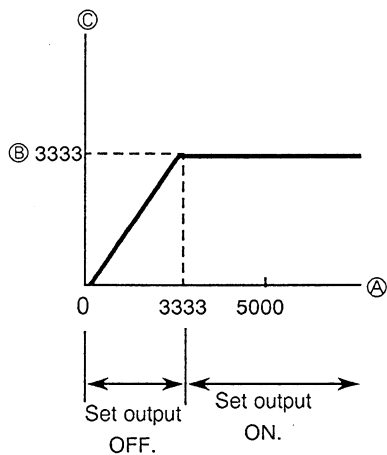
7.7 Special functions

UL		Upper Limit (FUN060)															
E x p r e s s i o n	<div>Input [① UL ② → ③] Decision output</div> <div><div><div></div><div></div><div></div><div></div><div></div></div><div><div></div><div></div><div></div><div></div><div></div></div><div><div></div><div></div><div></div><div></div><div></div></div></div> <div>Data displayed</div>															Number of Steps	
																4	
F u n c t i o n	Limits the value of register ① to those of register ② as an upper limit and stores the value in register ③.							Input		Operation						Output	
								OFF		No execution						OFF	
								ON		Execution				A ≤ B		OFF	
														A > B		ON	
O p e r a n d	Symbol	Name	R	X	Y	Z	RW	XW	YW	ZW	D	T	C			Constant	
	①	Operation data					<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>				
	②	Upper limit value					<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>				
	③	Limit result					<input type="radio"/>		<input type="radio"/>	<input type="radio"/>	<input type="radio"/>						

Sample Program



Operation



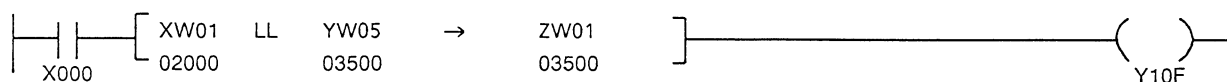
Description

- This instruction limits the contents of register XW01 (5000) to those of register YW05 (3333) as an upper limit and stores the value in register ZW01 when NO-contact X000 is ON. Because 5000 is greater than 3333, the instruction stores 3333 in ZW01 and sets the output ON.
- The instruction sets the output ON if the contents of XW01 exceed the limit value, and otherwise sets the output OFF.
- The instruction does not execute any upper limit operation and sets its output OFF when NO-contact X000 is OFF.

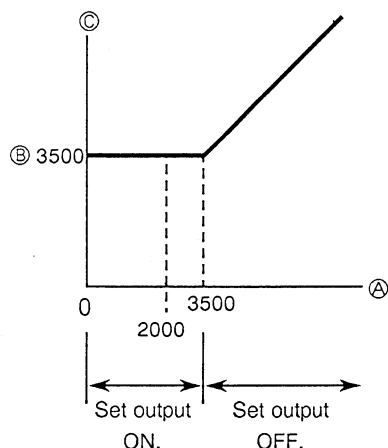
7. Instructions

LL		Lower Limit (FUN061)												Number of Steps	
E x p r e s s i o n	Input	<div> <div>Ⓐ</div> <div>LL</div> <div>Ⓑ</div> <div>→</div> <div>Ⓒ</div> </div> <div>Decision output</div>												4	
		<div> <div></div> <div></div> <div></div> </div> <div>Data displayed</div>													
F u n c t i o n	Limits the contents of register Ⓐ to those of register Ⓑ as a lower limit and stores the value in register Ⓒ.							Input		Operation					Output
								OFF		No execution					OFF
								ON		Execution		$A \geq B$		OFF	
												$A < B$		ON	
O p e r a n d	Symbol	Name	R	X	Y	Z	RW	XW	YW	ZW	D	T	C		Constant
	Ⓐ	Operation data					○	○	○	○	○	○	○		
	Ⓑ	Lower limit value					○	○	○	○	○	○	○		
	Ⓒ	Limit result					○		○	○	○				

Sample Program



Operation



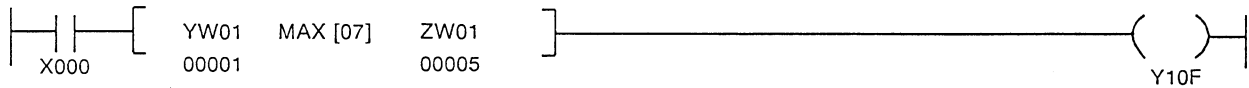
Description

- This instruction limits the contents of register XW01 (2000) to those of register YW05 (3500) as a lower limit and stores the value in register ZW01 when NO-contact X000 is ON. Because 2000 is smaller than 3500, the instruction stores 3500 in ZW01 and sets the output ON.
- The instruction sets the output ON if the contents of XW01 are below the limit value; and otherwise the output is OFF.
- The instruction does not execute any lower limit operation and sets its output OFF when NO-contact X000 is OFF.

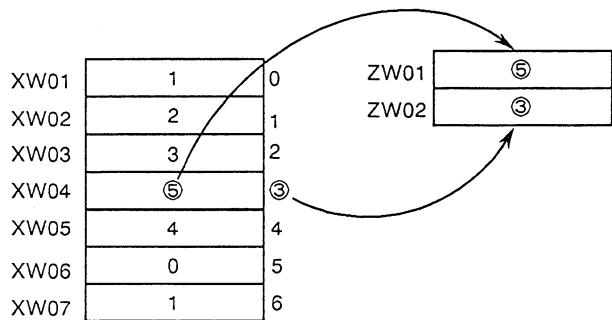
7. Instructions

MAX		Maximum (FUN062)													Number of Steps	
E x p r e s s i o n	Input	MAX [nn] Execution output													4	
		Data displayed														
F u n c t i o n	Finds the maximum value in (nn) registers in a table beginning with register ①, then stores the maximum value in register ② and its table pointer (index value) in register ② + 1.							Input		Operation					Output	
								OFF		No execution					OFF	
								ON		Execution					ON	
O p e r a n d	Symbol	Name	R	X	Y	Z	RW	XW	YW	ZW	D	T	C			Constant
	①	Table's first register					○	○	○	○	○	○	○			
	②	Maximum					○		○	○	○					
	nn	Table size														1 ~ 64

Sample Program



Operation



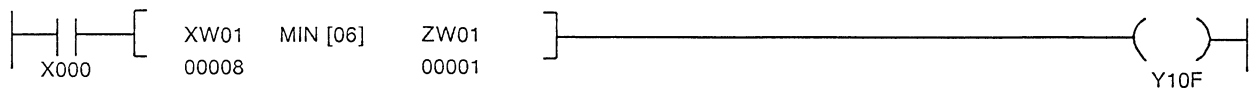
Description

- This instruction finds the maximum value (5) in a seven-register table beginning with register XW01, then stores the value in register ZW01 when NO-contact X000 is ON. It also stores in register ZW02 the table pointer (3) that indicates the location of the maximum, then sets the output ON.
- The instruction does not execute any operation and sets its output OFF when NO-contact X000 is OFF.

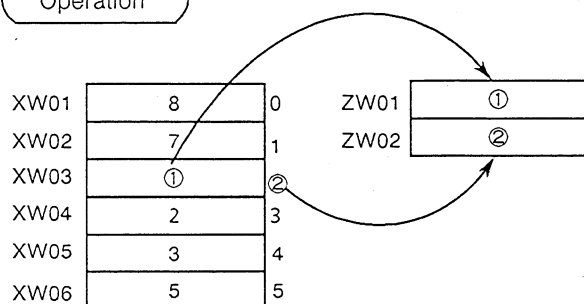
Note

- If there are two or more maximum values, the smallest pointer number among them is stored.
- The specifiable range of table size (nn) is from 1 to 64.

Sample Program



Operation



Description

- This instruction finds the minimum value (1) in a six-register table beginning with register XW01, then stores the value in register ZW01 when NO-contact X000 is ON. It also stores in register ZW02 the table pointer (2) that indicates the location of the minimum counted from the first address of the table, and sets the output ON.
- The instruction does not execute any operation and sets its output OFF when NO-contact X000 is OFF.

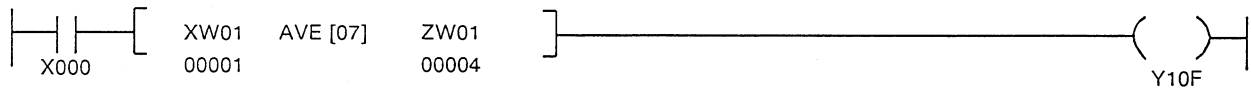
Note

- If there are two or more equal minimum values, the smallest pointer number is stored.
- The specifiable range of table size (nn) is from 1 to 64.

7. Instructions

AVE		Average Value (FUN064)																
E x p r e s s i o n	<div><div>Input</div><div><div>Ⓐ</div><div>AVE</div><div>[nn]</div><div>Ⓑ</div></div><div>Execution output</div></div> <div><div><div></div><div></div><div></div><div></div><div></div></div><div><div></div><div></div><div></div><div></div><div></div></div></div> <div>Data displayed</div>																Number of Steps	
																	4	
F u n c t i o n	Calculates the arithmetic average value of (nn) registers in a table beginning with register Ⓐ and stores it in register Ⓑ.							Input		Operation							Output	
								OFF		No execution							OFF	
								ON		Execution							ON	
O p e r a n d	Symbol	Name	R	X	Y	Z	RW	XW	YW	ZW	D	T	C			Constant		
	Ⓐ	Table's first register					<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>					
	Ⓑ	Average value					<input type="radio"/>		<input type="radio"/>	<input type="radio"/>	<input type="radio"/>							
	nn	Table size														1~64		

Sample Program



Operation

XW01	1	0
XW02	2	1
XW03	3	2
XW04	4	3
XW05	5	4
XW06	6	5
XW07	7	6

→ ZW01

4
Average value

Description

- This instruction calculates the arithmetical average of a seven-register table beginning with register XW01 and stores the average value in register ZW01 when NO-contact X000 is ON, then sets the output ON.
- The instruction does not execute any operation and sets its output OFF when NO-contact X000 is OFF.

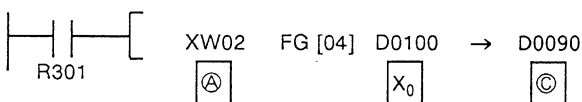
Note

- The specifiable range of table size (nn) is from 1 to 64.

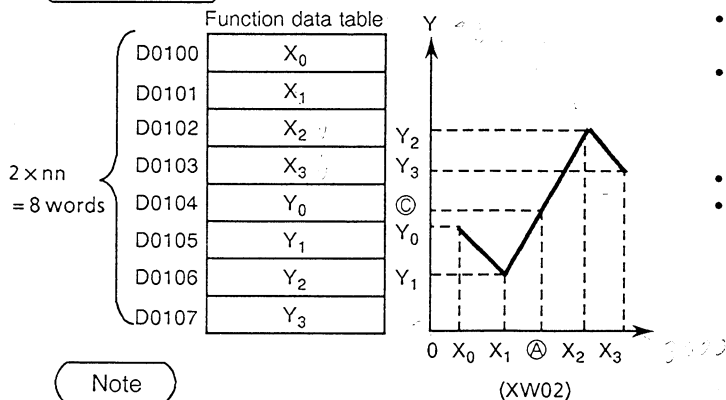
7. Instructions

FG		Function Generator (FUN065)												Number of Steps	
E x p r e s s i o n	Input	<div> <div>Ⓐ</div> <div>FG</div> <div>[nn]</div> <div>Ⓑ</div> <div>→</div> <div>Ⓒ</div> </div> <div>Execution output</div>												5	
		<div> <div></div> <div></div> <div></div> <div></div> <div></div> </div> <div>Data displayed</div>													
F u n c t i o n	Generates the optional function by parameters of 2 × (nn) table beginning with register Ⓑ.						Input		Operation						Output
							OFF		No execution						OFF
							ON		Execution						ON
O p e r a t i o n	Symbol	Name	R	X	Y	Z	RW	XW	YW	ZW	D	T	C		Constant
	Ⓐ	Input register					○	○	○	○	○	○	○		
	Ⓑ	First register for function data table					○	○	○	○	○				
	Ⓒ	Output data					○		○	○	○				
	nn	Number of function data registers													1 ~ 32

Sample Program



Operation



Note

- If $X_{i-1} > X_i$, X_i and Y_i are ignored because of the premise that $X_0 \leq X_1 \leq X_2 \dots \leq X_n$.
- Y_i may satisfy $Y_{i-1} \leq Y_i$ or $Y_{i-1} > Y_i$.

Description

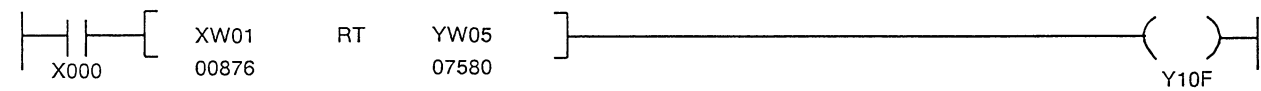
- This instruction executes a function when NO-contact R301 is ON.
- If an X-value Ⓐ specified by register XW02 satisfies the inequality $X_{i-1} < \text{Ⓐ} < X_i$, the following output data Ⓒ is stored in register D0090.

$$\text{Ⓒ} = Y_{i-1} + (\text{Ⓐ} - X_{i-1}) \times (Y_i - Y_{i-1}) / (X_i - X_{i-1})$$
- If $\text{Ⓐ} > X_3$, the value of Y_3 is stored.
- If $\text{Ⓐ} < X_0$, the value of Y_0 is stored.

7. Instructions

RT		Square Root (FUN070)															
E x p r e s s i o n	<div>Input<div>Ⓐ</div><div>RT</div><div>Ⓑ</div>Execution output</div> <div><div><div></div><div></div><div></div><div></div><div></div></div><div><div></div><div></div><div></div><div></div><div></div></div></div> <div>Display of data without sign</div>															Number of Steps	
																3	
F u n c t i o n	Finds the square root of double-length data in registers Ⓐ and Ⓐ + 1 then stores it in register Ⓑ.							Input		Operation						Output	
								OFF		No execution						OFF	
								ON		Execution						ON	
O p e r a n d	Symbol	Name	R	X	Y	Z	RW	XW	YW	ZW	D	T	C			Constant	
	Ⓐ	Operation data					<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>				
	Ⓑ	Square root					<input type="radio"/>		<input type="radio"/>	<input type="radio"/>	<input type="radio"/>						

Sample Program

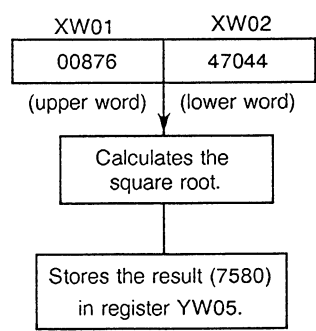


RT

YW05
07580

Y10F

Operation



Description

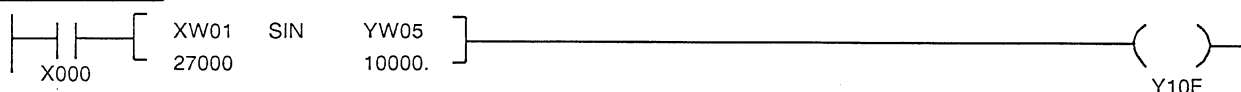
- This instruction calculates the square root of 32-bit binary data (double-length data) in register XW01 and XW02, stores it in register YW05, and sets the output ON when NO-contact X000 is ON.
- The instruction does not execute any operation and sets its output OFF when NO-contact X000 is OFF.

Note

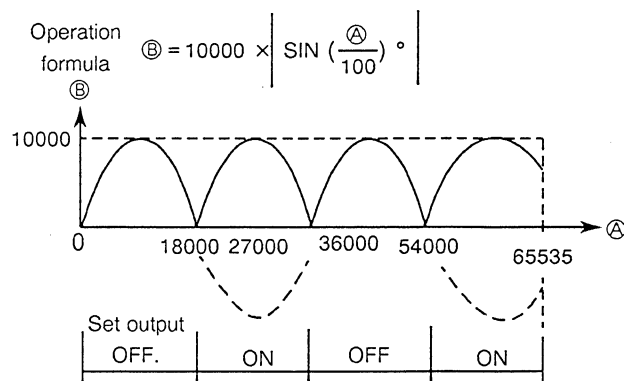
- Register Ⓐ may be an even or odd register.
- Register Ⓐ and Ⓐ + 1 are not altered by the square root extraction.
- Register Ⓐ and Ⓐ + 1 are treated as double-length data.

SIN		Sine Function (FUN071)															
E x p r e s s i o n	<div>Input <div>Ⓐ</div> SIN <div>Ⓑ</div> Execution output</div> <div><div><div></div><div></div><div></div><div></div><div></div></div><div><div></div><div></div><div></div><div></div><div></div></div></div> <div>Data displayed</div>														Number of Steps		
															3		
F u n c t i o n	Stores in register ⑦ the value that is 10000 times the sine of an angle equal to 1 / 100 of data ⑥ in degrees.							Input		Operation						Output	
								OFF		No execution						OFF	
								ON		Execution				The result is a negative value.		ON	
														The result is a positive value.		OFF	
O p e r a n d	Symbol	Name	R	X	Y	Z	RW	XW	YW	ZW	D	T	C			Constant	
	⑥	Operation data					○	○	○	○	○	○	○				
	⑦	Sine function value					○		○	○	○						

Sample Program



Operation



Note

- The range of operation data Ⓐ is from 0 to 65535.
- Relative error is $\pm 0.8\%$ or less.

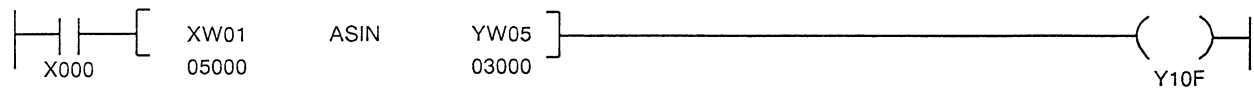
Description

- This instruction calculates the sine of data in register XW01 (27000) from the formula shown at the left, then stores it in register YW05 when NO-contact X000 is ON.
- Because the result is not a negative value (the formula gives an absolute value) but SIN (27000 / 100) is -1, which is negative, the output is set ON.
- If SIN {(A) / 100} is positive, the output is set OFF.
- The instruction does not execute any operation and sets its output OFF when NO-contact X000 is OFF.

7. Instructions

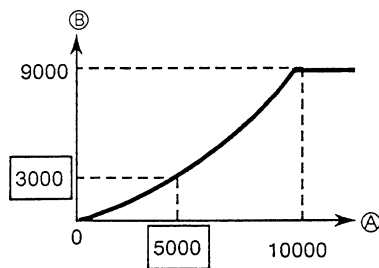
ASIN		Arc Sine Function (FUN072)															
E x p r e s s i o n	<div>Input</div> <div><div>Ⓐ</div><div>ASIN</div><div>Ⓑ</div></div> <div>Execution output</div> <div><div><div></div><div></div><div></div><div></div><div></div><div></div></div><div><div></div><div></div><div></div><div></div><div></div><div></div></div></div> <div>Data displayed</div>															Number of Steps	
																3	
F u n c t i o n	Stores in register Ⓑ the value that is 100 times the arc sine (in degrees) of 1 / 10000 of data Ⓐ.							Input		Operation						Output	
								OFF		No execution						OFF	
								ON		Execution						ON	
O p e r a n d	Symbol	Name	R	X	Y	Z	RW	XW	YW	ZW	D	T	C			Constant	
	Ⓐ	Operation data					○	○	○	○	○	○	○				
	Ⓑ	Arc sine function value					○		○	○	○						

Sample Program



Operation

Operation formula ② = 100 × SIN⁻¹ ($\frac{\text{①}}{10000}$)



Note

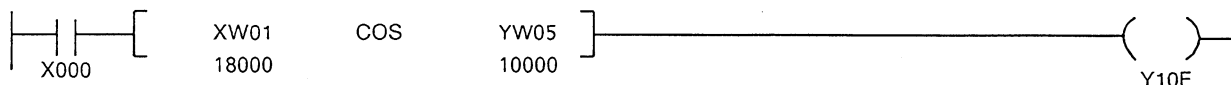
- The range of operation data ① is from 0 to 10000. The range of operation result ② is from 0 to 9000.
- If ① > 10000, operation result ② is limited to 9000.
- Relative error is ± 1% or less.

Description

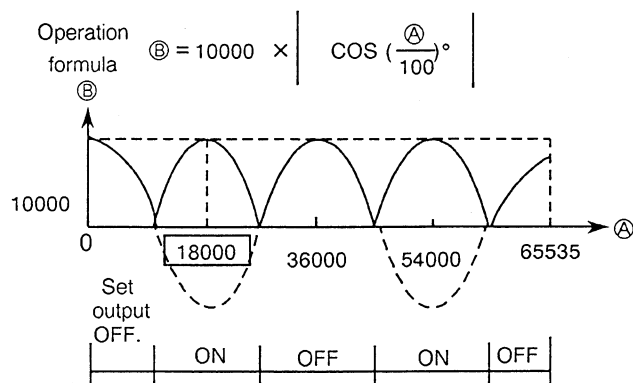
- This instruction calculates the arc sine of data in register XW01 (5000) from the formula shown at the left, stores it in register YW05, and sets the output ON when NO-contact X000 is ON.
- The instruction does not execute any operation and sets its output OFF when NO-contact X000 is OFF.

COS		Cosine Function (FUN073)												Number of Steps	
Expression	Input	Execution output												3	
		Data displayed													
Function	Stores in register ⑥ the value that is 10000 times the cosine value of an angle equal to 1 / 100 of data ④ in degrees.						Input		Operation						Output
							OFF		No execution						OFF
							ON		Execution				The result is a negative value. The result is a positive value.		ON OFF
Operand	Symbol	Name	R	X	Y	Z	RW	XW	YW	ZW	D	T	C	Constant	
	④	Operation data					○	○	○	○	○	○	○		
	⑥	Cosine function value					○		○	○	○				

Sample Program



Operation



Note

- The range of operation data ④ is from 0 to 65535.
- Relative error is $\pm 0.8\%$ or less.

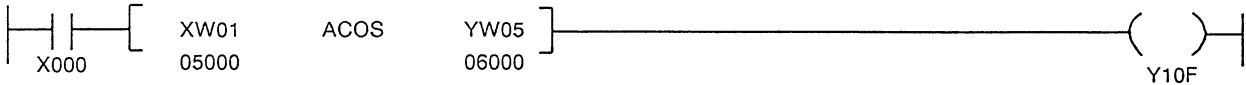
Description

- This instruction calculates the cosine of data in register XW01 (18000) from the formula shown at the left, then stores it in register YW05 when NO-contact X000 is ON.
- Because the result is not a negative value (the formula gives an absolute value) but $\cos(18000 / 100)$ is -1, which is negative, the output is set ON.
- If $\cos\{(A) / 100\}$ is positive, the output is set OFF.
- The instruction does not execute any operation and sets its output OFF when NO-contact X000 is OFF.

7. Instructions

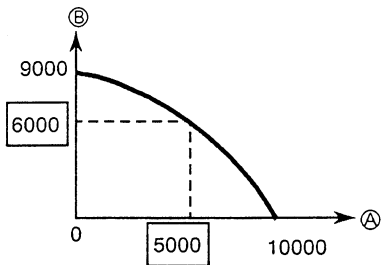
ACOS		Arc Cosine Function (FUN074)															
E x p r e s s i o n	<div>Input <div><div>Ⓐ</div><div>ACOS</div><div>Ⓑ</div></div> Execution output</div> <div><div><div></div><div></div><div></div><div></div><div></div></div><div><div></div><div></div><div></div><div></div><div></div></div></div> <div>Data displayed</div>															Number of Steps	
																4	
F u n c t i o n	Stores in register Ⓑ the value that is 100 times the arc cosine (in degrees) of 1 / 10000 of data Ⓐ.							Input		Operation						Output	
								OFF		No execution						OFF	
								ON		Execution						ON	
O p e r a n d	Symbol	Name	R	X	Y	Z	RW	XW	YW	ZW	D	T	C		Constant		
	Ⓐ	Operation data					<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>				
	Ⓑ	Arc cosine function value					<input type="radio"/>		<input type="radio"/>	<input type="radio"/>	<input type="radio"/>						

Sample Program



Operation

Operation formula ② = 100 × COS⁻¹($\frac{①}{10000}$)



Note

- The range of operation data ① is from 0 to 10000.
- If ① > 10000, operation result ② is limited to 0.
- Relative error is ± 1% or less.

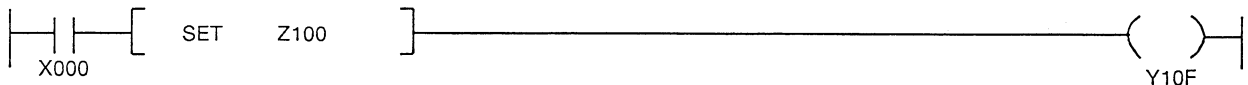
Description

- This instruction calculates the arc cosine of data in register XW01 (05000) from the formula shown at the left, stores it in register YW05, and sets the output ON when NO-contact X000 is ON.
- The instruction does not execute any operation and sets its output OFF when NO-contact X000 is OFF.

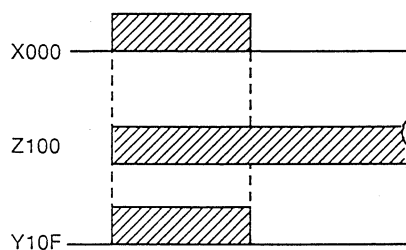
7.8 Special ladder functions

SET		Device Set (FUN080)															
E x p r e s s i o n	Input	<div> <div>SET</div> <div>Ⓐ</div> </div> <div>Execution output</div>														Number of Steps	
																2	
F u n c t i o n	Sets device Ⓐ to ON.							Input		Operation						Output	
								OFF		No execution						OFF	
								ON		Execution						ON	
O p e r a n d	Symbol	Name	R	X	Y	Z	RW	XW	YW	ZW	D	T	C			Constant	
	Ⓐ	Device to be set.	○		○	○											

Sample Program



Operation



Description

- This instruction sets device Z100 to ON and turns the output ON when NO-contact X000 is ON. It maintains the set state even if X000 goes to OFF.
- The instruction does not execute any device set operation and sets its output OFF when NO-contact X000 is OFF.

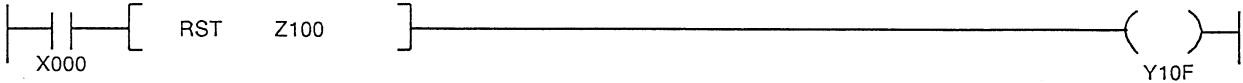
Note

- This instruction is useful when used with the device reset instruction.

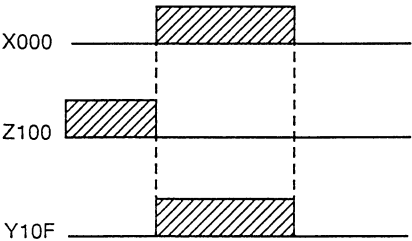
7. Instructions

RST		Device Reset (FUN081)															
E x p r e s s i o n	Input	RST Ⓐ Execution output														Number of Steps	
																2	
F u n c t i o n	Resets device Ⓐ to OFF.						Input		Operation							Output	
							OFF		No execution							OFF	
							ON		Execution							ON	
O p e r a n d	Symbol	Name	R	X	Y	Z	RW	XW	YW	ZW	D	T	C			Constant	
	Ⓐ	Device to be reset	○		○	○											

Sample Program



Operation



Description

- This instruction resets device Z100 to OFF and sets the output ON when NO-contact X000 is ON.
- It maintains the reset state even if X000 goes to OFF.
- The instruction does not execute any device reset operation and sets its output OFF when NO-contact X000 is OFF.

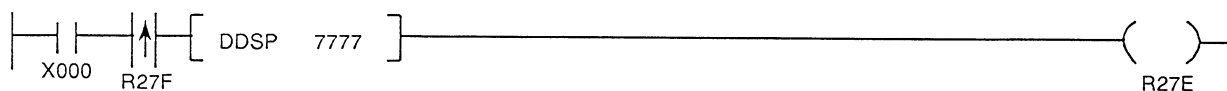
Note

- This instruction is useful when used with the device set instruction.

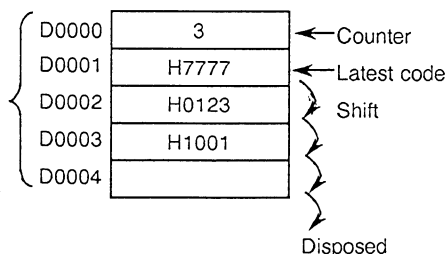
7. Instructions

DDSP		Diagnostic (FUN090)															
E x p r e s s i o n	Fault condition input	<div> <div>DDSP</div> <div>Ⓐ</div> <div>Execution output</div> </div>														Number of Steps	
																2	
F u n c t i o n	When the input comes ON, the diagnostic code set by Ⓐ is converted to BCD code and stored in D0001. And the counter, D0000, is increased by 1.							Input		Operation						Output	
								OFF		No execution						OFF	
								ON		No execution if D0000 ≠ 0 and A = D0001						OFF	
										Execution if D0000 = 0 or A ≠ D0001						ON	
O p e r a n d	Symbol	Name	R	X	Y	Z	RW	XW	YW	ZW	D	T	C			Constant	
	Ⓐ	Diagnostic code														1 ~ 9999	

Sample Program



Operation



DP Display Sample

1 DIAG. 7777
C06

Note

- The programmer can display the diagnostic code. If special coil R63E is output externally, it can be used as an alarm output.
- The error counter (D0000) and special coil R63E are cleared by a user program or by data setting with the programmer.

Description

- Data registers D0000 to D0004 must be reserved in advance.
- When NO-contact X000 is ON, this instruction converts 7777 to a BCD code, stores the code 7777 (hexadecimal) in D0001, sets special coil R63E and the output ON, updates the error counter, and shifts stored codes if the error counter (D0000) is 0 or converted data is not equal to D0001.
If the error counter is not 0 and converted data is equal to D0001 (if new code is the same as the previous one), the instruction executes no operation and turns the output OFF.

7. Instructions

DDSM		Diagnostic With Message (FUN091)															
E x p r e s s i o n	<div><div>Fault condition input</div><div><div>DDSM</div><div>Ⓐ</div><div>Ⓑ</div></div><div>Execution output</div><div>H<div><div></div><div></div><div></div><div></div></div></div><div>Data displayed</div></div>	Number of Steps															
		3															
F u n c t i o n	This instruction adds a message output function to the DDSP function.							Input		Operation						Output	
								OFF		No execution						OFF	
								ON		No execution if D0000 ≠ 0 and A = D0001						OFF	
Execution if D0000 = 0 or A ≠ D0001						ON											
O p e r a n d	Symbol	Name	R	X	Y	Z	RW	XW	YW	ZW	D	T	C			Constant	
	Ⓐ	Diagnostic code data														1 ~ 9999	
	Ⓑ	First register of message data									○						

Sample Program

X000

R27D

DDSM 8888 D0010 H4C49

R27C

Operation

D0000 4 ← Counter

D0001 H8888 ← Latest code

D0002 H7777

D0003 H0123

D0004 H1001

Shift

Message storage area

(MSB)

(LSB)

D0010 L I

D0011 M I

D0012 T —

D0013 O V

D0014 E R

D0015 — —

← ASCII code , 12 characters

Note

If two or more DDSM instructions are used in a program, their individual messages must be stored as 12 characters each in the message storage area. Reserve 6 registers at one message.

Refer to the description of the DDSP instruction.

Description

All functions other than those below are the same as those for the DDSP instruction.

The DDSM instruction can display messages stored in registers from the first register (B) of message data on a programmer together with the diagnostic code.

In the sample on the left, the instruction stores the message LIMIT OVER, corresponding to code 8888, in D0010 to D0015 in ASCII code as shown below, and displays it on the programmer.

D0010 ← H4C49 { L, I }

D0011 ← H4D49 { M, I }

D0012 ← H5420 { T, — }

D0013 ← H4F56 { O, V }

D0014 ← H4552 { E, R }

D0015 ← H2020 { —, — }

DP Display Sample

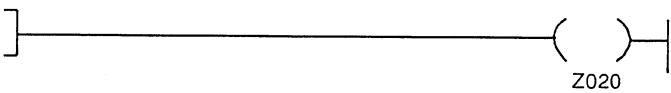
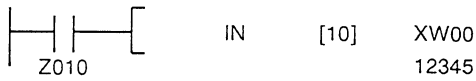
1 DIAG. 8888

C06 LIMIT OVER

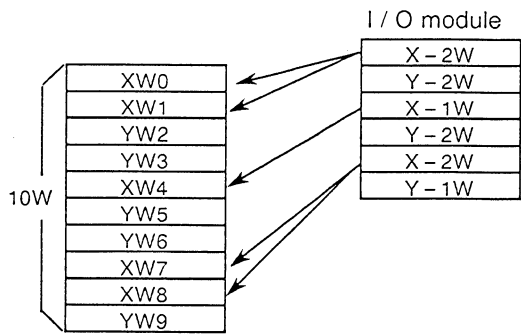
7. Instructions

IN		Immediate Input (FUN096)															
E x p r e s s i o n	<div><div>Input</div><div>[IN [nn] ⊕]</div><div>Execution output</div><div><div></div><div></div><div></div><div></div><div></div></div><div>Data displayed</div></div>															Number of Steps	
																3	
F u n c t i o n	Immediately updates the input data of [nn] registers starting with ⊕.							Input		Operation						Output	
								OFF		No execution						OFF	
								ON		Execution						ON	
O p e r a n d	Symbol	Name	R	X	Y	Z	RW	XW	YW	ZW	D	T	C			Constant	
	nn	Table size														1 ~ 16	
	⊕	First input register						○									

Sample Program



Operation



Note

- Output registers contained in the specified range are ignored.

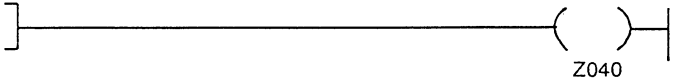
Description

- This instruction immediately inputs data from the input modules to a 10-word area beginning with XW00 register Ⓐ, then sets the output ON when NO-contact Z010 is ON.
- The instruction does not execute any input operation and sets its output OFF when NO-contact Z010 is OFF.

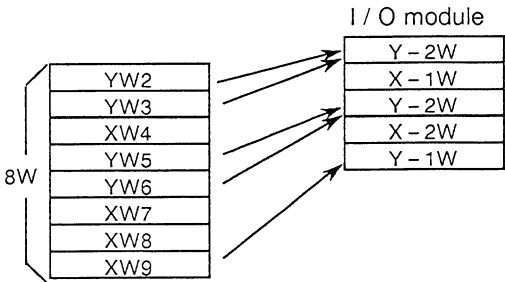
7. Instructions

OUT		Immediate Output (FUN097)														Number of Steps		
E x p r e s s i o n	<div><div>Input</div><div>OUT</div><div>[nn]</div><div>Ⓐ</div><div>Execution output</div></div> <div><div></div><div></div><div></div><div></div><div></div></div> <div>Data displayed</div>																3	
	F u n c t i o n	Immediately updates the output data of [nn] registers starting with Ⓐ.							Input		Operation						Output	
OFF									No execution						OFF			
ON									Execution						ON			
O p e r a n d	Symbol	Name	R	X	Y	Z	RW	XW	YW	ZW	D	T	C			Constant		
	nn	Table size														1~16		
	Ⓐ	First input register							○									

Sample Program



Operation



Description

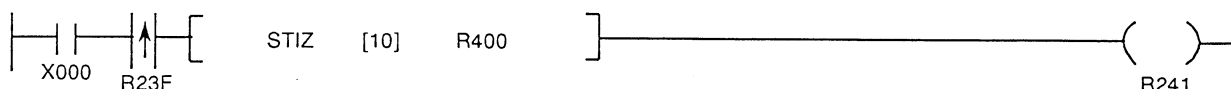
- This instruction immediately outputs the data of 8 registers, beginning with YW02 to the output modules, then sets the output ON when NO-contact Z030 is ON.
- The instruction does not execute any output operation and sets its output OFF when NO-contact Z030 is OFF.

Note

- Input registers contained in the specified range are ignored.

STIZ		Step Sequence Initialize (FUN100)													
E x p r e s s i o n	Input	<div> <div>STIZ</div> <div>[nn]</div> <div>Ⓐ</div> </div> <div>Execution output</div>												Number of Steps	
														3	
F u n c t i o n	Sets device Ⓐ ON and initializes (sets OFF) subsequent nn-1 devices.						Input		Operation						Output
							OFF		No execution						OFF
							ON		Execution						ON
O p e r a n d	Symbol	Name	R	X	Y	Z	RW	XW	YW	ZW	D	T	C		Constant
	Ⓐ	First device of step sequence	○												
	nn	Step sequence device size													1~64

Sample Program



Operation

R400	ON	0
R401	OFF	1
R402	OFF	2
⋮	⋮	⋮
⋮	⋮	⋮
⋮	⋮	⋮
R408	OFF	8
R409	OFF	9 (nn - 1)

Note

- This instruction is used together with the step sequence input instruction (FUN101) and step sequence output instruction (FUN102).
- Refer to the descriptions of the step sequence input and step sequence output instructions.

Description

- This initialization is executed before starting the step sequence.
- The instruction sets the first device R400 of the step sequence ON and sets devices R401 to R409 OFF at the rise of NO-contact X000. It then sets the output ON.
- The instruction does not execute any initialization and sets its output OFF when NO-contact X000 is OFF.

7. Instructions

STIN		Step Sequence Input (FUN101)													
E x p r e s s i o n	Input	Output												Number of Steps	
														2	
F u n c t i o n	Sets the output ON when the input is ON and device ④ is ON.	Input						Operation						Output	
								OFF						No execution	
								ON						A = ON	
								ON						A = OFF	
O p e r a n d	Symbol	Name	R	X	Y	Z	RW	XW	YW	ZW	D	T	C		Constant
	④	Step sequence device	○												

Sample Program

X000

R240

STIZ

[10]

R400

R241

R401

X001

Operation

X000

X001

R241

R400

R401

Step

Note

This instruction is used together with the step sequence initialize instruction (FUN100) and step sequence output instruction (FUN102).

In an input form, up to 14 symbols such as

Description



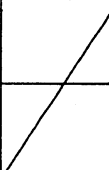


This example program executes the FUN100 step sequence initialize instruction, sets R400 ON, and sets R401 to R409 OFF.

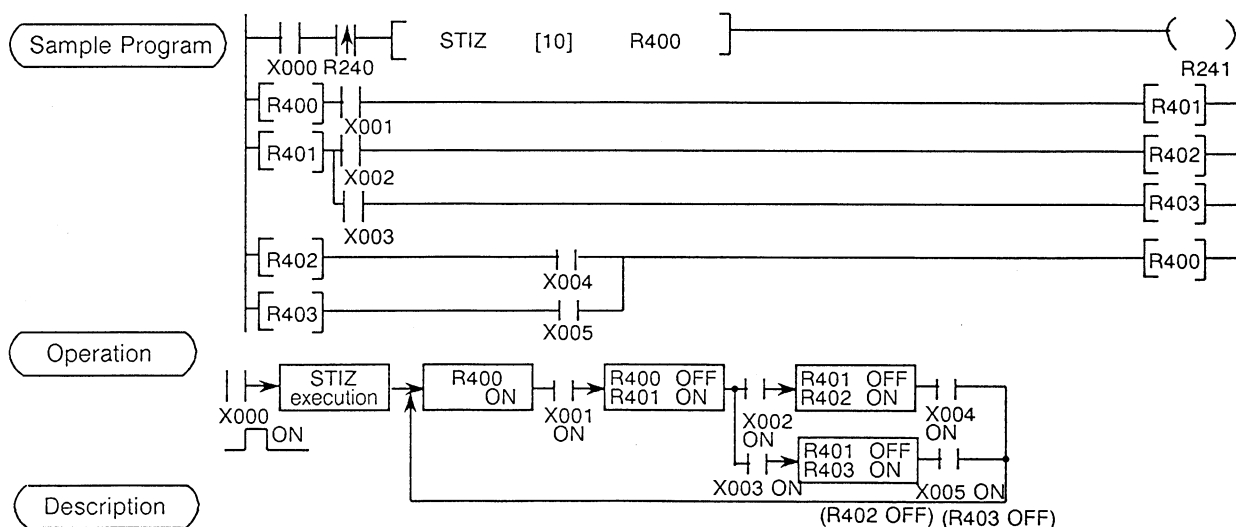
R400 is held ON and R401 is held OFF while X001 is OFF. When X001 goes to ON, R400 goes OFF and R401 goes ON.


[] [] [] and [] []

can be connected in series or in parallel.

Refer to the description of FUN100 and FUN102.

STOT		Step Sequence Output (FUN102)															
E x p r e s s i o n	<div>Input </div>														Number of Steps		
															2		
F u n c t i o n	Clears all step sequence input devices in the same rung, then sets the device  ON when the input is ON.							Input		Operation						Output	
								OFF		No execution							
								ON		Execution							
O p e r a n d	Symbol	Name	R	X	Y	Z	RW	XW	YW	ZW	D	T	C			Constant	
		Step sequence device															



- When the input for this instruction goes ON, the instruction clears all step sequence input devices in the same rung, then sets device  ON.
- When the above sample program is executed, operation shifts in step from R400 to R403.

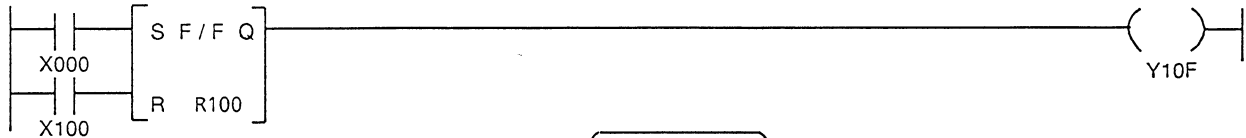
Note

- This instruction is used together with the STIZ (FUN100) and step sequence input instruction (FUN101).
- This instruction may be used in parallel as shown in the sample program.
- Refer to descriptions of FUN100 and FUN101.

7. Instructions

F / F		Flip-Flop (FUN110)															
E x p r e s s i o n	<div><div>Set input</div><div>Reset input</div><div>S F / F Q</div><div>R Ⓐ</div><div>Flip-flop output</div></div>															Number of Steps	
																2	
F u n c t i o n	Sets device Ⓐ to ON when the set output is ON, and resets device Ⓐ to OFF when the reset input is ON. This instruction is a reset-priority flip-flop.						Input		Operation						Output		
							—		(See below.)						Same as the output state of device Ⓐ		
O p e r a n d	Symbol	Name	R	X	Y	Z	RW	XW	YW	ZW	D	T	C			Constant	
	Ⓐ	Flip-flop device	○		○	○											

Sample Program

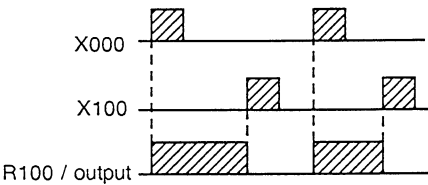


Operation

State of set input S (X000)	State of reset input R (X100)	R100 / output Q
OFF	OFF	Preceding state
OFF	ON	OFF
ON	OFF	ON
ON	ON	OFF

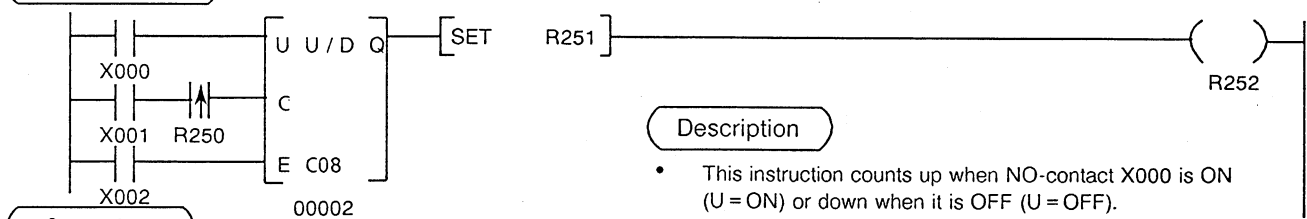
Description

- R100 and the output go ON only when set input X000 is ON and reset input X100 is OFF. When reset input X100 is ON, this instruction sets R100 and the output OFF, regardless of the state of set input X000.
- When both set input and reset input are OFF, R100 and the output remain unchanged.

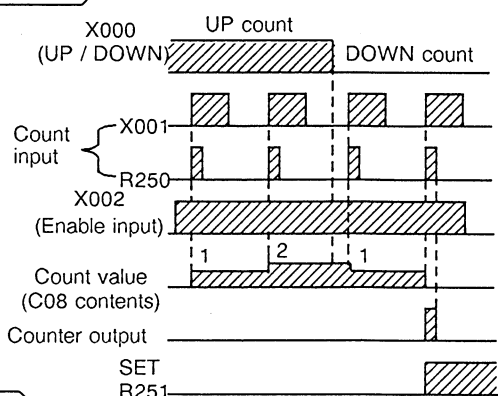


U / D		UP-DOWN Counter (FUN111)																							
E x p r e s s i o n	UP / DOWN select	U	U / D	Q	Counter output										Number of Steps										
	Count input	C													2										
F u n c t i o n	Enable input	E	Ⓐ																						
											Data displayed														
O p e r a t i o n	Counts the number of OFF to ON changes of count input C when the enable input is ON, then sets the output ON when the count reaches the limit. Counting direction can be selected as UP or DOWN.						Input		Operation						Output										
							E = OFF		No counting (count value cleared)						OFF										
							E = ON		When the count value is not the limit						OFF										
									Count value = limit, C = ON						ON										
O p e r a n d	Symbol	Name				R	X	Y	Z	RW	XW	YW	ZW	D	T	C		Constant							
	Ⓐ	Counter register														○									


Sample Program



Operation



Note

- For a count input, counting occurs every scan unless a transitional contact  is provided.

Description

- This instruction counts up when NO-contact X000 is ON (U = ON) or down when it is OFF (U = OFF).
- The instruction counts up or down when the enable input (NO-contact X002) is ON. It clears the count value and sets the output OFF when the enable input is OFF. It does not count and sets the output OFF while the count input is OFF, even if the enable input is ON.
- For counting up, the instruction sets the output ON when the count value reaches the limit of 65535. For counting down, it sets the output ON when the value reaches the limit of 0.
- In the sample on the left, the counter output goes ON only when the count value reaches the limit value, 0. Note that the output goes ON only when both the enable and counter inputs are ON.

7. Instructions

SR		Shift Register (FUN112)															
E x p r e s s i o n	<div><div><div><div>Data input</div><div>Shift input</div><div>Enable input</div></div><div><div>D</div><div>SR</div><div>Q</div></div><div>S [nn]</div><div>E</div><div>Ⓐ</div></div></div> <div>Shift register output</div>															Number of Steps	
																3	
F u n c t i o n	When the shift input comes ON while the enable input is ON, shifts the data in the range, (nn) devices starting with Ⓐ, by one bit.							Input		Operation						Output	
								E = OFF		Clears all target devices						The output always indicates the status of last device.	
								E = ON		S = ON Performs shift							
										S = OFF No execution							
O p e r a n d	Symbol	Name	R	X	Y	Z	RW	XW	YW	ZW	D	T	C			1 ~ 64	
	Ⓐ	First device	○		○	○											
	nn	Bit length															

Sample Program

X000

X010

X020

D

SR

Q

S [32]

E R016

R253

R254

Operation

32 devices

R035 R034 R033

R018 R017 R016

Shifted one bit

X000

Data input

(The output always indicates the status of R035.)

Note

For a shift input, counting occurs every scan unless a transitional contact is provided.

The range of the bit length size (nn) is from 1 to 64.

Description

The instruction clears the devices R016 to R035 when the enable input (NO-contact X020) is OFF.

The instruction executes no shift operation if the enable input is ON and the shift input is OFF.

The instruction inputs data (status of X000) to R016 and shifts the devices one bit if both the enable and shift register inputs are ON.

The output always indicates the status of the last device (R035) .

8. Basic Programming Procedures

8.1 System design overview

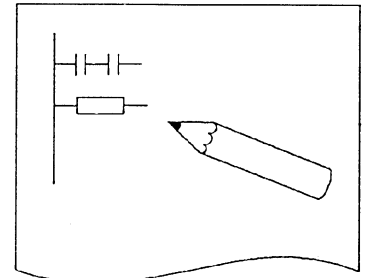
This section describes the basic operations from designing a program for the EX100 to programming the EX100 and debugging the program. You should follow this basic procedure when designing a program.

1. Designing the system

Study the configuration of the systems to be controlled, including the selection of the types of PCs. Study the system's operation and fail-safe sequences thoroughly. Make sure the controllers that you have selected can satisfy the number of I / O points, the memory capacity, required processing speed and other requirements.

2. Designing the program

Having designed the system, it is then necessary to write the program that will run the system. For the EX100, the I / O allocation should be decided first. Then write the EX100 program in accordance with the system operation sequence.



3. Assembling the unit

Select the EX100 memory setting (3K or 4K) and mount the I / O modules according to the planned I / O allocation.

P	C	I	I	
S	P	/	/	
U	O	O		

4. Initializing the system

Set the EX100 operation control switch to the HALT position and turn on the power. Execute the memory clear command from the programmer to initialize the memories in the EX100. It is then necessary to allocate the I / O registers. When these operations have been performed, the EX100 is ready to receive the program that you have written to run your system.

5. Entering the program

Write / load the program into the EX100. Refer to the appropriate manual for details on this procedure.

6. Debugging the program

After programming the EX100, set the operation control switch to the RUN position so that you can debug the program. Be careful not to damage the field devices or other equipment when simulating or debugging the program.

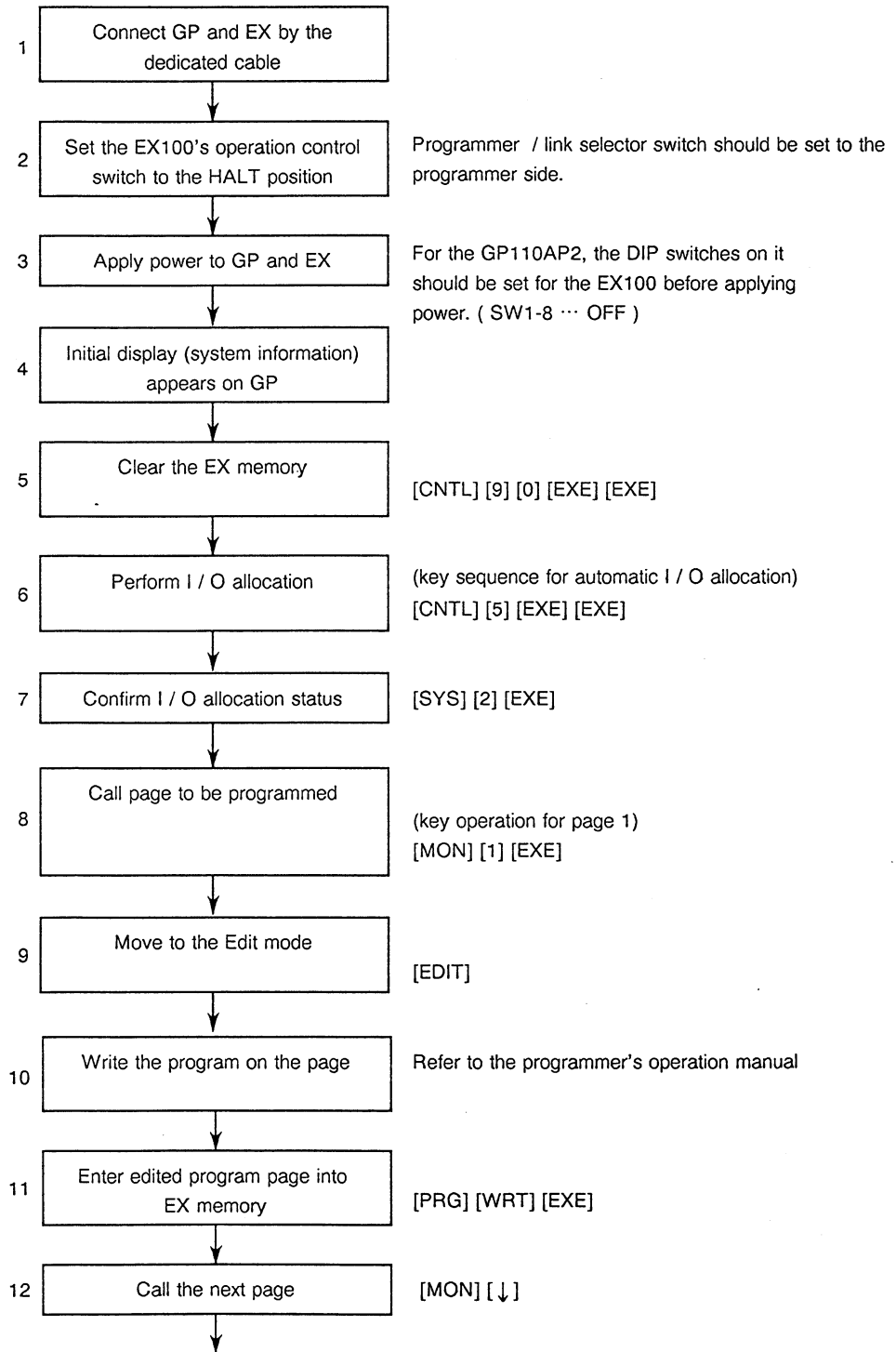
7. Writing the program into the EEPROM

After debugging the program, and BEFORE turning the power to the EX100 off, be sure to write the program into the EEPROM. In the EX100, the program is transferred from the EEPROM to the RAM when the power is turned on. Therefore, you must make sure that the program that has been modified is written into the EEPROM before turning the power off.

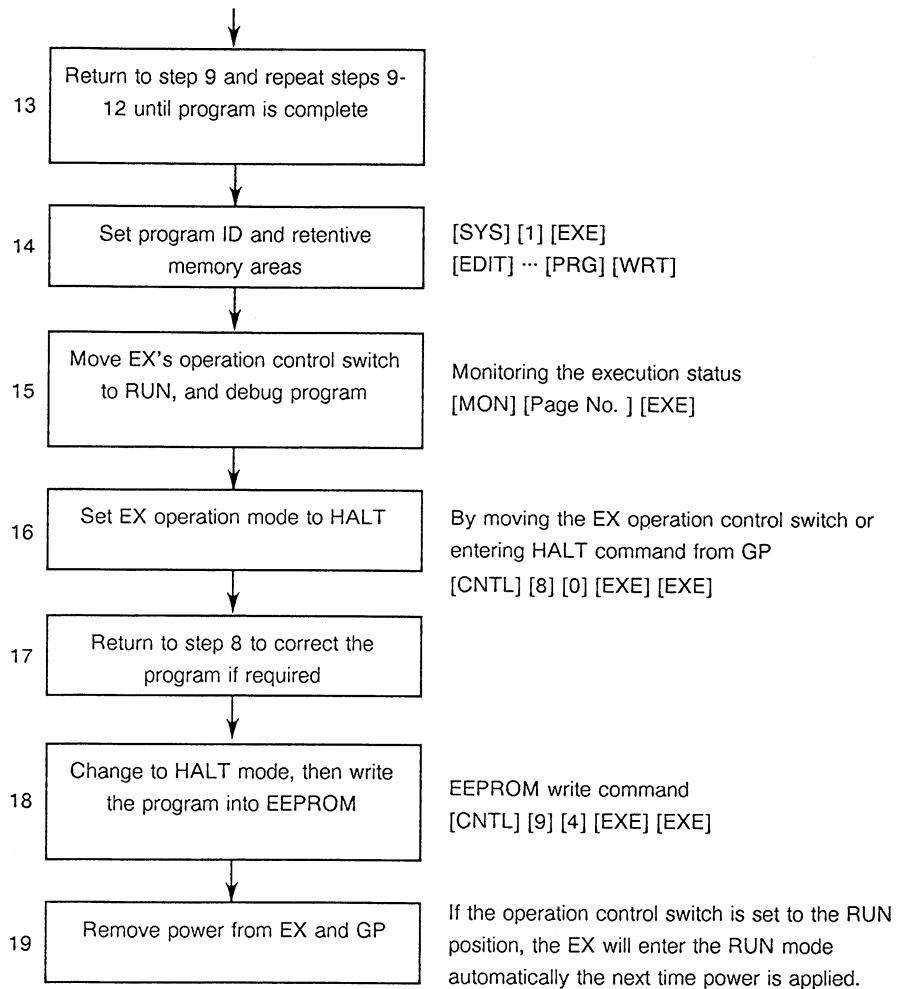
8. Basic Programming Procedures

8.2 Basic programming procedures

The following flowchart shows the programming procedure using the GP110. In the chart, GP and EX represent the GP110 and the EX100 respectively



8. Basic Programming Procedures



NOTE

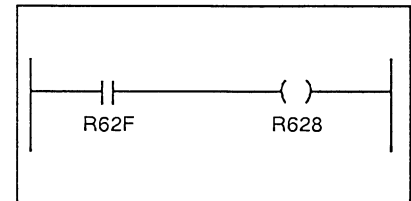
When using a programmer that does not have the EEPROM write command, the special relay R62E of the EX100 should be used to perform step 18. (See 2.6 and 5.1)

9.1 The clock-calendar function

The enhanced version of the CPU module for the EX100 contains a clock-calendar that automatically keeps track of the year, month, day, day of the week, hour, minute, second. This function greatly simplifies scheduled operations and batch processing.

Operating method

To use the clock-calendar function, create the circuit shown on the right on the first page of the program. Creating this circuit assigns the data registers D0005 to D0010 for clock-calendar data.



Calendar registers

	F	...	8	7	...	0	Example
D0005	—					Year	D0005 = H0090
D0006	—					Month	D0006 = H0002
D0007	Week					Day	D0007 = H0227
D0008	—					Hour	D0008 = H0013
D0009	—					Minute	D0009 = H0010
D0010	—					Second	D0010 = H0044

Feb. 27, 1990 (Tuesday)
13:10:44



(1) Clock-calendar data is expressed in two-digit BCD codes.

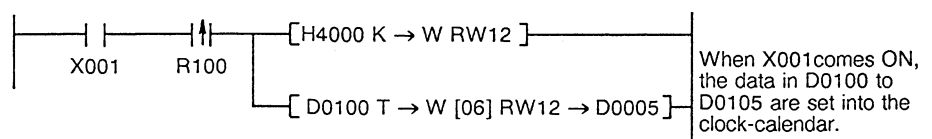
(2) The day of the week is expressed as follows.
0: Sunday, 1: Monday, 2: Tuesday, 3: Wednesday,
4: Thursday, 5: Friday, 6: Saturday

Initializing

To initialize the clock-calendar data, either of the following two methods are available.

- (1) Use the programmer or the computer link to write the initial values into D0005 to D0010.
- (2) Execute the clock-calendar data setting instruction.
(See page 112)

Example



9. Special Functions



- (1) When power is turned off, the clock-calendar is updated by the EX100's built-in capacitor (backup: 7 days / 25°C). If power is not turned on before this backup period has elapsed, the data in the clock-calendar registers may be lost, and may have to be re-initialized.
If the optional battery is used, the clock-calendar will be backed up for one year.
- (2) The accuracy of the clock-calendar is ± 30 seconds per month.

9.2 Forced operation (Automatic RUN-F)

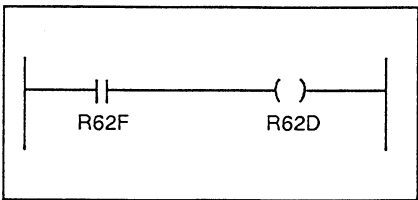
The EX100 can be operated when some of its slots are vacant, provided the I/O allocation for those slots is blank, SP, or OPT. However, if slots whose I / O allocation is not blank, SP, or OPT are left vacant, under normal operation, an error will result when the EX100 makes the I / O response check prior to operation.

Operation of the EX100 is not possible using the following configuration because I / O allocation does not match actual module installation.

Modules mounted	P S	C P U	X 1W	Y 1W	V a c a n t	V a c a n t	X 2W	Y 1W	V a c a n t
I / O allocation			X 1W	Y 1W	X 1W	Y 1W	X 2W	Y 1W	B l a n k

When necessary, however, the EX100 can be operated without all modules being mounted by using its forced operation, Auto RUN-F. This function enables a program to be debugged without the modules actually being installed.

Specifying forced operation
Create the circuit shown on the right on the first page of the program. RUN activation is now ready.



- (1) Although forced operation allows the EX100 to be operated if some modules are not mounted according to I / O allocation, an error results if modules of a different type are mounted.
- (2) This function is identical to the RUN-F command issued from a programmer.

9.3 The write-protect function

The EX100 provides a write-protect function using the program ID. By using the write-protect function, the following operations can be inhibited:

- Program modification, both in the RUN and HALT modes
- Changing the allocation of I / O registers
- Forcing or releasing devices and / or coils, both in the RUN and HALT modes.

Specifying write-protect

To specify the write-protect, set the three lower characters of the program ID to FFF

PROGRAM ID

*	*	*	*	*	*	*	F	F	F
---	---	---	---	---	---	---	---	---	---

*: Unrestricted



NOTE

- (1) This function does work with the computer link.
- (2) The program ID can be changed either in the RUN or HALT mode.
- (3) After specifying the write-protect, it is not possible to modify the program or perform any other writing operation. If writing is attempted, a MODE ERROR will be displayed.

9.4 The hold function

The hold function enables program execution to be stopped, with only input and output updating being executed. It is therefore possible to suspend program execution while holding the output state. Moreover, a desired output state can be established by setting any data in the external output register while in the hold state.

This function is useful for checking external lines or output devices. To enter the hold state, turn on special relay R629 while in the RUN mode. By turning on relay R629 by means of a program instruction, program execution can be stopped when in the desired condition, thus greatly simplifying program debugging.



NOTE

- (1) To reset the EX100 to the RUN mode, simply turn off special relay R629 by using a programmer.
- (2) When in the hold state, the RUN LED on the CPU module will blink.

9. Special Functions

9.5 The EEPROM read / write functions

When the EX100's memory selection is set to the 3K mode, the data of 1024 registers (D0512 to D1535) are stored in the EEPROM. (see 5.2)

The EEPROM read and write instructions enable access to the data stored in the EEPROM from the user program. This allows storage and read-out of the variable data, providing completely maintenance-free back up operation.

NOTE

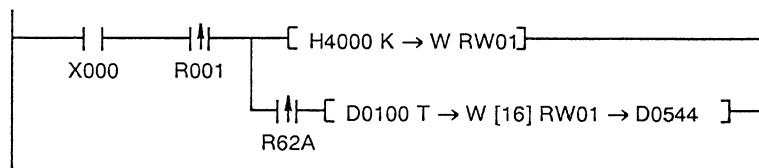


(1) This function is valid only when the EX100's memory selection is set to the 3K mode.

(2) See pages 109 and 118 for detailed explanations of these instructions.

Sample programs

- When X000 comes ON, 16 words of the data in D0100 to D0115 are written into D0544 to D0559 of the EEPROM.

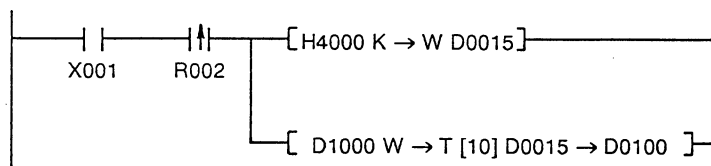


NOTE



The EEPROM write instruction functions by combining the transitional contact of R62A and FUN003.

- When X001 comes ON, 10 words of data in D1000 to D1009 of the EEPROM are read out to D0100 and subsequent registers of the RAM.

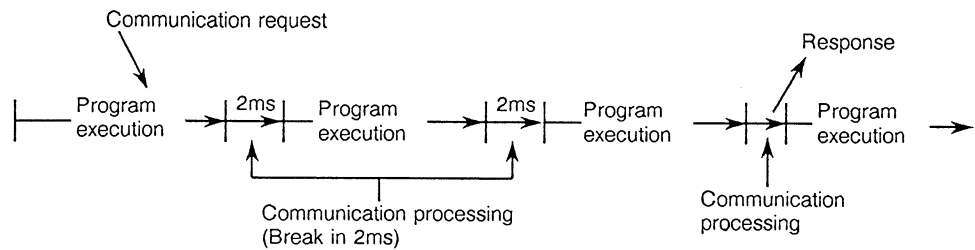


9.6 Communication priority mode

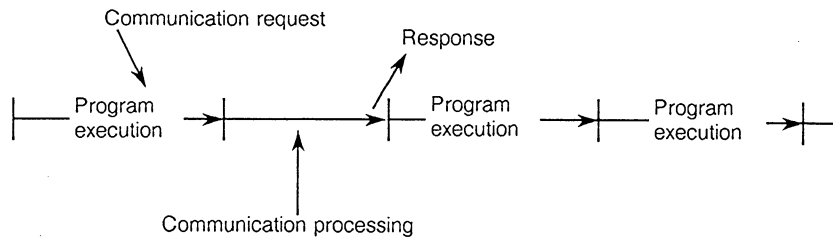
Normally the EX100 processes communication with the computer link or the programmer within a period of 2 ms at the end of each scan in order to provide high speed scanning. (See "Normal mode", below)

On the other hand, in the communication priority mode, the EX100 processes communication without a break after program execution. This mode is effective for applications requiring a more rapid response to the computer. (See "Communication priority mode", shown below)

- Normal mode

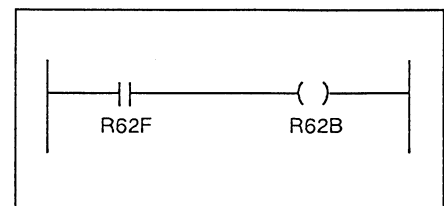


- Communication Priority mode



Selecting method

To select the communication priority mode, create the circuit shown on the right on the first page of the program.



9. Special Functions

9.7 Data input / output functions for special modules

To facilitate use of the special modules, the EX100 provides special instructions to exchange a large amount of data between the EX100's CPU and special modules such as the motion control module.

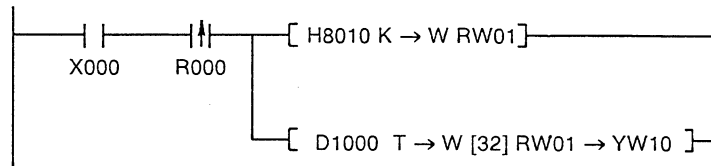
Thus, a complicated handshake program is not required for exchanging data.



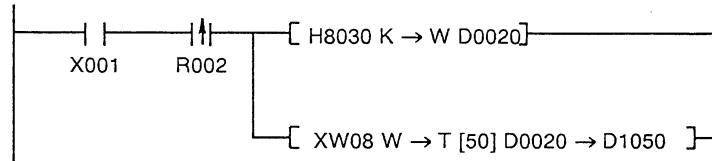
NOTE See pages 114 and 121 for detailed explanations of these instructions.

Sample programs

- When X000 comes ON, 32 words of data in D1000 to D1031 are transferred to locations beginning at address H10 of the internal memory of the special module that is allocated to YW10.



- When X001 comes ON, 50 words of data stored in address H30 and thereafter of the internal memory of the special module (XW08) are read out to D1050 and subsequent registers.



- NOTE**
- (1) Up to 128 words can be input / output at a time. However, if the computer link is used at a transmission rate of 9600 bps, simultaneous input / output capacity will be limited to 64 words or less.
 - (2) Refer to the user's manuals for special modules for details on the mapping of the internal memory and application examples.

10. Maintenance and Checks

10.1 Daily checks

Check the following items each day to ensure that the EX100 is in proper operating condition.

Item		Check		Corrective measure
Mounting		Tightness of the EX100 rack's mounting screws.		Tighten screws as necessary.
		Hooks on the I / O modules are securely engaged.		Push the module toward the rack until the hook is securely engaged.
		Detachable terminal blocks are securely engaged.		Secure the terminal blocks as necessary.
Connections		Tightness of terminal screws on power cable and I / O wiring.		Tighten screws as necessary.
		Tightness of expansion cable connectors.		Secure connectors as necessary.
Status indicators (LEDs)	Power supply module	POWER	Must be lit when power is on.	See Section 11 Troubleshooting procedures.
	CPU module	RUN	Must be lit EX100 is operating.	
		CPU	Must be lit when the CPU is normal.	
		I / O	Must be lit when the I / O is normal.	
		COM	Must blink when EX communicates with a peripheral device.	
	Input modules	Must be lit if input is on; off when input is off.		
	Output modules	Must be lit when output is on; off when output is off.		

10. Maintenance and Checks

10.2 Check the following items at least once every six months, or when
Periodic checks the operating environment changes.

Item	Check	Criterion
Power supply	Check power supply voltage at terminals.	Must be within specified range.
	Check the wiring screws.	Must not be loose.
	Visually check the wires and cables.	Must not be damaged.
I / O	Check the voltage at the I / O terminals.	Must be within specified range.
	Turn on input equipment and check that LEDs light.	Each corresponding LED must light.
	Forcibly turn on output and check that LEDs light.	Each corresponding LED must light.
	Check I / O module mounting.	Each module must be attached securely.
	Check mounting of detachable terminal blocks.	Must not be loose.
	Check wiring screws.	Must not be loose or be in contact with adjoining parts.
	Visually check wires and cables.	Must not be damaged.
	Check terminal block and base connectors.	Must be clean.
Environment	Check temperature, humidity, vibration, and dust levels, etc.	Must be within specifications.
Mounting	Check EX100's mounting.	Must not be loose.
Programs	Check program. Compare it with master program, if available.	There must no program errors.
Optional-battery check	Replace the battery once a year.	Wipe the new battery with a clean, dry cloth and check that it is inserted correctly.

10. Maintenance and Checks

10.3 Spare parts to keep in stock

Keep the following spare parts in stock so that system down time will be minimal in the event of a failure.

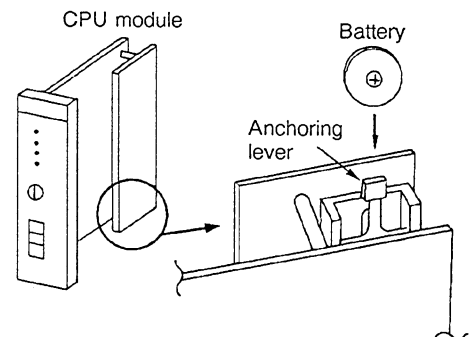
- Input and output modules
One of each type of input and output module used should be stocked as a back-up.
- Fuses
Spare fuses for each module used should be kept in stock. The following table lists fuse specifications.

Module		Fuse rating	Part No.	Quantity
Power supply	PS31	125V / 2A, normal fusing	EX10• SFB20	1
	PS51	125V / 2A, normal fusing	EX10• SFB20	1
	PS61	250V / 1A, normal fusing	EX10• SFB10	1
Output	DO31	250V / 5A, quick fusing	EX10• SFA50	1
	DO32	250V / 2A, quick fusing	EX10• SFA20	4
	AC61	250V / 2A, normal fusing	EX10• SFC20	3

10.4 Removing and installing the optional battery

To remove and install the optional battery:

1. Turn off power and remove the CPU module.
2. Locate the battery holder on the lower part of the CPU module.
3. If the holder already contains a battery, lift the anchoring lever and remove the old battery using a screwdriver or similar tool.
4. Wipe the exterior of the new battery with a clean cloth, then insert the battery with its positive terminal facing toward the anchoring lever.



Battery type: CR2032 (Toshiba)
Voltage: 3V
Capacity: 180mAh
Replacement
period: 1 year (recommended)

11.1 Troubleshooting procedure

If a system failure occurs, it is important to accurately determine the cause of the trouble. It is of primary importance to first determine whether the problem lies with the machinery or with the controller. In many cases, one problem causes secondary problems. When determining the cause of the initial problem, it is important to consider the system as a whole.

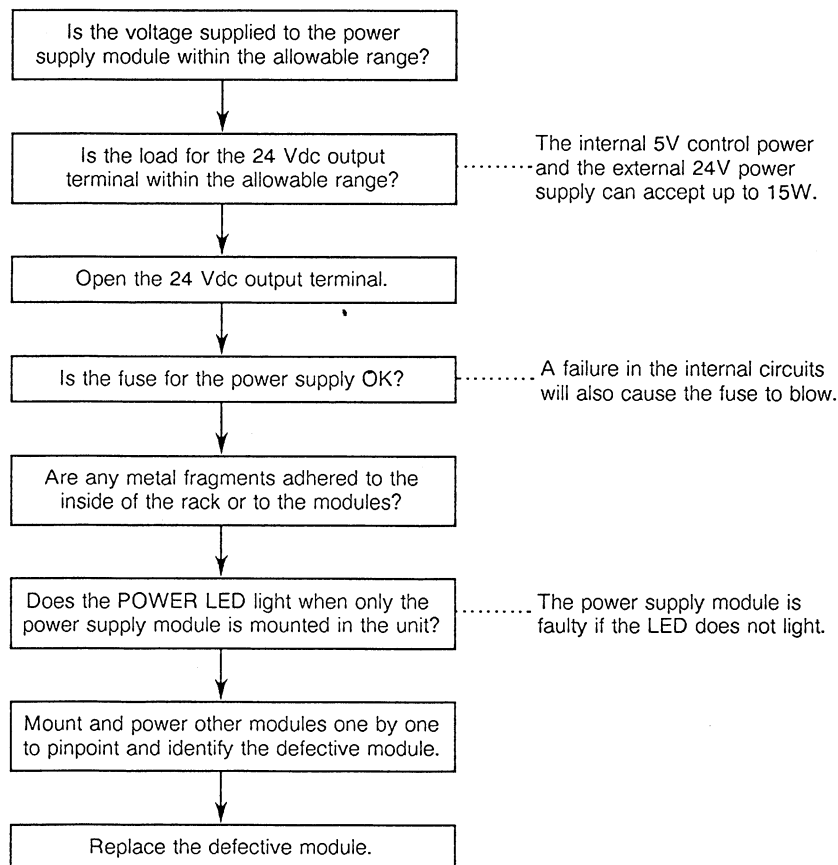


CAUTION When troubleshooting, take the following precautions to prevent human injury and / or damage to the EX100:

- (1) Disconnect power from the EX100 before changing modules or disconnecting cables.
- (2) Disconnect power to all I / O devices before changing modules.

Faulty power supply

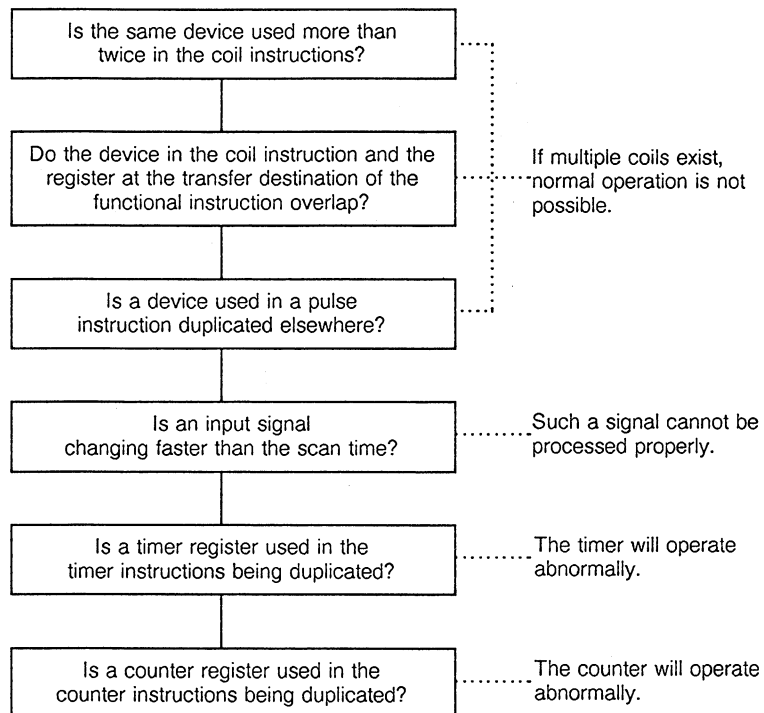
The following flowchart lists troubleshooting procedures for when the POWER LED does not light after connecting power to the EX100.



11. Troubleshooting

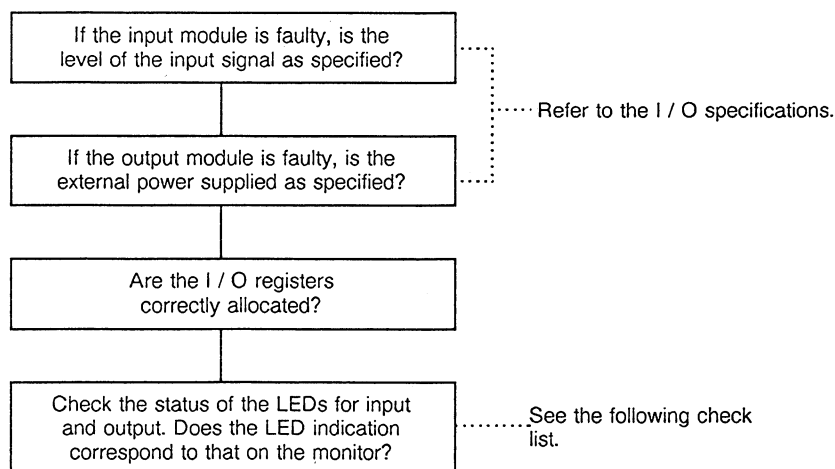
Faulty operation due to a software problem

If the system runs but the program execution is faulty, there is good reason to suspect that the problem is caused by the software. Check the following items.

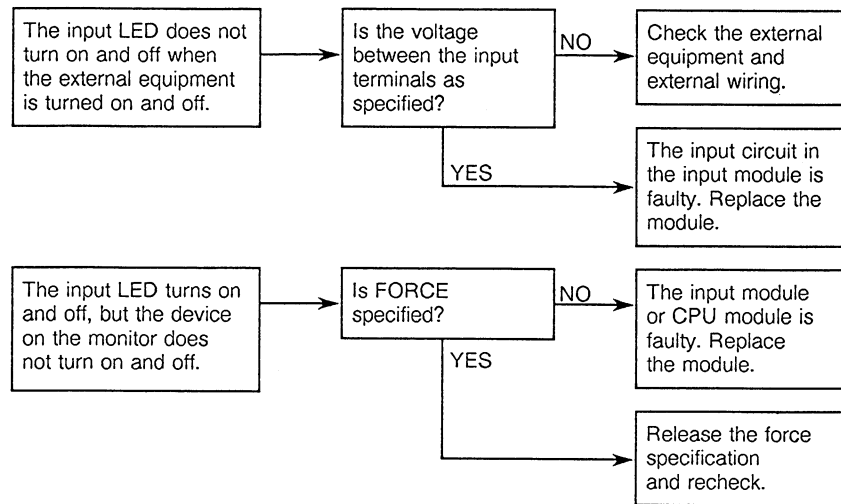


Faulty operation due to a hardware problem

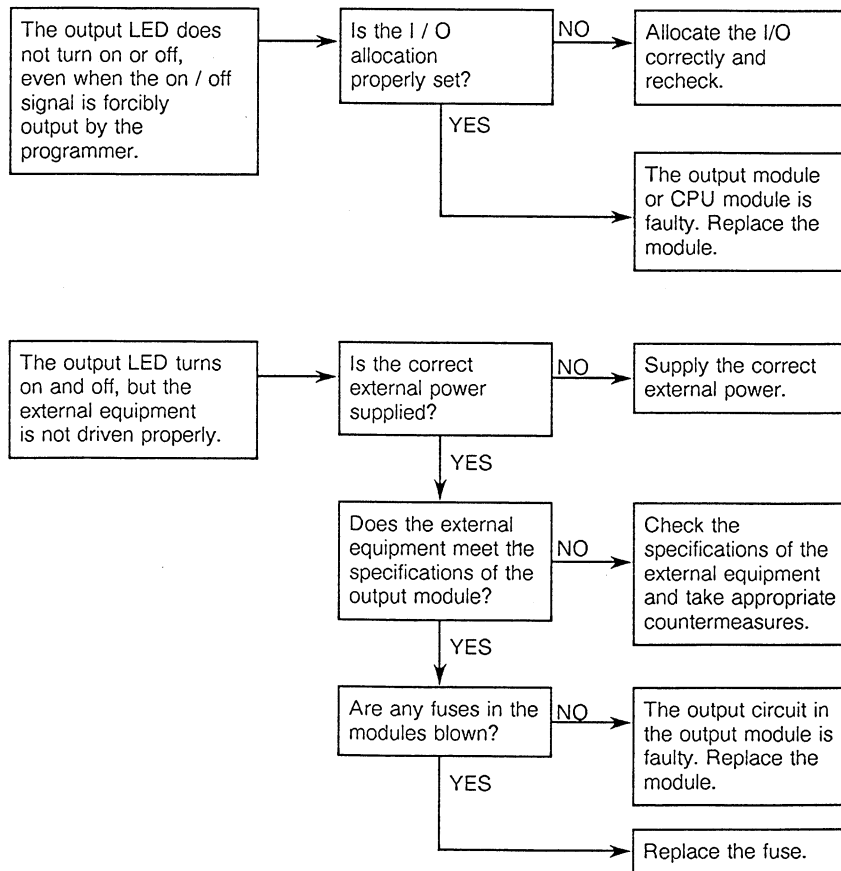
If the input data is not read properly when the system is operating, or if it is not possible to output data although the monitor indicates that the output is operating, check the following items to pinpoint the cause.



Input check



Output check



11. Troubleshooting


11.2 List of items for diagnostic check The following table lists the items checked during the EX100's diagnostic check. If the system does not operate properly even when the POWER LED on the power supply module is lit, use the following table as a guide for troubleshooting.






Item	Phenomenon	Error reg'd	Displayed message	
Illegal instruction	Illegal instruction detected during execution.	Yes	ILLEGAL INST	
Watchdog timer trouble	The watchdog timer cannot be reset within 350 ms.	Yes	WD-TIMER	
I / O bus trouble	A problem has been detected in the I / O bus, or the CPU is mounted in the wrong slot.	Yes	IO BUS ERROR	
I / O response error (When activated by the RUN switch)	An I / O module is not responding.	Yes	IO NO SYNC	
I / O response error (When activated by a programmer)	An I / O module is not responding.	No	IO NO SYNC	
I / O reference error (When activated by the RUN switch)	The I / O register allocation does not correspond to the mounted positions of the modules.	Yes	IO UNMATCH	
I / O reference error (When activated by a programmer)	The I / O register allocation does not correspond to the mounted positions of the modules.	No	IO UNMATCH	
CPU trouble	There is a problem in the hardware.	Yes	CPU ERROR	
EEPROM trouble	The EEPROM data is abnormal.	Yes	ROM ERROR	
Scan time over	A program scan has exceeded 200 ms.	Yes	SCAN OVER	
Program error (When activated by the RUN switch)	The program data is abnormal or contains an error.	Yes	*1)	
Program error (When activated by a programmer)	The program data is abnormal or contains an error.	No	*1)	
Transmission trouble	A problem has been detected in communication with a peripheral device.	No	*2)	
TOSLINE trouble	A problem has been detected in the TOSLINE transmission.	No	TL	
Computer link trouble	A problem has been detected in the computer link.	No	CL	



The messages listed in the above table are displayed on a peripheral device. The messages displayed by the handy programmer HP are used here as an example.

11. Troubleshooting

○ Not lit ● Lit  Blinking

	Special relay	LED indication				Action to be taken
		RUN	CPU	I/O	COM	
	R636	○		●	—	Cycle power off then on, then check and modify the program.
	R630	○	○	●	—	Cycle power off then on and check the program status. Take steps to reduce or eliminate signal interference.
	R634	○	●	○	—	Check the slot location of the CPU. Check the contact between the base connector and the module.
	R634	○	●	○	—	Check that the I / O modules are securely mounted.
	—	○	●		—	Check that the I / O modules are securely mounted. <i>Err</i>
	R635	○	●	○	—	Check the slot location of the I / O modules.
	—	○	●	●	—	Check the slot location of the I / O modules.
	R630	○	○	●	—	Replace the CPU module.
	R633	○		●	—	Check the program and write it to the EEPROM again.
	R637	○		●	—	Modify the program.
	R636	○		●	—	Correct the syntax of the program.
	—	○	●	●	—	Correct the syntax of the program.
	R63B	—	—	—	—	Check that the cable connecting the peripheral device is securely connected.
	R63C	—	—	—	—	Refer to the TOSLINE manual.
	R63D	—	—	—	—	Refer to the computer link manual.

*1) ① No END instruction → NO END ERROR

② Trouble with paired instructions → MC / JC ERROR

③ Operand error → OPERAND ERROR

*2) ① Error found in peripheral device →

② Error found in EX100 →

③ Communication timeout →

HP COMM ERROR

EX COMM ERROR

COMM TIMEOUT

A. Module current consumption

Module	Part No.	Internal 5V power supply	External power supply
CPU (Standard)	EX10•MPU11A	200 mA or less	—
CPU (Enhanced)	EX10•MPU12A	300 mA or less	—
6-slot rack (basic only)	EX10•UBA1	100 mA or less	—
9-slot rack (basic only)	EX10•UBA2	100 mA or less	—
6-slot rack (basic / exp.)	EX10•UBB1	250 mA or less	—
9-slot rack (basic / exp.)	EX10•UBB2	250 mA or less	—
16-pt dc / ac input (12-24V)	EX10•MDI31	15 mA or less	—
32-pt dc input (24V)	EX10•MDI32	80 mA or less	—
16-pt ac input (100-120 V)	EX10•MIN51	15 mA or less	—
16-pt ac input (200-240 V)	EX10•MIN61	15 mA or less	—
12-point relay output	EX10•MRO61	50 mA or less	24 Vdc, 140 mA
8-pt isolated relay output	EX10•MRO62	40 mA or less	24 Vdc, 100mA
16-point transistor output	EX10•MDO31	60 mA or less	5 to 24 Vdc, 35 mA
32-point transistor output	EX10•MDO32	250 mA or less	5 to 24 Vdc, 100mA
12-point triac output	EX10•MAC61	300 mA or less	—
4-ch A/D (4-20 mA / 1-5 V)	EX10•MAI21	50 mA or less	12 / 24 Vdc, 50 mA
4-ch A/D (0-10 V)	EX10•MAI31	50 mA or less	12 / 24 Vdc, 50 mA
4-ch A / D (4-20mA / 1-5V)	EX10•MAI22	50 mA or less	24 Vdc, 50mA
4-ch A / D ($\pm 10V$)	EX10•MAI32	50 mA or less	24 Vdc, 50mA
2-ch D / A (5/10 V / 20 mA)	EX10•MAO31	70 mA or less	24 Vdc, 90 mA
2-ch D / A (4-20mA / 1-5V)	EX10•MAO22	170 mA or less	24 Vdc, 90mA
2-ch D / A ($\pm 10V$)	EX10•MAO32	170 mA or less	24 Vdc, 90mA
1-ch pulse input	EX10•MPI21	80 mA or less	—
1-axis motion control	EX10•MMC11	200 mA or less	12 / 24 Vdc, 100mA
TOSLINE-30 (wire)	EX10•MLK11	250 mA or less	—
TOSLINE-30 (optical)	EX10•MLK12	200 mA or less	—



NOTE

- (1) The external power supplies given above are those required to operate the modules. They are not the input / output signals.
- (2) The current consumption of the peripheral devices are described in their respective manuals.

Appendices

B. Instruction execution times

Instruction		Execution time (μs)	
FUN No.	Symbol	Non-execution	Execution
		—	0.9
		—	0.9
		—	1.2
		—	0.9
		—	1.2
		—	1.2
	MCS	—	0.6
	MCR	—	0.6
	JCS	—	0.6
	JCR	—	0.6
	TON	1.8	96
	TOF	1.8	96
	SS	1.8	96
	CNT	1.8	92
	END	—	0.3
000	W → W	1.8	98
001	K → W	1.8/2.5	93
002	TINZ	2.5	98 + nn × 5
003	T → W	3.1	119
004	W → T	3.1	119
005	T → T	2.5	105 + nn × 11
010	+ (R-R)	2.5	110
011	- (R-R)	2.5	110
012	× (R-R)	2.5	168
013	/ (R-R)	2.5	342
014	> (R-R)	1.8	100
015	= (R-R)	1.8	100
016	< (R-R)	1.8	100
017	+ + (R-R)	2.5	125
018	- - (R-R)	2.5	130
020	+ . (R-K)	2.5/3.1	110
021	- . (R-K)	2.5/3.1	113
022	× . (R-K)	2.5/3.1	167
023	/ . (R-K)	2.5/3.1	343
024	> . (R-K)	1.8/2.5	98
025	= . (R-K)	1.8/2.5	98
026	< . (R-K)	1.8/2.5	98
030	AND (R-R)	2.5	107

Instruction		Execution time (μs)	
FUN No.	Symbol	Non-execution	Execution
031	OR (R-R)	2.5	107
032	EOR (R-R)	2.5	107
033	NOT	1.8	100
035	RTR	2.5	106 + nn × 5
036	RTL	2.5	106 + nn × 5
040	AND. (R-K)	2.5 / 3.1	109
041	OR. (R-K)	2.5 / 3.1	109
042	EOR. (R-K)	2.5 / 3.1	109
043	TEST	1.8 / 2.5	98
046	NEG	1.8	100
050	BIN	1.8	194
051	BCD1	1.8	125
052	BCD2	1.8	290
053	ENC	1.8	104
054	DEC	1.8	104
055	BITC	1.8	178
060	UL	2.5	116
061	LL	2.5	116
062	MAX	2.5	110 + nn × 9
063	MIN	2.5	110 + nn × 9
064	AVE	2.5	147 + nn × 9
065	FG	3.1	367 + nn × 37
070	RT	1.8	413
071	SIN	1.8	666
072	ASIN	1.8	819
073	COS	1.8	674
074	ACOS	1.8	824
080	SET	1.2	90
081	RST	1.2	93
090	DDSP	1.2 / 1.8	144 ~ 176
091	DDSM	1.2 / 1.8	161 ~ 189
096	IN	1.8	117 + nn × 63
097	OUT	1.8	117 + nn × 43
100	STIZ	1.8	105 ~ 154
101	STIN	1.2	98 ~ 118
102	STOT	—	83 ~ 141
110	F / F	—	102
111	U / D	—	98 ~ 116
112	SR	—	118 ~ 404

C. EX100 units and modules

Models and types

Unit / module	Description	Code	Part No.
Rack	6-slot (no connector for expansion unit)	UBA1	EX10•UBA1
	9-slot (no connector for expansion unit)	UBA2	EX10•UBA2
	6-slot (with connector for expansion unit)	UBB1	EX10•UBB1
	9-slot (with connector for expansion unit)	UBB2	EX10•UBB2
Power supply module	100-120 Vac	PS51	EX10•MPS51
	200-240 Vac	PS61	EX10•MPS61
	24 Vdc	PS31	EX10•MPS31
CPU module	Standard	PU11A	EX10•MPU11A
	Enhanced (with computer link and calendar)	PU12A	EX10•MPU12A
I / O modules	16-point dc / ac input (12-24 Vac / Vdc)	DI31	EX10•MDI31
	32-point dc input (24 Vdc)	DI32	EX10•MDI32
	16-point ac input (100-120 Vac)	IN51	EX10•MIN51
	16-point ac input (200-240 Vac)	IN61	EX10•MIN61
	12-point relay output (240 Vac / 24 Vdc)	RO61	EX10•MRO61
	8-point isolated relay output (240Vac / 24 Vdc)	RO62	EX10•MRO62
	16-point transistor output (5-24 Vdc)	DO31	EX10•MDO31
	32-point transistor output (5-24 Vdc)	DO32	EX10•MDO32
	12-point triac output (100-240 Vac)	AC61	EX10•MAC61
	4-ch analog input (4-20 mA / 1-5V) (8-bit)	AI21	EX10•MAI21
	4-ch analog input (0-10 V) (8-bit)	AI31	EX10•MAI31
	4-ch analog input (4-20mA / 1-5 V) (12-bit)	AI22	EX10•MAI22
	4-ch analog input ($\pm 10V$) (12-bit)	AI32	EX10•MAI32
	2-channel analog output (5 / 10 V, 20 mA) (8-bit)	AO31	EX10•MAO31
	2-ch analog output (4-20mA / 1-5 V) (12-bit)	AO22	EX10•MAO22
	2-ch analog output ($\pm 10V$) (12-bit)	AO32	EX10•MAO32
	1-channel pulse input (5 / 12 V, 100 kHz)	PI21	EX10•MPI21
	1-axis motion control	MC11	EX10•MMC11
Trans- mission	TOSLINE-30 (wire)	LK11	EX10•MLK11
	TOSLINE-30 (optical)	LK12	EX10•MLK12

Cables

Expansion cable (0.3 m)	EX10•CAR3
Expansion cable (0.5 m)	EX10•CAR5
Expansion cable (0.7 m)	EX10•CAR7

EX100 system options

Cover for vacant slot	EX10•ABP1
Lithium battery, CR2032 (Sold on the open market)	EX10•ACR2

RADIO SHACK NO. 23-162

Peripheral devices

Graphic programmer (Standard)	GP110	EX25UGP•110
Graphic programmer (Stand-alone, printer interface, FDD interface)	GP110AP1	EX25UG•110•AP1
Graphic programmer (same as AP1 plus EX2000 support)	GP110AP2	EX250GP•110•AP2
Handy programmer	HP100	EX25UHP•100
Mini programmer	MP100	EX25UMP•100
Data access panel	DP100	EX25UDP•100
Disk drive (FDD)	FD110	EX25UFD•110
Program storage module	RM11	EX10•PRM11

Spare parts

Fuse (PS51 / PS31) <i>125V 2A / 250V 1A</i>	EX10•SFB20
Fuse (PS61)	EX10•SFB10
Fuse (DO31)	EX10•SFA50
Fuse (DO32)	EX10•SFA20
Fuse (AC61)	EX10•SFC20

**TOSHIBA INTERNATIONAL
CORPORATION**

Industrial Equipment Division
13131 West Little York Road,
Houston, Texas 77041, USA
Phone: (713) 466-0277
Telex: 762078

TOSHIBA INTERNATIONAL (EUROPE) LTD.

1 Roundwood Avenue, Stockley Park Uxbridge
Middlesex UB11 1AR, England
Phone: (01) 848-4466
Telex: 265062 TSB LDN G

TOSHIBA INTERNATIONAL CORPORATION PTY, LTD.

Industrial Division
Unit 1, 9 Orion Road, Lane Cove N. S. W. 2066, Australia
Phone: (02) 428-2077 Telex: AA25192

TOSHIBA

TOSHIBA CORPORATION

Industrial Equipment Export Department
1-1, Shibaura 1-chome, Minato-ku, Tokyo 105, Japan
Telex: J22587 TOSHIBA Cable: TOSHIBA Tokyo
Phone: (03) 457-4900