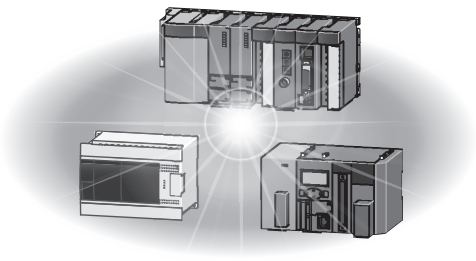
MITSUBISHI ELECTRIC

Programmable Controller

MELSEC Q series MELSEC L series

MELSEC-Q/L Programming Manual (Common Instruction)



SAFETY PRECAUTIONS

(Read these precautions before using this product.)

Before using this product, please read this manual and the related manuals introduced in this manual, and pay full attention to safety to handle the product correctly.

Please store this manual in a safe place and make it accessible when required. Always forward a copy of the manual to the end user.

CONDITIONS OF USE FOR THE PRODUCT

(1) MELSEC programmable controller ("the PRODUCT") shall be used in conditions;

i) where any problem, fault or failure occurring in the PRODUCT, if any, shall not lead to any major or serious accident; and

ii) where the backup and fail-safe function are systematically or automatically provided outside of the PRODUCT for the case of any problem, fault or failure occurring in the PRODUCT.

(2) The PRODUCT has been designed and manufactured for the purpose of being used in general industries.

MITSUBISHI ELECTRIC SHALL HAVE NO RESPONSIBILITY OR LIABILITY (INCLUDING, BUT NOT LIMITED TO ANY AND ALL RESPONSIBILITY OR LIABILITY BASED ON CONTRACT, WARRANTY, TORT, PRODUCT LIABILITY) FOR ANY INJURY OR DEATH TO PERSONS OR LOSS OR DAMAGE TO PROPERTY CAUSED BY the PRODUCT THAT ARE OPERATED OR USED IN APPLICATION NOT INTENDED OR EXCLUDED BY INSTRUCTIONS, PRECAUTIONS, OR WARNING CONTAINED IN MITSUBISHI ELECTRIC USER'S, INSTRUCTION AND/OR SAFETY MANUALS, TECHNICAL BULLETINS AND GUIDELINES FOR the PRODUCT.

("Prohibited Application")

Prohibited Applications include, but not limited to, the use of the PRODUCT in;

- Nuclear Power Plants and any other power plants operated by Power companies, and/or any other cases in which the public could be affected if any problem or fault occurs in the PRODUCT.
- Railway companies or Public service purposes, and/or any other cases in which establishment of a special quality assurance system is required by the Purchaser or End User.
- Aircraft or Aerospace, Medical applications, Train equipment, transport equipment such as Elevator and Escalator, Incineration and Fuel devices, Vehicles, Manned transportation, Equipment for Recreation and Amusement, and Safety devices, handling of Nuclear or Hazardous Materials or Chemicals, Mining and Drilling, and/or other applications where there is a significant risk of injury to the public or property.

Notwithstanding the above restrictions, Mitsubishi Electric may in its sole discretion, authorize use of the PRODUCT in one or more of the Prohibited Applications, provided that the usage of the PRODUCT is limited only for the specific applications agreed to by Mitsubishi Electric and provided further that no special quality assurance or fail-safe, redundant or other safety features which exceed the general specifications of the PRODUCTs are required. For details, please contact the Mitsubishi Electric representative in your region.

(3) Mitsubishi Electric shall have no responsibility or liability for any problems involving programmable controller trouble and system trouble caused by DoS attacks, unauthorized access, computer viruses, and other cyberattacks.

INTRODUCTION

This manual "MELSEC-Q/L Programming Manual (Common Instruction)" describes the common instructions required for programming of the QCPU and LCPU.
"Common instructions" are all instructions except for dedicated instructions for intelligent function modules; PID control instructions; process control instruction; SFC instructions; ST instructions; instructions for socket communication features; trigger logging instructions for the LCPU; and dedicated instructions for LCPU positioning/counter functionality.
Before using this product, please read this manual and the relevant manuals carefully and develop familiarity with the functions and performance of the MELSEC-Q or -L series programmable controller to handle the product correctly.
When applying the program examples introduced in this manual to an actual system, ensure the applicability and confirm that it will not cause system control problems.

Relevant CPU module

CPU module	Model
Basic model QCPU	Q00JCPU, Q00CPU, Q01CPU
High Performance model QCPU	Q02CPU, Q02HCPU, Q06HCPU, Q12HCPU, Q25HCPU
Process CPU	Q02PHCPU, Q06PHCPU, Q12PHCPU, Q25PHCPU
Redundant CPU	Q12PRHCPU, Q25PRHCPU
Universal model QCPU	Q00UJCPU, Q00UCPU, Q01UCPU, Q02UCPU, Q03UDCPU, Q03UDVCPU, Q03UDECPU, Q04UDHCPU, Q04UDVCPU, Q04UDPVCPU, Q04UDEHCPU, Q06UDHCPU, Q06UDVCPU, Q06UDPVCPU, Q06UDEHCPU, Q10UDHCPU, Q10UDEHCPU, Q13UDHCPU, Q13UDVCPU, Q13UDPVCPU, Q13UDEHCPU, Q20UDHCPU, Q20UDEHCPU, Q26UDHCPU, Q26UDVCPU, Q26UDPVCPU, Q26UDEHCPU, Q50UDEHCPU, Q100UDEHCPU
LCPU	L02SCPU, L02SCPU-P, L02CPU, L02CPU-P, L06CPU, L06CPU-P, L26CPU, L26CPU-P, L26CPU-BT, L26CPU-PBT

CONTENTS

MANUALS

To understand the main specifications, functions, and usage of the CPU module, refer to the basic manuals.
Read other manuals as well when using a different type of CPU module and its functions.
Order each manual as needed, referring to the following lists.
The numbers in the "CPU module" and the respective modules are as follows.

Number	CPU module
1)	Basic model QCPU
2)	High Performance model QCPU
3)	Process CPU
4)	Redundant CPU
5)	Universal model QCPU
6)	LCPU

●: Basic manual, ○: Other CPU module manuals

Manual name <Manual number>	Description	CPU module					
		1)	2)	3)	4)	5)	6)
■ User's manual							
QCPU User's Manual (Hardware Design, Maintenance and Inspection) <SH-080483ENG>	Specifications of the hardware (CPU modules, power supply modules, base units, extension cables, memory cards, SD memory cards, extended SRAM cassettes, and batteries), system maintenance and inspection, and troubleshooting	●	●	●	●	●	
QnUCPU User's Manual (Function Explanation, Program Fundamentals) <SH-080807ENG>	Functions, methods, and devices for programming					●	
Qn(H)/QnPH/QnPRHCPU User's Manual (Function Explanation, Program Fundamentals) <SH-080808ENG>	Functions, methods, and devices for programming	●	●	●	●		
QnUCPU User's Manual (Communication via Built-in Ethernet Port) <SH-080811ENG>	Functions for the communication via built-in Ethernet port of the CPU module					○	
MELSEC-L CPU Module User's Manual (Hardware Design, Maintenance and Inspection) <SH-080890ENG>	Specifications of the hardware (CPU modules, power supply modules, a branch module, an extension module, and SD memory cards), system maintenance and inspection, troubleshooting, and error codes						●
MELSEC-L CPU Module User's Manual (Function Explanation, Program Fundamentals) <SH-080889ENG>	Functions, methods, and devices for programming						●
MELSEC-L CPU Module User's Manual (Built-In I/O Function) <SH-080892ENG>	Built-in I/O Functionality of the CPU						○
MELSEC-L CPU Module User's Manual (Built-In Ethernet Function) <SH-080891ENG>	Functions for the communication via built-in Ethernet port of the CPU module						○
QnUDVCPU/LCPU User's Manual (Data Logging Function) <SH-080893ENG>	Data Logging Functionality of the CPU Module					○	○

Manual name <Manual number>	Description	CPU module					
		1)	2)	3)	4)	5)	6)
■ Programming manual							
MELSEC-Q/L Programming Manual (Common Instructions) <SH-080809ENG>	How to use sequence instructions, basic instructions, and application instructions	●	●	●	●	●	●
MELSEC-Q/L/QnA Programming Manual (SFC) <SH-080041>	System configuration, performance specifications, functions, programming, debugging, and error codes for SFC (MELSAP3) programs	○	○	○	○	○	○
MELSEC-Q/L Programming Manual (MELSAP-L) <SH-080076>	Programming methods, specifications, and functions for SFC (MELSAP-L) programs	○	○	○	○	○	○
MELSEC-Q/L Programming Manual (Structured Text) <SH-080366E>	Programming methods using structured languages	○	○	○	○	○	○
MELSEC-Q/L/QnA Programming Manual (PID Control Instructions) <SH-080040>	Dedicated instructions for PID control	○	○		○	○	○
MELSEC-Q Programming/Structured Programming Manual (Process Control Instructions) <SH-080316E>	Describes the dedicated instructions for performing process control.			○	○	○	

Related Manuals

Manual name <Manual number>	Description
MELSEC-Q CC-Link IE Controller Network Reference Manual <SH-080668ENG>	Specifications, procedures and settings before system operation, parameter settings, programming, and troubleshooting of the CC-Link IE Controller Network module
MELSEC-Q CC-Link IE Field Network Master/Local Module User's Manual <SH-080917ENG>	Specifications, procedures and settings before system operation, parameter settings, programming, and troubleshooting of the CC-Link IE Field Network module
MELSEC-L CC-Link IE Field Network Master/Local Module User's Manual <SH-080972ENG>	Specifications, procedures and settings before system operation, parameter settings, programming, and troubleshooting of the CC-Link IE Field Network module
Q Corresponding MELSECNET/H Network System Reference Manual (PLC to PLC network) <SH-080049>	Specifications, procedures and settings before system operation, parameter setting, programming, and troubleshooting of a MELSECNET/H network system (PLC to PLC network)
Q Corresponding MELSECNET/H Network System Reference Manual (Remote I/O network) <SH-080124>	Specifications, procedures and settings before system operation, parameter setting, programming, and troubleshooting of a MELSECNET/H network system (remote I/O network)
Type MELSECNET, MELSECNET/B Data Link System Reference Manual <IB-66350>	Describes the general concept, specifications, and part names and settings for MELSECNET (Ⅱ) and MELSECNET/B.
MELSEC-Q/L Ethernet Interface Module User's Manual (Application) <SH-080010>	E-mail function, programmable controller CPU status monitoring function, communication via CC-Link IE Field Network, CC-Link IE Controller Network, MELSECNET/H, or MELSECNET/10, communication using the data link instructions, and file transfer function (FTP server) of the Ethernet module

TERMS

This manual uses the generic names and abbreviations shown below to refer to Q/L series CPU modules, unless otherwise specified.

☐ indicates a part of the model or version.

Term	Description
A5☐B	A generic term for the power source-free type A52B, A55B, and A58B extension base unit on which the A Series I/O module and special function module can be mounted
A6☐B	A generic term for the A62B, A65B, and A68B extension base unit on which the A Series I/O module and special function module can be mounted
Basic model QCPU	A generic term for Q00JCPU, Q00CPU and Q01CPU
Built-in Ethernet port LCPU	A generic term for the L02CPU, L02CPU-P, L06CPU, L06CPU-P, L26CPU, L26CPU-P, L26CPU-BT, and L26CPU-PBT
Built-in Ethernet port QCPU	A generic term for Q03UDVCPU, Q03UDECPU, Q04UDVCPU, Q04UDPVCPU, Q04UDEHCPU, Q06UDVCPU, Q06UDPVCPU, Q06UDEHCPU, Q10UDEHCPU, Q13UDVCPU, Q13UDPVCPU, Q13UDEHCPU, Q20UDEHCPU, Q26UDVCPU, Q26UDPVCPU, Q26UDEHCPU, Q50UDEHCPU and Q100UDEHCPU
CC-Link IE	A generic term for the CC-Link IE Controller Network and the CC-Link IE Field Network
CPU module	A generic term for Basic model QCPU, High Performance model QCPU, Process CPU, Redundant CPU, Universal model QCPU and LCPU
GX Developer	The product names for MELSEC programmable controller software package. To check the versions that can be used for each CPU module, refer to "System Configuration" of User's Manual (Hardware Design, Maintenance and Inspection) for the CPU module used.
GX Works2	
High Performance model QCPU	A generic term for Q02CPU, Q02HCPU, Q06HCPU, Q12HCPU and Q25HCPU
High-speed Universal model QCPU	A generic term for Q03UDVCPU, Q04UDVCPU, Q06UDVCPU, Q13UDVCPU and Q26UDVCPU
Intelligent function module device	A generic term for intelligent function module devices and special function module devices
L series	The abbreviation for Mitsubishi Electric MELSEC-L series programmable controller
LCPU	A generic term for L02SCPU, L02SCPU-P, L02CPU, L02CPU-P, L06CPU, L06CPU-P, L26CPU, L26CPU-P, L26CPU-BT and L26CPU-PBT
MELSECNET(Ⅱ, /B)	The abbreviation for MELSECNET and MELSECNET/B data link system
MELSECNET/10	The abbreviation for MELSECNET/10 network system
MELSECNET/H	The abbreviation for MELSECNET/H network system
Process CPU	A generic term for Q02PHCPU, Q06PHCPU, Q12PHCPU and Q25PHCPU
Programming tool	A generic term for GX Developer and GX Works2
Q series	The abbreviation for Mitsubishi Electric MELSEC-Q series programmable controller
Q3☐DB	A generic term for the Q35DB, Q38DB and Q312DB type Multiple CPU high speed main base unit on which CPU module (except the Q00JCPU), Q series power supply module, Q series I/O module, and intelligent function module can be mounted
Q5☐B	A generic term for Q52B and Q55B extension base unit on which the Q Series I/O and intelligent function module can be mounted
Q6☐B	A generic term for Q63B, Q65B, Q68B and Q612B extension base unit on which Q Series power supply module, I/O module, intelligent function module can be mounted
Q6☐WRB	Another term for Q65WRB extension base unit for redundant system on which redundant power supply module, Q series I/O module, and intelligent function module can be mounted.
QA1S5☐B	A generic term for QA1S51B extension base unit on which AnS Series I/O module, special function module can be mounted
QA1S6☐B	A generic term for QA1S65B and QA1S68B extension base units on which AnS Series power supply module, I/O module, special function module can be mounted
QA6ADP	The abbreviation for the QA6ADP QA conversion adapter module
QA6ADP+A5☐B/A6☐B	The abbreviation for the A large type extension base unit equipped with the QA6ADP
QA6☐B	A generic term for the QA65B and QA68B extension base units on which the A Series power supply module, A Series I/O module, and special function module can be mounted
QnCPU	A generic term for Q00JCPU, Q00CPU, Q01CPU and Q02CPU
QnHCPU	A generic term for Q02HCPU, Q06HCPU, Q12HCPU and Q25HCPU
QnPHCPU	A generic term for Q02PHCPU, Q06PHCPU, Q12PHCPU and Q25PHCPU
QnPRHCPU	A generic term for Q12PRHCPU and Q25PRHCPU
QnU(D)(H)CPU	A generic term for Q02UCPU, Q03UDCPU, Q04UDHCPU, Q06UDHCPU, Q10UDHCPU, Q13UDHCPU, Q20UDHCPU and Q26UDHCPU

Term	Description
QnUCPU	A generic term for Q00UJCPU, Q00UCPU, Q01UCPU, Q02UCPU, Q03UDCPU, Q03UDVCPU, Q03UDECPU, Q04UDHCPU, Q04UDVCPU, Q04UDPVCPU, Q04UDEHCPU, Q06UDHCPU, Q06UDVCPU, Q06UDPVCPU, Q06UDEHCPU, Q10UDHCPU, Q10UDEHCPU, Q13UDHCPU, Q13UDVCPU, Q13UDPVCPU, Q13UDEHCPU, Q20UDHCPU, Q20UDEHCPU, Q26UDHCPU, Q26UDVCPU, Q26UDPVCPU, Q26UDEHCPU, Q50UDEHCPU and Q100UDEHCPU
QnUD(H)CPU	A generic term for Q03UDCPU, Q04UDHCPU, Q06UDHCPU, Q10UDHCPU, Q13UDHCPU, Q20UDHCPU and Q26UDHCPU
QnUDE(H)CPU	A generic term for Q03UDECPU, Q04UDEHCPU, Q06UDEHCPU, Q10UDEHCPU, Q13UDEHCPU, Q20UDEHCPU, Q26UDEHCPU, Q50UDEHCPU and Q100UDEHCPU
QnUDPVCPU	A generic term for Q04UDPVCPU, Q06UDPVCPU, Q13UDPVCPU and Q26UDPVCPU
QnUDVCPU	A generic term for Q03UDVCPU, Q04UDVCPU, Q06UDVCPU, Q13UDVCPU and Q26UDVCPU
Redundant CPU	A generic term for Q12PRHCPU and Q25PRHCPU
Universal model Process CPU	A generic term for Q04UDPVCPU, Q06UDPVCPU, Q13UDPVCPU and Q26UDPVCPU
Universal model QCPU	A generic term for Q00UJCPU, Q00UCPU, Q01UCPU, Q02UCPU, Q03UDCPU, Q03UDVCPU, Q03UDECPU, Q04UDHCPU, Q04UDVCPU, Q04UDPVCPU, Q04UDEHCPU, Q06UDHCPU, Q06UDVCPU, Q06UDPVCPU, Q06UDEHCPU, Q10UDHCPU, Q10UDEHCPU, Q13UDHCPU, Q13UDVCPU, Q13UDPVCPU, Q13UDEHCPU, Q20UDHCPU, Q20UDEHCPU, Q26UDHCPU, Q26UDVCPU, Q26UDPVCPU, Q26UDEHCPU, Q50UDEHCPU and Q100UDEHCPU

1 GENERAL DESCRIPTION

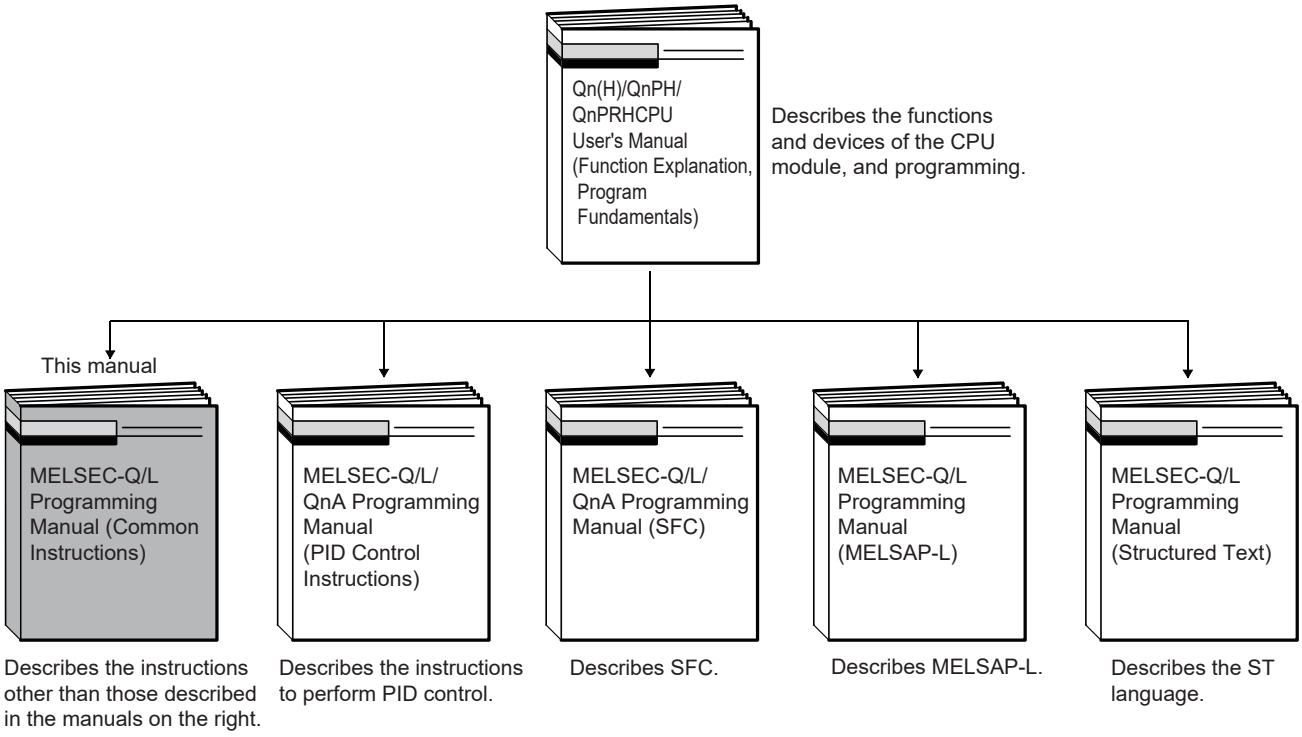
This manual describes the common instructions required for programming of the QCPU and LCPU.
"Common instructions" are all instructions except for dedicated instructions for intelligent function modules; PID control instructions; process control instruction; SFC instructions; ST instructions; instructions for socket communication features; trigger logging instructions for the LCPU; and dedicated instructions for LCPU positioning/counter functionality.

1.1 Related Programming Manuals

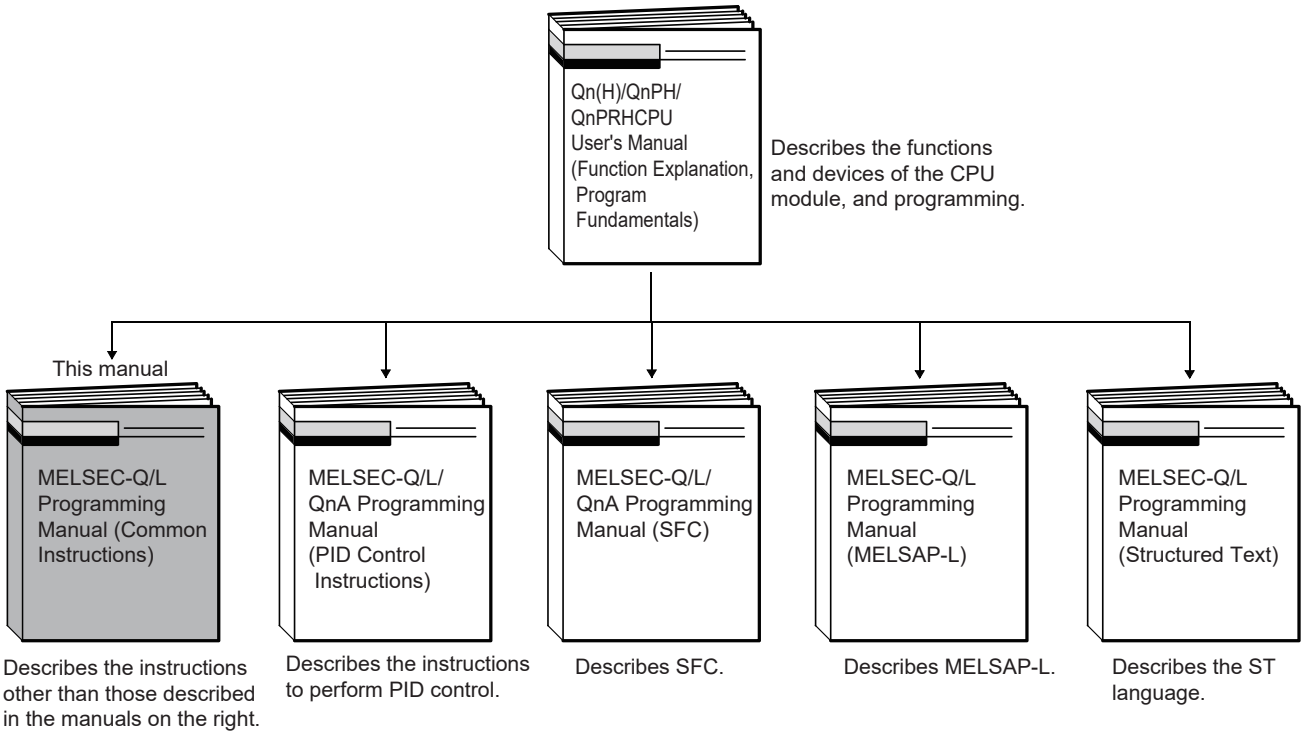
Before reading this manual, check the functions, programming methods, devices and others that are necessary to create programs with the CPU in the manuals below:

📖 QnUCPU User's Manual (Function Explanation, Program Fundamentals)
📖 Qn(H)/QnPH/QnPRHCPU User's Manual (Function Explanation, Program Fundamentals)
📖 MELSEC-L CPU Module User's Manual (Function Explanation, Program Fundamentals)

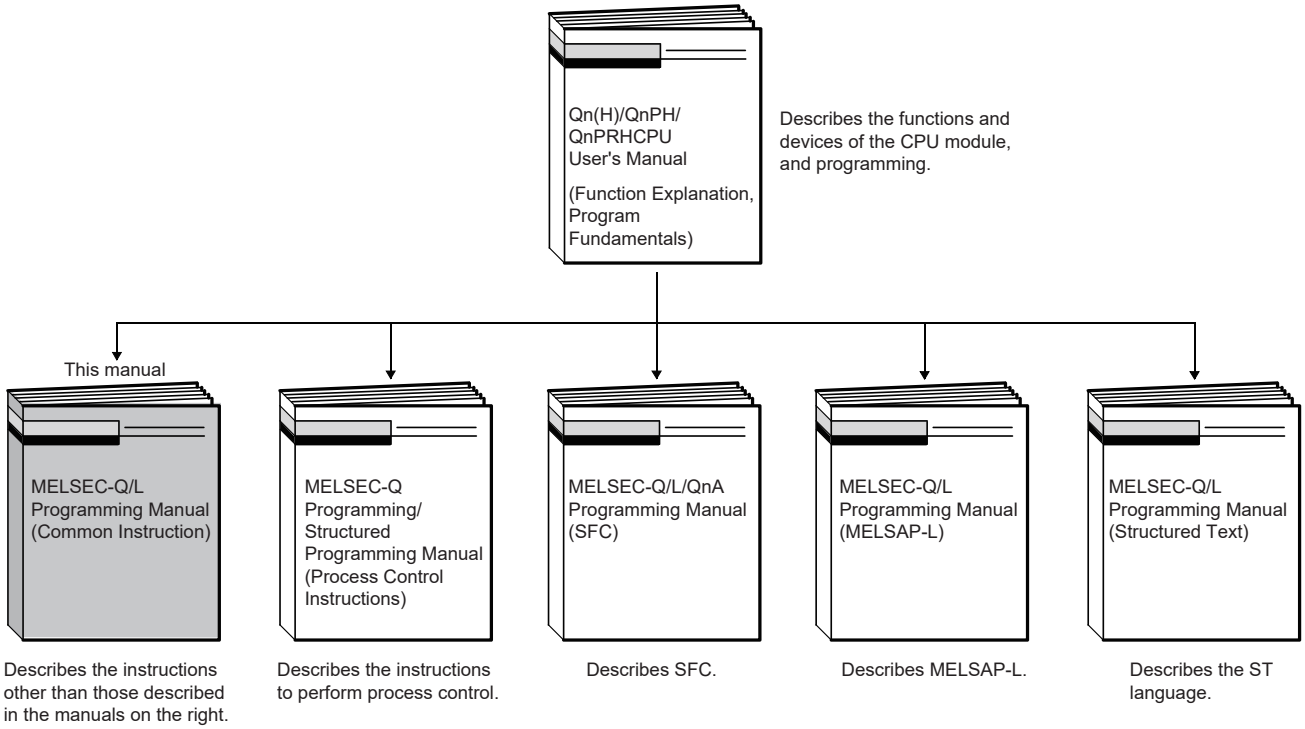
Basic model QCPU



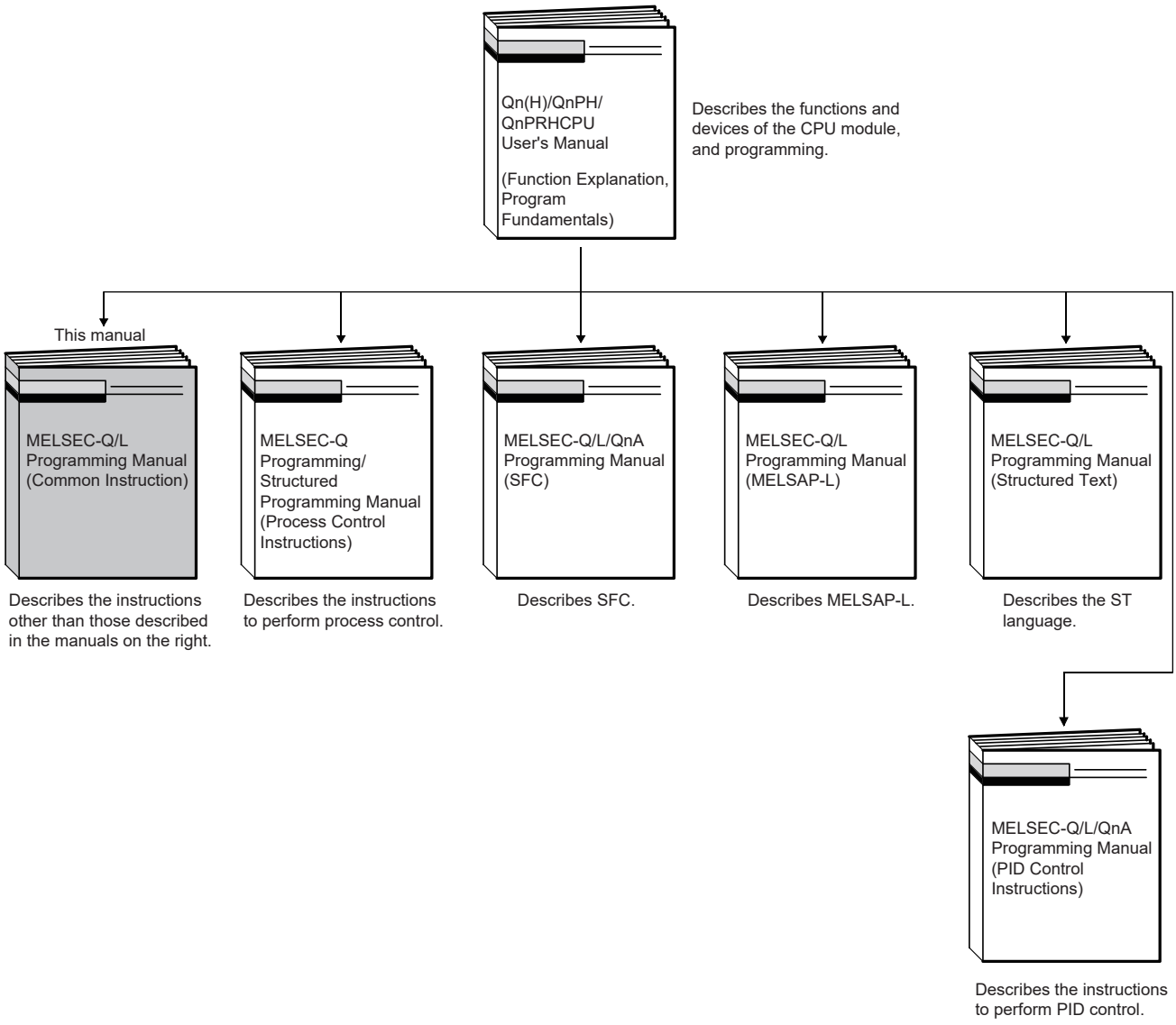
High Performance model QCPU



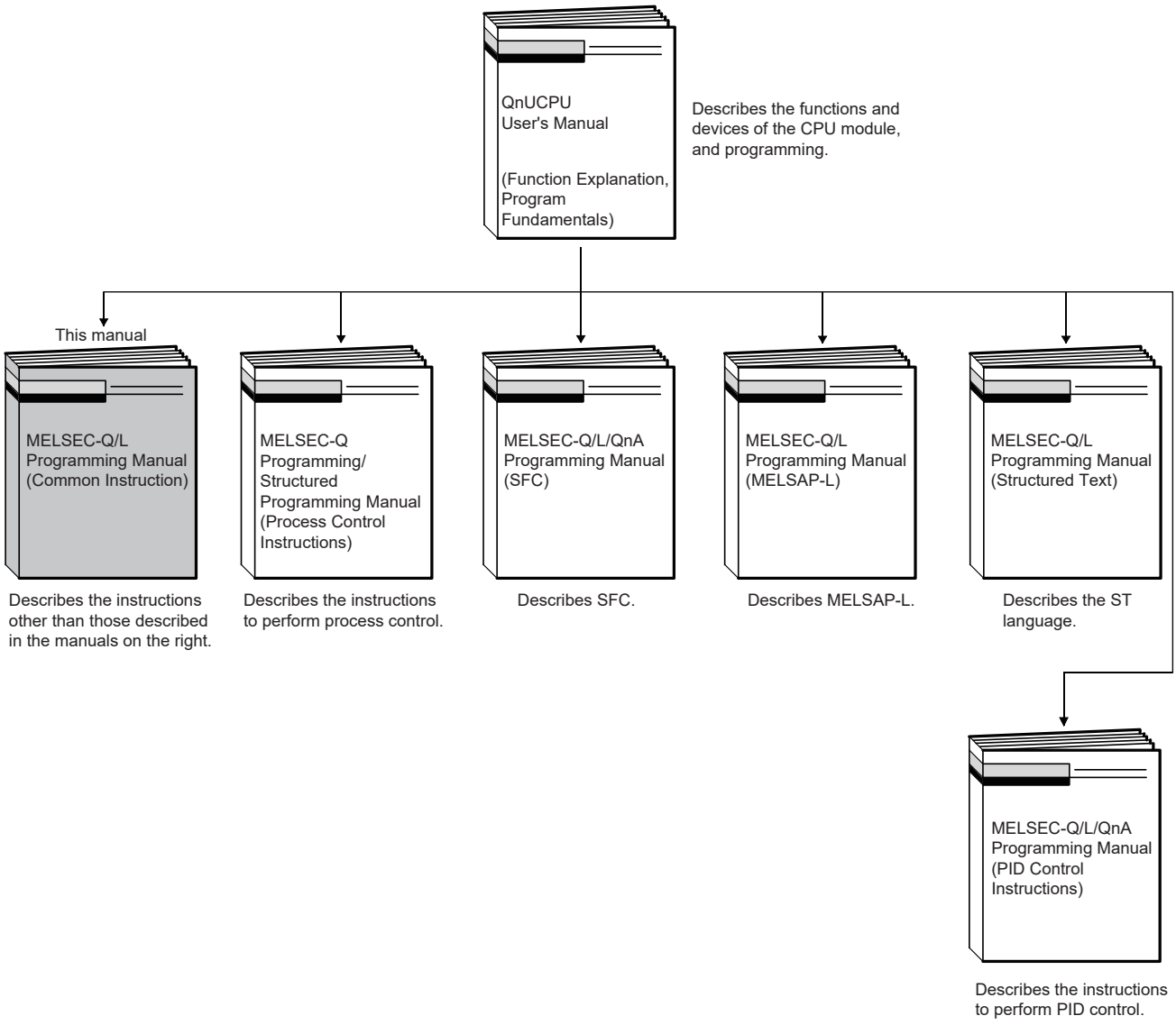
Process CPU



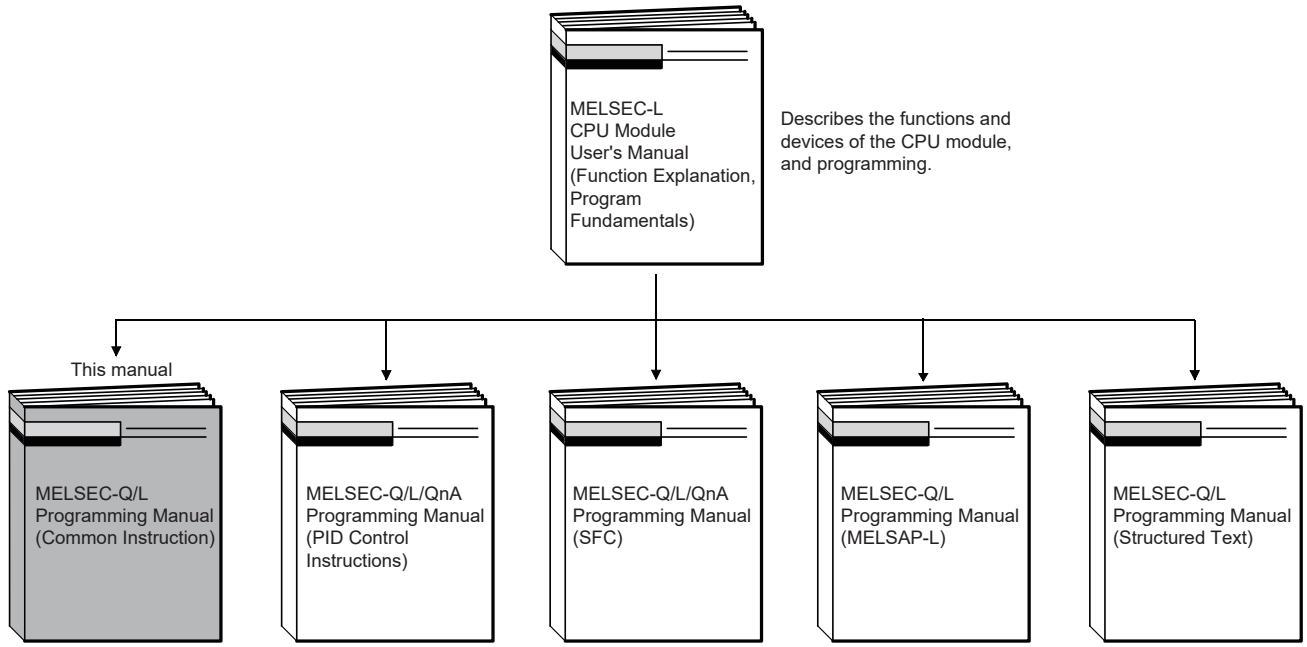
Redundant CPU



Universal model QCPU



LCPU


MELSEC-L
CPU Module
User's Manual
(Function Explanation,
Program
Fundamentals)
Describes the functions and devices of the CPU module, and programming.
This manual
MELSEC-Q/L
Programming Manual
(Common Instruction)
MELSEC-Q/L/QnA
Programming Manual
(PID Control
Instructions)
MELSEC-Q/L/QnA
Programming Manual
(SFC)
MELSEC-Q/L
Programming Manual
(MELSAP-L)
MELSEC-Q/L
Programming Manual
(Structured Text)
Describes the instructions other than those described in the manuals on the right.
Describes the instructions to perform process control.
Describes SFC.
Describes MELSAP-L.
Describes the ST language.

2 INSTRUCTION TABLES

2.1 Types of Instructions

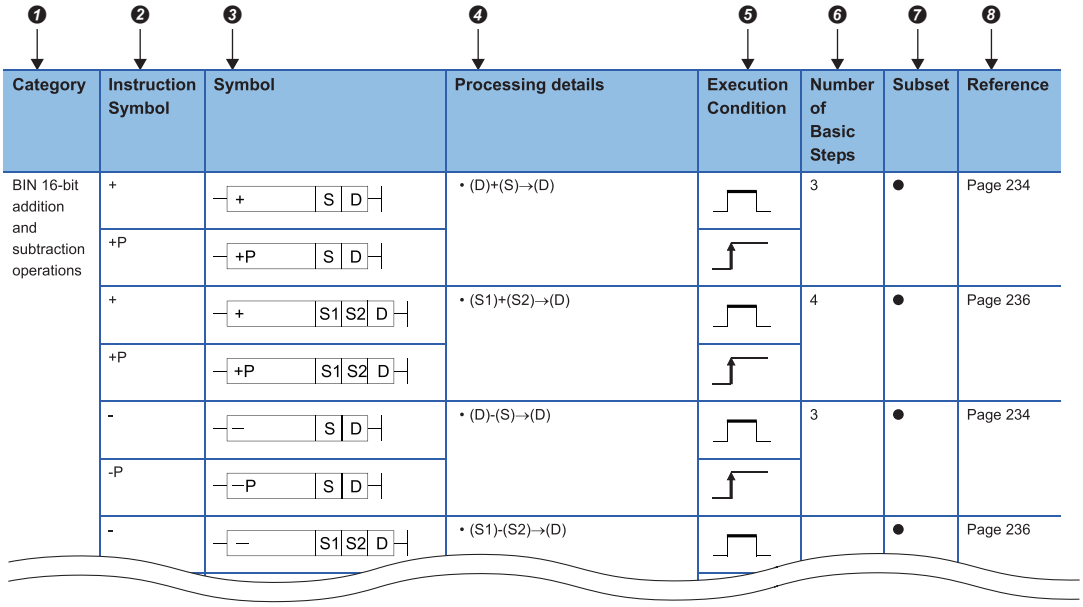
The major types of CPU module instructions consist of sequence instructions, basic instructions, application instructions, data link instructions, QCPU instructions and redundant system instructions. These types of instructions are listed in the following Table.

Types of instructions		Description	Reference
Sequence instruction	Contact instruction	Operation start, series connection, parallel connection	Page 132 SEQUENCE INSTRUCTIONS
	Association instruction	Ladder block connection, store/read operation results, creation of pulses from operation results	
	Output instruction	Bit device output, pulse output, output reversal	
	Shift instruction	Bit device shift	
	Master control instruction	Master control	
	Termination instruction	Program termination	
	Other instruction	Program stop, instructions such as no operation which do not fit in the above categories	
Basic instruction	Comparison operation instructions	Comparisons such as =, >, <	Page 189 BASIC INSTRUCTIONS
	Arithmetic operation instruction	Addition, subtraction, multiplication or division of BIN or BCD	
	BCD ↔ BIN conversion instruction	Conversion from BCD to BIN and from BIN to BCD	
	Data transfer instruction	Transmits designated data	
	Program branch instruction	Program jumps	
	Program run control instruction	Enables or inhibits interrupt programs	
	I/O refresh instruction	Executes partial refresh	
	Other convenient instruction	Instructions for: Counter increment/decrement, teaching timer, special function timer, rotary table shortest direction control, etc.	

Types of instructions		Description	Reference
Application instruction	Logical operation instruction	Logical operations such as logical sum, logical product, etc.	Page 357 APPLICATION INSTRUCTIONS
	Rotation instruction	Rotation of designated data	
	Shift instruction	Shift of designated data	
	Bit processing instruction	Bit set and reset, bit test, batch reset of bit devices	
	Data processing instruction	16-bit data searches, data processing such as decoding and encoding	
	Structure creation instruction	Repeated operation, subroutine program calls, indexing in ladder units	
	Table operation instruction	Data table read/write	
	Buffer memory access instruction	Data read/write from/to an intelligent function module	
	Display instruction	Print ASCII code, etc.	
	Debugging and failure diagnosis instruction	Check, status latch, sampling trace	
	Character string processing instruction	Conversion between BIN/BCD and ASCII; conversion between BIN and character string; conversion between floating decimal point data and character strings, character string processing, etc.	
	Special function instruction	Trigonometric functions, conversion between angles and radians, exponential operations, natural logarithm, common logarithm, square roots	
	Data control instruction	Upper and lower limit controls, dead band controls, zone controls	
	Switching instruction	File register block No. switches, designation of file registers and comment files	
	Clock instruction	• Reading/writing data of year, month, day, hour, minute, second, and day of the week • Adding/subtracting data of hour, minute, and second • Converting data of hour, minute, and second into second, and vice versa • Converting data of year, month, day, hour, minute, and second into second • Converting data of second into year, month, day, hour, minute, second, and day of the week • Comparing data of year, month, and day • Comparing data of hour, minute, and second	
	Expansion clock instruction	Reading of the values of year, month, day, hour, minute, second, millisecond, and day of the week; addition/subtraction of the values of hour, minute, second, and millisecond	
	Program control instruction	Instructions to switch program execution conditions	
	Other instruction	Instructions that do not fit in the above categories, such as watchdog timer reset instructions and timing clock instructions	
Data link instruction	Link refresh instruction	Designated network refresh	Page 827 INSTRUCTIONS FOR DATA LINK
	Routing information read/write instruction	Reads, writes, and registers routing information	
	Refresh device write/read instruction	Reads or writes the refresh device.	
Multiple CPU dedicated instruction	Multiple CPU dedicated instruction	Writing to host CPU shared memory, Reading from other CPU shared memory	Page 853 MULTIPLE CPU DEDICATED INSTRUCTIONS
Multiple CPU high-speed transmission dedicated instruction	Multiple CPU device write/read instruction	Writes/reads devices to/from another CPU.	Page 868 MULTIPLE CPU HIGH-SPEED TRANSMISSION DEDICATED INSTRUCTIONS
Redundant system instruction	Instruction for Redundant CPU	System switching	Page 887 REDUNDANT SYSTEM INSTRUCTIONS (FOR REDUNDANT CPU)

2.2 How to Read Instruction Tables

The instruction tables found from Page 25 Sequence Instructions to Page 80 Redundant System Instructions (For Redundant CPU) have been made according to the following format:



❶ Category	❷ Instruction Symbol	❸ Symbol	❹ Processing details	❺ Execution Condition	❻ Number of Basic Steps	❼ Subset	❽ Reference
BIN 16-bit addition and subtraction operations	+	+ S D	• (D)+(S)→(D)		3	●	Page 234
	+P	+P S D					
	+	+ S1 S2 D	• (S1)+(S2)→(D)		4	●	Page 236
	+P	+P S1 S2 D					
	-	– S D	• (D)-(S)→(D)		3	●	Page 234
	-P	–P S D					
	-	– S1 S2 D	• (S1)-(S2)→(D)			●	Page 236

❶ Classifies instructions according to their application.

❷ Indicates the instruction symbol added to the instruction in a program.

Instruction code is built around the 16-bit instruction.

The following notations are used to mark 32-bit instructions, instructions executed only at the rising edge of OFF to ON, real number instructions, and character string instructions:

• 32-bit instruction: The letter "D" is added to the first line of the instruction.

Example + ➡ D+

16-bit instruction / 32-bit instruction

• Instructions executed only at the rising edge of OFF to ON: The letter "P" is added to the end of the instruction.

Example + ➡ +P

Instruction executed when ON / Instruction executed only at the leading edge of OFF to ON

• Real number instructions: The letter "E" is added to the first line of the instruction.

Example + ➡ E+

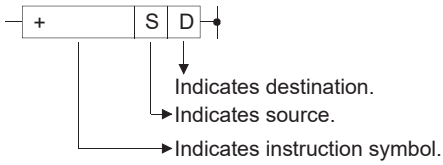
Real number instructions

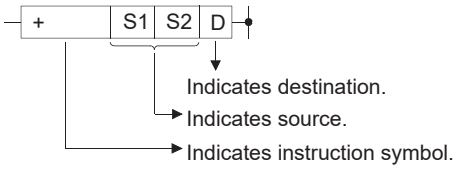
• Character string instructions: A dollar sign $ is added to the first line of the instruction.

Example + ➡ $+

Character string instructions

❸ Shows symbol diagram on the ladder.

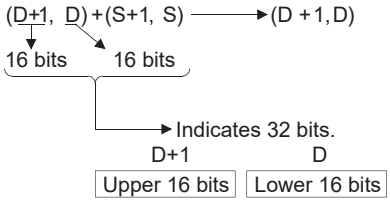




• Destination: Indicates where data will be sent after operation.

• Source: Stores data prior to operation.

❹ Indicates the type of processing that is performed by individual instructions.



❺ Indicates execution conditions for individual instructions.

Execution condition	Non-conditional execution	Executed at ON	Executed at the rising edge	Executed at OFF	Executed at the falling edge
Recorded code	No symbol recorded				

For execution conditions, refer to Page 118 Conditions for Execution of Instructions.

❻ Indicates the basic number of steps for individual instructions.

See Page 119 Counting Step Number for a description of the number of steps.

❼ The ● mark indicates instructions for which subset processing is possible.

See Page 109 Reducing Instruction Processing Time for details on subset processing.

❽ Indicates the page numbers where the individual instructions are explained.

2.3 Sequence Instructions

Contact instructions

Category	Instruction symbol	Symbol	Processing details	Execution condition	Number of basic steps	Subset	Reference
Contact	LD		• Starts logic operation (Starts A contact logic operation)		*1	●	Page 132
	LDI		• Starts logical NOT operation (Starts B contact logic operation)				
	AND		• Logical product (A contact series connection)				
	ANI		• Logical product NOT (B contact series connection)				
	OR		• Logical sum (A contact parallel connection)				
	ORI		• Logical sum NOT (B contact parallel connection)				
	LDP		• Starts rising edge pulse operation		*1	●	Page 135
	LDF		• Starts falling edge pulse operation				
	ANDP		• Rising edge pulse series connection				
	ANDF		• Falling edge pulse series connection				
	ORP		• Rising edge pulse parallel connection				
	ORF		• Falling edge pulse parallel connection				
	LDPI		• Starts rising edge pulse NOT operation		3*2*3	●	Page 137
	LDFI		• Starts falling edge pulse NOT operation		3*2*3		
	ANDPI		• Rising edge pulse NOT series connection		4*2*3		
	ANDFI		• Falling edge pulse NOT series connection		4*2*3		
	ORPI		• Rising edge pulse NOT parallel connection		4*2*3		
	ORFI		• Falling edge pulse NOT parallel connection		4*2*3		

*1 The number of steps may vary depending on the device being used.

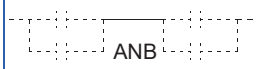
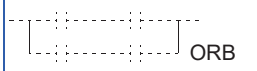
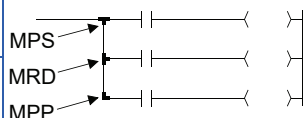
Device	Number of steps
Internal device, file register (R0 to R32767)	1
Direct access input (DX)	2
Devices other than above	3

*2 The number of steps may differ, depending on the device or CPU module to be used.

CPU module	Device	Number of steps
Basic model QCPU High Performance model QCPU Process CPU Redundant CPU	• Internal device, file register (R0 to R32767) • Direct access input (DX)	1
	Devices other than above	3
Universal model QCPU LCPU	Internal device, file register (R0 to R32767)	Number of basic steps
	• Serial number access format file register (ZR), Extended data register (D), Extended link register (W), Multiple CPU shared device (U3En\G10000) • Direct access input (DX)	Number of basic steps +1
	Devices other than above	Number of basic steps +2

*3 For the High-speed Universal model QCPU and Universal model Process CPU, the number of basic steps is two.

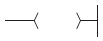
Association instructions

Category	Instruction symbol	Symbol	Processing details	Execution condition	Number of basic steps	Subset	Reference
Connection	ANB		• AND between logical blocks (Series connection between logical blocks)		1	—	Page 139
	ORB		• OR between logical blocks (Series connection between logical blocks)				
	MPS		• Memory storage of operation results		1	—	Page 141
	MRD		• Read of operation results stored with MPS instruction				
	MPP		• Read and reset of operation results stored with MPS instruction				
	INV		• Inversion of operation result		1	—	Page 144
	MEP		• Conversion of operation result to rising edge pulse		1	—	Page 146
	MEF		• Conversion of operation result to falling edge pulse				
	EGP		• Conversion of operation result to rising edge pulse (Stored at Vn)		1	—	Page 148
	EGF		• Conversion of operation result to falling edge pulse (Stored at Vn)		*1		

*1 The number of steps may differ, depending on CPU modules.

CPU module	Number of basic steps
High Performance model QCPU Process CPU Redundant CPU Universal model QCPU LCPU	1
Basic model QCPU	2

Output instructions

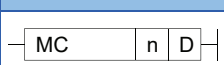
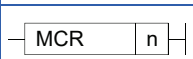
Category	Instruction symbol	Symbol	Processing details	Execution condition	Number of basic steps	Subset	Reference
Output	OUT		• Device output		*1	—	Page 150 Page 152 Page 156 Page 158
	SET		• Sets device	 Annunciator (F) 	*1	—	Page 160 Page 164
	RST		• Resets device	 Annunciator (F) 	*1	—	Page 162 Page 164
	PLS		• Generates 1 cycle program pulse at rising edge of input signal.		2	—	Page 166
	PLF		• Generates 1 cycle program pulse at falling edge of input signal.				
	FF		• Reversal of device output		2	—	Page 169
	DELTA		• Pulse conversion of direct output		2	—	Page 171
	DELTAP						

*1 The number of steps may vary depending on the device being used. See description pages of individual instructions for number of steps.

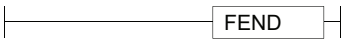
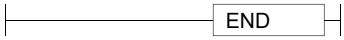
Shift instructions

Category	Instruction symbol	Symbol	Processing details	Execution condition	Number of basic steps	Subset	Reference
Shift	SFT		• 1-bit shift of device		2	—	Page 173
	SFTP						

Master control instructions

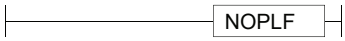
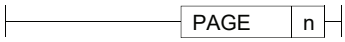
Category	Instruction symbol	Symbol	Processing details	Execution condition	Number of basic steps	Subset	Reference
Master control	MC		• Starts master control		2	—	Page 175
	MCR		• Resets master control		1		

Termination instructions

Category	Instruction symbol	Symbol	Processing details	Execution condition	Number of basic steps	Subset	Reference
Termination	FEND		• Termination of main program		1*1	1	Page 179
	END		• Termination of sequence program			1	Page 181

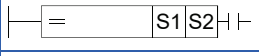
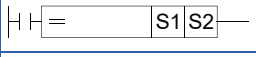
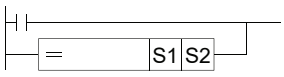
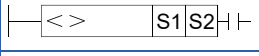
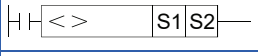
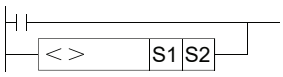
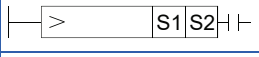
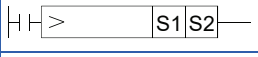
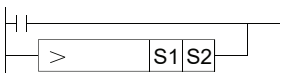
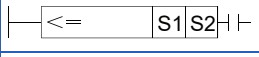
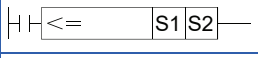
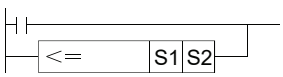
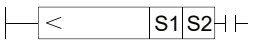
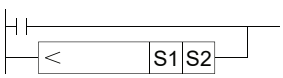
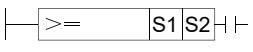
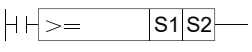
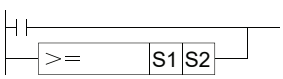
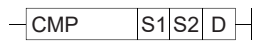
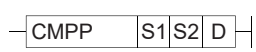
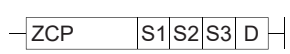
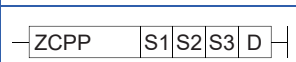
*1 For the High-speed Universal model QCPU and Universal model Process CPU, the number of basic steps is two.

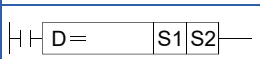
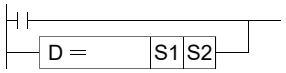
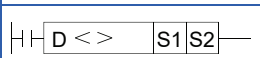
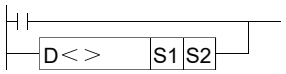
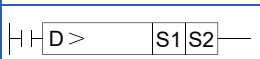
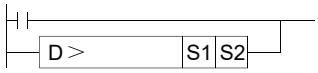
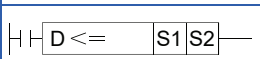
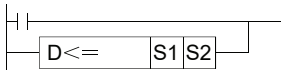
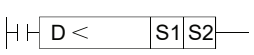
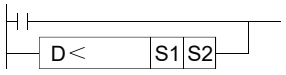
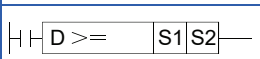
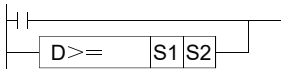
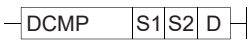
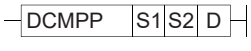
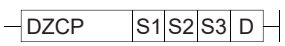
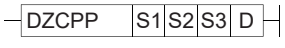
Other instructions

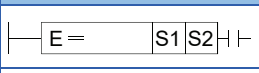
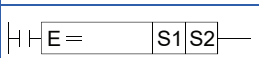
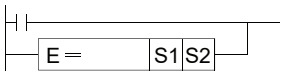
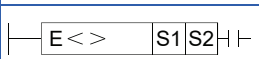
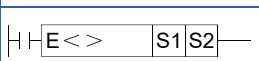
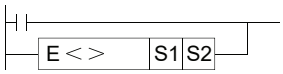
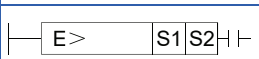
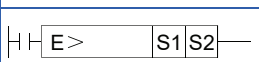
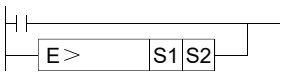
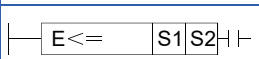
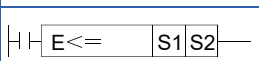
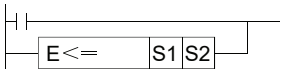
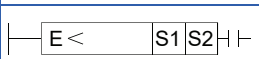
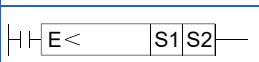
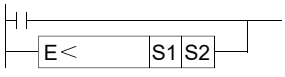
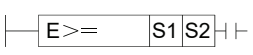
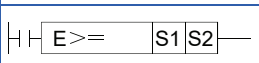
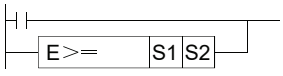
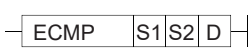
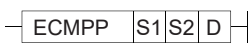
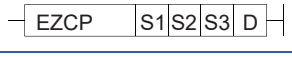
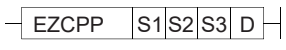
Category	Instruction symbol	Symbol	Processing details	Execution condition	Number of basic steps	Subset	Reference
Stop	STOP		• Terminates sequence operation after input condition has been met. • Sequence program is executed by placing the RUN/STOP key switch back in the RUN position.		1	—	Page 183
Ignored	NOP	—	• Ignored (For program deletion or space)		1	—	Page 185
	NOPLF		• Ignored (To change pages during printouts)				
	PAGE n		• Ignored (Subsequent programs will be controlled from step 0 of page n)				

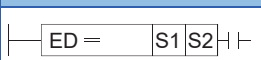
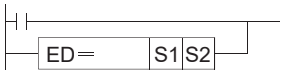
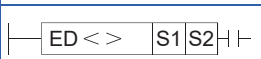
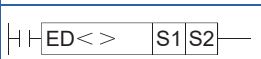
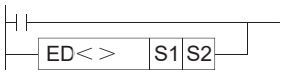
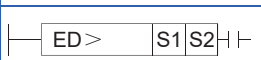
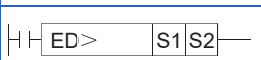
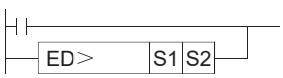
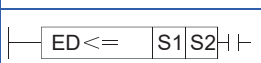
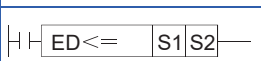
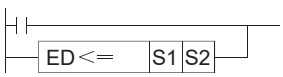
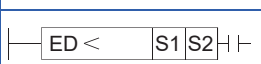
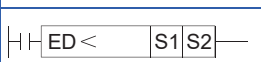
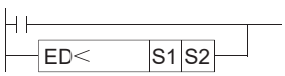
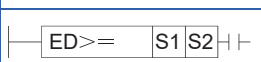
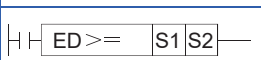
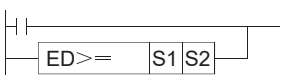
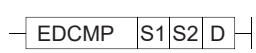
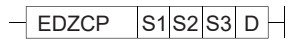
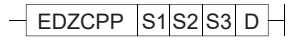
2.4 Basic Instructions

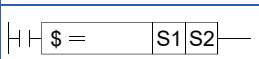
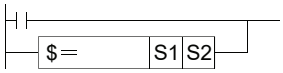
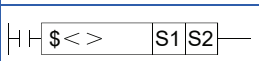
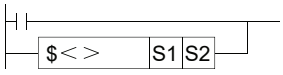
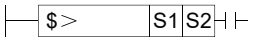
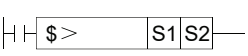
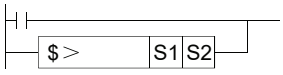
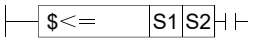
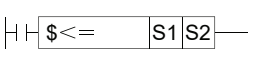
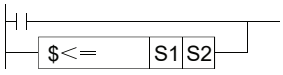
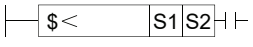
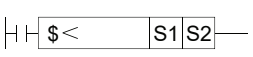
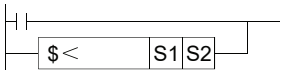
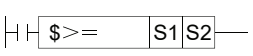
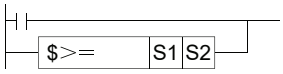
Comparison operation instructions

Category	Instruction symbol	Symbol	Processing details	Execution condition	Number of basic steps	Subset	Reference
BIN 16-bit data comparisons	LD=		• Conductive status when (S1) = (S2) • Non-conductive status when (S1) ≠ (S2)		3	●	Page 189
	AND=						
	OR=						
	LD<>		• Conductive status when (S1) ≠ (S2) • Non-conductive status when (S1) = (S2)		3	●	
	AND<>						
	OR<>						
	LD>		• Conductive status when (S1) > (S2) • Non-conductive status when (S1) ≤ (S2)		3	●	
	AND>						
	OR>						
	LD<=		• Conductive status when (S1) ≤ (S2) • Non-conductive status when (S1) > (S2)		3	●	
	AND<=						
	OR<=						
	LD<		• Conductive status when (S1) < (S2) • Non-conductive status when (S1) ≥ (S2)		3	●	
	AND<						
	OR<						
	LD>=		• Conductive status when (S1) ≥ (S2) • Non-conductive status when (S1) < (S2)		3	●	
	AND>=						
	OR>=						
BIN 16-bit data comparisons	CMP		• (D) Conductive status when (S1) > (S2) • (D)+1 Conductive status when (S1) = (S2) • (D)+2 Conductive status when (S1) < (S2)		4	—	Page 207
	CMPP						
	ZCP		• (D) Conductive status when (S1) > (S3) • (D)+1 Conductive status when (S1) ≤ (S3) ≤ (S2) • (D)+2 Conductive status when (S3) > (S2)		5	—	Page 210
	ZCPP						

Category	Instruction symbol	Symbol	Processing details	Execution condition	Number of basic steps	Subset	Reference
BIN 32-bit data comparisons	LDD=		• Conductive status when (S1+1, S1) = (S2+1, S2) • Non-Conductive status when (S1+1, S1) ≠ (S2+1, S2)		*1	●	Page 191
	ANDD=						
	ORD=						
	LDD<>		• Conductive status when (S1+1, S1) ≠ (S2+1, S2) • Non-Conductive status when (S1+1, S1) = (S2+1, S2)		*1	●	
	ANDD<>						
	ORD<>						
	LDD>		• Conductive status when (S1+1, S1) > (S2+1, S2) • Non-Conductive status when (S1+1, S1) ≤ (S2+1, S2)		*1	●	
	ANDD>						
	ORD>						
	LDD<=		• Conductive status when (S1+1, S1) ≤ (S2+1, S2) • Non-Conductive status when (S1+1, S1) > (S2+1, S2)		*1	●	
	ANDD<=						
	ORD<=						
	LDD<		• Conductive status when (S1+1, S1) < (S2+1, S2) • Non-Conductive status when (S1+1, S1) ≥ (S2+1, S2)		*1	●	
	ANDD<						
	ORD<						
	LDD>=		• Conductive status when (S1+1, S1) ≥ (S2+1, S2) • Non-Conductive status when (S1+1, S1) < (S2+1, S2)		*1	●	
	ANDD>=						
	ORD>=						
BIN 32-bit data comparisons	DCMP		• (D) Conductive status when (S1, S1+1) > (S2, S2+1) • (D)+1 Conductive status when (S1, S1+1) = (S2, S2+1) • (D)+2 Conductive status when (S1, S1+1) < (S2, S2+1)		4	—	Page 209
	DCMPP						
	DZCP		• (D) Conductive status when (S1, S1+1) > (S3, S3+1) • (D)+1 Conductive status when (S1, S1+1) ≤ (S3, S3+1) ≤ (S2, S2+1) • (D)+2 Conductive status when (S3, S3+1) > (S2, S2+1)		5	—	Page 212
	DZCPP						

Category	Instruction symbol	Symbol	Processing details	Execution condition	Number of basic steps	Subset	Reference
Floating decimal point data comparisons (single precision)	LDE=		• Conductive status when (S1+1, S1) = (S2+1, S2) • Non-Conductive status when (S1+1, S1) ≠ (S2+1, S2)		3	—	Page 193
	ANDE=						
	ORE=						
	LDE<>		• Conductive status when (S1+1, S1) ≠ (S2+1, S2) • Non-Conductive status when (S1+1, S1) = (S2+1, S2)		3	—	
	ANDE<>						
	ORE<>						
	LDE>		• Conductive status when (S1+1, S1) > (S2+1, S2) • Non-Conductive status when (S1+1, S1) ≤ (S2+1, S2)		3	—	
	ANDE>						
	ORE>						
	LDE<=		• Conductive status when (S1+1, S1) ≤ (S2+1, S2) • Non-Conductive status when (S1+1, S1) > (S2+1, S2)		3	—	
	ANDE<=						
	ORE<=						
	LDE<		• Conductive status when (S1+1, S1) < (S2+1, S2) • Non-Conductive status when (S1+1, S1) ≥ (S2+1, S2)		3	—	
	ANDE<						
	ORE<						
	LDE>=		• Conductive status when (S1+1, S1) ≥ (S2+1, S2) • Non-Conductive status when (S1+1, S1) < (S2+1, S2)		3	—	
	ANDE>=						
	ORE>=						
Floating point comparisons (single precision)	ECMP		• (D) is on when (S1, S1+1) > (S2, S2+1) • (D)+1 is on when (S1, S1+1) = (S2, S2+1) • (D)+2 is on when (S1, S1+1) < (S2, S2+1)		4	—	Page 214
	ECMPP						
Floating point band comparisons (single precision)	EZCP		• (D) is on when (S1, S1+1) > (S3, S3+1) • (D)+1 is on when (S1, S1+1) ≤ (S3, S3+1) ≤ (S2, S2+1) • (D)+2 is on when (S3, S3+1) > (S2, S2+1)		5	—	Page 218
	EZCPP						

Category	Instruction symbol	Symbol	Processing details	Execution condition	Number of basic steps	Subset	Reference
Floating decimal point data comparisons (double precision)	LDED=		• Conductive status when (S1+3, S1+2, S1+1, S1) = (S2+3, S2+2, S2+1, S2) • Non-Conductive status when (S1+3, S1+2, S1+1, S1) ≠ (S2+3, S2+2, S2+1, S2)		3	—	Page 195
	ANDED=						
	ORED=						
	LDED<>		• Conductive status when (S1+3, S1+2, S1+1, S1) ≠ (S2+3, S2+2, S2+1, S2) • Non-Conductive status when (S1+3, S1+2, S1+1, S1) = (S2+3, S2+2, S2+1, S2)		3	—	
	ANDED<>						
	ORED<>						
	LDED>		• Conductive status when (S1+3, S1+2, S1+1, S1) > (S2+3, S2+2, S2+1, S2) • Non-Conductive status when (S1+3, S1+2, S1+1, S1) ≤ (S2+3, S2+2, S2+1, S2)		3	—	
	ANDED>						
	ORED>						
	LDED<=		• Conductive status when (S1+3, S1+2, S1+1, S1) ≤ (S2+3, S2+2, S2+1, S2) • Non-Conductive status when (S1+3, S1+2, S1+1, S1) > (S2+3, S2+2, S2+1, S2)		3	—	
	ANDED<=						
	ORED<=						
	LDED<		• Conductive status when (S1+3, S1+2, S1+1, S1) < (S2+3, S2+2, S2+1, S2) • Non-Conductive status when (S1+3, S1+2, S1+1, S1) ≥ (S2+3, S2+2, S2+1, S2)		3	—	
	ANDED<						
	ORED<						
	LDED>=		• Conductive status when (S1+3, S1+2, S1+1, S1) ≥ (S2+3, S2+2, S2+1, S2) • Non-Conductive status when (S1+3, S1+2, S1+1, S1) < (S2+3, S2+2, S2+1, S2)		3	—	
	ANDED>=						
	ORED>=						
Floating point comparisons (double precision)	EDCMP		• (D) is on when (S1 to S1+3) > (S2 to S2+3) • (D)+1 is on when (S1 to S1+3) = (S2 to S2+3) • (D)+2 is on when (S1 to S1+3) < (S2 to S2+3)		4	—	Page 216
	EDCMPP						
Floating point band comparisons (double precision)	EDZCP		• (D) is on when (S1 to S1+3) > (S3 to S3+3) • (D)+1 is on when (S1 to S1+3) ≤ (S3 to S3+3) ≤ (S2 to S2+3) • (D)+2 is on when (S3 to S3+3) > (S2 to S2+3)		5	—	Page 220
	EDZCPP						

Category	Instruction symbol	Symbol	Processing details	Execution condition	Number of basic steps	Subset	Reference
Character string data comparisons	LD$=		• Compares character string S1 and character string S2 one character at a time.*2 • Conductive status when (character string S1) = (character string S2) • Non-Conductive status when (character string S1) ≠ (character string S2)		3	—	Page 198
	AND$=						
	OR$=						
	LD$<>		• Compares character string S1 and character string S2 one character at a time.*2 • Conductive status when (character string S1) ≠ (character string S2) • Non-Conductive status when (character string S1) = (character string S2)		3	—	
	AND$<>						
	OR$<>						
	LD$>		• Compares character string S1 and character string S2 one character at a time.*2 • Conductive status when (character string S1) > (character string S2) • Non-Conductive status when (character string S1) ≤ (character string S2)		3	—	
	AND$>						
	OR$>						
	LD$<=		• Compares character string S1 and character string S2 one character at a time.*2 • Conductive status when (character string S1) ≤ (character string S2) • Non-Conductive status when (character string S1) > (character string S2)		3	—	
	AND$<=						
	OR$<=						
	LD$<		• Compares character string S1 and character string S2 one character at a time.*2 • Conductive status when (character string S1) < (character string S2) • Non-Conductive status when (character string S1) ≥ (character string S2)		3	—	
	AND$<						
	OR$<						
	LD$>=		• Compares character string S1 and character string S2 one character at a time.*2 • Conductive status when (character string S1) ≥ (character string S2) • Non-Conductive status when (character string S1) < (character string S2)		3	—	
	AND$>=						
	OR$>=						

Category	Instruction symbol	Symbol	Processing details	Execution condition	Number of basic steps	Subset	Reference
BIN 16-bit Block data comparisons	BKCMP=	BKCMP= S1 S2 D n	• This instruction compares BIN 16-bit data stored in n-point devices starting from the device specified by S1 with BIN 16-bit data stored in n-point devices starting from the device specified by S2, and then stores the result into the nth device specified by (D) and up.		5	—	Page 201
	BKCMP<>	BKCMP<> S1 S2 D n					
	BKCMP>	BKCMP> S1 S2 D n					
	BKCMP<=	BKCMP<= S1 S2 D n					
	BKCMP<	BKCMP< S1 S2 D n					
	BKCMP>=	BKCMP>= S1 S2 D n					
	BKCMP=P	BKCMP=P S1 S2 D n					
	BKCMP<>P	BKCMP<>P S1 S2 D n					
	BKCMP>P	BKCMP>P S1 S2 D n					
	BKCMP<=P	BKCMP<=P S1 S2 D n					
	BKCMP<P	BKCMP<P S1 S2 D n					
	BKCMP>=P	BKCMP>=P S1 S2 D n					
BIN 32-bit block data comparisons	DBKCMP=	DBKCMP= S1 S2 D n	• This instruction compares BIN 32-bit data stored in n-point devices starting from the device specified by S1 with BIN 32-bit data stored in n-point devices starting from the device specified by a constant and S2, and then stores the result into the nth device specified by (D) and up.		5	—	Page 204
	DBKCMP<>	DBKCMP<> S1 S2 D n					
	DBKCMP>	DBKCMP> S1 S2 D n					
	DBKCMP<=	DBKCMP<= S1 S2 D n					
	DBKCMP<	DBKCMP< S1 S2 D n					
	DBKCMP>=	DBKCMP>= S1 S2 D n					
	DBKCMP=P	DBKCMP=P S1 S2 D n					
	DBKCMP<>P	DBKCMP<>P S1 S2 D n					
	DBKCMP>P	DBKCMP>P S1 S2 D n					
	DBKCMP<=P	DBKCMP<=P S1 S2 D n					
	DBKCMP<P	DBKCMP<P S1 S2 D n					
	DBKCMP>=P	DBKCMP>=P S1 S2 D n					

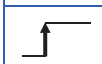
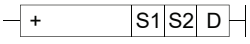
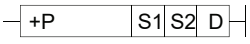
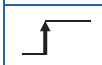
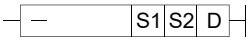
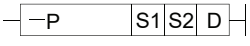
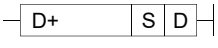
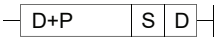
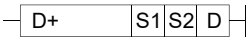
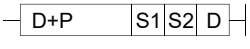
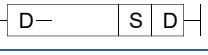
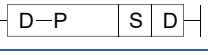
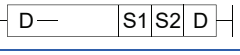
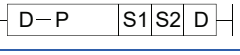
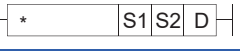
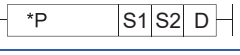
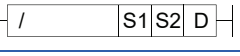
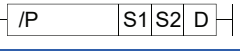
*1 The number of steps may differ, depending on the device or CPU module to be used.

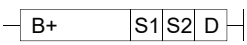
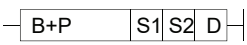
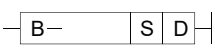
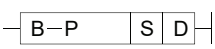
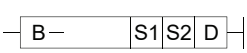
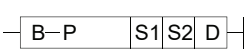
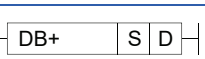
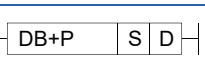
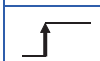
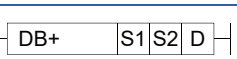
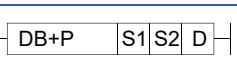
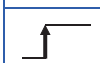
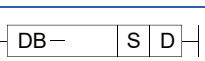
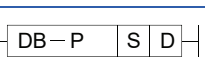
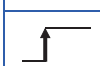
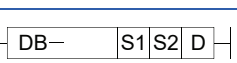
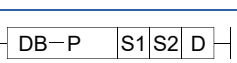
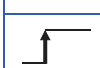
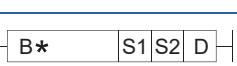
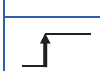
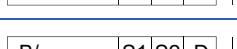
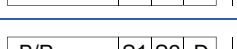
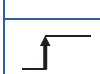
CPU module	Device	Number of steps	Remark
High Performance model QCPU Process CPU Redundant CPU	• Word device: Internal device (except for file register ZR) • Bit device: Devices whose device Nos. are multiples of 16, whose digit designation is K8, and which use no indexing. • Constant: No limitations	5	When using a High Performance model QCPU, Process CPU or Redundant CPU, the number of steps increases but the processing speed becomes faster.
	Devices other than above	3	The number of steps may increase depending on the conditions. ☞ Page 119 Conditions for increasing the number of steps
Basic model QCPU Universal model QCPU LCPU	All devices that can be used	3	

*2 The conditions under which character string comparisons can be made are as shown below:

- Match: All characters in the strings must match
- Larger string: If character strings are different, determines the string with the largest number of character codes. If the lengths of the character strings are different, determines the longest character string.
- Smaller string: If the character strings are different, determines the string with the smallest number of character codes. If the lengths of the character strings are different, determines the shortest character string.

Arithmetic operation instruction

Category	Instruction symbol	Symbol	Processing details	Execution condition	Number of basic steps	Subset	Reference
BIN 16-bit addition and subtraction operations	+		• D+(S)→(D)		3	●	Page 222
	+P						
	+		• (S1)+(S2)→(D)		4	●	Page 224
	+P						
	-		• (D)-(S)→(D)		3	●	Page 222
	-P						
	-		• (S1)-(S2)→(D)		4	●	Page 224
	-P						
BIN 32-bit addition and subtraction operations	D+		• (D+1, D)+(S+1, S)→(D+1, D)		*1	●	Page 226
	D+P						
	D+		• (S1+1, S1)+(S2+1, S2)→(D+1, D)		*2	●	Page 228
	D+P						
	D-		• (D+1, D)-(S+1, S)→(D+1, D)		*1	●	Page 226
	D-P						
	D-		• (S1+1, S1)-(S2+1, S2)→(D+1, D)		*2	●	Page 228
	D-P						
BIN 16-bit multiplication and division operations	*		• (S1)×(S2)→(D+1, D)		*3	●	Page 230
	*P						
	/		• (S1)÷(S2)→Quotient(D), Remainder (D+1)		4*4	●	
	/P						
BIN 32-bit multiplication and division operations	D*		• (S1+1, S1)×(S2+1, S2)→(D+3, D+2, D+1, D)		4*4	●	Page 232
	D*P						
	D/		• (S1+1, S1)÷(S2+1, S2)→Quotient (D+1, D), Remainder (D+3, D+2)		4*4	●	
	D/P						

Category	Instruction symbol	Symbol	Processing details	Execution condition	Number of basic steps	Subset	Reference
BCD 4-digit addition and subtraction operations	B+		• (D)+(S)→(D)		3	●	Page 234
B+P						
B+		• (S1)+(S2)→(D)		4	—	Page 236
B+P						
B-		• (D)-(S)→(D)		3	●	Page 234
B-P						
B-		• (S1)-(S2)→(D)		4	—	Page 236
B-P						
BCD 8-digit addition and subtraction operations	DB+		• (D+1, D)+(S+1, S)→(D+1, D)		3	—	Page 238
DB+P						
DB+		• (S1+1, S1)+(S2+1, S2)→(D+1, D)		4	—	Page 240
DB+P						
DB-		• (D+1, D)-(S+1, S)→(D+1, D)		3	—	Page 238
DB-P						
DB-		• (S1+1, S1)-(S2+1, S2)→(D+1, D)		4	—	Page 240
DB-P						
BCD 4-digit multiplication and division operations	B*		• (S1)×(S2)→(D+1, D)		4	●	Page 242
B*P						
B/		• (S1)÷(S2)→Quotient(D), Remainder (D+1)		4	●	
B/P						
BCD 8-digit multiplication and division operations	DB*		• (S1+1, S1)×(S2+1, S2)→(D+3, D+2, D+1, D)		4	—	Page 244
DB*P						
DB/		• (S1+1, S1)÷(S2+1, S2)→Quotient (D+1, D), Remainder (D+3, D+2)		4	●	
DB/P						

Category	Instruction symbol	Symbol	Processing details	Execution condition	Number of basic steps	Subset	Reference
Floating decimal point data addition and subtraction operations (single precision)	E+	E+ S D	• (D+1, D)+(S+1, S)→(D+1, D)		3	●*6	Page 246
	E+P	E+P S D					
	E+	E+ S1 S2 D	• (S1+1, S1)+(S2+1, S2)→(D+1, D)		4*4	●*6	Page 248
	E+P	E+P S1 S2 D					
	E-	E− S D	• (D+1, D)-(S+1, S)→(D+1, D)		3	●*6	Page 246
	E-P	E−P S D					
	E-	E− S1 S2 D	• (S1+1, S1)-(S2+1, S2)→(D+1, D)		4*4	●*6	Page 248
	E-P	E−P S1 S2 D					
Floating decimal point data addition and subtraction operations (double precision)	ED+	ED+ S D	• (D+3, D+2, D+1, D)+(S+3, S+2, S+1, S)→(D+3, D+2, D+1, D)		3	●	Page 250
	ED+P	ED+P S D					
	ED+	ED+ S1 S2 D	• (S1+3, S1+2, S1+1, S1)+(S2+3, S2+2, S2+1, S2)→(D+3, D+2, D+1, D)		4	●	Page 252
	ED+P	ED+P S1 S2 D					
	ED-	ED− S D	• (D+3, D+2, D+1, D)-(S+3, S+2, S+1, S)→(D+3, D+2, D+1, D)		3	●	Page 250
	ED-P	ED−P S D					
	ED-	ED− S1 S2 D	• (S1+3, S1+2, S1+1, S1)-(S2+3, S2+2, S2+1, S2)→(D+3, D+2, D+1, D)		4	●	Page 252
	ED-P	ED−P S1 S2 D					
Floating decimal point data multiplication and division operations (single precision)	E*	E* S1 S2 D	• (S1+1, S1)×(S2+1, S2)→(D+1, D)		3	●*6	Page 254
	E*P	E*P S1 S2 D					
	E/	E/ S1 S2 D	• (S1+1, S1)÷(S2+1, S2)→Quotient (D+1, D)		4	●*6	
	E/P	E/P S1 S2 D					
Floating decimal point data multiplication and division operations (double precision)	ED*	ED* S1 S2 D	• (S1+3, S1+2, S1+1, S1)×(S2+3, S2+2, S2+1, S2)→(D+3, D+2, D+1, D)		4	●	Page 256
	ED*P	ED*P S1 S2 D					
	ED/	ED/ S1 S2 D	• (S1+3, S1+2, S1+1, S1)÷(S2+3, S2+2, S2+1, S2)→Quotient (D+3, D+2, D+1, D)		4	●	
	ED/P	ED/P S1 S2 D					

Category	Instruction symbol	Symbol	Processing details	Execution condition	Number of basic steps	Subset	Reference
BIN 16-bit data block addition and subtraction operations	BK+	BK+ S1 S2 D n	• This instruction adds BIN 16-bit data stored in n-point devices starting from the device specified by (S1) to the n-point data stored in the devices starting from the device specified by (S2) in batch.		5	—	Page 258
	BK+P	BK+P S1 S2 D n					
	BK-	BK— S1 S2 D n	• This instruction subtracts BIN 16-bit data stored in the n-point devices starting from the devices specified by (S2) from BIN 16-bit data stored in n-point devices starting from the device specified by (S1) in batch.		5	—	
	BK-P	BK—P S1 S2 D n					
BIN 32-bit data block addition and subtraction operations	DBK+	DBK+ S1 S2 D n	• Adds BIN 32-bit data stored in the n-point devices starting from the device specified by (S1) and a constant to BIN 32-bit data stored in the n-point devices starting from the device specified by (S2) and stores the result into the nth device specified by (D) and up.		5	—	Page 261
	DBK+P	DBK+P S1 S2 D n					
	DBK-	DBK— S1 S2 D n	• Subtracts BIN 32-bit data stored in the n-point devices starting from the device specified by (S2) or a constant from BIN 32-bit data stored in n-point devices starting from the device specified by (S1) and stores the operation result into the nth device specified by (D) and up.		5	—	
	DBK-P	DBK—P S1 S2 D n					
Character string data Connection	$+	$+ S D	• Links character string designated with (S) to character string designated with (D), and stores the result from (D) onward.		3	—	Page 264
	$+P	$+P S D					
	$+	$+ S1 S2 D	• Links character string designated with (S2) to character string designated with (S1), and stores the result from (D) onward.		4	—	Page 266
	$+P	$+P S1 S2 D					
BIN data increment, decrement	INC	INC D	• (D)+1→(D)		2	●	Page 267
	INCP	INCP D					
	DINC	DINC D	• (D+1, D)+1→(D+1, D)		*5	●	Page 269
	DINCP	DINCP D					
	DEC	DEC D	• (D)-1→(D)		2	●	Page 267
	DECP	DECP D					
	DDEC	DDEC D	• (D+1, D)-1→(D+1, D)		*5	●	Page 269
	DDECP	DDECP D					

*1 The number of steps may differ, depending on the device or CPU module to be used.

CPU module	Device	Number of steps	Remark
High Performance model QCPU Process CPU Redundant CPU	• Word device: Internal device (except for file register ZR) • Bit device: Devices whose device Nos. are multiples of 16, whose digit designation is K8, and which use no indexing. • Constant: No limitations	5	When using a High Performance model QCPU, Process CPU or Redundant CPU, the number of steps increases but the processing speed becomes faster.
	Devices other than above	3	The number of steps may increase depending on the conditions. ☞ Page 119 Conditions for increasing the number of steps
Basic model QCPU Universal model QCPU LCPU	All devices that can be used	3	

*2 The number of steps may differ, depending on the device or CPU module to be used.

CPU module	Device	Number of steps	Remark
High Performance model QCPU Process CPU Redundant CPU	• Word device: Internal device (except for file register ZR) • Bit device: Devices whose device Nos. are multiples of 16, whose digit designation is K8, and which use no indexing. • Constant: No limitations	6	When using a High Performance model QCPU, Process CPU or Redundant CPU, the number of steps increases but the processing speed becomes faster.
	Devices other than above	4	The number of steps may increase depending on the conditions. ☞ Page 119 Conditions for increasing the number of steps
Basic model QCPU	All devices that can be used	4	
Universal model QCPU LCPU		3	

*3 The number of steps may differ, depending on the device or CPU module to be used.

CPU module	Device	Number of steps	Remark
QCPU, LCPU	• Word device: Internal device (except for file register ZR) • Bit device: Devices whose device Nos. are multiples of 16, whose digit designation is K8, and which use no indexing. • Constant: No limitations	3	—
	Devices other than above	4	The number of steps may increase depending on the conditions. ☞ Page 119 Conditions for increasing the number of steps

*4 The number of basic steps is three for the Universal model QCPU and LCPU.

*5 The number of steps may differ, depending on the device or CPU module to be used.

CPU module	Device	Number of steps	Remark
High Performance model QCPU Process CPU Redundant CPU	• Word device: Internal device (except for file register ZR) • Bit device: Devices whose device Nos. are multiples of 16, whose digit designation is K8, and which use no indexing. • Constant: No limitations	3	When using a High Performance model QCPU, Process CPU or Redundant CPU, the number of steps increases but the processing speed becomes faster.
	Devices other than above	2	The number of steps may increase depending on the conditions. ☞ Page 119 Conditions for increasing the number of steps
Basic model QCPU Universal model QCPU LCPU	All devices that can be used	2	

*6 The subset is effective only with Universal model QCPU and LCPU.

Data conversion instructions

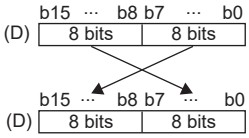
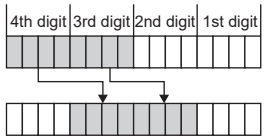
Category	Instruction symbol	Symbol	Processing details	Execution condition	Number of basic steps	Subset	Reference
BCD conversions	BCD	BCD S D	· (S) → (D) BCD conversions; BIN (0 to 9999)		3*1	●	Page 271
	BCDP	BCDP S D					
	DBCD	DBCD S D	· (S+1, S) → (D+1, D) BCD conversions; BIN (0 to 99999999)		3*1	●	
	DBCDP	DBCDP S D					
BIN conversions	BIN	BIN S D	· (S) → (D) BIN conversions; BCD (0 to 9999)		3*1	●	Page 273
	BINP	BINP S D					
	DBIN	DBIN S D	· (S+1, S) → (D+1, D) BIN conversions; BCD (0 to 99999999)		3*1	●	
	DBINP	DBINP S D					
BIN → Floating point conversions (single precision)	FLT	FLT S D	· (S) → (D+1, D) Conversion to real number; BIN(−32768 to 32767)		3*1	●*2	Page 275
	FLTP	FLTP S D					
	DFLT	DFLT S D	· (S+1, S) → (D+1, D) Conversion to real number; BIN(−2147483648 to 2147483647)		3*1	●*2	
	DFLTP	DFLTP S D					
BIN → Floating point conversions (double precision)	FLTD	FLTD S D	· (S) → (D+3, D+2, D+1, D) Conversion to real number; BIN(−32768 to 32767)		4	●	Page 277
	FLTDP	FLTDP S D					
	DFLTD	DFLTD S D	· (S+1, S) → (D+3, D+2, D+1, D) Conversion to real number; BIN(−2147483648 to 2147483647)		4	●	
	DFLTDP	DFLTDP S D					
Floating point (single precision) → BIN conversions	INT	INT S D	· (S+1, S) → (D) Conversion to BIN; Real number (-32768 to 32767)		3*1	●*2	Page 279
	INTP	INTP S D					
	DINT	DINT S D	· (S+1, S) → (D+1, D) Conversion to BIN; Real number (-2147483648 to 2147483647)		3*1	●*2	
	DINTP	DINTP S D					
Floating point (double precision) → BIN conversions	INTD	INTD S D	· (S+3, S+2, S+1, S) → (D) Conversion to BIN; Real number (-32768 to 32767)		3	●	Page 281
	INTDP	INTDP S D					
	DINTD	DINTD S D	· (S+3, S+2, S+1, S) → (D+1, D) Conversion to BIN; Real number (-2147483648 to 2147483647)		3	●	
	DINTDP	DINTDP S D					

Category	Instruction symbol	Symbol	Processing details	Execution condition	Number of basic steps	Subset	Reference
BIN 16-bit ↔ 32-bit conversions	DBL	DBL S D	· (S) → (D+1, D) Conversion; BIN (-32768 to 32767)		3*3	—	Page 283
	DBLP	DBLP S D					
	WORD	WORD S D	· (S+1, S) → (D) Conversion; BIN (-32768 to 32767)		3*3	—	Page 284
	WORDP	WORDP S D					
BIN → Gray code conversions	GRY	GRY S D	Conversion to gray code · (S) → (D); BIN (-32768 to 32767)		3*3	—	Page 285
	GRYP	GRYP S D					
	DGRY	DGRY S D	Conversion to gray code · (S+1, S) → (D+1, D); BIN (-2147483648 to 2147483647)		3*3	—	
	DGRYP	DGRYP S D					
Gray code → BIN conversions	GBIN	GBIN S D	Conversion to BIN data · (S) → (D); Gray code (-32768 to 32767)		3*3	—	Page 287
	GBINP	GBINP S D					
	DGBIN	DGBIN S D	Conversion to BIN data · (S+1, S) → (D+1, D); Gray code (-2147483648 to 2147483647)		3*3	—	
	DGBINP	DGBINP S D					
Complement to 2	NEG	NEG D	· (D) → (D); BIN data		2	—	Page 289
	NEGP	NEGP D					
	DNEG	DNEG D	· (D+1, D) → (D+1, D); BIN data		2	—	
	DNEGP	DNEGP D					
	ENEG	ENEG D	· (D+1, D) → (D+1, D); Real number data		2	—	Page 291
	ENEGP	ENEGP D					
	EDNEG	EDNEG D	· (D+3, D+2, D+1, D) → (D+3, D+2, D+1, D); Real number data		3	—	Page 292
	EDNEGP	EDNEGP D					
Block conversion	BKBCD	BKBCD S D n	• Batch converts BIN data n points from (S) to BCD data and stores the result from (D) onward.		4	—	Page 293
	BKBCDP	BKBCDP S D n					
	BKBIN	BKBIN S D n	• Batch converts BCD data n points from (S) to BIN data and stores the result from (D) onward.		4	—	Page 295
	BKBINP	BKBINP S D n					
Floating-point Single precision → Double precision	ECON	ECON S D	· (S+1, S) → (D+3, D+2, D+1, D) Conversion to double precision; 32-bit floating-point real number		3	—	Page 297
	ECONP	ECONP S D					
Floating-point Double precision → Single precision	EDCON	EDCON S D	· (S+3, S+2, S+1, S) → (D+1, D) Conversion to single precision; 64-bit floating-point real number		3	—	Page 298
	EDCONP	EDCONP S D					

*1 The number of basic steps is two for the Universal model QCPU and LCPU.
*2 The subset is effective only with Universal model QCPU and LCPU.
*3 For the High-speed Universal model QCPU and Universal model Process CPU, the number of basic steps is two.

Data transfer instruction

Category	Instruction symbol	Symbol	Processing details	Execution condition	Number of basic steps	Subset	Reference
16-bit data transfer	MOV	MOV S D	• (S)→(D)		*1	●	Page 299
	MOVP	MOVP S D					
32-bit data transfer	DMOV	DMOV S D	• (S+1, S)→(D+1, D)		*2	●	
	DMOVP	DMOVP S D					
Floating decimal point data transfer (single precision)	EMOV	EMOV S D	· (S+1, S) → (D+1, D) Real number data		*2	●*3	Page 301
	EMOVP	EMOVP S D					
Floating decimal point data transfer (double precision)	EDMOV	EDMOV S D	·(S+3, S+2, S+1, S)→(D+3, D+2, D+1, D) Real number data		2	●*3	Page 303
	EDMOVP	EDMOVP S D					
Character string data transfer	$MOV	$MOV S D	• Transfers character string designated by (S) to device designated by (D) onward.		3	—	Page 304
	$MOVP	$MOVP S D					
16-bit data negation transfer	CML	CML S D	• $\overline{(S)}$→(D)		*1	●	Page 306
	CMLP	CMLP S D					
32-bit data negation transfer	DCML	DCML S D	• $\overline{(S+1, S)}$→(D+1, D)		*2	●	
	DCMLP	DCMLP S D					
Block transfer	BMOV	BMOV S D n	(S) (D) n		4	●	Page 309
	BMOVP	BMOVP S D n					
Identical 16-bit data block transfers	FMOV	FMOV S D n	(S) (D) n		4	●	Page 312
	FMOVP	FMOVP S D n					
Identical 32-bit data block transfers	DFMOV	DFMOV S D n	(S+1,S) (D+1,D) n		4	●	Page 314
	DFMOVP	DFMOVP S D n					
16-bit data exchange	XCH	XCH D1 D2	• (D1)↔(D2)		3	●	Page 316
	XCHP	XCHP D1 D2					
32-bit data exchange	DXCH	DXCH D1 D2	• (D1+1, D1)↔(D2+1, D2)		3	●	
	DXCHP	DXCHP D1 D2					
Block data exchange	BXCH	BXCH D1 D2 n	(D1) (D2) n		4	—	Page 318
	BXCHP	BXCHP D1 D2 n					

Category	Instruction symbol	Symbol	Processing details	Execution condition	Number of basic steps	Subset	Reference
Exchange of upper and lower bytes	SWAP	SWAP D			3	—	Page 320
	SWAPP	SWAPP D					
Shift	SMOV	SMOV S n1 n2 D n3			6	—	Page 321
	SMOVP	SMOVP S n1 n2 D n3					

*1 The number of steps may differ, depending on the device or CPU module to be used.

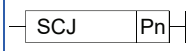
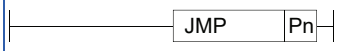
CPU module	Device	Number of steps	Remark
QCPU, LCPU	• Word device: Internal device (except for file register ZR) • Bit device: Devices whose device Nos. are multiples of 16, whose digit designation is K4, and which use no indexing. • Constant: No limitations	2	—
	Devices other than above	3	The number of steps may increase depending on the conditions. ☞ Page 119 Conditions for increasing the number of steps

*2 The number of steps may differ, depending on the device or CPU module to be used.

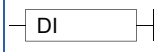
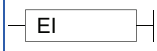
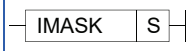
CPU module	Device	Number of steps	Remark
High Performance model QCPU Process CPU Redundant CPU	• Word device: Internal device (except for file register ZR) • Bit device: Devices whose device Nos. are multiples of 16, whose digit designation is K8, and which use no indexing. • Constant: No limitations	3	—
	Devices other than above	3	The number of steps may increase depending on the conditions. ☞ Page 119 Conditions for increasing the number of steps
Basic model QCPU	• Word device: Internal device (except for file register ZR) • Bit device: Devices whose device Nos. are multiples of 16, whose digit designation is K8, and which use no indexing. • Constant: No limitations (The number of steps is 3 when the above device + constant are used.)	2	—
	Devices other than above	3	The number of steps may increase depending on the conditions. ☞ Page 119 Conditions for increasing the number of steps
Universal model QCPU LCPU	All devices that can be used	2	

*3 The subset is effective only with Universal model QCPU and LCPU.

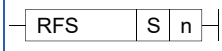
Program branch instructions

Category	Instruction symbol	Symbol	Processing details	Execution condition	Number of basic steps	Subset	Reference
Jump	CJ		• Jumps to Pn when input conditions are met.		2	●	Page 323
	SCJ		• Jumps to Pn from the scan after the meeting of input condition.		2	●	
	JMP		• Jumps unconditionally to Pn.		2	●	
	GOEND		• Jumps to END instruction when input condition is met.		1	—	Page 326

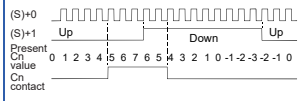
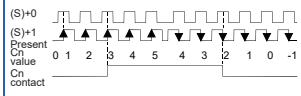
Program execution control instructions

Category	Instruction symbol	Symbol	Processing details	Execution condition	Number of basic steps	Subset	Reference
Disable interrupts	DI		• Prohibits the running of an interrupt program.		1	—	Page 327
Enable interrupts	EI		• Resets interrupt program execution prohibition.		1	—	
Interrupt disable/ enable setting	IMASK		• Inhibits or permits interrupts for each interrupt program.		2	—	
Return	IRET		• Returns to sequence program from an interrupt program.		1	—	Page 333

I/O refresh instructions

Category	Instruction symbol	Symbol	Processing details	Execution condition	Number of basic steps	Subset	Reference
I/O Refresh	RFS		• Refreshes the relevant I/O area during scan.		3	—	Page 334
	RFSP						

Other convenient instructions

Category	Instruction symbol	Symbol	Processing details	Execution condition	Number of basic steps	Subset	Reference
Up/Down counter	UDCNT1	UDCNT1 S D n			4	—	Page 336
	UDCNT2	UDCNT2 S D n			4	—	Page 338
Teaching timer	TTMR	TTMR D n	• (Time that TTMR is ON) × n → (D) n=0:1, n=1:10, n=2:100		3	—	Page 340
Special timer	STMR	STMR S n D	The 4 points from the bit device designated by (D) operate as shown below, depending on the ON/OFF status of the input conditions for the STMR instruction: • (D)+0: Off delay timer output • (D)+1: One shot after off timer output • (D)+2: One shot after on timer output • (D)+3: On delay and off delay timer output		3	—	Page 342
Shortest direction control	ROTC	ROTC S n1 n2 D	• Rotates a rotary table with n1 divisions from the stop position to the position designated by (S+1) in the shortest direction.		5	—	Page 345
Ramp signal	RAMP	RAMP n1 n2 D1 n3 D2	• Changes device data designated by D1 from n1 to n2 in n3 scans.		6	—	Page 347
Pulse density	SPD	SPD S n D	• Counts the pulse input from the device designated by (S) for the duration of time designated by n, and stores the count in the device designated by (D).		4	—	Page 349
Fixed cycle pulse output	PLSY	PLSY n1 n2 D	• Outputs a pulse at a frequency designated by n1 the number of times designated by n2, to the output number (Y) designated by (D).		4	—	Page 351
Pulse width modulation	PWM	PWM n1 n2 D	• Outputs the pulse of the cycle set by n2, for the amount of time ON designated by n1, to the output number (Y) designated by (D).		4	—	Page 353
Matrix input	MTR	MTR S D1 D2 n	• Reads data of 16 points × n rows from the devices starting from the one specified by (S), and stores them to the devices starting from the one specified by (D2).		5	—	Page 355

2.5 Application Instructions

Logical operation instructions

Category	Instruction symbol	Symbol	Processing details	Execution condition	Number of basic steps	Subset	Reference
Logical product	WAND	WAND S D	• (D)∧(S)→(D)		3	●	Page 358
	WANDP	WANDP S D					
	WAND	WAND S1 S2 D	• (S1)∧(S2)→(D)		4*1	●	Page 360
	WANDP	WANDP S1 S2 D					
	DAND	DAND S D	• (D+1, D)∧(S+1, S)→(D+1, D)		*2	●	Page 358
	DANDP	DANDP S D					
	DAND	DAND S1 S2 D	• (S1+1, S1)∧(S2+1, S2)→(D+1, D)		*3	●	Page 360
	DANDP	DANDP S1 S2 D					
	BKAND	BKAND S1 S2 D n	(S1) (S2) (D) ∧ → n		5	—	Page 363
	BKANDP	BKANDP S1 S2 D n					
Logical sum	WOR	WOR S D	• (D)∨(S)→(D)		3	●	Page 365
	WORP	WORP S D					
	WOR	WOR S1 S2 D	• (S1)∨(S2)→(D)		4*1	●	Page 367
	WORP	WORP S1 S2 D					
	DOR	DOR S D	• (D+1, D)∨(S+1, S)→(D+1, D)		*2	●	Page 365
	DORP	DORP S D					
	DOR	DOR S1 S2 D	• (S1+1, S1)∨(S2+1, S2)→(D+1, D)		*3	●	Page 367
	DORP	DORP S1 S2 D					
	BKOR	BKOR S1 S2 D n	(S1) (S2) (D) ∨ → n		5	—	Page 369
	BKORP	BKORP S1 S2 D n					

Category	Instruction symbol	Symbol	Processing details	Execution condition	Number of basic steps	Subset	Reference
Exclusive OR	WXOR	WXOR S D	• (D)∀(S)→(D)		3	●	Page 371
	WXORP	WXORP S D					
	WXOR	WXOR S1 S2 D	• (S1)∀(S2)→(D)		4*1	●	Page 373
	WXORP	WXORP S1 S2 D					
	DXOR	DXOR S D	• (D+1, D)∀(S+1, S)→(D+1, D)		*2	●	Page 371
	DXORP	DXORP S D					
	DXOR	DXOR S1 S2 D	• (S1+1, S1)∀(S2+1, S2)→(D+1, D)		*3	●	Page 373
	DXORP	DXORP S1 S2 D					
	BKXOR	BKXOR S1 S2 D n	(S1) (S2) (D) n		5	—	Page 375
	BKXORP	BKXORP S1 S2 D n					
NON exclusive logical sum	WXNR	WXNR S D	· $\overline{(D)\veebar(S)}$ →(D)		3	●	Page 377
	WXNRP	WXNRP S D					
	WXNR	WXNR S1 S2 D	· $\overline{(S1)\veebar(S2)}$ →(D)		4*1	●	Page 379
	WXNRP	WXNRP S1 S2 D					
	DXNR	DXNR S D	· $\overline{(D+1,D)\veebar(S+1,S)}$ →(D+1,D)		*2	●	Page 377
	DXNRP	DXNRP S D					
	DXNR	DXNR S1 S2 D	· $\overline{(S1+1,S1)\veebar(S2+1,S2)}$ →(D+1,D)		*3	●	Page 379
	DXNRP	DXNRP S1 S2 D					
	BKXNR	BKXNR S1 S2 D n	(S1) (S2) (D) n		5	—	Page 381
	BKXNRP	BKXNRP S1 S2 D n					

*1 The number of basic steps is three for the Universal model QCPU and LCPU.

*2 The number of steps may differ, depending on the device or CPU module to be used.

CPU module	Device	Number of steps	Remark
High Performance model QCPU Process CPU Redundant CPU	• Word device: Internal device (except for file register ZR) • Bit device: Devices whose device Nos. are multiples of 16, whose digit designation is K8, and which use no indexing. • Constant: No limitations	5	When using a High Performance model QCPU, Process CPU or Redundant CPU, the number of steps increases but the processing speed becomes faster.
	Devices other than above	3	The number of steps may increase depending on the conditions. ☞ Page 119 Conditions for increasing the number of steps
Basic model QCPU Universal model QCPU LCPU	All devices that can be used	3	

*3 The number of steps may differ, depending on the device or CPU module to be used.

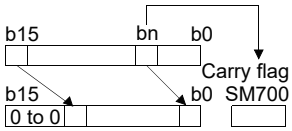
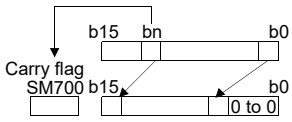
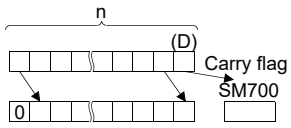
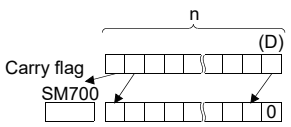
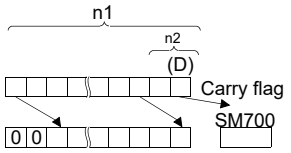
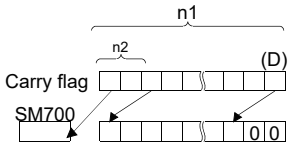
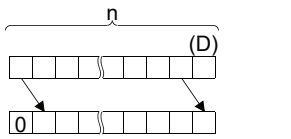
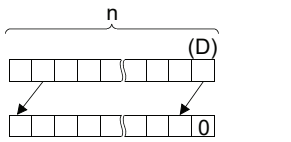
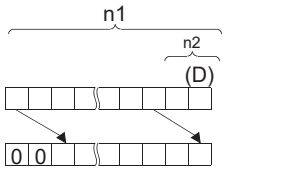
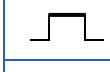
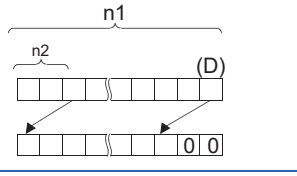
CPU module	Device	Number of steps	Remark
High Performance model QCPU Process CPU Redundant CPU	• Word device: Internal device (except for file register ZR) • Bit device: Devices whose device Nos. are multiples of 16, whose digit designation is K8, and which use no indexing. • Constant: No limitations	6	When using a High Performance model QCPU, Process CPU or Redundant CPU, the number of steps increases but the processing speed becomes faster.
	Devices other than above	4	The number of steps may increase depending on the conditions. ☞ Page 119 Conditions for increasing the number of steps
Basic model QCPU	All devices that can be used	4	
Universal model QCPU LCPU	All devices that can be used	3	

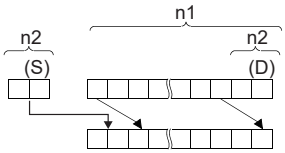
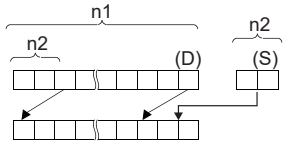
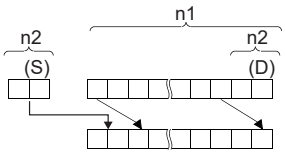
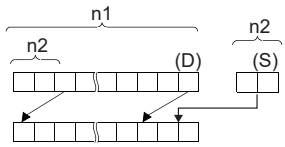
Rotation instructions

Category	Instruction symbol	Symbol	Processing details	Execution condition	Number of basic steps	Subset	Reference
Right rotation of 16-bit data	ROR	ROR D n	b15 (D) b0 SM700 Right rotation by n bits Carry flag		3*1	●	Page 383
	RORP	RORP D n					
	RCR	RCR D n	b15 (D) b0 SM700 Right rotation by n bits Carry flag		3*1	●	
	RCRP	RCRP D n					
Left rotation of 16-bit data	ROL	ROL D n	SM700 b15 (D) b0 Carry flag Left rotation by n bits		3*1	●	Page 386
	ROLP	ROLP D n					
	RCL	RCL D n	SM700 b15 (D) b0 Carry flag Left rotation by n bits		3*1	●	
	RCLP	RCLP D n					
Right rotation of 32-bit data	DROR	DROR D n	(D+1) (D) b31 to b16 b15 to b0 SM700 Right rotation by n bits Carry flag		3*1	●	Page 389
	DRORP	DRORP D n					
	DRCR	DRCR D n	(D+1) (D) b31 to b16 b15 to b0 SM700 Right rotation by n bits Carry flag		3*1	●	
	DRCRP	DRCRP D n					
Left rotation of 32-bit data	DROL	DROL D n	(D+1) (D) SM700 b31 to b16 b15 to b0 Carry flag Left rotation by n bits		3*1	●	Page 391
	DROLP	DROLP D n					
	DRCL	DRCL D n	(D+1) (D) SM700 b31 to b16 b15 to b0 Carry flag Left rotation by n bits		3*1	●	
	DRCLP	DRCLP D n					

*1 For the High-speed Universal model QCPU and Universal model Process CPU, the number of basic steps is four.

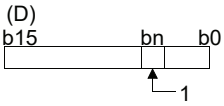
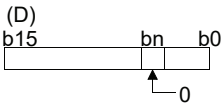
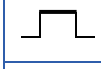
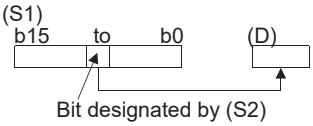
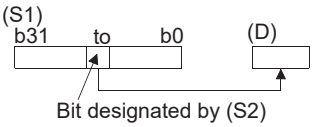
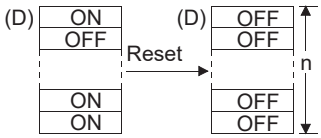
Shift instructions

Category	Instruction symbol	Symbol	Processing details	Execution condition	Number of basic steps	Subset	Reference
n-bit shift of 16-bit data	SFR	SFR D n			3*1	●	Page 393
	SFRP	SFRP D n					
	SFL	SFL D n			3*1	●	
	SFLP	SFLP D n					
1-bit shift of n-bit data	BSFR	BSFR D n			3	—	Page 396
	BSFRP	BSFRP D n					
	BSFL	BSFL D n			3	—	
	BSFLP	BSFLP D n					
n-bit shift of n-bit data	SFTBR	SFTBR D n1 n2			4	—	Page 398
	SFTBRP	SFTBRP D n1 n2					
	SFTBL	SFTBL D n1 n2			4	—	
	SFTBLP	SFTBLP D n1 n2					
1-word shift of n-words data	DSFR	DSFR D n			3	●	Page 401
	DSFRP	DSFRP D n					
	DSFL	DSFL D n			3	●	
	DSFLP	DSFLP D n					
n-words shift of n-words data	SFTWR	SFTWR D n1 n2			4	—	Page 403
	SFTWRP	SFTWRP D n1 n2					
	SFTWL	SFTWL D n1 n2			4	—	
	SFTWLP	SFTWLP D n1 n2					

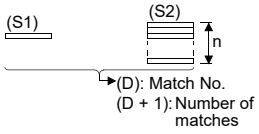
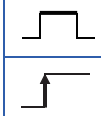
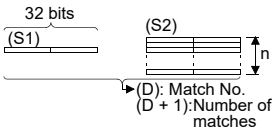
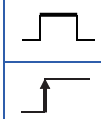
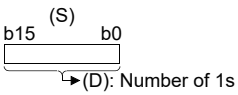
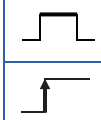
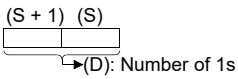
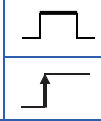
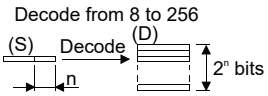
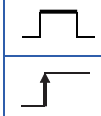
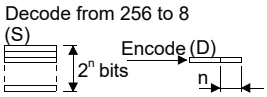
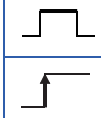
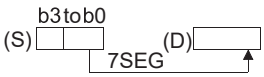
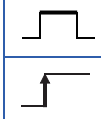
Category	Instruction symbol	Symbol	Processing details	Execution condition	Number of basic steps	Subset	Reference
Bit shift right/shift left	SFTR	SFTR S D n1 n2			5	—	Page 406
	SFTRP	SFTRP S D n1 n2					
	SFTL	SFTL S D n1 n2			5	—	Page 408
	SFTLP	SFTLP S D n1 n2					
Word shift right/shift left	WSFR	WSFR S D n1 n2			5	—	Page 410
	WSFRP	WSFRP S D n1 n2					
	WSFL	WSFL S D n1 n2			5	—	Page 410
	WSFLP	WSFLP S D n1 n2					

*1 For the High-speed Universal model QCPU and Universal model Process CPU, the number of basic steps is four.

Bit processing instructions

Category	Instruction symbol	Symbol	Processing details	Execution condition	Number of basic steps	Subset	Reference
Bit set/reset	BSET	BSET D n			3	●	Page 414
	BSETP	BSETP D n					
	BRST	BRST D n			3	●	
	BRSTP	BRSTP D n					
Bit tests	TEST	TEST S1 S2 D			4	—	Page 416
	TESTP	TESTP S1 S2 D					
	DTEST	DTEST S1 S2 D			4	—	
	DTESTP	DTESTP S1 S2 D					
Batch reset of bit devices	BKRST	BKRST D n			3	—	Page 418
	BKRSTP	BKRSTP D n					

Data processing instructions

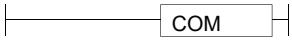
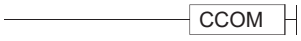
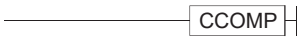
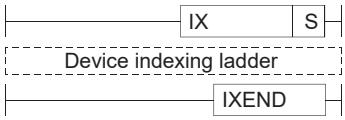
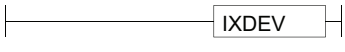
Category	Instruction symbol	Symbol	Processing details	Execution condition	Number of basic steps	Subset	Reference
Data searches	SER	SER S1 S2 D n			5	—	Page 420
	SERP	SERP S1 S2 D n					
	DSER	DSER S1 S2 D n			5	—	
	DSERP	DSERP S1 S2 D n					
Bit checks	SUM	SUM S D			3	●	Page 423
	SUMP	SUMP S D					
	DSUM	DSUM S D			3	●	
	DSUMP	DSUMP S D					
Decode	DECO	DECO S D n			4	—	Page 425
	DECOP	DECOP S D n					
Encode	ENCO	ENCO S D n			4	—	Page 427
	ENCOP	ENCOP S D n					
7-segment decode	SEG	SEG S D			3	●	Page 429
	SEGP	SEGP S D					

Category	Instruction symbol	Symbol	Processing details	Execution condition	Number of basic steps	Subset	Reference
Separating and linking	DIS	DIS S D n	• Separates 16-bit data designated by (S) into 4-bit units, and stores at the lower 4 bits of n points from (D). (n≤4)		4	—	Page 431
	DISP	DISP S D n					
	UNI	UNI S D n	• Links the lower 4 bits of n points from the device designated by (S) and stores at the device designated by (D). (n≤4)		4	—	Page 433
	UNIP	UNIP S D n					
	NDIS	NDIS S1 D S2	• Separates the data in the devices starting from the one specified by (S1) into bits specified by the devices from (S2), and stores them to the devices starting from the one specified by (D).		4	—	Page 435
	NDISP	NDISP S1 D S2					
	NUNI	NUNI S1 D S2	• Links the data in the devices starting from the one specified by (S1) with bits specified by the devices from (S2), and stores them to the devices starting from the one specified by (D).				
	NUNIP	NUNIP S1 D S2					
	WTOB	WTOB S D n	• Breaks n points of 16-bit data from the device designated by (S) into 8-bit units, and stores in sequence at the device designated by (D).		4	—	Page 439
	WTOBP	WTOBP S D n					
	BTOW	BTOW S D n	• Links the lower 8 bits of 16-bit data of n points from the device designated by (S) into 16-bit units, and stores in sequence at the device designated by (D).				
	BTOWP	BTOWP S D n					
Search	MAX	MAX S D n	• Searches the data of n points from the device designated by (S) in 16-bit units, and stores the maximum value at the device designated by (D).		4	—	Page 442
	MAXP	MAXP S D n					
	MIN	MIN S D n	• Searches the data of n points from the device designated by (S) in 16-bit units, and stores the minimum value at the device designated by (D).				Page 444
	MINP	MINP S D n					
	DMAX	DMAX S D n	• Searches the data of 2×n points from the device designated by (S) in 32-bit units, and stores the maximum value at the device designated by (D).		4	—	Page 442
	DMAXP	DMAXP S D n					
	DMIN	DMIN S D n	• Searches the data of 2×n points from the device specified by (S) in 32-bit units, and stores the minimum value in the device specified by (D).				Page 444
	DMINP	DMINP S D n					

Category	Instruction symbol	Symbol	Processing details	Execution condition	Number of basic steps	Subset	Reference
Sort	SORT	SORT S1 n S2 D1 D2 • S2: Number of comparisons to be made during a single run • D1: Device to be turned ON at the completion of sort • D2: For system use	• Sorts data of n points from device designated by (S1) in 16-bit units. (n × (n-1)/2 scans required)		6	—	Page 446
	DSORT	DSORT S1 n S2 D1 D2 • S2: Number of comparisons to be made during a single run • D1: Device to be turned ON at the completion of sort • D2: For system use	• Sorts data of 2×n points from device designated by (S1) in 32-bit units. (n × (n-1)/2 scans required)				
Total value calculations	WSUM	WSUM S D n	• Adds 16 bit BIN data of n points from the device specified by (S), and stores it in the device specified by (D).		4	—	Page 450
	WSUMP	WSUMP S D n					
	DWSUM	DWSUM S D n	• Adds 32 bit BIN data of n points from the device specified by (S), and stores it in the device specified by (D).				Page 451
	DWSUMP	DWSUMP S D n					
Calculation of averages	MEAN	MEAN S D n	• Calculates the mean of n-point devices (in 16-bit units) starting from the device specified by (S), and then stores the result into the device specified by (D).		4	—	Page 452
	MEANP	MEANP S D n					
	DMEAN	DMEAN S D n	• Calculates the mean of n-point devices (in 32-bit units) starting from the device specified by (S), and then stores the result into the device specified by (D).				
	DMEANP	DMEANP S D n					
Check code	CCD	CCD S D n	• Performs addition of the data stored in the devices designated by (S) to (S)+n-1 and calculates the horizontal parity, and stores the added data in the device designated by (D) and the horizontal parity in the device designated by (D)+1.		4	—	Page 454
	CCDP	CCDP S D n					
CRC operation	CRC	CRC S D n	• Generates the CRC value of n points of 8-bit data starting from the device designated by (S) and stores it in the device designated by (D).		4	—	Page 457
	CRCP	CRCP S D n					

Structure creation instructions

Category	Instruction symbol	Symbol	Processing details	Execution condition	Number of basic steps	Subset	Reference
Number of repeats	FOR	FOR n	• Executes n times between the [FOR] and [NEXT].		2	—	Page 459
	NEXT	NEXT			1	—	
	BREAK	BREAK D Pn	• Forcibly ends the execution of the [FOR] to [NEXT] cycle and jumps pointer Pn.	⌐‾¬	3	—	Page 462
	BREAKP	BREAKP D Pn		⌐↑			
Subroutine program calls	CALL	CALL Pn CALL Pn S1~Sn	• Executes subroutine program Pn when input condition is met. (S1 to Sn are arguments sent to subroutine program. n≤5)	⌐‾¬	2+n*1	●*3	Page 464
	CALLP	CALLP Pn CALLP Pn S1~Sn		⌐↑			
	RET	RET	• Returns from subroutine program		1	—	Page 469
	FCALL	FCALL Pn FCALL Pn S1~Sn	• Performs non-execution processing of subroutine program Pn if input conditions have not been met. (S1 to Sn are arguments sent to subroutine program. n≤5)	⌐‾¬	2+n*1	—	Page 470
	FCALLP	FCALLP Pn FCALLP Pn Sn~S1		⌐↓			
	ECALL	ECALL * Pn ECALL * Pn S1~Sn *: File name	• Executes subroutine program Pn from within designated program name when input condition is met. (S1 to Sn are arguments sent to subroutine program. n≤5)	⌐‾¬	3+n*2	—	Page 474
	ECALLP	ECALLP * ECALLP * Pn S1~Sn *: File name		⌐↑			
Subroutine program calls	EFCALL	EFCALL * Pn EFCALL * Pn S1toSn *:File name	• Performs non-execution processing of subroutine program Pn if input conditions have not been met. (S1 to Sn are arguments sent to subroutine program. n≤5)	⌐‾¬	3+n*2	—	Page 479
	EFCALLP	EFCALLP * Pn EFCALLP * Pn S1toSn *:File name		⌐↓			
	XCALL	XCALL Pn S1~Sn	• Executes subroutine program Pn when input condition is met. • Performs non-execution processing of subroutine program Pn if input conditions have not been met. (S1 to Sn are arguments sent to subroutine program. n≤5)	⌐‾¬	2+n*1	—	Page 483

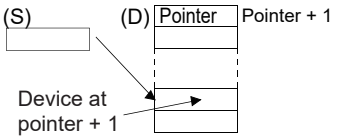
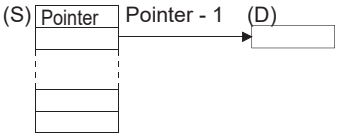
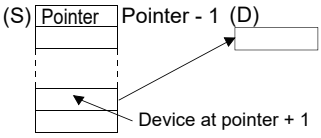
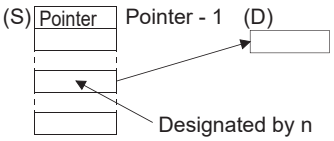
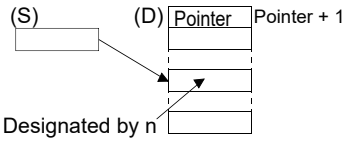
Category	Instruction symbol	Symbol	Processing details	Execution condition	Number of basic steps	Subset	Reference
Select refresh	COM		• Performs auto refresh of intelligent function modules, auto refresh of link refresh, and communications with peripherals.		1	—	Page 489
			• Performs auto refresh of intelligent function modules, link refresh, auto refresh of CPU shared memory, and communications with peripherals.		1	—	Page 491
	CCOM		• Performs auto refresh of intelligent function modules, auto refresh of CPU shared memory, and communications with peripherals after the input conditions are met.		1	—	Page 494
	CCOMP				1	—	
Fixed indexing	IX		• Perform indexing for individual devices used in device indexing ladder.		2	—	Page 495
	IXEND				1	—	
	IXDEV		• Stores indexing value used for indexing performed between the [IX] and [IXEND] to the device designated by D or later.		1	—	Page 498
	IXSET				3	—	

*1 n indicates number of arguments for subroutine program.

*2 n indicates the total of the number of arguments used in the subroutine program and the number of program name steps.
The number of program name steps is calculated as "number of characters in the program ÷ 2" (decimal fraction is rounded up).

*3 The subset is effective only with the Universal model QCPU and LCPU.

Data table operation instructions

Category	Instruction symbol	Symbol	Processing details	Execution condition	Number of basic steps	Subset	Reference
Data table processing	FIFW	FIFW S D			3	—	Page 500
	FIFWP	FIFWP S D					
	FIFR	FIFR S D			3	—	Page 502
	FIFRP	FIFRP S D					
	FPOP	FPOP S D			3	—	Page 504
	FPOPP	FPOPP S D					
	FDEL	FDEL S D n			4	—	Page 506
	FDELP	FDELP S D n					
	FINS	FINS S D n			4	—	Page 506
	FINSP	FINSP S D n					

Buffer memory access instructions

Category	Instruction symbol	Symbol	Processing details	Execution condition	Number of basic steps	Subset	Reference
Data read	FROM	FROM n1 n2 D n3	• Reads data in 16-bit units from an intelligent function module.		5	—	Page 508
	FROMP	FROMP n1 n2 D n3					
	DFRO	DFRO n1 n2 D n3	• Reads data in 32-bit units from an intelligent function module.		5	—	
	DFROP	DFROP n1 n2 D n3					
Data write	TO	TO n1 n2 S n3	• Writes data in 16-bit units to an intelligent function module.		5	—	Page 511
	TOP	TOP n1 n2 S n3					
	DTO	DTO n1 n2 S n3	• Writes data in 32-bit units to an intelligent function module.		5	—	
	DTOP	DTOP n1 n2 S n3					

Display instructions

Category	Instruction symbol	Symbol	Processing details	Execution condition	Number of basic steps	Subset	Reference
ASCII print	PR	* When SM701 is OFF PR S D	• Outputs ASCII code from device designated by (S) to 00H to output module.	↑	3	—	Page 514
	PR	* When SM701 is ON PR S D	• Outputs ASCII code of 8 points (16 characters) from device designated by (S) to output module.				
	PRC	PRC S D	• Converts comments from device designated by (S) to ASCII code and outputs to output module.				Page 517
Reset	LEDR	LEDR	• Resets an annunciator.	↑	1	—	Page 520

Debugging and failure diagnosis instructions

Category	Instruction symbol	Symbol	Processing details	Execution condition	Number of basic steps	Subset	Reference
Checks	CHKST	CHKST	• The CHK instruction is executed when CHKST is executable. • Jumps to the step following the CHK instruction when CHKST is in a non-executable status.		1	—	Page 522
	CHK	CHK Check condition	• During normal conditions → SM80: OFF, SD80: 0 • During abnormal conditions → SM80: ON, SD80: Failure No.				
	CHKCIR	CHKCIR	• Starts update in the ladder pattern being checked by the CHK instruction.		1	—	Page 526
	CHKEND	CHKEND	• Ends update in the ladder pattern being checked by the CHK instruction.				

Character string processing instructions

Category	Instruction symbol	Symbol	Processing details	Execution condition	Number of basic steps	Subset	Reference
BIN → Decimal ASCII	BINDA	BINDA S D	• Converts 1-word BIN value designated by (S) to a 5-digit, decimal ASCII value, and stores it at the word device designated by (D).		3	—	Page 530
	BINDAP	BINDAP S D					
	DBINDA	DBINDA S D	• Converts 2-word BIN value designated by (S) to a 10-digit, decimal ASCII value, and stores it at word devices following the word device number designated by (D).		3	—	
	DBINDAP	DBINDAP S D					
BIN → Hexadecimal ASCII	BINHA	BINHA S D	• Converts 1-word BIN value designated by (S) to a 4-digit, hexadecimal ASCII value, and stores it at a word device following the word device number designated by (D).		3	—	Page 533
	BINHAP	BINHAP S D					
	DBINHA	DBINHA S D	• Converts 2-word BIN value designated by (S) to a 8-digit, hexadecimal ASCII value, and stores it at a word device following the word device number designated by (D).		3	—	
	DBINHAP	DBINHAP S D					
BCD → Decimal ASCII	BCDDA	BCDDA S D	• Converts 1-word BCD value designated by (S) to a 4-digit, decimal ASCII value, and stores it at a word device following the word device number designated by (D).		3	—	Page 536
	BCDDAP	BCDDAP S D					
	DBCDDA	DBCDDA S D	• Converts 2-word BCD value designated by (S) to a 8-digit, decimal ASCII value, and stores it at a word device following the word device number designated by (D).		3	—	
	DBCDDAP	DBCDDAP S D					
Decimal ASCII → BIN	DABIN	DABIN S D	• Converts a 5-digit, decimal ASCII value designated by (S) to a 1-word BIN value, and stores it at a word device number designated by (D).		3	—	Page 539
	DABINP	DABINP S D					
	DDABIN	DDABIN S D	• Converts a 10-digit, decimal ASCII value designated by (S) to a 2-word BIN value, and stores it at a word device number designated by (D).		3	—	
	DDABINP	DDABINP S D					
Hexadecimal ASCII → BIN	HABIN	HABIN S D	• Converts a 4-digit, hexadecimal ASCII value designated by (S) to a 1-word BIN value, and stores it at a word device number designated by (D).		3	—	Page 542
	HABINP	HABINP S D					
	DHABIN	DHABIN S D	• Converts a 8-digit, hexadecimal ASCII value designated by (S) to a 2-word BIN value, and stores it at a word device number designated by (D).		3	—	
	DHABINP	DHABINP S D					

Category	Instruction symbol	Symbol	Processing details	Execution condition	Number of basic steps	Subset	Reference
Decimal ASCII → BCD	DABCD	DABCD S D	• Converts a 4-digit, decimal ASCII value designated by (S) to a 1-word BCD value, and stores it at a word device number designated by (D).		3	—	Page 545
	DABCDP	DABCDP S D					
	DDABCD	DDABCD S D	• Converts a 8-digit, decimal ASCII value designated by (S) to a 2-word BCD value, and stores it at a word device number designated by (D).		3	—	
	DDABCDP	DDABCDP S D					
Device comment read operation	COMRD	COMRD S D	• Stores comment from device designated by (S) at a device designated by (D).		3	—	Page 548
	COMRDP	COMRDP S D					
Character string length detection	LEN	LEN S D	• Stores data length (number of characters) in character string designated by (S) at a device designated by (D).		3	—	Page 551
	LENP	LENP S D					
BIN → Decimal character string	STR	STR S1 S2 D	• Converts a 1-word BIN value designated by (S2) to a decimal character string with the total number of digits and the number of decimal fraction digits designated by (S1) and stores them at a device designated by (D).		4	—	Page 553
	STRP	STRP S1 S2 D					
	DSTR	DSTR S1 S2 D	• Converts a 2-word BIN value designated by (S2) to a decimal character string with the total number of digits and the number of decimal fraction digits designated by (S1) and stores them at a device designated by (D).		4	—	
	DSTRP	DSTRP S1 S2 D					
Decimal character string → BIN	VAL	VAL S D1 D2	• Converts a character string including decimal point designated by (S) to a 1-word BIN value and the number of decimal fraction digits, and stores them into devices designated by (D1) and (D2).		4	—	Page 558
	VALP	VALP S D1 D2					
	DVAL	DVAL S D1 D2	• Converts a character string including decimal point designated by (S) to a 2-word BIN value and the number of decimal fraction digits, and stores them into devices designated by (D1) and (D2).		4	—	
	DVALP	DVALP S D1 D2					
Floating decimal point → Character string	ESTR	ESTR S1 S2 D	• Converts the 32-bit floating decimal point data designated by (S) to a character string, and stores it in devices designated by (D).		4	—	Page 563
	ESTRP	ESTRP S1 S2 D					
Character string → Floating decimal point	EVAL	EVAL S D	• Converts the character string designated by (S) to a 32-bit floating decimal point data, and stores it in devices designated by (D).		3	—	Page 570
	EVALP	EVALP S D					

Category	Instruction symbol	Symbol	Processing details	Execution condition	Number of basic steps	Subset	Reference
Hexadecimal BIN → ASCII	ASC	ASC S D n	• Converts the 1-word BIN value at the device numbers designated by (S) to hexadecimal ASCII, and stores n characters of them at the device numbers designated by (D) and after.		4	—	Page 574
	ASCP	ASCP S D n					
ASCII → Hexadecimal BIN	HEX	HEX S D n	• Converts n hexadecimal ASCII characters of the device numbers designated by (S) and after to BIN values, and stores them at the device numbers designated by (D).		4	—	Page 576
	HEXP	HEXP S D n					
Character string	RIGHT	RIGHT S D n	• Stores n characters from the end of a character string designated by (S) at the device designated by (D).		4	—	Page 578
	RIGHTP	RIGHTP S D n					
	LEFT	LEFT S D n	• Stores n characters from the beginning of a character string designated by (S) at the device designated by (D).				
	LEFTP	LEFTP S D n					
	MIDR	MIDR S1 D S2	• Stores the designated number of characters in the character string designated by (S1) from the position designated by (S2) at the device designated by (D).		4	—	Page 581
	MIDRP	MIDRP S1 D S2					
	MIDW	MIDW S1 D S2	• Stores the character string of (S1) in the specified number to the character string of (D) at the position specified by (S2).				
	MIDWP	MIDWP S1 D S2					
	INSTR	INSTR S1 S2 D n	• Searches character string (S1) from the nth character of character string (S2), and stores matched positions at (D).		5	—	Page 586
	INSTRP	INSTRP S1 S2 D n					
	STRINS	STRINS S D n	• Inserts the character string data specified by (S) to the (n)th character (insert position) from the initial character string data specified by (D).		4	—	Page 588
	STRINSP	STRINSP S D n					
	STRDEL	STRDEL D n1 n2	• Deletes the (n2) characters data specified by (D) starting from the device (insert position) specified by n1.		4	—	Page 590
	STRDELP	STRDELP D n1 n2					
Floating decimal point → BCD	EMOD	EMOD S1 S2 D	• Converts 32-bit floating decimal point data (S1) to BCD data with number of decimal fraction digits designated by (S2), and stores at device designated by (D).		4	—	Page 592
	EMODP	EMODP S1 S2 D					
BCD → Floating decimal point	EREXP	EREXP S1 S2 D	• Converts BCD data (S1) to 32-bit floating decimal point data with the number of decimal fraction digits designated by (S2), and stores at device designated by (D).		4	—	Page 594
	EREXPP	EREXPP S1 S2 D					

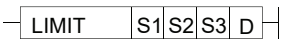
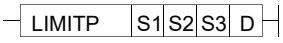
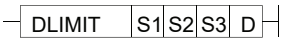
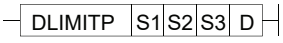
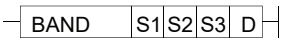
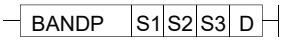
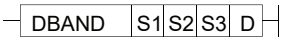
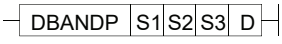
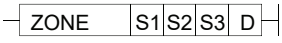
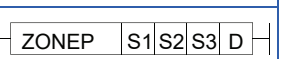
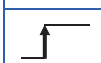
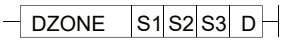
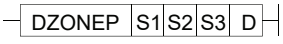
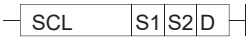
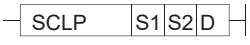
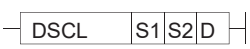
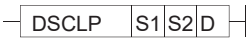
Special function instructions

Category	Instruction symbol	Symbol	Processing details	Execution condition	Number of basic steps	Subset	Reference
Trigonometric functions (Floating-point single-precision)	SIN	SIN S D	• Sin(S+1, S)→(D+1, D)		3	—	Page 596
	SINP	SINP S D					
	COS	COS S D	• Cos(S+1, S)→(D+1, D)		3	—	Page 600
	COSP	COSP S D					
	TAN	TAN S D	• Tan(S+1, S)→(D+1, D)		3	—	Page 604
	TANP	TANP S D					
	ASIN	ASIN S D	• Sin^{-1}(S+1, S)→(D+1, D)		3	—	Page 608
	ASINP	ASINP S D					
	ACOS	ACOS S D	• Cos^{-1}(S+1, S)→(D+1, D)		3	—	Page 612
	ACOSP	ACOSP S D					
	ATAN	ATAN S D	• Tan^{-1}(S+1, S)→(D+1, D)		3	—	Page 616
	ATANP	ATANP S D					
Trigonometric functions (Floating-point double-precision)	SIND	SIND S D	• Sin(S+3, S+2, S+1, S)→(D+3, D+2, D+1, D)		3	—	Page 598
	SINDP	SINDP S D					
	COSD	COSD S D	• Cos(S+3, S+2, S+1, S)→(D+3, D+2, D+1, D)		3	—	Page 602
	COSDP	COSDP S D					
	TAND	TAND S D	• Tan(S+3, S+2, S+1, S)→(D+3, D+2, D+1, D)		3	—	Page 606
	TANDP	TANDP S D					
	ASIND	ASIND S D	• Sin^{-1}(S+3, S+2, S+1, S)→(D+3, D+2, D+1, D)		3	—	Page 610
	ASINDP	ASINDP S D					
	ACOSD	ACOSD S D	• Cos^{-1}(S+3, S+2, S+1, S)→(D+3, D+2, D+1, D)		3	—	Page 614
	ACOSDP	ACOSDP S D					
	ATAND	ATAND S D	• Tan^{-1}(S+3, S+2, S+1, S)→(D+3, D+2, D+1, D)		3	—	Page 618
	ATANDP	ATANDP S D					

Category	Instruction symbol	Symbol	Processing details	Execution condition	Number of basic steps	Subset	Reference
Angles ↔ Radians conversion	RAD	RAD S D	• (S+1, S)→(D+1, D) Conversion from angles to radians		3	—	Page 620
	RADP	RADP S D					
	RADD	RADD S D	• (S+3, S+2, S+1, S)→(D+3, D+2, D+1, D) Conversion from angle to radian		3	—	Page 622
	RADDP	RADDP S D					
	DEG	DEG S D	• (S+1, S)→(D+1, D) Conversion from radians to angles		3	—	Page 624
	DEGP	DEGP S D					
	DEGD	DEGD S D	• (S+3, S+2, S+1, S)→(D+3, D+2, D+1, D) Conversion from radian to angle		3	—	Page 626
	DEGDP	DEGDP S D					
Exponentiation	POW	POW S1 S2 D	• $(S1+1, S1)^{(S2+1, S2)}$→(D+1, D)		4	—	Page 628
	POWP	POWP S1 S2 D					
	POWD	POWD S1 S2 D	• $(S1+3, S1+2, S1+1, S1)^{(S2+3, S2+2, S2+1, S2)}$→(D+3, D+2, D+1, D)		4	—	Page 630
	POWDP	POWDP S1 S2 D					
Square root	SQR	SQR S D	• $\sqrt{(S+1, S)}$→(D+1, D)		3	—	Page 632
	SQRP	SQRP S D					
	SQRD	SQRD S D	• $\sqrt{(S+3, S+2, S+1, S)}$→(D+3, D+2, D+1, D)		3	—	Page 634
	SQRDP	SQRDP S D					
Exponent operations	EXP	EXP S D	• $e^{(S+1, S)}$→(D+1, D)		3	—	Page 636
	EXPP	EXPP S D					
	EXPD	EXPD S D	• $e^{(S+3, S+2, S+1, S)}$→(D+3, D+2, D+1, D)		3	—	Page 638
	EXPDP	EXPDP S D					
Natural logarithms	LOG	LOG S D	• $\log_e$(S+1, S)→(D+1, D)		3	—	Page 640
	LOGP	LOGP S D					
	LOGD	LOGD S D	• $\log_e$(S+3, S+2, S+1, S)→(D+3, D+2, D+1, D)		3	—	Page 642
	LOGDP	LOGDP S D					
Common logarithm	LOG10	LOG10 S D	• $\log_{10}$(S+1, S)→(D+1, D)		3	—	Page 644
	LOG10P	LOG10P S D					
	LOG10D	LOG10D S D	• $\log_{10}$(S+3, S+2, S+1, S)→(D+3, D+2, D+1, D)		3	—	Page 646
	LOG10DP	LOG10DP S D					

Category	Instruction symbol	Symbol	Processing details	Execution condition	Number of basic steps	Subset	Reference
Random number generation	RND	RND D	• Generates a random number (from 0 to less than 32767) and stores it at the device designated by (D).		2	—	Page 648
	RNDP	RNDP D					
Random number series update	SRND	SRND S	• Updates random number series according to the 16-bit BIN data stored in the device designated by (S).				
	SRNDP	SRNDP S					
Square root	BSQR	BSQR S D	• $\sqrt{(S)}$ → (D)+0 Integer part +1 Decimal fraction part		3	—	Page 649
	BSQRP	BSQRP S D					
	BDSQR	BDSQR S D	• $\sqrt{(S+1, S)}$ → (D)+0 Integer part +1 Decimal fraction part		3	—	
	BDSQRP	BDSQRP S D					
Trigonometric functions	BSIN	BSIN S D	• Sin(S) → (D)+0 Sign +1 Integer part +2 Decimal fraction part		3	—	Page 652
	BSINP	BSINP S D					
	BCOS	BCOS S D	• Cos(S) → (D)+0 Sign +1 Integer part +2 Decimal fraction part		3	—	Page 654
	BCOSP	BCOSP S D					
	BTAN	BTAN S D	• Tan(S) → (D)+0 Sign +1 Integer part +2 Decimal fraction part		3	—	Page 656
	BTANP	BTANP S D					
	BASIN	BASIN S D	• $Sin^{-1}(S)$ → (D)+0 Sign +1 Integer part +2 Decimal fraction part		3	—	Page 658
	BASINP	BASINP S D					
	BACOS	BACOS S D	• $Cos^{-1}(S)$ → (D)+0 Sign +1 Integer part +2 Decimal fraction part		3	—	Page 660
	BACOSP	BACOSP S D					
	BATAN	BATAN S D	• $Tan^{-1}(S)$ → (D)+0 Sign +1 Integer part +2 Decimal fraction part		3	—	Page 662
	BATANP	BATANP S D					

Data control instructions

Category	Instruction symbol	Symbol	Processing details	Execution condition	Number of basic steps	Subset	Reference
Upper and lower limit controls	LIMIT		• When (S3) < (S1), a value of (S1) is stored at (D). • When (S1) ≤ (S3) ≤ (S2), a value of (S3) is stored at (D). • When (S2) < (S3), a value of (S2) is stored at (D).		5	—	Page 664
	LIMITP						
	DLIMIT		• When (S3+1, S3) < (S1+1, S1), values of (S1+1, S1) are stored at ((D)+1, (D)). • When (S1+1, S1) ≤ (S3+1, S3) ≤ (S2+1, S2), values of (S3+1, S3) are stored at ((D)+1, (D)). • When (S2, S2+1) ≤ (S3, S3+1), values of (S2+1, S2) are stored at ((D)+1, (D)).		5	—	
	DLIMITP						
Dead band controls	BAND		• 0 → (D) when (S1) ≤ (S3) ≤ (S2). • (S3)-(S1) → (D) when (S3) < (S1). • (S3)-(S2) → (D) when (S2) < (S3).		5	—	Page 667
	BANDP						
	DBAND		• 0 → ((D)+1, (D)) when (S1+1, S1) ≤ (S3+1, S3) ≤ (S2+1, S2). • (S3+1, S3)-(S1+1, S1) → ((D)+1, (D)) when (S3+1, S3) < (S1+1, S1). • (S3+1, S3)-(S2+1, S2) → ((D)+1, (D)) when (S2+1, S2) < (S3+1, S3).		5	—	
	DBANDP						
Zone controls	ZONE		• 0 → (D) when (S3) = 0. • (S3)+(S2) → (D) when (S3) > 0. • (S3)-(S1) → (D) when (S3) < 0.		5	—	Page 670
	ZONEP						
	DZONE		• 0 → ((D)+1, (D)) when (S3+1, S3) = 0. • (S3+1, S3)+(S2+1, S2) → ((D)+1, (D)) when (S3+1, S3) > 0. • (S3+1, S3)+(S1+1, S1) → ((D)+1, (D)) when (S3+1, S3) < 0.		5	—	
	DZONEP						
Point-by-point coordinate data	SCL		• Executes scaling for the scaling conversion data (16-bit data units) specified by (S2) with the input value specified by (S1), and then stores the result into the device specified by (D). The scaling conversion is executed based on the scaling conversion data stored in the device specified by (S2) and up.		4	—	Page 672
	SCLP						
	DSCL		• Executes scaling for the scaling conversion data (32-bit data units) specified by (S2) with the input value specified by (S1), and then stores the result into the device specified by (D). The scaling conversion is executed based on the scaling conversion data stored in the device specified by (S2) and up.		4	—	
	DSCLP						

Category	Instruction symbol	Symbol	Processing details	Execution condition	Number of basic steps	Subset	Reference
X or Y coordinate data	SCL2	SCL2 S1 S2 D	• Executes scaling for the scaling conversion data (16-bit data units) specified by (S2) with the input value specified by (S1), and then stores the result into the device specified by (D). The scaling conversion is executed based on the scaling conversion data stored in the device specified by (S2) and up.		4	—	Page 676
	SCL2P	SCL2P S1 S2 D					
	DSCL2	DSCL2 S1 S2 D	• Executes scaling for the scaling conversion data (32-bit data units) specified by (S2) with the input value specified by (S1), and then stores the result into the device specified by (D). The scaling conversion is executed based on the scaling conversion data stored in the device specified by (S2) and up.		4	—	
	DSCL2P	DSCL2P S1 S2 D					

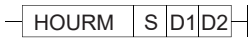
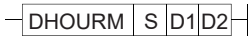
Switching instructions

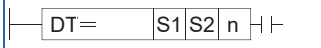
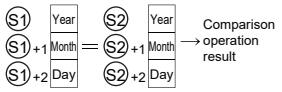
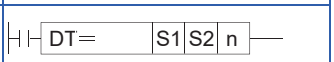
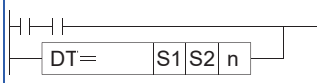
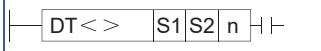
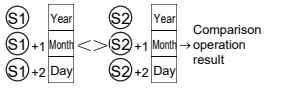
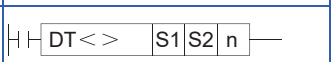
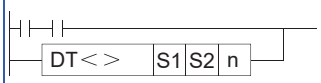
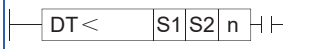
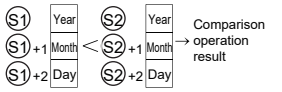
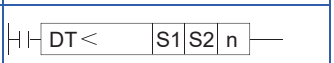
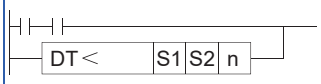
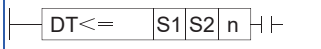
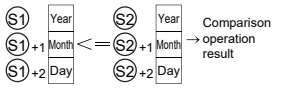
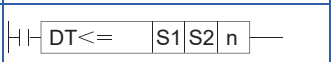
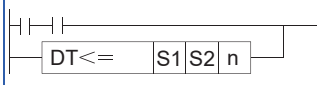
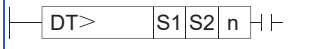
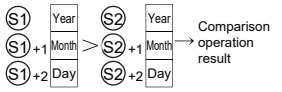
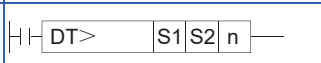
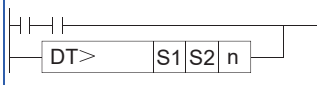
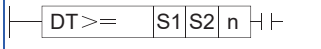
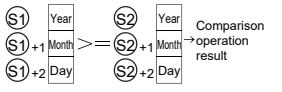
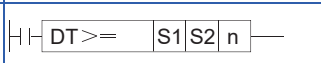
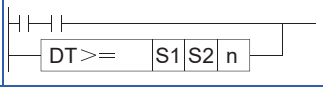
Category	Instruction symbol	Symbol	Processing details	Execution condition	Number of basic steps	Subset	Reference
Block number switching	RSET	RSET S	• Converts extension file register block number to number designated by (S).		2	—	Page 679
	RSETP	RSETP S					
File set	QDRSET	QDRSET File name	• Sets file names used as file registers.		2+n*1	—	Page 681
	QDRSETP	QDRSETP File name					
	QCDSET	QCDSET File name	• Sets file names used as comment files.		2+n*1	—	Page 683
	QCDSETP	QCDSETP File name					

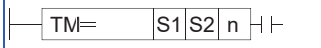
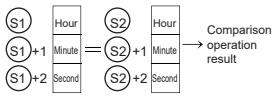
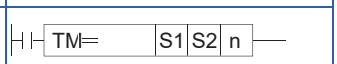
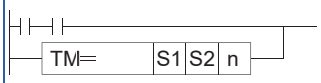
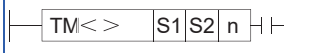
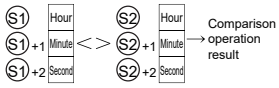
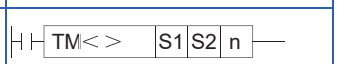
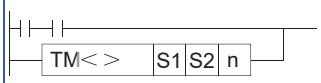
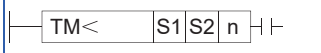
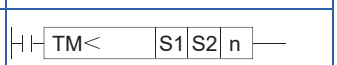
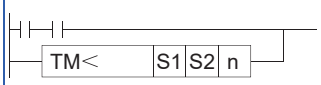
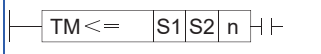
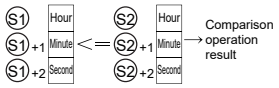
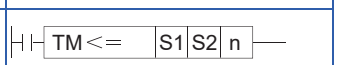
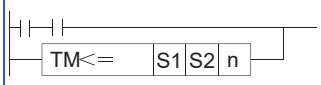
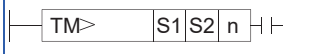
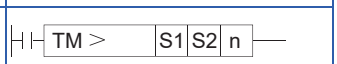
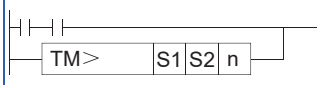
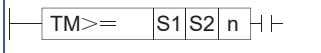
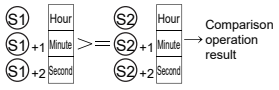
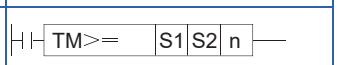
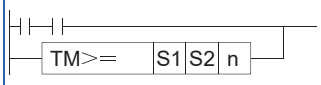
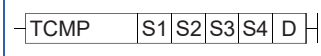
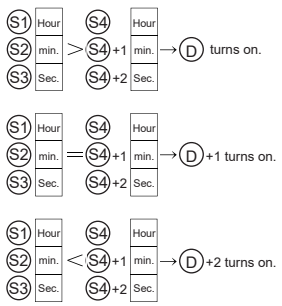
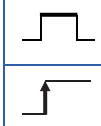
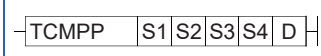
*1 n indicates ([number of file name characters] ÷ 2) steps. (Decimal fraction is rounded up.)

Clock instructions

Category	Instruction symbol	Symbol	Processing details	Execution condition	Number of basic steps	Subset	Reference
Reading/ writing clock data	DATERD	DATERD D	• (Clock elements) → (D) +0 Year, +1 Month, +2 Day, +3 Hour, +4 Minute, +5 Sec., +6 Day of the week		2	—	Page 685
	DATERDP	DATERDP D					
	DATEWR	DATEWR S	• (D) +0 Year, +1 Month, +2 Day, +3 Hour, +4 Minute, +5 Sec., +6 Day of the week → (Clock elements)		2	—	Page 687
	DATEWRP	DATEWRP S					
Clock data addition/ subtraction operation	DATE+	DATE+ S1 S2 D	(S1) Hour Minute Sec. + (S2) Hour Minute Sec. → (D) Hour Minute Sec.		4	—	Page 689
	DATE+P	DATE+P S1 S2 D					
	DATE-	DATE- S1 S2 D	(S1) Hour Minute Sec. − (S2) Hour Minute Sec. → (D) Hour Minute Sec.		4	—	Page 691
	DATE-P	DATE-P S1 S2 D					
Time data conversion	SECOND	SECOND S D	(S) Hour Minute Sec. → (D) Sec. (Lower 16 bits) Sec. (Upper 16 bits)		3	—	Page 693
	SECONDP	SECONDP S D					
	HOUR	HOUR S D	(S) Sec. (Lower 16 bits) Sec. (Upper 16 bits) → (D) Hour Minute Sec.		3	—	Page 695
	HOURP	HOURP S D					
Date and time data conversion	DATE2SEC	DATE2SEC S D	(S) Year Month Day Hour Minute Sec. → (D) Sec. (Lower 16 bits) Sec. (Upper 16 bits)		3	—	Page 697
	DATE2SECP	DATE2SECP S D					
	SEC2DATE	SEC2DATE S D	(S) Sec. (Lower 16 bits) Sec. (Upper 16 bits) → (D) Year Month Day Hour Minute Sec. Day of week		3	—	Page 699
	SEC2DATEP	SEC2DATEP S D					

Category	Instruction symbol	Symbol	Processing details	Execution condition	Number of basic steps	Subset	Reference
Hour meter	HOURM		• While the execution specification is on, the current value is stored in increments of hour in the device designated by (D1) and the current value less than an hour is stored in increments of second in the device designated by (D1)+1. • The device designated by (D2) turns on when the cumulative ON time of the execution command exceeds the time designated by (S).		4	—	Page 701
	DHOURM		• While the execution specification is on, the current value is stored in increments of hour in the devices designated by (D1), (D1)+1 and the current value less than an hour is stored in increments of second in the device designated by (D1)+2. • The device designated by (D2) turns on when the cumulative ON time of the execution command exceeds the time designated by (S).		4	—	Page 702

Category	Instruction symbol	Symbol	Processing details	Execution condition	Number of basic steps	Subset	Reference
Date comparison	LDDT=				4	—	Page 703
	ANDDT=						
	ORDT=						
	LDDT<>				4	—	
	ANDDT<>						
	ORDT<>						
	LDDT<				4	—	
	ANDDT<						
	ORDT<						
	LDDT<=				4	—	
	ANDDT<=						
	ORDT<=						
	LDDT>				4	—	
	ANDDT>						
	ORDT>						
	LDDT>=				4	—	
	ANDDT>=						
	ORDT>=						

Category	Instruction symbol	Symbol	Processing details	Execution condition	Number of basic steps	Subset	Reference
Time comparison	LDTM=				4	—	Page 707
	ANDTM=						
	ORTM=						
	LDTM<>				4	—	
	ANDTM<>						
	ORTM<>						
	LDTM<				4	—	
	ANDTM<						
	ORTM<						
	LDTM<=				4	—	
	ANDTM<=						
	ORTM<=						
	LDTM>				4	—	
	ANDTM>						
	ORTM>						
	LDTM>=				4	—	
	ANDTM>=						
	ORTM>=						
Clock data comparison	TCMP				6	—	Page 711
	TCMPP						

Category	Instruction symbol	Symbol	Processing details	Execution condition	Number of basic steps	Subset	Reference
Clock data band comparison	TZCP	TZCP S1 S2 S3 D	(S1) Hour, (S1)+1 min., (S1)+2 Sec. > (S3) Hour, (S3)+1 min., (S3)+2 Sec. → (D) turns on. (S1) Hour, (S1)+1 min., (S1)+2 Sec. ≦ (S3) Hour, (S3)+1 min., (S3)+2 Sec. ≦ (S2) Hour, (S2)+1 min., (S2)+2 Sec. → (D)+1 turns on. (S3) Hour, (S3)+1 min., (S3)+2 Sec. < (S2) Hour, (S2)+1 min., (S2)+2 Sec. → (D)+2 turns on.		5	—	Page 713
	TZCPP	TZCPP S1 S2 S3 D					

Expansion clock instructions

Category	Instruction symbol	Symbol	Processing details	Execution condition	Number of basic steps	Subset	Reference
Reading expansion clock data	S.DATERD	S.DATERD D	·(Clock elements)→(D)+0 Year, +1 Month, +2 Day, +3 Hour, +4 Minute, +5 Sec., +6 Day of the week, +7 1/1000 sec.		6	—	Page 715
	SP.DATERD	SP.DATERD D					
Expansion clock data addition/ subtraction operation	S.DATE+	S.DATE+ S1 S2 D	(S1) Hour, Minute, Sec., —, 1/1000 sec. + (S2) Hour, Minute, Sec., —, 1/1000 sec. → (D) Hour, Minute, Sec., —, 1/1000 sec.		8	—	Page 718
	SP.DATE+	SP.DATE+ S1 S2 D					
	S.DATE-	S.DATE- S1 S2 D	(S1) Hour, Minute, Sec., —, 1/1000 sec. − (S2) Hour, Minute, Sec., —, 1/1000 sec. → (D) Hour, Minute, Sec., —, 1/1000 sec.		8	—	
	SP.DATE-	SP.DATE- S1 S2 D					

Program control instructions

Category	Instruction symbol	Symbol	Processing details	Execution condition	Number of basic steps	Subset	Reference
Program control instructions	PSTOP	PSTOP File name	• Places designated program in standby status.		2+n*1	—	Page 725
	PSTOPP	PSTOPP File name					
	POFF	POFF File name	• Turns OUT instruction coil of designated program OFF, and places program in standby status.		2+n*1	—	Page 726
	POFFP	POFFP File name					
	PSCAN	PSCAN File name	• Registers designated program as scan execution type.		2+n*1	—	Page 728
	PSCANP	PSCANP File name					
	PLOW	PLOW File name	• Registers designated program as low-speed execution type.		2+n*1	—	Page 730
	PLOWP	PLOWP File name					
	LDPCHK	PCHK File name	• In conduction when program of specified file name is being executed. • In non-conduction when program of specified file name is not executed.		2+n*1	—	Page 731
	ANDPCHK	PCHK File name					
	ORPCHK	PCHK File name					

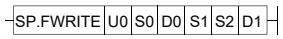
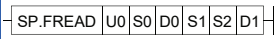
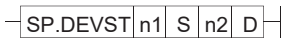
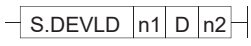
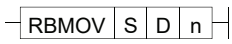
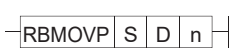
*1 n indicates ([number of file name characters] ÷ 2) steps. (Decimal fraction is rounded up.)

PID instruction

Category	Instruction symbol	Symbol	Processing details	Execution condition	Number of basic steps	Subset	Reference
PID operation	PID	PID S1 S2 S3 D	• Performs PID operation based on the value set in the devices designated by (S1), (S2), and (S3) and stores the operation result every sampling time in the device designated by (D).		5	—	Page 735

Other instructions

Category	Instruction symbol	Symbol	Processing details	Execution condition	Number of basic steps	Subset	Reference
WDT reset	WDT	WDT	• Resets watchdog timer during sequence program.		1	—	Page 758
	WDTP	WDTP					
Timing clock	DUTY	DUTY n1 n2 D	(D) n1 scans n2 scans SM420 to SM424, SM430 to SM434		4	—	Page 760
Time check	TIMCHK	TIMCHK S1 S2 D	• Turns ON device specified by (D) if measured ON time of input condition is longer than preset time continuously.		4	—	Page 762
Direct read/ write operations in 1-byte units	ZRRDB	ZRRDB n D	0 Lower 8 bits ZR0 1 Upper 8 bits 2 Lower 8 bits ZR1 3 Upper 8 bits n 8 bits → (D)		3	—	Page 763
	ZRRDBP	ZRRDBP n D					
	ZRWRB	ZRWRB n S	(S) 0 Lower 8 bits ZR0 1 Upper 8 bits 2 Lower 8 bits ZR1 3 Upper 8 bits n 8 bits		3	—	Page 765
	ZRWRBP	ZRWRBP n S					
	ADRSET	ADRSET S D	(S) → (D) Indirect address of designated device Device name		3	—	Page 767
	ADRSETP	ADRSETP S D					
Numerical key input from keyboard	KEY	KEY S n D1 D2	• Takes in ASCII data for 8 points of input unit designated by (S), converts to hexadecimal value following device number designated by (D1), and stores.		5	—	Page 768
Batch save of index register	ZPUSH	ZPUSH D	• Saves the contents of index registers to a location starting from the device designated by (D).		2	—	Page 772
	ZPUSHP	ZPUSHP D					
Batch recovery of index register	ZPOP	ZPOP D	• Reads the data stored in the location starting from the device designated by (D) to index registers.				
	ZPOPP	ZPOPP D					
Reading module information	UNIRD	UNIRD n1 D n2	• Reads the module information stored in the area starting from the I/O No. designated by (n1) by the points designated by (n2), and stores it in the area starting from the device designated by (D).		4	—	Page 775
	UNIRDP	UNIRDP n1 D n2					
Module model name read	TYPERD	TYPERD n D	• Reads the module model name of the head I/O No. designated by (n) and stores it in the area starting from the device designated by (D).		3	—	Page 779
	TYPERDP	TYPERDP n D					
Trace set	TRACE	TRACE	• Stores the trace data set with peripheral device by the number of times set when SM800, SM801 and SM802 turn on, to the sampling trace file.		1	—	Page 783
Trace reset	TRACER	TRACER	• Resets the data set the TRACE instruction.		1	—	

Category	Instruction symbol	Symbol	Processing details	Execution condition	Number of basic steps	Subset	Reference
Writing data to the designated file	SP.FWRITE		• Writes data to the designated file.		11	—	Page 785
Reading data from designated file	SP.FREAD		• Reads data from the designated file.		11	—	Page 795
Writing data to standard ROM	SP.DEVST		• Writes data to the device data storage file in the standard ROM.		9	—	Page 807
Reading data from standard ROM	S.DEVLD		• Reads data from the device data storage file in the standard ROM.		8	—	Page 809
	SP.DEVLD						
Loading program from memory	PLOADP		• Transfers the program stored in a memory card or standard memory (other than drive 0) to drive 0 and places the program in standby status.		3	—	Page 811
Unloading program from program memory	PUNLOADP		• Deletes the standby program stored in standard memory (drive 0).		3	—	Page 814
Loading + Unload	PSWAPP		• Deletes standby program stored in standard memory (drive 0) designated by (S1). Then, transfers the program stored in a memory card or standard memory (other than drive 0) designated by (S2) to drive 0 and places it in standby status.		4	—	Page 816
High-speed block transfer of file register	RBMOV		• Transfers n points of 16-bit data from the device designated by (S) to the devices of n points starting from the one designated by (D).		4	—	Page 818
	RBMOVP						
User message	UMSG		• Displays the specified character strings on the display unit as a user message.		2	—	Page 823

2.6 Instructions for Data Link

Instructions for network refresh

Category	Instruction symbol	Symbol	Processing details	Execution condition	Number of basic steps	Subset	Reference
Link instruction: Network refresh	S.ZCOM	S.ZCOM Jn	• Refreshes the designated network.		5	—	Page 827
	SP.ZCOM	SP.ZCOM Jn					
	S.ZCOM	S.ZCOM Un					
	SP.ZCOM	SP.ZCOM Un					

Instructions for reading/writing routing information

Category	Instruction symbol	Symbol	Processing details	Execution condition	Number of basic steps	Subset	Reference
Reading routing information	S.RTREAD	S.RTREAD n D	• Reads data set at routing parameters.		7	—	Page 832
	SP.RTREAD	SP.RTREAD n D					
Registering routing information	S.RTWRITE	S.RTWRITE n S	• Writes routing information to the area designated by routing parameters.		8	—	Page 834
	SP.RTWRITE	SP.RTWRITE n S					

Refresh device write/read instruction

Category	Instruction symbol	Symbol	Processing details	Execution condition	Number of basic steps	Subset	Reference
Refresh device write instruction	S.REFDVWRB	S.REFDVWRB n1 S1 S2 n2 D1	• Writes data in 1-bit units to the specified refresh device.		11	—	Page 836
	SP.REFDVWRB	SP.REFDVWRB n1 S1 S2 n2 D1					
	S.REFDVWRW	S.REFDVWRW n1 S1 S2 n2 D1	• Writes data in 16-bit units to the specified refresh device.		11	—	Page 840
	SP.REFDVWRW	SP.REFDVWRW n1 S1 S2 n2 D1					
Refresh device read instruction	S.REFDVRDB	S.REFDVRDB n1 S1 D1 n2 D2	• Reads data in 1-bit units from the specified refresh device.		11	—	Page 845
	SP.REFDVRDB	SP.REFDVRDB n1 S1 D1 n2 D2					
	S.REFDVRDW	S.REFDVRDW n1 S1 D1 n2 D2	• Reads data in 16-bit units from the specified refresh device.		11	—	Page 849
	SP.REFDVRDW	SP.REFDVRDW n1 S1 D1 n2 D2					

2.7 Multiple CPU Dedicated Instruction

Instructions for writing to the CPU shared memory of host CPU

Category	Instruction symbol	Symbol	Processing details	Execution condition	Number of basic steps	Subset	Reference
Write to host CPU shared memory	S.TO	S.TO n1 n2 n3 n4 D	• Writes device data of the host station to the host CPU shared memory.		5	—	Page 855
	SP.TO	SP.TO n1 n2 n3 n4 D					
	TO	TO n1 n2 S n3	• Writes device data of the host station to the host CPU shared memory.		5	—	Page 858
	TOP	TOP n1 n2 S n3					
	DTO	DTO n1 n2 S n3	• Writes device data of the host station to the host CPU shared memory in 32-bit units.		5	—	
	DTOP	DTOP n1 n2 S n3					

Instructions for reading from the CPU shared memory of another CPU

Category	Instruction symbol	Symbol	Processing details	Execution condition	Number of basic steps	Subset	Reference
Read from other CPU shared memory	FROM	FROM n1 n2 D n3	• Reads device data from the other CPU shared memories, and stores the data in the host station.		5	—	Page 863
	FROMP	FROMP n1 n2 D n3					
	DFRO	DFRO n1 n2 D n3	• Reads device data from the other CPU shared memories in 32-bit units, and stores the data in the host station.		5	—	
	DFROP	DFROP n1 n2 D n3					

2.8 Multiple CPU High-speed Transmission Dedicated Instruction

Instructions for multiple CPU high-speed transmission

Category	Instruction symbol	Symbol	Processing details	Execution condition	Number of basic steps	Subset	Reference
Writing devices to another CPU	D.DDWR	D.DDWR n S1 S2 D1 D2	• In multiple CPU system, data stored in a device specified by host CPU ((S2)) or later is stored by the number of write points specified by ((S1)+1) into a device specified by another CPU (n) ((D1)) or later		10	—	Page 879
	DP.DDWR	DP.DDWR n S1 S2 D1 D2			10	—	
Reading devices from another CPU	D.DDRD	D.DDRD n S1 S2 D1 D2	• In multiple CPU system, data stored in a device specified by another CPU (n) ((D1)) or later is stored by the number of read points specified by ((S1)+1) into a device specified by host CPU ((S2)) or late		10	—	Page 883
	DP.DDRD	DP.DDRD n S1 S2 D1 D2			10	—	

2.9 Redundant System Instructions (For Redundant CPU)

Instructions for redundant system (for Redundant CPU)

Category	Instruction symbol	Symbol	Processing details	Execution condition	Number of basic steps	Subset	Reference
System switching	SP.CONTSW	SP.CONTSW S D	• Switches between the control system and standby system at the END processing of the scan executed with the SP.CONTSW instruction.		8	—	Page 887

3 CONFIGURATION OF INSTRUCTIONS

3.1 Configuration of Instructions

Most CPU module instructions consist of an instruction part and a device part.

Each part is used for the following purpose:

- Instruction part: indicates the function of the instruction.
- Device part: indicates the data that is to be used with the instruction.

The device part consists of source data, destination data, and number of devices.

Source (S)

Source is the data used for operations.

The following source types are available, depending on the designated device:

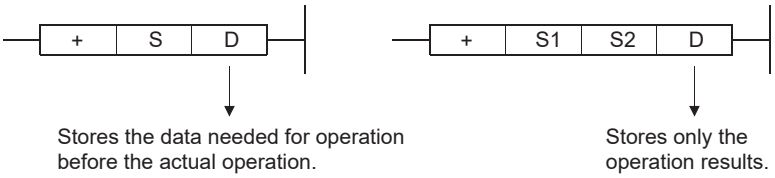
Type	Description
Constant	Designates a numeric value to be used in the operation. This is set when the program is created, and cannot be changed during the execution of the program. Constants should be indexed when used as variable data.
Bit devices and word devices	Designates the device that stores the data to be used in the operation. Data must be stored in the designated device until the operation is executed. By changing the data stored in a designated device during program execution, the data to be used in the instruction can be changed.

Destination (D)

The destination stores the data after the operation has been conducted. However, some instructions require storing the data to be used in an operation at the destination prior to the operation execution.

Ex.

An addition instruction involving BIN 16-bit data



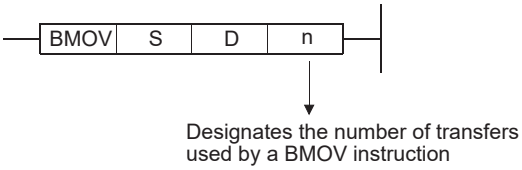
A device for the data storage must always be set to the destination.

Number of devices and number of transfers (n)

The number of devices and number of transfers designate the numbers of devices and transfers used by instructions involving multiple devices.

Ex.

Block transfer instruction

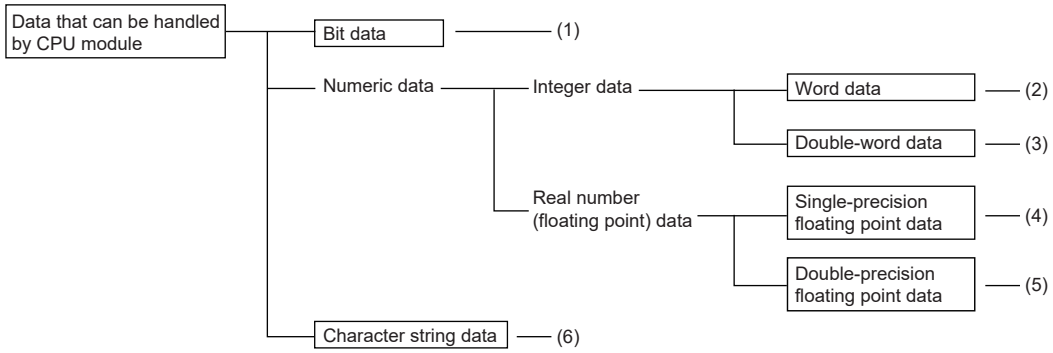


The number of devices or number of transfers can be set between 0 and 32767.

However, if the number is 0, the instruction will be a no-operation instruction.

3.2 Designating Data

The following six types of data can be used with CPU module instructions.



(1) ☞ Page 83 Using bit data
(2) ☞ Page 84 Using word (16 bits) data
(3) ☞ Page 86 Using double word (32 bits) data
(4) ☞ Page 88 Single-precision real number data (single-precision floating-point data)
(5) ☞ Page 89 Double-precision real number data (double-precision floating-point data)
(6) ☞ Page 92 Using character string data

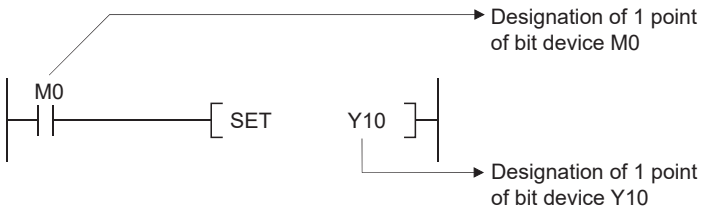
Using bit data

Bit data is data used in one-bit units, such as for contacts or coils.

"Bit devices" and "Bit designated word devices" can be used as bit data.

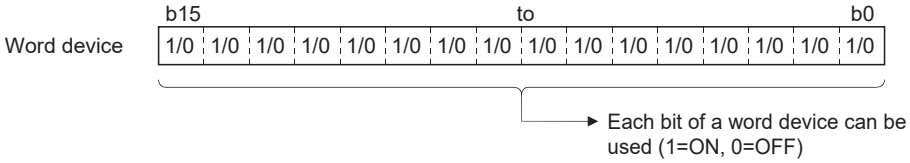
When using bit devices

Bit devices are designated in one-point units.



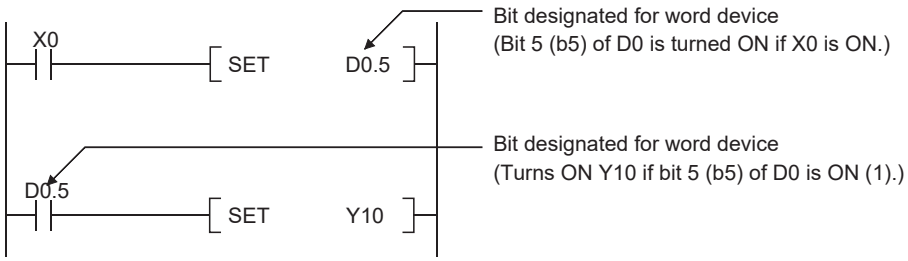
Using word devices

Word devices enable the use of a designated bit number 1/0 as bit data by the designation of that bit number.



Word device bit designation is done by designating "[Word device].[Bit No.]". (Designation of bit numbers is done in hexadecimal.)

For example, bit 5 (b5) of D0 is designated as D0.5, and bit 10 (b10) of D0 is designated as D0.A. However, there can be no bit designation for timers (T), retentive timers (ST), counters (C) or index register (Z). (Example Z0.0 is not available).



Using word (16 bits) data

Word data is 16-bit numeric data used by basic instructions and application instructions.
The following two types of word data can be used with CPU module:

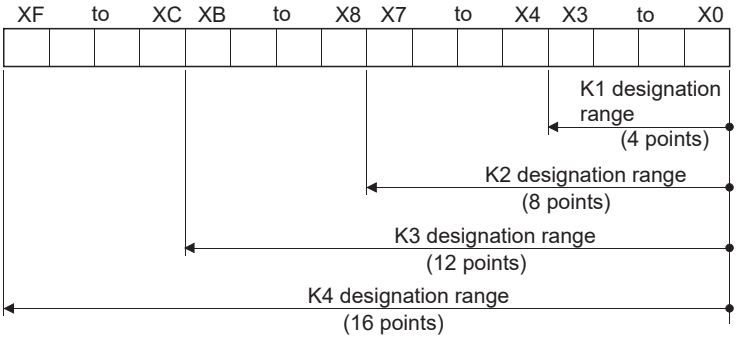
- Decimal constants: K-32768 to K32767
- Hexadecimal constants: H0000 to HFFFF

Word devices and bit devices designated by digit can be used as word data.
For direct access input (DX) and direct access output (DY), word data cannot be designated by digit. (For details of direct access input and direct access output, refer to the User's Manual (Function Explanation, Program Fundamentals) for the CPU module used.)

When Using Bit Devices

Bit devices can deal with word data when digits are designated.
Digit designation of bit devices is done by designating "[Number of digits][Head number of bit device]".
Digit designation of bit devices can be done in 4-point (4-bit) units, and designation can be made for K1 to K4.
(For link direct devices, designation is done by "J[Network No.]\[Number of digits][Head number of bit device]". When X100 to X10F are designated for Network No.2, it is done by J2\K4X100). For example, if X0 is designated for digit designation, the following points would be designated:

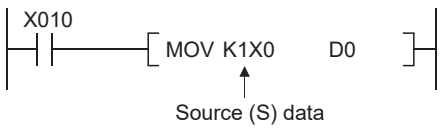
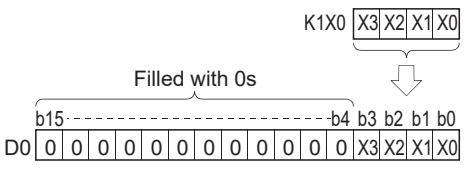
- K1X0 → The 4 points X0 to X3 are designated.
- K2X0 → The 8 points X0 to X7 are designated.
- K3X0 → The 12 points X0 to XB are designated.
- K4X0 → The 16 points X0 to XF are designated.



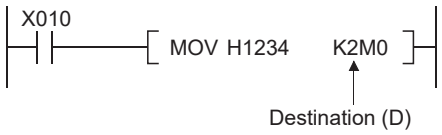
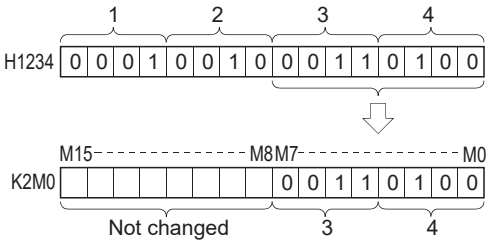
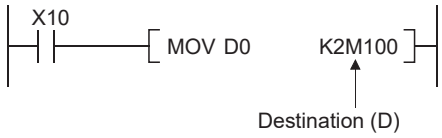
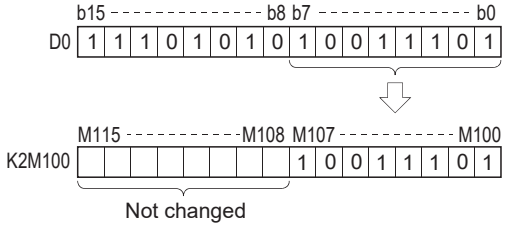
In cases where digit designation has been made at the source (S), the numeric values shown in the following table are those which can be dealt with as source data.

Number of digits designated	With 16-bit instructions
K1 (4 points)	0 to 15
K2 (8 points)	0 to 255
K3 (12 points)	0 to 4095
K4 (16 points)	-32768 to 32767

When destination (D) data is a word device, the word device for the destination becomes 0 following the bit designated by digit designation at the source.

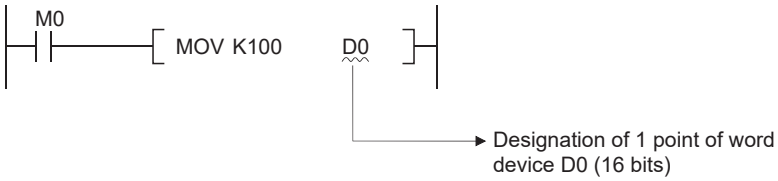
Ladder example	Processing
• With 16-bit instructions 	

In cases where digit designation is made at the destination (D), the number of points designated are used as the destination. Bit devices below the number of points designated as digits do not change.

Ladder example	Processing
• When source (S) data is a numerical value 	
• When source (S) data is a word device 	

Using word devices

Word devices are designated in 1-point (16 bits) units.



Point

- When digit designation processing is conducted, a random value can be used for the bit device initial device number.
- Digit designation cannot be made for the direct access I/O (DX, DY).

Using double word (32 bits) data

Double word data is 32-bit numerical data used by basic instructions and application instructions.
The two types of double word data that can be dealt with by CPU module are as follows:

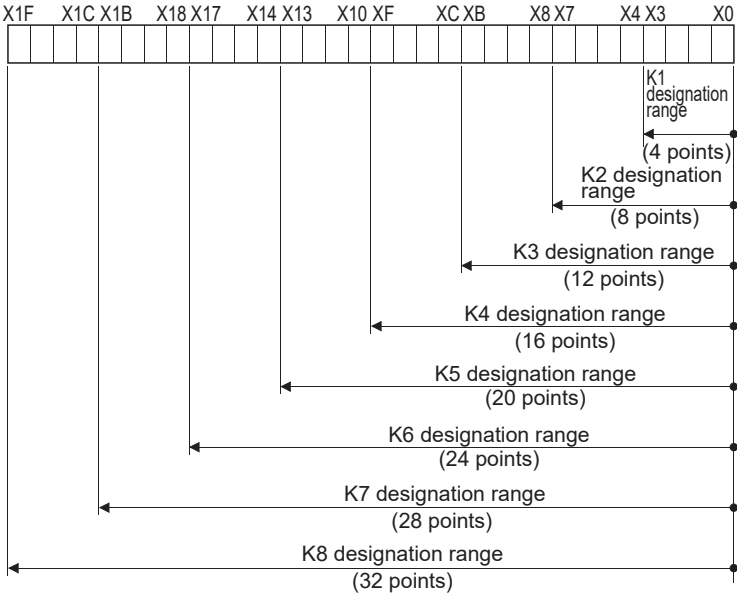
- Decimal constants: K-2147483648 to K2147483647
- Hexadecimal constants: H00000000 to HFFFFFFFF

Word devices and bit devices designated by digit designation can be used as double word data.
For direct access input (DX) and direct access output (DY), designation of double word data is not possible by digit designation.

When Using Bit Devices

Digit designation can be used to enable a bit device to deal with double word data.
Digit designation of bit devices is done by designating "[Number of digits][Head number of bit device]".
(For link direct devices, designation is done by "J[Network No.]\[Number of digits][Head number of bit device]". When X100 to X11F are designated for Network No.2, it is done by J2\K8X100.) Digit designation of bit devices can be done in 4-point (4-bit) units, and designation can be made for K1 to K8.
For example, if X0 is designated for digit designation, the following points would be designated:

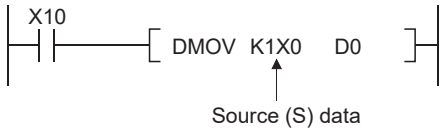
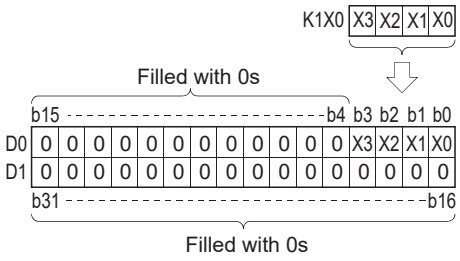
- K1X0 → The 4 points X0 to X3 are designated.
- K2X0 → The 8 points X0 to X7 are designated.
- K3X0 → The 12 points X0 to XB are designated.
- K4X0 → The 16 points X0 to XF are designated.
- K5X0 → The 20 points X0 to X13 are designated.
- K6X0 → The 24 points X0 to X17 are designated.
- K7X0 → The 28 points X0 to X1B are designated.
- K8X0 → The 32 points X0 to X1F are designated.



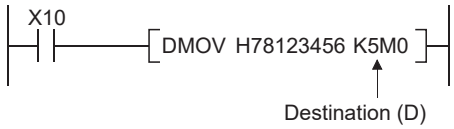
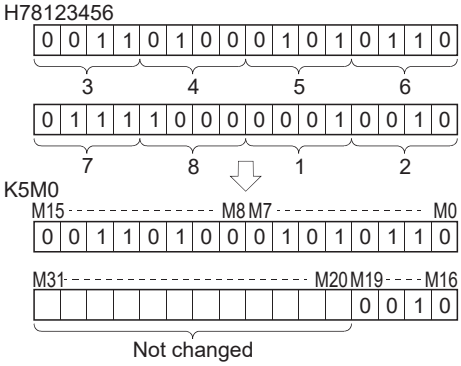
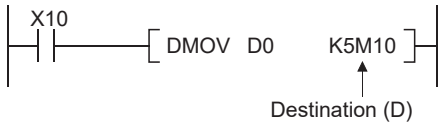
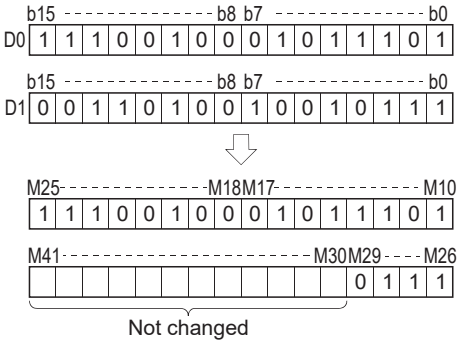
In cases where digit designation has been made at the source (S), the numeric values shown in the following table are those which can be dealt with as source data.

Number of digits designated	With 32-bit instructions	Number of digits designated	With 32-bit instructions
K1 (4 points)	0 to 15	K5 (20 points)	0 to 1048575
K2 (8 points)	0 to 255	K6 (24 points)	0 to 16777215
K3 (12 points)	0 to 4095	K7 (28 points)	0 to 268435455
K4 (16 points)	0 to 65535	K8 (32 points)	-2147483648 to 2147483647

When destination (D) data is a word device, the word device for the destination becomes 0 following the bit designated by digit designation at the source.

Ladder example	Processing
• With 32-bit instructions 	

In cases where digit designation is made at the destination (D), the number of points designated are used as the destination. Bit devices below the number of points designated as digits do not change.

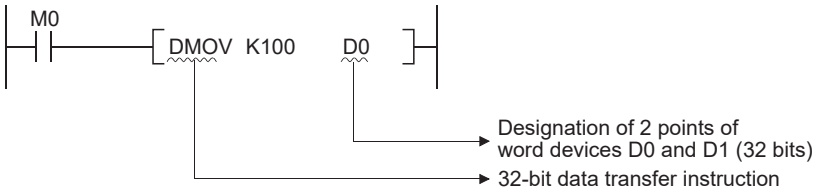
Ladder example	Processing
• When source (S) data is a numerical value 	
• When source (S) data is a word device 	

Point

- When digit designation processing is conducted, a random value can be used for the bit device initial device number.
- Digit designation cannot be made for the direct access I/O (DX, DY).

Using word devices

A word device designates devices used by the lower 16 bits of data. A 32-bit instruction uses (designation device number) and (designation device number + 1).

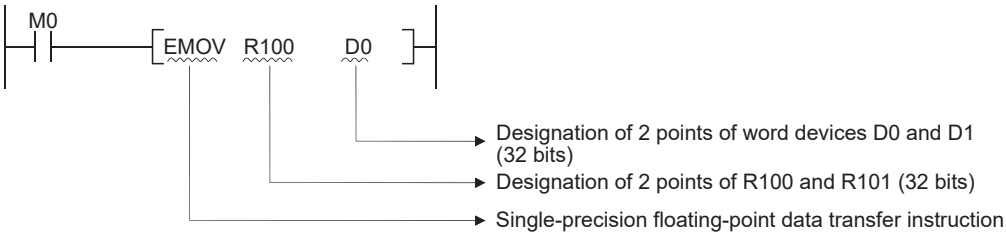


Using single/double-precision real number data

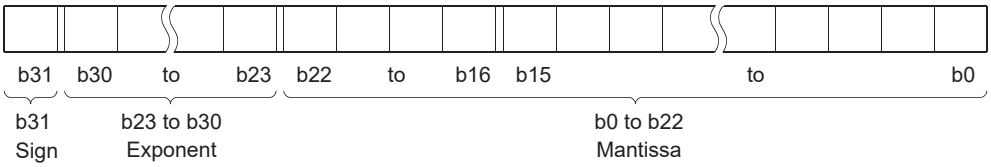
Real number data is floating decimal point data used with basic instructions and application instructions.
Only word devices are capable of storing real number data.

Single-precision real number data (single-precision floating-point data)

Instructions which deal with single-precision floating-point data designate devices which are used for the lower 16 bits of data.
Single-precision floating-point data are stored in the 32 bits which make up (designated device number) and (designated device number + 1).



In a sequence program, floating-point data are designated by E□.
Single-precision floating-point data can be represented as follows, using two word devices.
[Sign] 1.[Mantissa] $\times$ $2^{[Exponent]}$
The bit configuration and meaning for the internal representation of single-precision floating-point data are described below:



- Sign: The most significant bit, b31, is the sign bit.
  - 0: Positive
  - 1: Negative
- Exponent: The 8 bits, b23 to b30, represent the excess n of 2^n. The following shows the excess n according to the binary values in b23 to b30.

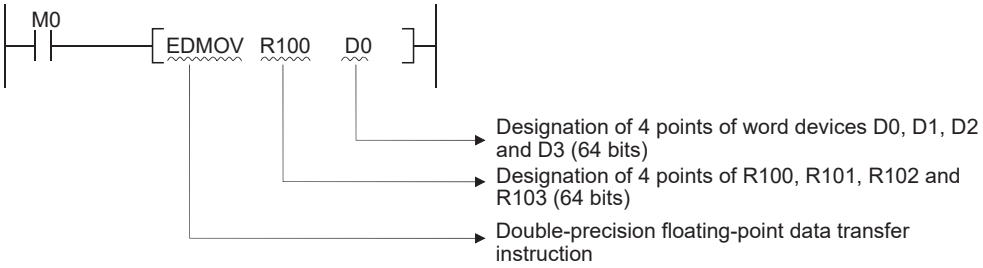
b23 to b30	FF$_H$	FE$_H$	FD$_H$		81	80	7F$_H$	7E$_H$		02	01	00
n	Not used	127	126		2	1	0	-1		-125	-126	Not used

- Mantissa: Each of the 23 bits, b0 to b22, represents the "XXXXXX..." portion when the data is represented in binary, "1.XXXXXX...".

Double-precision real number data (double-precision floating-point data)

Instructions which deal with double-precision floating-point data designate devices which are used for the lower 16 bits of data.

Double-precision floating-point data are stored in the 64 bits which make up (designated device number) to (designated device number + 3).

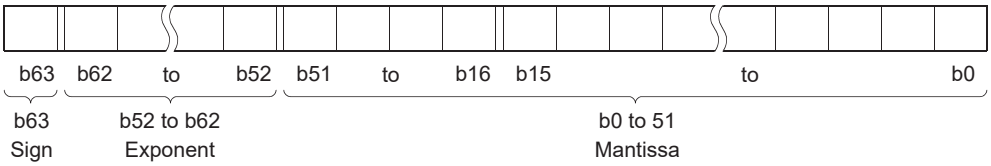


In a sequence program, floating-point data are designated by E□.

Double-precision floating-point data can be represented as follows, using four word devices.

[Sign] 1.[Mantissa] × $2^{[Exponent]}$

The bit configuration and meaning for the internal representation of double-precision floating-point data are described below:



- Sign: The most significant bit, b63, is the sign bit.
  - 0: Positive
  - 1: Negative
- Exponent: The 11 bits, b52 to b62, represent the excess n of 2^n. The following shows the excess n according to the binary values in b52 to b62.

b52 to b62	7FF$_H$	7FE$_H$	7FD$_H$		400$_H$	3FF$_H$	3FE$_H$	3FD$_H$	3FC$_H$		02$_H$	01$_H$	00$_H$
n	Not used	1023	1022		1	0	-1	-2	-3		-1021	-1022	Not used

- Mantissa: Each of the 52 bits, b0 to b51, represents the "XXXXXX..." portion when the data is represented in binary, "1.XXXXXX...".

Precautions

Precautions when an input value of a single/double-precision real number is set using a programming tool are shown below.

■Single-precision real number

Because single-precision real number data are processed as the 32-bit single-precision in a programming tool, the number of significant digits becomes approximately 7. An input value of the single-precision real number data exceeds 7 digits, 8th digit is rounded. If the value after rounding exceeds the range of -2147483648 to 2147483647, an operation error occurs.

Example 1: When '2147483647' is set for the input value

8th digit '6' is rounded.
The value is handled as '2147484000'.

Example 2: When 'E1.1754943562' is set for the input value

8th digit '3' is rounded.
The value is handled as 'E1.175494'.

■Double-precision real number

Because double-precision real number data are processed as the 64-bit double-precision in a programming tool, the number of significant digits becomes approximately 15. An input value of the double-precision real number data exceeds 15 digits, 16th digit is rounded. If the value after rounding exceeds the range of -2147483648 to 2147483647, an operation error occurs.

Example 1: When '2147483646.12345678' is set for the input value

16th digit '6' is rounded.
The value is handled as '2147483646.12346'.

Example 2: When 'E1.7976931348623157+307' is set for the input value

16th digit '5' is rounded.
The value is handled as 'E1.79769313486232+307'.

Point

The CPU module floating decimal point data can be monitored using the monitoring function of a programming tool.

When floating-point data is used to express 0, all data in the following range are turned to 0.

- Single-precision floating-point data: b0 to b31
- Double-precision floating-point data: b0 to b63

The setting range of floating decimal point data is as follows. *1

- Single-precision floating-point data: $-2^{128} < \text{Device data} \leq -2^{-126}$, 0, $2^{-126} \leq \text{Device data} < 2^{128}$
- Double-precision floating-point data: $-2^{1024} < \text{Device data} \leq -2^{-1022}$, 0, $2^{-1022} \leq \text{Device data} < 2^{1024}$

Do not specify -0 in floating-point data (only when the most significant bit of the floating-point real number is 1). (An operation error will occur if floating-point operation is performed with -0.) When -0 is specified, the following CPU module internally converts the value to 0 to perform a floating-point operation. Therefore an operation error does not occur.

- The High Performance model QCPU with the internal processing set to "double precision".*2(Double precision is set by default for the floating-point operation processing.)
- Universal model QCPU (QnUDVCPU and QnUDPVCPU only)

When -0 is specified, the following CPU module performs a floating-point operation with -0, keeping its processing speed. Therefore an operation error occurs.

- Basic model QCPU*3
- High Performance model QCPU where internal operation is set to single precision*2
- Process CPU
- Redundant CPU
- Universal model QCPU (QnUDVCPU and QnUDPVCPU are excluded)
- LCPU

*1 For operations when a real number is out of range and operations when an invalid value is input, refer to the QnUCPU User's Manual (Function Explanation, Program Fundamentals) or Qn(H)/QnPH/QnPRHCPU User's Manual (Function Explanation, Program Fundamentals).

*2 Switch between single precision and double precision of the internal operation of floating-point operation in the PLC system of the PLC parameter dialog box. For the single precision and double precision of floating-point operation, refer to the QnUCPU User's Manual (Function Explanation, Program Fundamentals) or Qn(H)/QnPH/QnPRHCPU User's Manual (Function Explanation, Program Fundamentals).

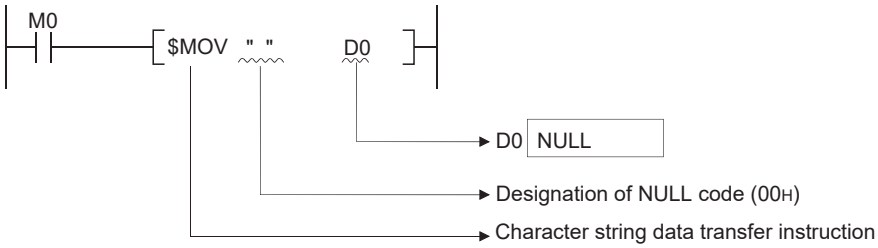
*3 The Basic model QCPU can perform floating-point operation if its first five digits of serial No. are "04122 or later".

Using character string data

Character string data is character data used by basic instructions and application instructions.
The target ranges from the designated character to the NULL code (00H) that indicates the end of the character string.

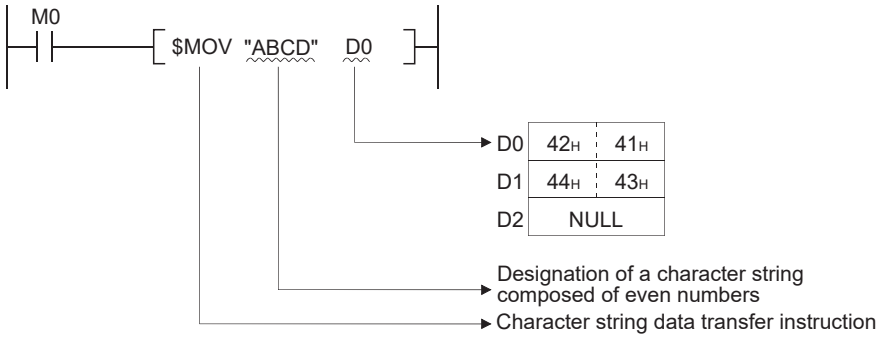
When designated character is the NULL code

One word is used to store the NULL code.



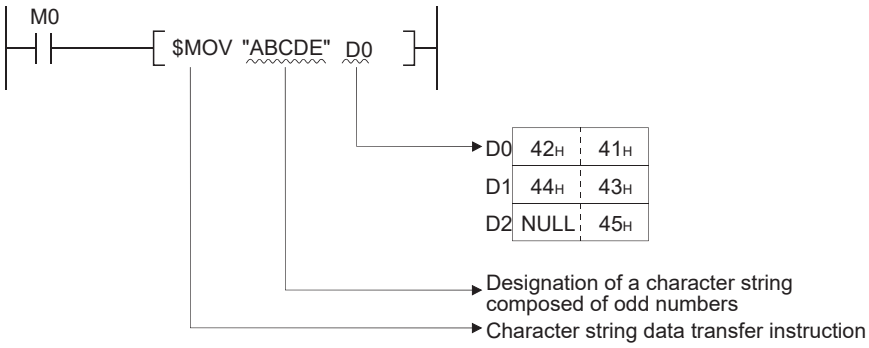
When the number of characters is even

Uses (number of characters/2 + 1) words, and stores character string and NULL code.
For example, if "ABCD" is transferred to D0, the character string ABCD is stored at D0 and D1, and the NULL code is stored at D2. (The NULL code is stored as the last one word.)



When number of characters is odd

Uses (number of characters/2) words (rounds up decimal fractions) and stores the character string and NULL code.
For example, if "ABCDE" is transferred to devices starting from D0, the character string (ABCDE) and the NULL code are stored from D0 to D2. (The NULL code is stored into the upper 8 bits of the last one word.)



3.3 Indexing

Overview of indexing

Indexing is an indirect setting made by using an index register.

When an Indexing is used in a sequence program, the device to be used will become the device number specified directly plus the contents of the index register.

For example, if D2Z2 has been specified, the specified device is calculated as follows: D(2+3) = D5 and the content of Z2 is 3 become the specified device.

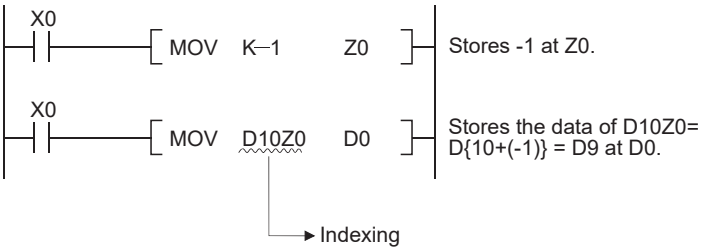
Indexing with 32-bit index registers in addition to 16-bit index registers is available with the Universal model QCPU and LCPU.

Indexing with 16-bit index registers

■Example of indexing

Each index register can be set between -32768 and 32767. *1

Indexing is performed in the way shown below:



*1 For the specifications of index registers, refer to the User's Manual (Function Explanation, Program Fundamentals) for the CPU module used.

■Device that indexing can be used

With the exception of the restrictions noted below, indexing can be used with devices used with contacts, coils, basic instructions, and application instructions.

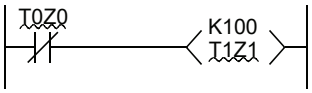
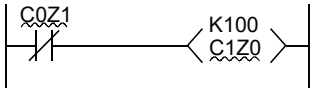
• Devices to which indexing cannot be used

Device	Description
E	Floating decimal point data
$	Character string data
□.□	Word device bit designation
FX, FY, FD	Function devices
P	Pointers used as labels
I	Interrupt pointers used as labels
Z	Index register
S	Step relay*2
TR	SFC transfer devices*1
BL	SFC block device*1*2

*1 SFC transfer devices and SFC block devices are devices for SFC use.
Refer to the manual below for how to use these devices.
MELSEC-Q/L/QnA Programming Manual (SFC)

*2 For the High-speed Universal model QCPU and Universal model Process CPU, the SFC block device (BL) and step relay (S) can be modified using indices within the following range.
• For the SFC block device (BL), the range is BL0 to BL319.
• For the step relay (S), the range is specified in the device settings using parameters.
Note that if a step relay (S) in the SFC block is specified, index modification can be specified within the range of S0 to S511.

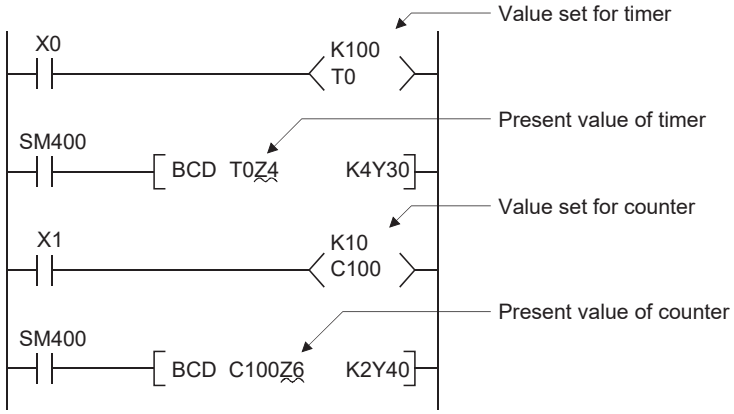
• Devices with limits for use with index registers*3

Device	Description	Application example
T, ST	• Only Z0 and Z1 can be used for timer contacts and coils.	
C	• Only Z0 and Z1 can be used for counter contacts and coils.	

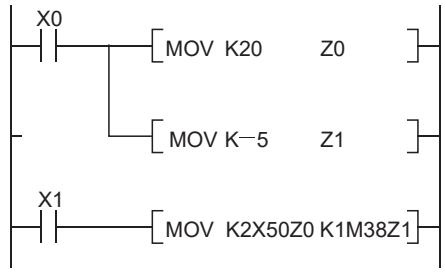
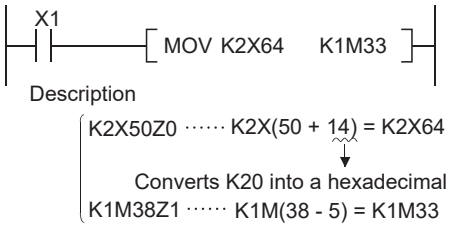
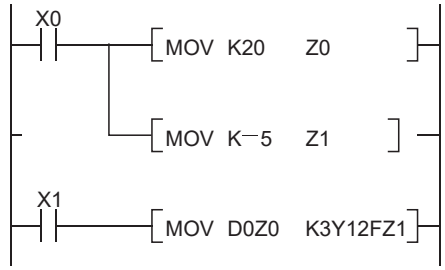
*3 The High-speed Universal model QCPU and Universal model Process CPU are excluded.

Point

For timer and counter present values, there are no limits on index register numbers used.



■A case where indexing has been performed, and the actual process device (Z0 =20, Z1 = -5)

Ladder example	Actual process device
	 Description K2X50Z0 ····· K2X(50 + 14) = K2X64 Converts K20 into a hexadecimal number. K1M38Z1 ····· K1M(38 - 5) = K1M33
	 Description D0Z0 ····· D (0 + 20) = D20 K3Y12FZ1 ····· K3Y(12F - 5) = K3Y12A Hexadecimal number

Indexing with 32-bit index registers*1

A method of specifying index registers in indexing with 32-bit can be selected from the following two methods.

- Specifying the index registers' range used for indexing with 32-bit.
- Specifying the 32-bit indexing using "ZZ" specification.

*1 The methods applies only to Universal model QCPU (excluding Q00UJCPU) and LCPU.

Point

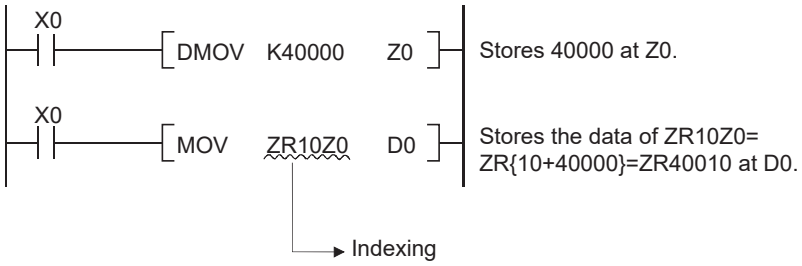
32-bit indexing with the "ZZ" specification is only available for the following CPU modules. See the programming tool operating manual for the available programming tools.

- The first five digits of the serial No. for QnU(D)(H)CPU is "10042" or higher (excluding Q00UJCPU).
- Built-in Ethernet port QCPU
- LCPU

■Example of specifying the range of index registers for use of 32-bit indexing

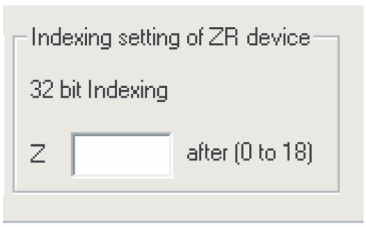
Each index register can be set between -2147483648 and 2147483647.

An example of indexing is shown below.

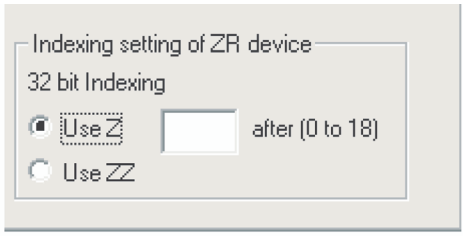


- Specification method

For indexing with a 32-bit index register, specify the head number of an index register to be used on the Device tab of the Q parameter setting screen.



GX Developer 8.68R or earlier



GX Developer 8.68W or later

Point

When the head number of the index register used is changed on the Device tab of the Q parameter setting screen, do not change the parameters only or do not write only the parameters into the programmable controller. Be sure to write the parameter into the programmable controller with the program.

When the parameter is forced to be written into the programmable controller, an error of CAN'T EXE. PRG. occurs. (Error code: 2500)

• Device that indexing can be used

Indexing can be used only for the device shown below.

Device	Description
ZR	Serial number access format file register
D	Extended data register (D)
W	Extended link register (W)

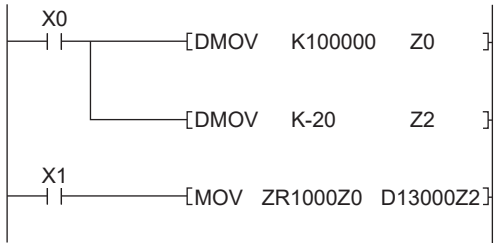
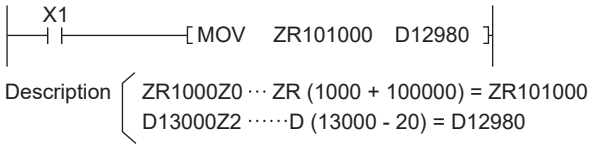
• Usable range of index registers

The following table shows the usable range of index registers for indexing with 32-bit index registers.

For indexing with 32-bit index registers, the specified index register (Zn) and the next index register of the specified register (Zn+1) are used. Be sure not to overlap index registers to be used.

Set value	Index registers to be used	Set value	Index registers to be used
Z0	Z0, Z1	Z10	Z10, Z11
Z1	Z1, Z2	Z11	Z11, Z12
Z2	Z2, Z3	Z12	Z12, Z13
Z3	Z3, Z4	Z13	Z13, Z14
Z4	Z4, Z5	Z14	Z14, Z15
Z5	Z5, Z6	Z15	Z15, Z16
Z6	Z6, Z7	Z16	Z16, Z17
Z7	Z7, Z8	Z17	Z17, Z18
Z8	Z8, Z9	Z18	Z18, Z19
Z9	Z9, Z10	Z19	Unusable

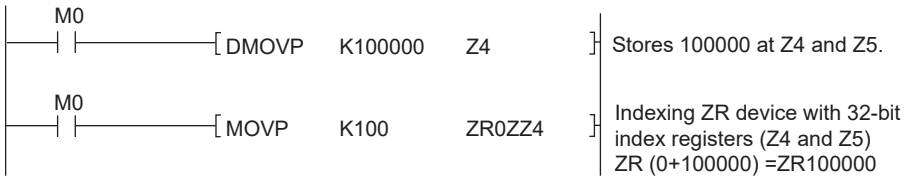
• A case where Indexing has been performed, and the actual process device (Z0 (32-bit) =100000, Z2 (32-bit) = -20)

Ladder example	Actual process device
	

■Example of specifying 32-bit indexing with "ZZ" specification

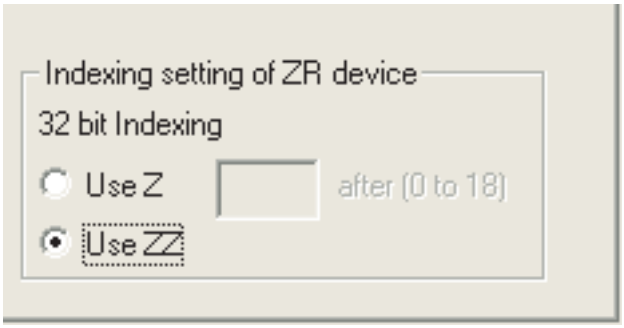
One index register can specify 32-bit indexing by using "ZZ" specification such as "ZR0ZZ4".

The 32-bit indexing with "ZZ" specification is as follows.



• Specification method

To perform 32-bit indexing by using "ZZ" specification, select "Use of ZZ" in "Indexing Setting for ZR Device" in PC parameter.



• Device that indexing can be used

The following device is available for indexing.

Device	Description
ZR	Serial number access format file register
D	Extended data register (D)
W	Extended link register (W)
M*1	Internal relay
B*1	Link relay
D*1	Data register
W*1	Link register
Jn\B*1	Link relay
Jn\W*1	Link register

*1 This device can be used only for the High-speed Universal model QCPU and Universal model Process CPU.

• Usable range of index registers

The following table shows the usable range of index registers in 32-bit indexing used "ZZ" specification.

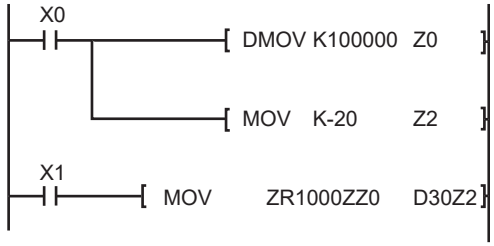
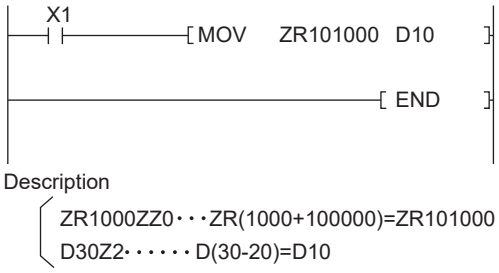
The 32-bit indexing with "ZZ" specification is specified as the format ZRmZZn.

Specifying ZRmZZn enables Zn and Zn+1 of 32-bit values to index the device number, ZRm.

"ZZ" specification*2	Index registers used	"ZZ" specification*2	Index registers used
□ZZ0	Z0, Z1	□ZZ10	Z10, Z11
□ZZ1	Z1, Z2	□ZZ11	Z11, Z12
□ZZ2	Z2, Z3	□ZZ12	Z12, Z13
□ZZ3	Z3, Z4	□ZZ13	Z13, Z14
□ZZ4	Z4, Z5	□ZZ14	Z14, Z15
□ZZ5	Z5, Z6	□ZZ15	Z15, Z16
□ZZ6	Z6, Z7	□ZZ16	Z16, Z17
□ZZ7	Z7, Z8	□ZZ17	Z17, Z18
□ZZ8	Z8, Z9	□ZZ18	Z18, Z19
□ZZ9	Z9, Z10	□ZZ19	Unusable

*2 □ refers to device name (ZR, D, W) for indexing target.

• A case where 32-bit indexing used "ZZ" specification has been performed, and the actual process device (Z0 (32-bit) =100000, Z2 (32-bit) = -20)

Ladder example	Actual process device
	

• Available functions for "ZZ" specification

The 32-bit indexing specification with "ZZ" specification applies in the following functions.

No.	Function name and description
1	Specifying devices in program instruction
2	Monitoring device registrations
3	Testing devices execution type
4	Testing devices with conditions
5	Setting monitor conditions
6	Tracing sampling (Trace point (specifying devices), trace target device)
7	Data logging function (Sampling interval (specifying devices), logging target data)

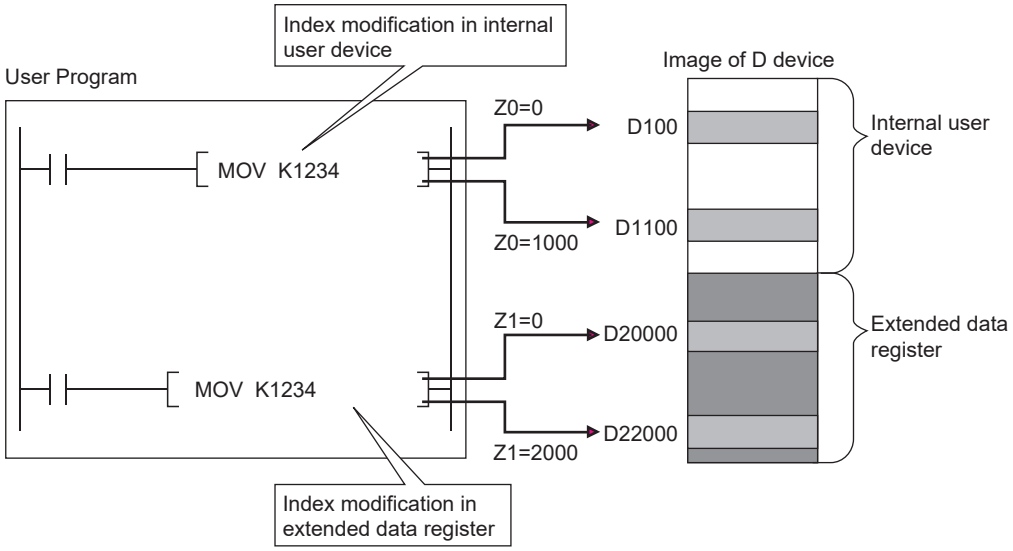
Point

ZZn cannot be used alone as a device like "DMOV K100000 ZZ0". When setting values of index registers to specify 32-bit indexing with "ZZ" specification, set the value of Zn (Z0 to Z19).
ZZn alone cannot be input to each function.

Index modification using extended data register (D) and extended link register (W)*1

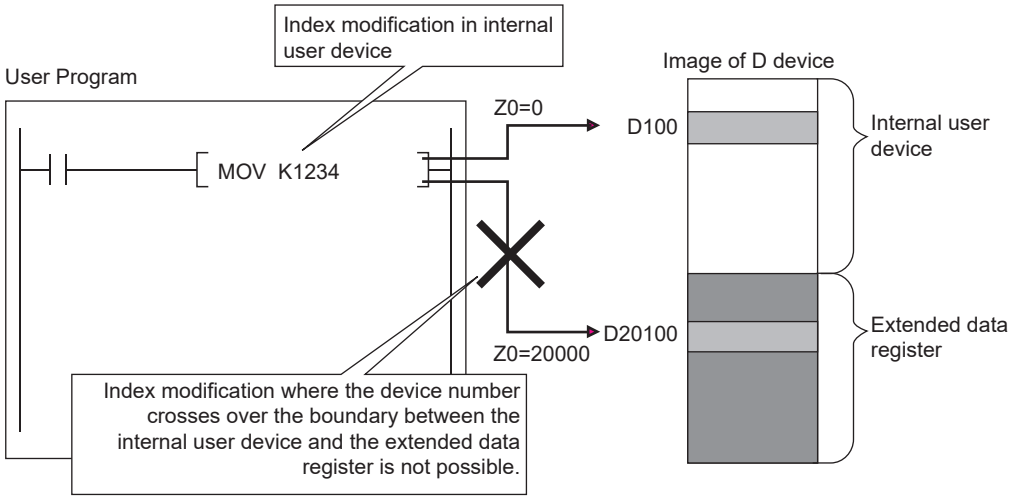
Like index modification using data register (D) and link register (W) of internal user device, a device can be specified by index modification within the range of the extended data register (D) and extended link register (W).

*1 This applies only to Universal model QCPU (excluding Q00UJCPU) and LCPU.



■Index modification where the device number crosses over the boundary between the internal user device and the extended data register (D) or extended link register (W)

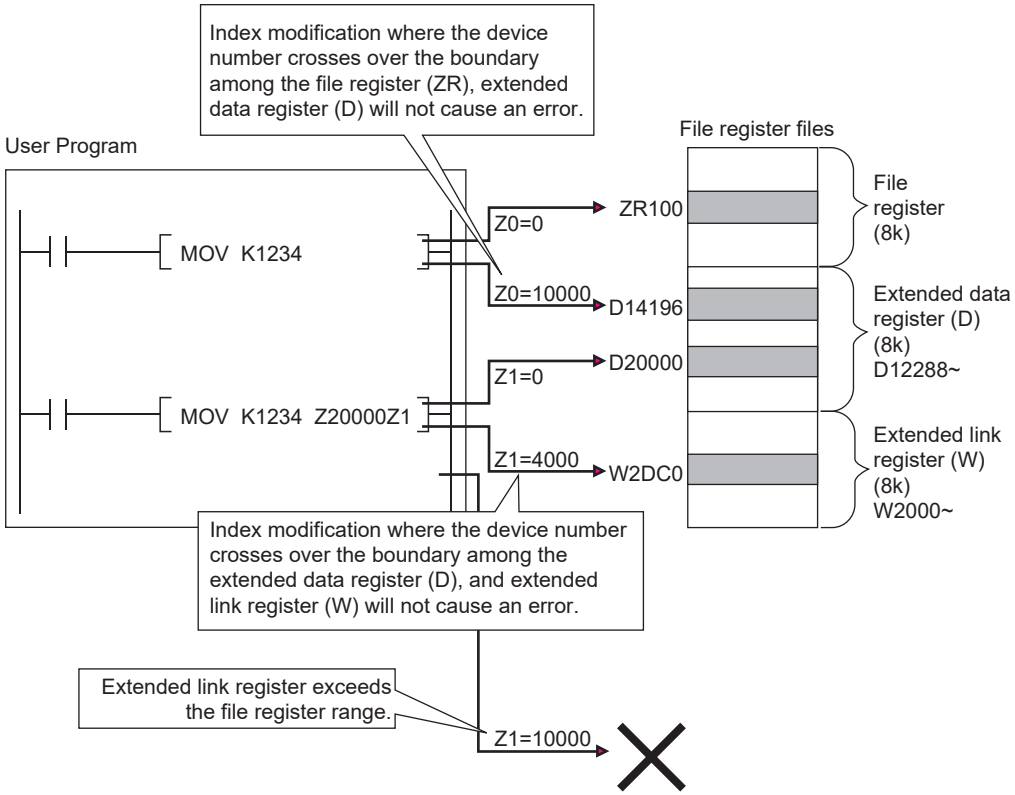
The specification of index modification where the device number crosses over the boundary between the internal user device and the extended data register (D) or extended link register (W) cannot be made. If doing so, an error occurs when the device range check is enabled at index modification (Error code: 4101).



■Index modification where the device number crosses over the boundary among the file register (ZR), extended data register (D), and extended link register (W)

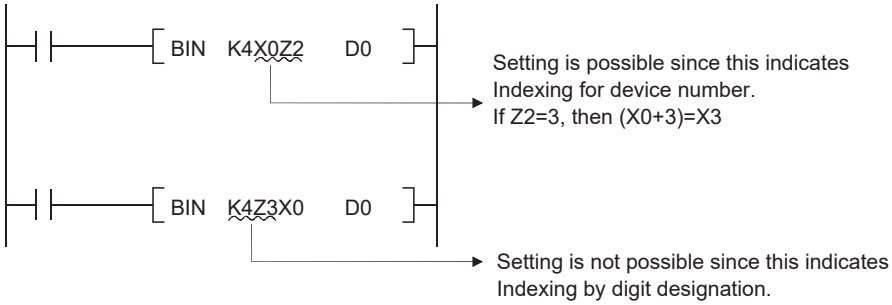
Index modification where the device number crosses over the boundary among the file register (ZR), extended data register (D), and extended link register (W) will not cause an error.

However, an error occurs if the index modification result of file register (ZR), extended data register (D), and extended link register exceeds the file register range (Error code: 4101).

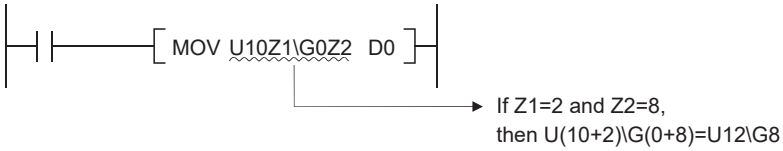


Other index modifications

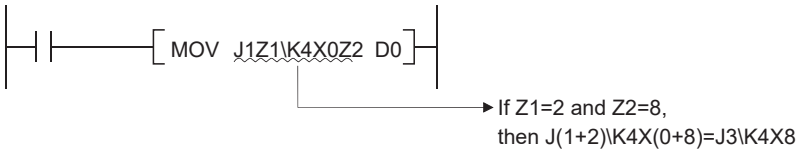
For bit data, device numbers can be index modified when performing digit designation.
However, Indexing is not possible by digit designation.



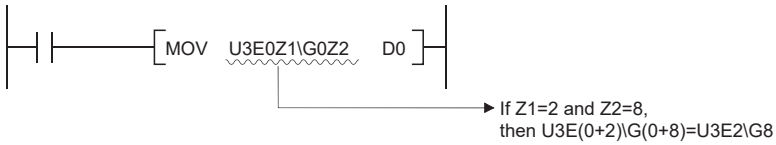
Both start I/O numbers and buffer memory addresses can be performed indexing with intelligent function module devices*1.



Both network numbers and device numbers can be performed indexing with link direct devices*1.



When indexing is used for multiple CPU shared devices*2, indexing for the head I/O numbers of CPU modules and indexing for the CPU shared memory address are automatically executed.



Like index modification using file register (ZR), index modification using extended data register (D) and extended link register (W) by 32 bits can be performed by the following two methods. *3

- Specifying the index registers' range used for indexing with 32-bit.
- Specifying the 32-bit indexing using "ZZ" specification.

*1 For the intelligent function module device and link direct devices, refer to the User's Manual (Function Explanation, Program Fundamentals) for the CPU module used.

*2 For the multiple CPU shared device, refer to the User's Manual (Function Explanation, Program Fundamentals) for the CPU module used.

*3 The methods applies only to Universal model QCPU (excluding Q00UJCPU) and LCPU.

Point

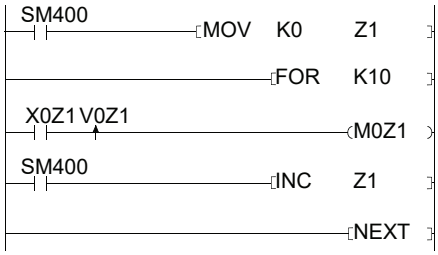
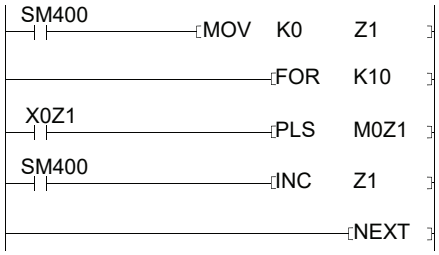
32-bit indexing with the "ZZ" specification is only available for the following CPU modules. See the programming tool operating manual for the available programming tools.

- The first five digits of the serial No. for QnU(D)(H)CPU is "10042" or higher (excluding Q00UJCPU).
- Built-in Ethernet port QCPU
- LCPU

Cautions

■Performing indexing between the FOR and NEXT instructions

Pulses can be output between the FOR and NEXT instructions by use of the edge relay (V). However, pulse output using the PLS/PLF/pulse (□P) instruction is not allowed.

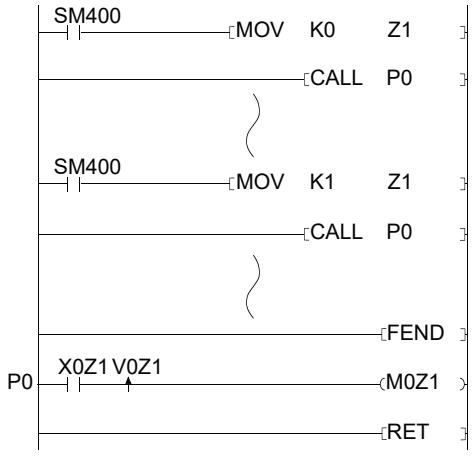
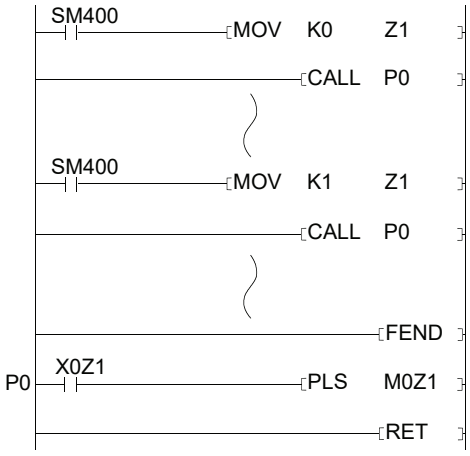
When edge relay is used	When edge relay is not used
M0Z1 provides normal pulse output. 	M0Z1 does not provide normal pulse output. 

Point

The ON/OFF data of X0Z1 is stored by the edge relay V0Z1.
For example, the ON/OFF data of X0 is stored by V0, and that of X1 by V1.

■Performing indexing with the CALL instruction

Pulses can be output with the CALL instruction by use of the edge relay (V). However, pulse output using the PLS/PLF/pulse (□P) instruction is not allowed.

When edge relay is used	When edge relay is not used
M0Z1 provides normal pulse output. 	M0Z1 does not provide normal pulse output. 

■Device range check during indexing

• Basic model QCPU, High Performance model QCPU, Process CPU, and Redundant CPU

Device range checks are not conducted during indexing.

Therefore, when the data after index modification exceed the user specified device range, the data is read by another device or written to another device without causing an error. (Note, however, that when the data after index modification is written to the device for system use exceeding the user specified device range, an error occurs. (Error code: 1103))

Take extra precaution when using indexing in programming.

• QnU(D)(H)CPU, QnUDE(H)CPU, and LCPU

The device range is checked for indexing.

With changing the settings of the PLC parameter, the device range is not checked.

The timings for checking the device range during exponent modification are shown below:

Instruction	Timings for checking
Contact instruction	Always*1
Association instruction	
Comparison operation instruction (LD☐)	
Comparison operation instruction (AND☐)	When previous conditions are ON*1
Comparison operation instruction (OR☐)	When previous conditions are OFF*1
Instructions other than the above	It follows the execution conditions for the instruction.*1*2*3

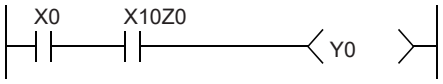
*1 When the data after index modification exceed the user specified device range, it may cause an error. (Error code: 4101).

*2 For the executions conditions for each instruction, refer to the descriptions page for each instruction.

*3 The PLS instruction and PLF instruction are excluded. (The PLS instruction and PLF instruction always check the device range during index modification.)

Ex.

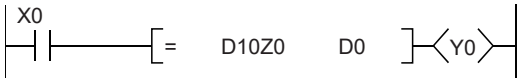
When X10Z0 exceeds the device range irrespective of ON or OFF of X0, it causes an error.



Ex.

When D10Z0 exceeds the device range while X0 is ON, it causes an error.

The device range is not checked during index modification when X0 is OFF. Therefore, even if the D10Z0 exceeds the device range, an error does not occur.



• For the QnUDVCPU and QnUDPVCPU:

The device range is checked during index modification.

It is also possible not to allow checking the device range using the parameters.

The timings for checking the device change during index modification are shown below.

Instruction	Timings for checking
Contact instruction	Always*4
Association instruction	
Comparison operation instruction (LD☐)	
Comparison operation instruction (AND☐)	
Comparison operation instruction (OR☐)	
Instructions other than the above	It follows the execution conditions for the instruction.*5*6*7

*4 When the data after index modification exceed the user specified device range, the operation results in OFF without causing an error.

*5 When the data after index modification exceed the user specified device range, it may cause an error. (Error code: 4101).

*6 For the executions conditions for each instruction, refer to the descriptions page for each instruction.

*7 The PLS instruction and PLF instruction are excluded. (The PLS instruction and PLF instruction always check the device range during index modification.)

■Changing indexing with 16-bit index register for indexing with 32-bit index register

For changing indexing with 16-bit index register for indexing with 32-bit index register, check if the program has enough spaces for indexing.

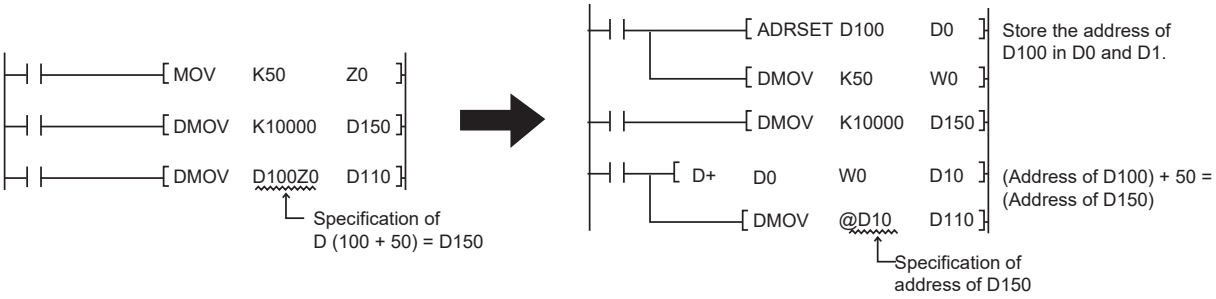
For indexing with 32-bit index registers, the specified index register (Zn) and the next index register of the specified register (Zn+1) are used. Be sure not to overlap index registers to be used.

3.4 Indirect Specification

Indirect Specification

Indirect specification is a method that specifies address of the device to be used in a sequence program using two word devices (two points of word device).

Use indirect specification as index modification when the index register is insufficient.



[When index resister is used] [When indirect specification is used]

Specify the device to be used for specifying the address as "@ + (word device number)".

For example, when @D100 is specified, the device address will be the contents of D101 and D100.

When using the indirect specification, be sure to execute the ADRESET instruction.

For the ADRSET instruction, refer to Page 767 Indirect address read operations.

Indirect specification available devices

The following table shows that the CPU module devices can be specified indirectly.

Device type		Availability of indirect specification	Example of indirect specification
Internal user device	Bit device*1	N/A	—
	Word device*1	Available	• @D100
Link direct device	Bit device*1	N/A	—
	Word device*1	Available*2	• @J1\W10
Intelligent function module device		Available*2	• @U10\G0
Index register		N/A	—
File register		Available	• @R0, @ZR20000
Extended data register (D)		Available	• @D1000 • @W1000
Extended link register (W)			
Nesting		N/A	—
Pointer			—
Constant			—
Others	SFC block device		—
	SFC transition device		
	Network No. specification device		
	I/O No. specification device		

*1 For the device names, refer to the User's Manual (Function Explanation, Program Fundamentals) for the CPU module used.

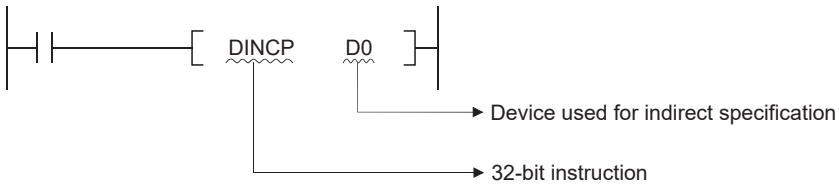
*2 Indirect specification is possible, but the address cannot be written with the ADRSET instruction.

Precautions

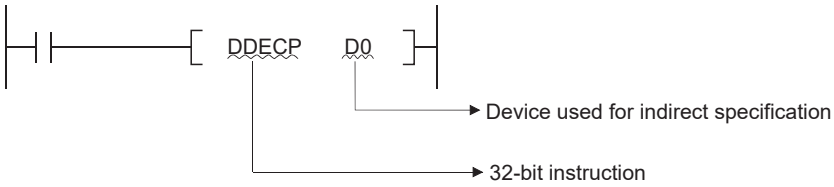
■Address for indirect specification

The address for indirect specification uses two words. Therefore, to substitute indirect specification for index modification, the addition/subtraction of 32-bit data is required. The following is the ladder used for the address addition/subtraction of the device stored in D1 and D0 for indirect specification.

[To add "1" to the address of the device for indirect specification]

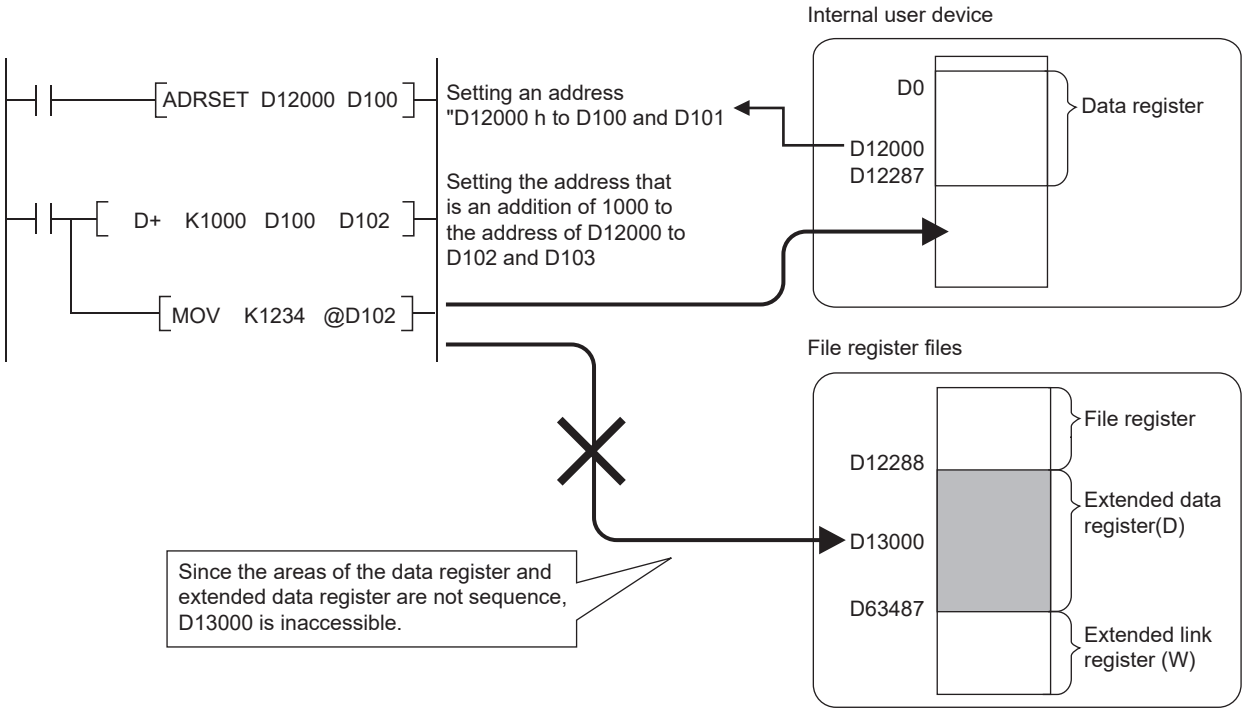


[To subtract "1" from the address of the device for indirect specification]



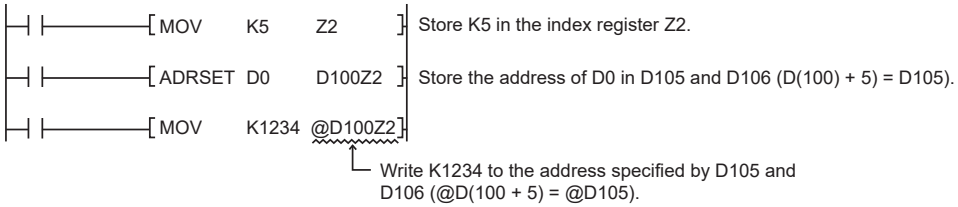
■Indirect specification of extended data register (D) and extended link register (W)

Indirect specification with indirect address can be performed in the extended data register (D) and extended link register (W). Note that when indirect specification is performed to the extended data register (D) and data register (D) in internal device or to the extended link register (W) and link register (W) in internal device, the areas of the internal user device and extended data register (D) or extended link register (W) are not treated as a sequence.



■Indirect specification and index modification

When a device is specified by both indirect specification and index modification, index modification is executed first and then the device is specified by indirect specification.



3.5 Reducing Instruction Processing Time

Subset processing

Subset processing is used to place limits on bit devices used by basic instructions and application instructions in order to increase processing speed.

However, the instruction symbol does not change.

To shorten scans, run instructions under the conditions indicated below.

Conditions which each device must meet for subset processing

■When using word data

Device	Condition
Bit device	• Designates a bit device number in a factor of 16. • Only K4 can be designated for digit designation. • Does not perform indexing.
Word device	• Internal user device • File register (R, ZR[*4]) • Multiple CPU shared device[*1*2] • Index register (Z) / Standard device register (Z)[*3]
Constant	• No limitations

■When using double word data

Device	Condition
Bit device	• Designates a bit device number in a factor of 16. • Only K8 can be designated for digit designation. • Does not perform indexing.
Word device	• Internal user device • File register (R, ZR[*4]) • Multiple CPU shared device[*1*2] • Index register (Z) / Standard device register (Z)[*3]
Constant	• No limitations

■When using bit data

Device	Condition
Bit device	• Internal user device (indexing possible)
Word device	• Bit specification of internal user device • Bit specification of file register (R, ZR[*4]) • Bit specification of multiple CPU shared device[*1*2]

*1 Only for Universal model QCPU

*2 Valid only for the multiple CPU high speed transmission area (from U3En\G10000)
(Excluding the case that indexing is executed for the head I/O number of the CPU module (U3En\G10000))

*3 Applies only to Universal model QCPU and LCPU.

*4 Applies only to Universal model QCPU (excluding Q00UJCPU) and LCPU.

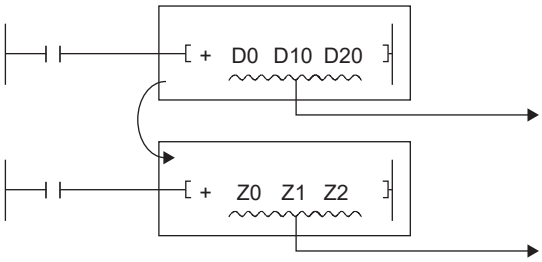
Instructions for which subset processing can be used

Types of instructions	Instruction symbol
Contact instructions	LD, LDI, AND, ANI, OR, ORI, LDP, LDF, ANDP, ANDF, ORP, ORF, LDPI, ANDPI, ANDFI, ORPI, ORFI
Output instructions	OUT, SET, RST
Comparison operation instructions	=, <>, <, <=, >, >=, D=, D<>, D<, D<=, D>, D>=
Arithmetic operation	+, -, *, /, INC, DEC, D+, D-, D*, D/, DINC, DDEC, B+, B-, B*, B/, E+, E-, E*, E/
Data conversion instructions	BCD, BIN, DBCD, DBIN, FLT, DFLT, INT, DINT
Data transfer instructions	MOV, DMOV, CML, DCML, XCH, DXCH, FMOV, BMOV, EMOV
Program branch instructions	CJ, SCJ, JMP
Logic operations	WAND, DAND, WOR, DOR, WXOR, DXOR, WXNR, DXNR
Rotation instructions	RCL, DRCL, RCR, DRCR, ROL, DROL, ROR, DROR
Shift instructions	SFL, DSFL, SFR, DSFR
Data processing instructions	SUM, SEG
Structure creation instructions	FOR, CALL

Operation processing with standard device registers (Z) (Universal model QCPU and LCPU only)

Operation processing time can be reduced with standard device registers (Z).

The following shows an example program with standard device registers.



Using data registers takes three steps and the operation processing time of 28.5 ns. (With the Q26UDEHCPU)

Using standard device registers instead of data registers takes one step and the operation processing time of 9.5 ns. (With the Q26UDEHCPU)

Operation processing time is reduced with the instructions that the subset processing is possible.

For the number of steps, refer to Page 119 Counting Step Number.

For the operation time for each instruction, refer to Page 890 Operation Processing Time.

Point

Because standard device registers are the same devices as index registers, do not use device numbers of the standard device registers for the index registers.

3.6 Cautions on Programming (Operation Errors)

Operation errors are returned in the following cases when executing basic instructions and application instructions with CPU module:

- An error listed on the explanatory page for the individual instruction occurred.
- When an intelligent function module device is used, no intelligent function module is installed at the specified I/O number position.
- When an intelligent function module device is used, the specified buffer memory address does not exist.
- The relevant network does not exist when using a link device.
- When a link device is used, no network module is installed at the specified I/O number position.
- When a multiple CPU shared device is used, a CPU module is not installed at the head I/O number position of the specified CPU module.
- When a multiple CPU shared device is used, the specified shared memory address does not exist.
- The setting of the device number crosses over the boundary between the internal user device and the extended data register (D) or extended link register (W). (Universal model QCPU (excluding Q00UJCPU) and LCPU)

Point

If data is read from or written to a file register when no file register file is set in parameter or the file register file set in parameter is not found, the following occurs.

• For the High Performance model QCPU, Process CPU, and Redundant CPU

An error does not occur even when writing/reading to/from file register is performed. However, "0H" is stored when reading from file register is performed.

• For the Universal model QCPU and LCPU

The OPERATION ERROR (error code: 4101) occurs when writing/reading to/from file register is performed. Note that the device range check can be disabled using the PLC Parameter so that an error will not be detected. (☞ Page 112 Device range check)

Device range check

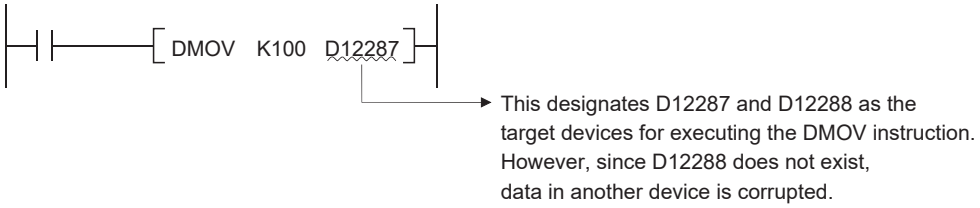
Device range checks for the devices used by basic instructions and application instructions in CPU module are as indicated below:

■Instructions for specified each device, including MOV and DMOV

• For the Basic model QCPU, High Performance model QCPU, Process CPU, and Redundant CPU.

The device range is not checked. In cases where the corresponding device range is exceeded, data is written to other devices. *1

For example, in a case where the data register has been allocated 12K points, there will be no error even if it exceeds D12287.



Device range checks are not conducted also in cases where indexing is being performed.

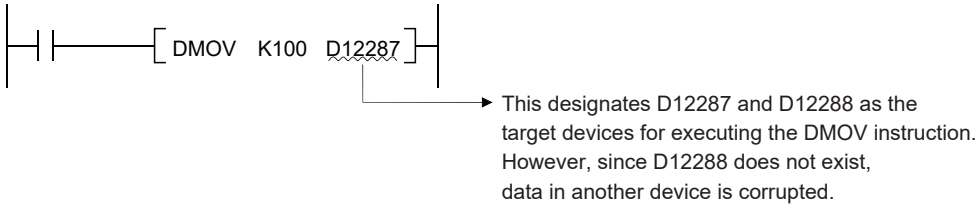
In cases where the corresponding device range is exceeded as the result of performing indexing, data is written to other devices. *1

*1 For the assignment order of internal user devices, refer to Page 114 Character string data.

• Universal model QCPU and LCPU

The device range is checked. When the device number is outside the device range, an operation error occurs.

For example, when 12K points are assigned to a data register, an error occurs if the device number of the data register exceeds D12287.



The device range is checked even though indexing is executed.

With changing the settings of the PLC parameter, the device range is not checked. *2

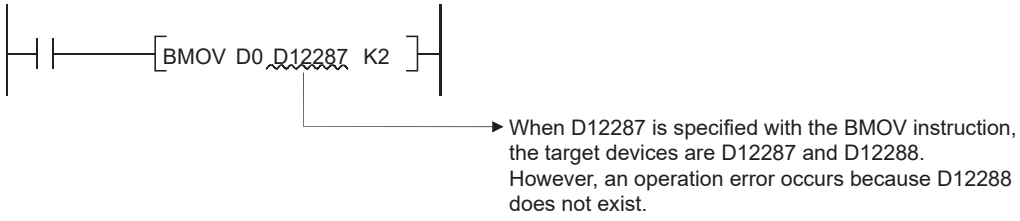
*2 For the method that the device range check is disabled when the index modification is specified, refer to the User's Manual (Function Explanation, Program Fundamentals) for the CPU module used.

■Instructions for a block of devices, including BMOV and FMOV

• For the Basic model QCPU, High Performance model QCPU, Process CPU, and Redundant CPU.

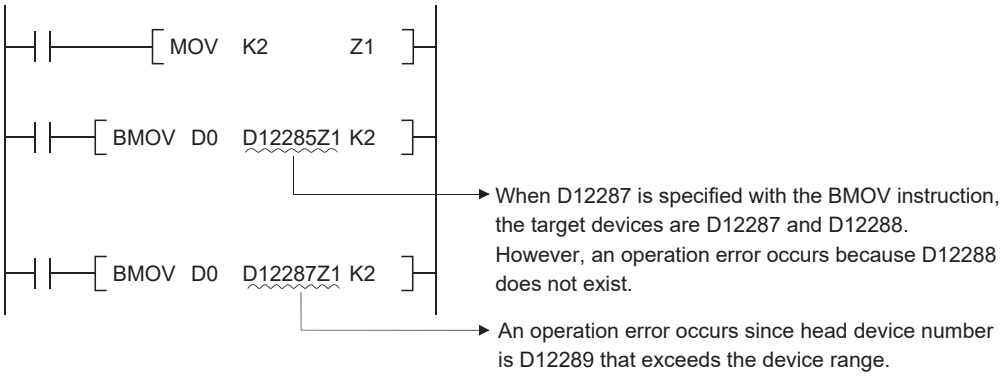
The device range is checked.

When the device number is outside the device range, an operation error occurs. For example, when 12K points are assigned to a data register, an error occurs if the device number of the data register exceeds D12287.



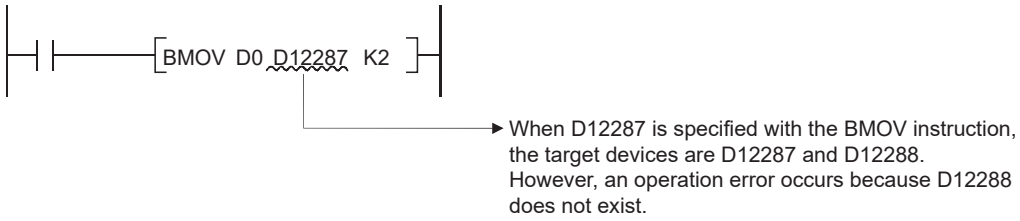
Device range checks are also conducted when indexing is performed.

However, if indexing has been conducted, there will be no error returned if the initial device number exceeds the relevant device range.

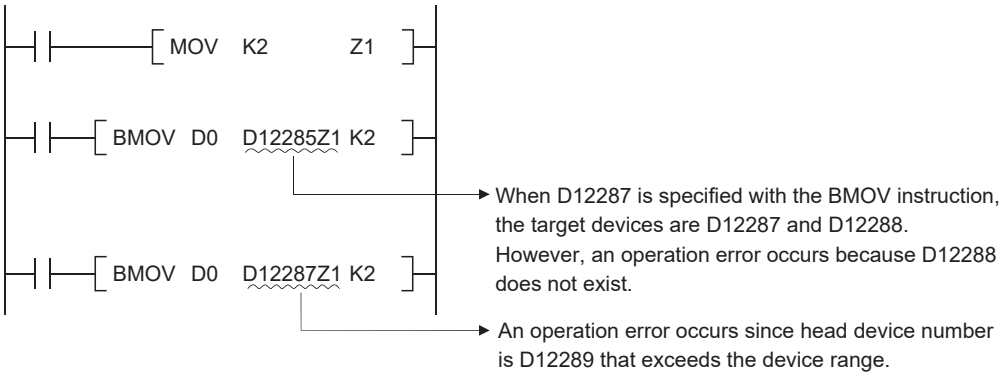


• Universal model QCPU and LCPU

The device range is checked. When the device number is outside the device range, an operation error occurs. For example, when 12K points are assigned to a data register, an error occurs if the device number of the data register exceeds D12287.



The device range is checked even though indexing is executed. An error occurs when the head device number of the devices with indexing exceeds the device range.



With changing the settings of the PLC parameter, the device range is not checked. *1

*1 For the method that the device range check is disabled when the index modification is specified, refer to the User's Manual (Function Explanation, Program Fundamentals) for the CPU module used.

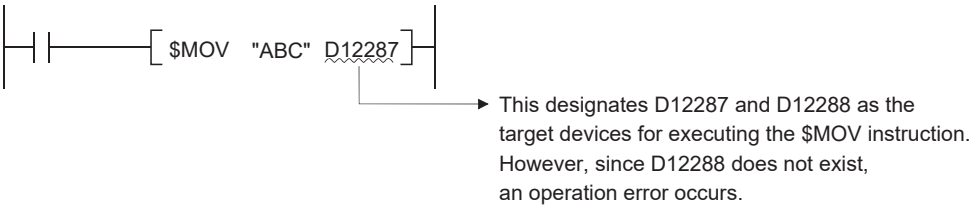
■Character string data

• For the Basic model QCPU, High Performance model QCPU, Process CPU, and Redundant CPU.

Because all character string data is of variable length, device range checks are performed.

When the device number is outside the device range, an operation error occurs.

For example, in a case where the data register has been allocated 12K points, there will be an error if it exceeds D12287.



The device range check is performed even when the index modification is specified.

In case that the index modification is specified, if the start device number exceeds the device range, an error does not occur.

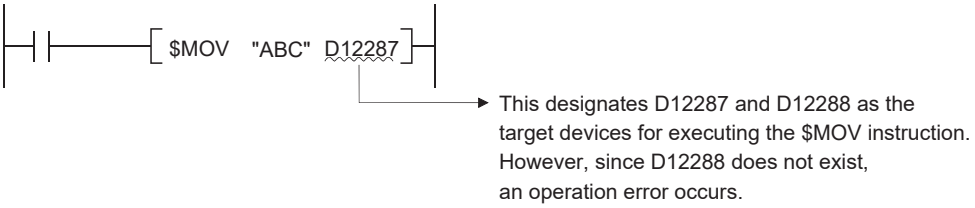
Another device is accessed.

• For the Universal model QCPU and LCPU

Because all character string data is of variable length, device range checks are performed.

When the device number is outside the device range, an operation error occurs.

For example, in a case where the data register has been allocated 12K points, there will be an error if it exceeds D12287.



The device range check is performed even when the index modification is specified. In case that the index modification is specified, an error occurs if the start device number exceeds the device range.

With changing the settings of the PLC parameter, the device range is not checked. *1

*1 For the method that the device range check is disabled when the index modification is specified, refer to the User's Manual (Function Explanation, Program Fundamentals) for the CPU module used.

■Direct access output (DY)

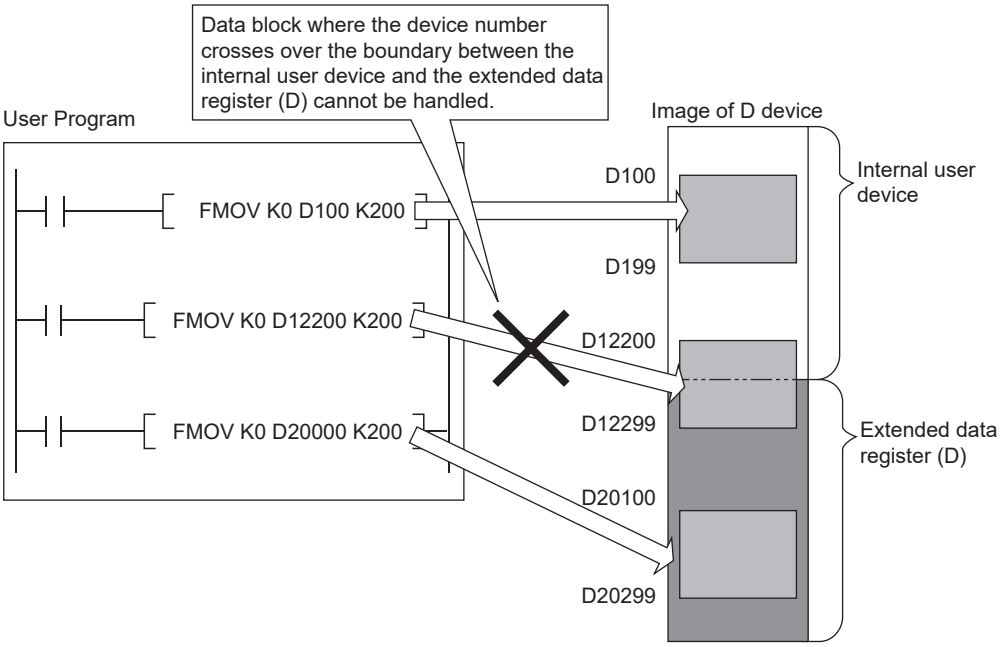
Device range checks are conducted when indexing is performed by direct access output (DY).

■Precautions for using the extended data register (D) or extended link register (W)*1

With the following specification methods, data cannot be specified crossing over the boundary of the internal user device and extended data register (D) or extended link register (W). Doing so causes "OPERATION ERROR" (error code: 4101).

*1 Universal model QCPU (models other than Q00UJCPU) and LCPU are applicable.

- Index modification
- Indirect specification
- Specification with the instructions that handle data blocks



Data block indicates the following data.

- Data used in the instructions, such as FMOV, BMOV, BK+, which multiple words are targeted for operation
- Control data, composed of two or more words, specified in the instructions, such as SP.FWRITE, SP.FREAD
- Data whose data type is 32-bit or more (BIN 32-bit, real number, indirect address of the device)

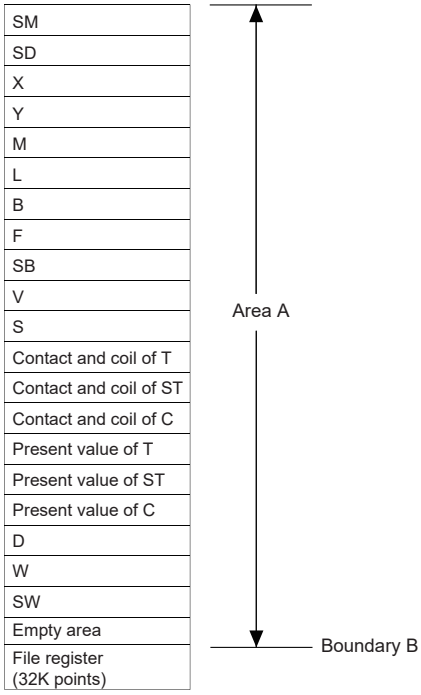
■Precautions when using Universal model QCPU/LCPU

For the Universal model QCPU and LCPU, an error occurs if any of the following accesses is performed using the following instructions and data. (Error code: 4101)

Instructions and data
• Instructions for specified each device, including MOV and DMOV • Instructions for a block of devices, including BMOV and FMOV • Character string data

Access method
(1) Access crossing the boundary of devices caused by index modification (range of A area)*1 (2) Access crossing the boundary of file registers caused by index modification (3) Access to file registers (R, ZR) without setting file register files (4) Access to file registers (R, ZR) exceeded the range of file register files

*1 The allocation order of individual devices is shown below:



Note that the device range check can be disabled using the PLC Parameter so that an error will not be detected even when the above accesses are performed.

As shown in the following table, the operation of the Universal model QCPU may differ, depending on the serial number. *2

Setting device range in indexing	First five digits of serial No. for Universal model QCPU		LCPU
	Serial No."10021" or lower	Serial No."10022" or higher	
Set	Detected errors in accesses 1) to 4)		
Not set	Detected errors in accesses 2) to 4)	Not detected	

*2 For the method that the device range check is disabled when the index modification is specified, refer to the User's Manual (Function Explanation, Program Fundamentals) for the CPU module used.

For Universal model QCPU and LCPU, the index modification that crosses internal user devices (SW) and file registers (R) cannot be applied. (Error code: 4101)

Point

For how to change the internal user device allocation, refer to the User's Manual (Function Explanation, Program Fundamentals) for the CPU module used.

Device data check

Device data checks for the devices used by basic instructions and application instructions in CPU module are as indicated below:

■When using BIN data

No error is returned even if the operation results in overflow or underflow. The carry flag (SM700) does not go on at such times, either.

■When using BCD data

- Each digit is check for BCD value (0 to 9). An operation error is returned if individual digits are outside the 0 to 9 (A to F) range.
- No error is returned even if the operation results in overflow or underflow. The carry flag (SM700) does not go on at such times, either.

■When using floating-point data

- An operation error occurs when the following operation results are returned with the single-precision floating-point operation instruction.

When the absolute value of the floating decimal point data is 1.0×2^{-127} or lower

When absolute value of floating decimal point data is 1.0×2^{128} or higher

- An operation error occurs when the following operation results are returned with the double-precision floating-point operation instruction.

When the absolute value of the floating decimal point data is 1.0×2^{-1023} or lower

When absolute value of floating decimal point data is 1.0×2^{1024} or higher

■Using character string data

No data check is conducted.

Buffer memory access

For accessing buffer memories, using instructions with intelligent function module devices (from Un\G0) is recommended.

Multiple CPU shared memory access

For accessing multiple CPU shared memories, using instructions with multiple CPU shared devices (from U3En\G10000) is recommended.

3.7 Conditions for Execution of Instructions

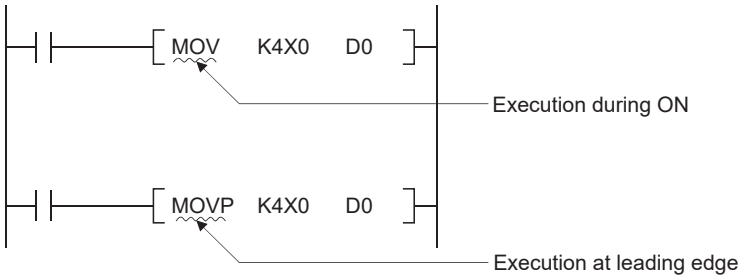
The following four types of execution conditions exist for the execution of CPU module sequence instructions, basic instructions, and application instructions:

Execution condition		Description
Non-conditional execution	—	An instruction is always executed regardless of whether the precondition of the instruction is on or off. When the precondition is off, the instruction performs off processing.
Executed at ON		An instruction is executed during on. It is executed only while the precondition is on. When the preconditions is off, the instruction is not executed.
Executed at the rising edge		An instruction is executed one time when turned on. It is executed only once on the rising edge (→off to on) of the precondition of the instruction and is no longer executed later even when the condition turns on.
Executed at OFF		An instruction is executed during off. It is executed only while the precondition is off. When the precondition is on, the instruction is not executed.
Executed at the falling edge		An instruction is executed one time when turned off. It is executed only once on the falling edge (→on to off) of the precondition of the instruction and is no longer executed later even when the condition turns off.

For coil or equivalent basic instructions or application instructions, where the same instruction can be designated for either execution at ON or rising edge execution, a "P" is added after the instruction name to specify the condition for execution.

- Instruction to be executed at ON [Instruction name]
- Instruction to be executed at rising edge [Instruction name] + P

Execution at ON and execution at rising edge for the MOV instruction are designated as follows:

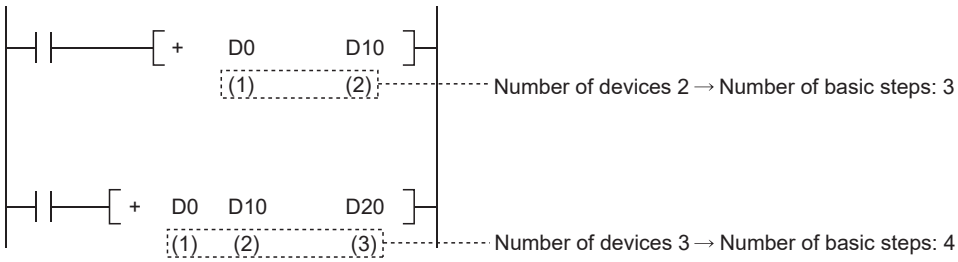


3.8 Counting Step Number

The number of steps in CPU module sequence instructions, basic instructions, and application instructions differs depending on whether indirect setting of the device used is possible or not.

Counting the number of basic steps

The basic number of steps for basic instructions and application instructions is calculated by adding the device number and 1. For example, the "+ instruction" would be calculated as follows:



Conditions for increasing the number of steps

The number of steps is increased over the number of basic steps in cases where a device is used that is designated indirectly or for which the number of steps is increased.

■When device is designated indirectly

In cases where indirect designation is done by @☐, the number of steps is increased 1 step over the number of basic steps. For example, when a 3-step MOV instruction is designated indirectly (example: MOV K4X0 @D0), one step is added and the instruction becomes 4 steps.

■Devices with additional steps (the Basic Model QCPU, High Performance model QCPU, Process CPU, and Redundant CPU)

Devices with additional steps	Added steps	Example
Intelligent function module device	1	MOV U4\G10 D0
Multiple CPU shared device		MOV U3E1\G0 D0
Link direct device		MOV J3\B20 D0
Index register		MOV Z0 D0
Serial number access format file register		MOV ZR123 D0
32-bit constant		DMOV K123 D0
Real constant		EMOV E0.1 D0
Character string constant	For even numbers: (number of characters) / 2 For odd numbers: (number of characters + 1) / 2	$MOV "123" D0

■Devices with additional steps (Universal model QCPU(except Q00UJCPU) and LCPU)

The following table shows steps depending on the devices.

Instruction symbol	Devices with additional steps	Added steps (number of instruction steps)	Number of basic steps
LD, LDI, AND, ANI, OR, ORI, LDP, LDF, ANDP, ANDF, ORP, ORF	Serial number access format file register, Extended data register (D), Extended link register (W)	1(2)	1
	Multiple CPU shared device*3		
LDPI, LDFI	Serial number access format file register, Extended data register (D), Extended link register (W)	1(4)	3
	Multiple CPU shared device*3		
ANDPI, ANDFI, ORPI, ORFI	Serial number access format file register, Extended data register (D), Extended link register (W)	1(5)	4
	Multiple CPU shared device*3		

Instruction symbol	Devices with additional steps	Added steps (number of instruction steps)	Number of basic steps
SET	Serial number access format file register, Extended data register (D), Extended link register (W)	1(2)	1
	Multiple CPU shared device*3		
OUT	Timer/Counter	3(4)	1
	Serial number access format file register, Extended data register (D), Extended link register (W)	1(2)	
	Multiple CPU shared device*3		
RST (bit device)	Serial number access format file register, Extended data register (D), Extended link register (W)	1(2)	1
	Multiple CPU shared device*3		
RST (word device)	Timer/Counter (Bit/word device)	2(4)	2
	Serial number access format file register, Extended data register (D), Extended link register (W)	1(3)	
	Multiple CPU shared device*3	1(3)	
LD=, LD<>, LD<, LD<=, LD>, LD>=, AND=, AND<>, AND<, AND<=, AND>, AND>=, OR=, OR<>, OR<, OR<=, OR>, OR>=	Standard device register*2	-1	3
	Serial number access format file register, Extended data register (D), Extended link register (W)	1	
	Multiple CPU shared device*3		
LDD=, LDD<>, LDD<, LDD<=, LDD>, LDD>=, ANDD=, ANDD<>, ANDD<, ANDD<=, ANDD>, AND>=, ORD=, ORD<>, ORD<, ORD<=, ORD>, ORD>=	Standard device register*2	-1	3
	Serial number access format file register, Extended data register (D), Extended link register (W)	1	
	Multiple CPU shared device*3		
	Decimal constant, hexadecimal constant, real constant		
+, -, +P, -P, WAND, WOR, WXOR, WXNR, WANDP, WORP, WXORP, WXNRP (2 devices)	Standard device register*2	(D): -1	3
	Serial number access format file register, Extended data register (D), Extended link register (W)	(S): 1, (D): 3	
	Multiple CPU shared device*3		
D+, D-, D+P, D-P, DAND, DOR, DXOR, DXNR, DANDP, DORP, DXORP, DXNRP (2 devices)	Standard device register*2	(D): -1	3
	Serial number access format file register, Extended data register (D), Extended link register (W)	(S): 1, (D): 3	
	Multiple CPU shared device*3		
	Decimal constant, hexadecimal constant, real constant	(S): 1	
+, -, +P, -P, WAND, WOR, WXOR, WXNR, WANDP, WORP, WXORP, WXNRP (3 devices)*1	Serial number access format file register, Extended data register (D), Extended link register (W)	(S1), (S2): 1, (D): 2	3
	Multiple CPU shared device*3		
D+, D-, D+P, D-P, DAND, DOR, DXOR, DXNR, DANDP, DORP, DXORP, DXNRP (3 devices)*1	Serial number access format file register, Extended data register (D), Extended link register (W)	(S1), (S2): 1, (D): 2	3
	Multiple CPU shared device*3		
	Decimal constant, hexadecimal constant, real constant	(S1), (S2): 1	
*, *P, /, /P	Serial number access format file register, Extended data register (D), Extended link register (W)	(S1), (S2): 1, (D): 2	3
	Multiple CPU shared device*3		
D*, D*P, D/, D/P, E*, E*P	Serial number access format file register, Extended data register (D), Extended link register (W)	(S1), (S2): 1, (D): 2	3
	Multiple CPU shared device*3		
	Decimal constant, hexadecimal constant, real constant	(S1), (S2): 1	
INC, INCP, DEC, DECP, DINC, DINCP, DDEC, DDECP	Index register / standard device register*2	-1	2
	Serial number access format file register, Extended data register (D), Extended link register (W)	3	
	Multiple CPU shared device*3		
MOV, MOVP	Serial number access format file register, Extended data register (D), Extended link register (W)	1	2
	Multiple CPU shared device*3		

Instruction symbol	Devices with additional steps	Added steps (number of instruction steps)	Number of basic steps
DMOV, DMOVP, EMOV, EMOVP	Serial number access format file register, Extended data register (D), Extended link register (W)	1	2
	Multiple CPU shared device*3		
	Decimal constant, hexadecimal constant, real constant		
BCD, BCDP, BIN, BINP, FLT, FLTP, CML, CMLP	Serial number access format file register, Extended data register (D), Extended link register (W)	(S): 1, (D): 2	2
	Multiple CPU shared device*3		
DBCD, DBCDP, DBIN, DBINP, INT, INTP, DINT, DINTP, DFLT, DFLTP, DCML, DCMLP	Serial number access format file register, Extended data register (D), Extended link register (W)	(S): 1, (D): 2	2
	Multiple CPU shared device*3		
	Decimal constant, hexadecimal constant, real constant	(S): 1	

*1 If the same device is used for (S1) and (S2), the number of basic steps increases by one.
*2 The number of steps decreases with a standard device register.
*3 Not available with LCPU.

When multiple standard device registers are used in an instruction applicable to subset processing, the number of steps decreases.

The following table shows the number of steps for each instruction.

Instruction symbol	Locations where standard device register is used	Added steps (number of instruction steps)	Number of basic steps
LD=, LD<>, LD<, LD<=, LD>, LD>=, AND=, AND<>, AND<, AND<=, AND>, AND>=, OR=, OR<>, OR<, OR<=, OR>, OR>=, LDD=, LDD<>, LDD<, LDD<=, LDD>, LDD>=, ANDD=, ANDD<>, ANDD<, ANDD<=, ANDD>, AND>=, ORD=, ORD<>, ORD<, ORD<=, ORD>, ORD>=	(S1) and (S2)	-2(1)	3
+, -, +P, -P, D+, D-, D+P, D-P, WAND, WOR, WXOR, WXNR, DAND, DOR, DXOR, DXNR, WANDP, WORP, WXORP, WXNRP, DANDP, DORP, DXORP, DXNRP (2 devices)	(S) and (D)	-2(1)	3
+, -, +P, -P, D+, D-, D+P, D-P, WAND, WOR, WXOR, WXNR, DAND, DOR, DXOR, DXNR, WANDP, WORP, WXORP, WXNRP, DANDP, DORP, DXORP, DXNRP (3 devices) *4	(S1), (S2), and (D)	-2(1)	3
	(S1), or (S2) and (D)	-1(2)	
	(S1) and (S2) (only when that device that the number of steps does not increase is specified for (D))	±0(3)	
	(S1) and (S2) (only when a serial number access format file register is specified for (D))	+2(5)	
*, *P, /, /P	(S1), (S2), and (D)	-2(1)	3
	(S1), or (S2) and (D)	-1(2)	
D*, D*P, D/, D/P, E*, E*P	(S1), (S2), and (D)	-2(1)	3
	(S1), or (S2) and (D)	-1(2)	
	(S1) and (S2) (only when that device that the number of steps does not increase is specified for (D))	±0(3)	
	(S1) and (S2) (only when a serial number access format file register is specified for (D))	+2(5)	
MOV, MOVP, DMOV, DMOVP, EMOV, EMOVP	(S) and (D)	-1(1)	2
BCD, BCDP, BIN, BINP, DBCD, DBCDP, DBIN, DBINP, FLT, FLTP, DFLT, DFLTP, INT, INTP, DINT, DINTP, CML, CMLP, DCML, DCMLP	(S) and (D)	-1(1)	2

*4 If the same device is used for (S1) and (D), the number of basic steps increases by one.

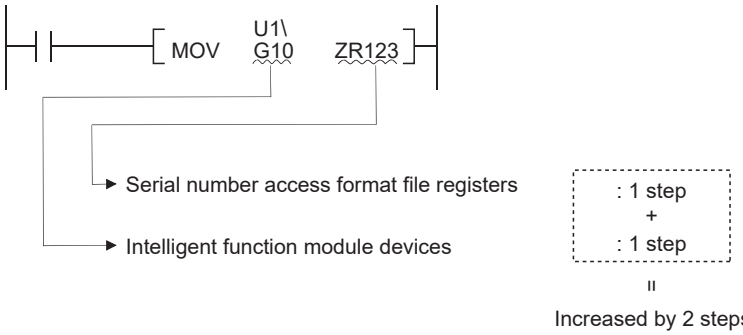
The following table shows steps depending on the devices.

Devices with additional steps	Added steps	Example
Intelligent function module device	1	MOV U4\G10 D0
Multiple CPU shared device		MOV U3E1\G10000 D0
Link direct device		MOV J3\B20 D0
Index register / standard device register		MOV Z0 D0
Serial number access format file register		MOV ZR123 D0
32-bit constant		DMOV K123 D0
Real constant		EMOV E0.1 D0
Character string constant	For even numbers: (number of characters) / 2 For odd numbers: (number of characters + 1) / 2	$MOV "123" D0

In cases where the conditions overlap, the number of steps becomes a cumulation of the two.

Ex.

MOV If U1\G10 ZR123 has been designated, a total of 2 steps are added.



■When replacing the QnUDE(H)CPU with the QnUDVCPU or QnUDPVCPU

The conditions for increasing or decreasing the number of steps are different when replacing the QnUDE(H)CPU with the QnUDVCPU or QnUDPVCPU. Accordingly the number of program steps may be increased or decreased.

For increase or decrease in the number of steps, refer to the QnUCPU User's Manual (Function Explanation, Program Fundamentals).

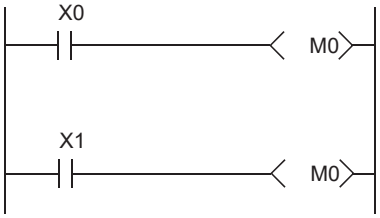
3.9 Operation When the OUT, SET/RST, or PLS/PLF Instructions Use the Same Device

The following describes the operation for executing multiple instructions of the OUT, SET/RST, or PLS/PLF that use the same device in one scan.

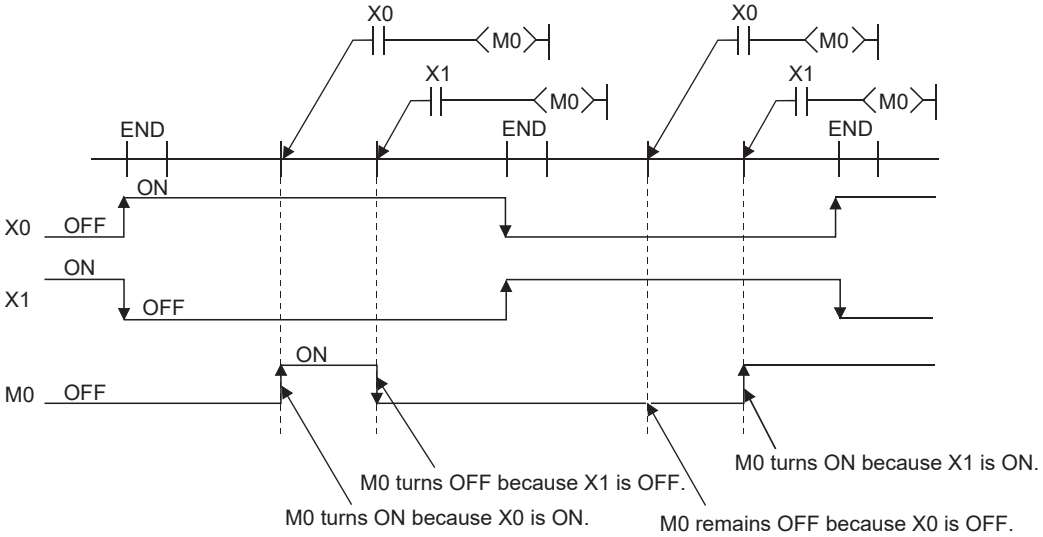
OUT instructions using the same device

Do not program more than one OUT instruction using the same device in one scan. If the OUT instructions using the same device are programmed in one scan, the specified device will turn ON or OFF every time the OUT instruction is executed, depending on the operation result of the program up to the relevant OUT instruction. Since turning ON or OFF of the device is determined when each OUT instruction is executed, the device may turn ON and OFF repeatedly during one scan. The following diagram shows an example of a ladder that turns the same internal relay (M0) with inputs X0 and X1 ON and OFF.

[Ladder]



[Timing Chart]

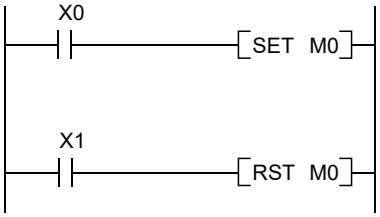


With the refresh type CPU module, when the output (Y) is specified by the OUT instruction, the ON/OFF status of the last OUT instruction of the scan will be output.

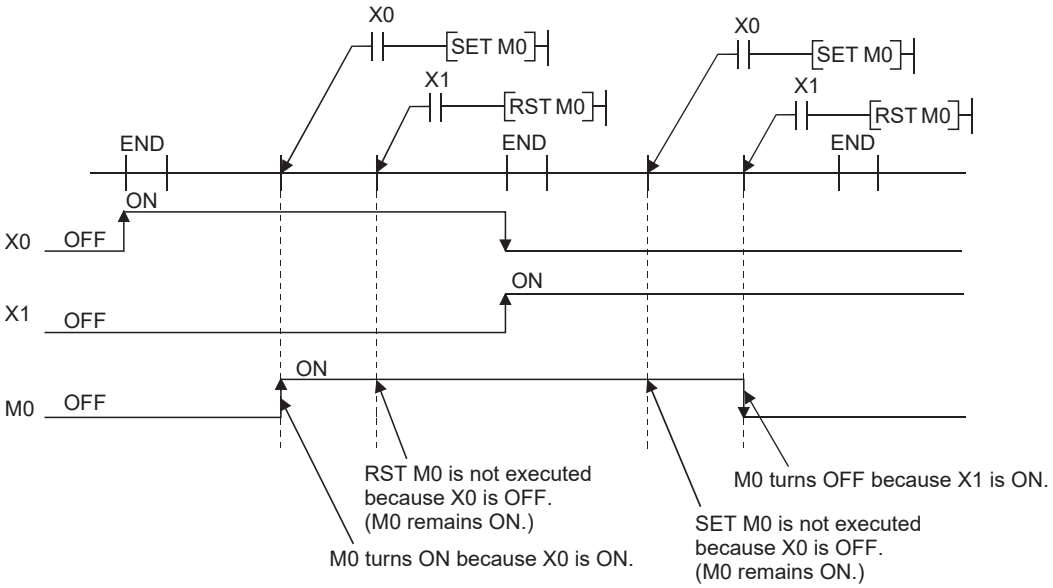
SET/RST instructions using the same device

- The SET instruction turns ON the specified device when the execution command is ON and performs nothing when the execution command is OFF. For this reason, when the SET instructions using the same device are executed two or more times in one scan, the specified device will be ON if any one of the execution commands is ON.
- The RST instruction turns OFF the specified device when the execution command is ON and performs nothing when the execution command is OFF. For this reason, when the RST instructions using the same device are executed two or more times in one scan, the specified device will be OFF if any one of the execution commands is ON.
- When the SET instruction and RST instruction using the same device are programmed in one scan, the SET instruction turns ON the specified device when the SET execution command is ON and the RST instruction turns OFF the specified device when the RST execution command is ON. When both the SET and RST execution commands are OFF, the ON/OFF status of the specified device will not be changed.

[Ladder]



[Timing Chart]

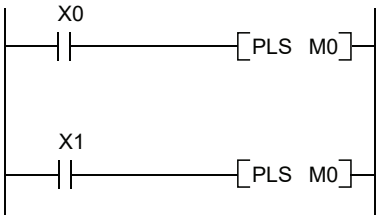


When using a refresh type CPU module and specifying output (Y) in the SET/RST instruction, the ON/OFF status of the device at the execution of the last instruction in the scan is returned as the output (Y).

PLS instructions using the same device

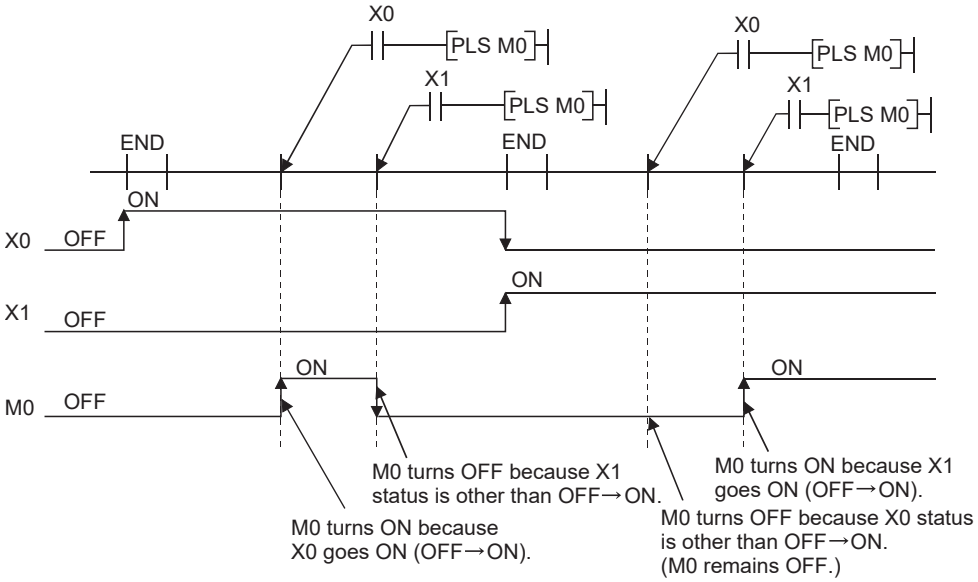
The PLS instruction turns ON the specified device when the execution command is turned ON from OFF. It turns OFF the device at any other time (OFF to OFF, ON to ON, or ON to OFF). If two or more PLS instructions using the same device are executed in one scan, each instruction turns ON the device when the corresponding execution command is turned ON from OFF and turns OFF the device in other cases. For this reason, if multiple PLS instructions using the same device are executed in a single scan, a device that has been turned ON by the PLS instruction may not be turned ON during one scan.

[Ladder]

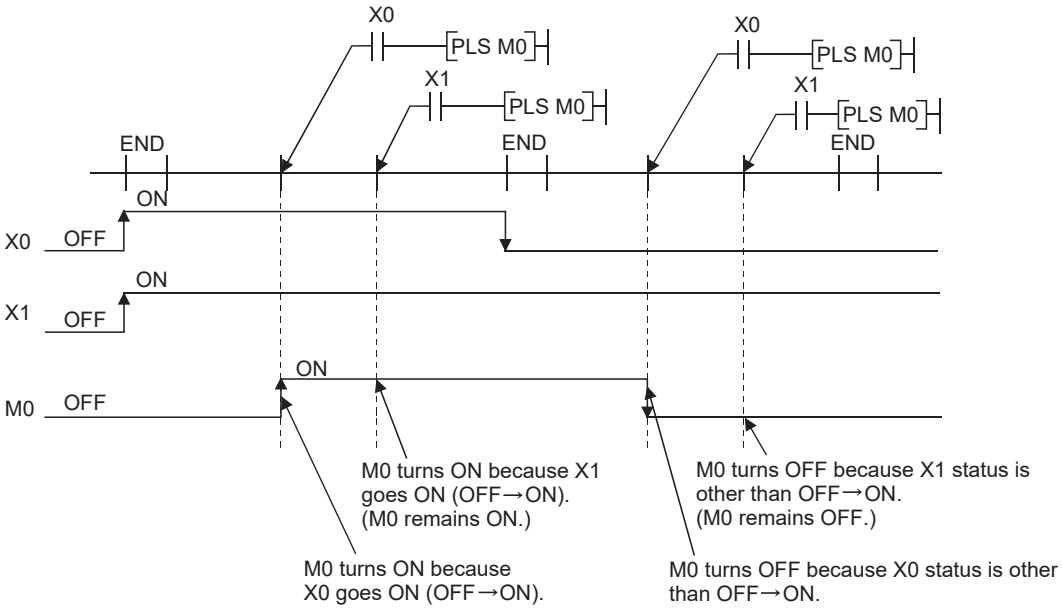


[Timing Chart]

• The ON/OFF timing of the X0 and X1 is different. (The specified device does not turn ON throughout the scan.)



• The X0 and X1 turn ON from OFF at the same time.

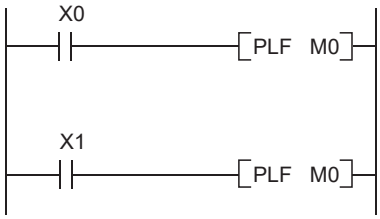


When using a refresh type CPU module and specifying output (Y) in the PLS instructions, the ON/OFF status of the device at the execution of the last PLS instruction in the scan is returned as the output (Y).

PLF instructions using the same device

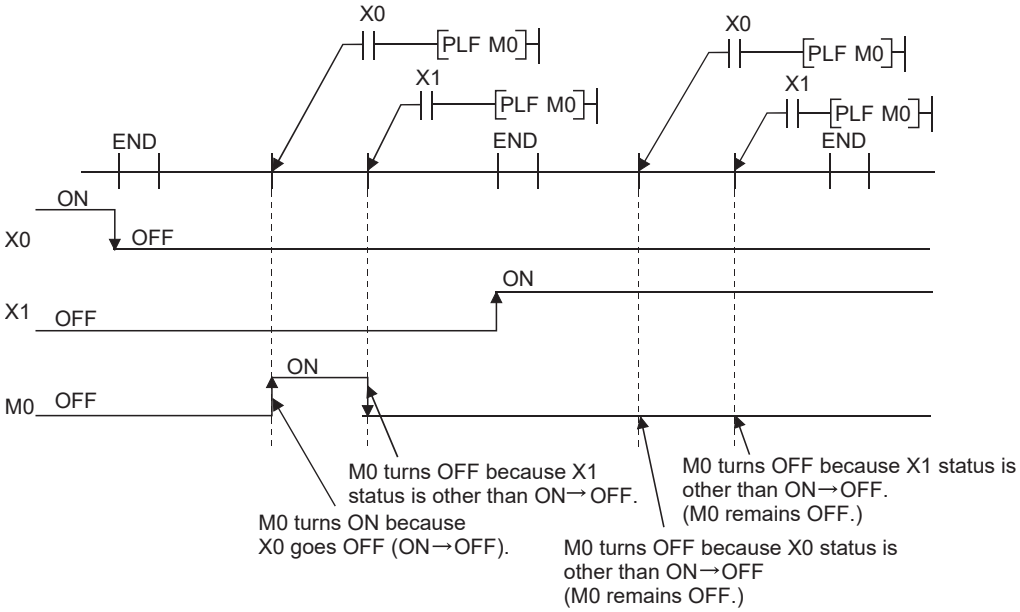
The PLF instruction turns ON the specified device when the execution command is turned OFF from ON. It turns OFF the device at any other time (OFF to OFF, OFF to ON, or ON to ON). If two or more PLF instructions using the same device are executed in one scan, each instruction turns ON the device when the corresponding execution command is turned OFF from ON and turns OFF the device in other cases. For this reason, if multiple PLF instructions using the same device are executed in a single scan, a device that has been turned ON by the PLF instruction may not be turn ON during one scan.

[Ladder]

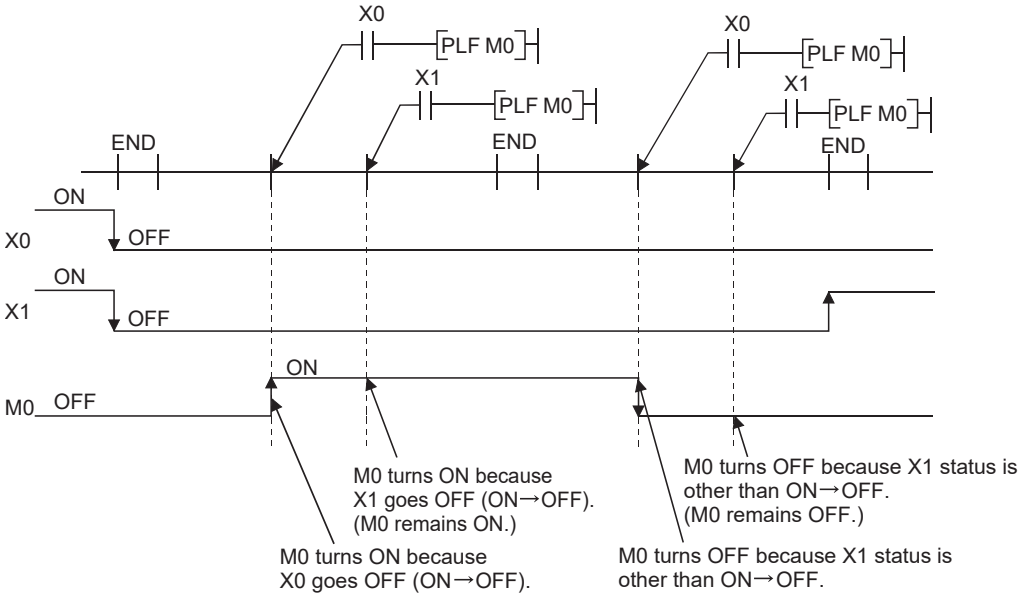


[Timing Chart]

• The ON/OFF timing of the X0 and X1 is different. (The specified device does not turn ON throughout the scan.)



• The X0 and X1 turn OFF from ON at the same time.



When using a refresh type CPU module and specifying output (Y) in the PLF instructions, the ON/OFF status of the device at the execution of the last PLF instruction in the scan is returned as the output (Y).

3.10 Precautions for Use of File Registers

This section explains the precautions for use of the file registers in the QCPU and LCPU.

CPU modules that cannot use file registers

The Q00JCPU and Q00UJCPU cannot use the file registers. When using the file registers, use the CPU module of other than the Q00JCPU and Q00UJCPU.

Setting of file registers to be used

When using the file registers, the file registers to be used must be set with the PLC parameter or QDRSET instruction. (The PLC parameters of the Q00CPU, Q01CPU and LCPU need not be set since they are preset to "Use file register". QDRSET instructions are not available with LCPU.) If the file registers to be used have not been set, normal operation cannot be performed with the instructions that use the file registers.

Point

Even when file registers to be used are not set in the PLC parameter, a program that uses file registers can be created. For the CPU module other than the Universal model QCPU and LCPU, an error does not occur when that program is written to the CPU module.
However, note that the correct data cannot be written/read to/from the file register.
For the Universal model QCPU and LCPU, an error occurs if the program where file registers are used is executed.

Securing of file register area

■Basic Model QCPU

The user does not need to secure a file register area, because such a file register area is reserved in advance in the standard RAM.

■High Performance model QCPU, Process CPU, Redundant CPU, and Universal model QCPU (except High-speed Universal model QCPU and Universal model Process CPU)

Register file registers in the standard RAM/memory card to secure a file register area.

■High-speed Universal model QCPU, Universal model Process CPU, and LCPU

Register file registers in the standard RAM to secure a file register area.

Point

For the setting method for file registers and memories available for the file registers of CPU modules, refer to the User's Manual (Function Explanation, Program Fundamentals) for the CPU module used.

Designation of file register number in excess of the registered number of points

■Basic model QCPU, High Performance model QCPU, Process CPU, and Redundant CPU

An error will not occur if data are written or read to or from the file registers that have numbers greater than the registered number of points. However, note that the read/write of correct data to/from the file registers cannot be performed.

■Universal model QCPU and LCPU

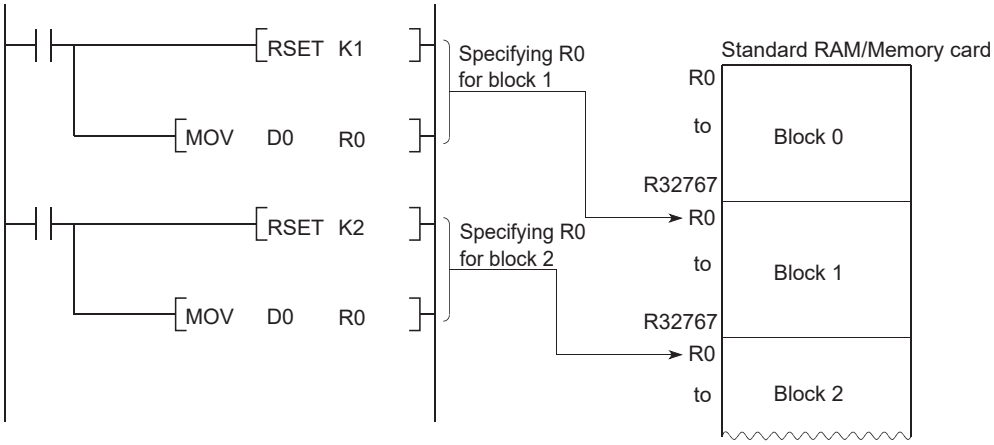
When data are written to or read from the file registers that are not registered, an error occurs. (Error code: 4101)

File register specifying method

There are the block switching method and serial number access method to specify the file registers.

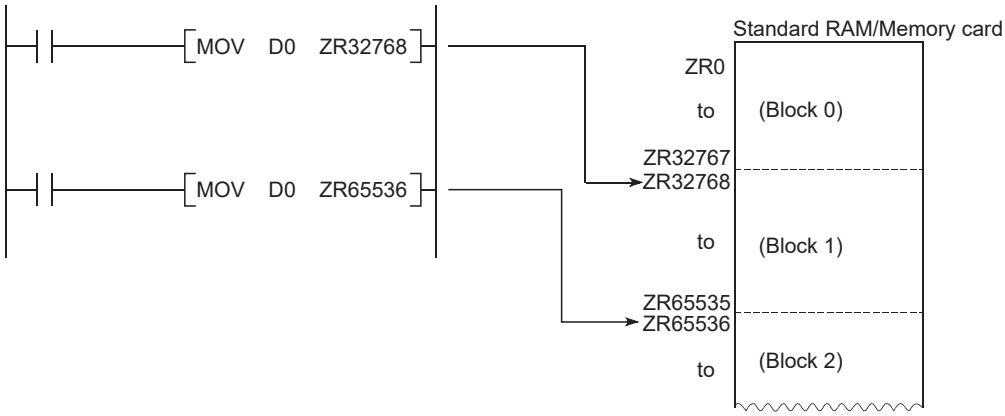
■Block switching method

In the block switching method, specify the number of used file register points in units of 32K points (one block).
For file registers of 32K points or more, specify the file registers by switching the block No. to be used with the RSET instruction. Specify each block as R0 to R32767.



■Serial number access method

In the serial number access method, specify the file registers beyond 32K points with consecutive device numbers.
The file registers of multiple blocks can be used as consecutive file registers.
Use "ZR" as the device name.



Settings and restrictions when refreshing file registers

■Settings

The settings of refresh devices are as follows.

- Refresh settings for CC-Link IE Controller Network (The settings are not available for LCPU.)
- Refresh settings for CC-Link IE Field Network (The settings are not available for Basic model QCPU, High Performance model QCPU, Process CPU, Redundant CPU, Universal model QCPU whose serial number (first five digits) is "12011" or earlier, and LCPU whose serial number (first five digits) is "13011" or earlier.)
- Refresh settings for MELSECNET/H (Cannot be set on LCPU.)
- Refresh settings for CC-Link
- Auto refresh settings for the intelligent function module
- Auto refresh settings for the multiple CPU system (Cannot be set on LCPU.)

■Restrictions

The restrictions when specifying file registers to refresh devices are as follows.

- On QCPU, Refresh cannot be performed correctly if the use of file register which has the same name as the program is specified by the PLC parameter. When the file register which has the same name as the program is used, refresh is performed to the data of the file register having the same name as the program that is set at the last number in the [Program] tab page of PLC parameter. To read/write the refresh data, specify the file register to the refresh device after switching the file register to the corresponding one with the QDRSET instruction.
- Refresh cannot be performed correctly if the file name of file register or the drive number is changed by the QDRSET instruction. (QDRSET instructions are not available with LCPU.) If the file name of file register or the drive number is changed by the QDRSET instruction, link refresh is performed to the data of the setting file at the time of the END instruction execution. To read/write the refresh data, specify the file register of the setting file at the time of the END instruction execution. If the drive number is changed by the QDRSET instruction when "ZR" is specified for the device in the CPU modules other than the Universal model QCPU, an error (LINK PARA ERROR (3101)) occurs. (Note that an error does not occur when "R" is specified for the device.)
- When a block number is switched by the RSET instruction, refresh is performed to the data of the file register (R) in the switched block number. When a block number is switched by the RSET instruction, refresh is performed to the data of the file register (R) in the block number at the time of the END instruction execution. To read/write the refresh data, specify the file register of the block number at the time of the END instruction execution.

Precautions when file registers in the flash memory are used

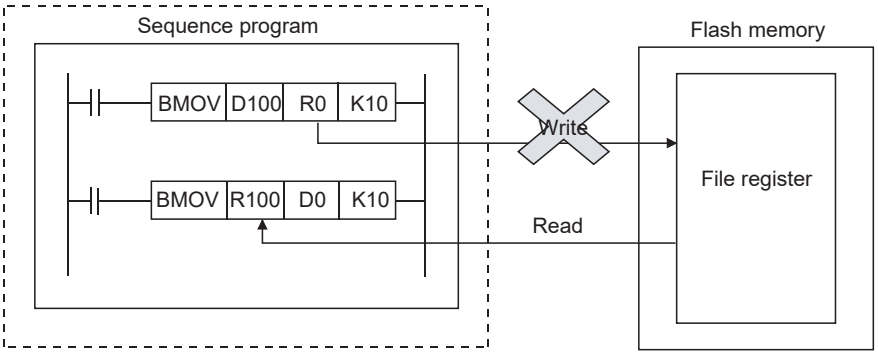
This section explains the precautions for use of the flash memory.

■Applicable flash memory

- Flash card

■Precautions

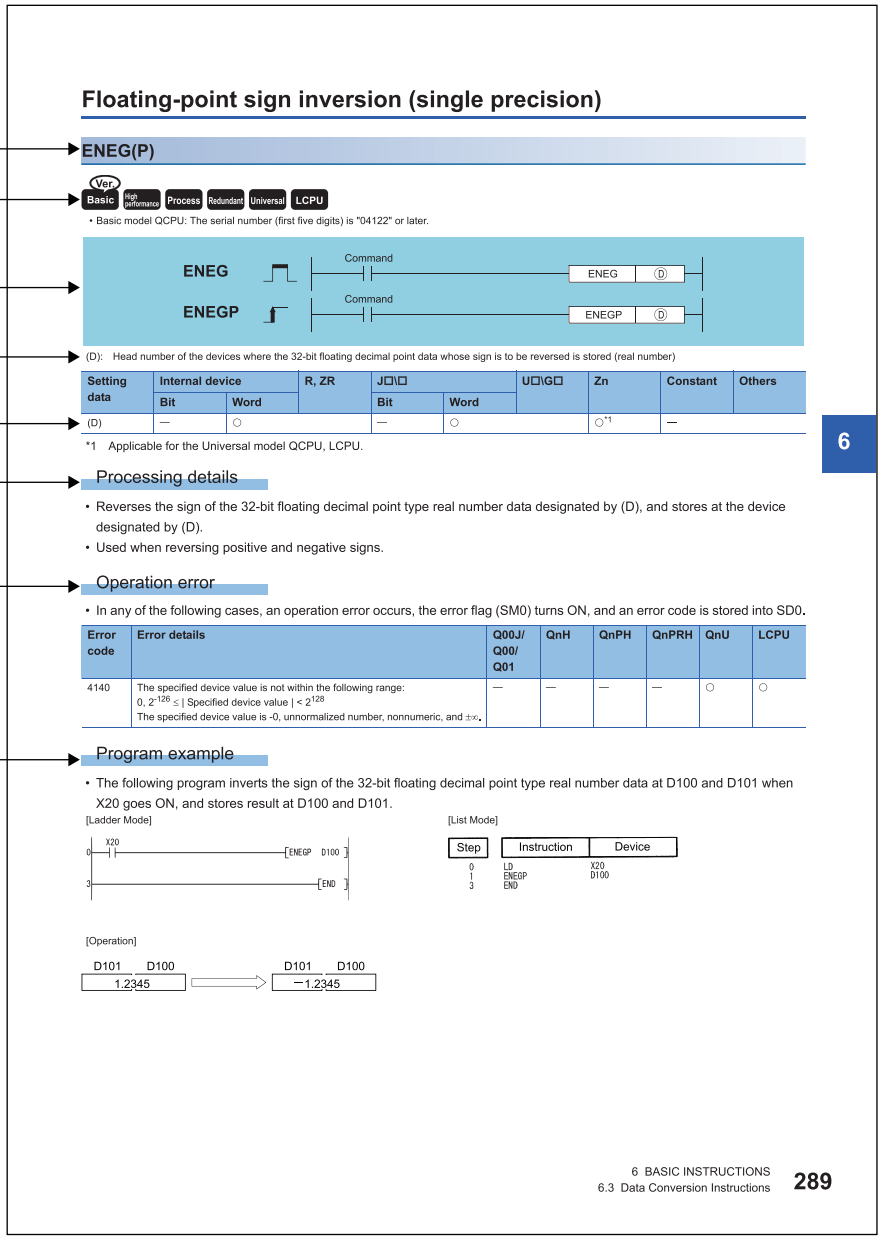
File registers in the flash memory can be only read in a sequence program.
(Write to the flash memory cannot be performed in a sequence program.)



When using the flash memory for the file registers, write data in advance.
Using GX Developer or GX Works2, write data to the flash card.

4 HOW TO READ INSTRUCTIONS

The description of instructions that are contained in the following chapters are presented in the following format.



Floating-point sign inversion (single precision)

❶ ENEG(P)

❷ Ver. Basic | High performance | Process | Redundant | Universal | LCPU

• Basic model QCPU: The serial number (first five digits) is "04122" or later.

❸ ENEG: Command → ENEG (D)
ENEGP: Command → ENEGP (D)

❹ (D): Head number of the devices where the 32-bit floating decimal point data whose sign is to be reversed is stored (real number)

❺

Setting data	Internal device		R, ZR	JO\IO		UO\GO	Zn	Constant	Others
	Bit	Word		Bit	Word				
(D)	—	○		—	○		○*1	—	

*1 Applicable for the Universal model QCPU, LCPU.

6

❻ Processing details

• Reverses the sign of the 32-bit floating decimal point type real number data designated by (D), and stores at the device designated by (D).
• Used when reversing positive and negative signs.

❼ Operation error

• In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4140	The specified device value is not within the following range: 0, $2^{-126} \leq$ | Specified device value | $< 2^{128}$ The specified device value is -0, unnormalized number, nonnumeric, and ±∞.	—	—	—	—	○	○

❽ Program example

• The following program inverts the sign of the 32-bit floating decimal point type real number data at D100 and D101 when X20 goes ON, and stores result at D100 and D101.

[Ladder Mode]

[List Mode]

Step	Instruction	Device
0	LD	X20
1	ENEGP	D100
3	END	

[Operation]

6 BASIC INSTRUCTIONS
6.3 Data Conversion Instructions 289

❶ Code used to write instruction (instruction symbol).
❷ Shows if instructions are enabled or disabled for each CPU module type.

Icon						Description
Basic model QCPU	High Performance model QCPU	Process CPU	Redundant CPU	Universal model QCPU	LCPU	
Basic	High performance	Process	Redundant	Universal	LCPU	A normal icon means the corresponding instruction can be used.
Ver. Basic	Ver. High performance	Ver. Process	Ver. Redundant	Ver. Universal	Ver. LCPU	The icon with Ver. means the instruction can be used with some restrictions (e.g., function version, software version).
Basic ✕	High performance ✕	Process ✕	Redundant ✕	Universal ✕	LCPU ✕	The icon with × (cross) means the corresponding instruction cannot be used.

❸ Indicates ladder mode expressions and execution conditions for instructions.

Execution condition	Non-conditional execution	Executed at ON	Executed at the rising edge	Executed at OFF	Executed at the falling edge
Code recorded on description page	No symbol recorded				

For execution conditions, refer to Page 118 Conditions for Execution of Instructions.

❹ Indicates the data set for each instruction and the data type.

Data type	Description
Bit	Bit data or head number in bit data
BIN 16 bits	BIN 16-bit data or head number in word device
BIN 32 bits	BIN 32-bit data or head number in word device
BIN 64 bits	BIN 64-bit data or head number in word device
BCD 4-digit	4-digit BCD data
BCD 8-digit	8-digit BCD data
Real number	Floating decimal point data
Character string	Character string data
Device name	Device name data

❺ Devices which can be used by the instruction in question are indicated with ○. The types of devices that can be used are as indicated below:

Setting data	Internal device (system, user)		File register R, ZR	Link direct device J□\□*4		Intelligent function module U□\G□	Index register Zn	Constant*5	Others*5
	Bit	Word		Bit	Word				
Usable devices*1	X, Y, M, L, SM, F, B, SB, FX*2, FY*2	T*3, ST*3, C*3, D, W, SD, SW, FD*2, @□	R, ZR	J□\X J□\Y J□\B J□\SB	J□\W J□\SW	U□\G□	Z	K, H, E, $	P, I, J, U, DX, DY, N, BL, TR, BL\S, V

*1 For the description for the individual devices, refer to the User's Manual (Function Explanation, Program Fundamentals) for the CPU module used.

*2 FX and FY can be used only for bit data, and FD only for word data.

*3 When T, ST, and C are used for other than the instructions below, only word data can be used. (Bit data cannot be used.)
[Instructions that can be used with bit data]
LD, LDI, AND, ANI, OR, ORI, LDP, LDF, ANDP, ANDF, ORP, ORF, LDPI, LDFI, ANDPI, ANDFI, ORPI, ORFI, OUT, RST, BKRST

*4 Usable with CC-Link IE Controller Network, CC-Link IE Field Network, MELSECNET/H, and MELSECNET/10.

*5 The "Constant" and "Other" columns describe the devices which can be set.

❻ Indicates the function of the instruction.

❼ Indicates conditions under which error is returned, and error number. See Page 111 Cautions on Programming (Operation Errors) for errors not included here.

❽ Indicates both ladder and list for simple program example. Also indicates the types of individual devices used when the program is executed.

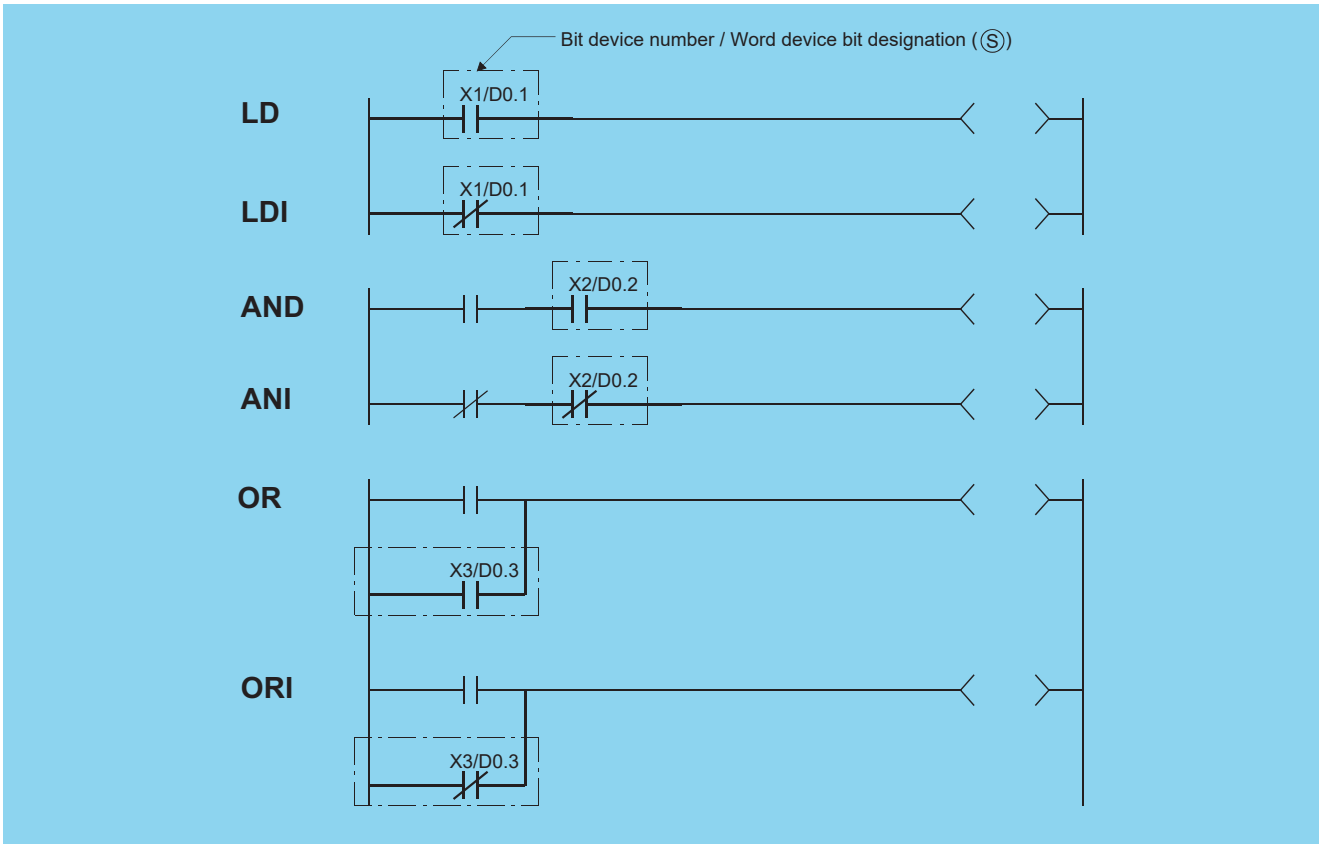
5 SEQUENCE INSTRUCTIONS

5.1 Contact Instructions

Operation start, series connection, parallel connection

LD, LDI, AND, ANI, OR, ORI

Basic | High performance | Process | Redundant | Universal | LCPU



(S): Devices used as contacts (bits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant	Others DX
	Bit	Word		Bit	Word				
(S)	○						—		○

Point

When BL, S, TR, BL\S, or BL\TR is used, refer to the SFC control instructions in the MELSEC-Q/L/QnA Programming Manual (SFC).

Processing details

■LD, LDI

- LD is the A contact operation start instruction, and LDI is the B contact operation start instruction. They read ON/OFF information from the designated device*1, and use that as an operation result.

*1 When a bit designation is made for a word device, the device turns ON or OFF depending on the 1/0 status of the designated bit.

■AND, ANI

- AND is the A contact series connection instruction, and ANI is the B contact series connection instruction. They read the ON/OFF data of the designated bit device*2, perform an AND operation on that data and the operation result to that point, and take this value as the operation result.

*1 When a bit designation is made for a word device, the device turns ON or OFF depending on the 1/0 status of the designated bit.

- There are no restrictions on the use of AND or ANI, but the following applies in the ladder mode of a programming tool:

Item	Description
Write	When AND and ANI are connected in series, a ladder with up to 24 stages can be displayed.
Read	When AND and ANI are connected in series, a ladder with up to 24 stages can be displayed. If the number exceeds 24 stages, up to 24 will be displayed.

■OR, ORI

- OR is the A contact single parallel connection instruction, and ORI is the B contact single parallel connection instruction. They read ON/OFF information from the designated device*1, and perform an OR operation with the operation results to that point, and use the resulting value as the operation result.

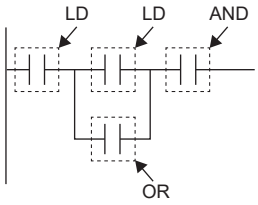
*1 When a bit designation is made for a word device, the device turns ON or OFF depending on the 1/0 status of the designated bit.

- There are no limits on the use of OR or ORI, but the following applies in the ladder mode of a programming tool:

Item	Description
Write	OR and ORI can be used to create connections of up to 23 ladders.
Read	OR and ORI can be used to create connections of up to 23 ladders. The 24th or subsequent ladders cannot be displayed properly.

■Operation using LD, LDI, AND, ANI, OR, and ORI combined

An example of operation using LD, AND, and OR combined is shown below. The same operation is performed by using LDI, ANI, and ORI instead.



Point

Word device bit designations are made in hexadecimal.
Bit b11 of D0 would be D0.0B.
See Page 83 Using bit data for more information on word device bit designation.

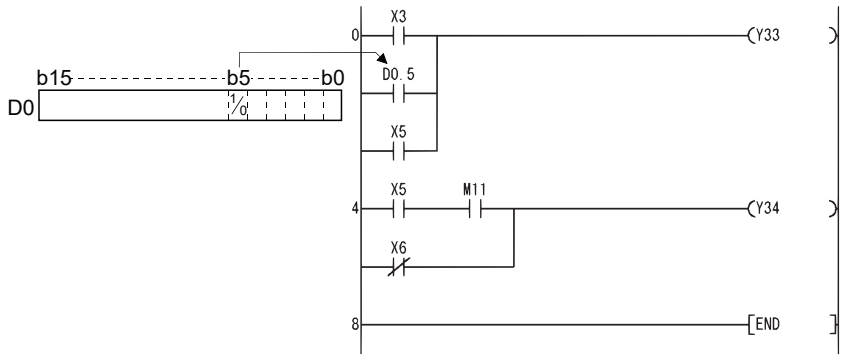
Operation error

- There is no operation error in the LD, LDI, AND, ANI, OR, or ORI instruction.

Program example

• A program using the LD, AND, OR, and ORI instructions.

[Ladder Mode]

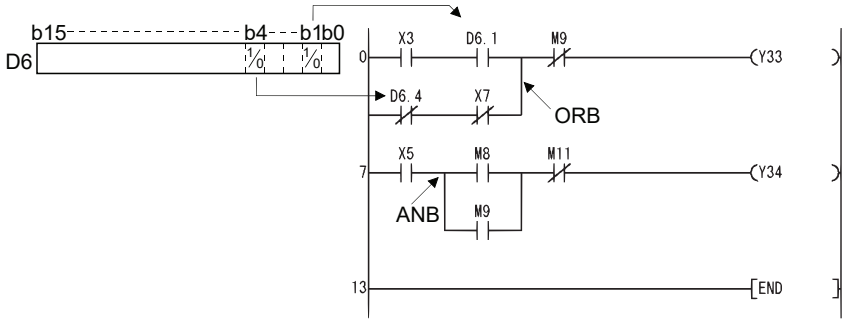


[List Mode]

Step	Instruction	Device	
0	LD	X3	
1	OR	D0.5	⋯Bit designated for word device
2	OR	X5	
3	OUT	Y33	
4	LD	X5	
5	AND	M11	
6	ORI	X6	
7	OUT	Y34	
8	END		

• A program linking contacts using the ANB and ORB instructions.

[Ladder Mode]

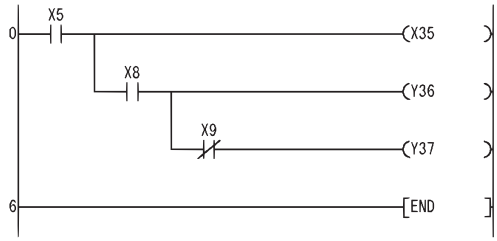


[List Mode]

Step	Instruction	Device	
0	LD	X3	
1	AND	D6.1	⋯Bit designated for word device
2	LDI	D6.4	
3	ANI	X7	
4	ORB		
5	ANI	M9	
6	OUT	Y33	
7	LD	X5	
8	LD	M8	
9	OR	M9	
10	ANB		
11	ANI	M11	
12	OUT	Y34	
13	END		

• A parallel program with the OUT instruction.

[Ladder Mode]



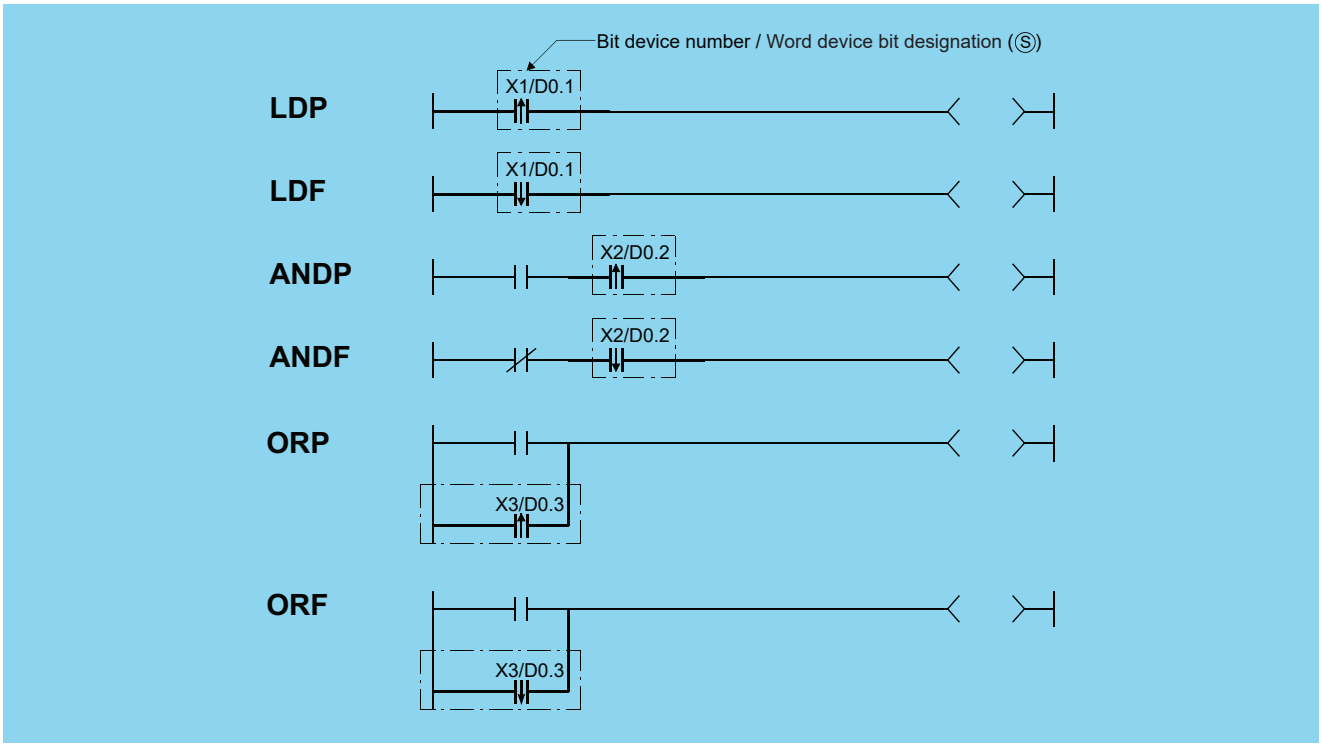
[List Mode]

Step	Instruction	Device
0	LD	X5
1	OUT	X35
2	AND	X8
3	OUT	Y36
4	ANI	X9
5	OUT	Y37
6	END	

Pulse operation start, pulse series connection, pulse parallel connection

LDP, LDF, ANDP, ANDF, ORP, ORF

Basic | High performance | Process | Redundant | Universal | LCPU



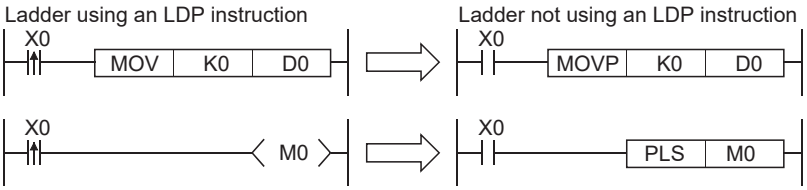
(S): Devices used as contacts (bits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant	Others DX
	Bit	Word		Bit	Word				
(S)	○						—		○

Processing details

■LDP, LDF

- LDP is the rising edge pulse operation start instruction, and is ON only at the rising edge of the designated bit device (when it goes from OFF to ON). If a word device has been designated, it is ON only when the designated bit changes from 0 to 1. In cases where there is only an LDP instruction, it acts identically to instructions for the creation of a pulse that are executed during ON (□P).



- LDF is the falling edge pulse operation start instruction, and is ON only at the falling edge of the designated bit device (when it goes from ON to OFF). If a word device has been designated, it is ON only when the designated bit changes from 1 to 0.

■ANDP, ANDF

- ANDP is a rising edge pulse series connection instruction, and ANDF is a falling edge pulse series connection instruction. They perform an AND operation with the operation result to that point, and take the resulting value as the operation result. The ON/OFF data used by ANDP and ANDF are indicated in the table below:

Device specified in ANDP or ANDF		ANDP state	ANDF state
Bit device	Word device bit designation		
OFF to ON	0 to 1	ON	OFF
OFF	0	OFF	
ON	1		
ON to OFF	1 to 0		ON

■ORP, ORF

- ORP is a rising edge pulse parallel connection instruction, and ORF is a falling edge pulse serial connection instruction. They perform an OR operation with the operation result to that point, and take the resulting value as the operation result. The ON/OFF data used by ORP and ORF are indicated in the table below:

Device specified in ORP or ORF		ORP state	ORF state
Bit device	Word device bit designation		
OFF to ON	0 to 1	ON	OFF
OFF	0	OFF	
ON	1		
ON to OFF	1 to 0		ON

■Operation using LDP, LDF, ANDP, ANDF, ORP, and ORF combined

An example of operation using LDP, LDF, ANDP, ANDF, ORP, and ORF combined is the same as that using LD, AND, and OR. (☞ Page 133 Operation using LD, LDI, AND, ANI, OR, and ORI combined)

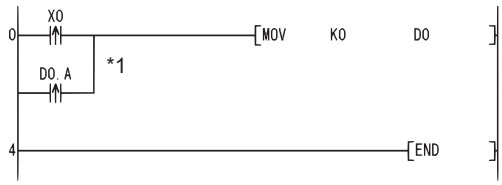
Operation error

- There is no operation error in the LDP, LDF, ANDP, ANDF, ORP, or ORF instruction.

Program example

- The following program executes the MOV instruction at input X0, or at the rising edge of b10 (bit 11) of data register D0.

[Ladder Mode]



[List Mode]

Step	Instruction	Device	
0	LDP	X0	
1	ORP	D0. A	
2	MOV	K0	D0
4	END		

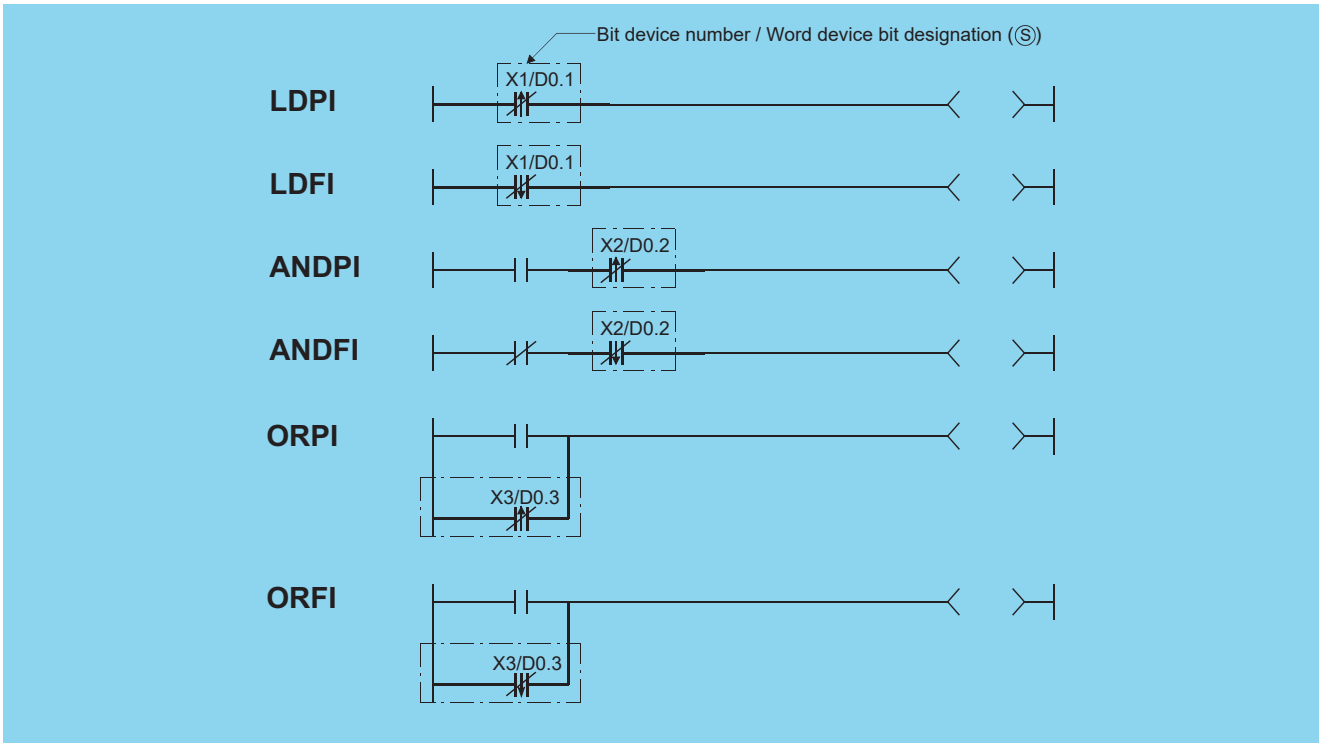
*1 Word device bit designation is performed in hexadecimal.
Bit b10 of D0 will be D0.A.

Pulse NOT operation start, pulse NOT series connection, pulse NOT parallel connection

LDPI, LDFI, ANDPI, ANDFI, ORPI, ORFI

Basic | High performance | Process | Redundant | Universal | LCPU

- QnU(D)(H)CPU, QnUDE(H)CPU: the serial number (first five digits) is "10102" or later
- Q00UJCPU, Q00UCPU, Q01UCPU, QnUDVCPU, QnUDPVCPU: Supported



(S): Devices used as contacts (bits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant	Others DX
	Bit	Word		Bit	Word				
(S)	○						—		○

Processing details

■LDPI, LDFI

- LDPI is the rising edge pulse NOT operation start instruction that is on only at the falling edge of the specified bit device (when the bit device goes from on to off) or when the bit device is on or off. If a word device has been specified, LDPI is on only when the specified bit is 0, 1, or changes from 1 to 0.
- LDFI is the falling edge pulse NOT operation start instruction that is on only at the rising edge of the specified bit device (when the bit device goes from off to on) or when the bit device is on or off. If a word device has been specified, LDPI is on only when the specified bit is 0, 1, or changes from 0 to 1.

Device specified in LDPI or LDFI		LDPI state	LDFI state
Bit device	Word device bit designation		
OFF to ON	0 to 1	OFF	ON
OFF	0	ON	ON
ON	1	ON	ON
ON to OFF	1 to 0	ON	OFF

■ANDPI, ANDFI

- ANDPI is a rising edge pulse NOT series connection, and ANDFI is a falling pulse NOT series connection. ANDPI and ANDFI execute an AND operation with the previous operation result, and take the resulting value as the operation result. The on or off data used by ANDPI and ANDFI are indicated in the table below.

Device specified in ANDPI or ANDFI		LDPI state	LDFI state
Bit device	Word device bit designation		
OFF to ON	0 to 1	OFF	ON
OFF	0	ON	ON
ON	1	ON	ON
ON to OFF	1 to 0	ON	OFF

■ORPI, ORFI

- ORPI is a rising edge pulse NOT parallel connection, and ORFI is a falling pulse NOT parallel connection. ORPI and ORFI execute an OR operation with the previous operation result, and take the resulting value as the operation result. The on or off data used by ORPI and ORFI are indicated in the table below.

Device specified in ORPI or ORFI		ORPI state	ORFI state
Bit device	Word device bit designation		
OFF to ON	0 to 1	OFF	ON
OFF	0	ON	ON
ON	1	ON	ON
ON to OFF	1 to 0	ON	OFF

■Operation using LDPI, LDFI, ANDPI, ANDFI, ORPI, and ORFI combined

An example of operation using LDPI, LDFI, ANDPI, ANDFI, ORPI, and ORFI combined is same as that using LD, AND, and OR. (☞ Page 133 Operation using LD, LDI, AND, ANI, OR, and ORI combined)

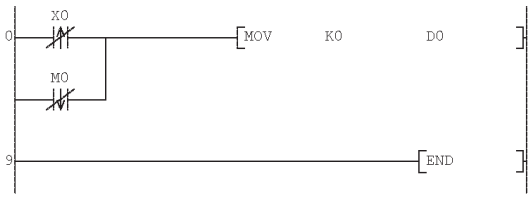
Operation error

- There is no operation error in the LDPI, LDFI, ANDPI, ANDFI, ORPI, or ORFI instruction.

Program example

- The following program stores 0 into D0 when X0 is on, off, or turns from on to off, or M0 is on, off, or turns from off to on.

[Ladder Mode]

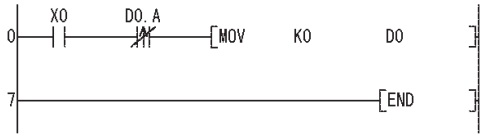


[List Mode]

Step	Instruction	Device	
0	LDPI	X0	
3	ORFI	M0	
7	MOV	K0	D0
9	END		

- The following program stores 0 into D0 when X0 is on and b10 (bit 11) of D0 is on, off, or turns from on to off.

[Ladder Mode]



[List Mode]

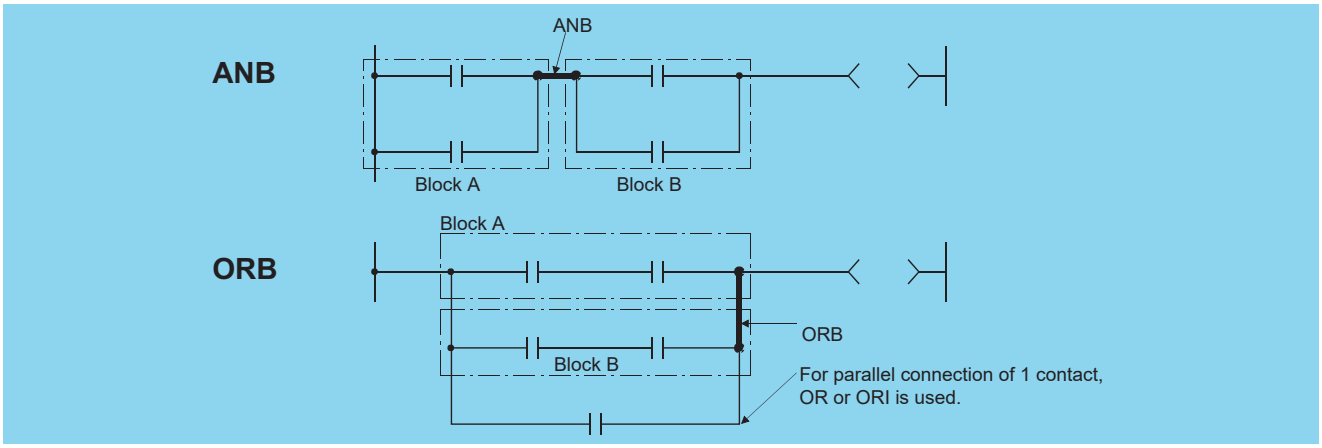
Step	Instruction	Device	
0	LD	X0	
1	ANDPI	D0.A	
5	MOV	K0	D0
7	END		

5.2 Association Instructions

Ladder block series connection, ladder block parallel connection

ANB, ORB

Basic | High performance | Process | Redundant | Universal | LCPU



Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant	Others
	Bit	Word		Bit	Word				
—	—								

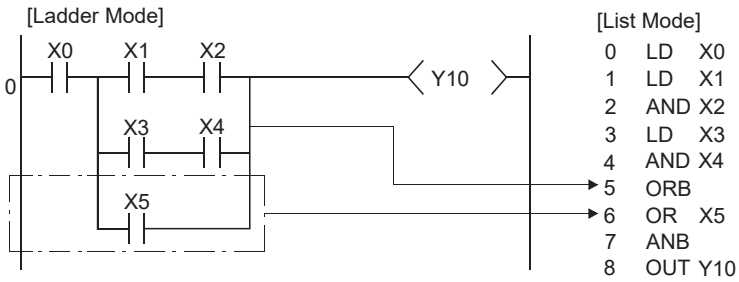
Processing details

■ANB

- Performs an AND operation on block A and block B, and takes the resulting value as the operation result.
- The symbol for ANB is not the contact symbol, but rather is the connection symbol.
- When programming in the list mode, up to 15 ANB instructions (16 blocks) can be written consecutively.

■ORB

- Conducts an OR operation on Block A and Block B, and takes the resulting value as the operation result.
- ORB is used to perform parallel connections for ladder blocks with two or more contacts. For ladder blocks with only one contact, use OR or ORI; there is no need for ORB in such cases.



- The ORB symbol is not the contact symbol, but rather is the connection symbol.
- When programming in the list mode, it is possible to use up to 15 ORB instructions successively (16 blocks).

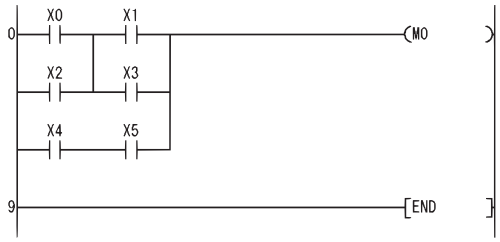
Operation error

- There is no operation error in the ANB or ORB instruction.

Program example

- A program using the ANB and ORB instructions.

[Ladder Mode]



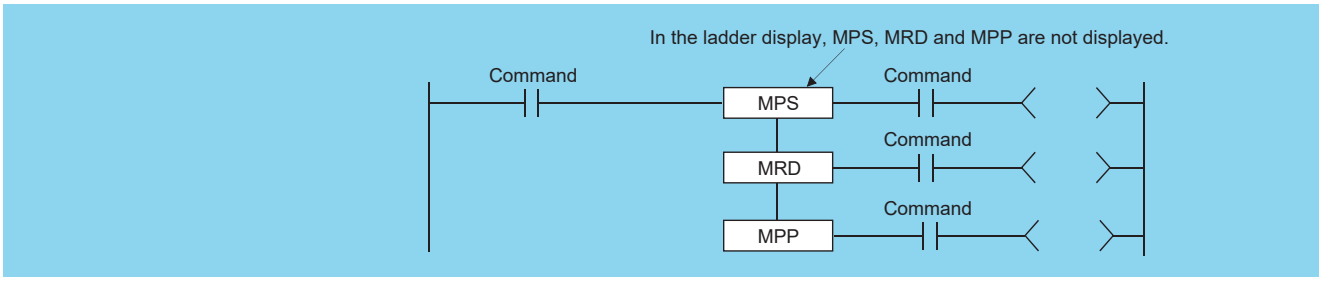
[List Mode]

Step	Instruction	Device
0	LD	X0
1	OR	X2
2	LD	X1
3	OR	X3
4	ANB	
5	LD	X4
6	AND	X5
7	ORB	
8	OUT	M0
9	END	

Operation results push, operation results read, operation results pop

MPS, MRD, MPP

Basic | High performance | Process | Redundant | Universal | LCPU



Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant	Others
	Bit	Word		Bit	Word				
—	—								

Processing details

■MPS

- Stores the memory of the operation result (ON or OFF) immediately prior to the MPS instruction.
- Up to 16 MPS instructions can be used successively. If the MPP instruction is used during this process, the number of uses calculated for the MPS instruction will be decremented by one.

■MRD

- Reads the operation result stored for the MPS instruction, and uses that result to perform the operation in the next step.

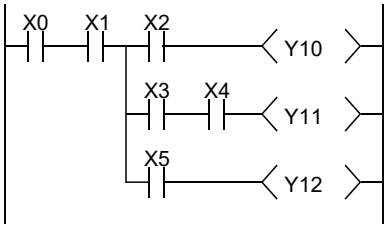
■MPP

- Reads the operation result stored for the MPS instruction, and uses that result to perform the operation in the next step.
- Clears the operation results stored by the MPS instruction.
- Subtracts 1 from the number of MPS instruction times of use.

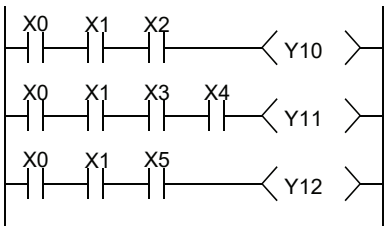
Point

The following shows ladders both using and not using the MPS, MRD, and MPP instructions.

• Ladder Using the MPS, MRD and MPP Instructions



• Ladder not Using MPS, MRD, and MPP Instructions



The MPS and MPP instructions must be used the same number of times.
Failure to observe this will not correctly display the ladder in the ladder mode of the programming tool.

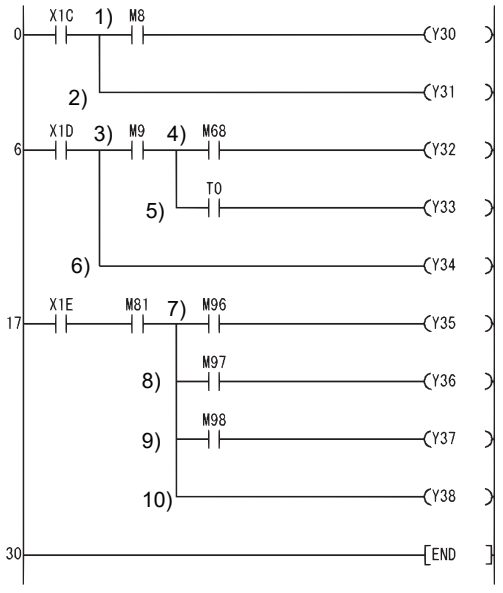
Operation error

- There is no operation error in the MPS, MRD, or MPP instruction.

Program example

- A program using the MPS, MRD, and MPP instructions.

[Ladder Mode]

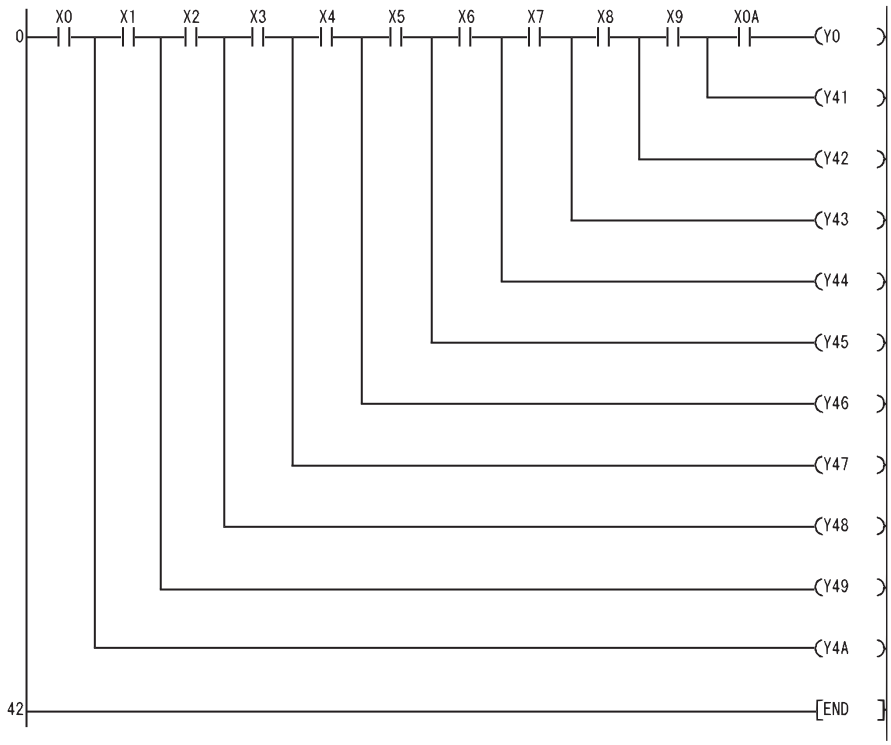


[List Mode]

Step		Instruction	Device
	0	LD	X1C
1)	1	MPS	
	2	AND	M8
	3	OUT	Y30
2)	4	MPP	
	5	OUT	Y31
	6	LD	X1D
3)	7	MPS	
	8	AND	M9
4)	9	MPS	
	10	AND	M68
	11	OUT	Y32
5)	12	MPP	
	13	AND	T0
	14	OUT	Y33
6)	15	MPP	
	16	OUT	Y34
	17	LD	X1E
	18	AND	M81
7)	19	MPS	
	20	AND	M96
	21	OUT	Y35
8)	22	MRD	
	23	AND	M97
	24	OUT	Y36
9)	25	MRD	
	26	AND	M98
	27	OUT	Y37
10)	28	MPP	
	29	OUT	Y38
	30	END	

• A program using the MPS and MPP instructions successively.

[Ladder Mode]



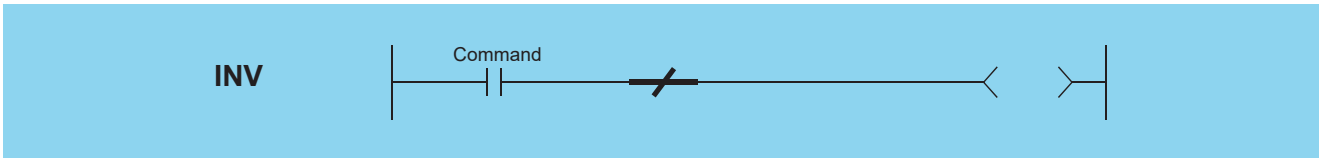
[List Mode]

Step	Instruction	Device
0	LD	X0
1	MPS	
2	AND	X1
3	MPS	
4	AND	X2
5	MPS	
6	AND	X3
7	MPS	
8	AND	X4
9	MPS	
10	AND	X5
11	MPS	
12	AND	X6
13	MPS	
14	AND	X7
15	MPS	
16	AND	X8
17	MPS	
18	AND	X9
19	MPS	
20	AND	X0A
21	OUT	Y0
22	MPP	
23	OUT	Y41
24	MPP	
25	OUT	Y42
26	MPP	
27	OUT	Y43
28	MPP	
29	OUT	Y44
30	MPP	
31	OUT	Y45
32	MPP	
33	OUT	Y46
34	MPP	
35	OUT	Y47
36	MPP	
37	OUT	Y48
38	MPP	
39	OUT	Y49
40	MPP	
41	OUT	Y4A
42	END	

Operation results inversion

INV

Basic | High performance | Process | Redundant | Universal | LCPU



Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant	Others
	Bit	Word		Bit	Word				
—	—								

Processing details

- Inverts the operation result immediately prior to the INV instruction.

Operation result immediately prior to the INV instruction	Operation result following the execution of the INV instruction
OFF	ON
ON	OFF

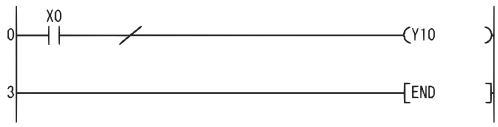
Operation error

- There is no operation error in the INV instruction.

Program example

- A program which inverts the X0 ON/OFF data, and outputs from Y10.

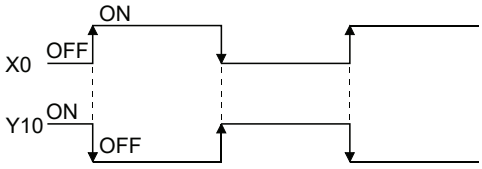
[Ladder Mode]



[List Mode]

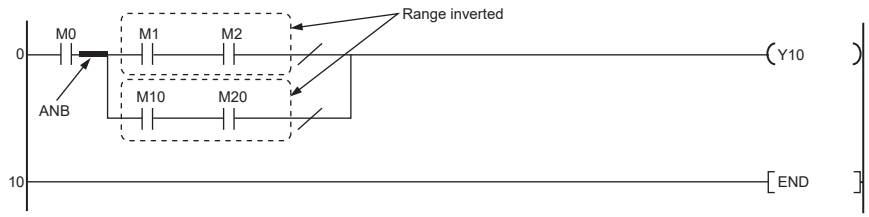
Step	Instruction	Device
0	LD	X0
1	INV	
2	OUT	Y10
3	END	

[Timing Chart]



Point

- The INV instruction operates based on the results of calculation made until the INV instruction is given. Accordingly, use it in the same position as that of the AND instruction. The INV instruction cannot be used at the LD and OR positions.
- When a ladder block is used, the operation result is inverted within the range of the ladder block. To operate a ladder using the INV instruction in combination with the ANB instruction, pay attention to the range that will be inverted.

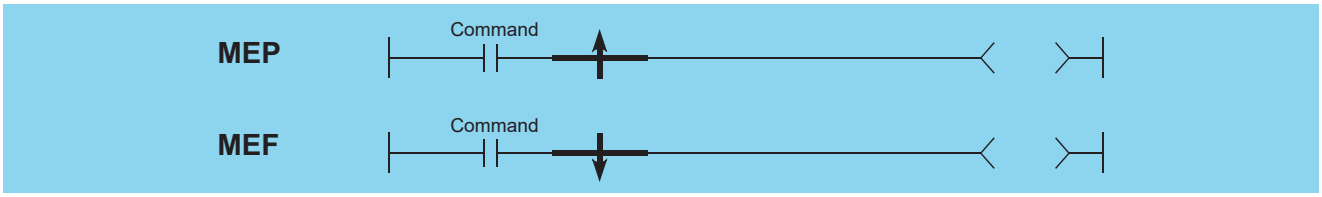


For details of the ANB instruction, refer to Page 139 Ladder block series connection, ladder block parallel connection.

Operation results conversion

MEP, MEF

Basic | High performance | Process | Redundant | Universal | LCPU



Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant	Others
	Bit	Word		Bit	Word				
—	—								

Processing details

■MEP

- If operation results up to the MEP instruction are rising edge (from OFF to ON), goes ON (continuity status). If operation results up to the MEP instruction are anything other than rising edge, goes OFF (non-continuity status).
- Use of the MEP instruction simplifies pulse conversion processing when multiple contacts are connected in series.

■MEF

- If operation results up to the MEF instruction are falling edge (from ON to OFF), goes ON (continuity status). If operation results up to the MEF instruction are anything other than falling edge, goes OFF (non-continuity status).
- Use of the MEF instruction simplifies pulse conversion processing when multiple contacts are connected in series.

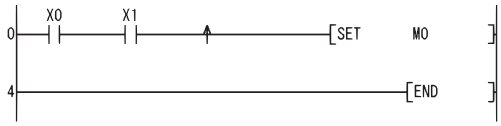
Operation error

- There is no operation error in the MEP or MEF instruction.

Program example

- A program which performs pulse conversion to the operation results of X0 and X1

[Ladder Mode]

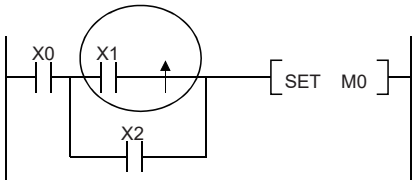


[List Mode]

Step	Instruction	Device
0	LD	X0
1	AND	X1
2	MEP	
3	SET	M0
4	END	

Point

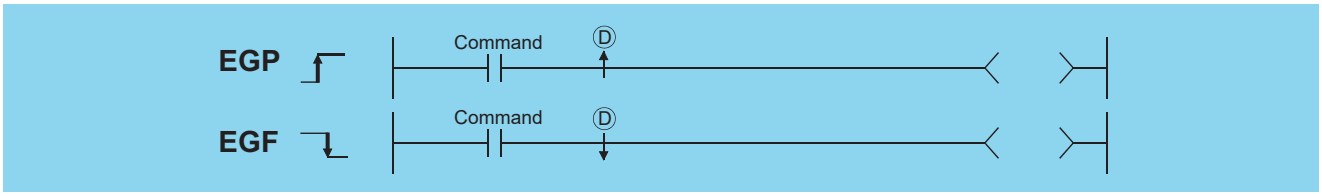
- The MEP and MEF instructions will occasionally not function properly when pulse conversion is conducted for a contact that has been indexed by a subroutine program or by the FOR to NEXT instructions. If pulse conversion is to be conducted for a contact that has been indexed by a subroutine program or by the FOR to NEXT instructions, use the EGP/EGF instructions.
- The MEP or MEF instruction operates based on the operation result performed starting from the LD instruction immediately before the MEP or MEF instruction to immediately before the MEP or MEF instruction. Therefore, use them at the same position as that of the AND instruction. The MEP and MEF instructions cannot be used at the LD or OR position.
- Do not use the MEP or MEF instructions at the circled location in a ladder block shown below. In such a ladder block, the SET instruction may not be executed depending on the timing of turning ON of X0 and X1.



Pulse conversion of edge relay operation results

EGP, EGF

Basic | High performance | Process | Redundant | Universal | LCPU



(D): Edge relay number where operation results are stored (bits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant	Others	
	Bit	Word		Bit	Word				V	
(D)	—								○	

Processing details

■EGP

- Operation results up to the EGP instruction are stored in memory by the edge relay (V).
- Goes ON (continuity status) at the rising edge (OFF to ON) of the operation result up to the EGP instruction. If the operation result up to the EGP instruction is other than a rising edge (i.e., from ON to ON, ON to OFF, or OFF to OFF), it goes OFF (non-continuity status).
- The EGP instruction is used for subroutine programs, and for conducting pulse operations for programs designated by indexing between the FOR and NEXT instructions.
- The EGP instruction can be used like an AND instruction.

■EGF

- Operation results up to the EGF instruction are stored in memory by the edge relay (V).
- Goes ON at the falling edge (from ON to OFF) of the operation result up to the EGF instruction. If the operation result up to the EGF instruction is other than a falling edge (i.e., from OFF to ON, ON to ON, or OFF to OFF), it goes OFF (non-continuity status).
- The EGF instruction is used for subroutine programs, and for conducting pulse operations for programs designated by indexing between the FOR and NEXT instructions.
- The EGF instruction can be used like an AND instruction.

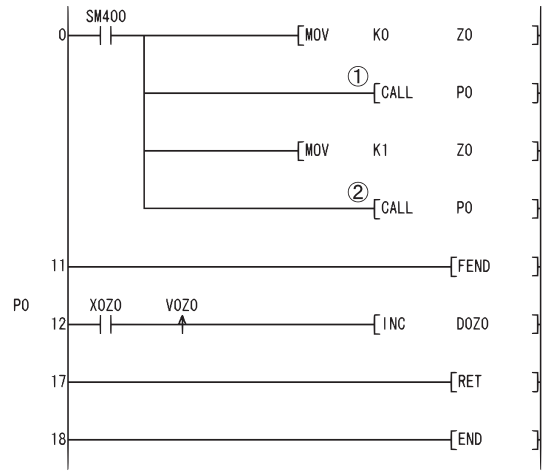
Operation error

- There is no operation error in the EGP or EGF instruction.

Program example

• A program using the EGP instruction in the subroutine program using the EGD instruction

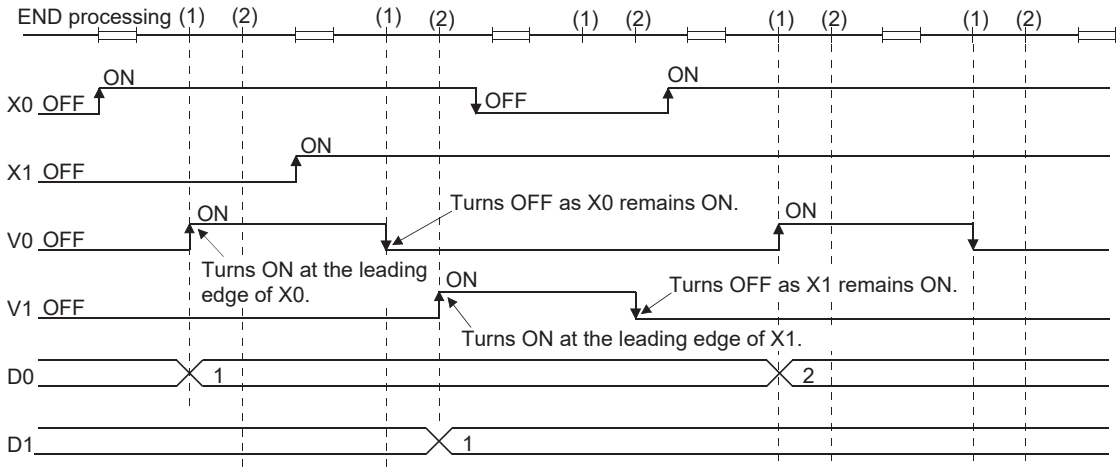
[Ladder Mode]



[List Mode]

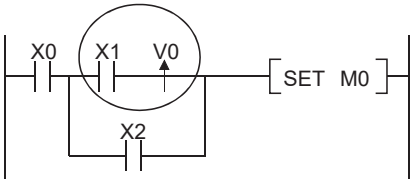
Step	Instruction	Device	
0	LD	SM400	
1	MOV	K0	Z0
4	CALL	P0	
6	MOV	K1	Z0
9	CALL	P0	
11	FEND		
12	P0		
13	LD	X0Z0	
14	EGP	V0Z0	
15	INC	D0Z0	
17	RET		
18	END		

[Operation]



Point

- The EGP or EGF instruction operates based on the operation result performed starting from the LD instruction immediately before the EGP or EGF instruction to immediately before the EGP or EGF instruction. Therefore, use them at the same position as that of the AND instruction. (Refer to ☞ Page 132 Operation start, series connection, parallel connection.) The EGP and EGF instruction cannot be used at the position of the LD or OR instruction.
- Do not use the EGP or EGF instructions at the circled location in a ladder block shown below. In such a ladder block, the SET instruction may not be executed depending on the timing of turning on of X0 and X1. In addition, even when the instruction is in the non-continuity status, the monitor of the engineering tool may show the ON of V0.

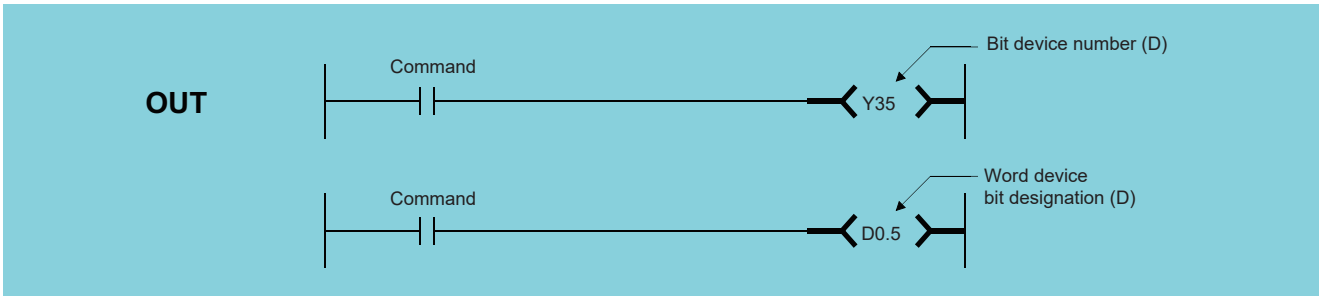


5.3 Output Instructions

Out (excluding timers, counters, and annunciators)

OUT

Basic | High performance | Process | Redundant | Universal | LCPU



(D): Number of the device to be turned ON and OFF (bits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant	Others DY
	Bit	Word		Bit	Word				
(D)	○ (Other than T, C, and F)	○					—		○

Processing details

- Operation results up to the OUT instruction are output to the designated device.

Item	Operation results	Coil
When using bit devices	OFF	OFF
	ON	ON

Item	Operation results	Bit designated
When bit designation has been made for word device	OFF	0
	ON	1

- The number of basic steps for the OUT instructions is as follows:

Item		Number of steps
When using internal device or file register (R)		1
When using direct access output (DY)		2
When using serial number access format file register	Universal model QCPU and LCPU	2
	Basic model QCPU, High Performance model QCPU, Process CPU, and Redundant CPU	3
Devices other than above		3

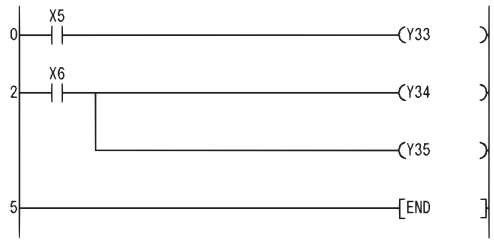
Operation error

- There is no operation error in the OUT instruction.

Program example

- When using bit devices

[Ladder Mode]

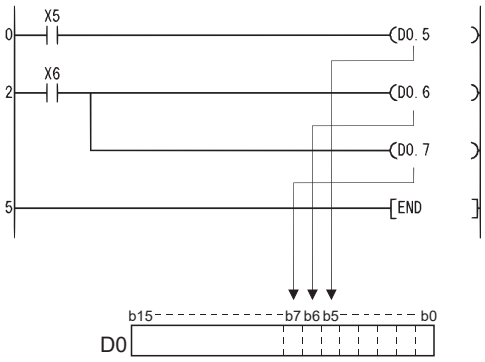


[List Mode]

Step	Instruction	Device
0	LD	X5
1	OUT	Y33
2	LD	X6
3	OUT	Y34
4	OUT	Y35
5	END	

- When bit designation has been made for word device

[Ladder Mode]



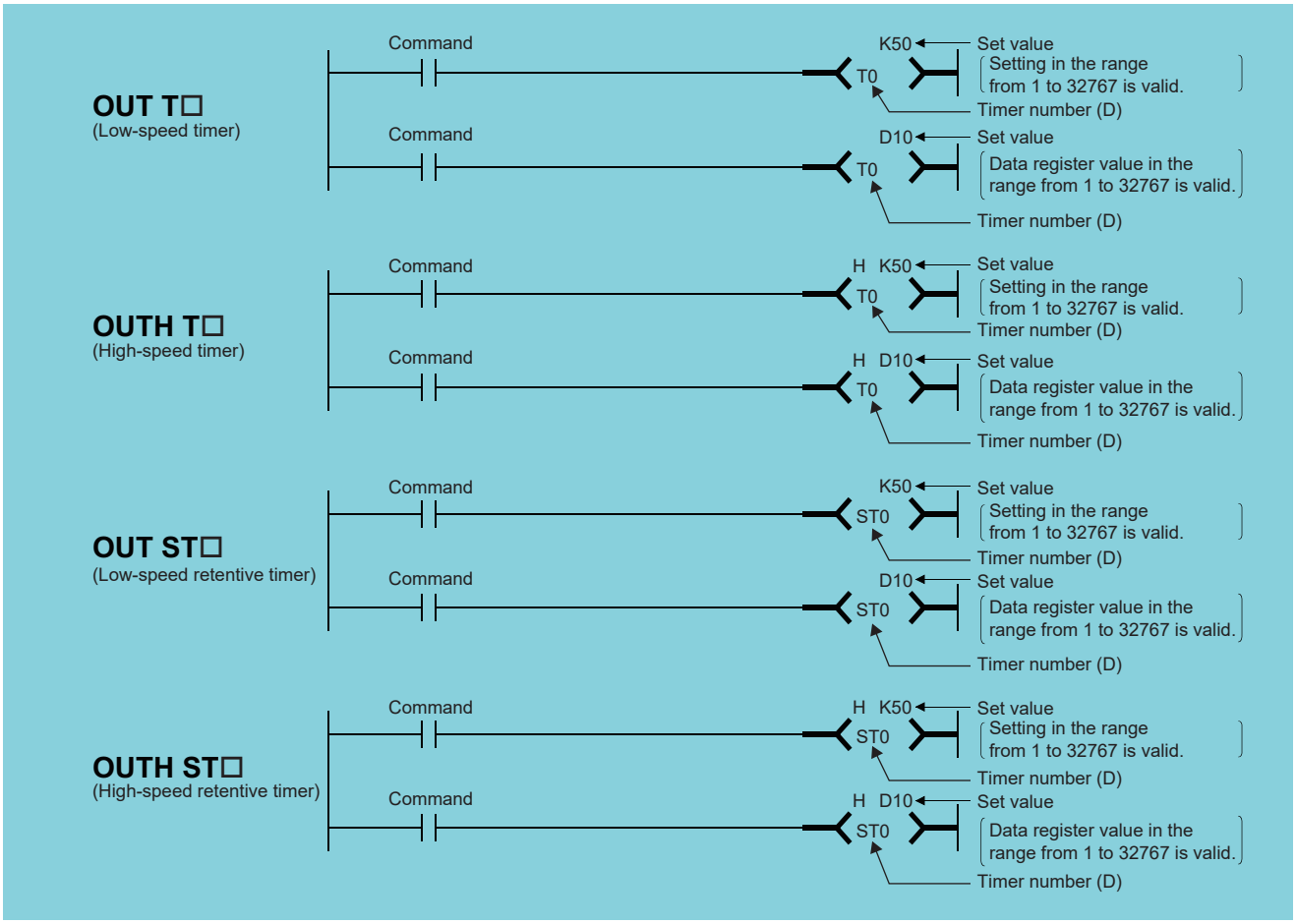
[List Mode]

Step	Instruction	Device
0	LD	X5
1	OUT	D0.5
2	LD	X6
3	OUT	D0.6
4	OUT	D0.7
5	END	

Low-speed timer, high-speed timer, low-speed retentive timer, high-speed retentive timer

OUT T, OUTH T, OUT ST, OUTH ST

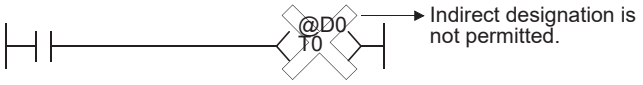
Basic | High performance | Process | Redundant | Universal | LCPU



(D): Timer number (bit)
Setting value Value set for timer (BIN 16 bits*1)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K	Others
	Bit	Word		Bit	Word				
(D)	○ (Only T, ST)	—	—	—	—		—	—	—
Set value	—	○ (Other than T, ST, C)	○	—	○		—	○*2	—

*1 The value setting for the timer cannot be designated indirectly.



See Page 106 Indirect Specification for further information on indirect designation.

*2 Timer values can be set only as a decimal constant (K). Hexadecimal constants (H) and real numbers cannot be used for timer settings.

Processing details

- When the operation results up to the OUT instruction are ON, the timer coil goes ON and the timer counts up to the value that has been set; when the time up status (total numeric value is equal to or greater than the setting value), the contact responds as follows:
  - A Contact: Continuity
  - B Contact: Non-continuity
- The contact responds as follows when the operation result up to the OUT instruction is a change from ON to OFF:

Type of timer	Timer coil	Present value of timer	Prior to time up		After time up	
			A contact	B contact	A contact	B contact
Low speed timer	OFF	0	Non-continuity	Continuity	Non-continuity	Continuity
High speed timer						
Low-speed retentive timer	OFF	Maintains the present value	Non-continuity	Continuity	Continuity	Non-continuity
High-speed retentive timer						

- To clear the present value of a retentive timer and turn the contact OFF after time up, use the RST instruction.
- A negative number (-32768 to -1) cannot be set as the setting value for the timer. *3 If the setting value is 0, the timer will time out when the OUT instruction is executed.

*3 When specifying a setting value for the timer using a word device (D, W, R, ZR, J□\□, or U□\G□), whether the value is in the setting range is not checked. Check the value in the user program so that a negative number is not set.

- The following processing is conducted when the OUT instruction is executed:
  - OUT T□ coil turned ON or OFF
  - OUT T□ contact turned ON or OFF
  - OUT T□ present value updated

In cases where a JMP instruction or the like is used to jump to an OUT T□ instruction while the OUT T□ instruction is ON, no present value update or contact ON/OFF operation is conducted.

Also, if the same OUT T□ instruction is conducted two or more times during the same scan, the present value of the number of repetitions executed will be updated.

- Indexing for timer coils or contacts can be conducted only by Z0 or Z1. (QnUDVCPU and QnUDPVCPU are excluded.)
- Timer setting value has no limitation for indexing.
- The number of basic steps of the OUT T□ instruction is 4.

Point

- Time limit of the timer is set in the PLC system of the PLC parameter dialog box.

[Basic model QCPU, High Performance model QCPU, Process CPU, and Redundant CPU]
Low speed timer/Low-speed retentive timer: 1ms to 1000ms (default: 100ms, setting unit: 1ms)
High speed timer/High-speed retentive timer: 0.1ms to 100.0ms (default: 10.0ms, setting unit: 0.1ms)
[Universal model QCPU and LCPU]
Low speed timer/Low-speed retentive timer: 1ms to 1000ms (default: 100ms, setting unit: 1ms)
High speed timer/High-speed retentive timer: 0.01ms to 100.0ms (default: 10.0ms, setting unit: 0.01ms)

- For information on timer counting methods, refer to the User's Manual (Function Explanation, Program Fundamentals) for the CPU module used.

Operation error

- There is no operation error in the OUT instruction.

Precautions

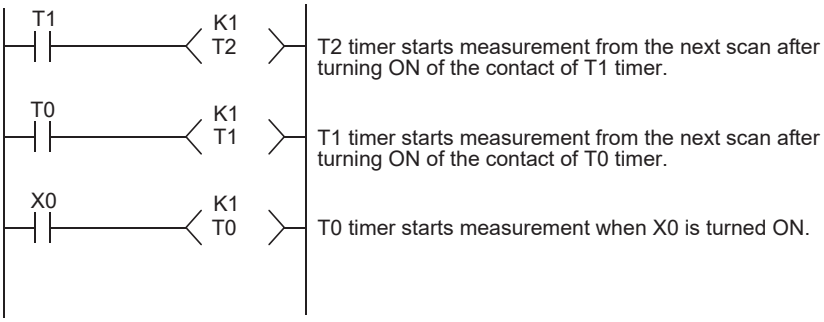
When creating a program in which the operation of the timer contact triggers the operation of another timer, create the program for the timer that operates later first.

In the following cases, all timers go ON at the same scan if the program is created in the order the timers operate.

- If the set value is smaller than a scan time.
- If "1" is set

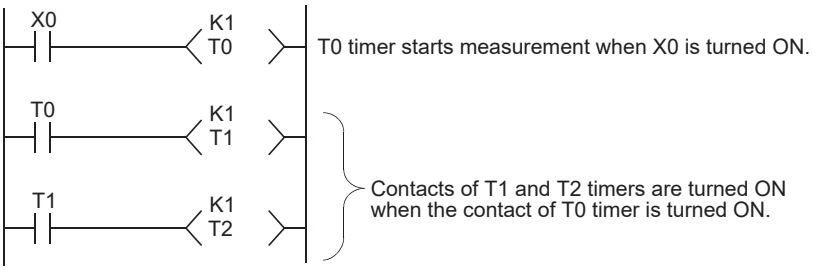
Ex.

For timers T0 to T2, the program is created in the order the timer operates later.



Ex.

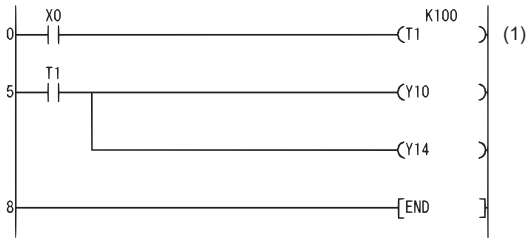
For timers T0 to T2, the program is created in the order of timer operation.



Program example

• The following program turns Y10 and Y14 ON 10 seconds after X0 has gone ON.

[Ladder Mode]



[List Mode]

Step	Instruction	Device	
0	LD	X0	
1	OUT	T1	K100
5	LD	T1	
6	OUT	Y10	
7	OUT	Y14	
8	END		

(1) The setting value of the low-speed timer indicates its default time limit (100ms).

• The following program uses the BCD data at X10 to X1F as the timer's set value.

[Ladder Mode]

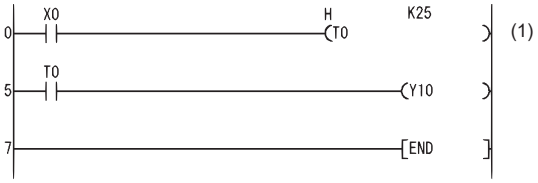
Converts the BCD data at X10 to X1F to BIN and stores the converted value at D10.
When X2 is turned ON, T2 starts measurement using the data stored in D10 as the set value.
Y15 goes ON at the count-up of T2.

[List Mode]

Step	Instruction	Device	
0	LD	X0	
1	BINP	K4X10	D10
4	LD	X2	
5	OUT	T2	D10
9	LD	T2	
10	OUT	Y15	
11	END		

• The following program turns Y10 ON 250ms after X0 goes ON.

[Ladder Mode]



[List Mode]

Step	Instruction	Device	
0	LD	X0	
1	OUTH	T0	K25
5	LD	T0	
6	OUT	Y10	
7	END		

(1) The setting value of the high-speed timer indicates its default time limit (10ms).

Counter

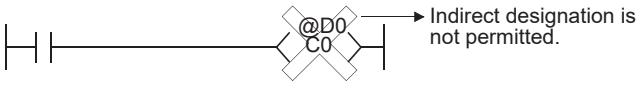
OUT C

Basic | High performance | Process | Redundant | Universal | LCPU

(D): Counter number (bits)
Setting value Counter setting value (BIN 16 bits*1)

Setting data	Internal device		R, ZR	J☐\☐		U☐\G☐	Zn	Constant K	Others
	Bit	Word		Bit	Word				
(D)	○ (Only C)	—	—	—	—		—	—	—
Set value	—	○ (Other than T, ST, C)	○	—	○		—	○*2	—

*1 Counter value cannot be set by indirect designation.



See Page 106 Indirect Specification for further information on indirect designation.

*2 Counter value can be set only with a decimal constant (K). Hexadecimal constants (H) and real numbers cannot be used for timer settings.

Processing details

- When the operation results up to the OUT instruction change from OFF to ON, 1 is added to the present value (count value) and the count up status (present value ≥ set value), and the contacts respond as follows:
  - A Contact: Continuity
  - B Contact: Non-continuity
- No count is conducted with the operation results at ON. (There is no need to perform pulse conversion on count input.)
- After the count up status is reached, there is no change in the count value or the contacts until the RST instruction is executed.
- A negative number (-32768 to -1) cannot be set as the setting value for the timer. If the set value is 0, the processing is identical to that which takes place for 1.
- Only Z0 or Z1 can be used for indexing for the counter coil and contact. (QnUDVCPU and QnUDPVCPU are excluded.) Counter setting value has no limitation for indexing.
- The number of basic steps of the OUT C☐ instruction is 4.

Point

For counter counting methods, refer to the User's Manual (Function Explanation, Program Fundamentals) for the CPU module used.

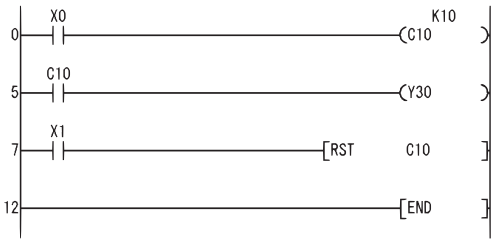
Operation error

- There is no operation error in the OUT instruction.

Program example

• The following program turns Y30 ON after X0 has gone ON 10 times, and resets the counter when X1 goes ON.

[Ladder Mode]

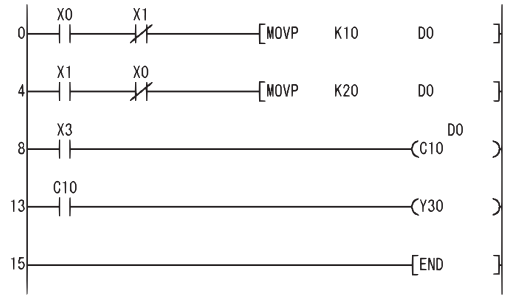


[List Mode]

Step	Instruction	Device	
0	LD	X0	
1	OUT	C10	K10
5	LD	C10	
6	OUT	Y30	
7	LD	X1	
8	RST	C10	
12	END		

• The following program sets the value for C10 at 10 when X0 goes ON, and at 20 when X1 goes ON.

[Ladder Mode]



Stores 10 at D0 when X0 goes ON.

Stores 20 at D0 when X1 goes ON.

C10 executes counting using the data stored in D0 as the set value.

Y30 goes ON at the count-up of C10.

[List Mode]

Step	Instruction	Device	
0	LD	X0	
1	ANI	X1	
2	MOVP	K10	D0
4	LD	X1	
5	ANI	X0	
6	MOVP	K20	D0
8	LD	X3	
9	OUT	C10	D0
13	LD	C10	
14	OUT	Y30	
15	END		

Annunciator output

OUT F

Basic | High performance | Process | Redundant | Universal | LCPU

(D): Number of the annunciator to be turned ON (bits)

Setting data	Internal device		R, ZR	J☐\☐		U☐\G☐	Zn	Constant	Others
	Bit	Word		Bit	Word				
(D)	○ (Only F)	—							

Processing details

- Operation results up to the OUT instruction are output to the designated annunciator.
- The following responses occur when an annunciator (F) is turned ON.
  - The "USER"/"ERR." LED goes ON.
  - The annunciator numbers which are ON (F numbers) are stored in special registers (SD64 to SD79).
  - The value of SD63 is incremented by 1.
- If the value of SD63 is 16 (which happens when 16 annunciators are already ON), even if a new annunciator is turned ON, its number will not be stored at SD64 to SD79.
- The following responses occur when the annunciator is turned OFF by the OUT instruction.
  - The coil goes OFF, but there are no changes in the status of the "USER" / "ERR." LED and the contents of the values stored in SD63 to SD79.
  - Use the RST F☐ instruction to make the "USER"/"ERR." LED go OFF as well as to delete the annunciator which was turned OFF by the OUT F☐ instruction from SD63 to SD79.
- The number of basic steps of the OUT F☐ instruction is 2.
- For the LED which turns on when an annunciator is turned on, refer to the following table depending on the CPU module used.

CPU module	LED which turns on
High Performance model QCPU, Process CPU, Redundant CPU, Universal model QCPU, LCPU	"USER" LED
Basic model QCPU	"ERR." LED

Point

For details of annunciators, refer to the User's Manual (Function Explanation, Program Fundamentals) for the CPU module used.

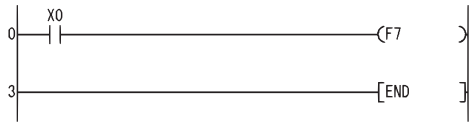
Operation error

- There is no operation error in the OUT instruction.

Program example

• The following program turns F7 ON when X0 goes ON, and stores the value 7 from SD64 to SD79.

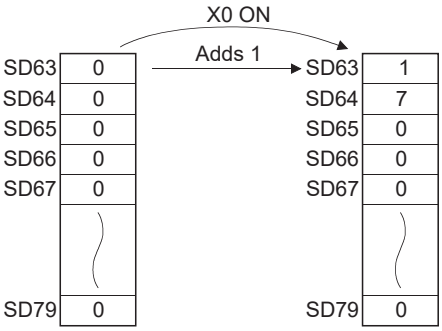
[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	X0
1	OUT	F7
3	END	

[Operation]



Setting devices (excluding annunciators)

SET

Basic | High performance | Process | Redundant | Universal | LCPU

(D): Bit device number to be set (ON)/Word device bit designation (bits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant	Others DY
	Bit	Word		Bit	Word				
(D)	○	○ (Other than T, ST, C)		○			—		○

Point

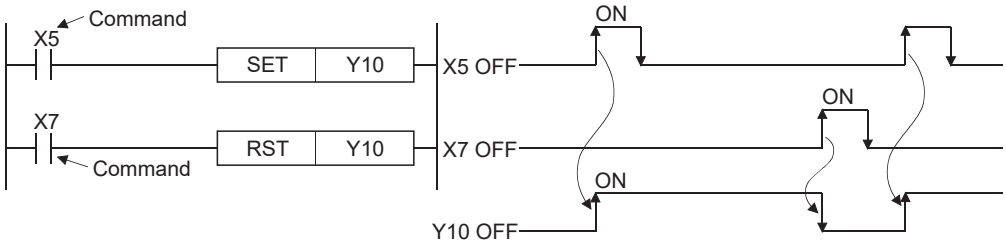
When BL, S, TR, BL\S, or BL\TR is used, refer to the SFC control instructions in the MELSEC-Q/L/QnA Programming Manual (SFC).
When F is used, refer to Page 164 Setting annunciators, resetting annunciators.

Processing details

- When the execution command is turned ON, the status of the designated devices becomes as shown below:

Device	Device status
Bit device	Coils and contacts turned ON
When bit designation has been made for word device	Designation bit set at 1

- Devices turned ON by the instruction remain ON when the same command is turned OFF. Devices turned ON by the SET instruction can be turned OFF by the RST instruction.



- When the execution command is OFF, the status of devices does not change.
- The number of basic steps for the SET instruction is as follows:

Item		Number of basic steps
When internal device or file register (R0 to R32767) are in use		1
When direct access output (DY) or SFC program device (BL) are in use		2
When using serial number access format file register	Universal model QCPU and LCPU	2
	Basic model QCPU, High Performance model QCPU, Process CPU, and Redundant CPU	3
When some other device is in use		3

Point

When using X as a device, use the device numbers that are not used for the actual input. If the same number is used for the actual input device and input X, the data of the actual input will be written over the input X specified in the SET instruction.

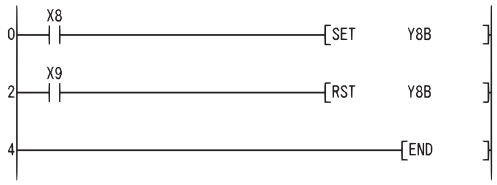
Operation error

- There is no operation error in the SET instruction.

Program example

• The following program sets Y8B (ON) when X8 goes ON, and resets Y8B (OFF) when X9 goes ON.

[Ladder Mode]

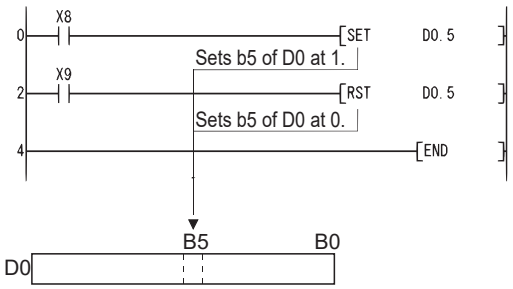


[List Mode]

Step	Instruction	Device
0	LD	X8
1	SET	Y8B
2	LD	X9
3	RST	Y8B
4	END	

• The following program sets the value of D0 bit 5 (b5) to 1 when X8 goes ON, and set the bit value to 0 when X9 goes ON.

[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	X8
1	SET	D0.5
2	LD	X9
3	RST	D0.5
4	END	

Resetting devices (excluding annunciators)

RST

Basic | High performance | Process | Redundant | Universal | LCPU

(D): Bit device number to be reset/ Word device bit designation (bits) Word device number to be reset (BIN 16 bits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant	Others DY
	Bit	Word		Bit	Word				
(D)	○							—	○

Point

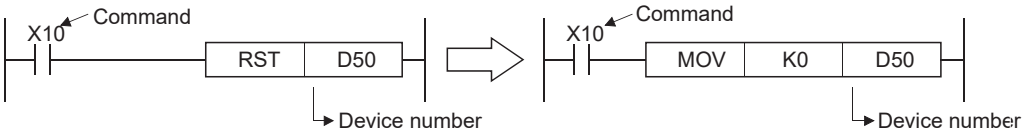
When BL, S, TR, BL\S, or BL\TR is used, refer to the SFC control instructions in the MELSEC-Q/L/QnA Programming Manual (SFC).
When F is used, refer to Page 164 Setting annunciators, resetting annunciators.

Processing details

- When the execution command is turned ON, the status of the designated devices becomes as shown below:

Device	Device status
Bit device	Turns coils and contacts OFF
Timers and counters	Sets the present value to 0, and turns coils and contacts OFF
When bit designation has been made for word device	Sets value of designated bit to 0
Word devices other than timers and counters	Sets contact to 0

- When the execution command is OFF, the status of devices does not change.
- The functions of the word devices designated by the RST instruction are identical to the following ladder:



- The number of basic steps for the RST instruction is as follows.

For bit processing		Number of basic steps
Internal device (bit to be specified by bit device or word device)		1
Direct access output		2
Timer, counter		4
When using serial number access format file register	Universal model QCPU and LCPU	2
	Basic model QCPU, High Performance model QCPU, Process CPU, and Redundant CPU	3
Other than above		3

For word processing		Number of basic steps
Internal device		2
Index register		2
When using serial number access format file register	Universal model QCPU and LCPU	2
	Basic model QCPU, High Performance model QCPU, Process CPU, and Redundant CPU	3
Other than above		3

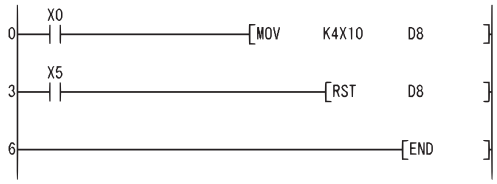
Operation error

- There is no operation error in the RST instruction.

Program example

- The following program sets the value of the data register to 0.

[Ladder Mode]



Stores the contents at X10 to X1F in D8 when X0 is turned ON.

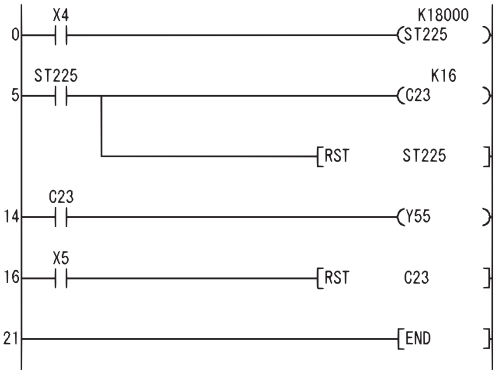
Resets D8 to 0 when X5 is turned ON.

[List Mode]

Steps	Instruction	Device
0	LD	X0
1	MOV	K4X10 D8
3	LD	X5
4	RST	D8
6	END	

- The following program resets the 100ms retentive timer and counter.

[Ladder Mode]



When ST225 is set as retentive timer, it is turned ON when X4 ON time reaches 30 min.

Counts the number of times ST225 was turned ON.

Resets the coil, contact and present value of ST225 when the contact of ST225 is turned ON.

Y55 goes ON at the count-up of C23.

Resets C23 to 0 when X5 is turned ON.

[List Mode]

Step	Instruction	Device
0	LD	X4
1	OUT	ST225 K18000
5	LD	ST225
6	OUT	C23 K16
10	RST	ST225
14	LD	C23
15	OUT	Y55
16	LD	X5
17	RST	C23
21	END	

Setting annunciators, resetting annunciators

SET F, RST F

Basic | High performance | Process | Redundant | Universal | LCPU

SET (D) : Number of the annunciator to be set (F number) (bits)
RST (D) : Number of the annunciator to be reset (F number) (bits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant	Others
	Bit	Word		Bit	Word				
(D)	○ (Only F)	—							

Point

For details on the annunciator, refer to the User's Manual (Function Explanation, Program Fundamentals) for the CPU module used.

Processing details

■SET

- The annunciator designated by (D) is turned ON when the execution command is turned ON.
- The following responses occur when an annunciator (F) is turned ON.
  - The "USER" LED goes ON.*1
  - The annunciator numbers which are ON (F numbers) are stored in special registers (SD64 to SD79).
  - The value of SD63 is incremented by 1.

*1 When using the Basic model QCPU, the "ERR."LED goes ON.

- If the value of SD63 is 16 (which happens when 16 annunciators are already ON), even if a new annunciator is turned ON, its number will not be stored at SD64 to SD79.
- The number of basic steps for the SET F□ instruction is 2.

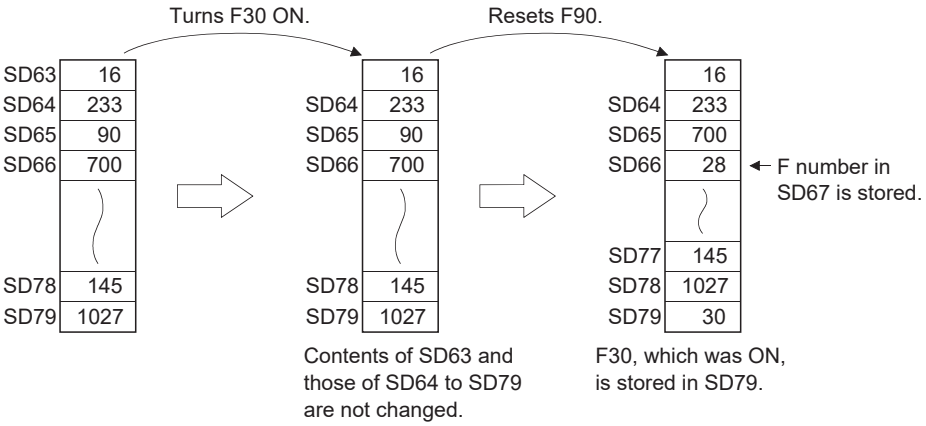
■RST

- The annunciator designated by (D) is turned OFF when the execution command is turned ON.
- The annunciator numbers (F numbers) of annunciators that have gone OFF are deleted from the special registers (SD64 to SD79), and the value of SD63 is decremented by 1.
- When the value of SD63 is "16", the annunciator numbers are deleted from SD64 to SD79 by the use of the RST instruction. If the annunciators whose numbers are not registered in SD64 to SD79 are ON, these numbers will be registered. If all annunciator numbers from SD64 to SD79 are turned OFF, the "USER" LED in the front of the CPU module turns OFF. *1
- The number of basic steps for the RST F□ instruction is 2.

*1 When using the Basic model QCPU, the "ERR." LED goes OFF.

Ex.

[Operations which take place when SD63 is 16]



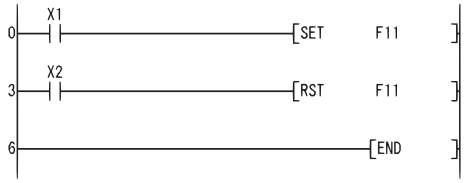
Operation error

- There is no operation error in the SET F or RST F instruction.

Program example

- The following program turns annunciator F11 ON when X1 goes ON, and stores the value 11 at the special register (SD64 to SD79). Further, the program resets annunciator F11 if X2 goes ON, and deletes the value 11 from the special registers (SD64 to SD79).

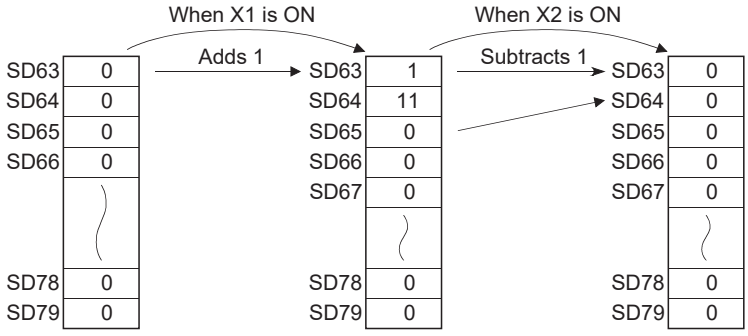
[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	X1
1	SET	F11
3	LD	X2
4	RST	F11
6	END	

[Operation]



Rising edge output, falling edge output

PLS, PLF

Basic | High performance | Process | Redundant | Universal | LCPU

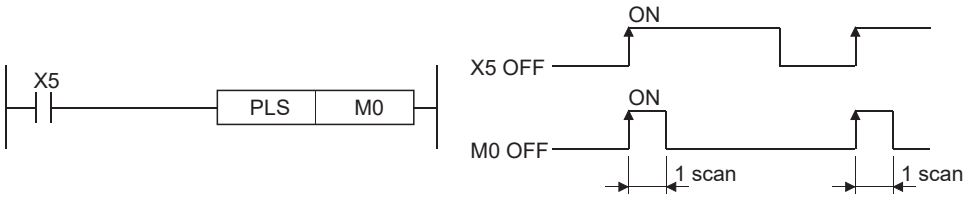
(D): Pulse conversion device (bits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant	Others
	Bit	Word		Bit	Word				DY
(D)	○						—		○

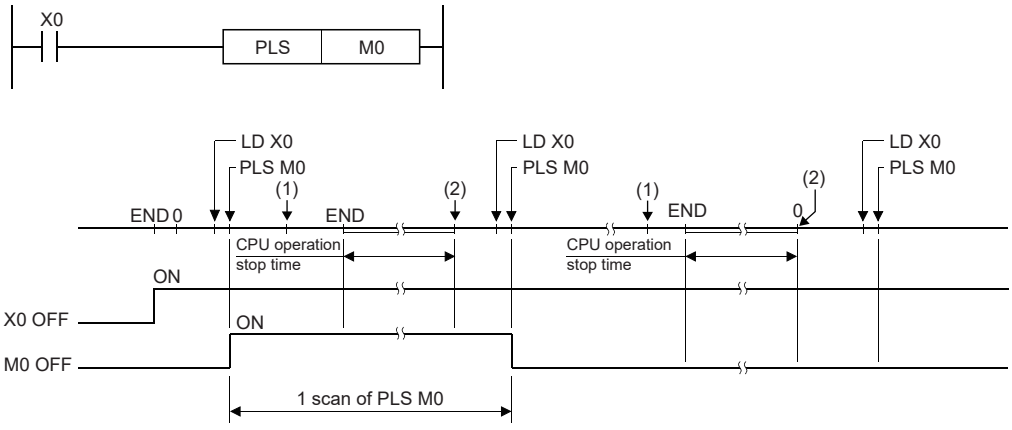
Processing details

■PLS

- Turns ON the designated device when the execution command is turned OFF → ON, and turns OFF the device in any other case the execution command is turned OFF → ON (i.e., at ON → ON, ON → OFF or OFF → OFF of the execution command).
- When there is one PLS instruction for the device designated by (D) during one scan, the specified device turns ON one scan.
- See Page 123 Operation When the OUT, SET/RST, or PLS/PLF Instructions Use the Same Device for the operation to be performed when the PLS instruction for the same device is executed more than once during one scan.



- If the CPU module is changed from RUN to STOP state after the execution of the PLS instruction, the PLS instruction will not be executed again even if the switch is set back to RUN.

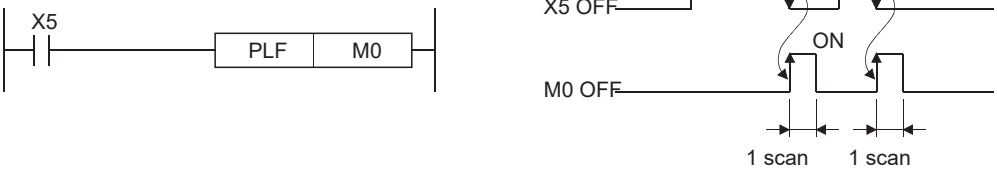


(1) Operating the RUN/STOP switch or RUN/STOP/RESET switch of the CPU module "RUN" to "STOP".
(2) Operating the RUN/STOP switch or RUN/STOP/RESET switch of the CPU module "STOP" to "RUN".

- When designating a latch relay (L) for the execution command and turning the power supply OFF to ON with the latch relay ON, the execution command turns OFF to ON at the first scan, executing the PLS instruction and turning ON the designated device. The device turned ON at the first scan after power-ON turns OFF at the next PLS instruction.

■PLF

- Turns ON the designated device when the execution command is turned ON → OFF, and turns OFF the device in any other case the execution command is turned ON → OFF (i.e., at OFF → OFF, OFF → ON or ON → ON of the execution command).
- When there is one PLF instruction for the device designated by (D) during one scan, the specified device turns ON one scan.
- See Page 123 Operation When the OUT, SET/RST, or PLS/PLF Instructions Use the Same Device for the operation to be performed when the PLF instruction for the same device is executed more than once during one scan.



- If the RUN/STOP key switch is changed from RUN to STOP after the execution of the PLF instruction, the PLF instruction will not be executed again even if the switch is set back to RUN.

Point

Note that the device designated by (D) may remain ON for more than one scan if the PLS or PLF instruction is jumped by the CJ instruction or if the executed subroutine program was not called by the CALL instruction.

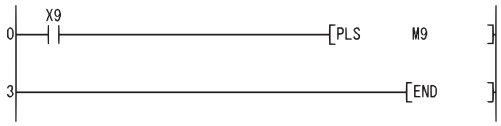
Operation error

- There is no operation error in the PLS or PLF instruction.

Program example

• The following program executes the PLS instruction when X9 goes ON.

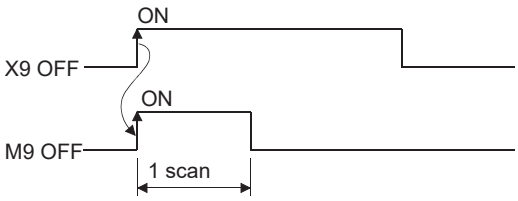
[Ladder Mode]



[List Mode]

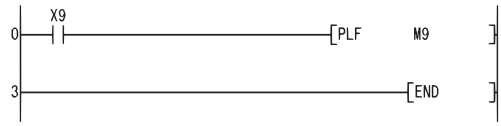
Step	Instruction	Device
0	LD	X9
1	PLS	M9
3	END	

[Timing Chart]



• The following program executes the PLF instruction when X9 goes OFF.

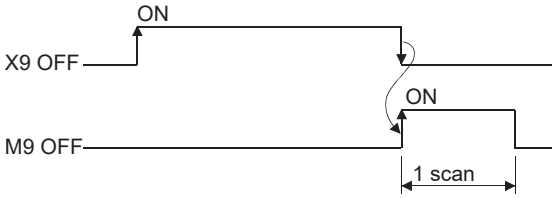
[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	X9
1	PLF	M9
3	END	

[Timing Chart]



Bit device output inversion

FF

Basic | High performance | Process | Redundant | Universal | LCPU

(D): Device number of the device to be reversed (bits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant	Others DY
	Bit	Word		Bit	Word				
(D)	○						—		○

Processing details

- Reverses the output status of the device designated by (D) when the execution command is turned OFF → ON.

Device	Device status	
	Prior to FF execution	After FF execution
Bit device	OFF	ON
	ON	OFF
Word device bit designation	0	1
	1	0

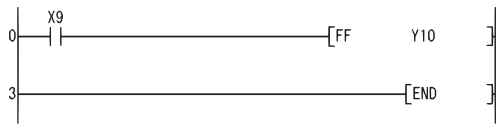
Operation error

- There is no operation error in the FF instruction.

Program example

- The following program reverses the output of Y10 when X9 goes ON.

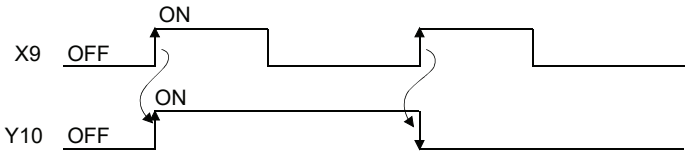
[Ladder Mode]



[List Mode]

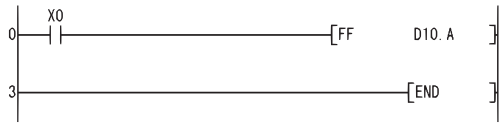
Step	Instruction	Device
0	LD	X9
1	FF	Y10
3	END	

[Timing Chart]



• The following program reverses b10 (bit 10) of D10 when X0 goes ON.

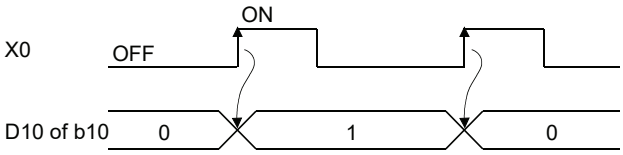
[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	X0
1	FF	D10. A
3	END	

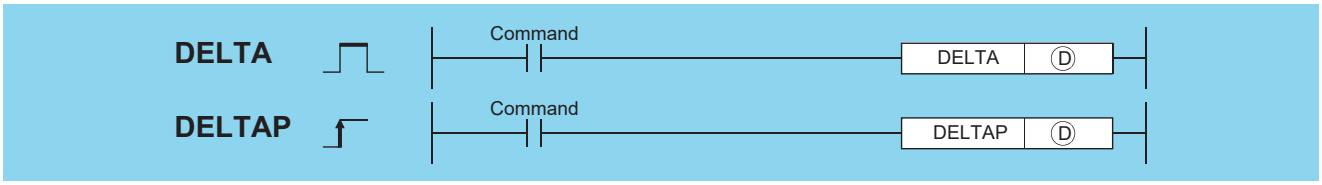
[Timing Chart]



Pulse conversion of direct output

DELTA (P)

Basic | High performance | Process | Redundant | Universal | LCPU



(D): Bit for which pulse conversion is to be conducted (bits)

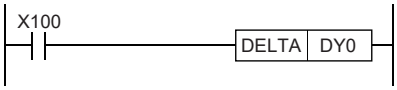
Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant	Others
	Bit	Word		Bit	Word				DY
(D)	—								○

5

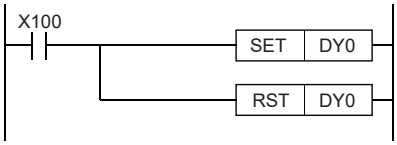
Processing details

- Conducts pulse output of direct access output (DY) designated by (D).
- If DELTA DY0 has been designated, the resulting operation will be identical to the ladder shown below, which uses the SET/RST instructions.

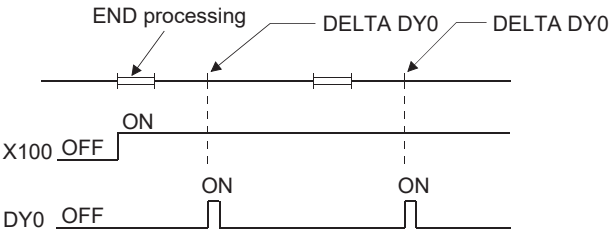
[Ladder using the DELTA instruction]



[Ladder using the SET/RST instructions]



[Operation]



- The DELTA (P) instruction is used by commands for rising edge execution for an intelligent function module.

Operation error

- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

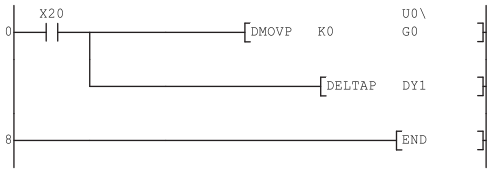
Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4101	The specified direct access output number exceeds the CPU module output range.	○	○	○	○	○	○

Program example

The DELTA(P) instruction is used for setting preset values of high-speed counter modules.

- The following program presets CH1 of the QD62 mounted at slot 0 of the main base unit, when X20 goes ON.

[Ladder Mode]



Stores the preset value (0) in the buffer memory addresses (0 and 1) of the QD62.

Outputs the preset command.

[List Mode]

Step	Instruction	Device	
0	LD	X20	
1	DMOVP	K0	U0\G0
6	DELTAP	DY1	
8	END		

5.4 Shift Instructions

Bit device shift

SFT(P)

Basic | High performance | Process | Redundant | Universal | LCPU

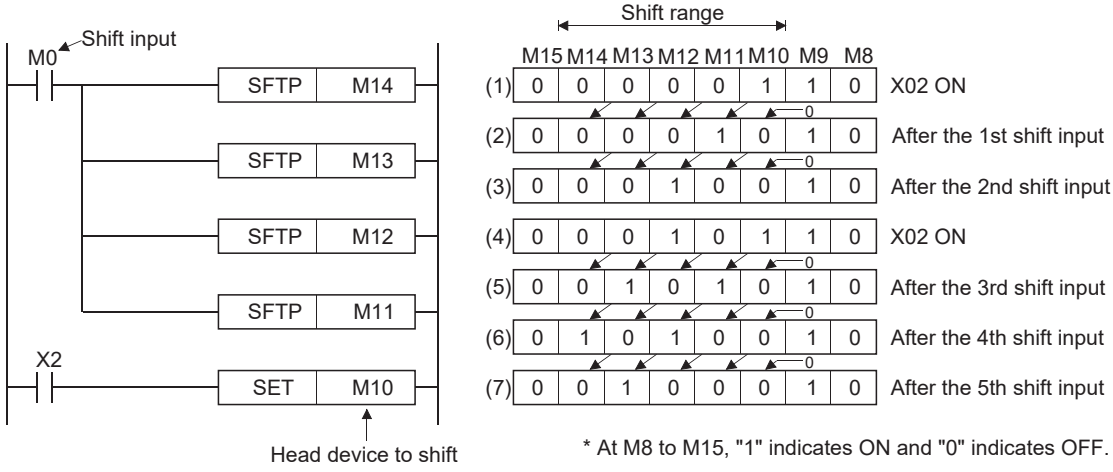
(D): Head number of the devices to be shifted (bits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant	Others DY
	Bit	Word		Bit	Word				
(D)	○ (Other than T, ST, C)						—		○

Processing details

■When bit device is used

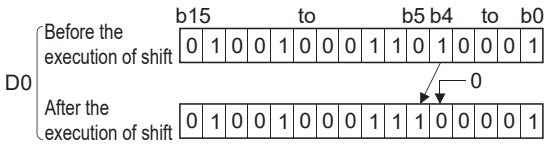
- Shifts to a device designated by (D) the ON/OFF status of the device immediately prior to the one designated by (D), and turns the prior device OFF. For example, if M11 has been designated by the SFT instruction, when the SFT instruction is executed, it will shift the ON/OFF status of M10 to M11, and turn M10 OFF.
- Turn the first device to be shifted ON with the SET instruction.
- When the SFT and SFTP are to be used consecutively, the program starts from the device with the larger number.



* At M8 to M15, "1" indicates ON and "0" indicates OFF.

■When word device bit designation is used

- Shifts to a bit in the device designated by (D) the 1/0 status of the bit immediately prior to the one designated by (D), and turns the prior bit to 0. For example, if D0.5 (bit 5 [b5] of D0) has been designated by the SFT instruction, when the SFT instruction is executed, it will shift the 1/0 status of b4 of D0 to b5, and turn b4 to 0.



Operation error

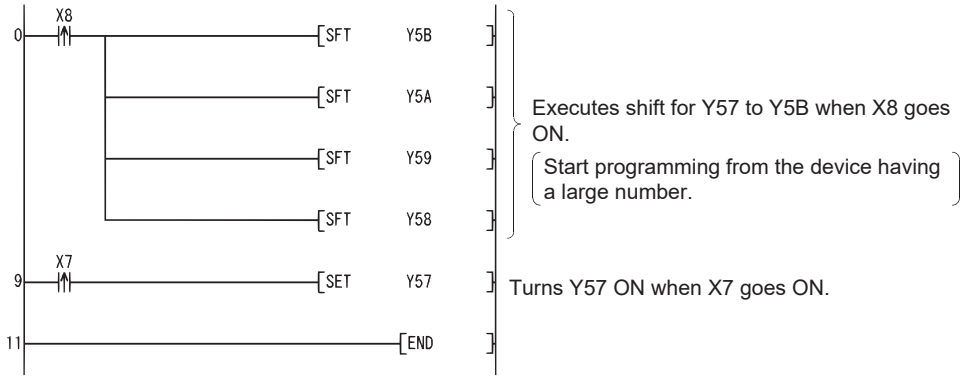
• In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4101	The points of the specified device exceed those of the corresponding device.	○	○	○	○	○	○

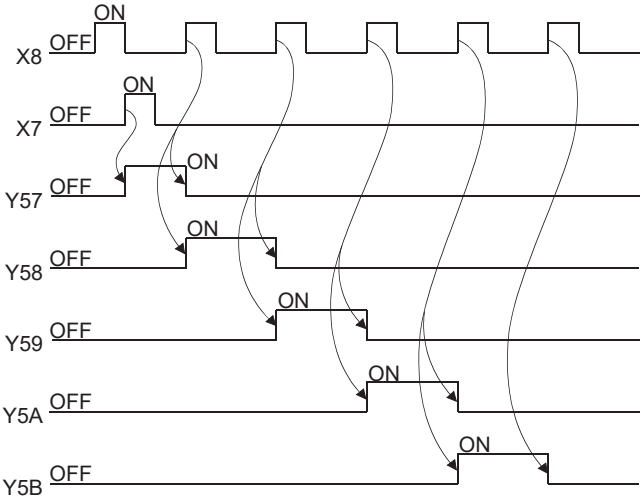
Program example

• The following program shifts Y57 to Y5B when X8 goes ON.

[Ladder Mode]



[Timing Chart]



[List Mode]

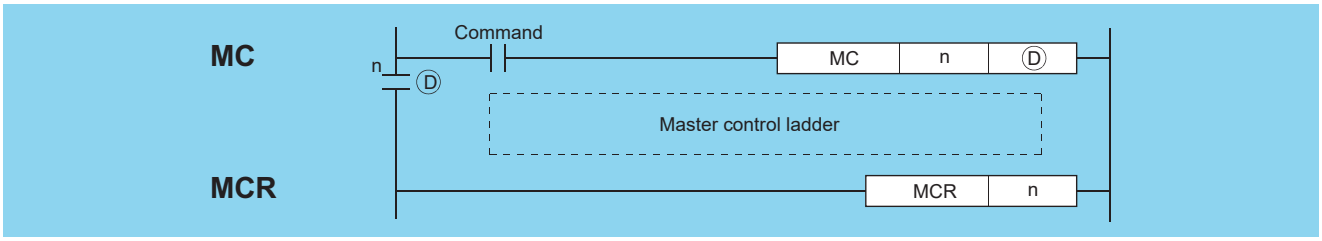
Step	Instruction	Device
0	LDP	X8
1	SFT	Y5B
3	SFT	Y5A
5	SFT	Y59
7	SFT	Y58
9	LDP	X7
10	SET	Y57
11	END	

5.5 Master Control Instructions

Setting the master control, resetting the master control

MC, MCR

Basic | High performance | Process | Redundant | Universal | LCPU

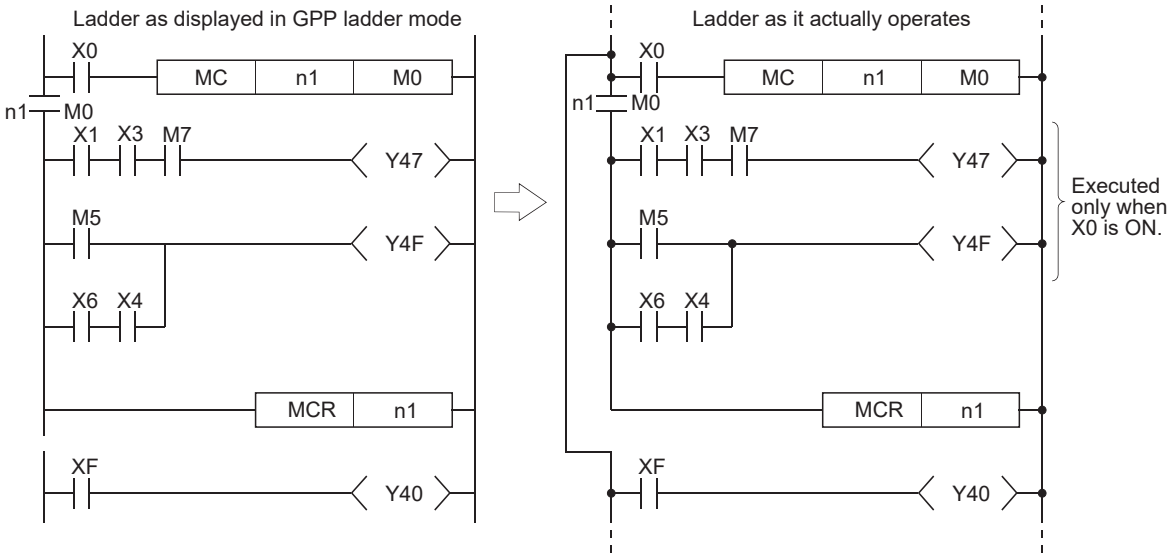


n: Nesting (N0 to N14) (Nesting)
(D): Device number to be turned ON (bits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant	Others	
	Bit	Word		Bit	Word				N	DY
n	—						—		○	—
(D)	○ (Other than T, ST, C)						—		—	○

Processing details

- The master control instruction is used to enable the creation of highly efficient ladder switching sequence programs, through the opening and closing of a common bus for ladders.
- A ladder using the master control is as follows:



Point

Inputting of contacts on the vertical bus is not necessary when programming in the write mode of a programming tool. These will be automatically displayed when the "conversion" operation is conducted after the creation of the ladder and then "read" mode is set.

■MC

- If the execution command of the MC instruction is ON when master control is started, the result of the operation from the MC instruction to the MCR instruction will be exactly as the instruction (ladder) shows. If the execution command of the MC instruction is OFF, the result of the operation from the MC instruction to the MCR instruction will be as shown below:

Device	Device status
High speed timer Low speed timer	Count value goes to 0, coils and contacts all go OFF.
High speed retentive timer Low speed retentive timer Counter	Coils go OFF, but counter values and contacts all maintain current status.
Devices in OUT instruction	All turned OFF
Devices in SET instruction or RST instruction Devices in SFT instruction Devices in Basic instruction or Application instruction	Maintain current status

- Even when the MC instruction is OFF, instructions from the MC instruction to the MCR instruction will be executed, so scan time will not be shortened.

Point

When a ladder with master control contains instructions that do not require any contact instruction (such as FOR to NEXT, EI, DI instructions), the CPU module executes these instructions regardless of the ON/OFF status of the MC instruction execution command.

- By changing the device designated by (D), the MC instruction can use the same nesting (N) number as often as desired.
- Coils from devices designated by (D) are turned ON when the MC instruction is ON. Further, using these same devices with the OUT instruction or other instructions will cause them to become double coils, so devices designated by (D) should not be used within other instructions.

■MCR

- This is the instruction for recovery from the master control, and indicates the end of the master control range of operation.
- Do not place contact instructions before the MCR instruction.
- Use the MC instruction and MCR instruction of the same nesting number as a set.

However, when the MCR instructions are nested in one place, all master controls can be terminated with the lowest nesting (N) number.

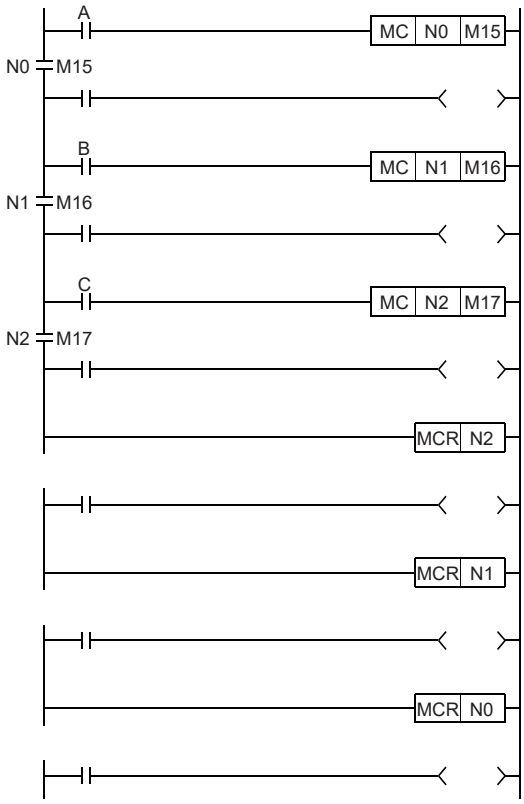
Operation error

- There is no operation error in the MC or MCR instruction.

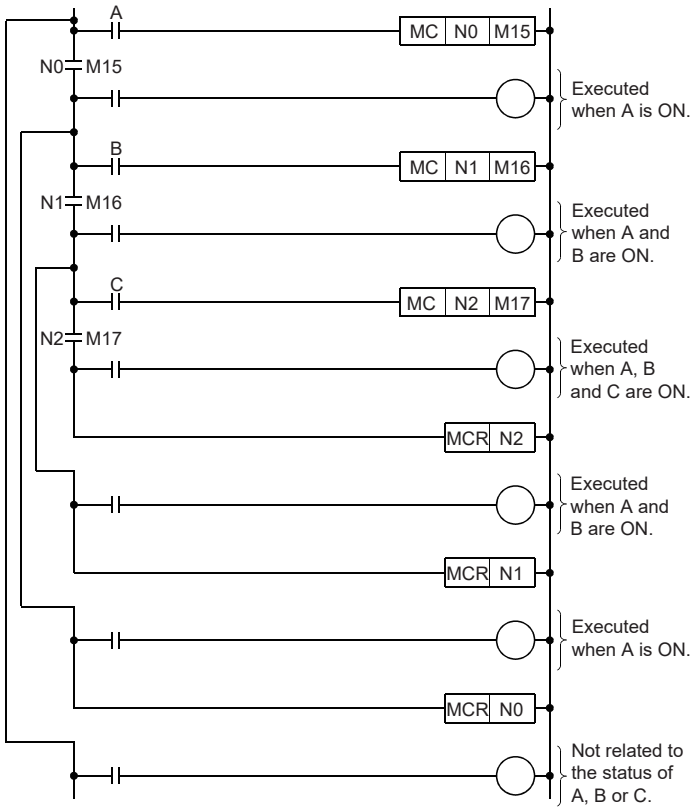
Program example

The master control instruction can be used in nesting. The different master control regions are distinguished by nesting (N).
Nesting can be performed from N0 to N14.
The use of nesting enables the creation of ladders which successively limit the execution condition of the program.
A ladder using nesting would appear as shown below:

[Ladder as displayed in the GPP ladder mode]



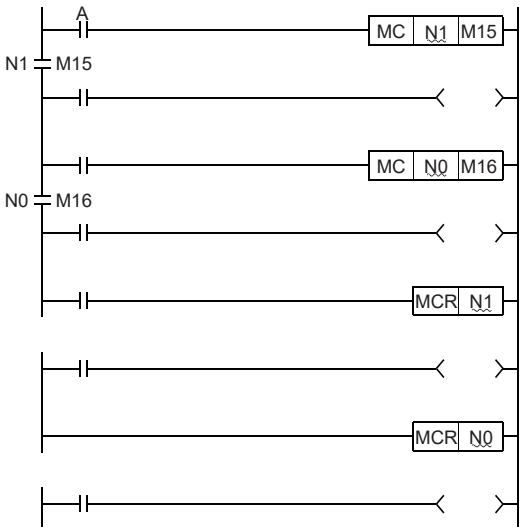
[Ladder as it actually operates]



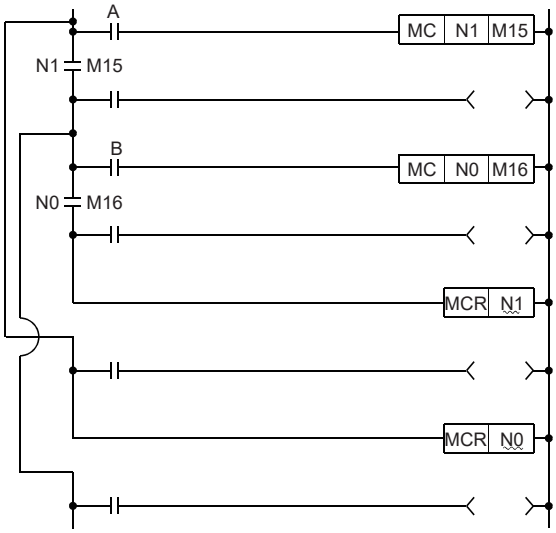
Precautions

- Nesting can be used up to 15 times (N0 to N14). When using nesting, nests should be inserted from the lower to higher nesting number (N) with the MC instruction, and from the higher to the lower order with the MCR instruction. If this order is reversed, there will be no nesting architecture, and the CPU module will not be capable of performing correct operations. For example, if nesting is designated in the order N1 to N0 by the MC instruction, and also designated in the N1 to N0 order by the MCR instruction, the vertical bus will intersect and a correct master control ladder will not be produced.

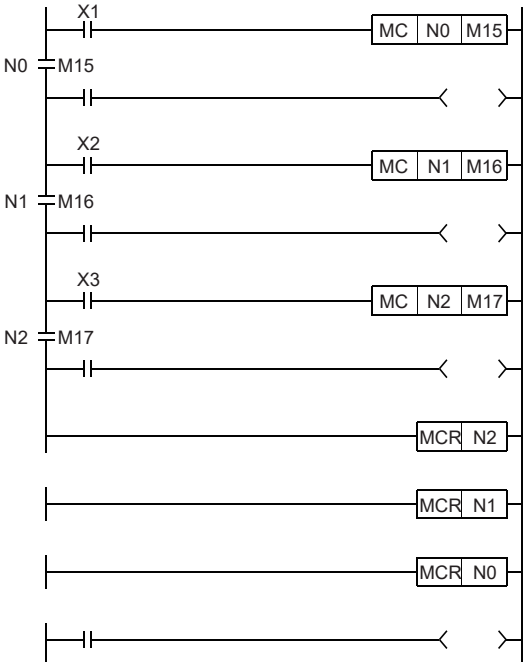
[Ladder as displayed in the GPP ladder mode]

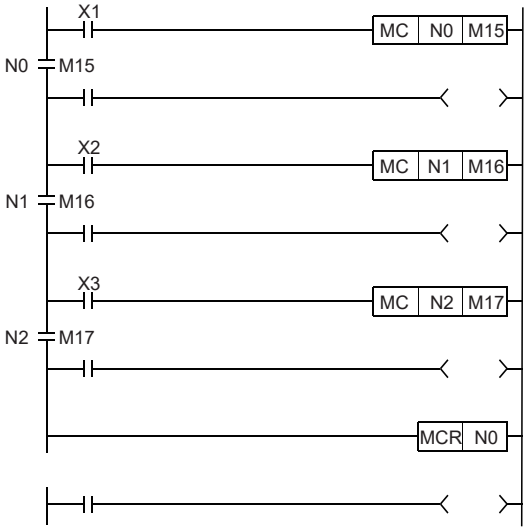


[Ladder as it actually operates]



- If the nesting architecture results in MCR instructions concentrated in one location, all master controls can be terminated by use of just the lowest nesting number (N).





5.6 Termination Instructions

Main routine program end

FEND

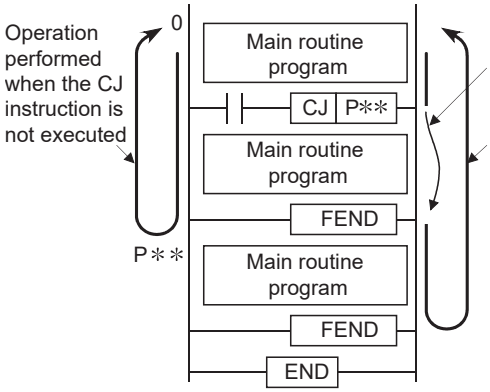
Basic | High performance | Process | Redundant | Universal | LCPU

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant	Others
	Bit	Word		Bit	Word				
—	—								

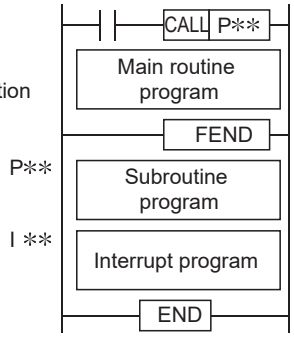
5

Processing details

- The FEND instruction is used in cases where the CJ instruction or other instructions are used to create a branch in the sequence program operations, and in cases where the main routine program is to be split from a subroutine program or an interrupt program.
- Execution of the FEND instruction will cause the CPU module to terminate the program it was executing.
- Even sequence programs following the FEND instruction can be displayed in ladder display at a programming tool. (Programming tool continues to display ladders until encountering the END instruction.)



(a) When the CJ instruction is used



(b) When there are subroutine and interrupt programs

Operation error

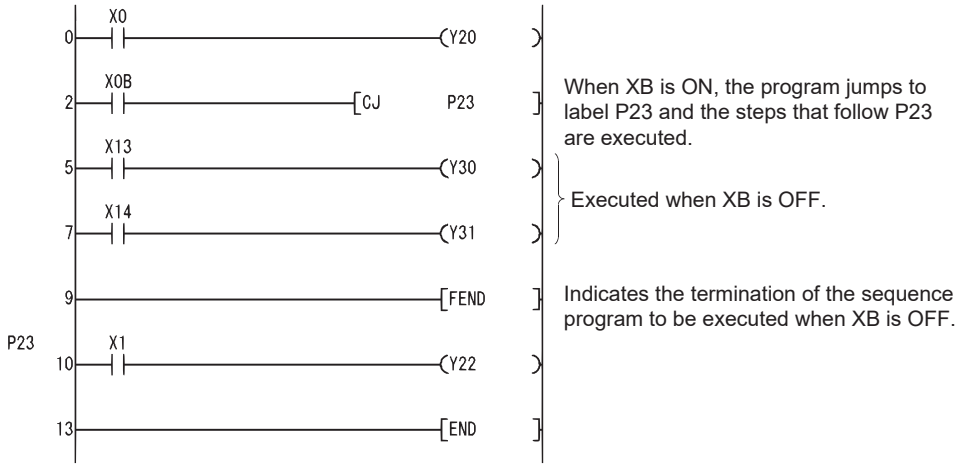
- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4200	The FEND instruction was executed after the execution of the FOR instruction, and before the execution of the NEXT instruction.	○	○	○	○	○	○
4211	The FEND instruction was executed after the execution of the CALL, FCALL, ECALL, or EFCALL instruction, and before the execution of the RET instruction.	○	○	○	○	○	○
4221	The FEND instruction was executed before the execution of the IRET instruction in an interrupt program.	○	○	○	○	○	○
4230	The FEND instruction was executed between the CHKCIR and CHKEND instructions.	—	○	○	○	—	—
4231	The FEND instruction was executed between the IX and IXEND instructions.	○	○	○	○	—	—

Program example

• The following program uses the CJ instruction.

[Ladder Mode]



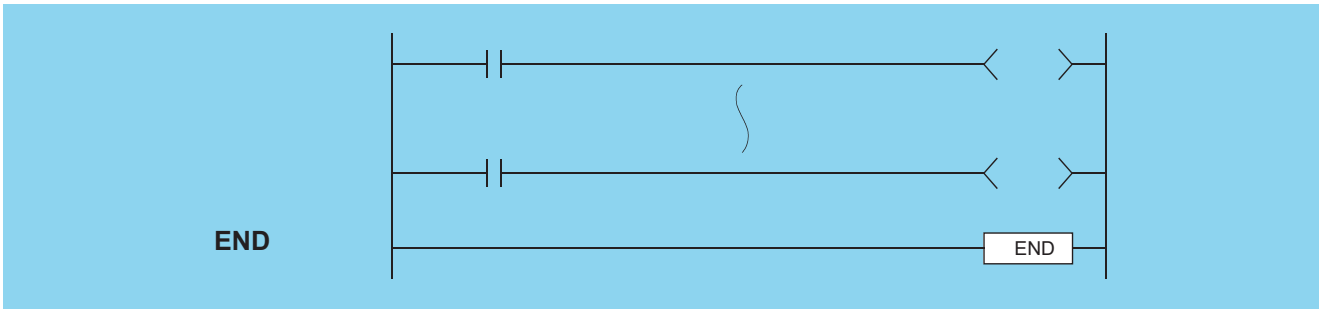
[List Mode]

Step	Instruction	Device
0	LD	X0
1	OUT	Y20
2	LD	X0B
3	CJ	P23
5	LD	X13
6	OUT	Y30
7	LD	X14
8	OUT	Y31
9	FEND	
10	P23	
11	LD	X1
12	OUT	Y22
13	END	

Sequence program end

END

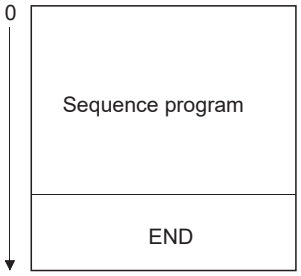
Basic | High performance | Process | Redundant | Universal | LCPU



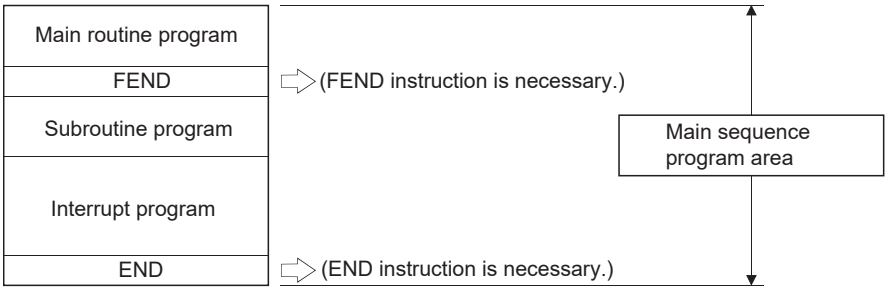
Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant	Others
	Bit	Word		Bit	Word				
—	—								

Processing details

- Indicates termination of programs, including main routine program, subroutine program, and interrupt programs.
- Execution of the END instruction will cause the CPU module to terminate the program that was being executed.



- The END instruction cannot be used during the execution of the main sequence program. If it is necessary to perform END processing during the execution of a program, use the FEND instruction.
- When programming in the ladder mode of a programming tool, it is not necessary to input the END instruction.
- The use of the END and FEND instructions is broken down as follows for main routine programs, subroutine programs, and interrupt programs:



Operation error

• In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4200	The END instruction was executed before the execution of the NEXT instruction and after the execution of the FOR instruction.	○	○	○	○	○	○
4211	The END instruction was executed before the execution of the RET instruction and after the execution of the CALL, FCALL, ECALL, or EFCALL instruction.	○	○	○	○	○	○
4221	The END instruction was executed before the execution of the IRET instruction in an interrupt program.	○	○	○	○	○	○
4230	The END instruction was executed between the CHKCIR to CHKEND instructions.	—	○	○	○	—	—
4231	The END instruction was executed between the IX to IXEND instructions.	○	○	○	○	—	—

5.7 Other Instructions

Sequence program stop

STOP

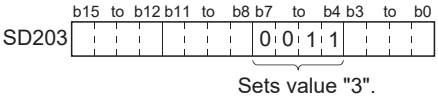
Basic | High performance | Process | Redundant | Universal | LCPU

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant	Others
	Bit	Word		Bit	Word				
—	—								

5

Processing details

- Resets the output (Y) and stops the CPU module operation when the execution command is turned ON. (The same result will take place if switch is turned to the STOP setting.)
- Execution of the STOP instruction will cause the value of b4 to b7 of the special register SD203 to become "3".



- In order to restart CPU module operations after the execution of the STOP instruction, return switch, which has been changed from RUN to STOP, back to the RUN position.

Operation error

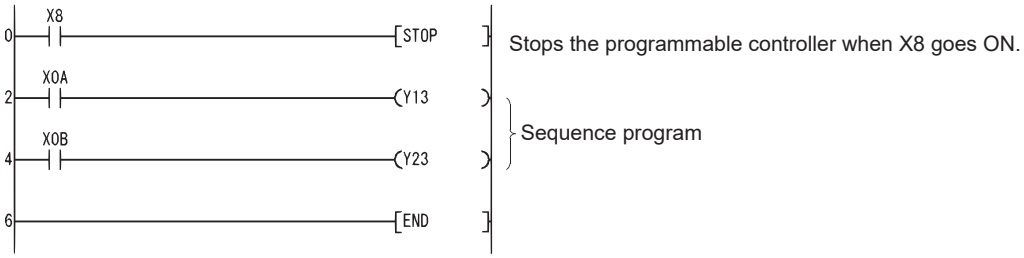
- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4200	The STOP instruction was executed before the execution of the NEXT instruction and after the execution of the FOR instruction.	○	○	○	○	○	○
4211	The STOP instruction was executed before the execution of the RET instruction and after the execution of the CALL/FCALL/ECALL/EFCALL/ XCALL instruction.	○	○	○	○	○	○
4221	The STOP instruction was executed before the execution of the IRET instruction in an interrupt program.	○	○	○	○	○	○
4223	The STOP instruction was executed in the fixed scan execution type program.	—	—	—	—	○	○
4230	The STOP instruction was executed between the CHKCIR to CHKEND instructions.	—	○	○	○	—	—
4231	The STOP instruction was executed between the IX to IXEND instructions.	○	○	○	○	—	—

Program example

• The following program stops the CPU module when X8 goes ON.

[Ladder Mode]



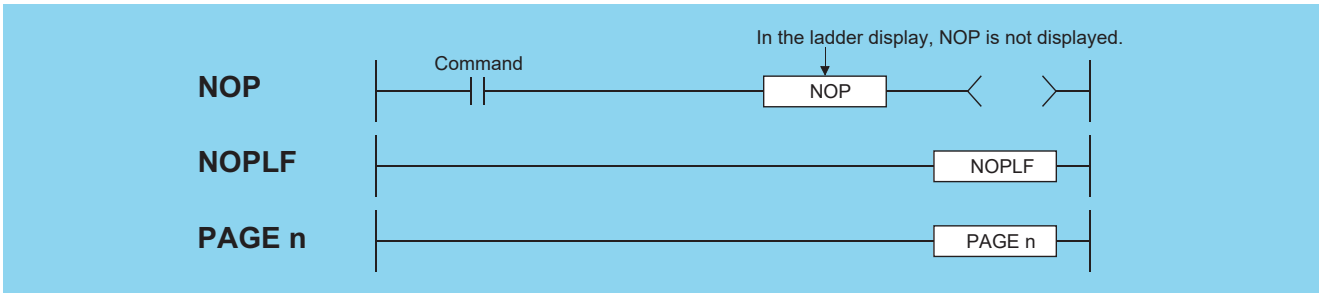
[List Mode]

Step	Instruction	Device
0	LD	X8
1	STOP	
2	LD	X0A
3	OUT	Y13
4	LD	X0B
5	OUT	Y23
6	END	

No operations

NOP, NOPLF, PAGE n

Basic | High performance | Process | Redundant | Universal | LCPU



Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant	Others
	Bit	Word		Bit	Word				
n	—							○*1	—

*1 Only the decimal constant (K) can be used. The hexadecimal constant (H) and the real number (E) cannot be used.

Processing details

■NOP

- This is a no operation instruction that has no impact on any operations up to that point.
- The NOP instruction is used in the following cases:
  - To insert space for sequence program debugging.
  - To delete an instruction without having to change the number of steps. (Replace the instruction with NOP.)
  - To temporarily delete an instruction.

■NOPLF

- This is a no operation instruction that has no impact on any operations up to that point.
- The NOPLF instruction is used when printing from a programming tool to force a page change at any desired location.

Item	Description
When printing ladders	• A page break will be inserted between ladder blocks with the presence of the NOPLF instruction. • The ladder cannot be displayed correctly if an NOPLF instruction is inserted in the midst of a ladder block. • Do not insert an NOPLF instruction in the midst of a ladder block.
When printing instruction lists	• The page will be changed after the printing of the NOPLF instruction.

- Refer to the Operating Manual for the programming tool in use for details of printouts from the programming tool.

■PAGE n

- This is a no operation instruction that has no impact on any operations up to that point.
- No processing is performed at a programming tool with this instruction.

Operation error

- There is no operation error in the NOP, NOPLF, or PAGE instruction.

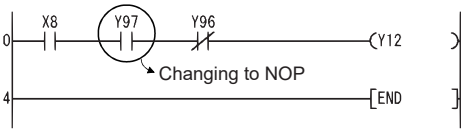
Program example

■NOP

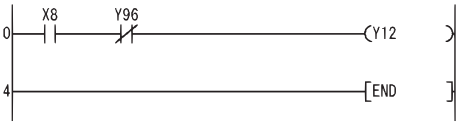
- Contact closed: Deletes the AND or ANI instruction.

[Ladder Mode]

Before change



After change



[List Mode]

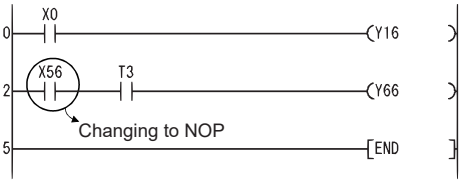
Step	Instruction	Device
0	LD	X8
1	AND	Y97
2	ANI	Y96
3	OUT	Y12
4	END	

Step	Instruction	Device
0	LD	X8
1	NOP	
2	ANI	Y96
3	OUT	Y12
4	END	

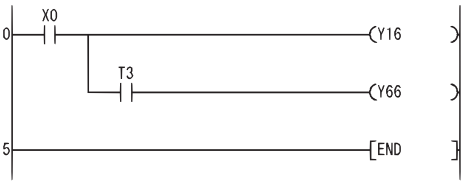
- Contact closed: LD, LDI changed to NOP. (Note carefully that changing the LD and LDI instructions to NOP completely changes the nature of the ladder.)

[Ladder Mode]

Before change



After change



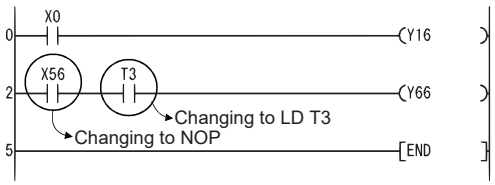
[List Mode]

Step	Instruction	Device
0	LD	X0
1	OUT	Y16
2	LD	X56
3	AND	T3
4	OUT	Y66
5	END	

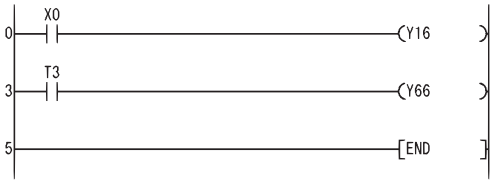
Step	Instruction	Device
0	LD	X0
1	OUT	Y16
2	NOP	
3	AND	T3
4	OUT	Y66
5	END	

[Ladder Mode]

Before change



After change



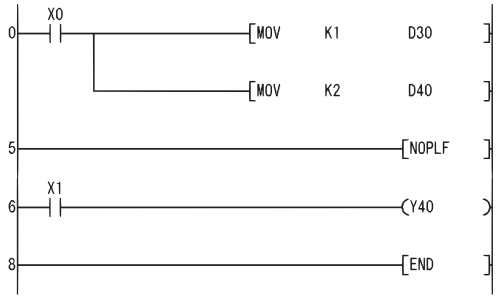
[List Mode]

Step	Instruction	Device
0	LD	X0
1	OUT	Y16
2	LD	X56
3	AND	T3
4	OUT	Y66
5	END	

Step	Instruction	Device
0	LD	X0
1	OUT	Y16
2	NOP	
3	LD	T3
4	OUT	Y66
5	END	

■NOPLF

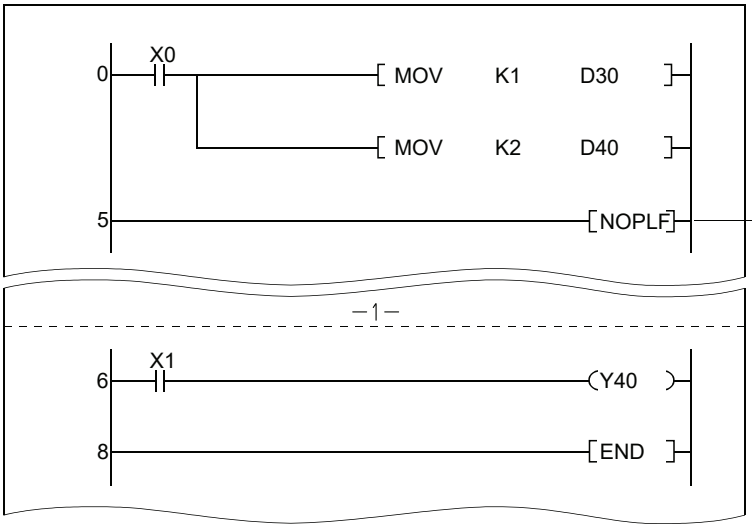
[Ladder Mode]



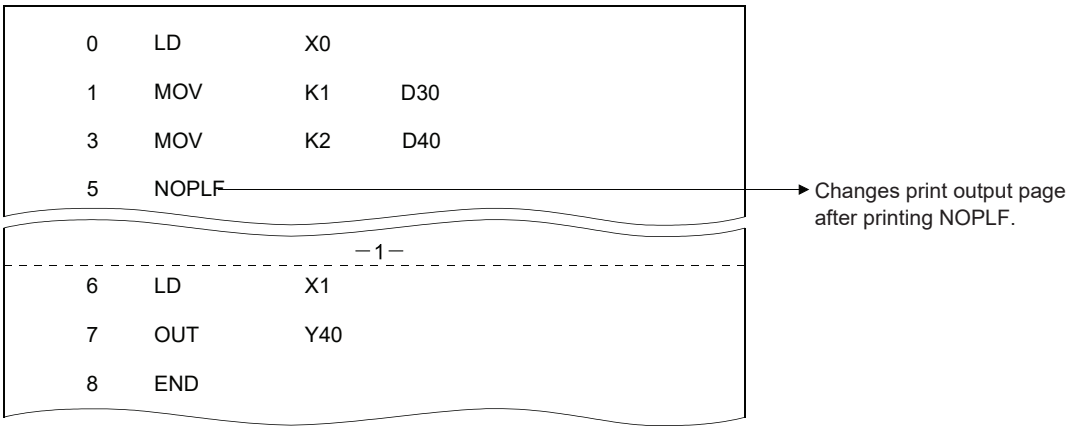
[List Mode]

Step	Instruction	Device	
0	LD	X0	
1	MOV	K1	D30
3	MOV	K2	D40
5	NOPLF		
6	LD	X1	
7	OUT	Y40	
8	END		

• Printing the ladder will result in the following:

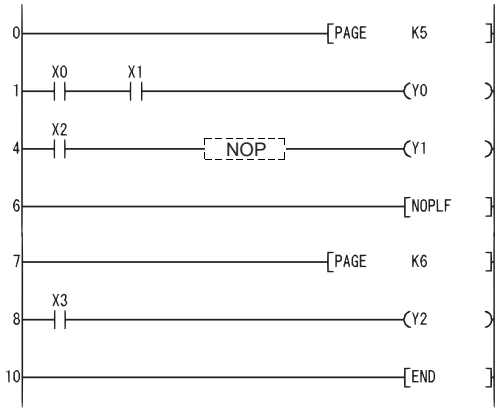


• Printing an instruction list with the NOPLF instruction will result in the following:



■PAGE n

[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	PAGE	K5
1	LD	X0
2	AND	X1
3	OUT	Y0
4	LD	X2
5	NOP	
6	OUT	Y1
7	NOPLF	
8	PAGE	K6
9	LD	X3
10	OUT	Y2
11	END	

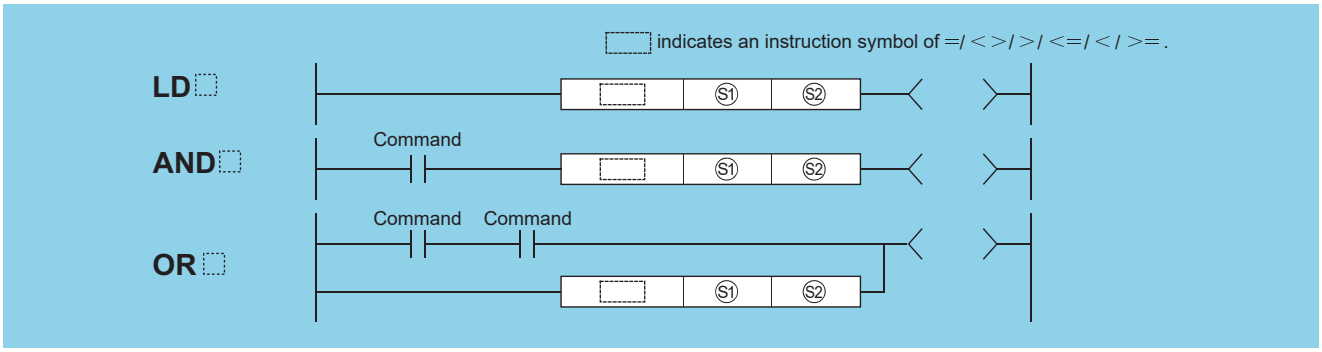
6 BASIC INSTRUCTIONS

6.1 Comparison Operation Instructions

BIN 16-bit data comparisons

LD□, AND□, OR□

Basic | High performance | Process | Redundant | Universal | LCPU



(S1), (S2): Data for comparison or head number of the devices where the data for comparison is stored (BIN 16 bits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(S1)	○								—
(S2)	○								—

Processing details

- Treats BIN 16-bit data from device designated by (S1) and BIN 16-bit data from device designated by (S2) as a normally-open contact, and performs comparison operation.
- The results of the comparison operations for the individual instructions are as follows:

Instruction symbol	Condition	Comparison operation result	Instruction symbol	Condition	Comparison operation result
=	(S1)=(S2)	Continuity	=	(S1)≠(S2)	Non-continuity
<>	(S1)≠(S2)		<>	(S1)=(S2)	
>	(S1)>(S2)		>	(S1)≤(S2)	
<=	(S1)≤(S2)		<=	(S1)>(S2)	
<	(S1)<(S2)		<	(S1)≥(S2)	
>=	(S1)≥(S2)		>=	(S1)<(S2)	

- When (S1) and (S2) are assigned by a hexadecimal constant and the numerical value (8 to F) whose most significant bit (b15) is "1" is designated as a constant, the value is considered as a negative BIN value in comparison operation.

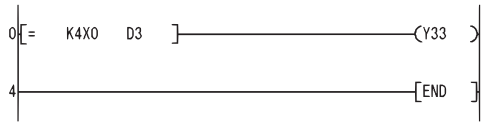
Operation error

- There is no operation error in the LD□ instruction, AND□ instruction, or OR□ instruction.

Program example

- The following program compares the data at X0 to XF with the data at D3, and turns Y33 ON if the data is identical.

[Ladder Mode]

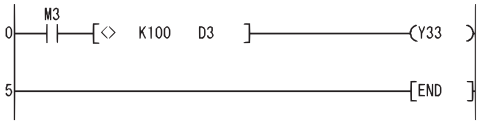


[List Mode]

Step	Instruction	Device	
0	LD=	K4X0	D3
3	OUT	Y33	
4	END		

- The following program compares BIN value K100 to the data at D3, and establishes continuity if the data in D3 is something other than 100.

[Ladder Mode]

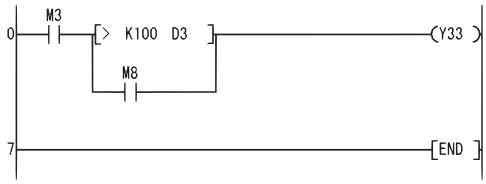


[List Mode]

Step	Instruction	Device	
0	LD	M3	
1	AND<>	K100	D3
4	OUT	Y33	
5	END		

- The following program compares the BIN value 100 with the data at D3, and establishes continuity if the D3 data is less than 100.

[Ladder Mode]

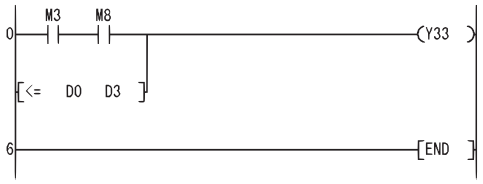


[List Mode]

Step	Instruction	Device	
0	LD	M3	
1	LD>	K100	D3
4	OR	M8	
5	ANB		
6	OUT	Y33	
7	END		

- The following program compares the data in D0 and D3, and if the data in D0 is equal to or less than the data in D3, establishes continuity.

[Ladder Mode]



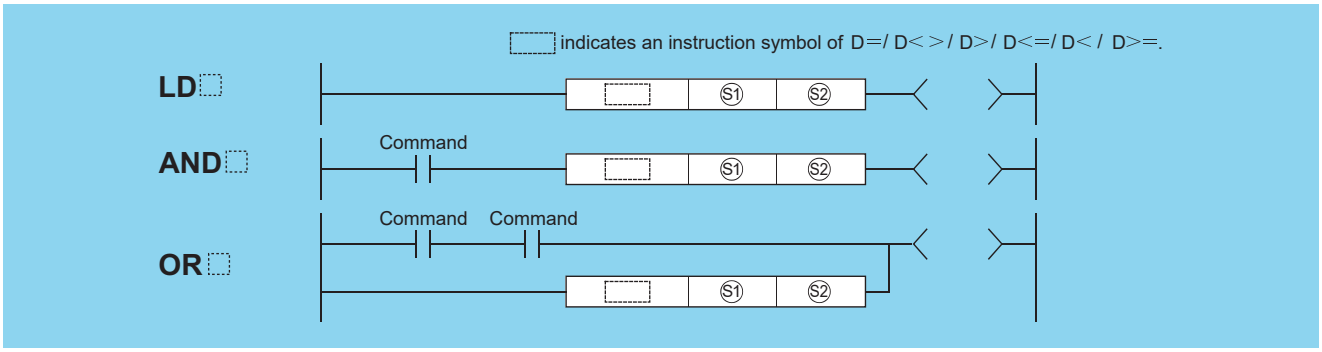
[List Mode]

Step	Instruction	Device	
0	LD	M3	
1	AND	M8	
2	OR<=	D0	D3
5	OUT	Y33	
6	END		

BIN 32-bit data comparisons

LDD□, ANDD□, ORD□

Basic | High performance | Process | Redundant | Universal | LCPU



(S1), (S2): Data for comparison or head number of the devices where the data for comparison is stored (BIN 32 bits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(S1)	○								—
(S2)	○								—

Processing details

- Treats BIN 32-bit data from device designated by (S1) and BIN 32-bit data from device designated by (S2) as a normally-open contact, and performs comparison operation.
- The results of the comparison operations for the individual instructions are as follows:

Instruction symbol	Condition	Comparison operation result	Instruction symbol	Condition	Comparison operation result
D=	(S1)=(S2)	Continuity	D=	(S1)≠(S2)	Non-continuity
D<>	(S1)≠(S2)		D<>	(S1)=(S2)	
D>	(S1)>(S2)		D>	(S1)≤(S2)	
D<=	(S1)≤(S2)		D<=	(S1)>(S2)	
D<	(S1)<(S2)		D<	(S1)≥(S2)	
D>=	(S1)≥(S2)		D>=	(S1)<(S2)	

- When (S1) and (S2) are assigned by a hexadecimal constant and the numerical value (8 to F) whose most significant bit (b31) is "1" is designated as a constant, the value is considered as a negative BIN value in comparison operation.
- Data used for comparison should be designated by a 32-bit instruction (DMOV instruction, etc.). If designation is made with a 16-bit instruction (MOV instruction, etc.), comparisons of large and small values cannot be performed correctly.

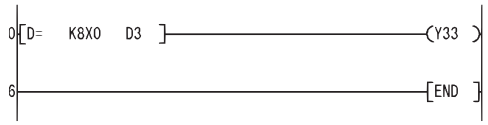
Operation error

- There is no operation error in the LDD□ instruction, ANDD□ instruction, or ORD□ instruction.

Program example

- The following program compares the data at X0 to X1F with the data at D3 and D4, and turns Y33 ON, if the data at X0 to X1F and the data at D3 and D4 match.

[Ladder Mode]

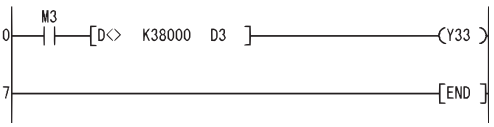


[List Mode]

Step	Instruction	Device	
0	LDD=	K8X0	D3
5	OUT	Y33	
6	END		

- The following program compares BIN value K38000 to the data at D3, and D4, and establishes continuity if the data in D3 and D4 is something other than 38000.

[Ladder Mode]

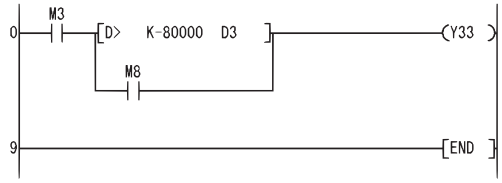


[List Mode]

Step	Instruction	Device	
0	LD	M3	
1	ANDD<>	K38000	D3
6	OUT	Y33	
7	END		

- The following program compares BIN value K-80000 to the data at D3 and D4, and establishes continuity if the data in D3 and D4 is less than -80000.

[Ladder Mode]

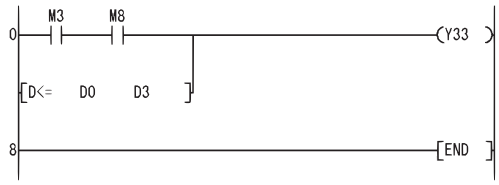


[List Mode]

Step	Instruction	Device	
0	LD	M3	
1	LDD>	K-80000	D3
6	OR	M8	
7	ANB		
8	OUT	Y33	
9	END		

- The following program compares the data in D0 and D1 with the data in D3 and D4, and establishes continuity if the data in D0 and D1 is equal to or less than the data in D3 and D4.

[Ladder Mode]



[List Mode]

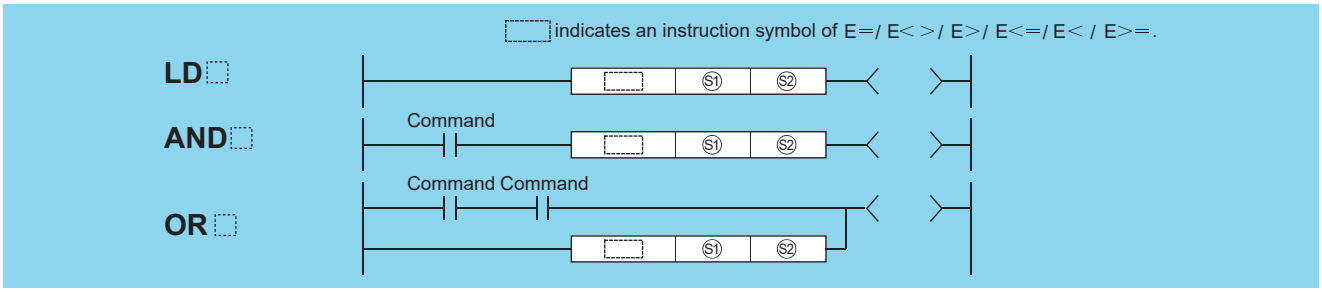
Step	Instruction	Device	
0	LD	M3	
1	AND	M8	
2	ORD<=	D0	D3
7	OUT	Y33	
8	END		

Floating-point data comparisons (single precision)

LDE□, ANDE□, ORE□

Ver. Basic | High performance | Process | Redundant | Universal | LCPU

• Basic model QCPU: The serial number (first five digits) is "04122" or later.



(S1), (S2): Data for comparison or head number of the devices where the data for comparison is stored (real number)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant E	Others
	Bit	Word		Bit	Word				
(S1)	—	○		—	○		○*1	○	—
(S2)	—	○		—	○		○*1	○	—

*1 Applicable for the Universal model QCPU, LCPU.

Processing details

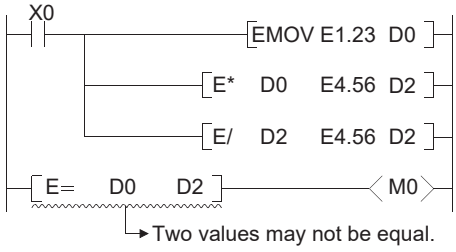
- The 32-bit floating decimal point data from device designated by (S1) and 32-bit floating decimal point data from device designated by (S2) as a normally-open contact, and performs comparison operation.
- The results of the comparison operations for the individual instructions are as follows:

Instruction symbol	Condition	Comparison operation result	Instruction symbol	Condition	Comparison operation result
E=	(S1)=(S2)	Continuity	E=	(S1)≠(S2)	Non-continuity
E<>	(S1)≠(S2)		E<>	(S1)=(S2)	
E>	(S1)>(S2)		E>	(S1)≤(S2)	
E<=	(S1)≤(S2)		E<=	(S1)>(S2)	
E<	(S1)<(S2)		E<	(S1)≥(S2)	
E>=	(S1)≥(S2)		E>=	(S1)<(S2)	

Point

Note that use of the E= instruction can on occasion result in situations where errors cause the two values not to be equal.

Example



- When an input value is set using a programming tool, a rounding error may occur. For precautions, refer to Page 90 Precautions.

Operation error

- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

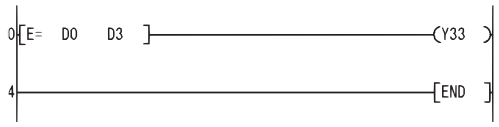
Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4100	The specified device value is -0.*2	○	○*2	○	○	—	—
4140	The specified device value is not within the following range: 0, $2^{-126} \leq$ | Specified device value | $< 2^{128}$ The specified device value is -0, unnormalized number, nonnumeric, and ±∞.	—	—	—	—	○*2	○

*2 There are CPU modules that will not result in an operation error if -0 is specified. For details, refer to Page 88 Using single/double-precision real number data.

Program example

- The following program compares 32-bit floating decimal point real number data at D0 and D1 to 32-bit floating decimal point real number data at D3 and D4.

[Ladder Mode]

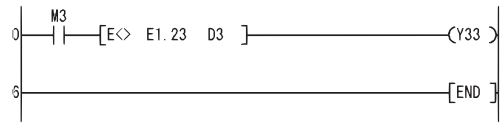


[List Mode]

Step	Instruction	Device	
0	LDE=	D0	D3
3	OUT	Y33	
4	END		

- The following program compares the floating decimal point real number 1.23 to the 32-bit floating decimal point real number data at D3 and D4.

[Ladder Mode]

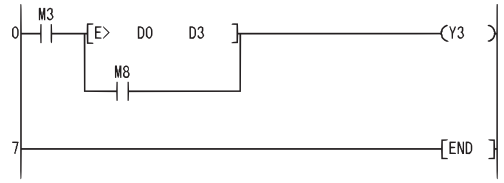


[List Mode]

Step	Instruction	Device	
0	LD	M3	
1	ANDE<>	E1.23	D3
5	OUT	Y33	
6	END		

- The following program compares 32-bit floating decimal point real number data at D0 and D1 to 32-bit floating decimal point real number data at D3 and D4.

[Ladder Mode]

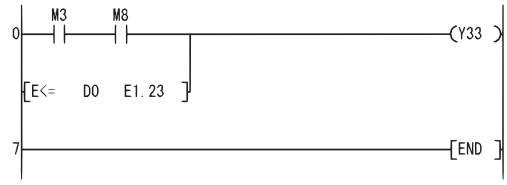


[List Mode]

Step	Instruction	Device	
0	LD	M3	
1	LDE>	D0	D3
4	OR	M8	
5	ANB		
6	OUT	Y3	
7	END		

- The following program compares the 32-bit floating decimal point data at D0 and D1 to the floating decimal point real number 1.23.

[Ladder Mode]



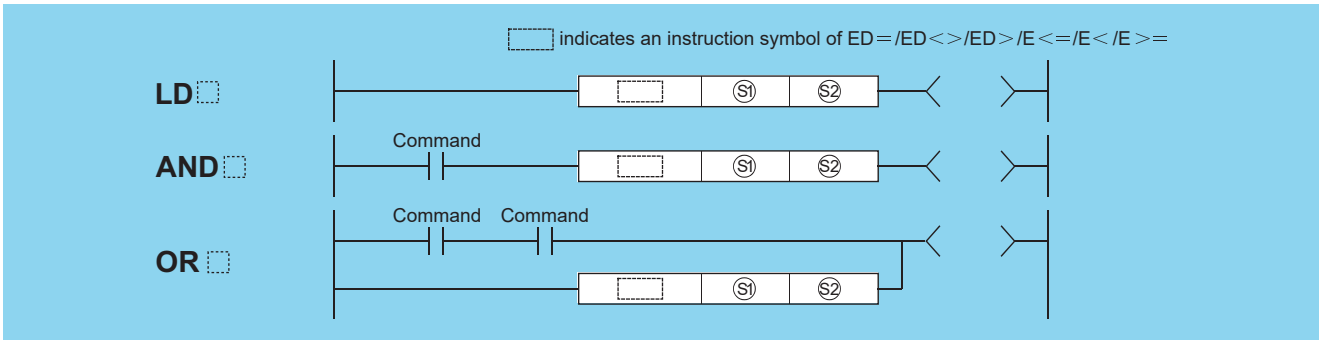
[List Mode]

Step	Instruction	Device	
0	LD	M3	
1	AND	M8	
2	ORE<=	D0	E1.23
6	OUT	Y33	
7	END		

Floating-point data comparisons (double precision)

LDED□, ANDED□, ORED□

Basic | High performance | Process | Redundant | Universal | LCPU



(S1), (S2): Data for comparison or head number of the devices where the data for comparison is stored (real number)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant E	Others
	Bit	Word		Bit	Word				
(S1)	—	○		—			—	○	—
(S2)	—	○		—			—	○	—

6

Processing details

- The 64-bit floating decimal point real number from device designated by (S1) and 64-bit floating decimal point real number from device designated by (S2) as a normally-open contact, and performs comparison operation.
- The results of the comparison operations for the individual instructions are as follows:

Instruction symbol	Condition	Comparison operation result	Instruction symbol	Condition	Comparison operation result
ED=	(S1)=(S2)	Continuity	ED=	(S1)≠(S2)	Non-continuity
ED<>	(S1)≠(S2)		ED<>	(S1)=(S2)	
ED>	(S1)>(S2)		ED>	(S1)≤(S2)	
ED<=	(S1)≤(S2)		ED<=	(S1)>(S2)	
ED<	(S1)<(S2)		ED<	(S1)≥(S2)	
ED>=	(S1)≥(S2)		ED>=	(S1)<(S2)	

- When an input value is set using a programming tool, a rounding error may occur. For precautions, refer to Page 90 Precautions.

Operation error

- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

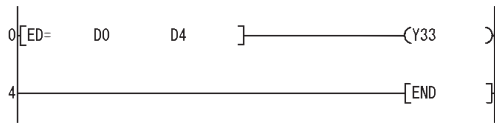
Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4140	The specified device value is not within the following range: 0, $2^{-1022} \leq$ | Specified device value | $< 2^{1024}$ The specified device value is -0, unnormalized number, nonnumeric, and ±∞.	—	—	—	—	○*1	○

*1 There are CPU modules that will not result in an operation error if -0 is specified. For details, refer to Page 88 Using single/double-precision real number data.

Program example

- The following program compares 64-bit floating decimal point real number data at D0 to D3 with 64-bit floating decimal point real number data at D4 to D7.

[Ladder Mode]

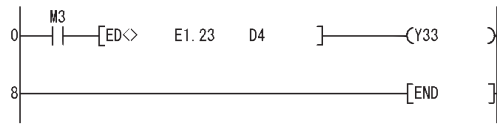


[List Mode]

Step	Instruction	Device	
0	LDED=	D0	D4
3	OUT	Y33	
4	END		

- The following program compares the floating decimal point real number 1.23 with the 64-bit floating decimal point real number data at D4 to D7.

[Ladder Mode]

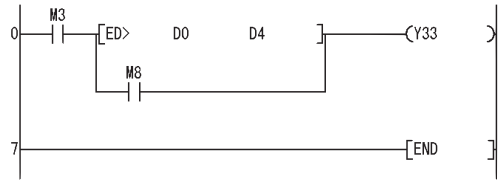


[List Mode]

Step	Instruction	Device	
0	LD	M3	
1	ANDED<>	E1.23	D4
7	OUT	Y33	
8	END		

- The following program compares 64-bit floating decimal point real number data at D0 to D3 with 64-bit floating decimal point real number data at D4 to D7.

[Ladder Mode]

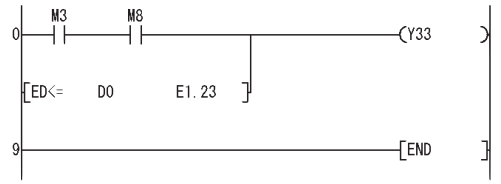


[List Mode]

Step	Instruction	Device	
0	LD	M3	
1	LDED>	D0	D4
4	OR	M8	
5	ANB		
6	OUT	Y33	
7	END		

- The following program compares the 64-bit floating decimal point data at D0 to D3 with the floating decimal point real number 1.23.

[Ladder Mode]



[List Mode]

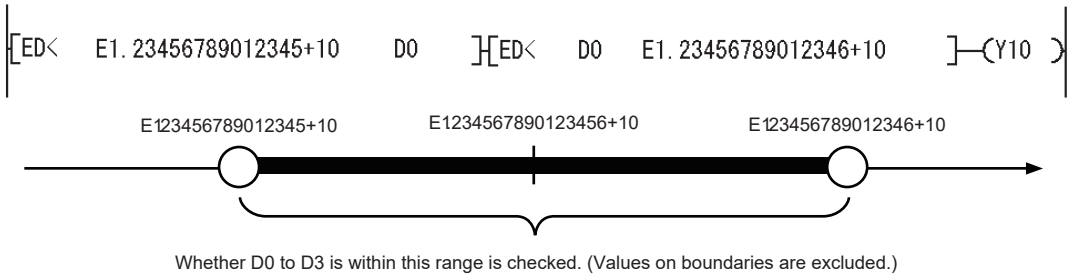
Step	Instruction	Device	
0	LD	M3	
1	AND	M8	
2	ORED<=	D0	E1.23
8	OUT	Y33	
9	END		

Precautions

- Since the number of digits of the real number that can be input by Programing Tool is up to 15 digits, the comparison with the real number whose number of significant digits is 16 or more cannot be made by the instruction shown in this section. When judging match/mismatch with the real number whose significant digits is 16 or more by the instruction in this section, compare it with the approximate values of the real number to be compared and judge by the sizes.

Ex.

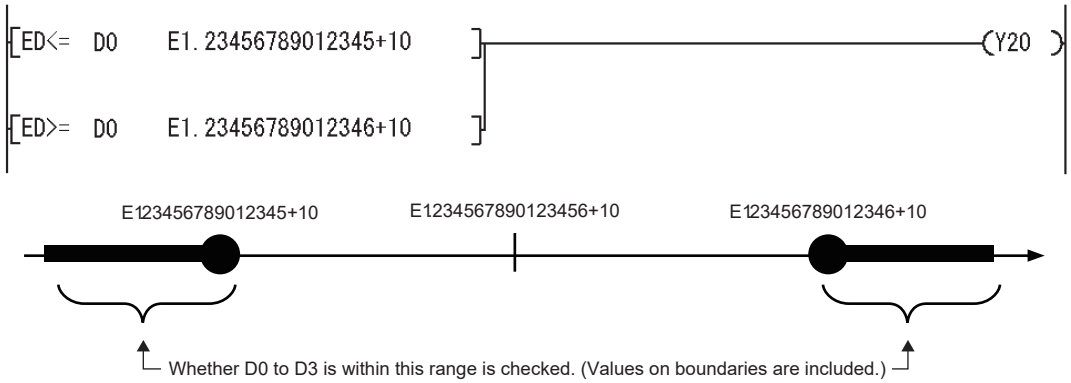
When judging the match of E1.234567890123456+10 (Number of significant digits is 16) and the double-precision floating-point data.



Whether D0 to D3 is within this range is checked. (Values on boundaries are excluded.)

Ex.

When judging the mismatch of E1.234567890123456+10 (Number of significant digits is 16) and the double-precision floating-point data.

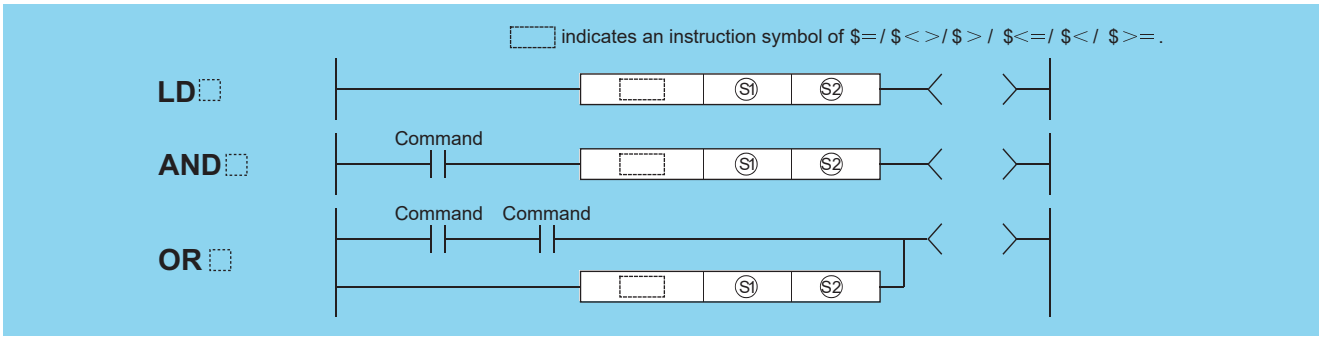


Whether D0 to D3 is within this range is checked. (Values on boundaries are included.)

Character string data comparisons

LD$□, AND$□, OR$□

Basic | High performance | Process | Redundant | Universal | LCPU



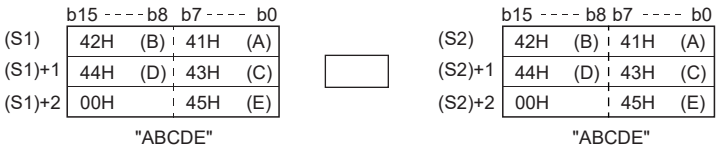
(S1), (S2): Data for comparison or head number of the devices where the data for comparison is stored (character string)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant $	Others
	Bit	Word		Bit	Word				
(S1)	—	○		—				○	—
(S2)	—	○		—				○	—

Processing details

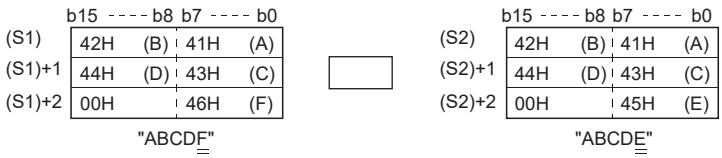
- Compares the character string data designated by (S1) with the character string data designated by (S2) as a normally-open contact.
- A comparison operation involves the character-by-character comparison of the ASCII code of the first character in the character string.
- The character string data of (S1) and (S2) for comparison refers to the data stored within the range from the specified device number to the device number where the NULL code "00H" is stored.

• If all character strings match, the comparison result will be matched.



Instruction symbol	Comparison operation result	Instruction symbol	Comparison operation result
$=	Continuity	$<=	Continuity
$<>	Non-continuity	$<	Non-continuity
$>	Non-continuity	$>=	Continuity

• If the character strings are different, the character string with the larger character code will be the larger.



Instruction symbol	Comparison operation result	Instruction symbol	Comparison operation result
$=	Non-continuity	$<=	Non-continuity
$<>	Continuity	$<	Non-continuity
$>	Continuity	$>=	Continuity

• If the character strings are different, the first different sized character code will determine whether the character string is larger or smaller.

	b15 ---- b8		b7 ---- b0	
(S1)	32H	(2)	31H	(1)
(S1)+1	34H	(4)	33H	(3)
(S1)+2	00H		35H	(5)

"12345"

	b15 ---- b8		b7 ---- b0	
(S2)	32H	(2)	31H	(1)
(S2)+1	33H	(3)	34H	(4)
(S2)+2	00H		35H	(5)

"12435"

Instruction symbol	Comparison operation result	Instruction symbol	Comparison operation result
$=	Non-continuity	$<=	Continuity
$<>	Continuity	$<	Continuity
$>	Non-continuity	$>=	Non-continuity

• If the character strings designated by (S1) and (S2) are of different lengths, the data with the longer character string will be larger.

	b15 ---- b8		b7 ---- b0	
(S1)	32H	(2)	31H	(1)
(S1)+1	34H	(4)	33H	(3)
(S1)+2	36H	(6)	35H	(5)
(S1)+3	00H		37H	(7)

"1234567"

	b15 ---- b8		b7 ---- b0	
(S2)	32H	(2)	31H	(1)
(S2)+1	34H	(4)	33H	(3)
(S2)+2	36H	(6)	35H	(5)
(S2)+3	00H		00H	

"123456"

Instruction symbol	Comparison operation result	Instruction symbol	Comparison operation result
$=	Non-continuity	$<=	Non-continuity
$<>	Continuity	$<	Non-continuity
$>	Continuity	$>=	Continuity

Operation error

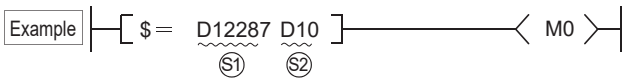
• In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4101	The NULL code "00H" does not exist within the range of the corresponding devices, starting from the device specified by (S1) and (S2). The number of character strings of (S1) and (S2) exceeds 16383.	—	○	○	○	○	○

Point

The character string data comparison instruction checks the device range while comparing the designated character string data.

For this reason, even though the NULL code "00H" does not exist within the range of the corresponding devices, the instruction outputs a comparison result instead of returning an operation error when no match of characters is detected.



Data of (S1)

D12287	B	A
W0	00H	C

Data of (S2)

D10	Z	A
D11	00H	C

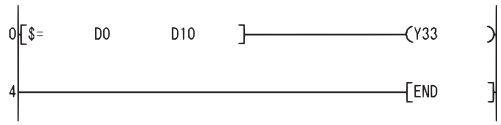
If (S1) and (S2) data are as shown above, the second character of (S1) does not match with that of (S2), and the comparison result is expressed as (S1)≠(S2) (the operation result is "non-conductive").

Though the NULL code "00H" is not included within the (S1) device range, no operation error is returned, because mismatch is detected at D12287, which is within the device range.

Program example

• The following program compares character strings stored following D0 and characters following D10.

[Ladder Mode]

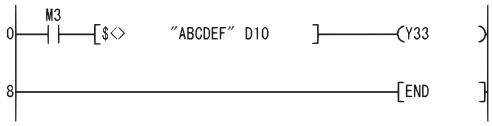


[List Mode]

Step	Instruction	Device	
0	LD$=	D0	D10
3	OUT	Y33	
4	END		

• The following program compares the character string "ABCDEF" with the character string stored following D10.

[Ladder Mode]

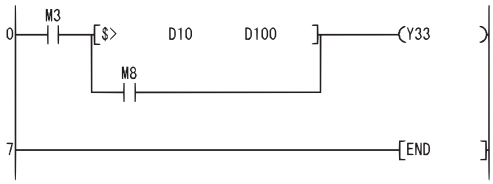


[List Mode]

Step	Instruction	Device
0	LD	M3
1	AND$<>	"ABCDEF" D10
7	OUT	Y33
8	END	

• The following program compares the character string stored following D10 with the character string stored following D100.

[Ladder Mode]

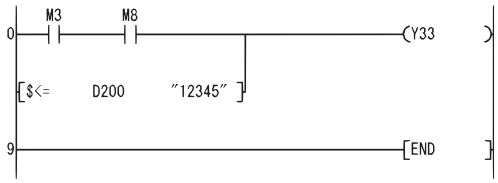


[List Mode]

Step	Instruction	Device	
0	LD	M3	
1	LD$>	D10	D100
4	OR	M8	
5	ANB		
6	OUT	Y33	
7	END		

• The following program compares the character string stored following D200 with the character string "12345".

[Ladder Mode]



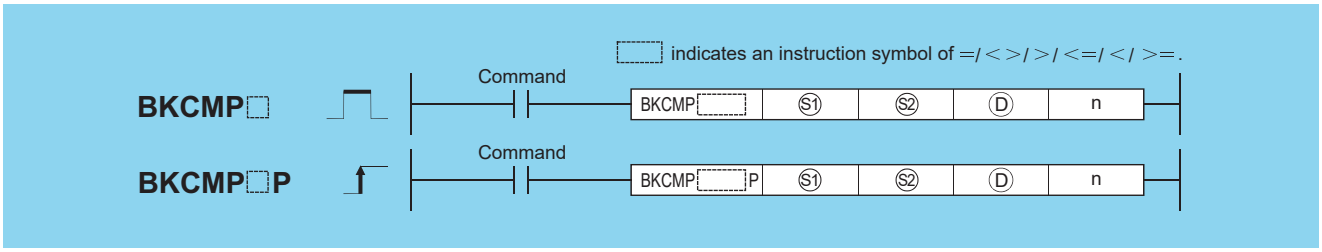
[List Mode]

Step	Instruction	Device	
0	LD	M3	
1	AND	M8	
2	OR$<=	D200	"12345"
8	OUT	Y33	
9	END		

BIN 16-bit block data comparisons

BKCMP□(P)

Basic | High performance | Process | Redundant | Universal | LCPU



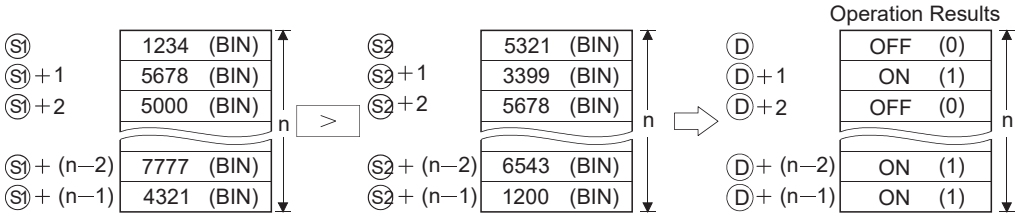
(S1): Data to be compared or head number of the devices where the data to be compared is stored (BIN 16 bits)
(S2): Head number of the devices where the comparison data is stored (BIN 16 bits)
(D): Head number of the devices where the comparison operation result will be stored (bits)
n: Number of comparison data blocks (BIN 16 bits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(S1)	—	○		—				○	—
(S2)	—	○		—				—	—
(D)	○	○		—				—	—
n	○	○		○				○	—

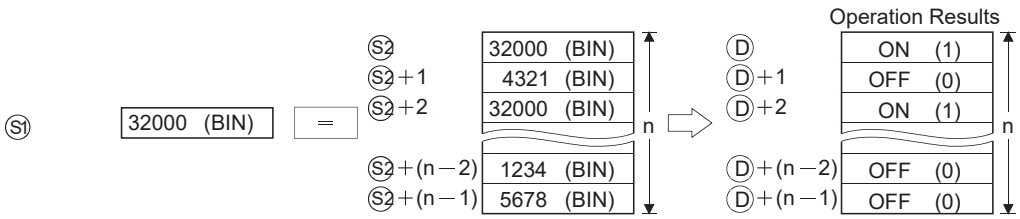
6

Processing details

- Compares BIN 16-bit data the nth point from the device number designated by (S1) with BIN 16-bit data the nth point from the device number designated by (S2), and stores the result from the device designated by (D) onward.
  - If the comparison condition has been met, the device designated by (D) will be turned on.
  - If the comparison condition has not been met, the device designated by (D) will be turned OFF.



- The comparison operation is conducted in 16-bit units.
- The constant designated by (S1) can be between -32768 and 32767 (BIN 16-bit data).



• The results of the comparison operations for the individual instructions are as follows:

Instruction symbol	Condition	Comparison operation result	Instruction symbol	Condition	Comparison operation result
BKCMP=	(S1)=(S2)	ON (1)	BKCMP=	(S1)≠(S2)	OFF (0)
BKCMP<>	(S1)≠(S2)		BKCMP<>	(S1)=(S2)	
BKCMP>	(S1)>(S2)		BKCMP>	(S1)≤(S2)	
BKCMP<=	(S1)≤(S2)		BKCMP<=	(S1)>(S2)	
BKCMP<	(S1)<(S2)		BKCMP<	(S1)≥(S2)	
BKCMP>=	(S1)≥(S2)		BKCMP>=	(S1)<(S2)	

• If all comparison results stored n points from (D) are ON (1), SM704 (block comparison signal) turns ON.

Operation error

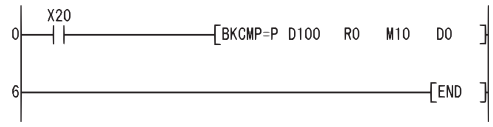
• In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4101	The points specified in n exceed those of the corresponding device specified in (S1), (S2), or (D). The ranges of devices starting from the one specified in (S1) and (D) overlap by n points. The ranges of devices starting from the one specified in (S2) and (D) overlap by n points.	○	○	○	○	○	○

Program example

• The following program compares, when X20 is turned ON, the data stored at D100 to D103 with the data stored at R0 to R3 and stores the operation result into the area starting from M10.

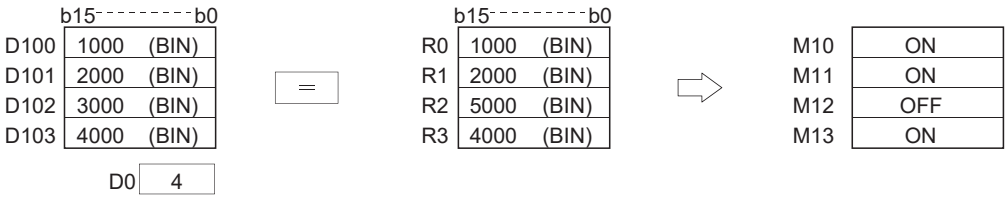
[Ladder Mode]



[List Mode]

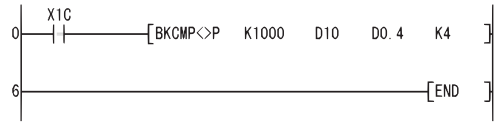
Step	Instruction	Device
0	LD	X20
1	BKCMP=P	D100 R0 M10 D0
6	END	

[Operation]



• The following program compares, when X1C is turned ON, the constant K1000 with the data stored at D10 to D13, and stores the operation result at b4 to b7 in D0.

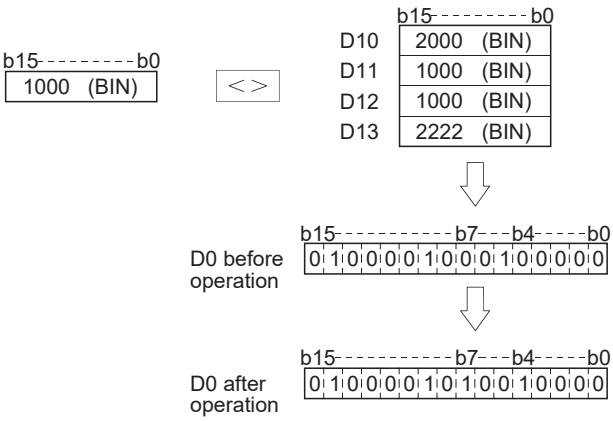
[Ladder Mode]



[List Mode]

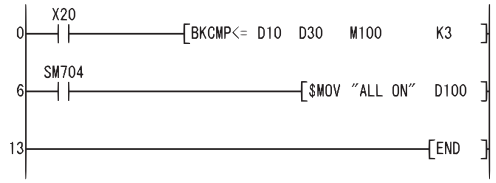
Step	Instruction	Device
0	LD	X1C
1	BKCMP<>P	K1000 D10 D0.4 K4
6	END	

[Operation]



• The following program compares, when X20 is turned ON, the data at D10 to D12 with the data at D30 to D32, and stores the operation result into the area starting from M100. The following program transfers the character string "ALL ON" to the area starting from D100 when all devices from M100 have reached the 1 "ON" state.

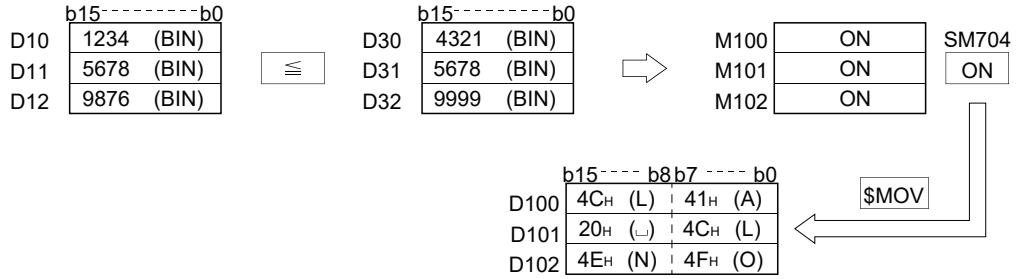
[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	X20
1	BKCMP<=	D10 D30 M100 K3
6	LD	SM704
7	$MOV	"ALL ON" D100
13	END	

[Operation]

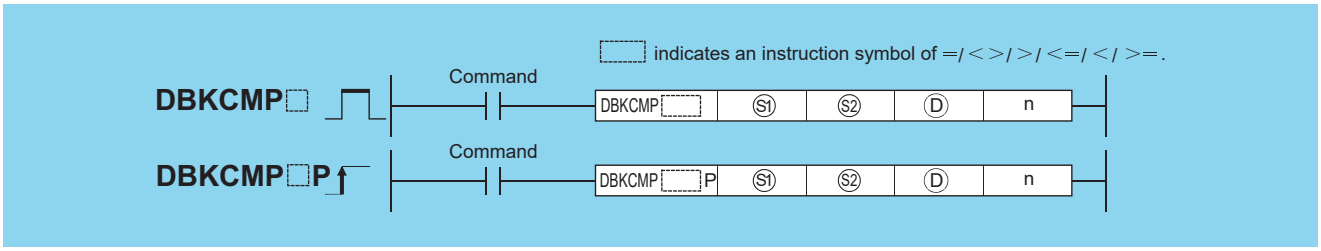


BIN 32-bit block data comparisons

DBKCMP□(P)

Basic | High performance | Process | Redundant | Ver. Universal | LCPU

- QnU(D)(H)CPU, QnUDE(H)CPU: the serial number (first five digits) is "10102" or later
- Q00UJCPU, Q00UCPU, Q01UCPU, QnUDVCPU, QnUDPVCPU: Supported

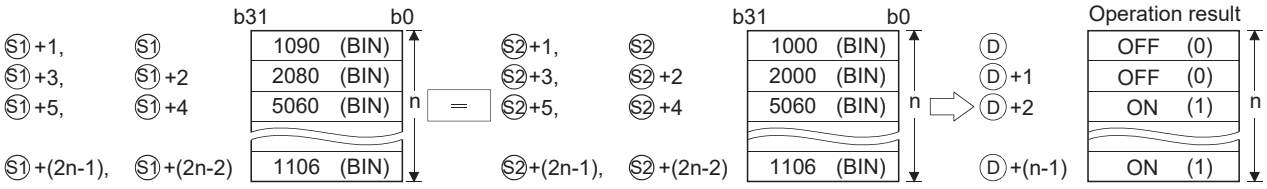


(S1): Data to be compared or head number of the devices where the data to be compared are stored (BIN 32 bits)
(S2): Head number of the devices where the comparison data are stored (BIN 32 bits)
(D): Head number of the devices where the comparison operation result will be stored (bits)
n: Number of comparison data blocks (BIN 16 bits)

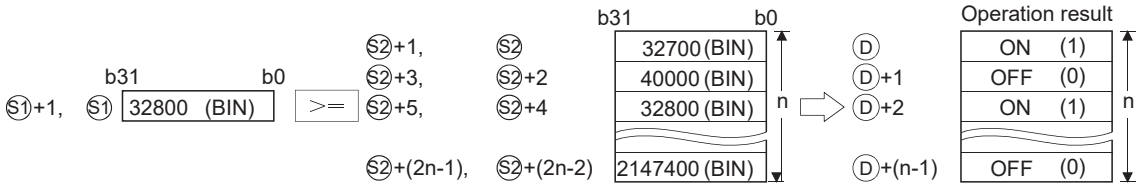
Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(S1)	—	○	○	—				○	—
(S2)	—	○	○	—				—	—
(D)	○	—	○	—				—	—
n	—	○	○	○				○	—

Processing details

- This instruction compares BIN 32-bit data stored in n-point devices starting from the device specified by (S1) with BIN 32-bit data stored in n-point devices starting from the device specified by a constant and (S2) and then stores the result into the nth device specified by (D) and up.
  - If the comparison condition has been met, the corresponding devices specified by (D) will be turned on.
  - If the comparison condition has not been met, the device designated by (D) will be turned OFF.



- The comparison operation is conducted in 32-bit units.
- The constant in the device specified by (S1) can be between -2147483648 and 2147483647 (BIN 32-bit data).



- Specify (D) out of the range of n-point devices starting from the device specified by (S1) and (S2).

• The results of the comparison operations for the individual instructions are as follows:

Instruction symbol	Condition	Comparison operation result	Instruction symbol	Condition	Comparison operation result
DBKCMP=	(S1)=(S2)	ON (1)	DBKCMP=	(S1)≠(S2)	OFF (0)
DBKCMP<>	(S1)≠(S2)		DBKCMP<>	(S1)=(S2)	
DBKCMP>	(S1)>(S2)		DBKCMP>	(S1)≤(S2)	
DBKCMP<=	(S1)≤(S2)		DBKCMP<=	(S1)>(S2)	
DBKCMP<	(S1)<(S2)		DBKCMP<	(S1)≥(S2)	
DBKCMP>=	(S1)≥(S2)		DBKCMP>=	(S1)<(S2)	

• If all comparison results stored into the devices starting from the device specified by (D) to nth device are on(1), or one of the results is off(2), the special relays will be on or off in accordance with the conditions as follows.

No.	Number	When all results of comparison operations are on(1)			When results of comparison operations have a result of off(0)		
		Initial execution/ Scan	Interrupt (other than I45)/Fixed scan execution	Interrupt(I45)	Initial execution/ Scan	Interrupt (other than I45)/Fixed scan execution	Interrupt(I45)
1	SM704	ON	ON	ON	OFF	OFF	OFF
2	SM716	ON	—	—	OFF	—	—
3	SM717	—	ON	—	—	OFF	—
4	SM718	—	—	ON	—	—	OFF

In a standby program, a special relay depending on the caller program turns on or off.

• If the value specified by n is 0, the instruction will be not processed.

Operation error

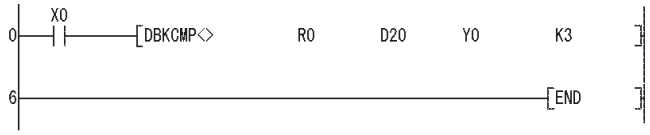
• In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4100	A negative value is specified for n.	—	—	—	—	○	○
4101	The points specified in n exceed those of the corresponding device specified in (S1), (S2), or (D). The ranges of devices starting from the one specified in (S1) and (D) overlap by n points. The ranges of devices starting from the one specified in (S2) and (D) overlap by n points.	—	—	—	—	○	○

Program example

• The following program compares the value data stored at R0 to R5 with the value data stored at D20 to D25, and then stores the operation result into Y0 to Y2, when M0 is turned on.

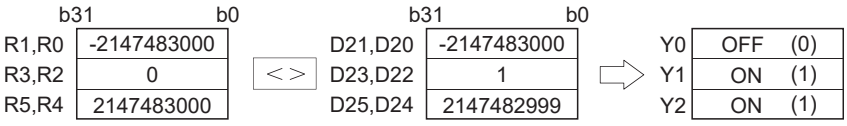
[Ladder Mode]



[List Mode]

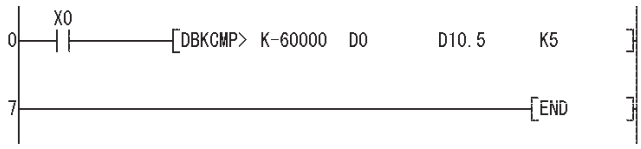
Step	Instruction	Device			
0	LD	X0			
1	DBKCMP<>	R0	D20	Y0	K3
6	END				

[Operation]



• The following program compares the constant with the value data stored at D0 to D9, and then stores the operation result into D10.5 to D10.9, when M0 is turned on.

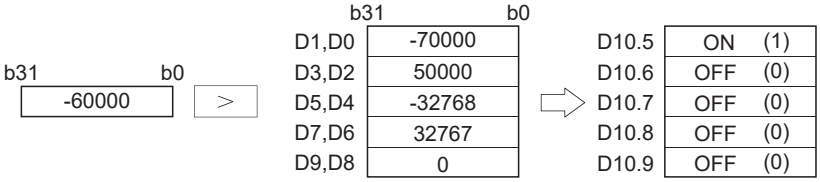
[Ladder Mode]



[List Mode]

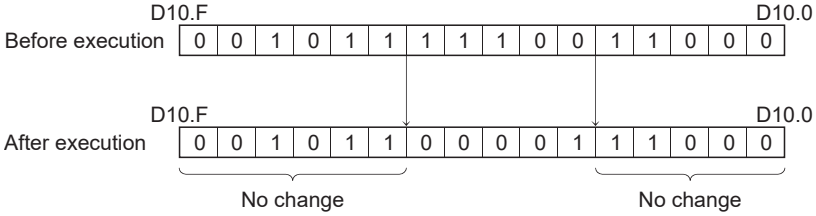
Step	Instruction	Device
0	LD	M0
1	DBKCMP>= K-60000 D0	D10.5 K5
7	END	

[Operation]



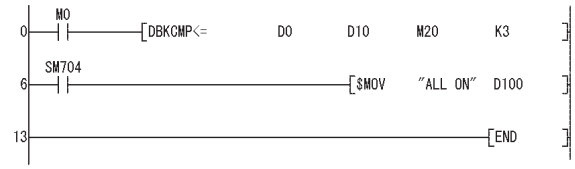
Point

When certain bits are specified in a word device, bits other than the certain bits that store the operation result do not change.



• The following program compares the value data stored at D0 to D5 with the value data stored at D10 to D15, and then stores the operation result into M20 to M22, when M0 is turned on. Also, the program transfers the character string "ALL ON" to D100 and up when all devices from M20 to M22 have reached the on status.

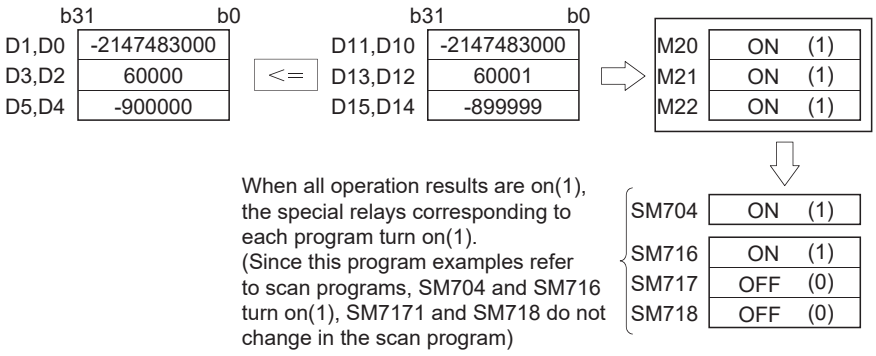
[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	M0
1	DBKCMP<=	D0 D10 M20 K3
6	LD	SM704
7	$MOV	"ALLON" D100
13	END	

[Operation]

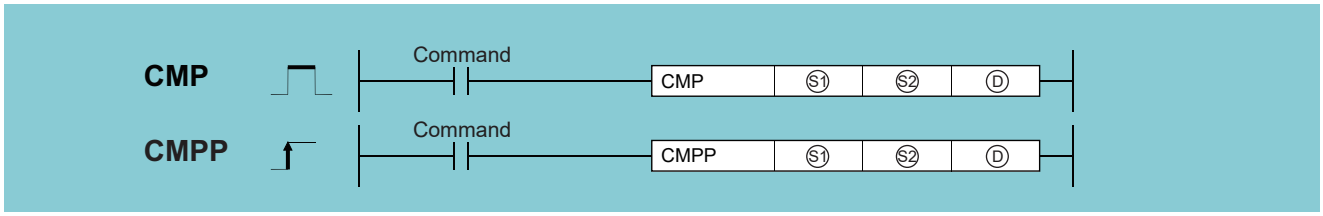


BIN 16-bit data comparisons (small, match, large)

CMP(P)

Basic | High performance | Process | Redundant | Universal (Ver.) | LCPU (Ver.)

- LCPU: The serial number (first five digits) is "16042" or later.
- QnUDVCPU, QnUDPVCPU: The serial number (first five digits) is "16043" or later.

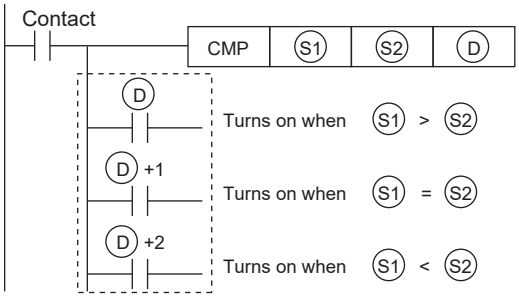


(S1): Comparison data or start number of the devices where the comparison data is stored (BIN 16 bits)
(S2): Comparison data or start number of the devices where the comparison data is stored (BIN 16 bits)
(D): Head number of the devices where the comparison result will be stored (bits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(S1)	—	○	○	—				○	—
(S2)	—	○	○	—				○	—
(D)	○	—	○	—				—	—

Processing details

- One of (D), (D)+1, and (D)+2 turns on depending on the results (small, match, large) after comparison data (S1) and comparison data (S2) are compared.



- When the contact turns off, (D) to (D)+2 hold the conditions immediately before the contact turns on and off.

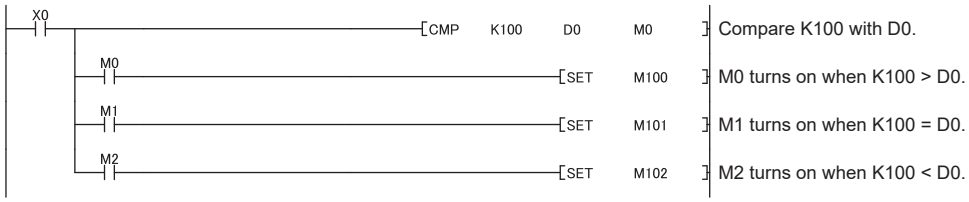
Operation error

- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4101	The range of 3 points from the device specified in (D) exceeds the range of the corresponding device.	—	—	—	—	○	○

Program example

- Program that compares the value of D0 with the constant value K100

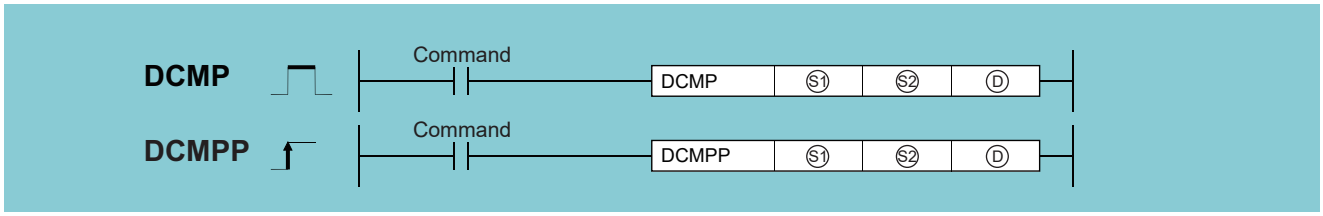


BIN 32-bit data comparisons (small, match, large)

DCMP(P)

Basic | High performance | Process | Redundant | Ver. Universal | Ver. LCPU

- LCPU: The serial number (first five digits) is "16042" or later.
- QnUDVCPU, QnUDPVCPU: The serial number (first five digits) is "16043" or later.

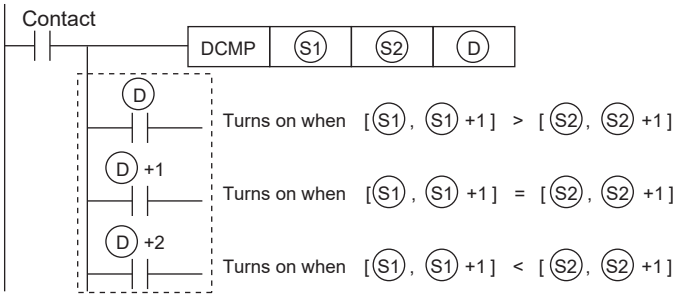


(S1): Comparison data or start number of the devices where the comparison data is stored (BIN 32 bits)
(S2): Comparison data or start number of the devices where the comparison data is stored (BIN 32 bits)
(D): Head number of the devices where the comparison result will be stored (bits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(S1)	—	○	○	—				○	—
(S2)	—	○	○	—				○	—
(D)	○	—	○	—				—	—

Processing details

- One of (D), (D)+1, and (D)+2 turns on depending on the results (small, match, large) after comparison data (S1), (S1)+1 and comparison data (S2), (S2)+1 are compared.



- When the contact turns off, (D) to (D)+2 hold the conditions immediately before the contact turns on and off.

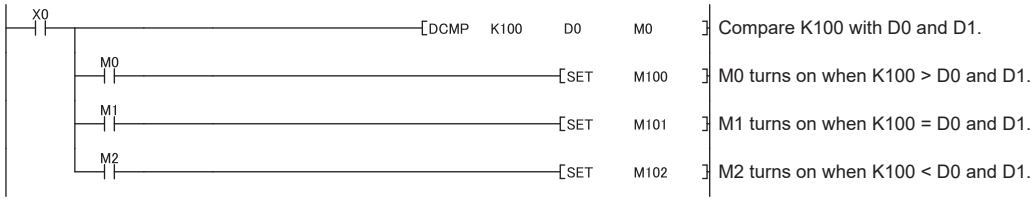
Operation error

- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4101	The range of 3 points from the device specified in (D) exceeds the range of the corresponding device.	—	—	—	—	○	○

Program example

- Program that compares the values of D0 and D1 with the constant value K100

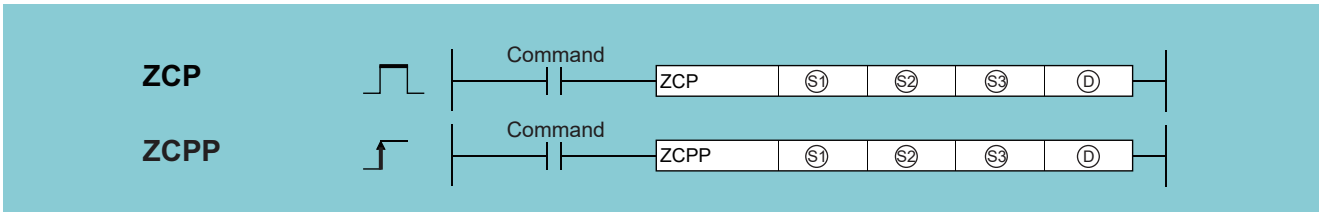


BIN 16-bit data band comparisons

ZCP(P)

Basic | High performance | Process | Redundant | Universal (Ver.) | LCPU (Ver.)

- LCPU: The serial number (first five digits) is "16042" or later.
- QnUDVCPU, QnUDPVCPU: The serial number (first five digits) is "16043" or later.

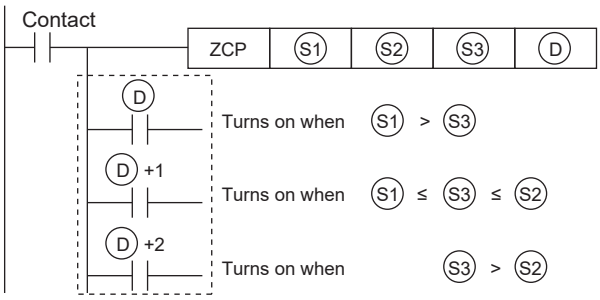


(S1): Start number of the devices where the lower limit value is stored (BIN 16 bits)
(S2): Start number of the devices where the upper limit value is stored (BIN 16 bits)
(S3): Head number of device where data for comparison or comparison data is stored (BIN 16-bit)
(D): Head number of device where the comparison results are stored (Bit)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(S1)	—	○	○	—				○	—
(S2)	—	○	○	—				○	—
(S3)	—	○	○	—				○	—
(D)	○	—	○	—				—	—

Processing details

- One of (D), (D)+1, and (D)+2 turns on depending on the results (small, within the band, large) after comparison data (S3) is compared with the lower limit value (S1) and upper limit value (S2).



- When the contact turns off, (D) to (D)+2 hold the conditions immediately before the contact turns on and off.

Operation error

- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

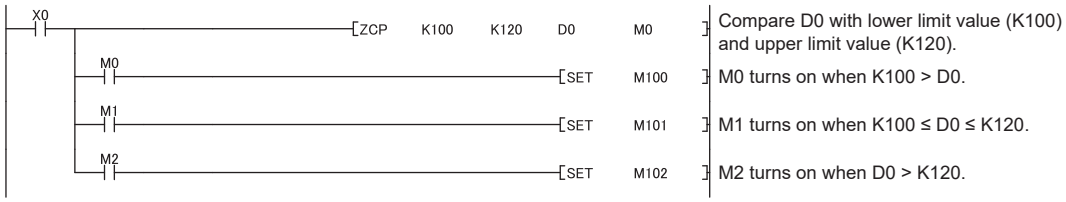
Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4101	The range of 3 points from the device specified in (D) exceeds the range of the corresponding device.	—	—	—	—	○	○

Precautions

- Set a value smaller than the upper limit value (S2) as the lower limit value (S1). If the lower limit value (S1) is larger than the upper limit value (S2), the upper limit value (S2) will be handled as the same value as the lower limit value (S1).

Program example

- Program that compares the band of the value of D0 with the lower limit value (K100) and the upper limit value (K120)

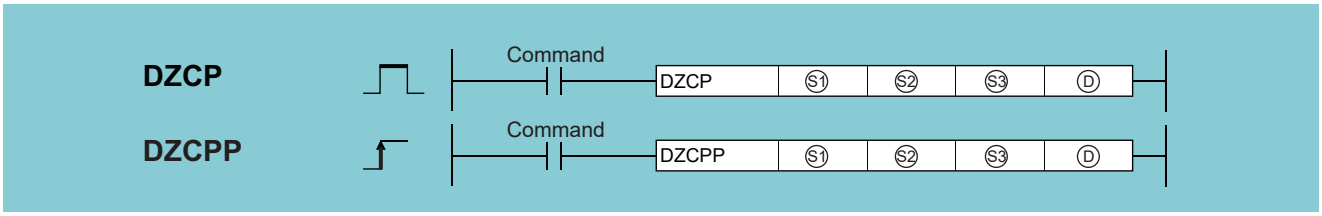


BIN 32-bit data band comparisons

DZCP(P)

Basic | High performance | Process | Redundant | Universal (Ver.) | LCPU (Ver.)

- LCPU: The serial number (first five digits) is "16042" or later.
- QnUDVCPU, QnUDPVCPU: The serial number (first five digits) is "16043" or later.

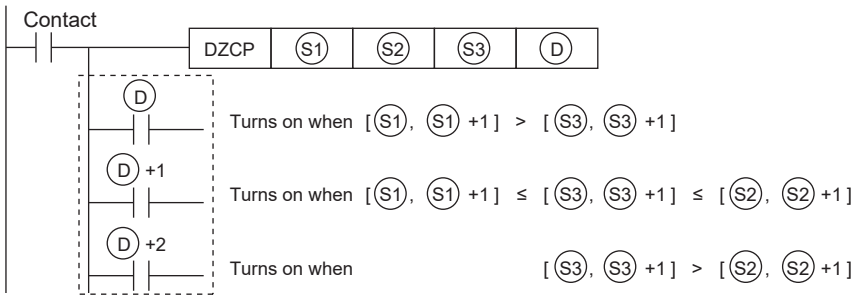


(S1): Start number of the devices where the lower limit value is stored (BIN 32 bits)
(S2): Start number of the devices where the upper limit value is stored (BIN 32 bits)
(S3): Head number of device where data for comparison or comparison data is stored (BIN 32-bit)
(D): Head number of devices where the comparison results are stored (Bit)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(S1)	—	○	○	—				○	—
(S2)	—	○	○	—				○	—
(S3)	—	○	○	—				○	—
(D)	○	—	○	—				—	—

Processing details

- One of (D), (D)+1, and (D)+2 turns on depending on the results (small, within the band, large) after comparison data (S3), (S3)+1 is compared with the lower limit values (S1), (S1)+1 and the upper limit values (S2), (S2)+1.



- When the contact turns off, "(D) to (D)+2" hold the conditions immediately before the contact turns on and off.

Operation error

- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4101	The range of 3 points from the device specified in (D) exceeds the range of the corresponding device.	—	—	—	—	○	○

Precautions

- Set values smaller than the upper limit values (S2), (S2)+1 as the lower limit values (S1), (S1)+1. If the lower limit values (S1), (S1)+1 are larger than the upper limit values (S2), (S2)+1, the upper limit values (S2), (S2)+1 will be handled as the same values as the lower limit values (S1), (S1)+1.

Program example

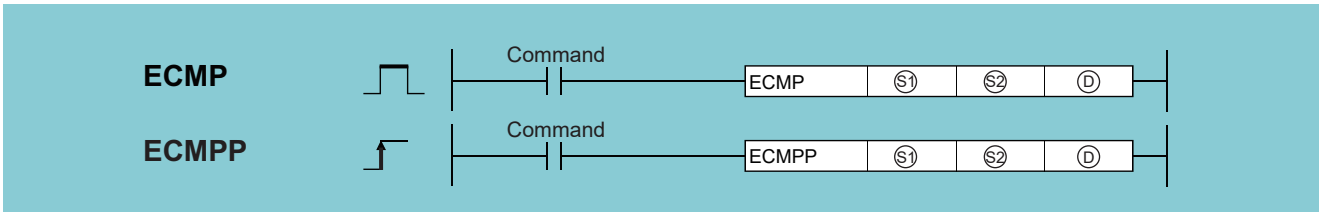
- Program that compares the bands of the values of D0 and D1 with the lower limit value (K100) and upper limit value (K120)

Floating point comparisons (single precision)

ECMP(P)

Basic | High performance | Process | Redundant | Ver. Universal | Ver. LCPU

• QnUDVCPU, QnUDPVCPU, LCPU: The serial number (first five digits) is "16112" or later.

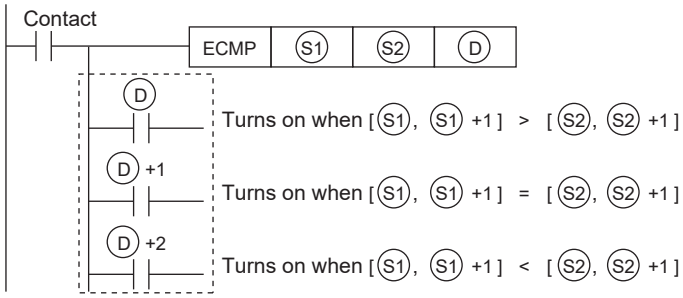


(S1): Comparison data or start number of the devices where the comparison data is stored (real number)
(S2): Comparison data or start number of the devices where the comparison data is stored (real number)
(D): Start bit device number to which the comparison result is output (Bit)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant E	Others
	Bit	Word		Bit	Word				
(S1)	—	○	○	—				○	—
(S2)	—	○	○	—				○	—
(D)	○	—	○	—				—	—

Processing details

• One of (D), (D)+1, and (D)+2 turns on depending on the results (small, match, large) after comparison data "(S1) and (S1)+1" and comparison data "(S2) and (S2)+1" are compared.



• When the contact turns off, "(D) to (D)+2" hold the conditions immediately before the contact turns on and off.

Precautions

• The three points starting from the device specified by (D) are used. Therefore, do not overlap any device used by others.

Operation error

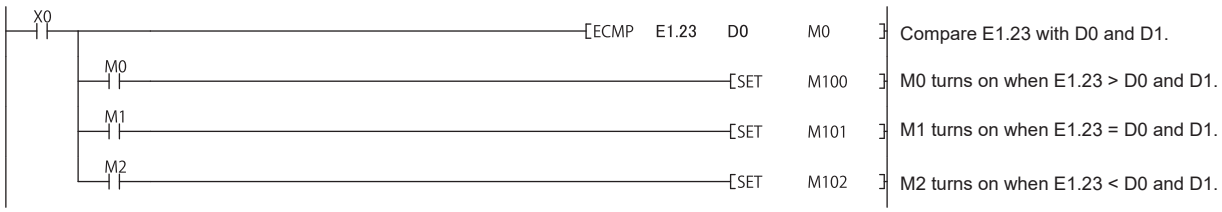
• In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4101	The range of three points from the device specified by (D) exceeds the range of the corresponding device.	—	—	—	—	○	○
4140	The specified device value is -0, a subnormal number, NaN (not a number), or ±∞.	—	—	—	—	○	○

Program example

- Program that compares the values of D0 and D1 with the real number 1.23

[Ladder Mode]

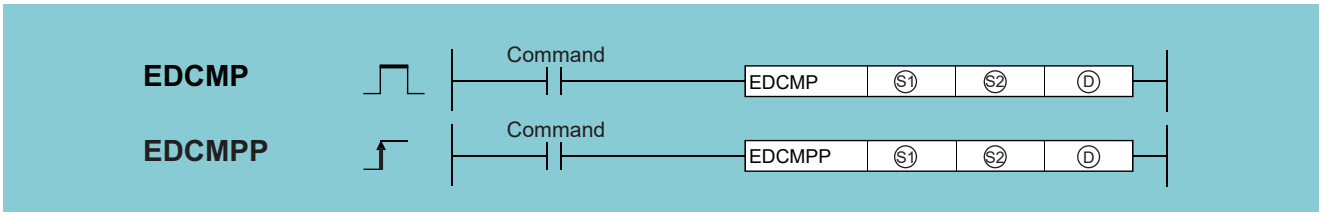


Floating point comparisons (double precision)

EDCMP(P)

Basic | High performance | Process | Redundant | Universal (Ver.) | LCPU (Ver.)

• QnUDVCPU, QnUDPVCPU, LCPU: The serial number (first five digits) is "16112" or later.

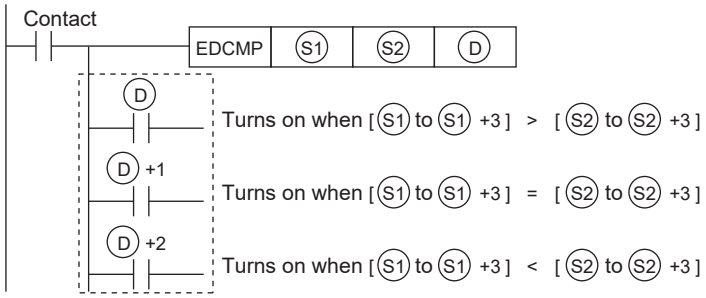


(S1): Comparison data or start number of the devices where the comparison data is stored (real number)
(S2): Comparison data or start number of the devices where the comparison data is stored (real number)
(D): Start bit device number to which the comparison result is output (Bit)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant E	Others
	Bit	Word		Bit	Word				
(S1)	—	○	○	—				○	—
(S2)	—	○	○	—				○	—
(D)	○	—	○	—				—	—

Processing details

• One of (D), (D)+1, and (D)+2 turns on depending on the results (small, match, large) after comparison data "(S1) to (S1)+3" and comparison data "(S2) to (S2)+3" are compared.



• When the contact turns off, "(D) to (D)+2" hold the conditions immediately before the contact turns on and off.

Precautions

• The three points starting from the device specified by (D) are used. Therefore, do not overlap any device used by others.

Operation error

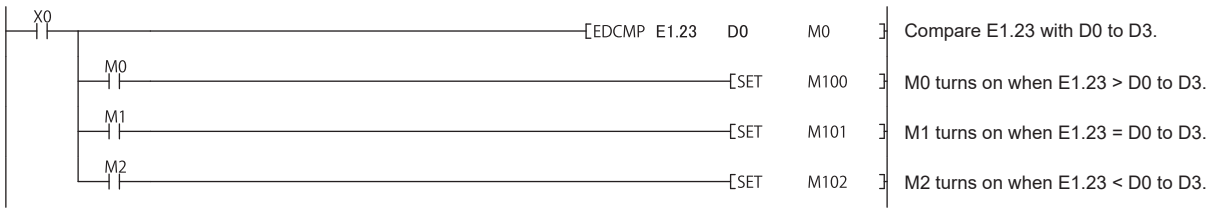
• In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4101	The range of three points from the device specified by (D) exceeds the range of the corresponding device.	—	—	—	—	○	○
4140	The specified device value is -0, a subnormal number, NaN (not a number), or ±∞.	—	—	—	—	○	○

Program example

- Program that compares the values of D0 to D3 with the real number 1.23

[Ladder Mode]

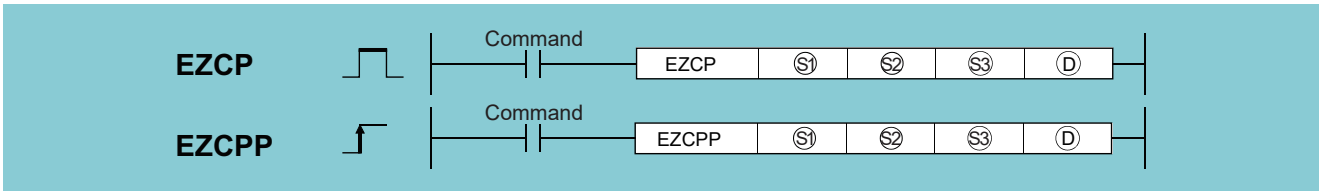


Floating point band comparisons (single precision)

EZCP(P)

Basic | High performance | Process | Redundant | Ver. Universal | Ver. LCPU

• QnUDVCPU, QnUDPVCPU, LCPU: The serial number (first five digits) is "16112" or later.

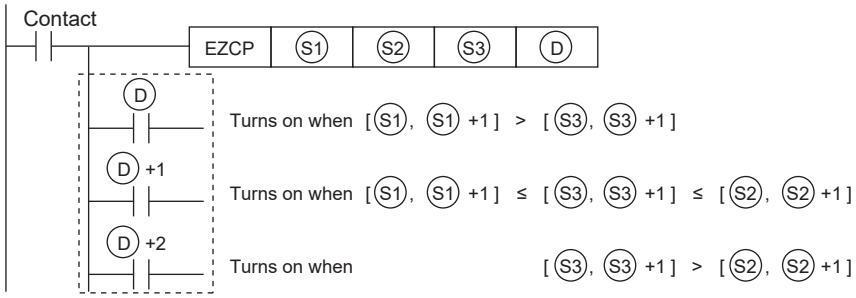


(S1): Start number of the devices where the lower limit value is stored (real number)
(S2): Start number of the devices where the upper limit value is stored (real number)
(S3): Data to be compared or start number of the device where the data to be compared is stored (real number)
(D): Start bit device number to which the comparison result is output (Bit)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant E	Others
	Bit	Word		Bit	Word				
(S1)	—	○	○	—				○	—
(S2)	—	○	○	—				○	—
(S3)	—	○	○	—				○	—
(D)	○	—	○	—				—	—

Processing details

• One of (D), (D)+1, and (D)+2 turns on depending on the results (small, within the band, large) after the lower limit values "(S1) and (S1)+1" and the upper limit values "(S2) and (S2)+1" are compared with the comparison data "(S3) and (S3)+1".



• When the contact turns off, "(D) to (D)+2" hold the conditions immediately before the contact turns on and off.

Precautions

• The three points starting from the device specified by (D) are used. Therefore, do not overlap any device used by others.
• Set values smaller than the upper limit values (S2), (S2)+1 as the lower limit values (S1), (S1)+1. If the lower limit values (S1), (S1)+1 are larger than the upper limit values (S2), (S2)+1, the upper limit values (S2), (S2)+1 will be handled as the same values as the lower limit values (S1), (S1)+1.

Operation error

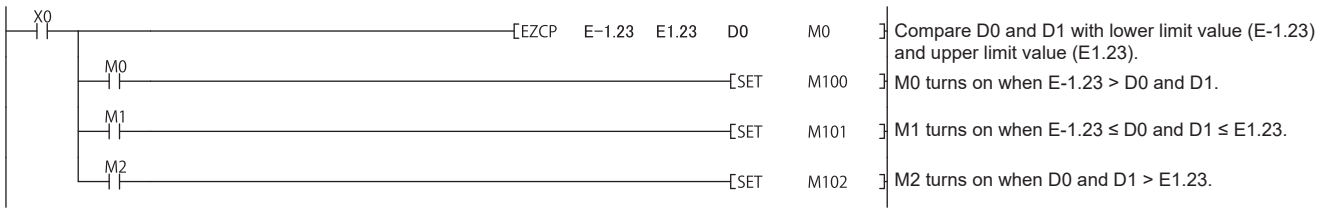
• In any of the following cases, an operation error occurs, the error flag (SM0) turns on, and an error code is stored in SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4101	The range of three points from the device specified by (D) exceeds the range of the corresponding device.	—	—	—	—	○	○
4140	The specified device value is -0, a subnormal number, NaN (not a number), or ±∞.	—	—	—	—	○	○

Program example

- Program that compares the values of D0 and D1 with the lower limit value (E-1.23) and upper limit value (E1.23)

[Ladder Mode]

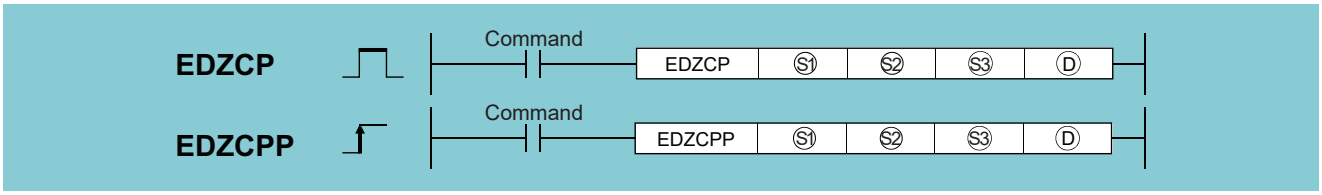


Floating point band comparisons (double precision)

EDZCP(P)

Basic | High performance | Process | Redundant | Ver. Universal | Ver. LCPU

• QnUDVCPU, QnUDPVCPU, LCPU: The serial number (first five digits) is "16112" or later.

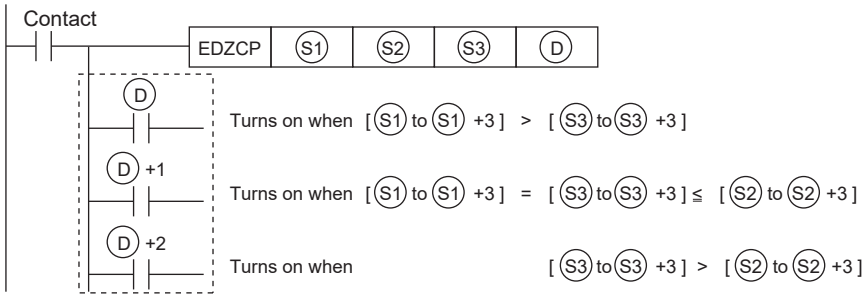


(S1): Start number of the devices where the lower limit value is stored (real number)
(S2): Start number of the devices where the upper limit value is stored (real number)
(S3): Data to be compared or start number of the device where the data to be compared is stored (real number)
(D): Start bit device number to which the comparison result is output (Bit)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant E	Others
	Bit	Word		Bit	Word				
(S1)	—	○	○	—				○	—
(S2)	—	○	○	—				○	—
(S3)	—	○	○	—				○	—
(D)	○	—	○	—				—	—

Processing details

• One of (D), (D)+1, and (D)+2 turns on depending on the results (small, within the band, large) after the lower limit values "(S1) to (S1)+3" and the upper limit values "(S2) to (S2)+3" are compared with the comparison data "(S3) to (S3)+3".



• When the contact turns off, "(D) to (D)+2" hold the conditions immediately before the contact turns on and off.

Precautions

• The three points starting from the device specified by (D) are used. Therefore, do not overlap any device used by others.
• Set values smaller than the upper limit values (S2) to (S2)+3 as the lower limit values (S1) to (S1)+3. If the lower limit values (S1) to (S1)+3 are larger than the upper limit values (S2) to (S2)+3, the upper limit values (S2) to (S2)+3 will be handled as the same values as the lower limit values (S1) to (S1)+3.

Operation error

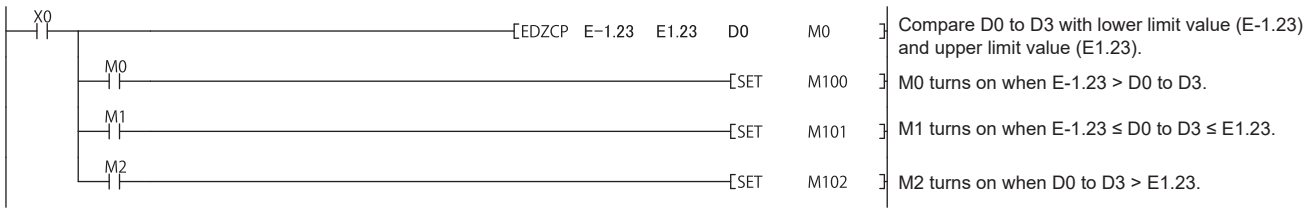
• In any of the following cases, an operation error occurs, the error flag (SM0) turns on, and an error code is stored in SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4101	The range of three points from the device specified by (D) exceeds the range of the corresponding device.	—	—	—	—	○	○
4140	The specified device value is -0, a subnormal number, NaN (not a number), or ±∞.	—	—	—	—	○	○

Program example

- Program that compares the values of D0 to D3 with the lower limit value (E-1.23) and upper limit value (E1.23).

[Ladder Mode]

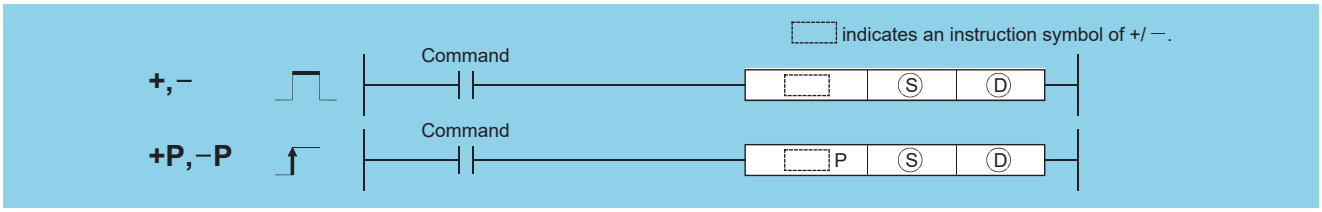


6.2 Arithmetic Operation Instructions

BIN 16-bit addition and subtraction operations

+(P), -(P) [When two data are set]

Basic | High performance | Process | Redundant | Universal | LCPU



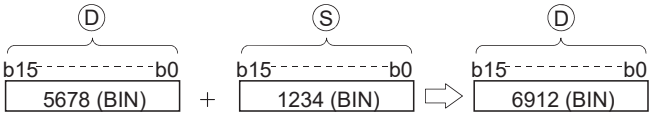
(S): Data for adding/subtracting or head number of the devices where the data for adding/subtracting is stored (BIN 16 bits)
(D): Head number of the devices where the data to be added to/subtracted from is stored (BIN 16 bits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(S)	○							○	—
(D)	○							—	—

Processing details

■+

- Adds 16-bit BIN data designated by (D) to 16-bit BIN data designated by (S) and stores the result of the addition at the device designated by (D).



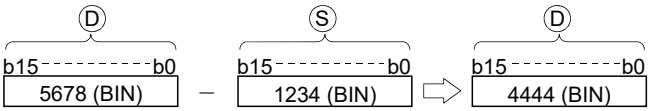
- Values for (S) and (D) can be designated between -32768 and 32767 (BIN, 16 bits).
- The judgment of whether data is positive or negative is made by the most significant bit (b15).
  - 0 ··· Positive
  - 1 ··· Negative
- The following will happen when an underflow or overflow is generated in an operation result: The carry flag (SM700) in this case does not turn ON.

· K32767 (7FFFH) +K2 (0002H) ⟶ K−32767 (8001H) ····· Since bit 15 value is "1", result of operation takes a negative value.

· K−32768 (8000H) +K−2 (FFFEH) ⟶ K32766 (7FFEH) ····· Since bit 15 value is "0", result of operation takes a positive value.

■-

- Subtracts 16-bit BIN data designated by (D) from 16-bit BIN data designated by (S) and stores the result of the subtraction at the device designated by (D).



- Values for (S) and (D) can be designated between -32768 and 32767 (BIN, 16 bits).
- The judgment of whether data is positive or negative is made by the most significant bit (b15).
  - 0 ··· Positive
  - 1 ··· Negative
- The following will happen when an underflow or overflow is generated in an operation result: The carry flag (SM700) in this case does not turn ON.

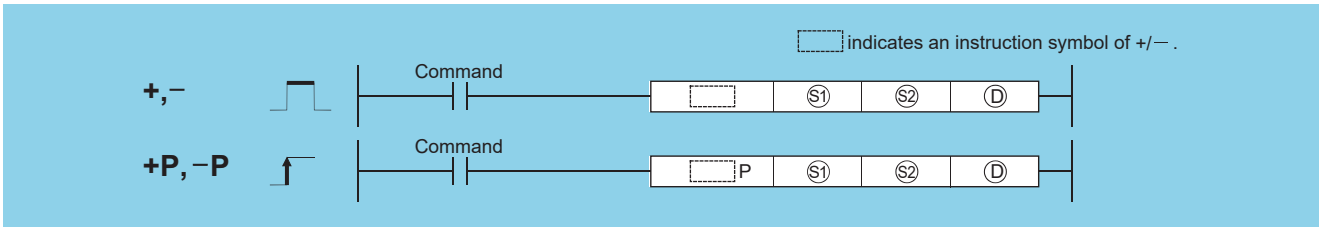
· K−32768 (8000H) − K2 (0002H) ⟶ K32766 (7FFEH) ····· Since bit 15 value is "0", result of operation takes a positive value.

· K32767 (7FFFH) − K−2 (FFFEH) ⟶ K−32767 (8001H) ····· Since bit 15 value is "1", result of operation takes a negative value.

Operation error

- There is no operation error in the +(P) or -(P) instruction.

+(P), -(P) [When three data are set]



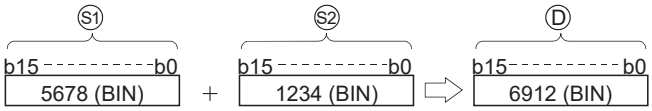
(S1): Data to be added to/subtracted from or head number of the devices where the data to be added to/subtracted from is stored (BIN 16 bits)
(S2): Data for adding/subtracting or head number of the devices where the data for adding/subtracting is stored (BIN 16 bits)
(D): Head number of the devices where the operation result will be stored (BIN 16 bits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(S1)	○							○	—
(S2)	○							○	—
(D)	○							—	—

Processing details

■+

- Adds 16-bit BIN data designated by (S1) to 16-bit BIN data designated by (S2) and stores the result of the addition at the device designated by (D).



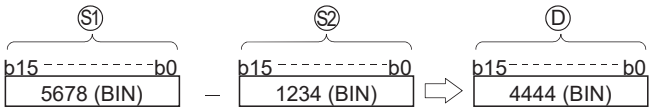
- Values for (S1), (S2) and (D) can be designated between (D) -32768 and 32767 (BIN, 16 bits).
- The judgment of whether data is positive or negative is made by the most significant bit (b15).
  - 0 ··· Positive
  - 1 ··· Negative
- The following will happen when an underflow or overflow is generated in an operation result: The carry flag (SM700) in this case does not turn ON.

· K32767 (7FFFH) +K2 (0002H) → K−32767 (8001H) ····· Since bit 15 value is "1", result of operation takes a negative value.

· K−32768 (8000H) +K−2 (FFFEH) → K32766 (7FFEH) ····· Since bit 15 value is "0", result of operation takes a positive value.

■-

- Subtracts 16-bit BIN data designated by (S1) from 16-bit BIN data designated by (S2) and stores the result of the subtraction at the device designated by (D).



- Values for (S1), (S2) and (D) can be designated between (D) -32768 and 32767 (BIN, 16 bits).
- The judgment of whether data is positive or negative is made by the most significant bit (b15).
  - 0 ··· Positive
  - 1 ··· Negative
- The following will happen when an underflow or overflow is generated in an operation result: The carry flag (SM700) in this case does not turn ON.

· K−32768 (8000H) −K2 (0002H) → K32766 (7FFEH) ····· Since bit 15 value is "0", result of operation takes a positive value.

· K32767 (7FFFH) −K−2 (FFFEH) → K−32767 (8001H) ····· Since bit 15 value is "1", result of operation takes a negative value.

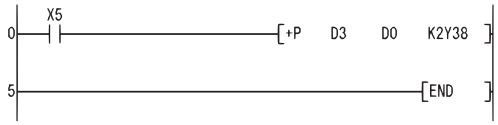
Operation error

- There is no operation error in the +(P) or -(P) instruction.

Program example

- The following program adds, when X5 is turned ON, the data at D3 and D0 and outputs the operation result at Y38 to Y3F.

[Ladder Mode]

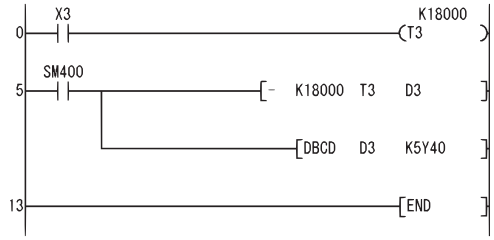


[List Mode]

Step	Instruction	Device
0	LD	X5
1	+P	D3 D0 K2Y38
5	END	

- The following program outputs the difference between the set value for timer T3 and its present value in BCD to Y40 to Y53.

[Ladder Mode]



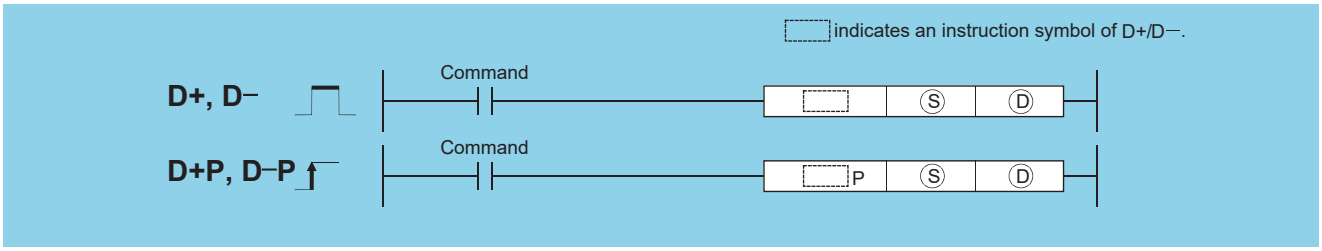
[List Mode]

Step	Instruction	Device
0	LD	X3
1	OUT	T3 K18000
5	LD	SM400
6	-	K18000 T3 D3
10	DBCD	D3 K5Y40
13	END	

BIN 32-bit addition and subtraction operations

D+(P), D-(P) [When two data are set]

Basic | High performance | Process | Redundant | Universal | LCPU



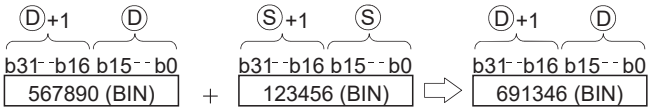
(S): Data for adding/subtracting or head number of the devices where the data for adding/subtracting is stored (BIN 32 bits)
(D): Head number of the devices where the data to be added to/subtracted from is stored (BIN 32 bits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(S)	○							○	—
(D)	○							—	—

Processing details

■D+

- Adds 32-bit BIN data designated by (D) to 32-bit BIN data designated by (S), and stores the result of the addition at the device designated by (D).



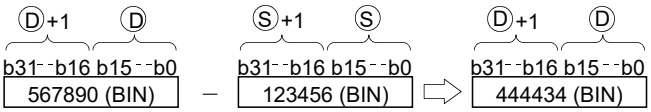
- The values for (S) and (D) can be designated at between -2147483648 and 2147483647 (BIN 32 bits).
- Judgment of whether the data is positive or negative is made on the basis of the most significant bit (b31).
  - 0 ··· Positive
  - 1 ··· Negative
- The following will happen when an underflow or overflow is generated in an operation result: The carry flag (SM700) in this case does not turn ON.

· K2147483647 (7FFFFFFFH) +K2 (00000002H) → K−2147483647 (80000001H) ····· Since bit 31 value is "1", result of operation takes a negative value.

· K−2147483648 (80000000H) +K−2 (FFFFFFFEH) → K2147483646 (7FFFFFFEH) ····· Since bit 31 value is "0", result of operation takes a positive value.

■D-

- Subtracts 32-bit BIN data designated by (D) from 32-bit BIN data designated by (S) and stores the result of the subtraction at the device designated by (D).



- The values for (S) and (D) can be designated at between -2147483648 and 2147483647 (BIN 32 bits).
- Judgment of whether the data is positive or negative is made on the basis of the most significant bit (b31).
  - 0 ··· Positive
  - 1 ··· Negative
- The following will happen when an underflow or overflow is generated in an operation result: The carry flag (SM700) in this case does not turn ON.

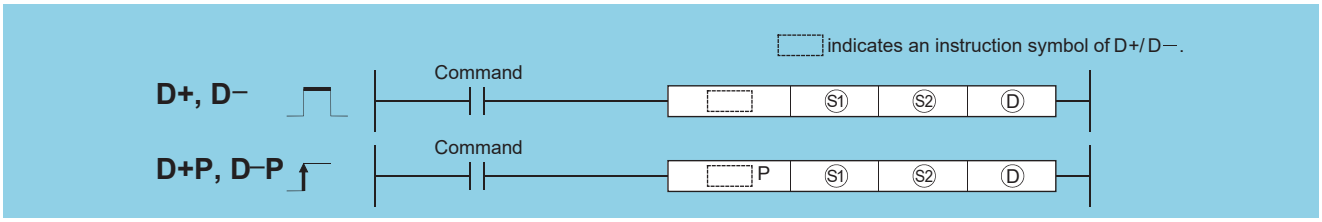
· K−2147483648 − K2 ⟶ K2147483646 ····· Since bit 31 value is "0", result of operation takes a positive value.
(80000000H) (00000002H) (7FFFFFFEH)

· K2147483647 − K−2 ⟶ K−2147483647 ····· Since bit 31 value is "1", result of operation takes a negative value.
(80000000H) (FFFFFFFEH) (80000001H)

Operation error

- There is no operation error in the D+(P) or D-(P) instruction.

D+(P), D-(P) [When three data are set]



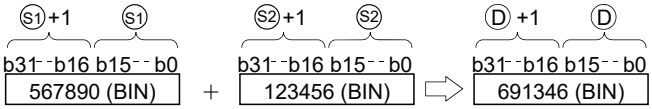
(S1): Data to be added to/subtracted from or head number of the devices where the data to be added to/subtracted from is stored (BIN 32 bits)
(S2): Data for adding/subtracting or head number of the devices where the data for adding/subtracting is stored (BIN 32 bits)
(D): Head number of the devices where the multiplication/division operation result will be stored (BIN 32 bits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(S1)	○							○	—
(S2)	○							○	—
(D)	○							—	—

Processing details

■D+

- Adds 32-bit BIN data designated by (S1) to 32-bit BIN data designated by (S2), and stores the result of the addition at the device designated by (D).



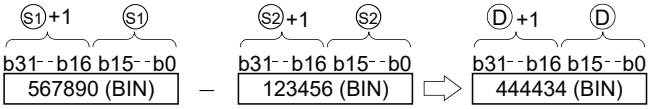
- The values for (S1), (S2) and (D) can be designated at between -2147483648 and 2147483647 (BIN 32 bits).
- Judgment of whether the data is positive or negative is made on the basis of the most significant bit (b31).
  - 0 ··· Positive
  - 1 ··· Negative
- The following will happen when an underflow or overflow is generated in an operation result: The carry flag (SM700) in this case does not turn ON.

· K2147483647 (7FFFFFFFH) +K2 (00000002H) → K−2147483647 (80000001H) ··· Since bit 31 value is "1", result of operation takes a negative value.

· K−2147483648 (80000000H) +K−2 (FFFFFFFEH) → K2147483646 (7FFFFFFEH) ····· Since bit 31 value is "0", result of operation takes a positive value.

■D-

- Subtracts 32-bit BIN data designated by (S1) from 32-bit BIN data designated by (S2) and stores the result of the subtraction at the device designated by (D).



- The values for (S1), (S2) and (D) can be designated at between -2147483648 and 2147483647 (BIN 32 bits).
- Judgment of whether the data is positive or negative is made on the basis of the most significant bit (b31).
  - 0 ··· Positive
  - 1 ··· Negative
- The following will happen when an underflow or overflow is generated in an operation result: The carry flag (SM700) in this case does not turn ON.

· K−2147483648 (80000000H) − K2 (00000002H) → K2147483646 (7FFFFFFEH) ····· Since bit 31 value is "0", result of operation takes a positive value.

· K2147483647 (7FFFFFFFH) − K−2 (FFFFFFFEH) → K−2147483647 (80000001H) ··· Since bit 31 value is "1", result of operation takes a negative value.

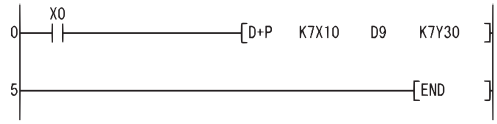
Operation error

- There is no operation error in the D+(P) or D-(P) instruction.

Program example

- The following program adds 28-bit data from X10 to X2B to the data at D9 and D10 when X0 goes ON, and outputs the result of the operation to Y30 to Y4B.

[Ladder Mode]

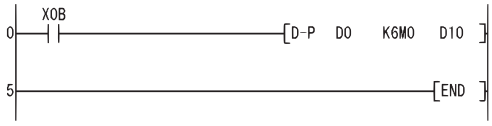


[List Mode]

Step	Instruction	Device
0	LD	X0
1	D+P	K7X10 D9 K7Y30
5	END	

- The following program subtracts the data from M0 to M23 from the data at D0 and D1 when XB goes ON, and stores the result at D10 and D11.

[Ladder Mode]



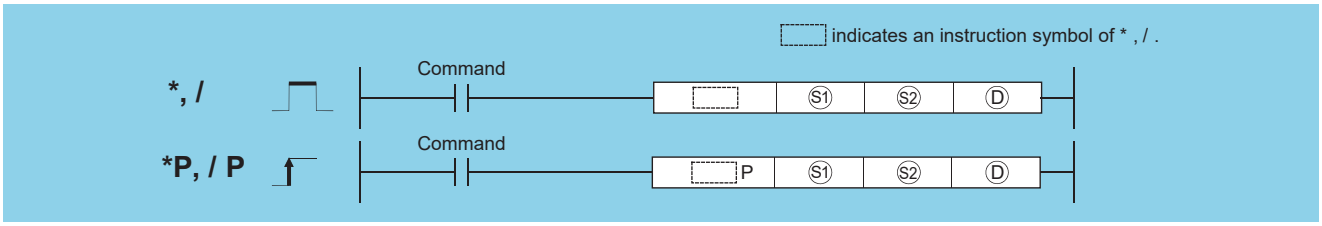
[List Mode]

Step	Instruction	Device
0	LD	X0B
1	D-P	D0 K6M0 D10
5	END	

BIN 16-bit multiplication and division operations

*(P), /(P)

Basic | High performance | Process | Redundant | Universal | LCPU



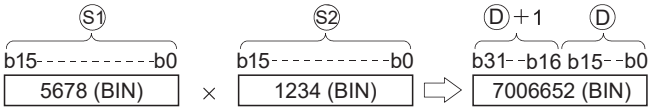
(S1): Data to be multiplied/divided or head number of the devices where the data to be multiplied/divided is stored (BIN 16 bits)
(S2): Data for multiplying/dividing or head number of the devices where the data for multiplying/dividing is stored (BIN 16 bits)
(D): Head number of the devices where the multiplication/division operation result will be stored (BIN 32 bits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(S1)	○							○	—
(S2)	○							○	—
(D)	○							—	—

Processing details

■*

- Multiplies BIN 16-bit data designated by (S1) and BIN 16-bit data designated by (S2), and stores the result in the device designated by (D).



- If (D) is a bit device, designation is made from the lower bits.

Ex.

K1 ··· Lower 4 bits (b0 to b3)

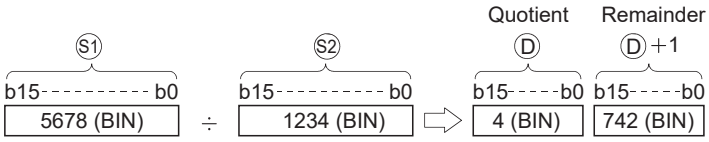
K4 ··· Lower 16 bits (b0 to b15)

K ··· 32 bits (b0 to b31)

- Values for (S1) and (S2) can be designated between -32768 and 32767 (BIN, 16 bits).
- Judgments whether (S1), (S2), and (D) are positive or negative are made on the basis of the most significant bit (b15 for (S1), and (S2), for (D) and b31).
  - 0 ··· Positive
  - 1 ··· Negative

■/

- Divides BIN 16-bit data designated by (S1) and BIN 16-bit data designated by (S2), and stores the result in the device designated by (D).



- If a word device has been used, the result of the division operation is stored as 32 bits, and both the quotient and remainder are stored; if a bit device has been used, 16 bits are used and only the quotient is stored.
  - Quotient ··· Stored at the lower 16 bits.
  - Remainder ··· Stored at the upper 16 bits (Stored only when using a word device).
- Values for (S1) and (S2) can be specified between the range of -32768 and 32767 (BIN 16 bits).
- Judgment whether values for (S1), (S2), (D) and (D)+1 are positive or negative is made on the basis of the most significant bit (b15). (Sign is attached to both the quotient and remainder.)
  - 0 ··· Positive
  - 1 ··· Negative

Operation error

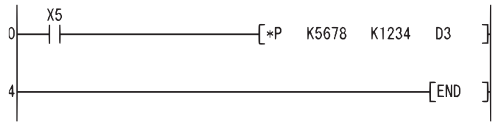
- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4100	The divisor is 0.	○	○	○	○	○	○

Program example

- The following program multiplies "5678" by "1234" in BIN and stores the result at D3 and D4 when X5 turns ON.

[Ladder Mode]

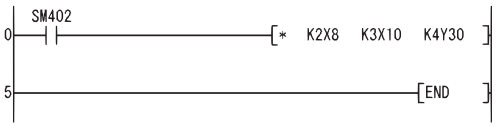


[List Mode]

Step	Instruction	Device
0	LD	X5
1	*P	K5678 K1234 D3
4	END	

- The following program multiplies BIN data at X8 to XF by BIN data at X10 to X1B, and outputs the result of the multiplication to Y30 to Y3F.

[Ladder Mode]

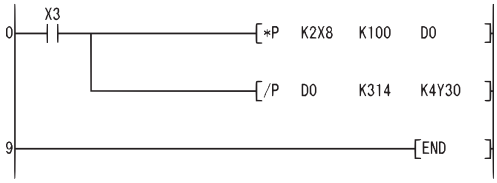


[List Mode]

Step	Instruction	Device
0	LD	SM402
1	*	K2X8 K3X10 K4Y30
5	END	

- The following program divides, when X3 is turned ON, the data at X8 to XF by 3.14 and outputs the operation result at Y30 to Y3F.

[Ladder Mode]



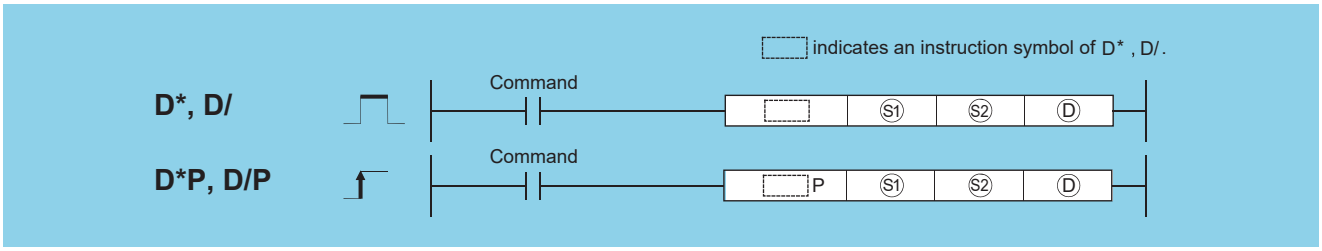
[List Mode]

Step	Instruction	Device
0	LD	X3
1	*P	K2X8 K100 D0
5	/P	D0 K314 K4Y30
9	END	

BIN 32-bit multiplication and division operations

D*(P), D/(P)

Basic | High performance | Process | Redundant | Universal | LCPU



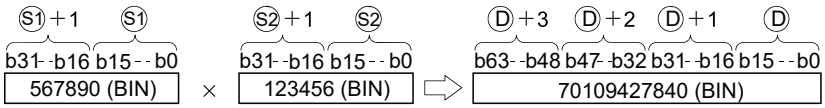
(S1): Data to be multiplied/divided or head number of the devices where the data to be multiplied/divided is stored (BIN 32 bits)
(S2): Data for multiplying/dividing or head number of the devices where the data for multiplying/dividing is stored (BIN 32 bits)
(D): Head number of the devices where the multiplication/division operation result will be stored (BIN 64 bits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(S1)	○			○					—
(S2)	○			○					—
(D)	○			—					—

Processing details

■D*

- Multiplies BIN 32-bit data designated by (S1) and BIN 32-bit data designated by (S2), and stores the result in the device designated by (D).



- If (D) is a bit device, only the lower 32 bits of the multiplication result will be considered, and the upper 32 bits cannot be designated.

Ex.

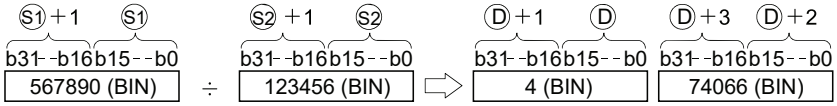
K1 ··· Lower 4 bits (b0 to b3)
K4 ··· Lower 16 bits (b0 to b15)
K8 ··· Lower 32 bits (b0 to b31)

If the upper 32 bits of the bit device are required for the result of the multiplication operation, first temporarily store the data in a word device, then transfer the word device data to the bit device by designating ((D)+2) and ((D)+3) data.

- The values for (S1) and (S2) can be designated at between -2147483648 and 2147483647 (BIN 32 bits).
- Judgments whether (S1), (S2), and (D) are positive or negative are made on the basis of the most significant bit (b31 for (S1) and (S2), b63 for (D)).
  - 0 ··· Positive
  - 1 ··· Negative

■D/

- Divides BIN 32-bit data designated by (S1) and BIN 32-bit data designated by (S2), and stores the result in the device designated by (D).



- With a word device, the division operation result is stored in 64 bits and both the quotient and remainder are stored. With a bit device, only the quotient is stored as the operation result in 32 bits.
  - Quotient ··· Stored at the lower 32 bits.
  - Remainder ··· Stored at the upper 32 bits (Stored only when using a word device).
- The values for (S1) and (S2) can be designated at between -2147483648 and 2147483647 (BIN 32 bits).
- Judgment whether values for (S1), (S2), (D) and (D)+2 are positive or negative is made on the basis of the most significant bit (b31). (Sign is attached to both the quotient and remainder.)
  - 0 ··· Positive
  - 1 ··· Negative

Operation error

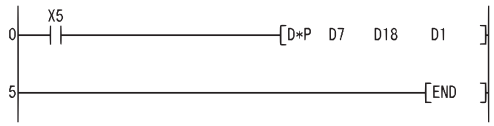
- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4100	The divisor is 0.	○	○	○	○	○	○

Program example

- The following program multiplies the BIN data at D7 and D8 by the BIN data at D18 and D19 when X5 is ON, and stores the result at D1 to D4.

[Ladder Mode]

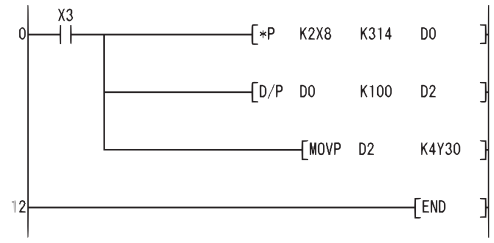


[List Mode]

Step	Instruction	Device
0	LD	X5
1	D*P	D7 D18 D1
5	END	

- The following program outputs the value resulting when the data at X8 to XF is multiplied by 3.14 to Y30 to Y3F when X3 is ON.

[Ladder Mode]



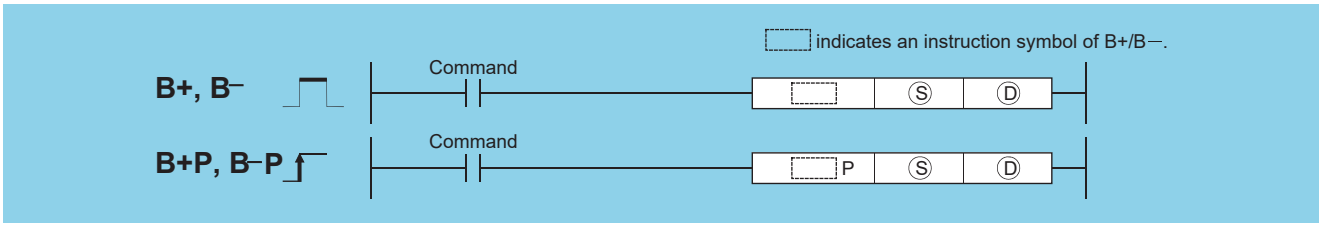
[List Mode]

Step	Instruction	Device
0	LD	X3
1	*P	K2X8 K314 D0
5	D/P	D0 K100 D2
10	MOVP	D2 K4Y30
12	END	

BCD 4-digit addition and subtraction operations

B+(P), B-(P) [When two data are set]

Basic | High performance | Process | Redundant | Universal | LCPU



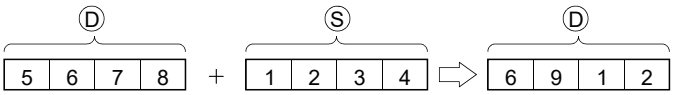
(S): Data for adding/subtracting or head number of the devices where the data for adding/subtracting is stored (BCD 4 digits)
(D): Head number of the devices where the data to be added to/subtracted from is stored (BCD 4 digits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(S)	○							○	—
(D)	○							—	—

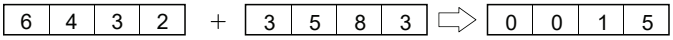
Processing details

■B+

- Adds the BCD 4-digit data designated by (D) and the BCD 4-digit data designated by (S), and stores the result of the addition at the device designated by (D).

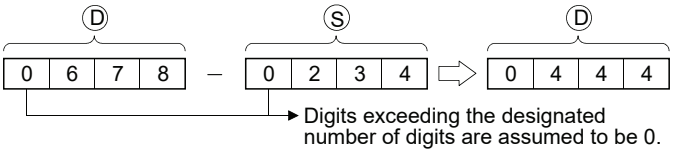


- 0 to 9999 (BCD 4 digits) can be assigned to (S) and (D).
- If the result of the addition operation exceeds 9999, the higher bits are ignored. The carry flag (SM700) in this case does not turn ON.

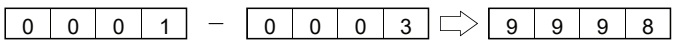


■B-

- Subtracts the BCD 4-digit data designated by (S) and the BCD 4-digit data designated by (D), and stores the result of the subtraction at the device designated by (D).



- 0 to 9999 (BCD 4 digits) can be assigned to (S) and (D).
- The following will result if an underflow is generated by the subtraction operation: The carry flag (SM700) in this case does not turn ON.



Operation error

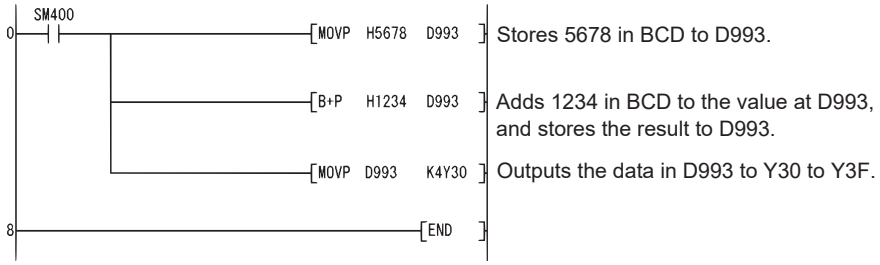
- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4100	The (S) or (D) BCD data is outside the 0 to 9999 range.	○	○	○	○	○	○

Program example

- The following program adds BCD data 5678 and 1234, stores it at D993, and at the same time outputs it to from Y30 to Y3F.

[Ladder Mode]



Stores 5678 in BCD to D993.

Adds 1234 in BCD to the value at D993, and stores the result to D993.

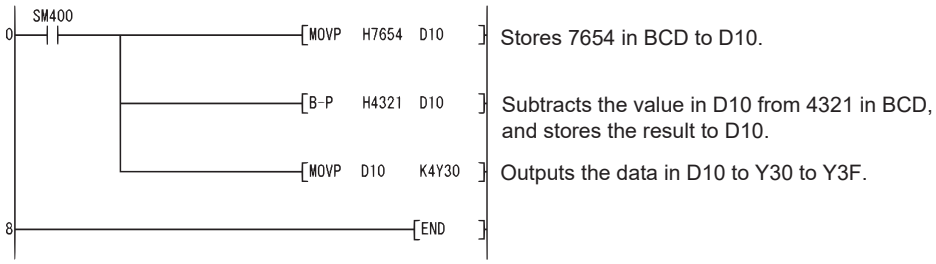
Outputs the data in D993 to Y30 to Y3F.

[List Mode]

Step	Instruction	Device	
0	LD	SM400	
1	MOVP	H5678	D993
3	B+P	H1234	D993
6	MOVP	D993	K4Y30
8	END		

- The following program subtracts the BCD data 4321 from 7654, stores the result at D10, and at the same time outputs it to Y30 to Y3F.

[Ladder Mode]



Stores 7654 in BCD to D10.

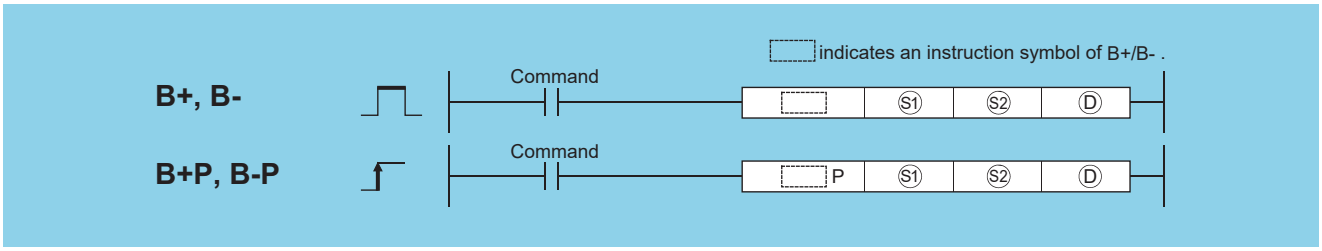
Subtracts the value in D10 from 4321 in BCD, and stores the result to D10.

Outputs the data in D10 to Y30 to Y3F.

[List Mode]

Step	Instruction	Device	
0	LD	SM400	
1	MOVP	H7654	D10
3	B-P	H4321	D10
6	MOVP	D10	K4Y30
8	END		

B+(P), B-(P) [When three data are set]



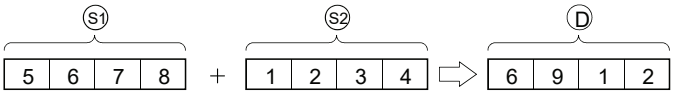
(S1): Data to be added to/subtracted from or head number of the devices where the data to be added to/subtracted from is stored (BCD 4 digits)
(S2): Data for adding/subtracting or head number of the devices where the data for adding/subtracting is stored (BCD 4 digits)
(D): Head number of the devices where the operation result will be stored (BCD 4 digits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(S1)	○							○	—
(S2)	○							○	—
(D)	○							—	—

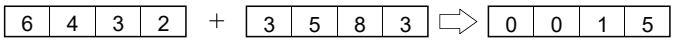
Processing details

■B+

- Adds the BCD 4-digit data designated by (S1) and the BCD 4-digit data designated by (S2), and stores the result of the addition at the device designated by (D).

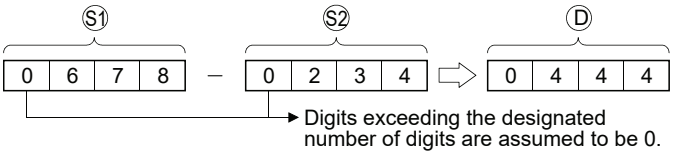


- 0 to 9999 (BCD 4 digits) can be assigned to (S1), (S2) and (D).
- If the result of the addition operation exceeds 9999, the higher bits are ignored. The carry flag (SM700) in this case does not turn ON.

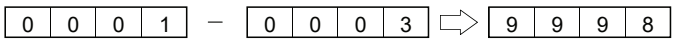


■B-

- Subtracts the BCD 4-digit data designated by (S1) and the BCD 4-digit data designated by (S2), and stores the result of the subtraction at the device designated by (D).



- 0 to 9999 (BCD 4 digits) can be assigned to (S1), (S2) and (D).
- The following will result if an underflow is generated by the subtraction operation: The carry flag (SM700) in this case does not turn ON.



Operation error

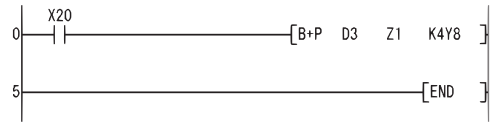
- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4100	The (S1) or (S2) BCD data is outside the 0 to 9999 range.	○	○	○	○	○	○

Program example

- The following program adds the D3 BCD data and the Z1 BCD data when X20 goes ON, and outputs the result to Y8 to Y17.

[Ladder Mode]

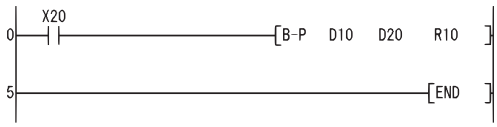


[List Mode]

Step	Instruction	Device		
0	LD	X20		
1	B+P	D3	Z1	K4Y8
5	END			

- The following program subtracts the BCD data at D20 from the BCD data at D10 when X20 goes ON, and stores the result at R10.

[Ladder Mode]



[List Mode]

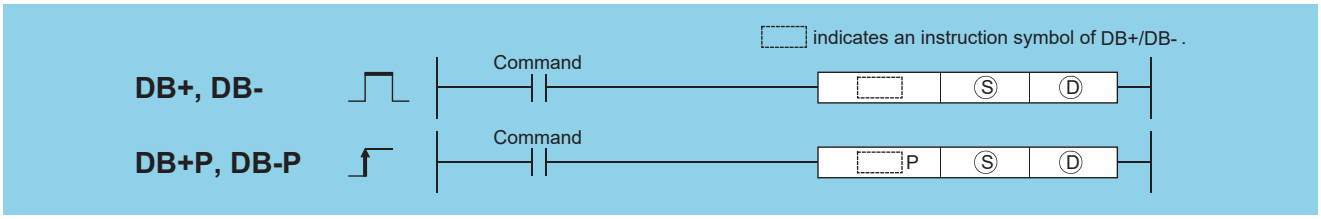
Step	Instruction	Device		
0	LD	X20		
1	B-P	D10	D20	R10
5	END			

6

BCD 8-digit addition and subtraction operations

DB+(P), DB-(P) [When two data are set]

Basic | High performance | Process | Redundant | Universal | LCPU



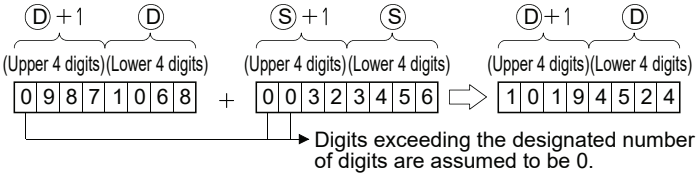
(S): Data for adding/subtracting or head number of the devices where the data for adding/subtracting is stored (BCD 8 digits)
(D): Head number of the devices where the data to be added to/subtracted from is stored (BCD 8 digits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(S)	○							○	—
(D)	○							—	—

Processing details

■DB+

- Adds the BCD 8-digit data designated by (D) and the BCD 8-digit data designated by (S), and stores the result of the addition at the device designated by (D).

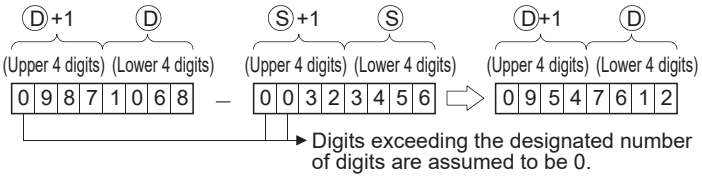


- 0 to 99999999 (BCD 8 digits) can be assigned to (S) and (D).
- If the result of the addition operation exceeds 99999999, the upper bits will be ignored. The carry flag (SM700) in this case does not turn ON.

9 9 0 0 0 0 0 0 + 0 1 6 5 4 3 2 1 ⇨ 0 0 6 5 4 3 2 1

■DB-

- Subtracts the BCD 8-digit data designated by (D) and the BCD 8-digit data designated by (S), and stores the result of the subtraction at the device designated by (D).



- 0 to 99999999 (BCD 8 digits) can be assigned to (S) and (D).
- The following will result if an underflow is generated by the subtraction operation: The carry flag (SM700) in this case does not turn ON.

1 2 3 4 5 6 7 8 – 1 2 3 4 5 6 7 9 ⇨ 9 9 9 9 9 9 9 9

Operation error

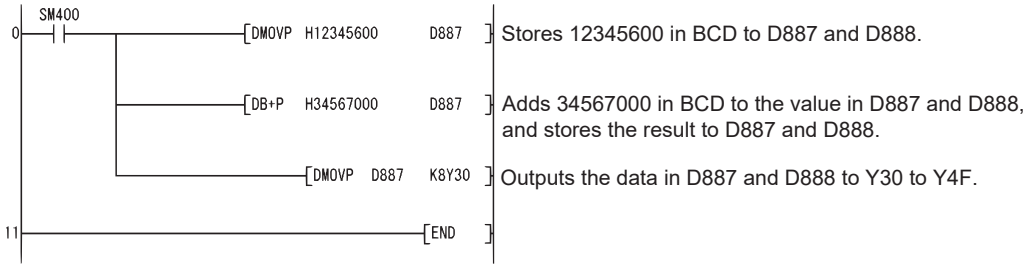
- In the following case, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4100	The (S) or (D) BCD data is outside the 0 to 99999999 range.	○	○	○	○	○	○

Program example

- The following program adds the BCD data 12345600 and 34567000, stores the result at D887 and D888, and at the same time outputs them to from Y30 to Y4F.

[Ladder Mode]

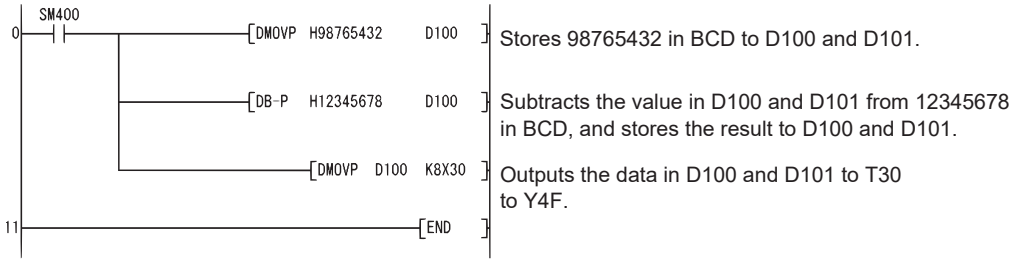


[List Mode]

Step	Instruction	Device	
0	LD	SM400	
1	DMOVP	H12345600	D887
4	DB+P	H34567000	D887
8	DMOVP	D887	K8Y30
11	END		

- The following program subtracts the BCD data 98765432 from 12345678, stores the result at D100 and D101, and at the same time outputs it from Y30 to Y4F.

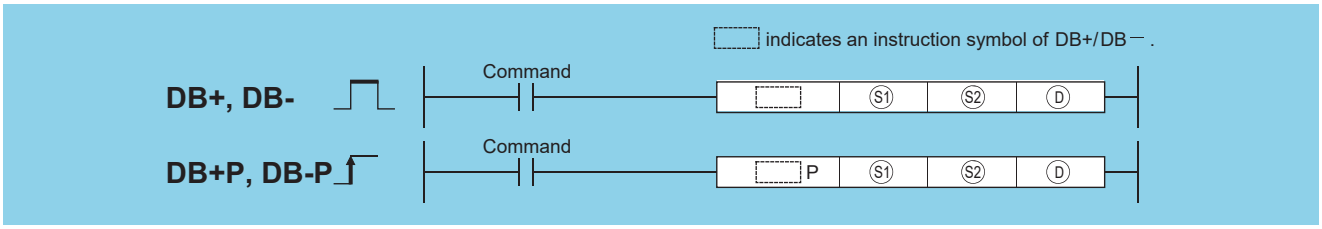
[Ladder Mode]



[List Mode]

Step	Instruction	Device	
0	LD	SM400	
1	DMOVP	H98765432	D100
4	DB-P	H12345678	D100
8	DMOVP	D100	K8X30
11	END		

DB+(P), DB-(P) [When three data are set]



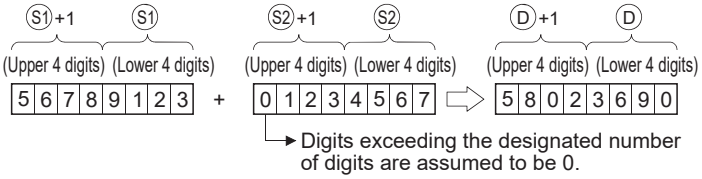
(S1): Data to be added to/subtracted from or head number of the devices where the data to be added to/subtracted from is stored (BCD 8 digits)
(S2): Data for adding/subtracting or head number of the devices where the data for adding/subtracting is stored (BCD 8 digits)
(D): Head number of the devices where the addition/subtraction operation result is stored (BCD 8 digits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(S1)	○							○	—
(S2)	○							○	—
(D)	○							—	—

Processing details

■DB+

- Adds the BCD 8-digit data designated by (S1) and the BCD 8-digit data designated by (S2), and stores the result of the addition at the device designated by (D).

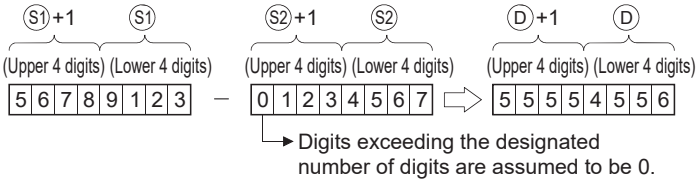


- 0 to 99999999 (BCD 8 digits) can be assigned to (S1), (S2) and (D).
- If the result of the addition operation exceeds 99999999, the upper bits will be ignored. The carry flag (SM700) in this case does not turn ON.

9 9 0 0 0 0 0 0 + 0 1 6 5 4 3 2 1 ⇨ 0 0 6 5 4 3 2 1

■DB-

- Subtracts the BCD 8-digit data designated by (S1) and the BCD 8-digit data designated by (S2), and stores the result of the subtraction at the device designated by (D).



- 0 to 99999999 (BCD 8 digits) can be assigned to (S1), (S2) and (D).
- The following will result if an underflow is generated by the subtraction operation: The carry flag (SM700) in this case does not turn ON.

1 2 3 4 5 6 7 8 − 1 2 3 4 5 6 7 9 ⇨ 9 9 9 9 9 9 9 9

Operation error

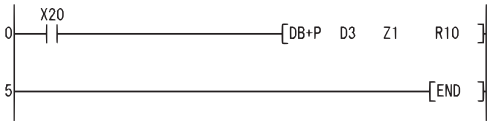
- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4100	The (S1) or (S2) BCD data is outside the 0 to 99999999 range.	○	○	○	○	○	○

Program example

- The following program adds the BCD data at D3 and D4 to the BCD data at Z1 and Z2 when X20 goes ON, and stores the result at R10 and R11.

[Ladder Mode]



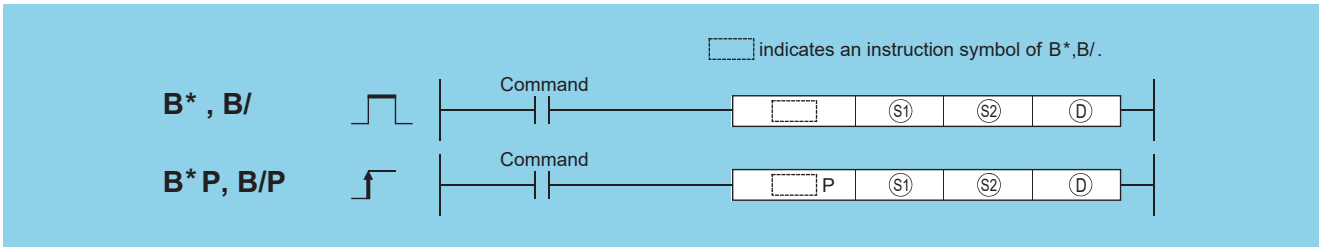
[List Mode]

Step	Instruction	Device
0	LD	X20
1	DB+P	D3 Z1 R10
5	END	

BCD 4-digit multiplication and division operations

B*(P), B/(P)

Basic | High performance | Process | Redundant | Universal | LCPU



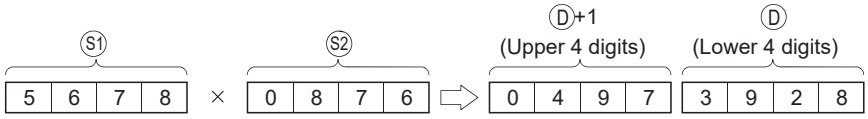
(S1): Data to be multiplied/divided or head number of the devices where the data to be multiplied/divided is stored (BCD 4 digits)
(S2): Data for multiplying/dividing or head number of the devices where the data for multiplying/dividing is stored (BCD 4 digits)
(D): Head number of the devices where the multiplication/division operation result will be stored (BCD 8 digits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(S1)	○							○	—
(S2)	○							○	—
(D)	○							—	—

Processing details

■B*

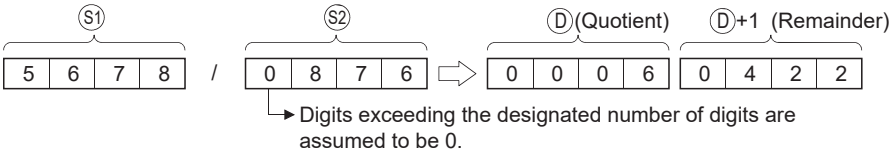
- Multiplies BCD data designated by (S1) and BCD data designated by (S2), and stores the result in the device designated by (D).



- 0 to 9999 (BCD 4 digits) can be assigned to (S1) and (S2).

■B/

- Divides BCD data designated by (S1) and BCD data designated by (S2), and stores the result in the device designated by (D).



- Uses 32 bits to store the result of the division as quotient and remainder
  - Quotient (BCD 4 digits) ··· Stored at the lower 16 bits.
  - Remainder (BCD 4 digits) ··· Stored at the upper 16 bits.
- If (D) has been designated as a bit device, the remainder of the operation will not be stored.

Operation error

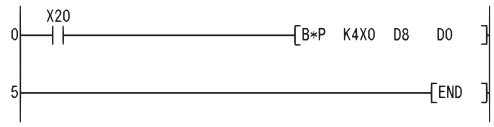
- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4100	The (S1) or (S2) BCD data is outside the 0 to 9999 range. The divisor is 0.	○	○	○	○	○	○

Program example

- The following program multiplies, when X20 is turned ON, the BCD data at X0 to XF by the BCD data at D8 and stores the operation result at D0 to D1.

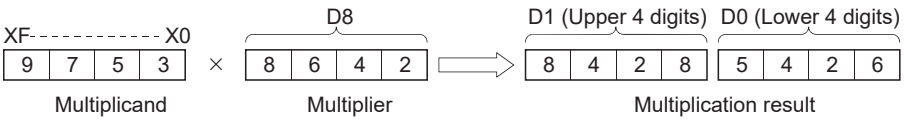
[Ladder Mode]



[List Mode]

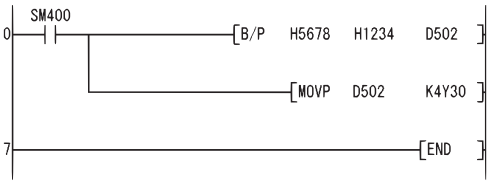
Step	Instruction	Device		
0	LD	X20		
1	B*P	K4X0	D8	D0
5	END			

[Operation]



- The following program divides 5678 by the BCD data 1234, stores the result at D502 and D503, and at the same time outputs the quotient to Y30 to Y3F.

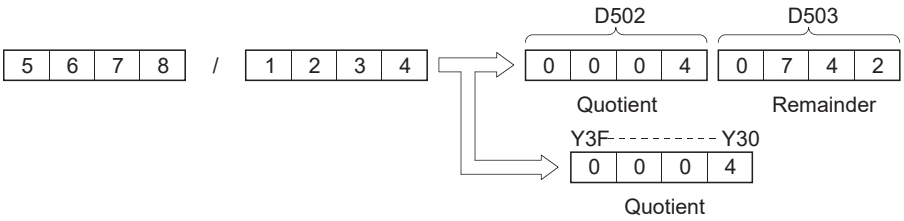
[Ladder Mode]



[List Mode]

Step	Instruction	Device		
0	LD	SM400		
1	B/P	H5678	H1234	D502
5	MOVP	D502	K4Y30	
7	END			

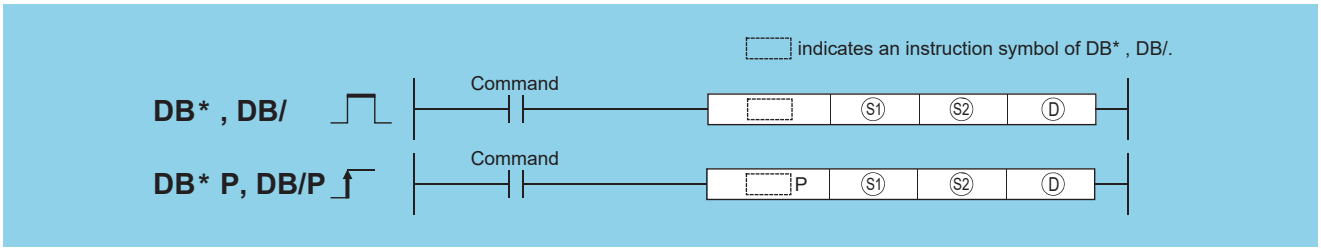
[Operation]



BCD 8-digit multiplication and division operations

DB*(P), DB/(P)

Basic | High performance | Process | Redundant | Universal | LCPU



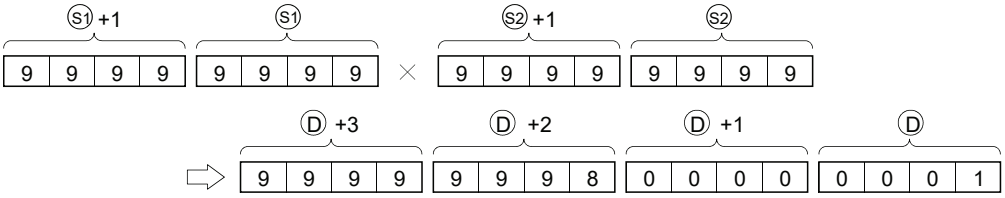
(S1): Data to be multiplied/divided or head number of the devices where the data to be multiplied/divided is stored (BCD 8 digits)
(S2): Data for multiplying/dividing or head number of the devices where the data for multiplying/dividing is stored (BCD 8 digits)
(D): Head number of the devices where the multiplication/division operation result will be stored (BCD 16 digits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(S1)	○			○					—
(S2)	○			○					—
(D)	○			—					—

Processing details

■DB*

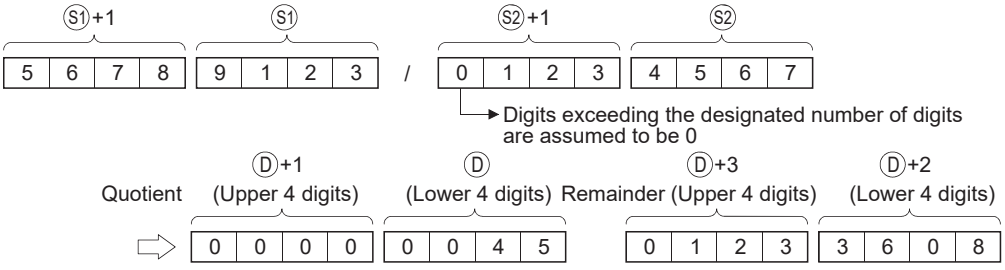
- Multiplies the BCD 8-digit data designated by (S1) and the BCD 8-digit data designated by (S2), and stores the product at the device designated by (D).



- If (D) has designated a bit device, the lower 8 digits (lower 32 bits) will be used for the product, and the higher 8 digits (upper 32 bits) cannot be designated.
  - K1 ··· Lower 1 digit (b0 to 3)
  - K4 ··· Lower 4 digits (b0 to 15)
  - K8 ··· Lower 8 digits (b0 to 31)
- 0 to 99999999 (BCD 8 digits) can be assigned to (S1) and (S2).

■DB/

- Divides 8-digit BCD data designated by (S1) and 8-digit BCD data designated by (S2), and stores the result in the device designated by (D).



- 64 bits are used for the result of the division operation, and stored as quotient and remainder.
  - Quotient (BCD 8 digits) ··· Stored at the lower 32 bits.
  - Remainder (BCD 8 digits) ··· Stored at the upper 32 bits.
- If (D) has been designated as a bit device, the remainder of the operation will not be stored.

Operation error

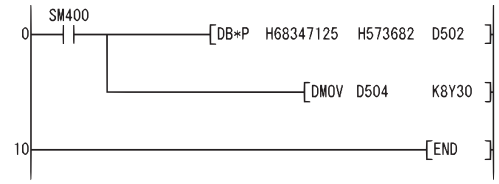
- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4100	The (S1) or (S2) BCD data is outside the 0 to 99999999 range. The divisor is 0.	○	○	○	○	○	○

Program example

- The following program multiplies the BCD data 67347125 and 573682, stores the result from D502 to D505, and at the same time outputs the upper 8 digits to Y30 to Y4F.

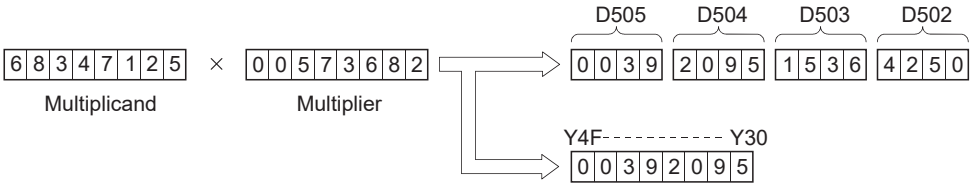
[Ladder Mode]



[List Mode]

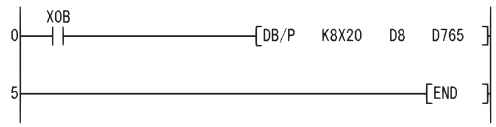
Step	Instruction	Device
0	LD	SM400
1	DB*P	H68347125 H573682 D502
7	DMOV	D504 K8Y30
10	END	

[Operation]



- The following program divides the BCD data from X20 to X3F by the BCD data at D8 and D9 when X0B goes ON, and stores the result from D765 to D768.

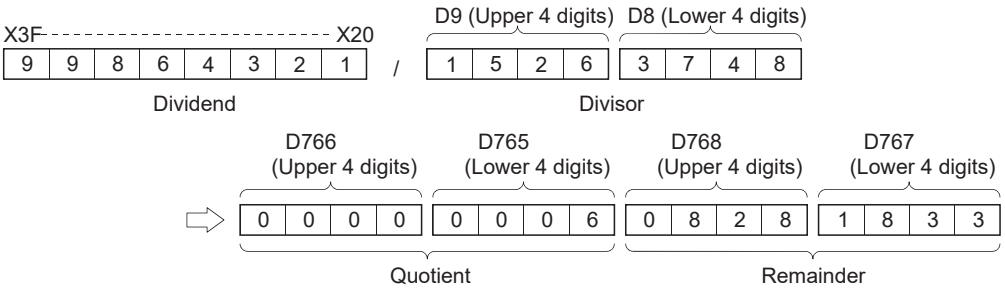
[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	X0B
1	DB/P	K8X20 D8 D765
5	END	

[Operation]

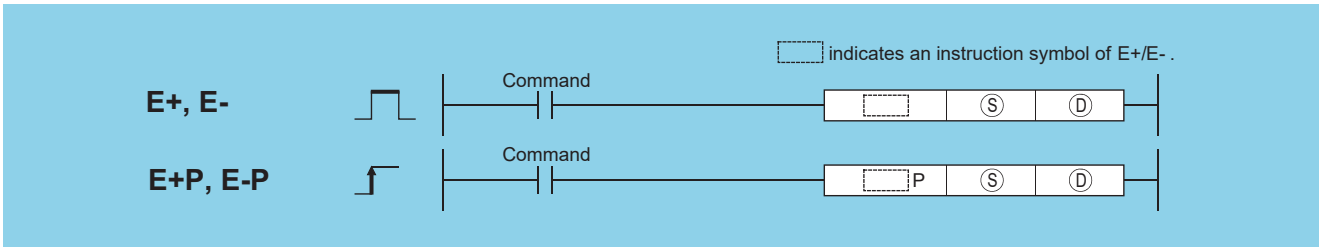


Addition and subtraction of floating-point data (single precision)

E+(P), E-(P) [When two data are set]

Ver. Basic | High performance | Process | Redundant | Universal | LCPU

• Basic model QCPU: The serial number (first five digits) is "04122" or later.



(S): Data for adding/subtracting or head number of the devices where the data for adding/subtracting is stored (real number)
(D): Head number of the devices where the data to be added to/subtracted from is stored (real number)

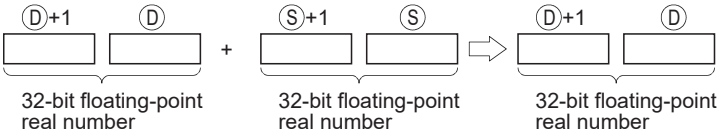
Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant E	Others
	Bit	Word		Bit	Word				
(S)	—	○		—	○		○*1	○	—
(D)	—	○		—	○		○*1	—	—

*1 Applicable for the Universal model QCPU, LCPU.

Processing details

■E+

• Adds the 32-bit floating decimal point type real number designated at (D) and the 32-bit floating decimal point type real number designated at (S), and stores the sum in the device designated at (D).



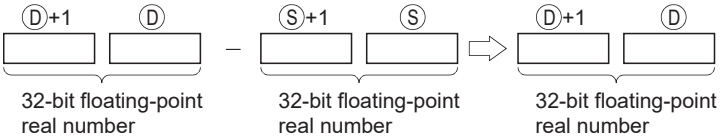
• Values which can be designated at (S) and (D) and which can be stored, are as follows:

$0, 2^{-126} \leq |$ Designated value (stored value) $| < 2^{128}$

• When an input value is set using a programming tool, a rounding error may occur. For precautions, refer to Page 90 Precautions.

■E-

• Subtracts a 32-bit floating decimal point type real number designated by (D) and a 32-bit floating decimal point type real number designated by (S), and stores the result at a device designated by (D).



• Values which can be designated at (S) and (D) and which can be stored, are as follows:

$0, 2^{-126} \leq |$ Designated value (stored value) $| < 2^{128}$

• When an input value is set using a programming tool, a rounding error may occur. For precautions, refer to Page 90 Precautions.

Operation error

- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

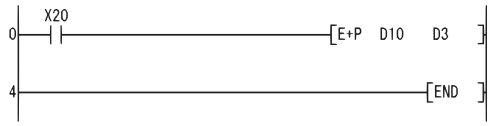
Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4100	The specified device value is not within the following range: 0, $2^{-126} \leq$ | Specified device value | $< 2^{128}$ The specified device value is -0.*1	○	○	○	○	—	—
4141	The operation result exceeds the following range. (when an overflow occurs): |Operation result| $< 2^{128}$	○	○	○	○	○	○
4140	The specified device value is -0, unnormalized number, nonnumeric, and ±∞.	—	—	—	—	○	○

*1 There are CPU modules that will not result in an operation error if -0 is specified. For details, refer to Page 88 Using single/double-precision real number data.

Program example

- The following program adds the 32-bit floating decimal point type real number at D3 and D4 and the 32-bit floating decimal point type real number at D10 and D11 when X20 turns ON, and stores the result at D3 and D4.

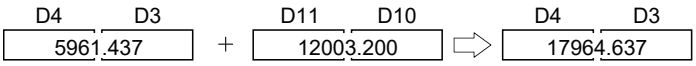
[Ladder Mode]



[List Mode]

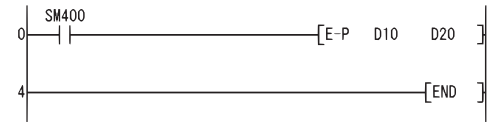
Step	Instruction	Device	
0	LD	X20	
1	E+P	D10	D3
4	END		

[Operation]



- The following program subtracts the 32-bit floating decimal point type real number at D10 and D11 from the 32-bit floating decimal point type real number at D20 and D21, and stores the result of the subtraction at D20 and D21.

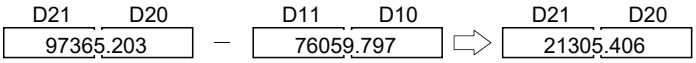
[Ladder Mode]



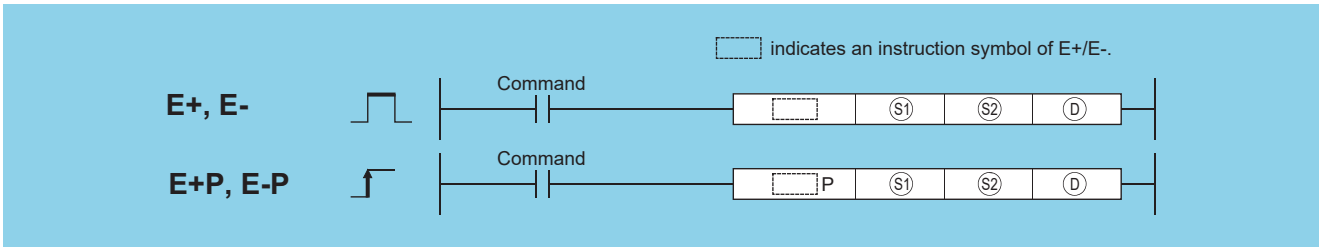
[List Mode]

Step	Instruction	Device	
0	LD	SM400	
1	E-P	D10	D20
4	END		

[Operation]



E+(P), E-(P) [When three data are set]



(S1): Data to be added to/subtracted from or head number of the devices where the data to be added to/subtracted from is stored (real number)
(S2): Data for adding/subtracting or head number of the devices where the data for adding/subtracting is stored (real number)
(D): Head number of the devices where the operation result will be stored (real number)

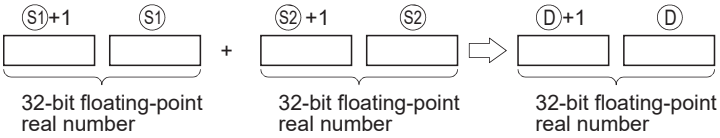
Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant E	Others
	Bit	Word		Bit	Word				
(S1)	—	○		—	○		○*1	○	—
(S2)	—	○		—	○		○*1	○	—
(D)	—	○		—	○		○*1	—	—

*1 Applicable for the Universal model QCPU, LCPU.

Processing details

■E+

- Adds the 32-bit floating decimal point type real number designated at (S1) and the 32-bit floating decimal point type real number designated at (S2), and stores the sum in the device designated at (D).



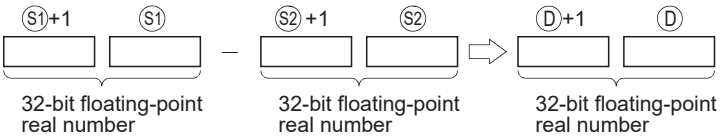
- Values which can be designated at (S1), (S2) and (D) and which can be stored, are as follows:

$0, 2^{-126} \leq |$ Designated value (stored value) $| < 2^{128}$

- When an input value is set using a programming tool, a rounding error may occur. For precautions, refer to Page 90 Precautions.

■E-

- Subtracts a 32-bit floating decimal point type real number designated by (S1) and a 32-bit floating decimal point type real number designated by (S2), and stores the result at a device designated by (D).



- Values which can be designated at (S1), (S2) and (D) and which can be stored, are as follows:

$0, 2^{-126} \leq |$ Designated value (stored value) $| < 2^{128}$

- When an input value is set using a programming tool, a rounding error may occur. For precautions, refer to Page 90 Precautions.

Operation error

- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

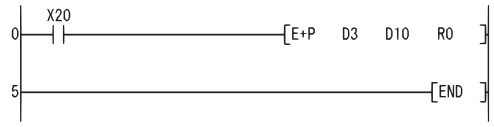
Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4100	The specified device value is not within the following range: 0, $2^{-126} \leq$ | Specified device value | $< 2^{128}$ The specified device value is -0.*1	○	○	○	○	—	—
4141	The operation result exceeds the following range. (when an overflow occurs): |Operation result| $< 2^{128}$	—	—	—	—	○	○
4140	The specified device value is -0, unnormalized number, nonnumeric, and ±∞.	—	—	—	—	○	○

*1 There are CPU modules that will not result in an operation error if -0 is specified. For details, refer to Page 88 Using single/double-precision real number data.

Program example

- The following program adds the 32-bit floating decimal point type real number at D3 and D4 and the 32-bit floating decimal point type real number at D10 and D11 when X20 goes ON, and outputs the result to R0 and R1.

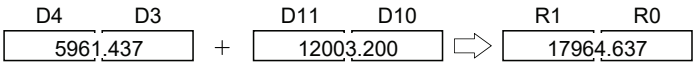
[Ladder Mode]



[List Mode]

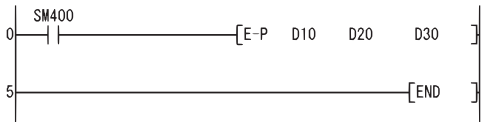
Step	Instruction	Device		
0	LD	X20		
1	E+P	D3	D10	R0
5	END			

[Operation]



- The following programs subtracts the 32-bit floating decimal point type real number at D20 and D21 from the 32-bit floating decimal point type real number at D11 and D10, and stores the result at D30 and D31.

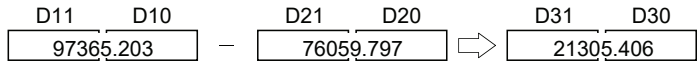
[Ladder Mode]



[List Mode]

Step	Instruction	Device		
0	LD	SM400		
1	E-P	D10	D20	D30
5	END			

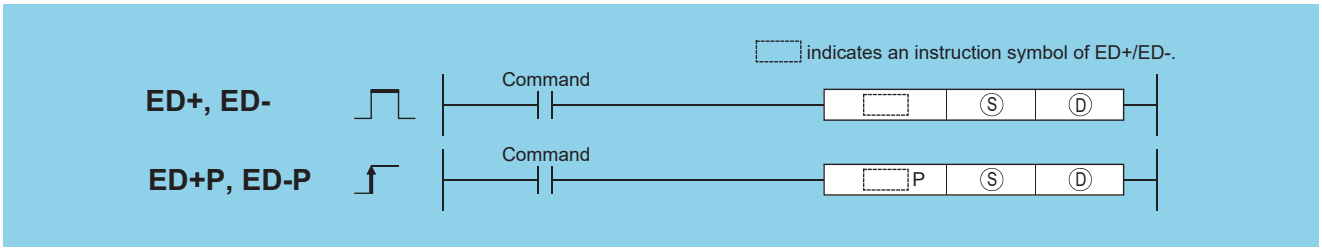
[Operation]



Addition and subtraction of floating-point data (double precision)

ED+(P), ED-(P) [When two data are set]

Basic | High performance | Process | Redundant | Universal | LCPU



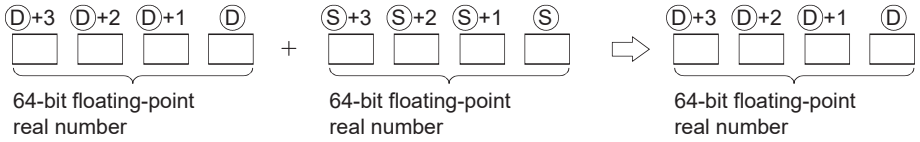
(S): Data for adding/subtracting or head number of the devices where the data for adding/subtracting is stored (real number)
(D): Head number of the devices where the data to be added to/subtracted from is stored (real number)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant E	Others
	Bit	Word		Bit	Word				
(S)	—	○		—				○	—
(D)	—	○		—				—	—

Processing details

■ED+

- Adds the 64-bit floating decimal point type real number designated at (D) and the 64-bit floating decimal point type real number designated at (S), and stores the sum in the device designated at (D).



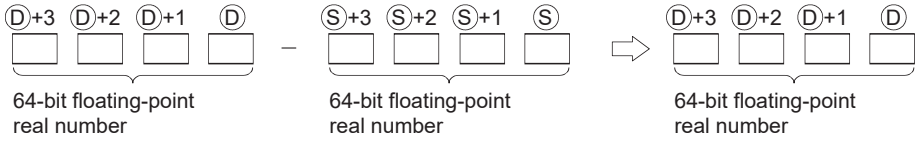
- Values which can be designated at (S) and (D) and which can be stored, are as follows:

$0, 2^{-1022} \leq |$ Designated value (stored value) $| < 2^{1024}$

- When an input value is set using a programming tool, a rounding error may occur. For precautions, refer to Page 90 Precautions.

■ED-

- Subtracts a 64-bit floating decimal point type real number designated by (D) and a 64-bit floating decimal point type real number designated by (S), and stores the result at a device designated by (D).



- Values which can be designated at (S) and (D) and which can be stored, are as follows:

$0, 2^{-1022} \leq |$ Designated value (stored value) $| < 2^{1024}$

- When an input value is set using a programming tool, a rounding error may occur. For precautions, refer to Page 90 Precautions.

Operation error

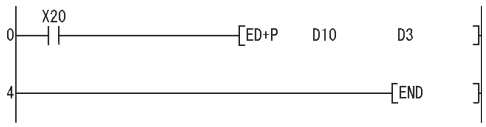
- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4140	The specified device value is not within the following range: 0, $2^{-1022} \leq$ | Specified device value | $< 2^{1024}$ The specified device value is -0, unnormalized number, nonnumeric, and ±∞.	—	—	—	—	○	○
4141	The operation result exceeds the following range. (when an overflow occurs): |Operation result| $< 2^{1024}$	—	—	—	—	○	○

Program example

- The following program adds the 64-bit floating decimal point type real number at D3 to D6 and the 64-bit floating decimal point type real number at D10 to D13 when X20 goes ON, and stores the result at D3 to D6.

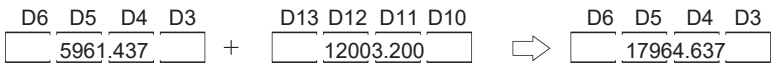
[Ladder Mode]



[List Mode]

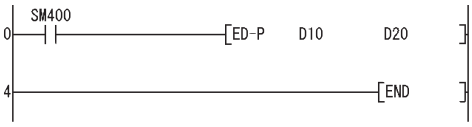
Step	Instruction	Device	
0	LD	X20	
1	ED+P	D10	D3
4	END		

[Operation]



- The following program subtracts the 64-bit floating decimal point type real number at D10 to D13 from the 64-bit floating decimal point type real number at D20 to D23, and stores the result of the subtraction at D20 to D23.

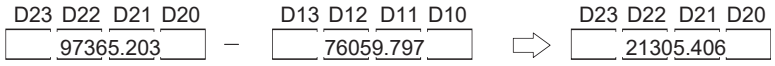
[Ladder Mode]



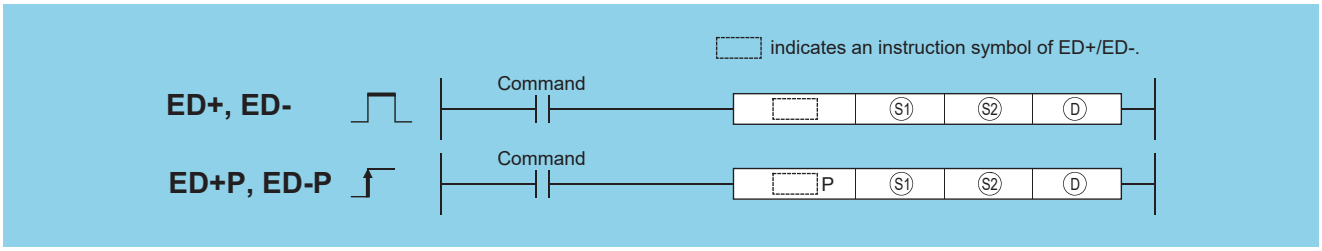
[List Mode]

Step	Instruction	Device	
0	LD	SM400	
1	ED-P	D10	D20
4	END		

[Operation]



ED+(P), ED-(P) [When three data are set]



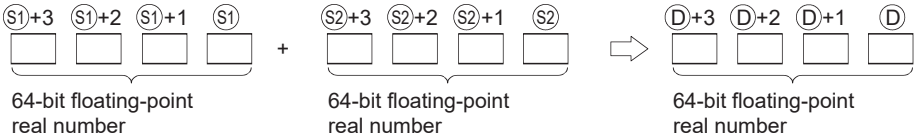
(S1): Data to be added to/subtracted from or head number of the devices where the data to be added to/subtracted from is stored (real number)
(S2): Data for adding/subtracting or head number of the devices where the data for adding/subtracting is stored (real number)
(D): Head number of the devices where the operation result will be stored (real number)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant E	Others
	Bit	Word		Bit	Word				
(S1)	—	○		—				○	—
(S2)	—	○		—				○	—
(D)	—	○		—				—	—

Processing details

■ED+

- Adds the 64-bit floating decimal point type real number designated at (S1) and the 64-bit floating decimal point type real number designated at (S2), and stores the sum in the device designated at (D).



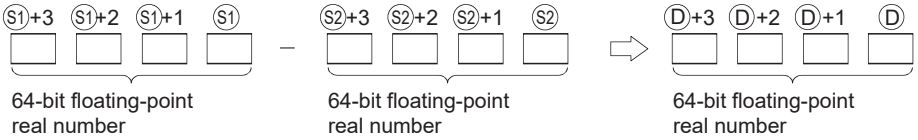
- Values which can be designated at (S1), (S2) and (D) and which can be stored, are as follows:

$0, 2^{-1022} \leq |$ Designated value (stored value) $| < 2^{1024}$

- When an input value is set using a programming tool, a rounding error may occur. For precautions, refer to Page 90 Precautions.

■ED-

- Subtracts a 64-bit floating decimal point type real number designated by (S1) and a 64-bit floating decimal point type real number designated by (S2), and stores the result at a device designated by (D).



- Values which can be designated at (S1), (S2) and (D) and which can be stored, are as follows:

$0, 2^{-1022} \leq |$ Designated value (stored value) $| < 2^{1024}$

- When an input value is set using a programming tool, a rounding error may occur. For precautions, refer to Page 90 Precautions.

Operation error

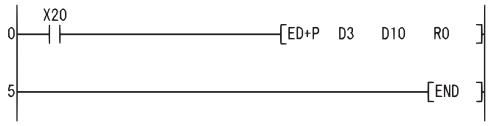
- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4140	The specified device value is not within the following range: 0, $2^{-1022} \leq$ | Specified device value | $< 2^{1024}$ The specified device value is -0, unnormalized number, nonnumeric, and $\pm\infty$.	—	—	—	—	○	○
4141	The operation result exceeds the following range. (when an overflow occurs): |Operation result| $< 2^{1024}$	—	—	—	—	○	○

Program example

- The following program adds the 64-bit floating decimal point type real number at D3 to D6 and the 64-bit floating decimal point type real number at D10 to D13 when X20 goes ON, and outputs the result at R0 to R3.

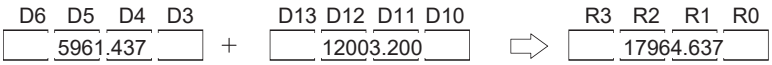
[Ladder Mode]



[List Mode]

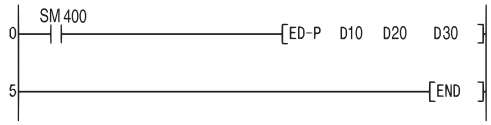
Step	Instruction	Device		
0	LD	X20		
1	ED+P	D3	D10	R0
5	END			

[Operation]



- The following programs subtracts the 64-bit floating decimal point type real number at D20 to D23 from the 64-bit floating decimal point type real number at D10 to D13, and stores the result at D30 to D33.

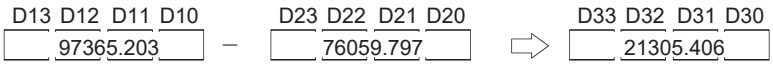
[Ladder Mode]



[List Mode]

Step	Instruction	Device		
0	LD	SM400		
1	ED-P	D10	D20	D30
5	END			

[Operation]

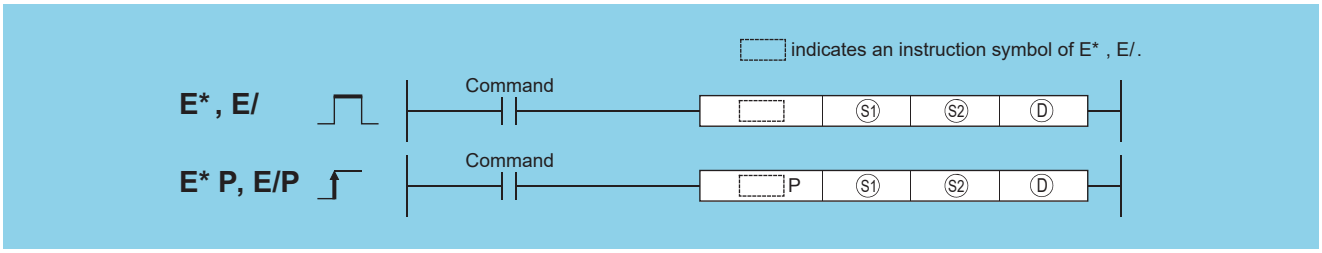


Multiplication and division of floating-point data (single precision)

E*(P), E/(P)

Ver. Basic | High performance | Process | Redundant | Universal | LCPU

• Basic model QCPU: The serial number (first five digits) is "04122" or later.



(S1): Data to be multiplied/divided or head number of the devices where the data to be multiplied/divided is stored (real number)
(S2): Data for multiplying/dividing or head number of the devices where the data for multiplying/dividing is stored (real number)
(D): Head number of the devices where the operation result will be stored (real number)

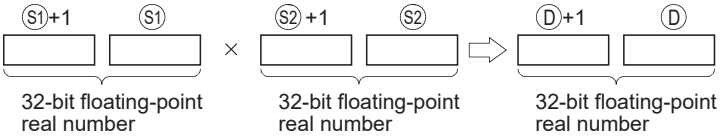
Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant E	Others
	Bit	Word		Bit	Word				
(S1)	—	○		—	○		○*1	○	—
(S2)	—	○		—	○		○*1	○	—
(D)	—	○		—	○		○*1	—	—

*1 Applicable for the Universal model QCPU, LCPU.

Processing details

■E*

• Multiplies the 32-bit floating decimal point real number designated by (S1) by the 32-bit floating decimal point real number designated by (S2) and stores the operation result at the device designated by (D).



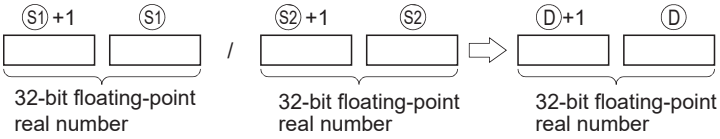
• Values which can be designated at (S1), (S2) and (D) and which can be stored, are as follows:

$0, 2^{-126} \leq |$ Designated value (stored value) $| < 2^{128}$

• When an input value is set using a programming tool, a rounding error may occur. For precautions, refer to Page 90 Precautions.

■E/

• Divides the 32-bit floating decimal point real number designated by (S1) by the 32-bit floating decimal point real number designated by (S2) and stores the operation result at the device designated by (D).



• Values which can be designated at (S1), (S2) and (D) and which can be stored, are as follows:

$0, 2^{-126} \leq |$ Designated value (stored value) $| < 2^{128}$

• When an input value is set using a programming tool, a rounding error may occur. For precautions, refer to Page 90 Precautions.

Operation error

- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

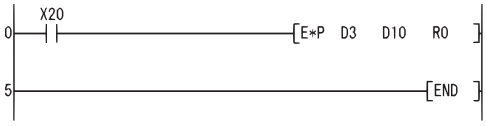
Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4100	The specified device value is not within the following range: 0, $2^{-126} \leq$ | Specified device value | $< 2^{128}$ The specified device value is -0.*1	○	○	○	○	—	—
	The divisor is 0.	○	○	○	○	○	○
4141	The operation result exceeds the following range. (when an overflow occurs): $2^{128} \leq$ | Operation result | |Operation result| $< 2^{128}$	—	—	—	—	○	○
4140	The specified device value is -0, unnormalized number, nonnumeric, and ±∞.	—	—	—	—	○	○

*1 There are CPU modules that will not result in an operation error if -0 is specified. For details, refer to Page 88 Using single/double-precision real number data.

Program example

- The following program multiplies the 32-bit floating decimal point real number at D3 and D4 and the 32-bit floating decimal point real number at D10 and D11, and stores the result at R0 and R1.

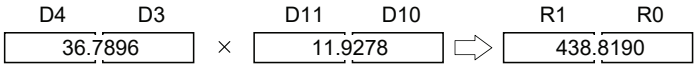
[Ladder Mode]



[List Mode]

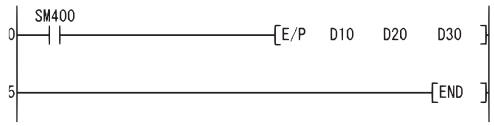
Step	Instruction	Device		
0	LD	X20		
1	E*P	D3	D10	R0
5	END			

[Operation]



- The following program divides the 32-bit floating decimal point real number at D10 and D11 by the 32-bit floating decimal point real number at D20 and D21, and stores the result at D30 and D31.

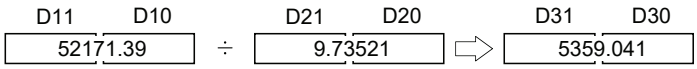
[Ladder Mode]



[List Mode]

Step	Instruction	Device		
0	LD	SM400		
1	E/P	D10	D20	D30
5	END			

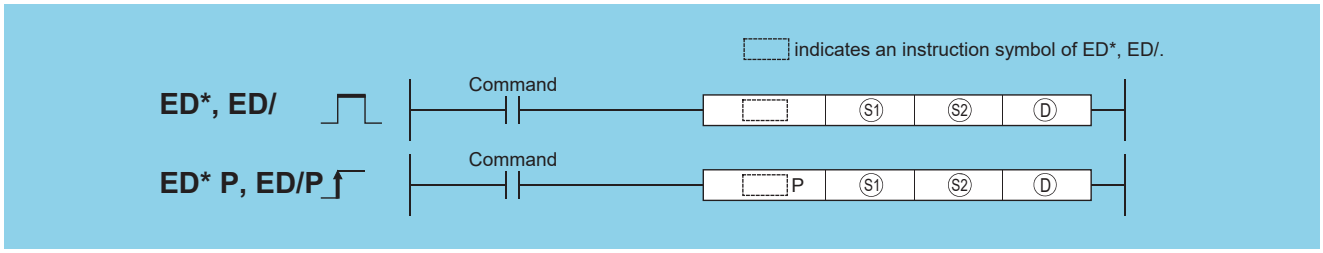
[Operation]



Multiplication and division of floating-point data (double precision)

ED*(P), ED/(P)

Basic | High performance | Process | Redundant | Universal | LCPU



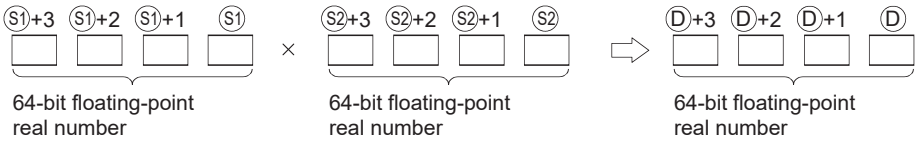
(S1): Data to be multiplied/divided or head number of the devices where the data to be multiplied/divided is stored (real number)
(S2): Data for multiplying/dividing or head number of the devices where the data for multiplying/dividing is stored (real number)
(D): Head number of the devices where the operation result will be stored (real number)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant E	Others
	Bit	Word		Bit	Word				
(S1)	—	○		—				○	—
(S2)	—	○		—				○	—
(D)	—	○		—				—	—

Processing details

■ED*

- Multiplies the 64-bit floating decimal point real number designated by (S1) by the 64-bit floating decimal point real number designated by (S2) and stores the operation result at the device designated by (D).



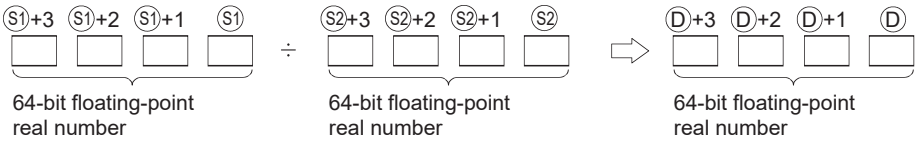
- Values which can be designated at (S1), (S2) and (D) and which can be stored, are as follows:

$0, 2^{-1022} \leq |$ Designated value (stored value) $| < 2^{1024}$

- When the operation results in -0 or an underflow, the result is processed as 0.
- When an input value is set using a programming tool, a rounding error may occur. For precautions, refer to Page 90 Precautions.

■ED/

- Divides the 64-bit floating decimal point real number designated by (S1) by the 64-bit floating decimal point real number designated by (S2) and stores the operation result at the device designated by (D).



- Values which can be designated at (S1), (S2) and (D) and which can be stored, are as follows:

$0, 2^{-1022} \leq |$ Designated value (stored value) $| < 2^{1024}$

- When the operation results in -0 or an underflow, the result is processed as 0.
- When an input value is set using a programming tool, a rounding error may occur. For precautions, refer to Page 90 Precautions.

Operation error

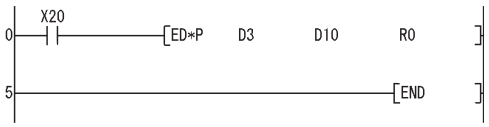
- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4140	The specified device value is not within the following range: 0, $2^{-1022} \leq$ | Specified device value | $< 2^{1024}$ The specified device value is -0, unnormalized number, nonnumeric, and $\pm\infty$.	—	—	—	—	○	○
4100	The divisor is 0.	—	—	—	—	○	○
4141	The operation result exceeds the following range. (when an overflow occurs): |Operation result| $< 2^{1024}$	—	—	—	—	○	○

Program example

- The following program multiplies the 64-bit floating decimal point real number at D3 to D6 and the 64-bit floating decimal point real number at D10 to D13, and stores the result at R0 to R3.

[Ladder Mode]



[List Mode]

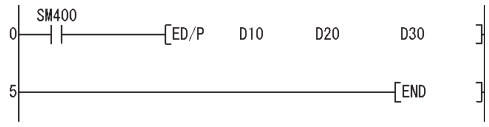
Step	Instruction	Device		
0	LD	X20		
1	ED*P	D3	D10	R0
5	END			

[Operation]

D6	D5	D4	D3		D13	D12	D11	D10		R3	R2	R1	R0
	36.	7896		×		11.	9278		⇨		438.	8190	

- The following program divides the 64-bit floating decimal point real numbers at D10 to D13 by the 64-bit floating decimal point real numbers at D20 to D23, and stores the result at D30 to D33.

[Ladder Mode]



[List Mode]

Step	Instruction	Device		
0	LD	SM400		
1	ED/P	D10	D20	D30
5	END			

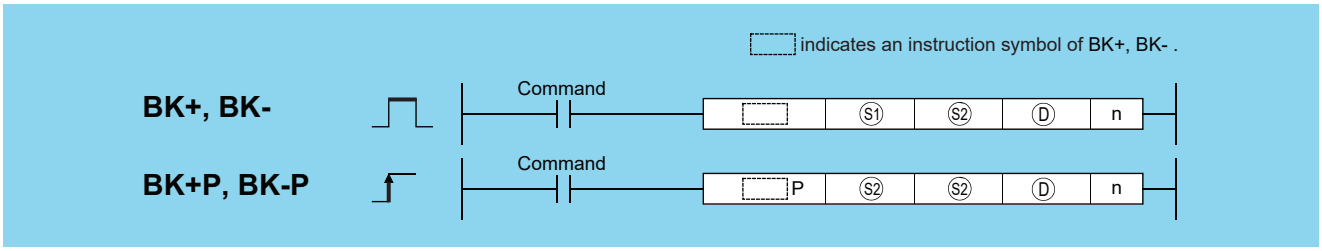
[Operation]

D13	D12	D11	D10		D23	D22	D21	D20		D33	D32	D31	D30
	5217	1.39		÷		9.73	521		⇨		5359	.041	

BIN 16-bit data block addition and subtraction operations

BK+(P), BK-(P)

Basic | High performance | Process | Redundant | Universal | LCPU



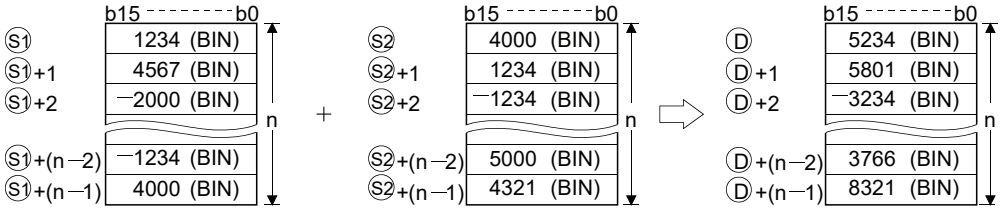
(S1): Head number of the devices where the data to be added to/subtracted from is stored (BIN 16 bits)
(S2): Data for adding/subtracting or head number of the devices where the data for adding/subtracting is stored (BIN 16 bits)
(D): Head number of the devices where the operation result will be stored (BIN 16 bits)
n: Number of addition/subtraction data blocks (BIN 16 bits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(S1)	—	○		—				—	—
(S2)	—	○		—				○	—
(D)	—	○		—				—	—
n	○	○		○				○	—

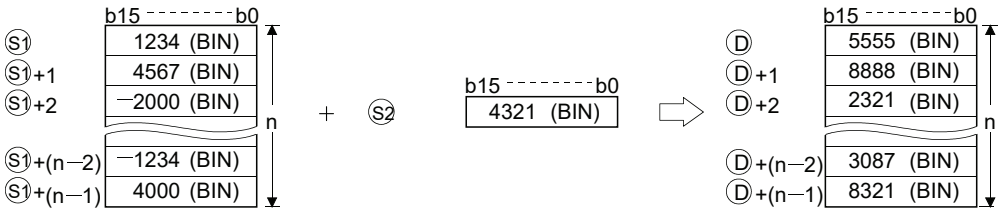
Processing details

■BK+

- Adds n points of BIN data from the device designated by (S1) and n-points of BIN data from the device designated by (S2) and stores the result to the area starting from the device designated by (D).



- Block addition is performed in 16-bit units.
- The constant designated by (S2) can be between -32768 and 32767 (BIN 16-bit data).



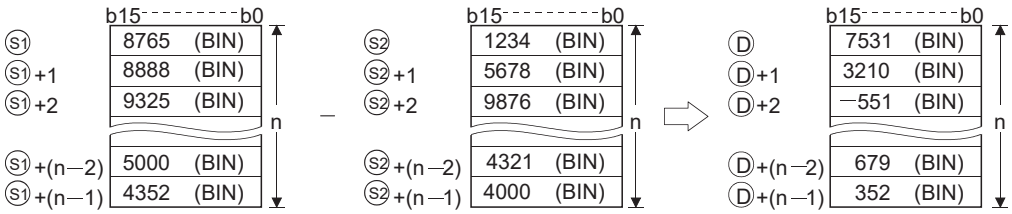
- The following will happen when an underflow or overflow is generated in an operation result: The carry flag (SM700) in this case does not turn ON.

· K32767 +K2 → K−32767
(7FFFH) (0002H) (8001H)

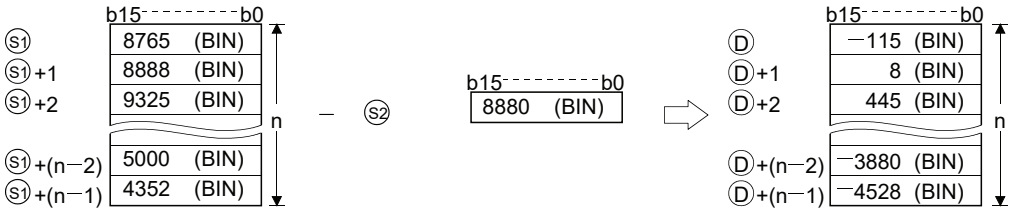
· K−32767 +K−2 → K32767
(8001H) (FFFEH) (7FFFH)

■BK-

- Subtracts n points of BIN data from the device designated by (S1) and n-points of BIN data from the device designated by (S2) and stores the result to the area starting from the device designated by (D).



- Block subtraction is performed in 16-bit units.
- The constant designated by (S2) can be between -32768 and 32767 (BIN 16-bit data).



- The following will happen when an underflow or overflow is generated in an operation result: The carry flag (SM700) in this case does not turn ON.

· K−32768 − K2 ⟶ K32766
(8000H) (0002H) (7FFEH)

· K32767 − K−2 ⟶ −32767
(7FFFH) (FFFEH) (8001H)

Operation error

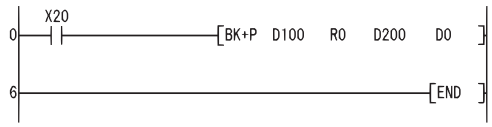
- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4101	The points specified in n exceed those of the corresponding device specified in (S1), (S2), or (D). The ranges of devices starting from the one specified in (S1) and (D) overlap by n points. (except when the same device is specified in (S1) and (D)). The ranges of devices starting from the one specified in (S2) and (D) overlap by n points. (except when the same device is specified in (S2) and (D)).	○	○	○	○	○	○

Program example

- The following program adds, when X20 is turned ON, the data stored at D100 to D103 to the data stored at R0 to R3 and stores the operation result into the area starting from D200.

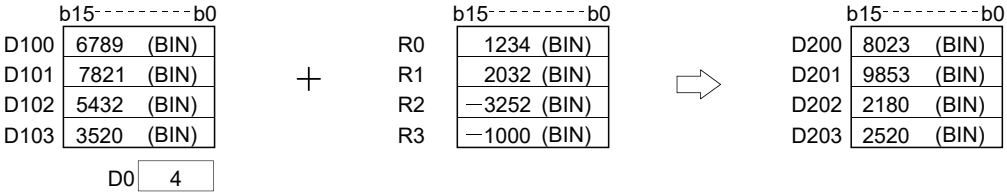
[Ladder Mode]



[List Mode]

Step	Instruction	Device			
0	LD	X20			
1	BK+P	D100	R0	D200	D0
6	END				

[Operation]



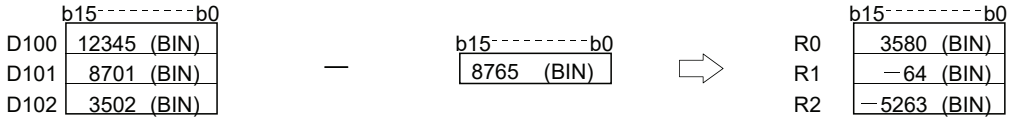
- The following program subtracts, when X1C is turned ON, the constant 8765 from the data at D100 to D102 and stores the operation result into the area starting from R0.

[Ladder Mode]

[List Mode]

Step	Instruction	Device			
0	LD	X1C			
1	BK-P	D100	K8765	R0	K3
6	END				

[Operation]

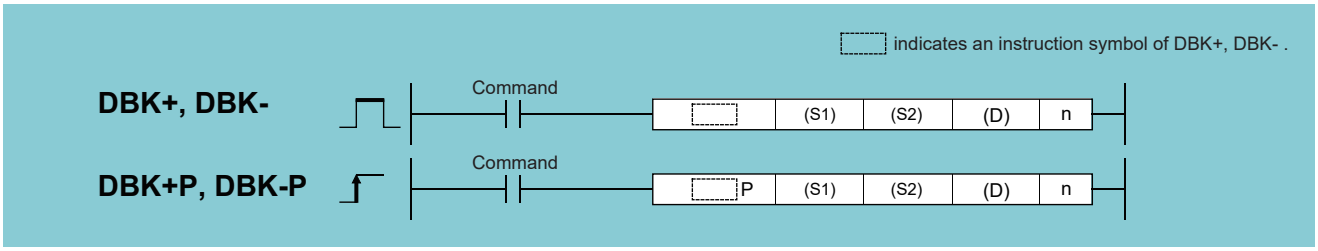


BIN 32-bit data block addition and subtraction operations

DBK+(P), DBK-(P)

Basic | High performance | Process | Redundant | Ver. Universal | LCPU

- QnU(D)(H)CPU, QnUDE(H)CPU: the serial number (first five digits) is "10102" or later
- Q00UJCPU, Q00UCPU, Q01UCPU, QnUDVCPU, QnUDPVCPU: Supported



(S1): Head number of the devices where the data to be added and subtracted are stored (BIN 32 bits)
(S2): Data for adding/subtracting or head number of the devices where the data for adding/subtracting is stored (BIN 32 bits)
(D): Head number of the devices where the addition and subtraction operation result will be stored (BIN 32 bits)
n: Number of addition/subtraction data blocks (BIN 16 bits)

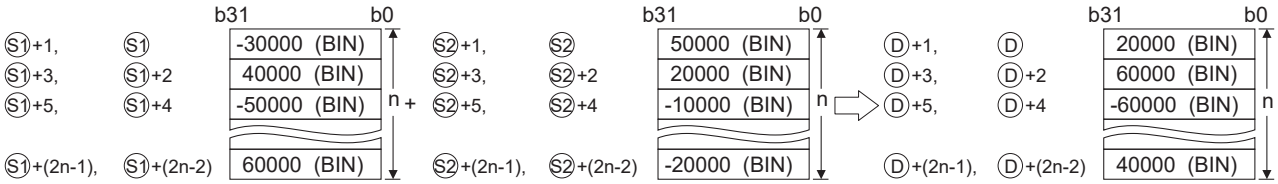
Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(S1)	—	○		—				—	—
(S2)	—	○		—				○	—
(D)	—	○		—				—	—
n	○	○		○				○	—

Processing details

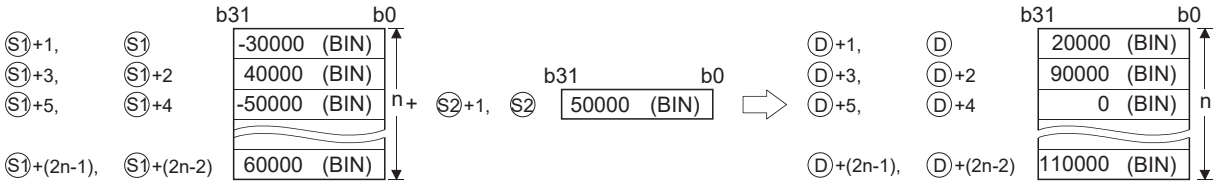
■DBK+

- This instruction adds BIN 32-bit data stored in n-point devices starting from the device specified by (S1) to BIN 32-bit data stored in n-point devices starting from the device specified by (S2) or a constant and then stores the operation result into the nth device specified by (D) and up.

When a device is specified for (S2)



When a constant is specified for (S2)



- Block addition is performed in 32-bit units.
- The constant in the device specified by (S2) can be between -2147483648 to 2147483647 (BIN 32-bit data).
- If the value specified by n is 0, the instruction will be not processed.

• The following will happen if an overflow occurs in an operation result: The carry flag (SM700) in this case does not go ON.

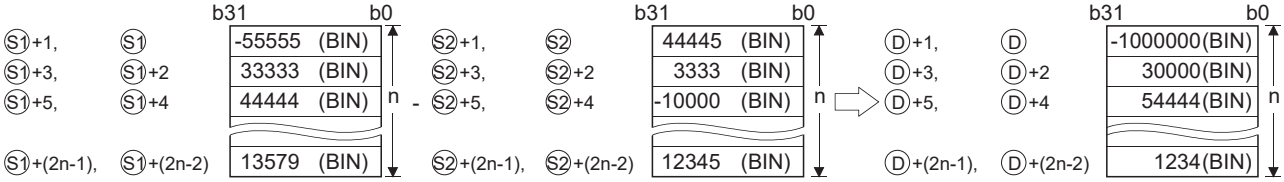
· K2147483647+K2 ⟶ K−2147483647
(7FFFFFFF$_H$) (00000002$_H$) (80000001$_H$)

· K−2147483647+K −2 ⟶ K2147483647
(80000001$_H$) (FFFFFFFE$_H$) (7FFFFFFF$_H$)

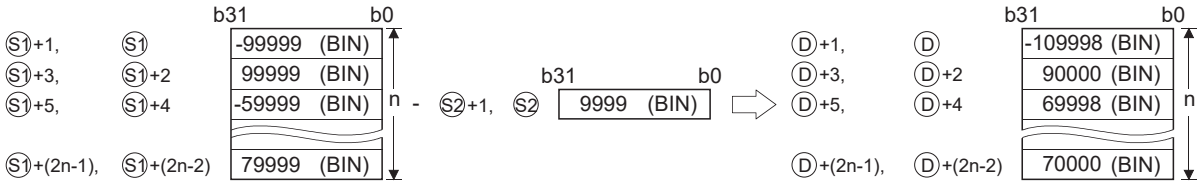
■DBK-

• This instruction subtracts BIN 32-bit data stored in the n-point devices starting from the device specified by (S2) or a constant from BIN 32-bit data stored in n-point devices starting from the device specified by (S1), and then stores the operation result into the nth device specified by (D) and up.

When a device is specified for (S2)



When a constant is specified for (S2)



• Block subtraction is performed in 32-bit units.

• The constant in the device specified by (S2) can be between -2147483648 to 2147483647 (BIN 32-bit data).

• If the value specified by n is 0, the instruction will be not processed.

• Specify (D) out of the range of n-point devices starting from the device specified by (S1) and (S2). However, (S1) and (S2) can specify the same device.

• The following will happen if an overflow occurs in an operation result: The carry flag (SM700) in this case does not go ON.

· K2147483647 −K−2 ⟶ K−2147483647
(7FFFFFFF$_H$) (00000002$_H$) (80000001$_H$)

· K−2147483647 −K2 ⟶ K2147483647
(80000001$_H$) (FFFFFFFE$_H$) (7FFFFFFF$_H$)

Operation error

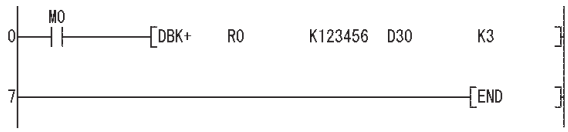
• In any of the following cases, an operation error occurs, the error flag (SM0) turns on, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4100	A negative value is specified for n.	—	—	—	—	○	○
4101	The points specified in n exceed those of the corresponding device specified in (S1), (S2), or (D). The ranges of devices starting from the one specified in (S1) and (D) overlap by n points. (except when the same device is specified in (S1) and (D)). The ranges of devices starting from the one specified in (S2) and (D) overlap by n points. (except when the same device is specified in (S2) and (D)).	—	—	—	—	○	○

Program example

- The following program adds the value data stored at R0 to R5 to the constant, and then stores the operation result into D30 to D35, when M0 is turned on.

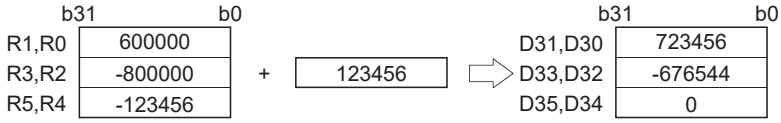
[Ladder Mode]



[List Mode]

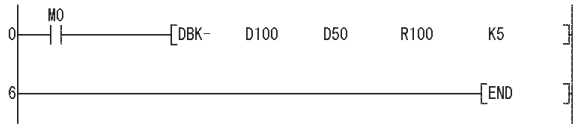
Step	Instruction	Device
0	LD	M0
1	DBK+	R0 K123456 D30 K3
7	END	

[Operation]



- The following program subtracts the value data stored at D50 to D59 from the value data stored at D100 to D109, and then stores the operation result into R100 to R109, when M0 is turned on.

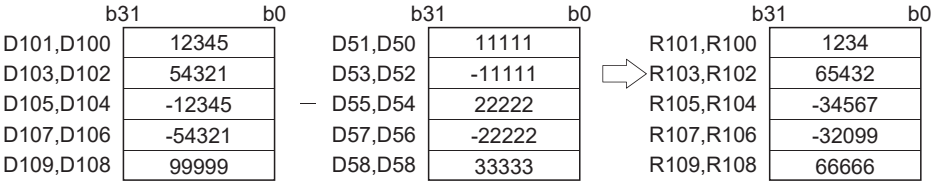
[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	M0
1	DBK-	D100 D50 R100 K5
6	END	

[Operation]



Linking character strings

$+(P) [When two data are set]

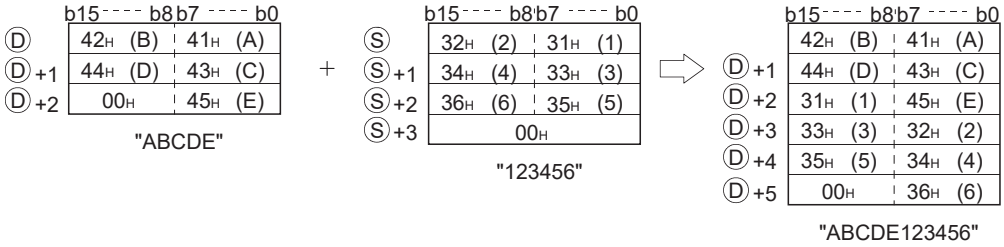
Basic | High performance | Process | Redundant | Universal | LCPU

(S): Data for linking or head number of the devices where the data for linking is stored (character string)
(D): Head number of the devices where the data to be linked is stored (character string)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant $	Others
	Bit	Word		Bit	Word				
(S)	—	○		—				○	—
(D)	—	○		—				—	—

Processing details

- Links the character string data designated by (S) after the character string data designated by (D) and stores the result into the area starting with the device number designated by (D).
- The object of character string data is that character string data stored from device numbers designated at (D) and (S) to that stored at "00H".



- When character strings are linked, the "00H", which indicates the end of character string data designated at (D), is ignored, and the character string designated at (S) is appended to the last character of the (D) string.

Operation error

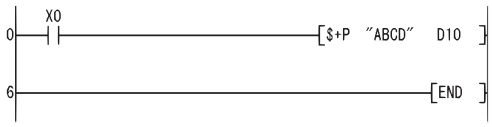
- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4101	The number of device points starting from the device specified in (D) is insufficient to store all character strings. The storage device numbers for the character strings specified by (S) and (D) overlap. The number of characters of (S) and (D) exceeds 16383.	—	○	○	○	○	○

Program example

• The following program links the character string stored from D10 to D12 to the character string "ABCD" when X0 is ON.

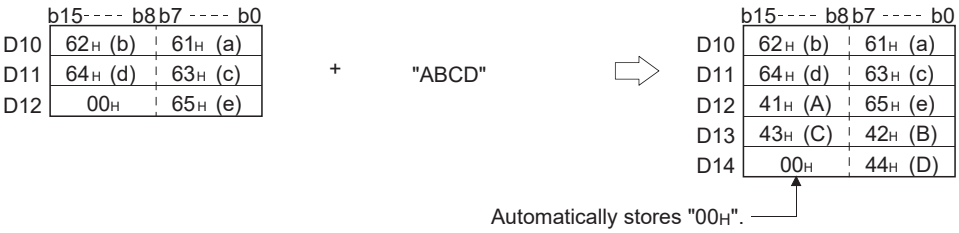
[Ladder Mode]



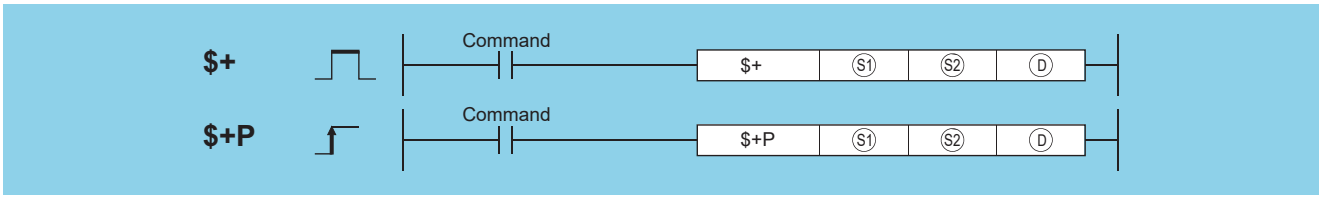
[List Mode]

Step	Instruction	Device
0	LD	X0
1	$+P	"ABCD" D10
6	END	

[Operation]



$+(P) [When three data are set]

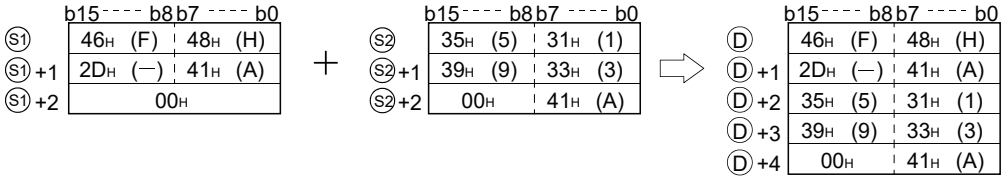


(S1): Data for linking or head number of the devices where the data for linking is stored (character string)
(S2): Data to be linked or head number of the devices where the data to be linked is stored (character string)
(D): Head number of the devices where the linking result will be stored (character string)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant $	Others
	Bit	Word		Bit	Word				
(S1)	—	○		—				○	—
(S2)	—	○		—				○	—
(D)	—	○		—				—	—

Processing details

- Links the character string data designated by (S2) after the character string data designated by (S1) and stores the result into the area starting with the device number designated by (D).



- When character strings are linked, the "00H" which indicates the end of character string data indicated by (S1), is ignored, and the character string indicated by (S2) is appended to the last character of the (S1) string.

Operation error

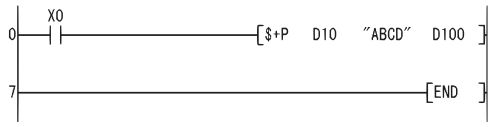
- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4101	The number of device points starting from the device specified in (D) is insufficient to store all character strings. The storage device numbers for the character strings specified by (S2) and (D) overlap. The number of characters of (S1), (S2) and (D) exceeds 16383.	—	○	○	○	○	○

Program example

- The following program links the character string stored from D10 to D12 with the character string "ABCD" when X0 is ON, and stores them in the area starting from D100.

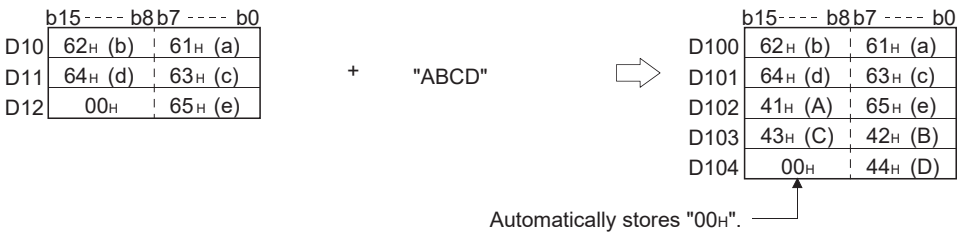
[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	X0
1	$+P	D10 "ABCD" D100
7	END	

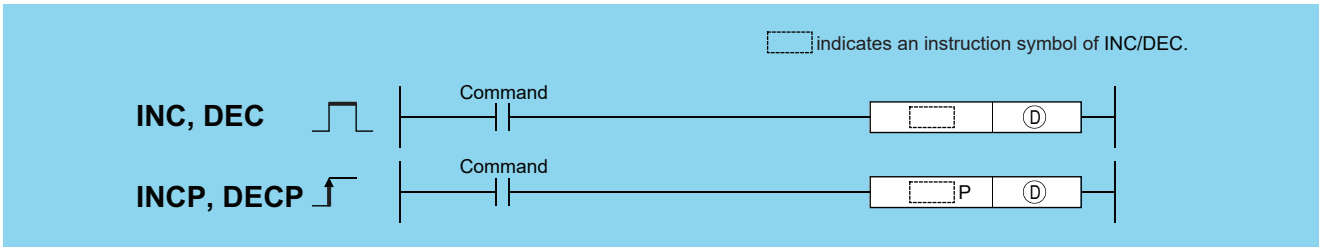
[Operation]



16-bit BIN data increment, 16-bit BIN data decrement

INC(P), DEC(P)

Basic | High performance | Process | Redundant | Universal | LCPU



(D): Head number of devices for INC (+1)/DEC (-1) operation (BIN 16 bits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant	Others
	Bit	Word		Bit	Word				
(D)	○							—	

Processing details

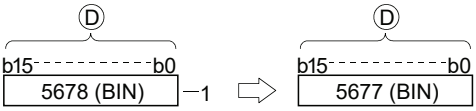
■INC

- Adds 1 to the device designated by (D) (16-bit data).

- When INC/INCP operation is executed for the device designated by (D), whose content is 32767, the value -32768 is stored at the device designated by (D).

■DEC

- Subtracts 1 from the device designated by (D) (16-bit data).



- When DEC/DECP operation is executed for the device designated by (D), whose content is -32768, the value 32767 is stored at the device designated by (D).

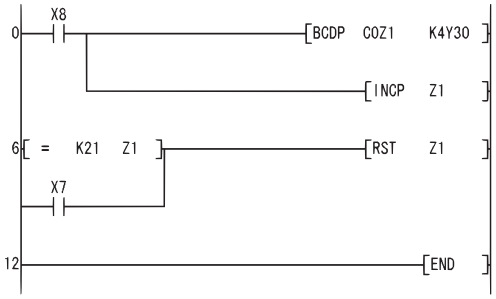
Operation error

- There is no operation error in the INC(P) or DEC(P) instruction.

Program example

- The following program outputs the present value at the counter C0 to C20 to the area Y30 to Y3F in BCD, every time X8 is turned ON. (When present value is less than 9999)

[Ladder Mode]



Outputs the present value of (0+Z1) to Y30 to Y3F in BCD.

Executes Z1 + 1.

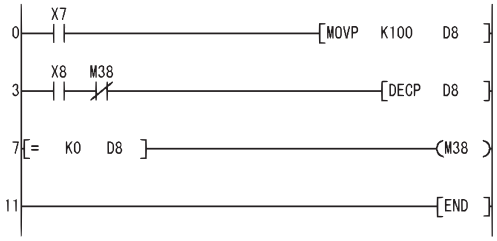
Sets Z1 at "0" when Z1=21 or X7 (reset input) is ON.

[List Mode]

Step	Instruction	Device	
0	LD	X8	
1	BCDP	C0Z1	K4Y30
4	INCP	Z1	
6	LD=	K21	Z1
9	OR	X7	
10	RST	Z1	
12	END		

- The following is a down counter program.

[Ladder Mode]



Transfers 100 to D8 when X7 goes ON.

In the state M38=OFF, decrement at D8 (D8 - 1) is executed when X8 goes from OFF to ON.

At D8=0, M38 goes ON.

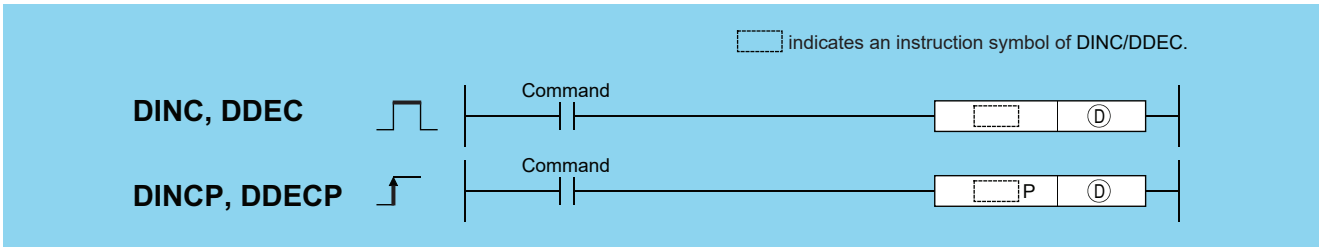
[List Mode]

Step	Instruction	Device	
0	LD	X7	
1	MOVP	K100	D8
3	LD	X8	
4	ANI	M38	
5	DECP	D8	
7	LD=	K0	D8
10	OUT	M38	
11	END		

32-bit BIN data increment, 32-bit BIN data decrement

DINC(P), DDEC(P)

Basic | High performance | Process | Redundant | Universal | LCPU



(D): Head number of devices for DINC(+1) or DDEC(-1) operation (BIN 32 bits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant	Others
	Bit	Word		Bit	Word				
(D)	○							—	

Processing details

■DINC

- Adds +1 to the device designated by (D) (32-bit data).

- When DINC/DINCP operation is executed for the device designated by (D), whose content is 2147483647, the value -2147483648 is stored at the device designated by (D).

■DDEC

- Subtracts 1 from the device designated by (D) (32-bit data).

- When DDEC/DDECP operation is executed for the device designated by (D), whose content is 0, the value -1 is stored at the device designated by (D).

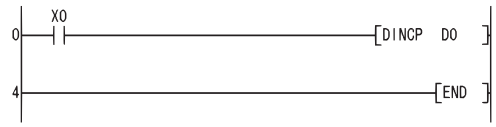
Operation error

- There is no operation error in the DINC(P) or DDEC(P) instruction.

Program example

• The following program adds 1 to the data at D0 and D1 when X0 is ON.

[Ladder Mode]

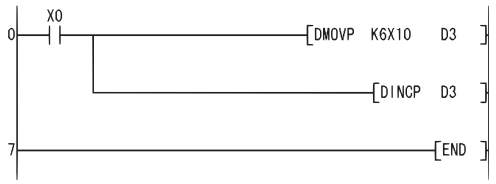


[List Mode]

Step	Instruction	Device
0	LD	X0
1	DINCP	D0
4	END	

• The following program adds 1 to the data set at X10 to X27 when X0 goes ON, and stores the result at D3 and D4.

[Ladder Mode]

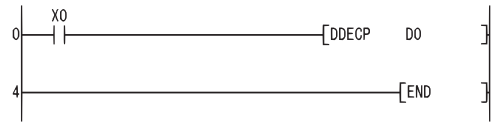


[List Mode]

Step	Instruction	Device
0	LD	X0
1	DMOVP	K6X10 D3
4	DINCP	D3
7	END	

• The following program subtracts 1 from the data at D0 and D1 when X0 goes ON.

[Ladder Mode]

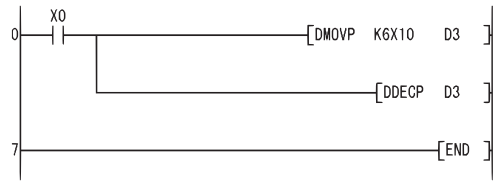


[List Mode]

Step	Instruction	Device
0	LD	X0
1	DDECP	D0
4	END	

• The following program subtracts 1 from the data set at X10 to X27 when X0 goes ON, and stores the result at D3 and D4.

[Ladder Mode]



[List Mode]

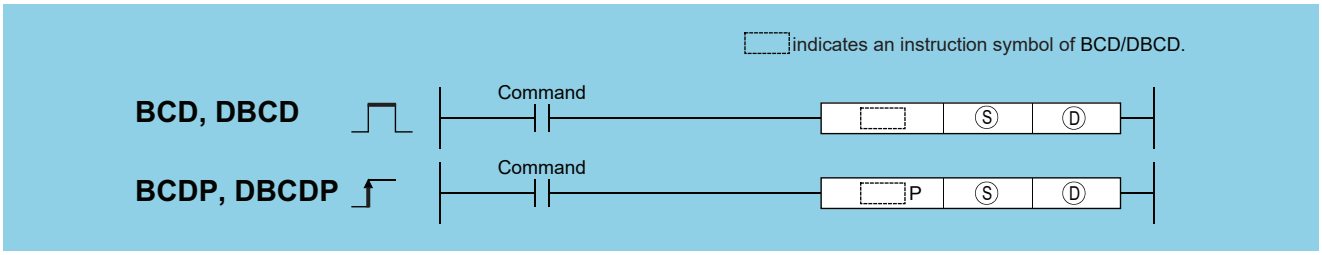
Step	Instruction	Device
0	LD	X0
1	DMOVP	K6X10 D3
4	DDECP	D3
7	END	

6.3 Data Conversion Instructions

Conversion from BIN data to BCD 4-digit data, conversion from BIN data to BCD 8-digit data

BCD(P), DBCD(P)

Basic | High performance | Process | Redundant | Universal | LCPU



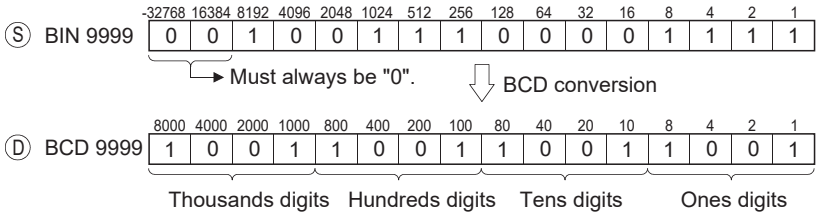
(S): BIN data or head number of the devices where the BIN data is stored (BIN 16/32 bits)
(D): Head number of the devices where BCD data will be stored (BCD 4/8 digits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(S)	○							○	—
(D)	○							—	—

Processing details

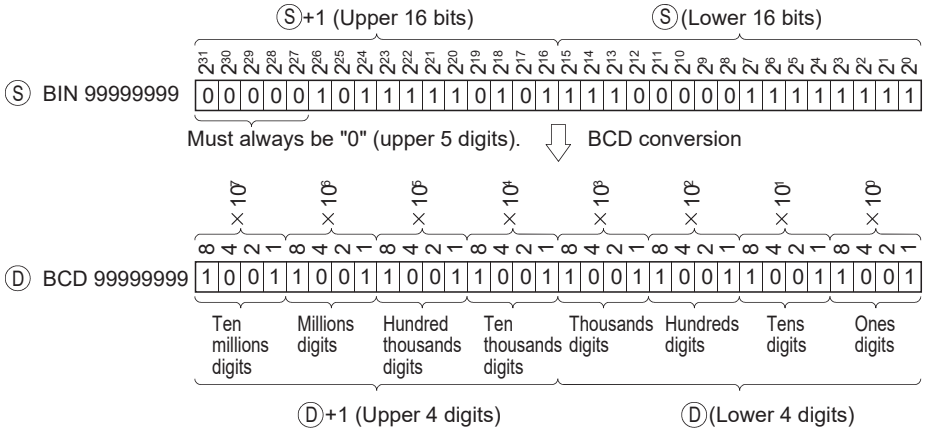
■BCD

- Converts BIN data (0 to 9999) at the device designated by (S) to BCD data, and stores it at the device designated by (D).



■DBCD

- Converts BIN data (0 to 99999999) at the device designated by (S) to BCD data, and stores it at the device designated by (D).



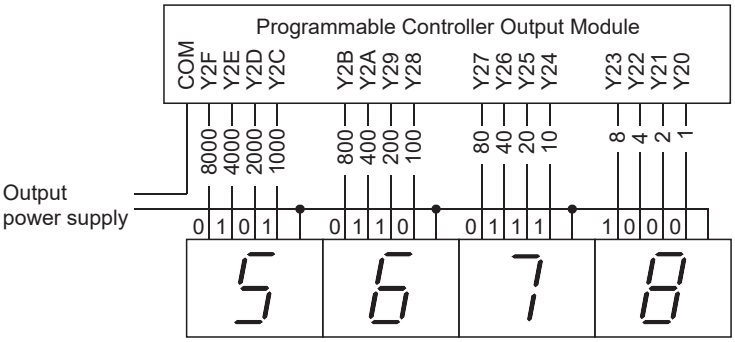
Operation error

• In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4100	The data of (S) is other than 0 to 9999 when the BCD instruction is executed.	○	○	○	○	○	○
	The data of (S) or (S)+1 is other than 0 to 99999999 when the DBCD instruction is executed.	○	○	○	○	○	○

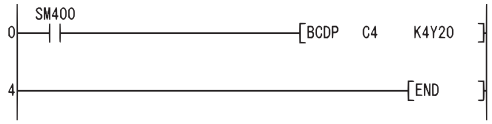
Program example

• The following program outputs the present value of C4 from Y20 to Y2F to the BCD display device.



7-segment display unit

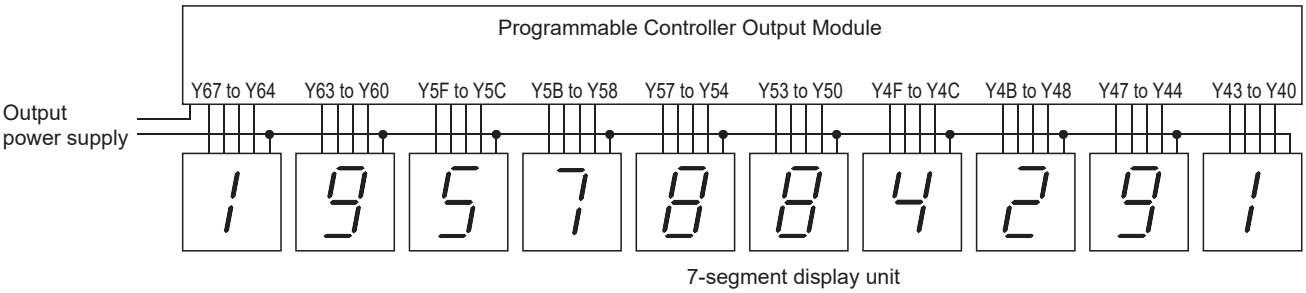
[Ladder Mode]



[List Mode]

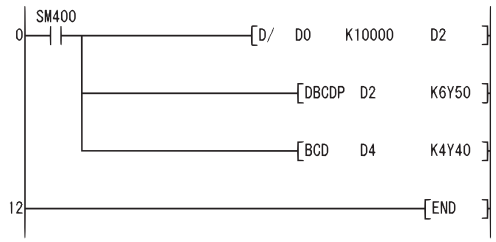
Step	Instruction	Device	
0	LD	SM400	
1	BCDP	C4	K4Y20
4	END		

• The following program outputs 32-bit data from D0 to D1 to Y40 to Y67.



7-segment display unit

[Ladder Mode]



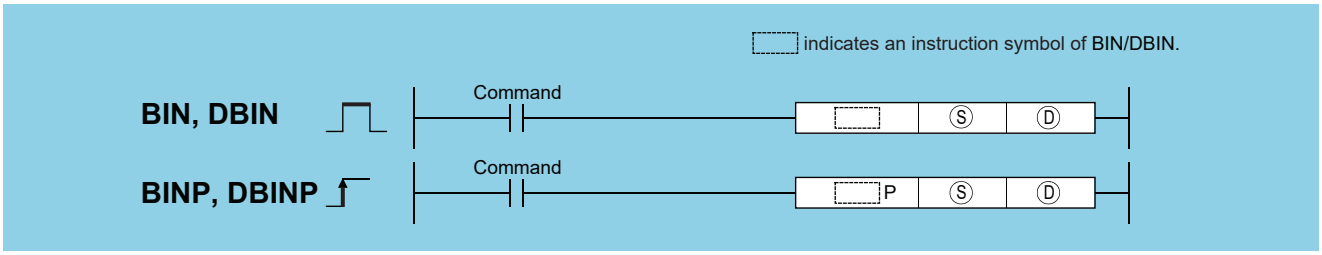
[List Mode]

Step	Instruction	Device		
0	LD	SM400		
1	D/	D0	K10000	D2
6	DBCDP	D2	K6Y50	
9	BCD	D4	K4Y40	
12	END			

Conversion from BCD 4-digit data to BIN data, conversion from BCD 8-digit data to BIN data

BIN(P), DBIN(P)

Basic | High performance | Process | Redundant | Universal | LCPU



(S): BCD data or head number of the devices where the BCD data is stored (BCD 4/8 digits)
(D): Head number of the devices where BIN data will be stored (BIN 16/32 bits)

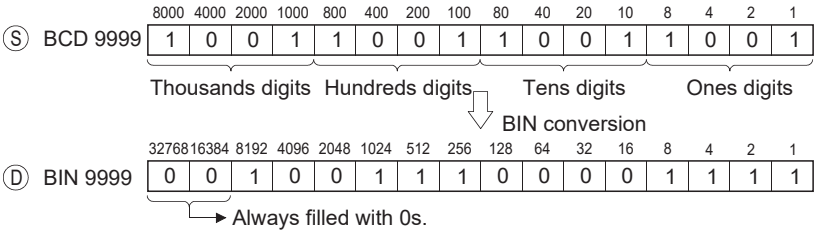
Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(S)	○							○	—
(D)	○							—	—

6

Processing details

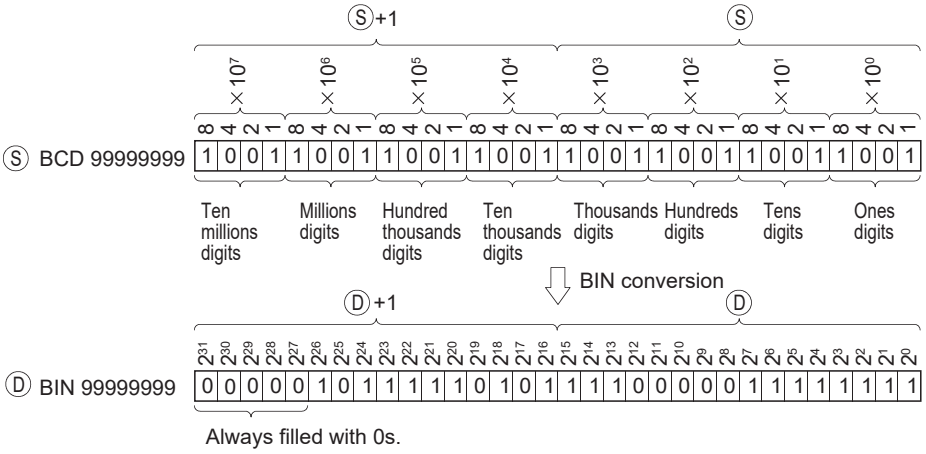
■BIN

• Converts BCD data (0 to 9999) at device designated by (S) to BIN data, and stores at the device designated by (D).



■DBIN

• Converts BCD data (0 to 99999999) at device designated by (S) to BIN data, and stores at the device designated by (D).



Operation error

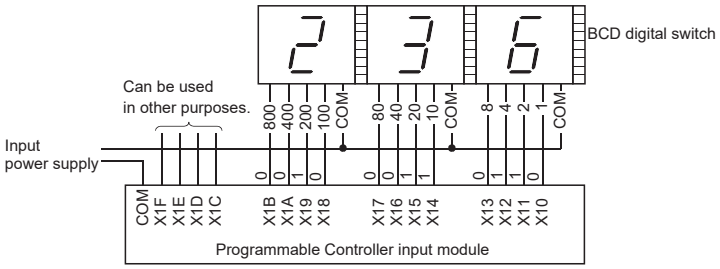
• In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4100	When values other than 0 to 9 are specified to any digits of (S).	○	○	○	○	○	○

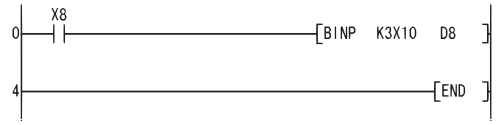
The error above can be suppressed by turning ON SM722.
However, the instruction is not executed regardless of whether SM722 is turned ON or OFF if the designated value is out of the available range.
For the BINP/DBINP instruction, the next operation will not be performed until the command (execution condition) is turned from OFF to ON regardless of the presence/absence of an error.

Program example

• The following program converts the BCD data at X10 to X1B to BIN when X8 is ON, and stores it at D8.



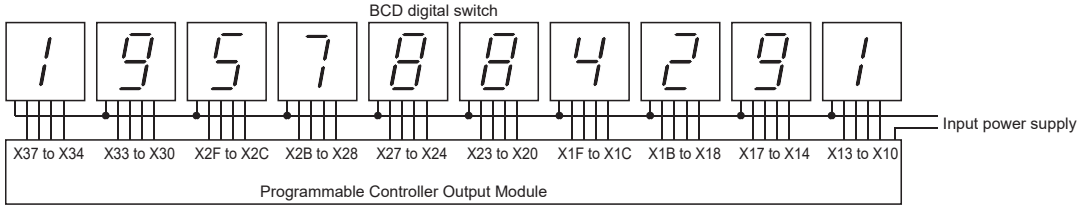
[Ladder Mode]



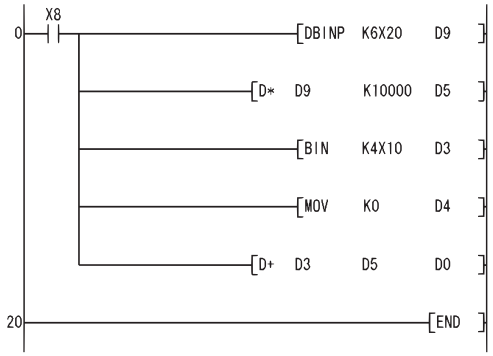
[List Mode]

Step	Instruction	Device
0	LD	X8
1	BINP	K3X10 D8
4	END	

• The following program converts the BCD data at X10 to X37 to BIN when X8 is ON, and stores it at D0 and D1. (Addition of the BIN data converted from BCD at X20 to X37 and the BIN data converted from BCD at X10 to X1F)



[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	X8
1	DBINP	K6X20 D9
4	D*	D9 K10000 D5
9	BIN	K4X10 D3
12	MOV	K0 D4
14	D+	D3 D5 D0
20	END	

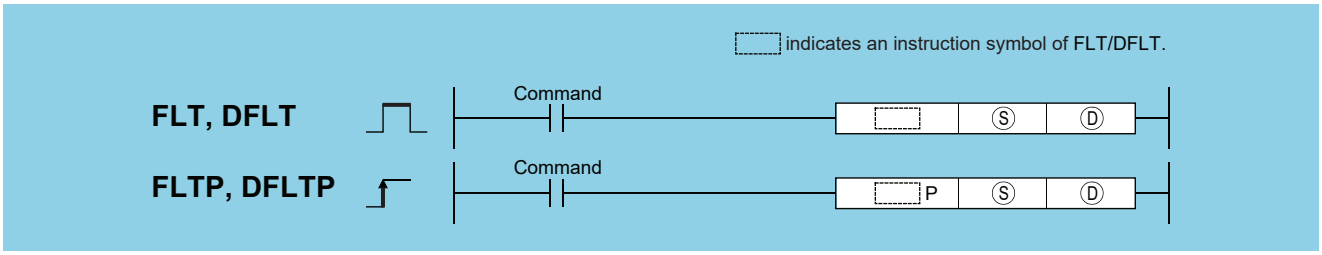
If the data set at X10 to X37 is a BCD value which exceeds 2147483647, the value at D0 and D1 will be a negative value, because it exceeds the range of numerical values that can be handled by a 32-bit device.

Conversion from BIN 16-bit data to floating-point data (single precision), conversion from BIN 32-bit data to floating-point data (single precision)

FLT(P), DFLT(P)

Ver. Basic | High performance | Process | Redundant | Universal | LCPU

• Basic model QCPU: The serial number (first five digits) is "04122" or later.



(S): Integer data to be converted to 32-bit floating decimal point data or head number of the devices where the integer data is stored (BIN 16/32 bits)
(D): Head number of the devices where the converted 32-bit floating decimal point data will be stored (real number)

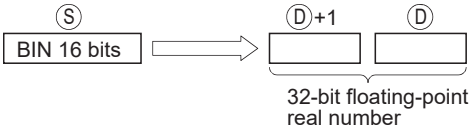
Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(S)	○	○		○	○		○	○	—
(D)	—	○		—	○		○*1	—	—

*1 Applicable for the Universal model QCPU, LCPU.

Processing details

■FLT

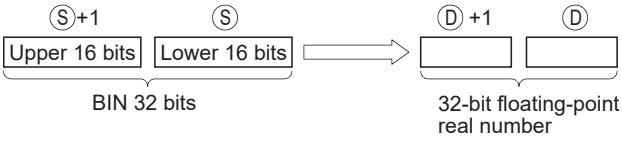
• Converts 16-bit BIN data designated by (S) to 32-bit floating decimal point type real number, and stores at device number designated by (D).



• BIN values between -32768 to 32767 can be designated by (S).

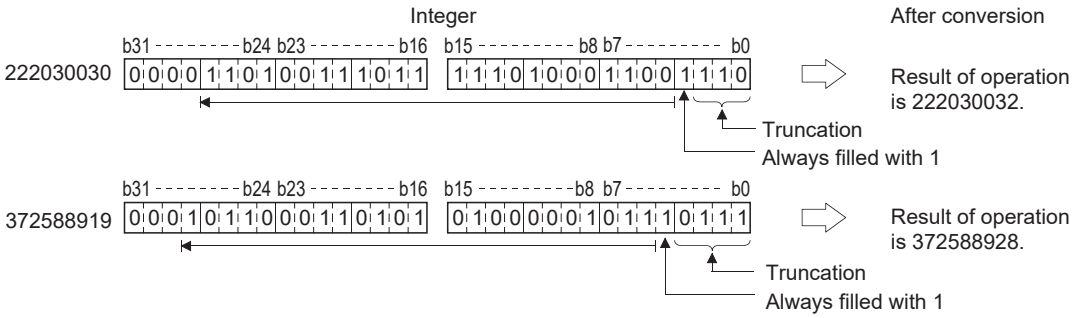
■DFLT

• Converts 32-bit BIN data designated by (S) to 32-bit floating decimal point type real number, and stores at device number designated by (D).



• BIN values between -2147483648 to 2147483647 can be designated by (S)+1 and (S).

- Due to the fact that 32-bit floating decimal point type real numbers are processed by simple 32-bit processing, the number of significant digits is 24 bits if the display is binary and approximately 7 digits if the display is decimal. For this reason, if the integer exceeds the range of -16777216 to 16777215 (24-bit BIN value), errors can be generated in the conversion value. As for the conversion result, the 25th bit from the upper bit of the integer is always filled with 1 and 26th bit and later bits are truncated.



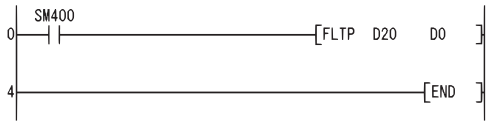
Operation error

- There is no operation error in the FLT(P) or DFLT(P) instruction.

Program example

- The following program converts the BIN 16-bit data at D20 to a 32-bit floating decimal point type real number and stores the result at D0 and D1.

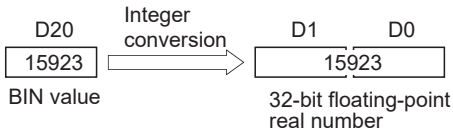
[Ladder Mode]



[List Mode]

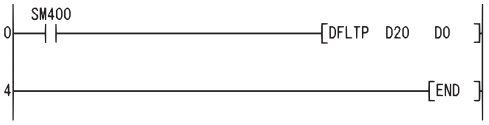
Step	Instruction	Device	
0	LD	SM400	
1	FLTP	D20	D0
4	END		

[Operation]



- The following program converts the BIN 32-bit data at D20 and D21 to a 32-bit floating decimal point type real number, and stores the result at D0 and D1.

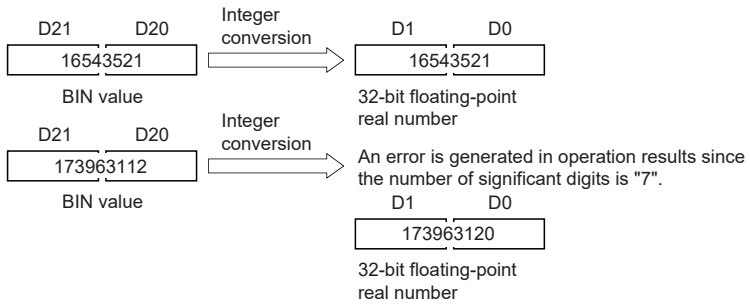
[Ladder Mode]



[List Mode]

Step	Instruction	Device	
0	LD	SM400	
1	DFLTP	D20	D0
4	END		

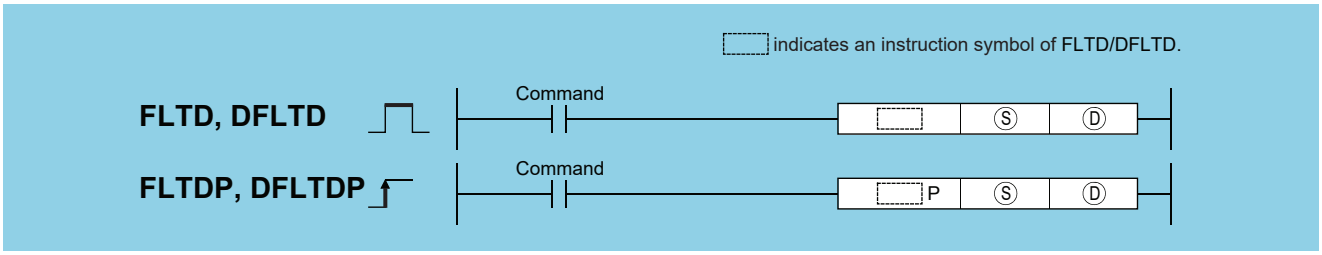
[Operation]



Conversion from BIN 16-bit data to floating-point data (double precision), conversion from BIN 32-bit data to floating-point data (double precision)

FLTD(P), DFLTD(P)

Basic | High performance | Process | Redundant | Universal | LCPU



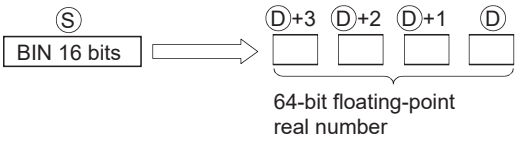
(S): Integer data to be converted to 64-bit floating decimal point data or head number of the devices where the integer data is stored (BIN 16/32 bits)
(D): Head number of the devices where the converted 64-bit floating decimal point data will be stored (real number)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(S)	○	○		—			○		—
(D)	—	○		—			—		—

Processing details

■FLTD

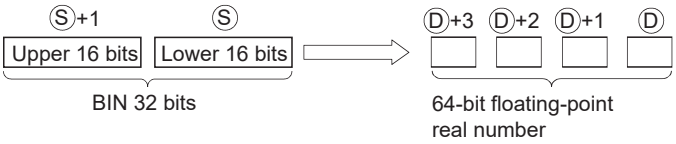
- Converts 16-bit BIN data designated by (S) to 64-bit floating decimal point type real number, and stores at device number designated by (D).



- BIN values between -32768 to 32767 can be designated by (S).

■DFLTD

- Converts 64-bit BIN data designated by (S) to 32-bit floating decimal point type real number, and stores at device number designated by (D).



- BIN values between -2147483648 to 2147483647 can be designated by (S)+1 and (S).

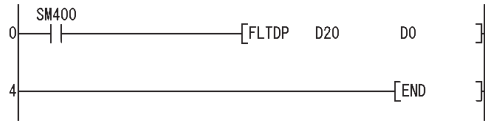
Operation error

- There is no operation error in the FLT(P) or DFLT(P) instruction.

Program example

- The following program converts the BIN 16-bit data at D20 to a 64-bit floating decimal point type real number and stores the result at D0 to D3.

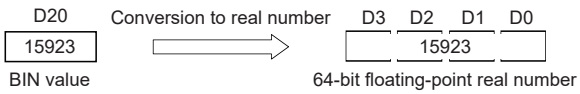
[Ladder Mode]



[List Mode]

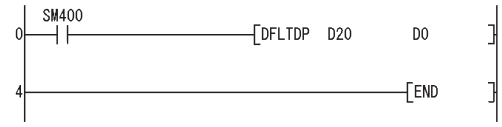
Step	Instruction	Device
0	LD	SM400
1	FLTDP	D20 D0
4	END	

[Operation]



- The following program converts the BIN 32-bit data at D20 and D21 to a 64-bit floating decimal point type real number, and stores the result at D0 to D3.

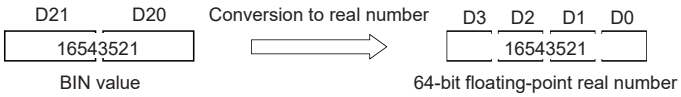
[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	SM400
1	DFLTDP	D20 D0
4	END	

[Operation]

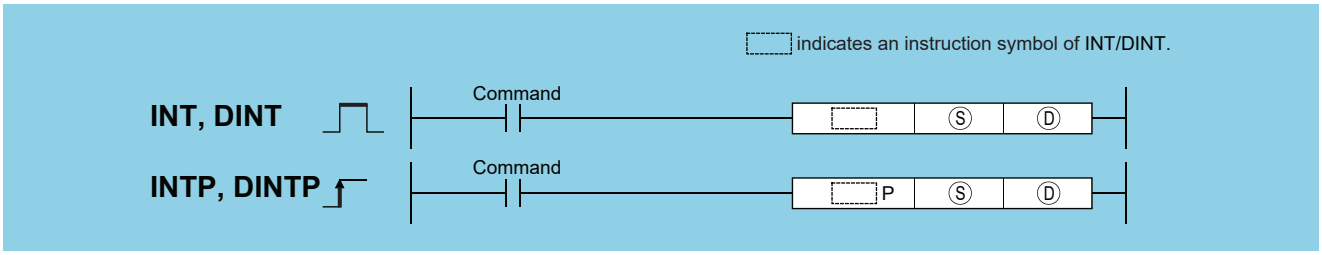


Conversion from floating-point data to BIN 16-bit data (single precision), conversion from floating-point data to BIN 32-bit data (single precision)

INT(P), DINT(P)

Ver. Basic | High performance | Process | Redundant | Universal | LCPU

• Basic model QCPU: The serial number (first five digits) is "04122" or later.



(S): 32-bit floating decimal point data to be converted to BIN value or head number of the devices where the floating decimal point data is stored (real number)
(D): Head number of the devices where the converted BIN value will be stored (BIN 16/32 bits)

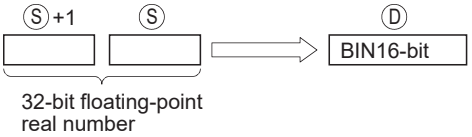
Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant E	Others
	Bit	Word		Bit	Word				
(S)	—	○		—	○		○*1	○	—
(D)	○	○		○	○		○	—	—

*1 Applicable for the Universal model QCPU, LCPU.

Processing details

■INT

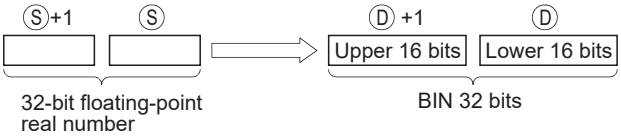
• Converts the 32-bit floating decimal point real number designated at (S) into BIN 16-bit data and stores it at the device number designated at (D).



• The range of 32-bit floating decimal point type real numbers that can be designated at (S)+1 or (S) is from -32768 to 32767.
• Stores integer values stored at (D) as BIN 16-bit values.
• After conversion, the first digit after the decimal point of the real number is rounded off.
• When an input value is set using a programming tool, a rounding error may occur. For precautions, refer to Page 90 Precautions.

■DINT

• Converts 32-bit floating decimal point type real number designated by (S) to BIN 32-bit data, and stores the result at the device number designated by (D).



• The range of 32-bit floating decimal point type real numbers that can be designated at (S)+1 or (S) is from -2147483648 to 2147483647.
• The integer value stored at (D)+1 and (D) is stored as BIN 32 bits.
• After conversion, the first digit after the decimal point of the real number is rounded off.
• When an input value is set using a programming tool, a rounding error may occur. For precautions, refer to Page 90 Precautions.

Operation error

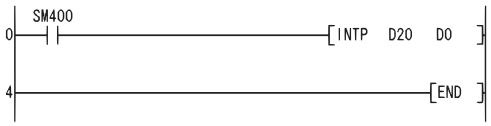
- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/Q01	QnH	QnPH	QnPRH	QnU	LCPU
4140	The specified device value is not within the following range: 0, $2^{-126} \leq$ | Specified device value | $< 2^{128}$ The specified device value is -0, unnormalized number, nonnumeric, and $\pm\infty$.	—	—	—	—	○	○
4100	The 32-bit floating point data specified by (S) when the INT instruction is used is outside the -32768 to 32767 range.	○	○	○	○	○	○
	The 32-bit floating point data specified by (S) when the DINT instruction is used is outside the -2147483648 to 2147483647 range.	○	○	○	○	○	○

Program example

- The following program converts the 32-bit floating decimal point type real number at D20 and D21 to BIN 16-bit data, and stores the result at D0.

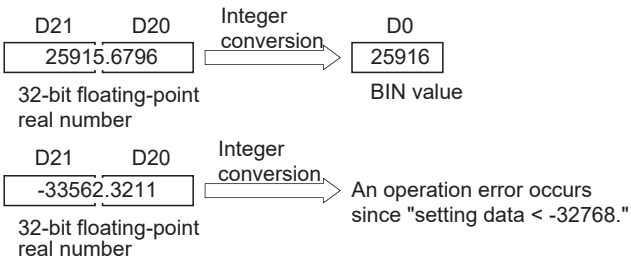
[Ladder Mode]



[List Mode]

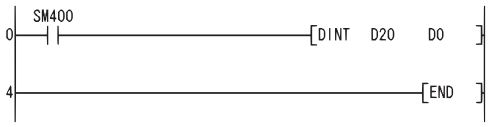
Step	Instruction	Device	
0	LD	SM400	
1	INTP	D20	D0
4	END		

[Operation]



- The following program converts the 32-bit floating decimal point type real number at D20 and D21 to BIN 32-bit data and stores the result at D0 and D1.

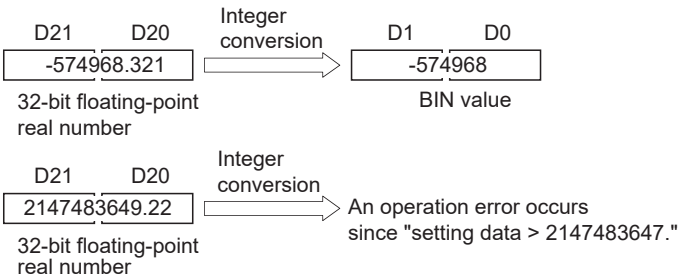
[Ladder Mode]



[List Mode]

Step	Instruction	Device	
0	LD	SM400	
1	DINT	D20	D0
4	END		

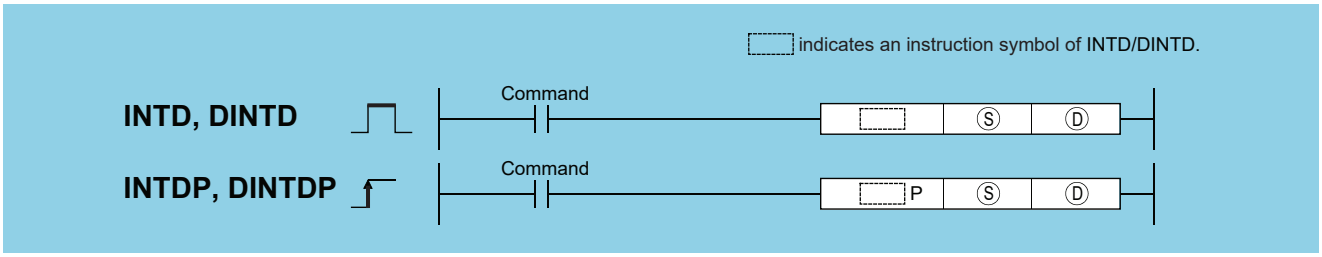
[Operation]



Conversion from floating-point data to BIN 16-bit data (double precision), conversion from floating-point data to BIN 32-bit data (double precision)

INTD(P), DINTD(P)

Basic | High performance | Process | Redundant | Universal | LCPU



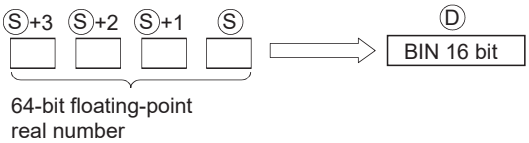
(S): 64-bit floating decimal point data to be converted to BIN value or head number of the devices where the floating decimal point data is stored (real number)
(D): Head number of the devices where the converted BIN value will be stored (BIN 16/32 bits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant E	Others
	Bit	Word		Bit	Word				
(S)	—	○		—			—	○	—
(D)	○	○		—			○	—	—

Processing details

■INTD

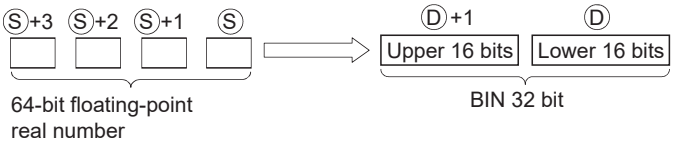
- Converts the 64-bit floating decimal point real number designated at (S) into BIN 16-bit data and stores it at the device number designated at (D).



- The range of 64-bit floating decimal point type real numbers that can be designated at (S)+3, (S)+2, (S)+1 or (S) is from -32768 to 32767.
- Stores integer values stored at (D) as BIN 16-bit values.
- The converted data is the value rounded 64-bit floating-point real number to the first digit after the decimal point.
- When an input value is set using a programming tool, a rounding error may occur. For precautions, refer to Page 90 Precautions.

■DINTD

- Converts 64-bit floating decimal point type real number designated by (S) to BIN 32-bit data, and stores the result at the device number designated by (D).



- The range of 64-bit floating decimal point type real numbers that can be designated at (S)+3, (S)+2, (S)+1 or (S) is from -2147483648 to 2147483647.
- The integer value stored at (D)+1 and (D) is stored as BIN 32 bits.
- The converted data is the value rounded 64-bit floating-point real number to the first digit after the decimal point.
- When an input value is set using a programming tool, a rounding error may occur. For precautions, refer to Page 90 Precautions.

Operation error

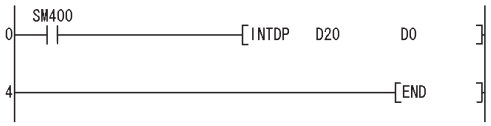
- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/Q01	QnH	QnPH	QnPRH	QnU	LCPU
4140	(S) is -0, a subnormal number, NaN (not a number), ±∞, or a value outside the following ranges. $0, 2^{-1022} \leq \|(S)\| < 2^{1024}$	—	—	—	—	○	○
4100	The 64-bit floating point data specified by (S) when the INTD instruction is used is outside the -32768 to 32767 range.	—	—	—	—	○	○
	The 64-bit floating point data specified by (S) when the DINTD instruction is used is outside the -2147483648 to 2147483647 range.	—	—	—	—	○	○

Program example

- The following program converts the 64-bit floating decimal point type real number at D20 to D23 with BIN 16-bit data, and stores the result at D0.

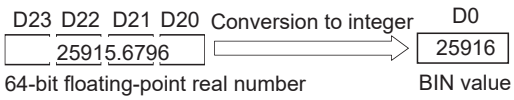
[Ladder Mode]



[List Mode]

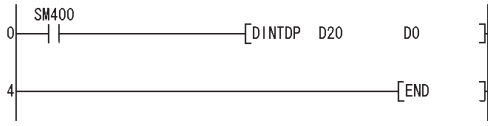
Step	Instruction	Device	
0	LD	SM400	
1	INTDP	D20	D0
4	END		

[Operation]



- The following program converts the 64-bit floating decimal point type real number at D20 to D23 with BIN 32-bit data and stores the result at D0 and D1.

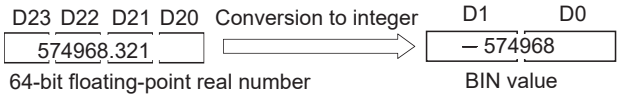
[Ladder Mode]

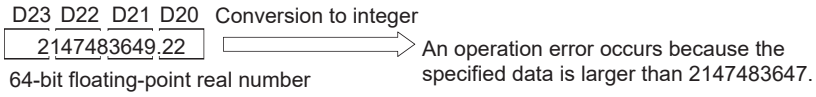


[List Mode]

Step	Instruction	Device	
0	LD	SM400	
1	DINTDP	D20	D0
4	END		

[Operation]





Conversion from BIN 16-bit to BIN 32-bit data

DBL(P)

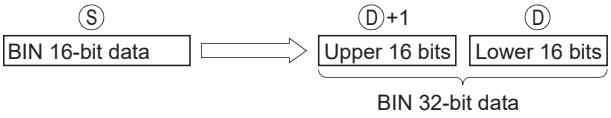
Basic | High performance | Process | Redundant | Universal | LCPU

(S): BIN 16-bit data or head number of the devices where the BIN 16-bit data is stored (BIN 16 bits)
(D): Head number of the devices where the converted BIN 32-bit data will be stored (BIN 32 bits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(S)	○							○	—
(D)	○							—	—

Processing details

- Converts BIN 16-bit data at device designated by (S) to BIN 32-bit data with sign, and stores the result at a device designated by (D).



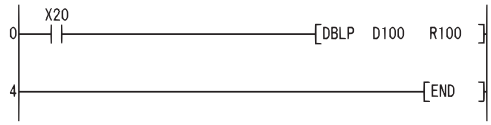
Operation error

- There is no operation error in the DBL(P) instruction.

Program example

- The following program converts the BIN 16-bit data stored at D100 to BIN 32-bit data when X20 is ON, and stores at R100 and R101.

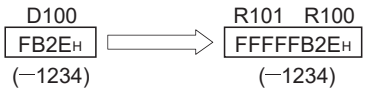
[Ladder Mode]



[List Mode]

Step	Instruction	Device	
0	LD	X20	
1	DBLP	D100	R100
4	END		

[Operation]



Conversion from BIN 32-bit to BIN 16-bit data

WORD(P)

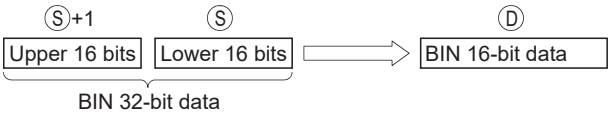
Basic | High performance | Process | Redundant | Universal | LCPU

(S): BIN 32-bit data or head number of the devices where the BIN 32-bit data is stored (BIN 32 bits)
(D): Head number of the devices where the converted BIN 16-bit data will be stored (BIN 16 bits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(S)	○							○	—
(D)	○							—	—

Processing details

- Converts BIN 32-bit data at device designated by (S) to BIN 16-bit data with sign, and stores the result at a device designated by (D).
- Devices can be designated in the range from -32768 to 32767.



Operation error

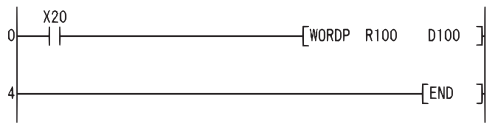
- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4100	The data specified by (S)+1 and (S) are outside the range of -32768 to 32767.	○	○	○	○	○	○

Program example

- The following program converts the BIN 32-bit data at R100 and R101 to BIN 16-bit data when X20 is ON, and stores it at D100.

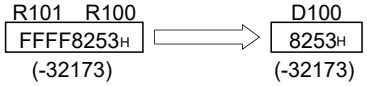
[Ladder Mode]



[List Mode]

Step	Instruction	Device	
0	LD	X20	
1	WORDP	R100	D100
4	END		

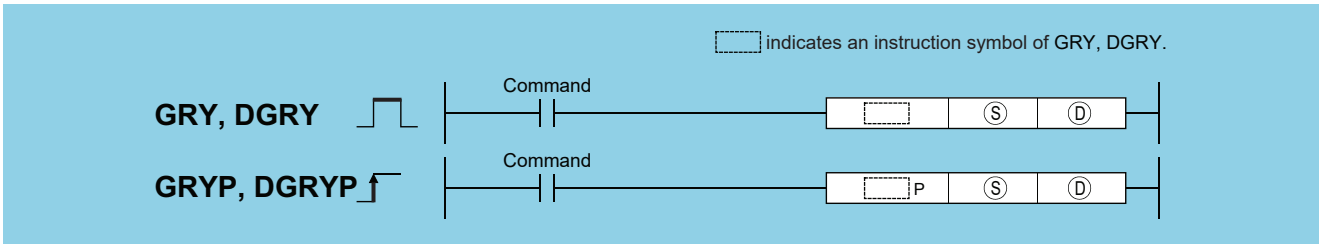
[Operation]



Conversion from BIN 16-bit data to Gray code, conversion from BIN 32-bit data to Gray code

GRY(P), DGRY(P)

Basic | High performance | Process | Redundant | Universal | LCPU



(S): BIN data or head number of the devices where the BIN data is stored (BIN 16/32 bits)
(D): Head number of the devices where the converted Gray code will be stored (BIN 16/32 bits)

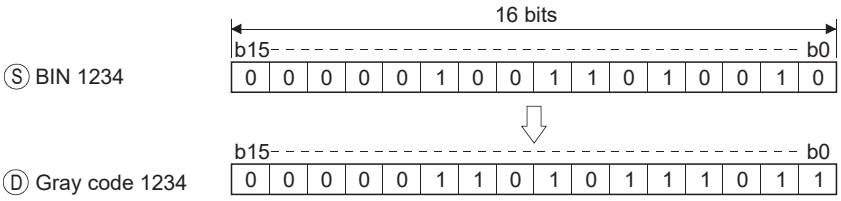
Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(S)	○							○	—
(D)	○							—	—

6

Processing details

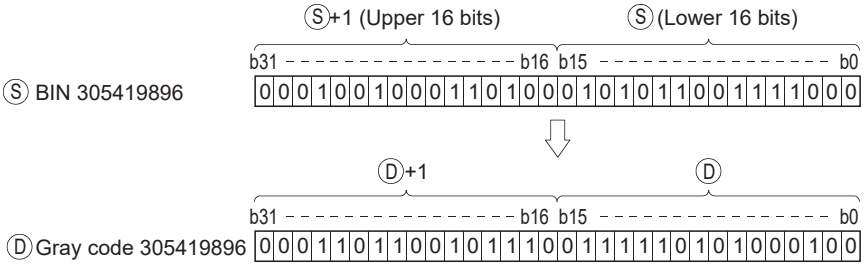
■GRY

- Converts BIN 16-bit data at the device designated by (S) to Gray code, and stores result at device designated by (D).



■DGRY

- Converts BIN 32-bit data at the device designated by (S) to Gray code, and stores result at device designated by (D).



Operation error

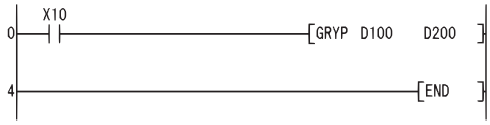
- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4100	The data at (S) is a negative value.	○	○	○	○	○	○

Program example

- The following program converts the BIN data at D100 to Gray code when X10 is ON, and stores result at D200.

[Ladder Mode]

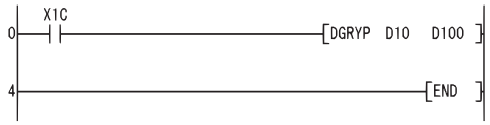


[List Mode]

Step	Instruction	Device	
0	LD	X10	
1	GRYP	D100	D200
4	END		

- The following program converts the BIN data at D10 and D11 to Gray code when X1C is ON, and stores it at D100 and D101.

[Ladder Mode]



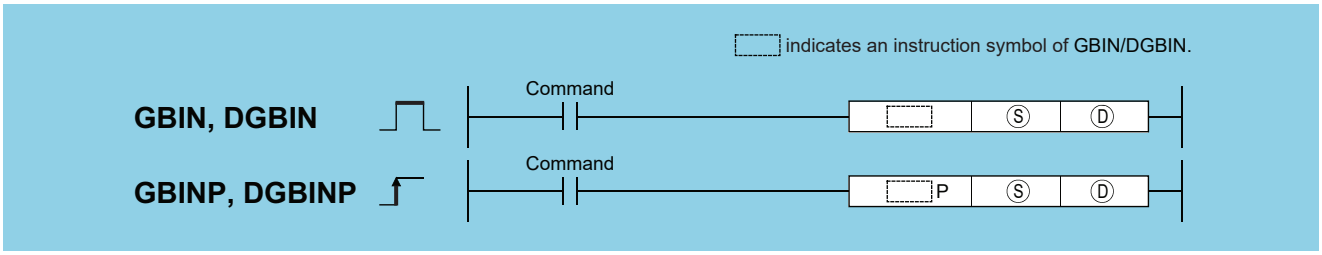
[List Mode]

Step	Instruction	Device	
0	LD	X1C	
1	DGRYP	D10	D100
4	END		

Conversion from Gray code to BIN 16-bit data, conversion from Gray code to BIN 32-bit data

GBIN(P), DGBIN(P)

Basic | High performance | Process | Redundant | Universal | LCPU



(S): Gray code data or head number of the devices where the Gray code data is stored (BIN 16/32 bits)
(D): Head number of the devices where the converted BIN data will be stored (BIN 16/32 bits)

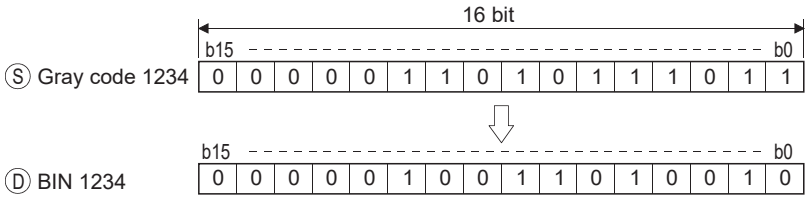
Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(S)	○							○	—
(D)	○							—	—

6

Processing details

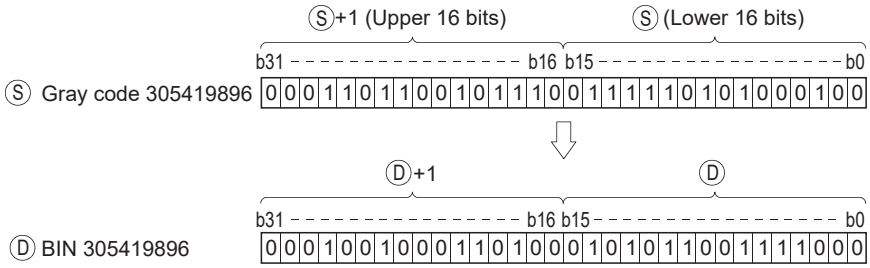
■GBIN

• Converts Gray code data at device designated by (S) to BIN 16-bit data and stores at device designated by (D).



■DGBIN

• Converts Gray code data at device designated by (S) to BIN 32-bit data and stores at device designated by (D).



Operation error

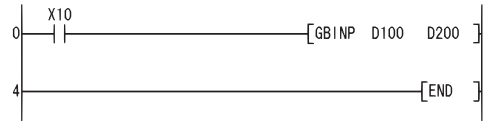
• In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4100	The data of (S) is other than 0 to 32767 when the GBIN instruction is executed.	○	○	○	○	○	○
	The data of (S) is other than 0 to 2147483647 when the DGBIN instruction is executed.	○	○	○	○	○	○

Program example

• The following program converts the Gray code data at D100 when X10 is ON to BIN data, and stores the result at D200.

[Ladder Mode]

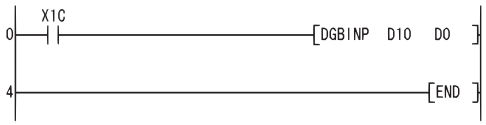


[List Mode]

Step	Instruction	Device	
0	LD	X10	
1	GBINP	D100	D200
4	END		

• The following program converts the Gray code data at D10 and D11 to BIN data when X1C is ON, and stores the result at D0 and D1.

[Ladder Mode]



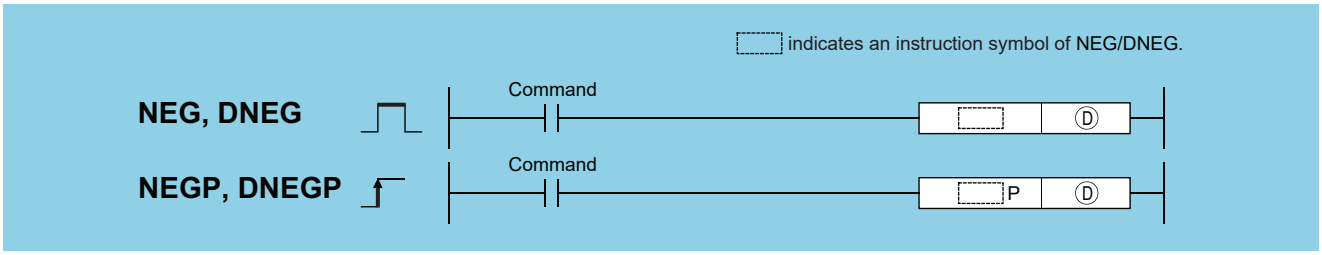
[List Mode]

Step	Instruction	Device	
0	LD	X1C	
1	DGBINP	D10	D0
4	END		

Complement of 2 of BIN 16-bit data (sign inversion), complement of 2 of BIN 32-bit data (sign inversion)

NEG(P), DNEG(P)

Basic | High performance | Process | Redundant | Universal | LCPU



(D): Head number of the devices where the data for which complement of 2 is performed is stored (BIN 16/32 bits)

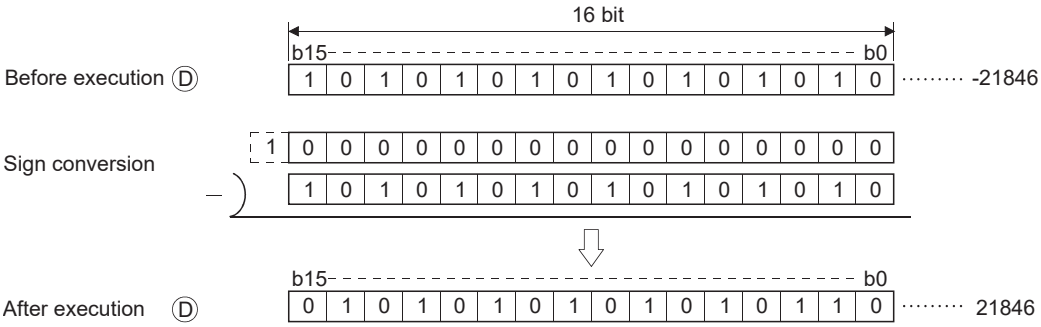
Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant	Others
	Bit	Word		Bit	Word				
(D)	○							—	

6

Processing details

■NEG

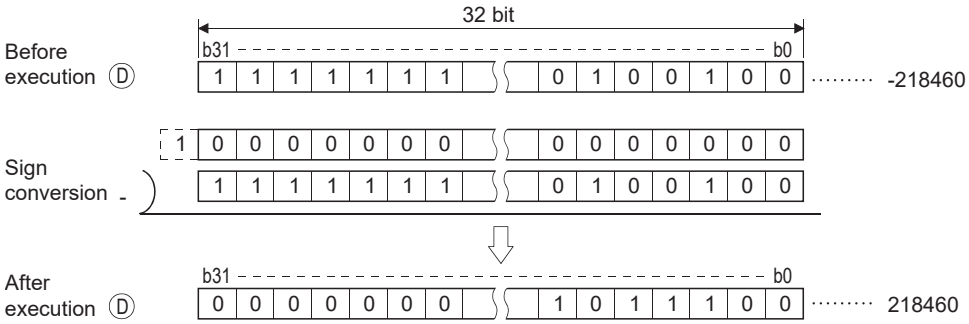
- Reverses the sign of the 16-bit device designated by (D) and stores at the device designated by (D).



- Used when reversing positive and negative signs.

■DNEG

- Reverses the sign of the 32-bit device designated by (D) and stores at the device designated by (D).



- Used when reversing positive and negative signs.

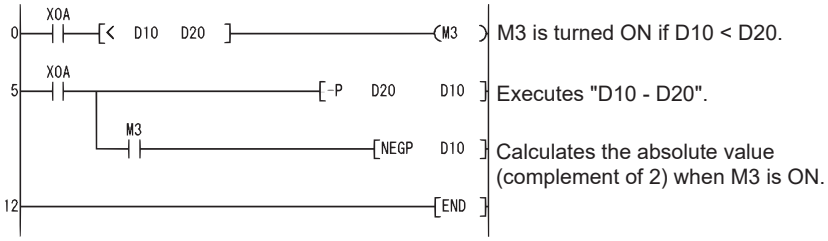
Operation error

- There is no operation error in the NEG(P) or DNEG(P) instruction.

Program example

- The following program calculates a total for the data at D10 through D20 when XA goes ON, and seeks an absolute value if the result is negative.

[Ladder Mode]



[List Mode]

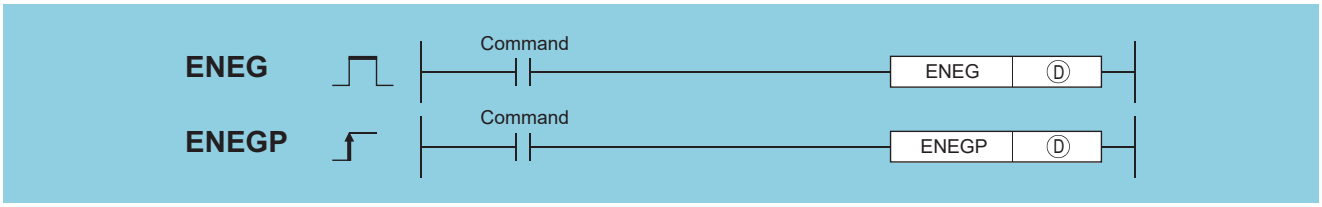
Step	Instruction	Device	
0	LD	XOA	
1	AND<	D10	D20
4	OUT	M3	
5	LD	XOA	
6	-P	D20	D10
9	AND	M3	
10	NEGP	D10	
12	END		

Floating-point sign inversion (single precision)

ENEG(P)

Ver. Basic | High performance | Process | Redundant | Universal | LCPU

• Basic model QCPU: The serial number (first five digits) is "04122" or later.



(D): Head number of the devices where the 32-bit floating decimal point data whose sign is to be reversed is stored (real number)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant	Others
	Bit	Word		Bit	Word				
(D)	—	○		—	○		○*1	—	

*1 Applicable for the Universal model QCPU, LCPU.

Processing details

• Reverses the sign of the 32-bit floating decimal point type real number data designated by (D), and stores at the device designated by (D).

• Used when reversing positive and negative signs.

Operation error

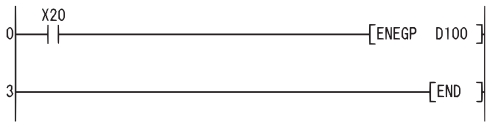
• In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4140	The specified device value is not within the following range: 0, $2^{-126} \leq$ | Specified device value | $< 2^{128}$ The specified device value is -0, unnormalized number, nonnumeric, and ±∞.	—	—	—	—	○	○

Program example

• The following program inverts the sign of the 32-bit floating decimal point type real number data at D100 and D101 when X20 goes ON, and stores result at D100 and D101.

[Ladder Mode]



[List Mode]

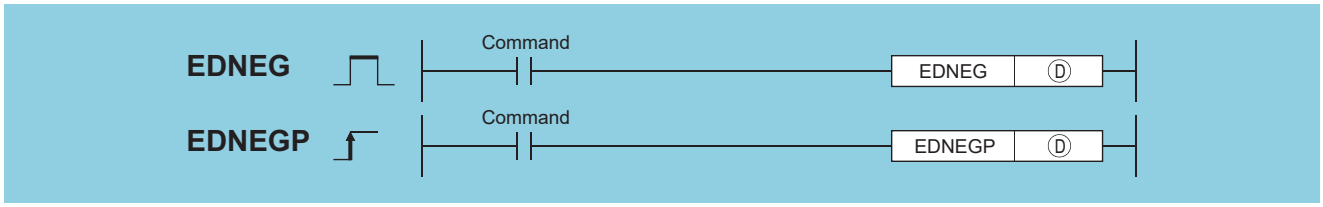
Step	Instruction	Device
0	LD	X20
1	ENEGP	D100
3	END	

[Operation]

Floating-point sign inversion (double precision)

EDNEG(P)

Basic | High performance | Process | Redundant | Universal | LCPU



(D): Head number of the devices where the 64-bit floating decimal point data whose sign is to be reversed is stored (real number)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant	Others
	Bit	Word		Bit	Word				
(D)	—	○		—					

Processing details

- Reverses the sign of the 64-bit floating decimal point type real number data designated by (D), and stores at the device designated by (D).
- Used when reversing positive and negative signs.

Operation error

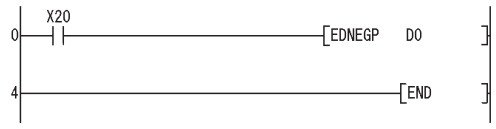
- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4140	The specified device value is not within the following range: 0, $2^{-1022} \leq$ | Specified device value | $< 2^{1024}$ The specified device value is -0, unnormalized number, nonnumeric, and ±∞.	—	—	—	—	○	○

Program example

- The following program inverts the sign of the 64-bit floating decimal point type real number data at D0 to D3 when X20 goes ON, and stores result at D0 to D3.

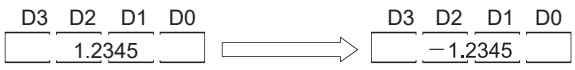
[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	X20
1	EDNEGP	D0
3	END	

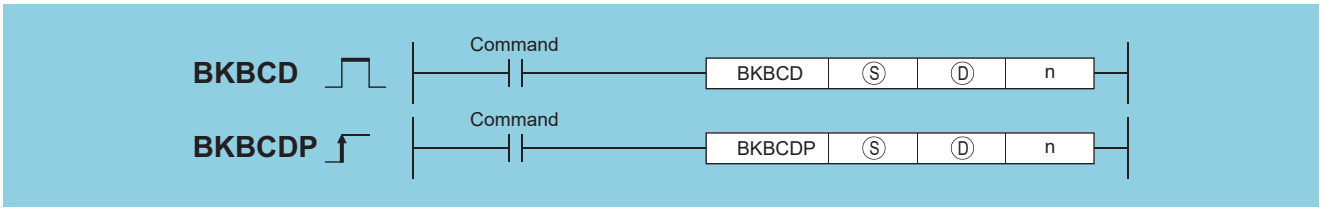
[Operation]



Conversion from block BIN 16-bit data to BCD 4-digit data

BKBCD(P)

Basic | High performance | Process | Redundant | Universal | LCPU

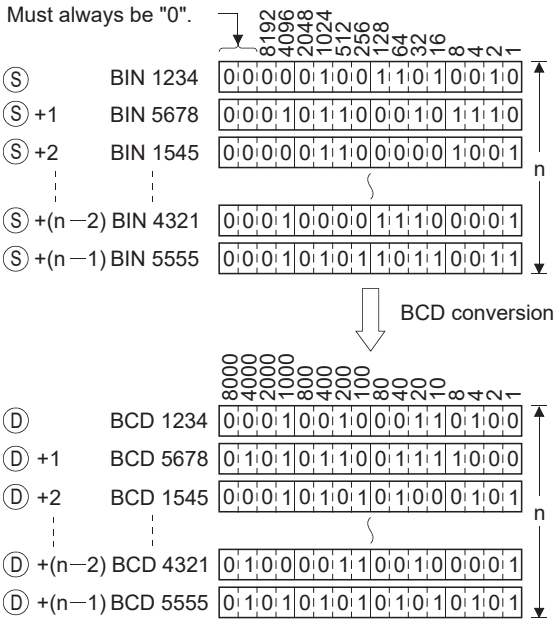


(S): Head number of the devices where BIN data is stored (BIN 16 bits)
(D): Head number of the devices where the converted BCD data will be stored (BCD 4 digits)
n: Number of variable data blocks (BIN 16 bits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(S)	—	○		—					—
(D)	—	○		—					—
n	○	○		○					—

Processing details

- Converts BIN data (0 to 9999) n points from device designated by (S) to BCD, and stores result following the device designated by (D).



Operation error

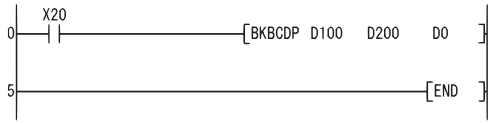
- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4100	The nth data from the device specified by (S) is outside the 0 to 9999 range.	○	○	○	○	○	○
4101	The points specified in n exceed those of the corresponding device specified in (S) or (D). The same device is specified in (S) and (D).	○	○	○	○	○	○

Program example

- The following program converts, when X20 is turned ON, the BIN data stored at D100 to D102 to BCD and stores the operation result into the area starting from D200.

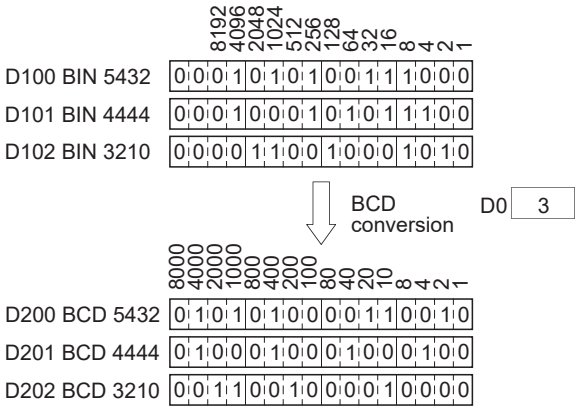
[Ladder Mode]



[List Mode]

Step	Instruction	Device		
0	LD	X20		
1	BKBCDP	D100	D200	D0
5	END			

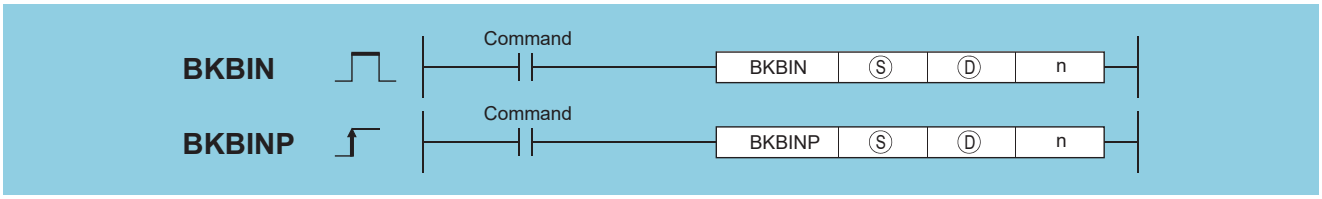
[Operation]



Conversion from block BCD 4-digit data to block BIN 16-bit data

BKBIN(P)

Basic | High performance | Process | Redundant | Universal | LCPU

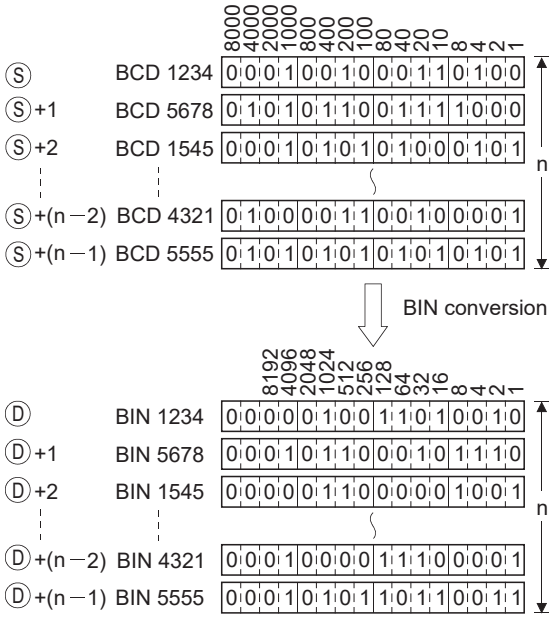


(S): Head number of the devices where BCD data is stored (BCD 4 digits)
(D): Head number of the devices where the converted BIN data will be stored (BIN 16 bits)
n: Number of variable data blocks (BIN 16 bits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(S)	—	○		—					—
(D)	—	○		—					—
n	○	○		○					—

Processing details

- Converts BCD data (0 to 9999) n points from device designated by (S) to BIN, and stores result following the device designated by (D).



Operation error

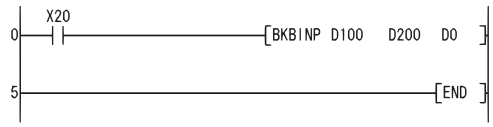
- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4100	The nth data from the device specified by (S) is outside the 0 to 9999 range.	○	○	○	○	○	○
4101	The points specified in n exceed those of the corresponding device specified in (S) or (D). The same device is specified in (S) and (D).	○	○	○	○	○	○

Program example

- The following program converts, when X20 is turned ON, the BCD data stored at D100 to D102 to BIN and stores the operation result into the area starting from D200.

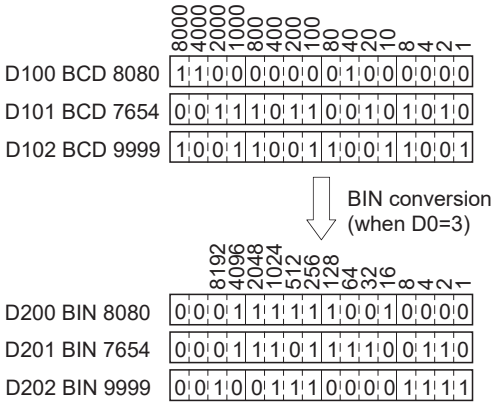
[Ladder Mode]



[List Mode]

Step	Instruction	Device		
0	LD	X20		
1	BKBINP	D100	D200	D0
5	END			

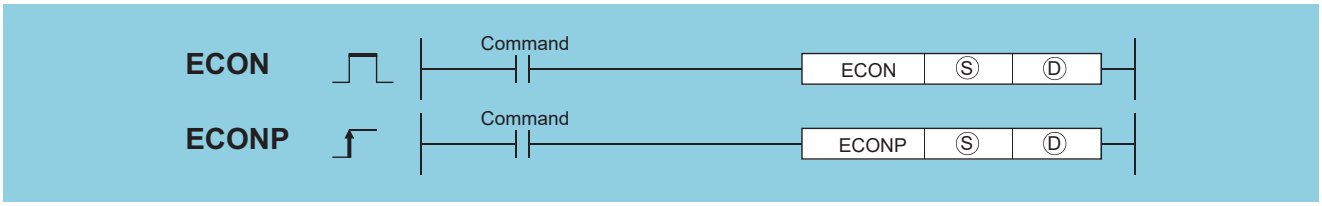
[Operation]



Conversion from single precision to double precision

ECON(P)

Basic | High performance | Process | Redundant | Universal | LCPU



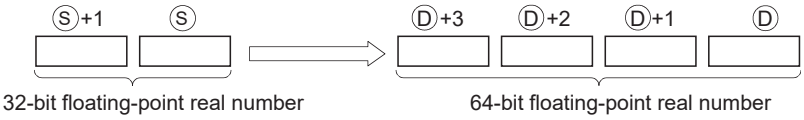
(S): Conversion source data, or head number of the device where conversion source data is stored (Real number (single precision))
(D): Head number of the device where the converted data is stored (Real number (double precision))

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant E	Others
	Bit	Word		Bit	Word				
(S)	—	○		—			○		—
(D)	—	○		—			—		—

6

Processing details

- Converts 32-bit floating-point real number specified for (S) into 64-bit floating-point real number, and stores the conversion result to the device specified for (D).



- When an input value is set using a programming tool, a rounding error may occur. For precautions, refer to Page 90 Precautions.

Operation error

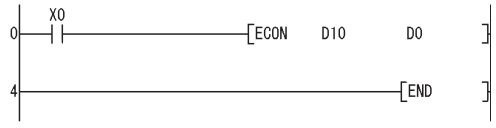
- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4140	(S) is -0, a subnormal number, NaN (not a number), ±∞, or a value outside the following ranges. 0, $2^{-126} \leq |(S)| < 2^{128}$	—	—	—	—	○	○

Program example

- The program which converts 32-bit floating-point real number of the devices, D10 to D11, into 64-bit floating-point real number when X0 turns ON, and outputs the conversion result to the devices, D0 to D3.

[Ladder Mode]



[List Mode]

Step	Instruction	Device	
0	LD	X0	
1	ECON	D10	D0
4	END		

Conversion from double precision to single precision

EDCON(P)

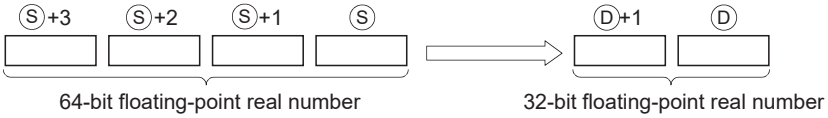
Basic | High performance | Process | Redundant | Universal | LCPU

(S): Conversion source data, or head number of the device where conversion source data is stored (Real number (double precision))
(D): Head number of the device where the converted data is stored (Real number (single precision))

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant E	Others
	Bit	Word		Bit	Word				
(S)	—	○		—			—	○	—
(D)	—	○		—			○	—	—

Processing details

- Converts 64-bit floating-point real number specified for (S) into 32-bit floating-point real number, and stores the conversion result to the device specified for (D).



- When an input value is set using a programming tool, a rounding error may occur. For precautions, refer to Page 90 Precautions.

Operation error

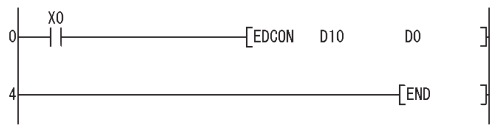
- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4140	The specified device value is not within the following range: 0, $2^{-1022} \leq$ | Specified device value | $< 2^{1024}$ The specified device value is -0, unnormalized number, nonnumeric, and ±∞.	—	—	—	—	○	○
4141	The conversion result exceeds the following range (when an overflow occurs): |Operation result| $< 2^{128}$	—	—	—	—	○	○

Program example

- The program which converts 64-bit floating-point real number of the devices, D10 to D13, into 32-bit floating-point real number when X0 turns ON, and outputs the conversion result to the devices, D0 to D1.

[Ladder Mode]



[List Mode]

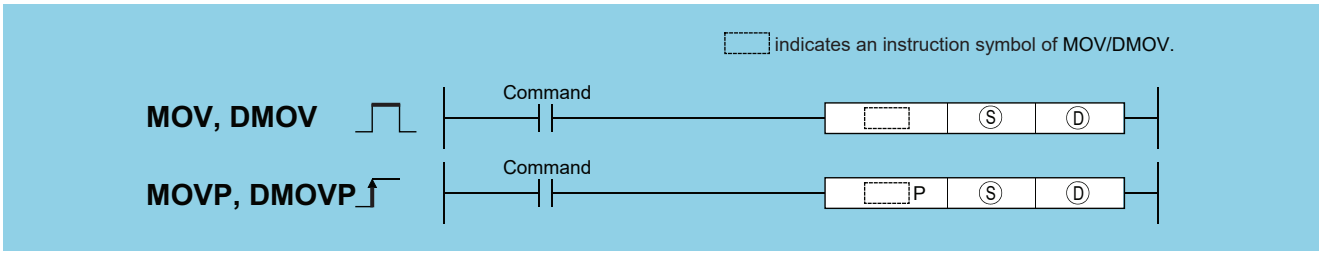
Step	Instruction	Device
0	LD	X0
1	EDCON	D10 D0
4	END	

6.4 Data Transfer Instructions

16-bit data transfer, 32-bit data transfer

MOV(P), DMOV(P)

Basic | High performance | Process | Redundant | Universal | LCPU



(S): Data to be transferred or the number of the device where the data to be transferred is stored (BIN 16/32 bits)
(D): Number of the device where the data will be transferred (BIN 16/32 bits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(S)	○							○	—
(D)	○							—	—

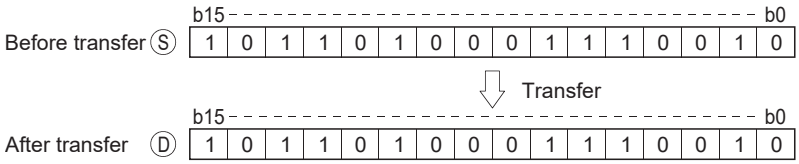
Point

When BL, S, TR, BL\S, or BL\TR is used, refer to the SFC control instructions in the MELSEC-Q/L/QnA Programming Manual (SFC).

Processing details

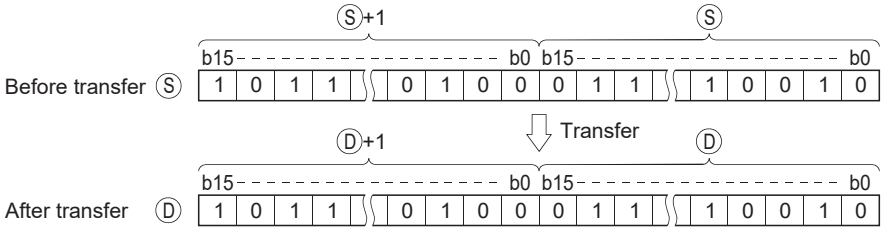
■MOV

- Transfers the 16-bit data from the device designated by (S) to the device designated by (D).



■DMOV

- Transfers 32-bit data at the device designated by (S) to the device designated by (D).



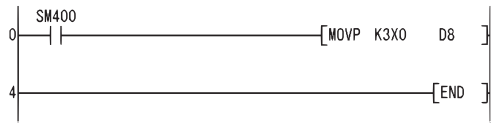
Operation error

- There is no operation error in the MOV(P) or DMOV(P) instruction.

Program example

• The following program stores input data from X0 to XB at D8.

[Ladder Mode]

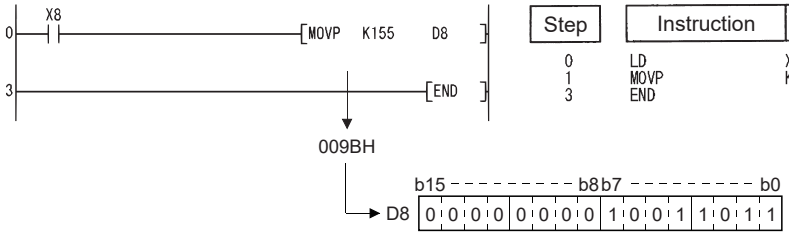


[List Mode]

Step	Instruction	Device
0	LD	SM400
1	MOVP	K3X0 D8
4	END	

• The following program stores the constant K155 at D8 when X8 goes ON.

[Ladder Mode]

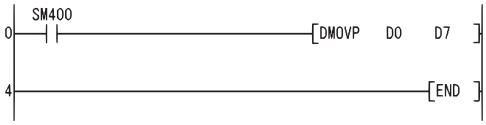


[List Mode]

Step	Instruction	Device
0	LD	X8
1	MOVP	K155 D8
3	END	

• The following program stores the data from D0 and D1 at D7 and D8.

[Ladder Mode]

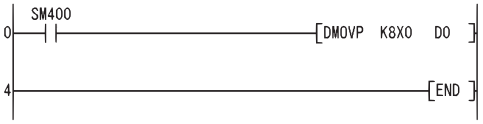


[List Mode]

Step	Instruction	Device
0	LD	SM400
1	DMOVP	D0 D7
4	END	

• The following program stores the data from X0 to X1F at D0 and D1.

[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	SM400
1	DMOVP	K8X0 D0
4	END	

Floating-point data transfer (single precision)

EMOV(P)

Ver. Basic | High performance | Process | Redundant | Universal | LCPU

• Basic model QCPU: The serial number (first five digits) is "04122" or later.

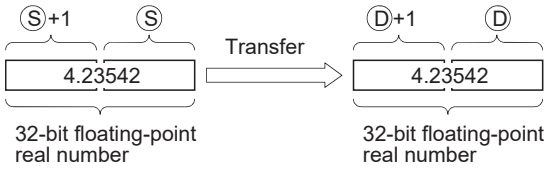
(S): Data to be transferred or number of the device to which the data to be transferred is stored (real number)
(D): The number of the device to which the transferred data will be stored (real number)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant E	Others
	Bit	Word		Bit	Word				
(S)	—	○		—	○		○*1	○	—
(D)	—	○		—	○		○*1	—	—

*1 Applicable for the Universal model QCPU, LCPU.

Processing details

• Transfers 32-bit floating decimal point type real number data being stored at the device designated by (S) to a device designated by (D).



• When an input value is set using a programming tool, a rounding error may occur. For precautions, refer to Page 90 Precautions.

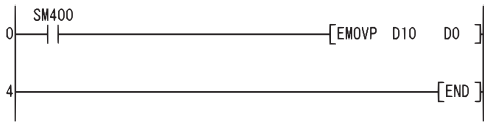
Operation error

• There is no operation error in the EMOV(P) instruction.

Program example

• The following program stores the real numbers at D10 and D11 at D0 and D1.

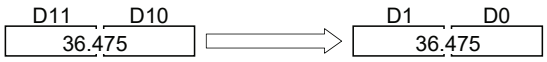
[Ladder Mode]



[List Mode]

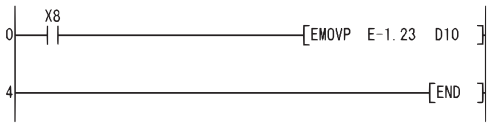
Step	Instruction	Device	
0	LD	SM400	
1	EMOVP	D10	D0
4	END		

[Operation]



• The following program stores the real number -1.23 at D10 and D11 when X8 is ON.

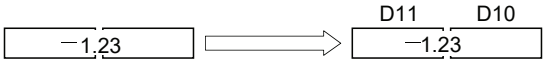
[Ladder Mode]



[List Mode]

Step	Instruction	Device	
0	LD	X8	
1	EMOVP	E-1.23	D10
4	END		

[Operation]



Floating-point data transfer (double precision)

EDMOV(P)

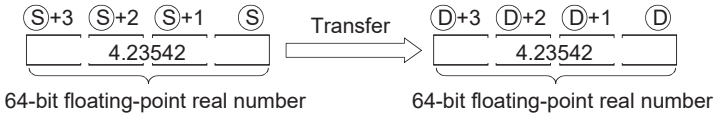
Basic | High performance | Process | Redundant | Universal | LCPU

(S): Data to be transferred or number of the device to which the data to be transferred is stored (real number)
(D): The number of the device to which the transferred data will be stored (real number)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant E	Others
	Bit	Word		Bit	Word				
(S)	—	○		—				○	—
(D)	—	○		—				—	—

Processing details

- Transfers 64-bit floating decimal point type real number data being stored at the device designated by (S) to a device designated by (D).



64-bit floating-point real number　　64-bit floating-point real number

- When an input value is set using a programming tool, a rounding error may occur. For precautions, refer to Page 90 Precautions.

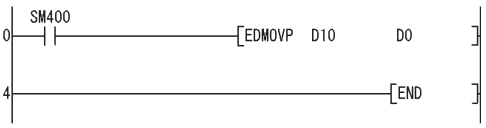
Operation error

- There is no operation error in the EDMOV(P) instruction.

Program example

- The following program stores the 64-bit floating decimal point type real number at D10 to D13 at D0 to D3.

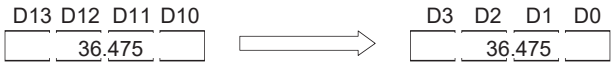
[Ladder Mode]



[List Mode]

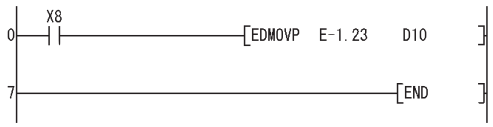
Step	Instruction	Device
0	LD	SM400
1	EDMOVP	D10 D0
4	END	

[Operation]



- The following program stores the real number -1.23 at D10 to D13 when X8 is ON.

[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	X8
1	EDMOVP	E-1.23 D10
7	END	

[Operation]

Character string transfer

$MOV(P)

Basic | High performance | Process | Redundant | Universal | LCPU

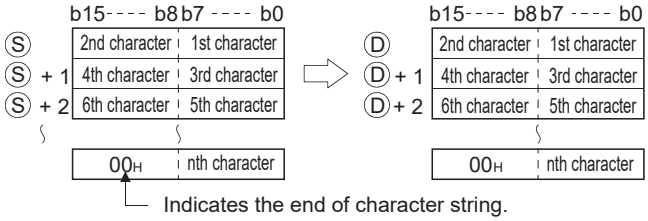
(S): Character string to be transferred (maximum string length: 32 characters) or head number of the devices where the character string to be transferred is stored (character string)

(D): Head number of the devices where the transferred character string will be stored (character string)

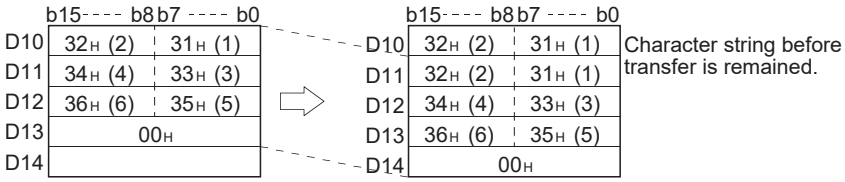
Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant $	Others
	Bit	Word		Bit	Word				
(S)	—	○		—				○	—
(D)	—	○		—				—	—

Processing details

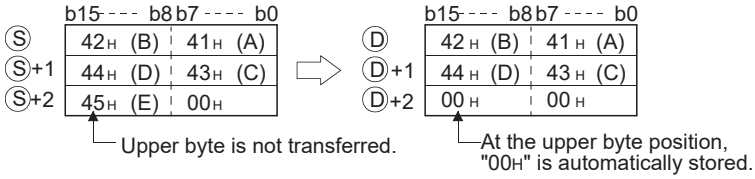
- Transfers the character string data designated by (S) to the area starting from the device designated by (D).
- The character string data enclosed in " (double quotes) specified by (S) or the string from the device number to the number for the device storing the NULL code "00H" is transferred all at once.



- Processing will be performed without error even in cases where the range for the devices storing the character data to be transferred ((S) to (S)+n) overlaps with the range of the devices which will store the character string data after it has been transferred ((D) to (D)+n). The following occurs when the character string data that had been stored from D10 to D13 is transferred to D11 to D14:



- If the NULL code "00H" is stored at lower bytes of (S)+n, the NULL code "00H" will be stored at both the upper bytes and lower bytes of (D)+n.



- For the operation to be performed when the character string data enclosed in double quotes is specified by (S), refer to Page 92 Using character string data.

Operation error

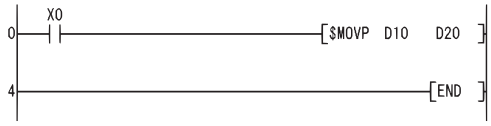
• In any of the following cases, an operation error occurs, the error flag (SM0) turns on, and an error code is stored in SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4101	There is no NULL code "00H" within the range of the corresponding devices starting from the device number specified by (S). The entire character string cannot be stored in the points between the device number specified by (D) and the last device number of the corresponding device. The character string of (S) exceeds 16383 characters.	○	○	○	○	○	○

Program example

• The character string data stored in D10 to D12 is transferred to D20 to D22 when X0 goes ON.

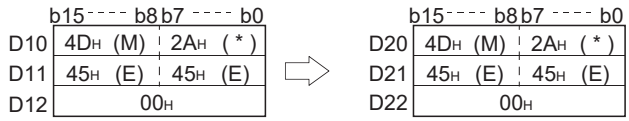
[Ladder Mode]



[List Mode]

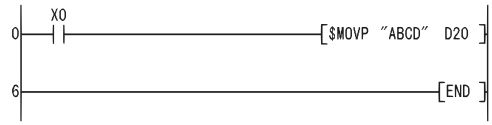
Step	Instruction	Device	
0	LD	X0	
1	$MOVP	D10	D20
4	END		

[Operation]



• When X0 is turned ON, the character string "ABCD" is transferred to D20 and D22.

[Ladder Mode]



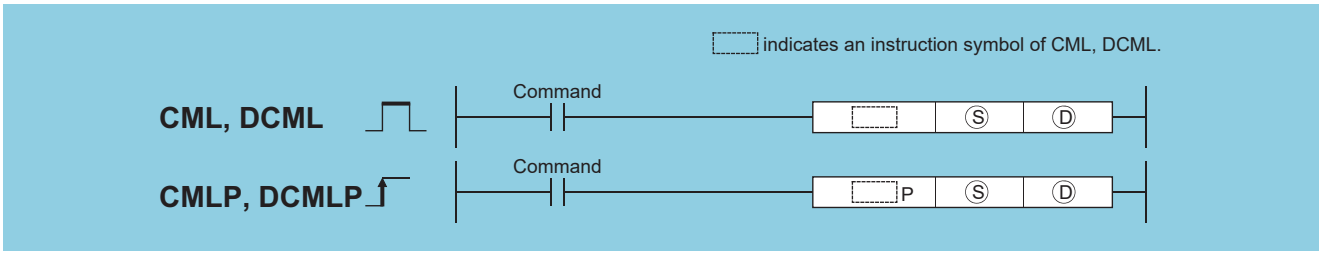
[List Mode]

Step	Instruction	Device	
0	LD	X0	
1	$MOVP	"ABCD"	D20
6	END		

16-bit data negation transfer, 32-bit data negation transfer

CML(P), DCML(P)

Basic | High performance | Process | Redundant | Universal | LCPU



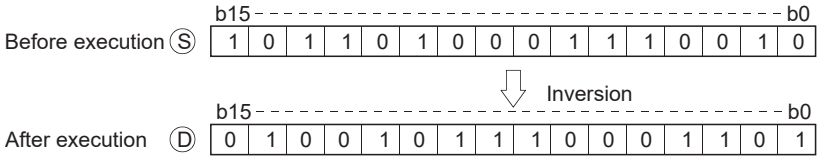
(S): Data to be reversed or the number of the device where data to be reversed is stored (BIN 16/32 bits)
(D): Number of the device where the reversing result will be stored (BIN 16/32 bits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(S)	○							○	—
(D)	○							—	—

Processing details

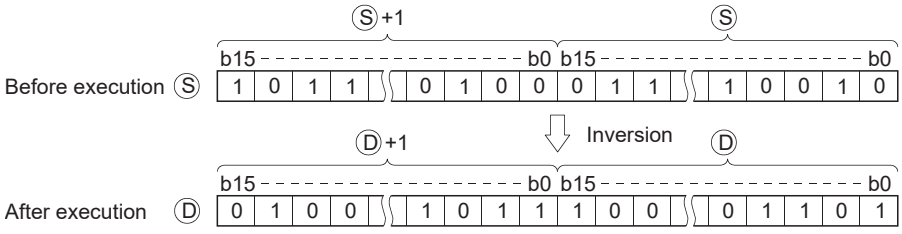
■CML

- Inverts 16-bit data designated by (S) bit by bit, and transfers the result to the device designated by (D).



■DCML

- Inverts 32-bit data designated by (S) bit by bit, and transfers the result to the device designated by (D).



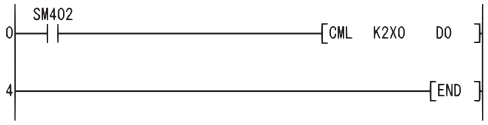
Operation error

- There is no operation error in the CML(P) or DCML(P) instruction.

Program example

• The following program inverts the data from X0 to X7, and transfers result to D0.

[Ladder Mode]

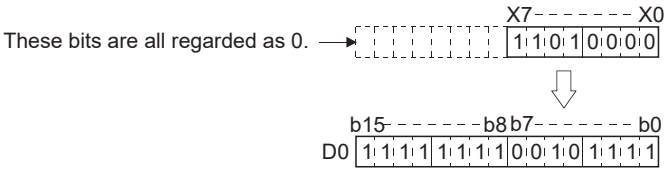


[List Mode]

Step	Instruction	Device	
0	LD	SM402	
1	CML	K2X0	D0
4	END		

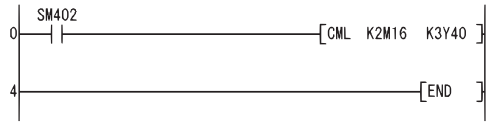
[Operation]

If "Number of bits of Ⓢ < Number of bits of Ⓓ"



• The following program inverts the data at M16 to M23, and transfers the result to Y40 to Y47.

[Ladder Mode]

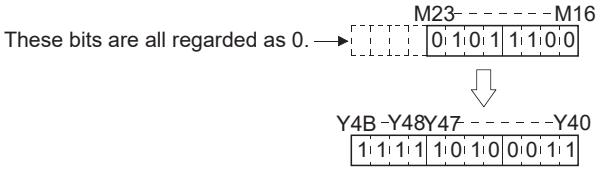


[List Mode]

Step	Instruction	Device	
0	LD	SM402	
1	CML	K2M16	K3Y40
4	END		

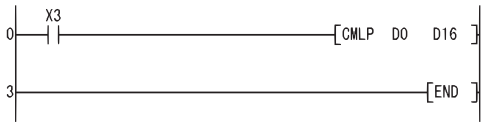
[Operation]

If "Number of bits of Ⓢ < Number of bits of Ⓓ"



• The following program inverts the data at D0 when X3 is ON, and stores the result at D16.

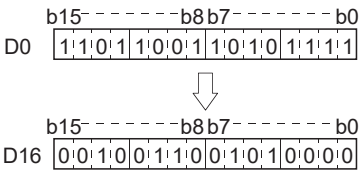
[Ladder Mode]



[List Mode]

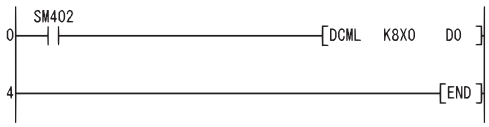
Step	Instruction	Device	
0	LD	X3	
1	CMLP	D0	D16
3	END		

[Operation]



• The following program inverts the data at X0 to X1F, and transfers results to D0 and D1.

[Ladder Mode]

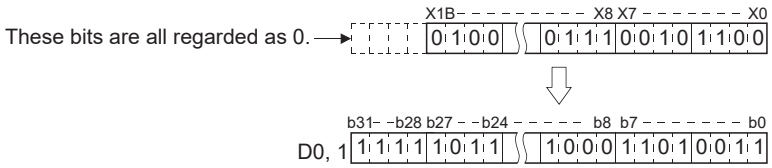


[List Mode]

Step	Instruction	Device
0	LD	SM402
1	DCML	K8X0 D0
4	END	

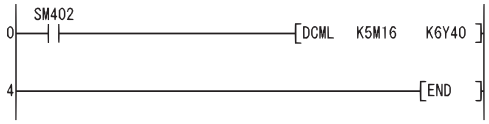
[Operation]

If "Number of bits of Ⓢ < Number of bits of Ⓓ"



• The following program inverts the data at M16 to M35, and transfers it to Y40 to Y63.

[Ladder Mode]

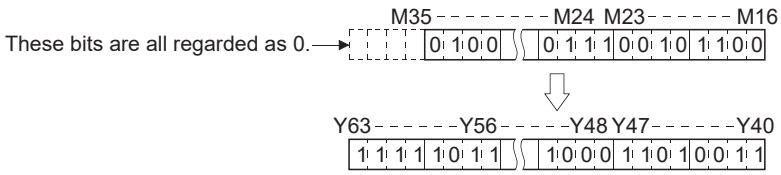


[List Mode]

Step	Instruction	Device
0	LD	SM402
1	DCML	K5M16 K6Y40
4	END	

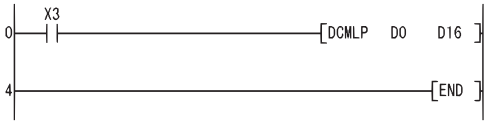
[Operation]

If "Number of bits of Ⓢ < Number of bits of Ⓓ"



• Inverts the data at D0 and D1 when X3 is ON, and stores the result at D16 and D17.

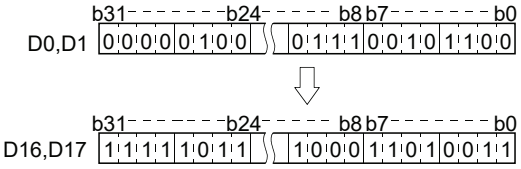
[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	X3
1	DCMLP	D0 D16
4	END	

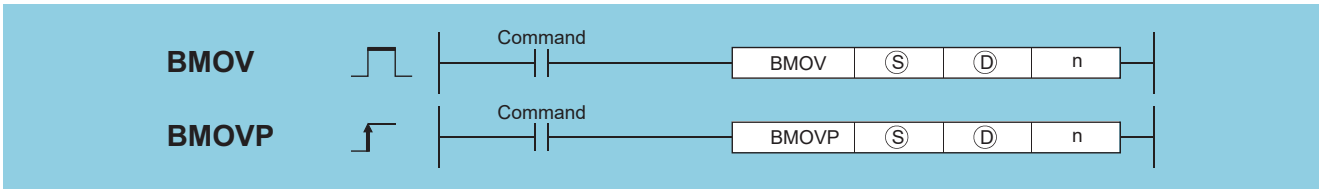
[Operation]



Block 16-bit data transfer

BMOV(P)

Basic | High performance | Process | Redundant | Universal | LCPU



(S): Head number of the devices where the data to be transferred is stored (BIN 16 bits)
(D): Head number of the devices of transfer destination (BIN 16 bits)
n: Number of data to be transferred (BIN 16 bits)

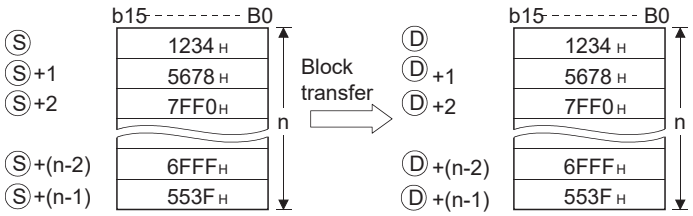
Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(S)	○						—		—
(D)	○						—		—
n	○						○		—

Point

When BL, S, TR, BL\S, or BL\TR is used, refer to the SFC control instructions in the MELSEC-Q/L/QnA Programming Manual (SFC).

Processing details

- Transfers in batch 16-bit data of n points from the device designated by (S) to location n points from the device designated by (D).



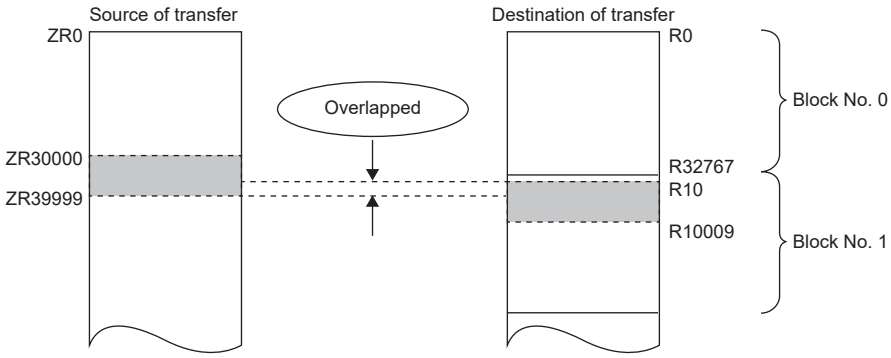
- Transfers can be accomplished even in cases where there is an overlap between the source and destination device. In the case of transmission to the smaller device number, transmission is from (S); for transmission to the larger device number, transmission is from (S)+(n-1). However, as shown in the example below, when transferring data from R to ZR, or from ZR to R, the range to be transferred (source) and the range of destination must not overlap. (QnUDVCPU and QnUDPVCPU are excluded.) Transfer from R to R, or from ZR to ZR can be performed without any problem.
  - ZR transfer range ((specified head No. of ZR) to (specified head No. of ZR + the number of transfers -1))
  - R transfer range ((specified head No. of R + file register block No. × 32768) to (specified head No. of R + file register block No. × 32768 + the number of transfers -1))

Ex.

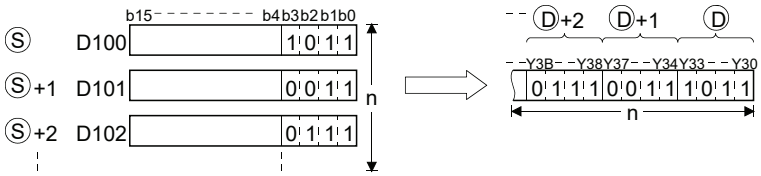
Transfer ranges of ZR and R overlap when transferring 10000 blocks of data from ZR30000 (source) to R10 (block No.1 of the destination).

- ZR transfer range → (30000) to (30000+10000-1) → (30000) to (39999)
- R transfer range → (10+(1×32768)) to (10+(1×32768)+10000-1) → (32778) to (42777)

Therefore, the range 32778 to 39999 overlaps and the data is not correctly transferred.



- When (S) is a word device and (D) is a bit device, the object for the word device will be the number of bits designated by the bit device digit designation. If K1Y30 has been designated by (D), the lower four bits of the word device designated by (S) will become the object.



- If bit device has been designated for (S) and (D), then (S) and (D) should always have the same number of digits.
- When using a link direct device and an intelligent function module device for (S) and (D), only either of (S) or (D) can be used.
- Selection whether to check a device range

Whether to check a device range during execution of the BMOV instruction can be set with SM237 (Device range check inhibit flag). (Only when the conditions of the subset processing are established)
While SM237 is ON, whether (S) to (S) + (n) -1 and (D) to (D) + (n) - 1 are within the device range or not are not checked.

Precautions

While SM237 is on, do not make the following access.

- The indexing target exceeds the device range.
- The value obtained from "(D) to (D) + (n) - 1" is over the boundaries of the device ranges.*1
- Accessing the file register with file register not set.
- Accessing the area where the multiple CPU high speed transmission area device is not available (only for the QCPU).

*1 Refer to the DFMOV instruction.

Point

SM237 is available only for the following CPU modules.

- The Universal model QCPU whose serial number (first five digits) is "10012" or later,
- LCPU

Operation error

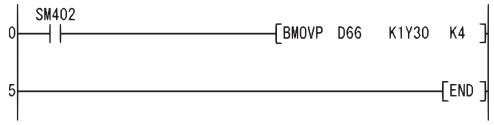
• In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4101	The points specified in n exceed those of the corresponding device specified in (S) or (D). The link direct device and the unit access device are specified in both (S) and (D).	○	○	○	○	○	○

Program example

• The following program outputs the lower 4 bits of data at D66 to D69 to Y30 to Y3F in 4-point units.

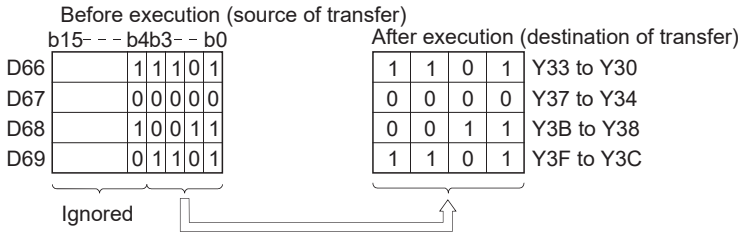
[Ladder Mode]



[List Mode]

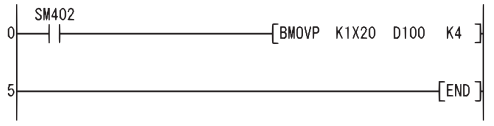
Step	Instruction	Device		
0	LD	SM402		
1	BMOVP	D66	K1Y30	K4
5	END			

[Operation]



• The following program outputs the data at X20 to X2F to D100 to D103 in 4-point units.

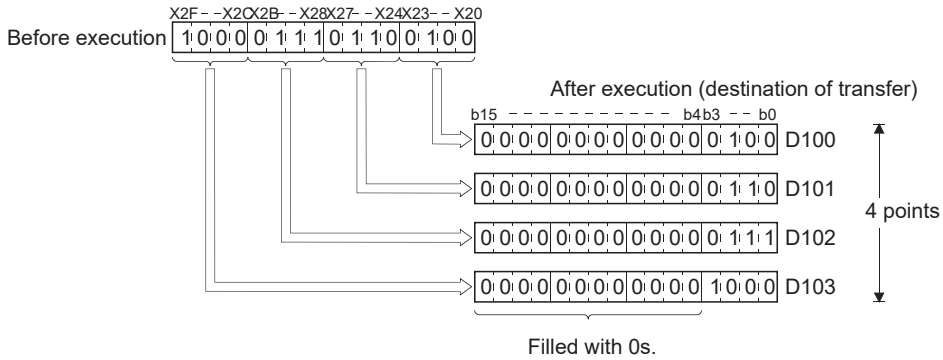
[Ladder Mode]



[List Mode]

Step	Instruction	Device		
0	LD	SM402		
1	BMOVP	K1X20	D100	K4
5	END			

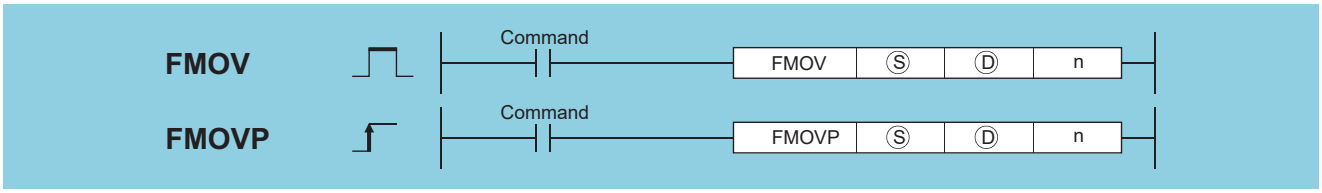
[Operation]



Identical 16-bit data block transfer

FMOV(P)

Basic | High performance | Process | Redundant | Universal | LCPU

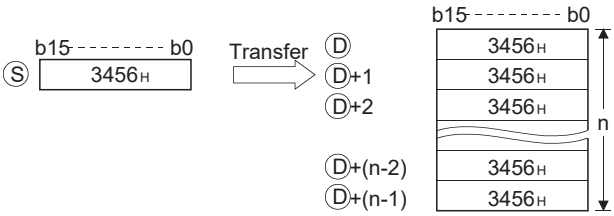


(S): Data to be transferred or the head number of the devices where the data to be transferred is stored (BIN 16 bits)
(D): Head number of the devices of transfer destination (BIN 16 bits)
n: Number of data to be transferred (BIN 16 bits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(S)	○						○		—
(D)	○						—		—
n	○						○		—

Processing details

- Transfers 16-bit data at the device designated by (S) to n points of devices starting from the one designated by (D).



- In cases where (S) designates a word device and (D) a bit device, the number of bits designated by digit designation for the bit device will be the object bits for the word device (S). If K1Y30 has been designated by (D), the lower 4 bits of the word device designated by (S) will become the object.

- If bit device has been designated for (S) and (D), then (S) and (D) should always have the same number of digits.
- Selection whether to check a device range

Whether to check a device range during execution of the FMOV instruction can be set with SM237 (Device range check inhibit flag). (Only when the conditions of the subset processing are established)
While SM237 is ON, whether (D) to (D) + (n) - 1 is within the device range or not is not checked.
For details of SM237, refer to the User's Manual (Hardware Design, Maintenance and Inspection) for the CPU module used.

Precautions

While SM237 is on, do not make the following access.

- The indexing target exceeds the device range.
- The value obtained from "(D) to (D) + (n) - 1" is over the boundaries of the device ranges.*1
- Accessing the file register with file register not set.
- Accessing the area where the multiple CPU high speed transmission area device is not available (only for the QCPU).

*1 Refer to the DFMOV instruction.

Point

SM237 is available only for the following CPU modules.

- The Universal model QCPU whose serial number (first five digits) is "10012" or later,
- LCPU

Operation error

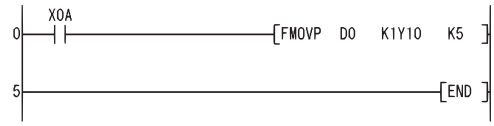
- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4101	The points specified in n exceed those of the corresponding device specified in (S) or (D).	○	○	○	○	○	○

Program example

- The following program outputs the lower 4 bits of D0 when XA goes ON to Y10 to Y23 in 4-bit units.

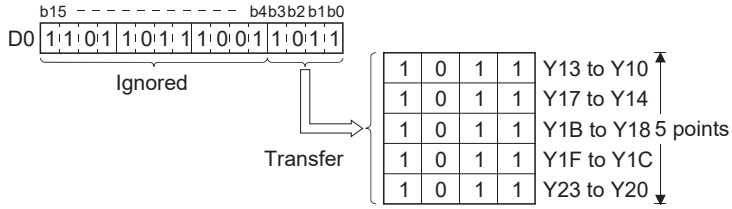
[Ladder Mode]



[List Mode]

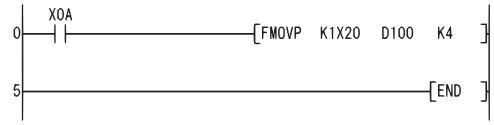
Step	Instruction	Device		
0	LD	XOA		
1	FMOVP	D0	K1Y10	K5
5	END			

[Operation]



- The following program outputs the data at X20 through X23 to D100 through D103 when XA goes ON.

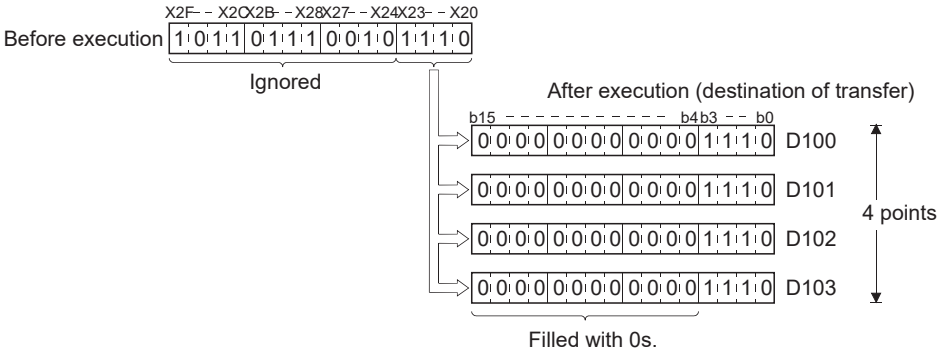
[Ladder Mode]



[List Mode]

Step	Instruction	Device		
0	LD	XOA		
1	FMOVP	K1X20	D100	K4
5	END			

[Operation]

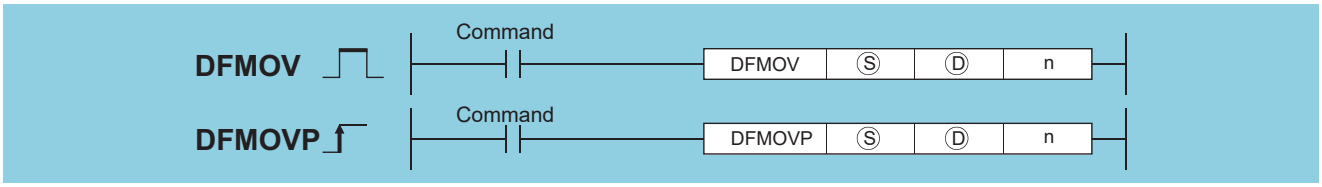


Identical 32-bit data block transfer

DFMOV(P)

Basic | High performance | Process | Redundant | Ver. Universal | LCPU

- QnU(D)(H)CPU, QnUDE(H)CPU: the serial number (first five digits) is "10102" or later
- Q00UJCPU, Q00UCPU, Q01UCPU, QnUDVCPU, QnUDPVCPU: Supported

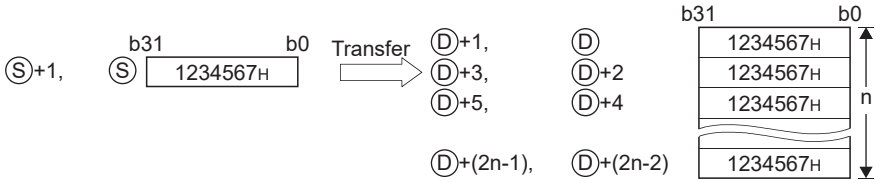


(S): Data to be transferred or head number of the devices where the data to be transferred are stored (BIN 32 bits)
(D): Head number of the devices of transfer destination (BIN 32 bits)
n: Number of data to be transferred (BIN 16 bits)

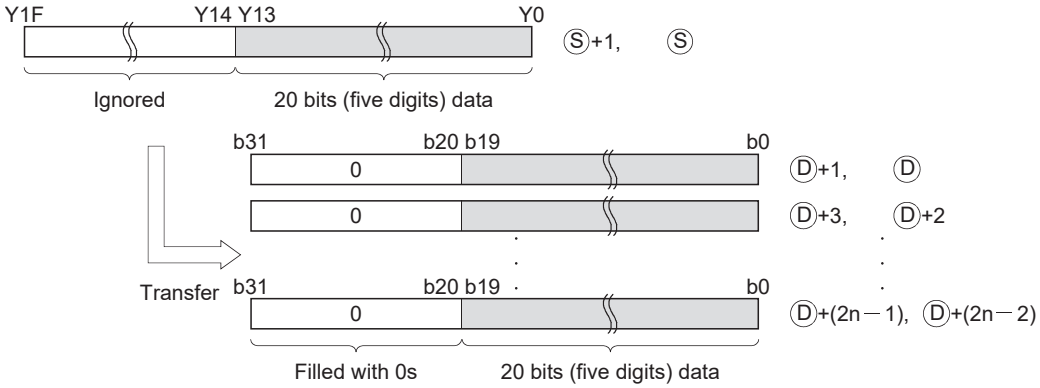
Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(S)	○						○		—
(D)	○						—		—
n	○						○		—

Processing details

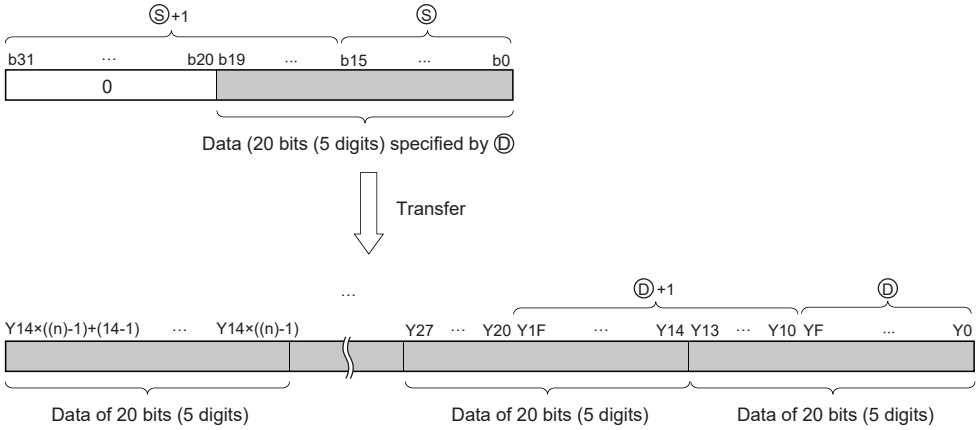
- This instruction transfers 32-bit data of the device specified by (S) to the n-point devices starting from the device specified by (D).



- If (S) specifies data of a device with digit specification, the amount of data to be transferred will be the amount of the data specified digit. If K5Y0 is specified by (S), the lower 20 bits (five digits) of the word device specified by (S) will be the object.



- If (D) specifies data of a device with digit specification, the amount of data stored in the device specified by (D) will be transferred. If K5Y0 is specified by (D), the lower 20 bits of the word device specified by (S) will be the object. If both (S) and (D) specify data of a device with digit specification, the amount of data specified by (D) will be transferred regardless of the number of digits.



- If the value specified by n is 0, the instruction will be not processed.
- Whether to check a device range during the execution of the DFMOV instruction can be set with SM237 (Device range check inhibit flag). (Only when the conditions of the subset processing are established)

Operation error

- In any of the following cases, an operation error occurs, the error flag (SM0) turns on, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4100	The value specified for n is negative.	—	—	—	—	○	○
4101	Data points to be transferred (n) exceed the points of the device specified in (D).	—	—	—	—	○	○

Program example

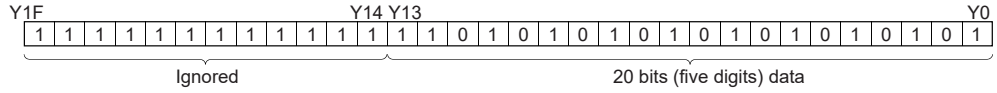
- The following program stores the value data stored at Y0 to Y13(20 bits) into D10 to D17, when M0 is turned on.

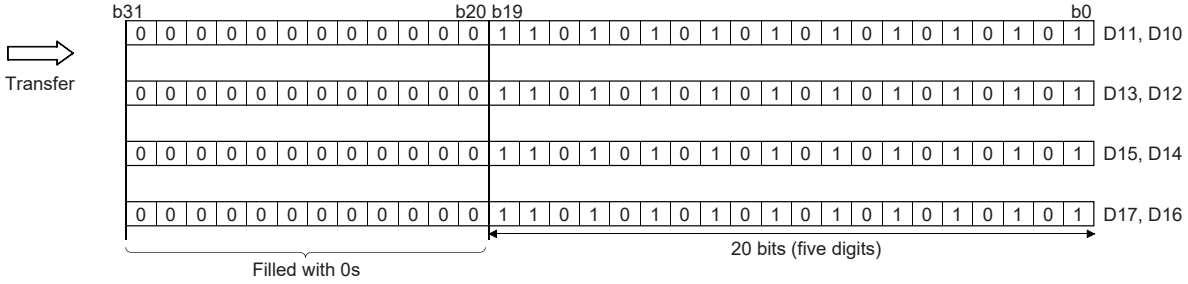
[Ladder Mode]

[List Mode]

Step	Instruction	Device
0	LD	M0
1	DFMOV	K5Y0 D10 K4
5	END	

[Operation]

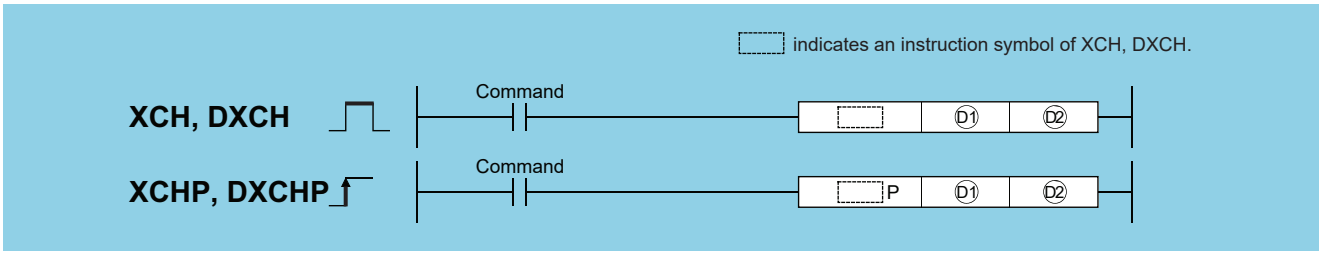




16-bit data exchanges, 32-bit data exchanges

XCH(P), DXCH(P)

Basic | High performance | Process | Redundant | Universal | LCPU



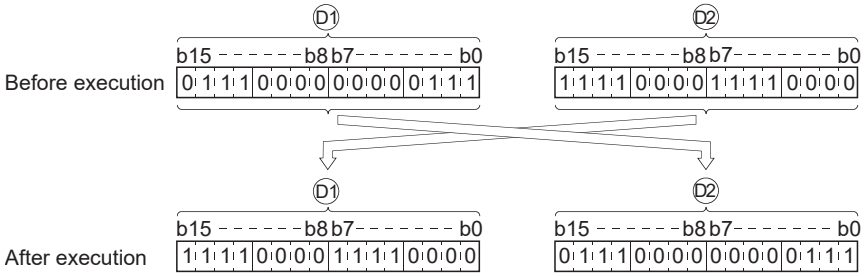
(D1), (D2): Head number of the devices where the data to be exchanged is stored (BIN 16/32 bits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant	Others
	Bit	Word		Bit	Word				
(D1)	○							—	
(D2)	○							—	

Processing details

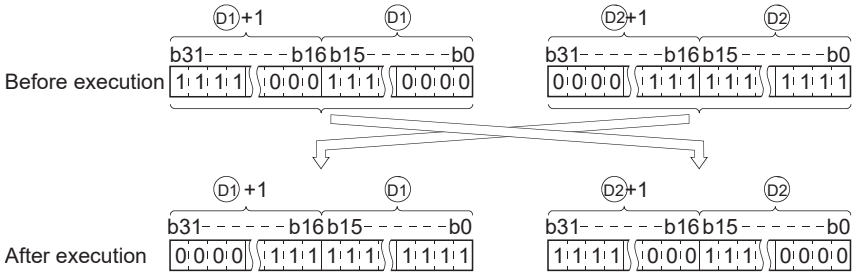
■XCH

- Conducts 16-bit data exchange between (D1) and (D2).



■DXCH

- Conducts 32-bit data exchange between (D1)+1, (D1) and (D2)+1, (D2).



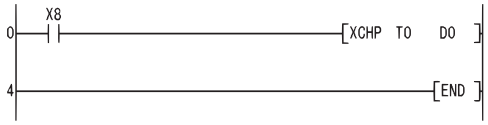
Operation error

- There is no error in the XCH(P) or DXCH(P) instruction.

Program example

• The following program exchanges the present value of T0 with the contents of D0 when X8 goes ON.

[Ladder Mode]

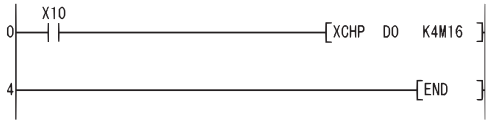


[List Mode]

Step	Instruction	Device	
0	LD	X8	
1	XCHP	T0	D0
4	END		

• The following program exchanges the contents of D0 with the data from M16 to M31 when X10 goes ON.

[Ladder Mode]

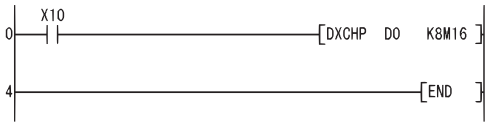


[List Mode]

Step	Instruction	Device	
0	LD	X10	
1	XCHP	D0	K4M16
4	END		

• The following program exchanges the contents of D0 and D1 with the data at M16 to M47 when X10 goes ON.

[Ladder Mode]

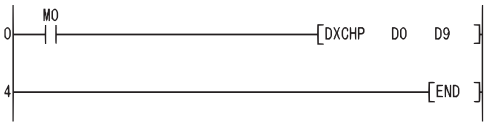


[List Mode]

Step	Instruction	Device	
0	LD	X10	
1	DXCHP	D0	K8M16
4	END		

• The following program exchanges the contents of D0 and D1 with those of D9 and D10 when M0 goes ON.

[Ladder Mode]



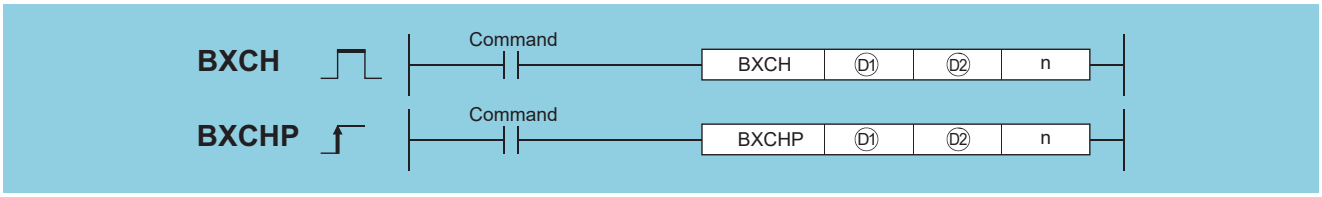
[List Mode]

Step	Instruction	Device	
0	LD	M0	
1	DXCHP	D0	D9
4	END		

Block 16-bit data exchanges

BXCH(P)

Basic | High performance | Process | Redundant | Universal | LCPU

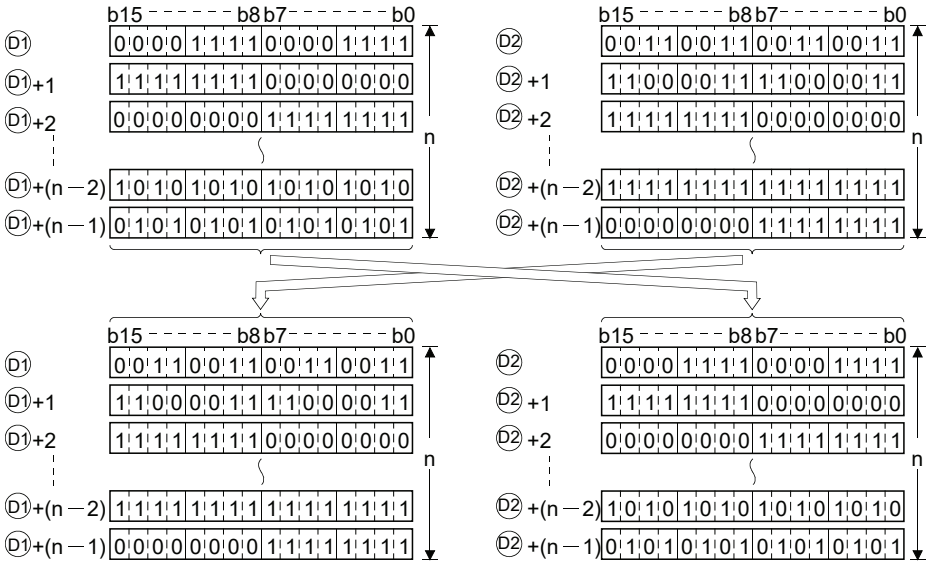


(D1), (D2) : Head number of the devices where the data to be exchanged is stored (BIN 16 bits)
n: Number of exchanges (BIN 16 bits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(D1)	—	○		—					—
(D2)	—	○		—					—
n	○	○		○					—

Processing details

- Exchanges 16-bit data of n points from device designated by (D1) and 16-bit data of n points from device designated by (D2).



Operation error

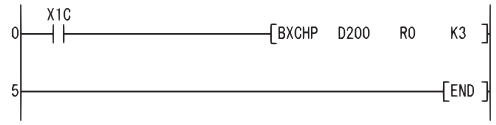
- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4101	The points specified in n exceed those of the corresponding device specified in (D1) or (D2). The (D1) and (D2) devices overlap.	○	○	○	○	○	○

Program example

- The following program exchanges 16-bit data for 3 points from D200 for 16-bit data for 3 points from R0 when X1C goes ON.

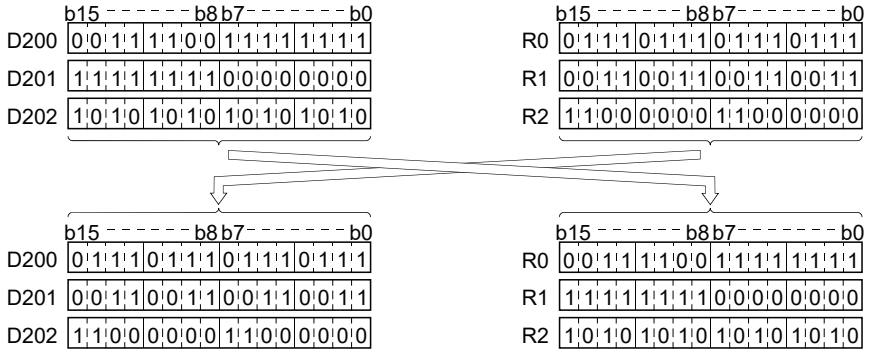
[Ladder Mode]



[List Mode]

Step	Instruction	Device		
0	LD	X1C		
1	BXCHP	D200	R0	K3
5	END			

[Operation]

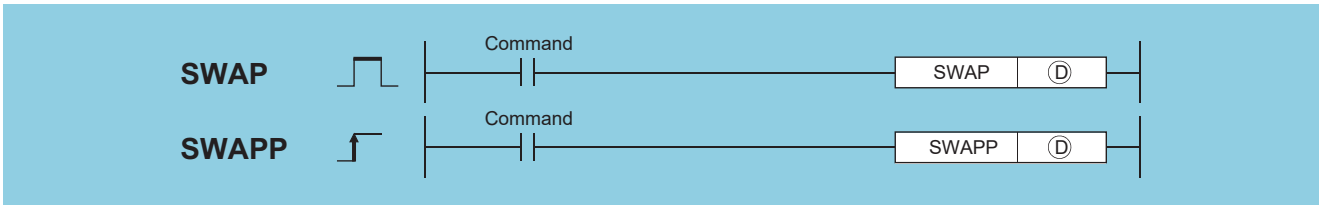


6

Upper and lower byte exchanges

SWAP(P)

Basic | High performance | Process | Redundant | Universal | LCPU

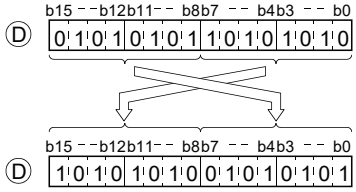


(D): Head number of the devices where the data is stored (BIN 16 bits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant	Others
	Bit	Word		Bit	Word				
(D)	○							—	

Processing details

- Exchanges the higher and lower 8 bits of the device designated by (D).



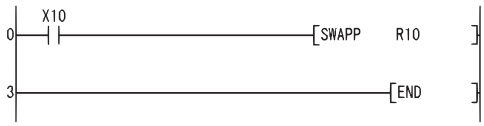
Operation error

- There is no operation error in the SWAP(P) instruction.

Program example

- The following program exchanges the higher 8 bits and lower 8 bits of R10 when X10 goes ON.

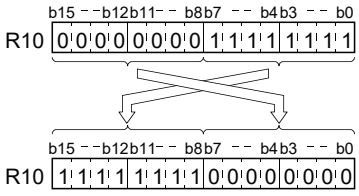
[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	X10
1	SWAPP	R10
3	END	

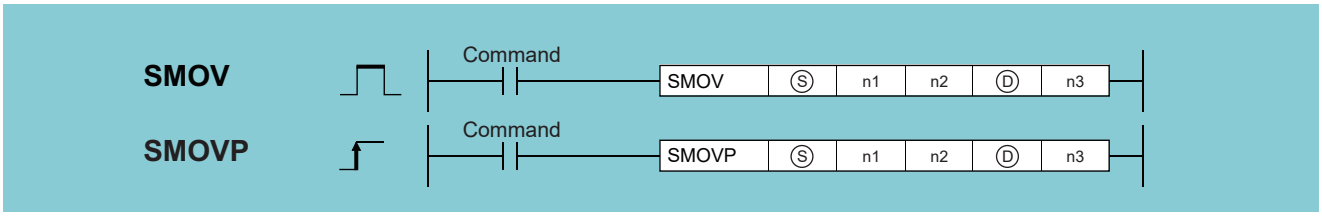
[Operation]



SMOV(P)

Basic (×) | High performance (×) | Process (×) | Redundant (×) | Ver. Universal | Ver. LCPU

- LCPU: The serial number (first five digits) is "16042" or later.
- QnUDVCPU, QnUDPVCPU: The serial number (first five digits) is "16043" or later.

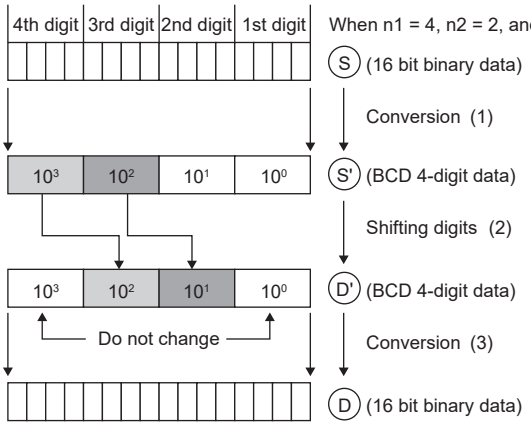


(S): Start number of the devices where the data of the specified digit is stored (Device name)
n1: Start position where data to be moved is stored (BIN 16 bits)
n2: Number of digit to be shifted (BIN 16-bit)
(D): Head number of device where data, which was shifted by digit units, is stored (device name)
n3: Position of head line of the shift destination (BIN 16-bit)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(S)	—	○		—	○			—	—
n1	—	○		—	○			○	—
n2	—	○		—	○			○	—
(D)	—	○		—	○			—	—
n3	—	○		—	○			○	—

Processing details

- When SM719 (SMOV Instruction BCD conversion inhibit flag) is off: The values of (S) and (D) are converted into a four-digit BCD (0000 to 9999), and the values of the n1-th digit to the lower n2 digits are transferred with the n3-th digit of (D) being placed at the start, converted into BIN, and then stored in (D).

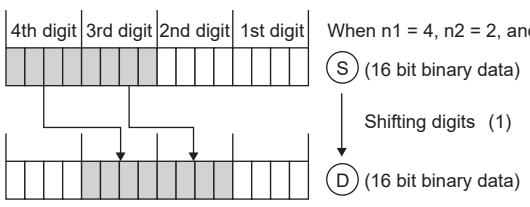


(1) Convert Ⓢ from BIN data to BCD data.

(2) Transfer n2 digits of data from the n1th digit Ⓢ' to the n3th and n2th digits to compose Ⓓ' data. 103 and 100 of Ⓓ' data remain the same.

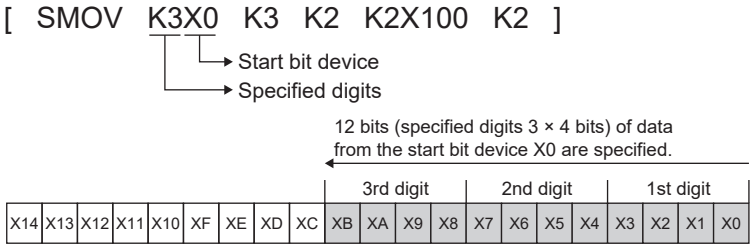
(3) Covert the composed data from BCD to BIN, and store it in Ⓓ.

- When SM719 is on: BIN → BCD conversion is not conducted and data is shifted in units of 4 bits.



(1) Transfer (n2 × 4 bits) of data from n1th digit Ⓢ to the n3th and n2th digits to compose Ⓓ data. Bit 16 to 13 and 4 to 1 of Ⓓ data remain the same.

• When the digit specification by bit is conducted for the transfer source device, the digit corresponding to the specified digit number × 4 is specified from the bit device at head. The operation examples are shown below.



Operation error

• In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4100	The BCD data of the (S) and (D) are those other than 0 to 9999.	—	—	—	—	○	○
4101	The values of device designated by the instructions exceed the setting range.	—	—	—	—	○	○

Program example

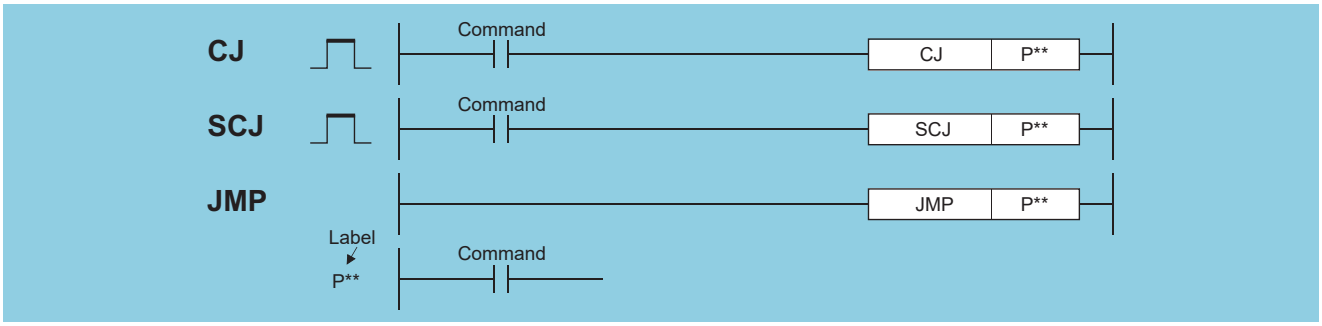
• Program that transfers the one digit (BCD) of D10 into the 3rd digit (BCD) of D20 and then converts them into the BIN.

6.5 Program Branch Instructions

Pointer branch

CJ, SCJ, JMP

Basic | High performance | Process | Redundant | Universal | LCPU



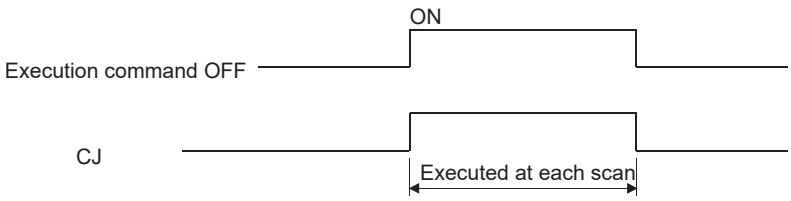
P**: Pointer number of jump destination (Device name)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant	Others
	Bit	Word		Bit	Word				P
P	—								○

Processing details

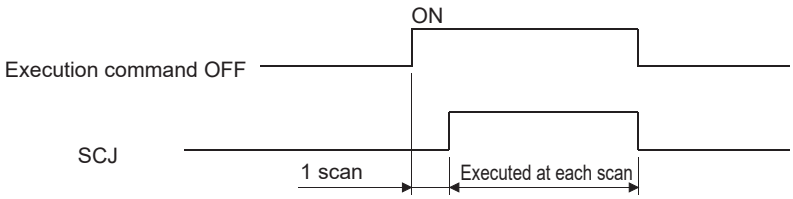
■CJ

- Executes the program specified by the pointer number within the same program file, when the execution command is ON.
- When the execution command is OFF, the program at the next step is executed.



■SCJ

- Executes the program specified by the pointer number within the same program file starting with the scan immediately after OFF→ON of the execution command.
- When the execution command is OFF or turned ON→OFF, the program at the next step is executed.



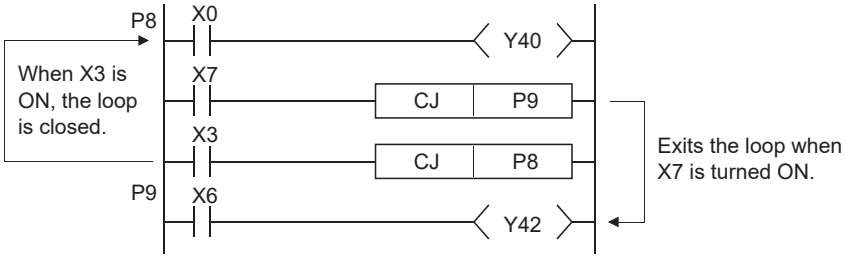
■JMP

- Unconditionally executes program of designated pointer number within the same program file.

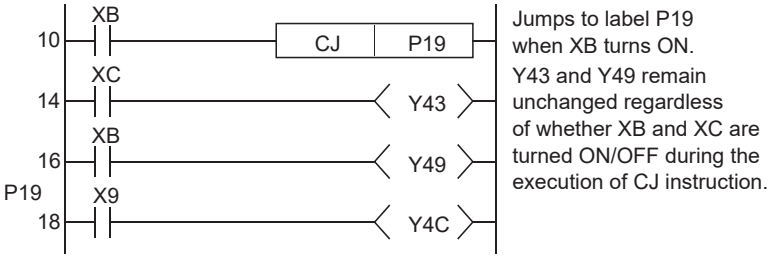
Point

Note the following points when using the jump instruction.

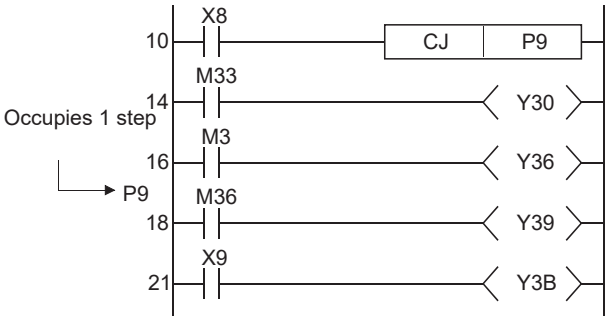
- After the timer coil has gone ON, accurate measurements cannot be made if there is an attempt to jump the timer of a coil that has been turned ON using the CJ, SCJ or JMP instructions.
- Scan time is shortened if the CJ, SCJ or JMP instruction is used to force a jump to the OUT instruction.
- Scan time is shortened if the CJ, SCJ or JMP instruction is used to force a jump to the rear.
- The CJ, SCJ, and JMP instructions can be used to jump to a step prior to the step currently being executed. However, it is necessary to consider methods to get out of the loop so that the watchdog timer does not time out in the process.



- The device to which a jump has been made with the CJ, SCJ or JMP does not change.



- The label (P*) occupies step 1.



- The jump instructions can be used only for pointer numbers within the same program file.

Operation error

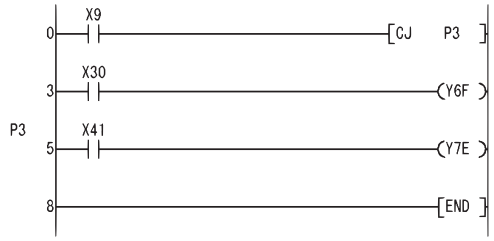
- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4210	There is no specified pointer number before the END instruction. A pointer number which is not in use as a label in the same program has been specified. A common pointer in another program is specified.	○	○	○	○	○	○

Program example

- The following program jumps to P3 when X9 goes ON.

[Ladder Mode]

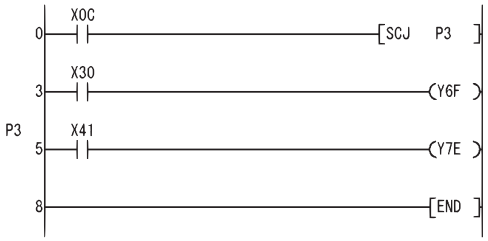


[List Mode]

Step	Instruction	Device
0	LD	X9
1	CJ	P3
3	LD	X30
4	OUT	Y6F
5	P3	
6	LD	X41
7	OUT	Y7E
8	END	

- The following program jumps to P3 from the next scan after XC goes ON.

[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	XOC
1	SCJ	P3
3	LD	X30
4	OUT	Y6F
5	P3	
6	LD	X41
7	OUT	Y7E
8	END	

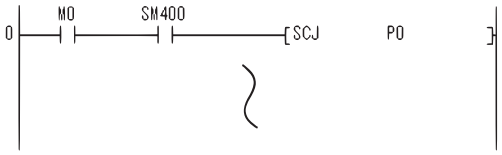
Precautions

- When using the Universal model QCPU and LCPU with the SCJ instruction, inserting "AND SM400" (or the NOP instruction) in immediately before the SCJ instruction is required.

Ex.

Program example 1

[Ladder Mode]



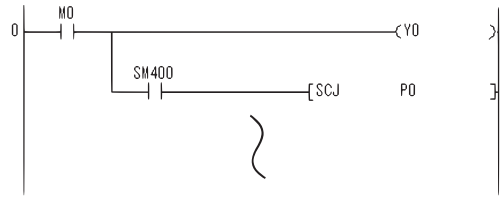
[List Mode]

Step	Instruction	Device
0	LD	M0
1	AND	SM400
2	SCJ	P0

Ex.

Program example 2

[Ladder Mode]



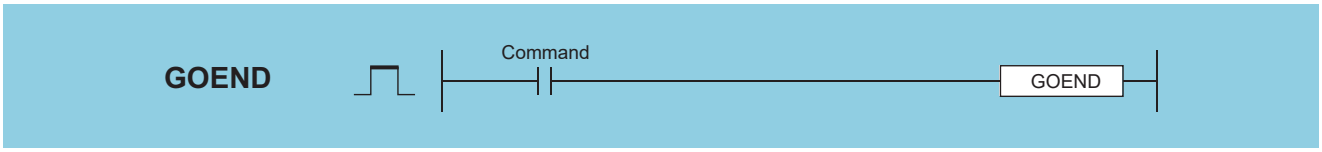
[List Mode]

Step	Instruction	Device
0	LD	M0
1	OUT	Y0
2	AND	SM400
3	SCJ	P0

Jump to END

GOEND

Basic | High performance | Process | Redundant | Universal | LCPU



Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant	Others
	Bit	Word		Bit	Word				
—	—								

Processing details

- Jumps to the FEND or END instruction in the same program file.

Operation error

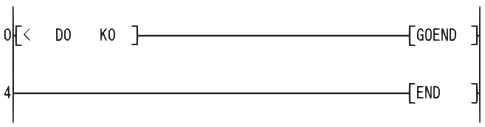
- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4200	After the FOR instruction was executed, the GOEND instruction was executed prior to the NEXT instruction.	○	○	○	○	○	○
4211	After the CALL, ECALL instruction was executed, the GOEND instruction was executed prior to the the RET instruction.	○	○	○	○	○	○
4221	During an interrupt program, the GOEND instruction was executed prior to the IRET instruction.	○	○	○	○	○	○
4230	The GOEND instruction was executed during the CHKCIR to CHKEND instruction execution.	—	○	○	○	—	—
4231	The GOEND instruction was executed during the IX to IXEND instruction execution.	○	○	○	○	—	—

Program example

- The following program jumps to the END instruction if D0 holds a negative number.

[Ladder Mode]



[List Mode]

Step	Instruction	Device	
0	LD<	D0	K0
3	GOEND		
4	END		

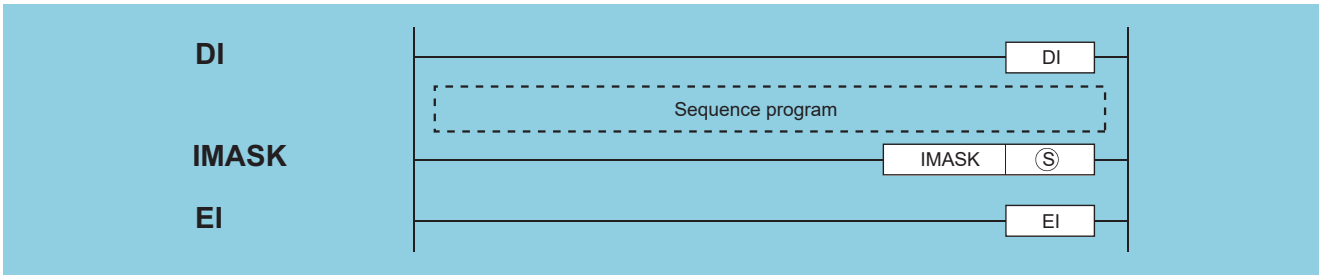
6.6 Program Execution Control Instructions

Interrupt disable, interrupt enable, interrupt program mask

DI, EI, IMASK

Basic | High performance | Process | Redundant | Universal | LCPU

■When the Basic model QCPU is used



(S): Interrupt mask data or head number of the devices where the interrupt mask data is stored (BIN 16 bits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant	Others
	Bit	Word		Bit	Word				
(S)	—	○		—					

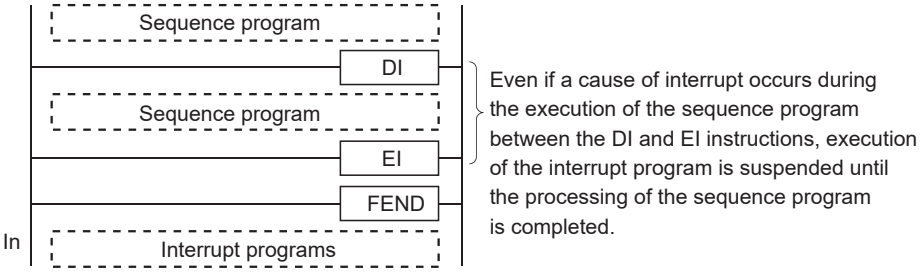
Processing details

■DI

- Disables the execution of an interrupt program until the EI instruction has been executed, even if a start cause for the interrupt program occurs.
- A DI state is entered when power is turned ON or when the CPU module is reset.

■EI

- The EI instruction is used to clear the interrupt disable state resulting from the execution of the DI instruction, and to create a state in which the interrupt program designated by the interrupt pointer number certified by the IMASK instruction can be executed.
- When the IMASK instruction is not executed, I32 to I47 are disabled.



■IMASK

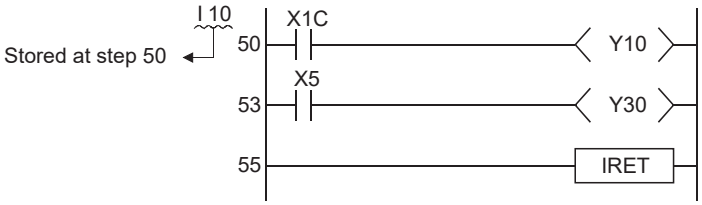
- Enables/disables the execution of the interrupt program marked by the designated interrupt pointer by using the bit pattern of 8 points from the device designated by (S).
  - 1(ON) ··· Interrupt program execution enabled
  - 0(OFF) ··· Interrupt program execution disabled
- The interrupt pointer numbers corresponding to the individual bits are as shown below:

	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
(S)	I15	I14	I13	I12	I11	I10	I9	I8	I7	I6	I5	I4	I3	I2	I1	I0
(S)+1	I31	I30	I29	I28	I27	I26	I25	I24	I23	I22	I21	I20	I19	I18	I17	I16
(S)+2	I47	I46	I45	I44	I43	I42	I41	I40	I39	I38	I37	I36	I35	I34	I33	I32
(S)+3	I63	I62	I61	I60	I59	I58	I57	I56	I55	I54	I53	I52	I51	I50	I49	I48
(S)+4	I79	I78	I77	I76	I75	I74	I73	I72	I71	I70	I69	I68	I67	I66	I65	I64
(S)+5	I95	I94	I93	I92	I91	I90	I89	I88	I87	I86	I85	I84	I83	I82	I81	I80
(S)+6	I111	I110	I109	I108	I107	I106	I105	I104	I103	I102	I101	I100	I99	I98	I97	I96
(S)+7	I127	I126	I125	I124	I123	I122	I121	I120	I119	I118	I117	I116	I115	I114	I113	I112

- When the system is powered on or when the CPU module is reset, the execution of interrupt programs I0 to I31 and I48 to I127 is enabled, and the execution of interrupt programs I32 to I47 is disabled.
- The statuses of devices (S), (S)+1, (S)+2, and (S)+3 to (S)+7 are stored in SD715 to SD717 and SD781 to SD785 (storage area for the IMASK instruction mask pattern).
- Although the special registers are separated as SD715 to SD717 and SD781 to SD785, device numbers should be designated as (S) to (S)+7 successively.

Point

- An interrupt pointer occupies 1 step.



- For the information on interrupt conditions, link direct devices, refer to the QnUCPU User's Manual (Function Explanation, Program Fundamentals) or Qn(H)/QnPH/QnPRHCPU User's Manual (Function Explanation, Program Fundamentals)
- The DI state (interrupt disabled) is active during the execution of an interrupt program. Do not insert the EI instructions in interrupt programs to attempt the execution of multiple interrupts, with interrupt programs running inside interrupt programs.
- If there are the EI and DI instructions within a master control, these instructions will be executed regardless of the execution/non-execution status of the MC instruction.

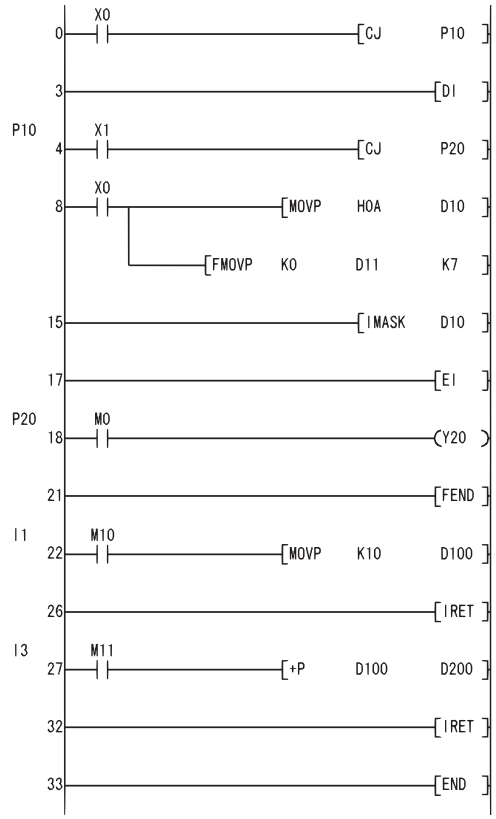
Operation error

- There is no operation error in the DI, EI, or IMASK instruction.

Program example

- The following program is designed to enable the execution of only the interrupt programs having the interrupt pointer numbers I1 and I3 while X0 is ON.

[Ladder Mode]

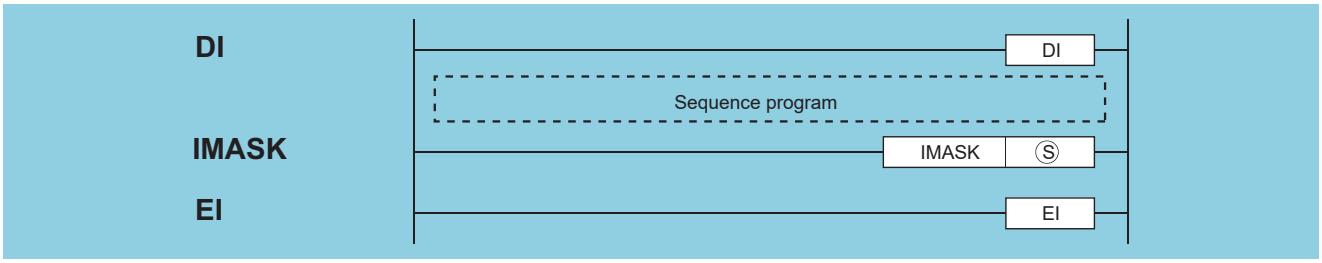


[List Mode]

Step	Instruction	Device		
0	LD	X0		
1	CJ	P10		
3	DI			
4	P10			
5	LD	X1		
6	CJ	P20		
8	LD	X0		
9	MOVP	H0A	D10	
11	FMOVP	K0	D11	K7
15	IMASK	D10		
17	EI			
18	P20			
19	LD	M0		
20	OUT	Y20		
21	FEND			
22	I1			
23	LD	M10		
24	MOVP	K10	D100	
26	IRET			
27	I3			
28	LD	M11		
29	+P	D100	D200	
32	IRET			
33	END			

DI, EI, IMASK

■When the High Performance model QCPU/Process CPU/Redundant CPU/Universal model QCPU or LCPU is used



(S): Head number of the devices where the interrupt mask data is stored (BIN 16 bits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant	Others
	Bit	Word		Bit	Word				
(S)	—	○		—					

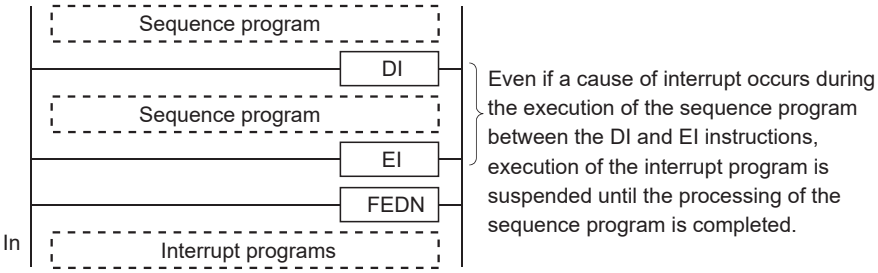
Processing details

■DI

- Disables the execution of an interrupt program until the EI instruction has been executed, even if a start cause for the interrupt program occurs.
- A DI state is entered when power is turned ON or when the CPU module is reset.

■EI

- The EI instruction is used to clear the interrupt disable state resulting from the execution of the DI instruction, and to create a state in which the interrupt program designated by the interrupt pointer number enabled by the IMASK instruction and the fixed cycle execution type program can be executed.
- When the IMASK instruction is not executed, I32 to I47 are disabled.



■IMASK

- Enables/disables the execution of the interrupt program marked by the designated interrupt pointer by using the bit pattern of 16 points from the device designated by (S).
  - 1(ON) ··· Interrupt program execution enabled
  - 0(OFF) ··· Interrupt program execution disabled
- The interrupt pointer numbers corresponding to the individual bits are as shown below:

	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
(S)	I15	I14	I13	I12	I11	I10	I9	I8	I7	I6	I5	I4	I3	I2	I1	I0
(S)+1	I31	I30	I29	I28	I27	I26	I25	I24	I23	I22	I21	I20	I19	I18	I17	I16
(S)+2	I47	I46	I45	I44	I43	I42	I41	I40	I39	I38	I37	I36	I35	I34	I33	I32
(S)+3	I63	I62	I61	I60	I59	I58	I57	I56	I55	I54	I53	I52	I51	I50	I49	I48
(S)+4	I79	I78	I77	I76	I75	I74	I73	I72	I71	I70	I69	I68	I67	I66	I65	I64
(S)+5	I95	I94	I93	I92	I91	I90	I89	I88	I87	I86	I85	I84	I83	I82	I81	I80
(S)+6	I111	I110	I109	I108	I107	I106	I105	I104	I103	I102	I101	I100	I99	I98	I97	I96
(S)+7	I127	I126	I125	I124	I123	I122	I121	I120	I119	I118	I117	I116	I115	I114	I113	I112
(S)+8	I143	I142	I141	I140	I139	I138	I137	I136	I135	I134	I133	I132	I131	I130	I129	I128
(S)+9	I159	I158	I157	I156	I155	I154	I153	I152	I151	I150	I149	I148	I147	I146	I145	I144
(S)+10	I175	I174	I173	I172	I171	I170	I169	I168	I167	I166	I165	I164	I163	I162	I161	I160
(S)+11	I191	I190	I189	I188	I187	I186	I185	I184	I183	I182	I181	I180	I179	I178	I177	I176
(S)+12	I207	I206	I205	I204	I203	I202	I201	I200	I199	I198	I197	I196	I195	I194	I193	I192
(S)+13	I223	I222	I221	I220	I219	I218	I217	I216	I215	I214	I213	I212	I211	I210	I209	I208
(S)+14	I239	I238	I237	I236	I235	I234	I233	I232	I231	I230	I229	I228	I227	I226	I225	I224
(S)+15	I255	I254	I253	I252	I251	I250	I249	I248	I247	I246	I245	I244	I243	I242	I241	I240

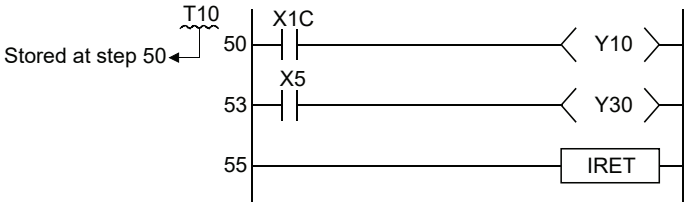
- When the power is turned on or the CPU module is reset, the interrupt programs are as follows.

CPU module	Description
For the High Performance model QCPU, Process CPU, and Redundant CPU	Execution of interrupt programs I0 to I31 and I48 to I255 is enabled, and execution of interrupt programs I32 to I47 is disabled.
Universal model QCPU and LCPU	Execution of interrupt programs I0 to I31 and I45 to I255 is enabled, and execution of interrupt programs I32 to I44 is disabled.

- The status of devices (S), (S)+1, (S)+2, and (S)+3 to (S)+15 are stored in SD715 to SD717 and SD781 to SD793 (storage area for the IMASK instruction mask pattern).
- Although the special registers are separated as SD715 to SD717 and SD781 to SD793, device numbers should be designated as (S) to (S)+15 successively.

Point

- An interrupt pointer occupies 1 step.



- For the information on interrupt conditions, link direct devices, refer to the QnUCPU User's Manual (Function Explanation, Program Fundamentals) or Qn(H)/QnPH/QnPRHCPU User's Manual (Function Explanation, Program Fundamentals)
- The DI state (interrupt disabled) is active during the execution of an interrupt program. Do not insert the EI instructions in interrupt programs to attempt the execution of multiple interrupts, with interrupt programs running inside interrupt programs.
- If there are the EI and DI instructions within a master control, these instructions will be executed regardless of the execution/non-execution status of the MC instruction.

Operation error

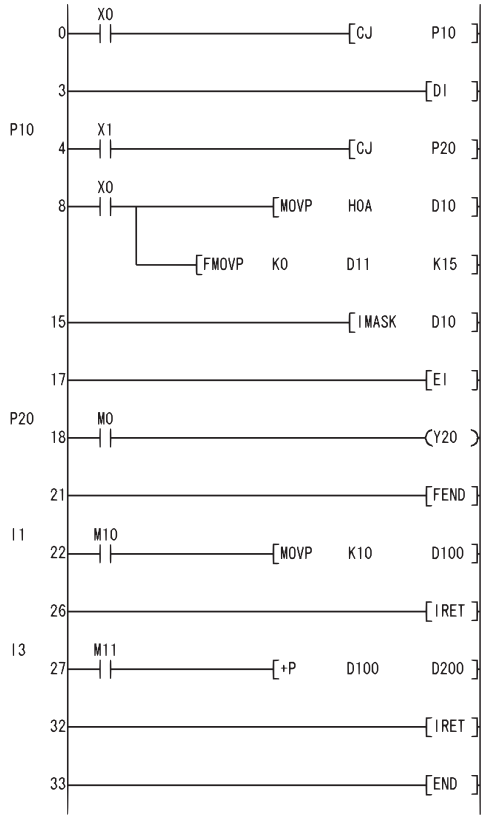
- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4101	The device specified in (S) exceeds the range of the corresponding device.	—	—	—	—	○	○

Program example

- The following program creates an execution enabled state for the interrupt program marked by the interrupt pointer number when X0 is ON.

[Ladder Mode]



[List Mode]

Step	Instruction	Device		
0	LD	X0		
1	CJ	P10		
3	DI			
4	P10			
5	LD	X1		
6	CJ	P20		
8	LD	X0		
9	MOVP	H0A	D10	
11	FMOVP	K0	D11	K15
15	IMASK	D10		
17	EI			
18	P20			
19	LD	M0		
20	OUT	Y20		
21	FEND			
22	I1			
23	LD	M10		
24	MOVP	K10	D100	
26	IRET			
27	I3			
28	LD	M11		
29	+P	D100	D200	
32	IRET			
33	END			

Recovery from interrupt programs

IRET

Basic | High performance | Process | Redundant | Universal | LCPU

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant	Others
	Bit	Word		Bit	Word				
—	—								

Processing details

- Indicates the completion of interrupt program processing.
- Returns to sequence program processing following the execution of the IRET instruction.

Operation error

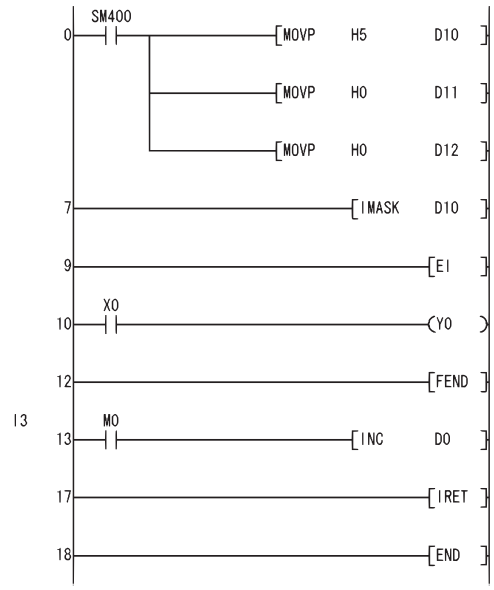
- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4220	There is no pointer corresponding to the interrupt number.	○	○	○	○	○	○
4221	After an interrupt occurred, the END, FEND, GOEND, or STOP instruction was executed prior to the IRET instruction.	○	○	○	○	○	○
4223	The IRET instruction was executed before the interrupt program is executed.	○	○	○	○	○	○
	The IRET instruction was executed during the fixed scan execution type program.	—	—	—	—	○	○

Program example

- The following program adds 1 to D0 if M0 is ON when the number 3 interrupt is generated.

[Ladder Mode]



[List Mode]

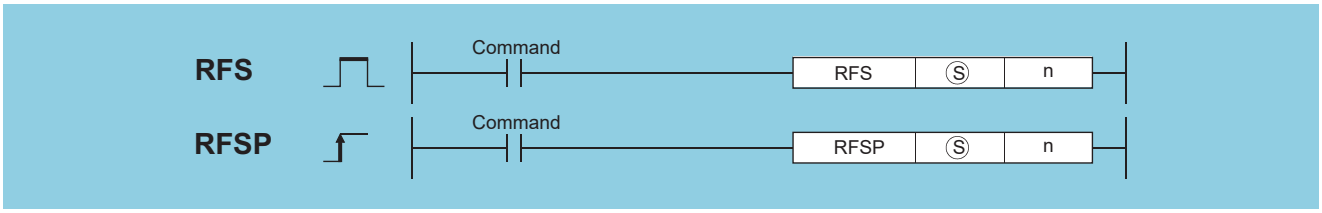
Step	Instruction	Device	
0	LD	SM400	
1	MOVP	H5	D10
3	MOVP	H0	D11
5	MOVP	H0	D12
7	IMASK	D10	
9	EI		
10	LD	X0	
11	OUT	Y0	
12	FEND		
13	I3		
14	LD	M0	
15	INC	D0	
17	IRET		
18	END		

6.7 I/O Refresh Instructions

I/O refresh

RFS(P)

Basic | High performance | Process | Redundant | Universal | LCPU



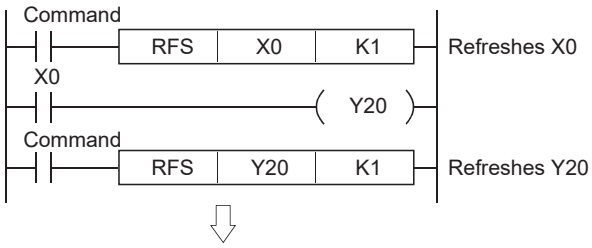
(S): Head number of the devices to be refreshed (bits)
n: Number of refreshes (BIN 16 bits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(S)	○ (Only X, Y)	—							—
n	○	○							—

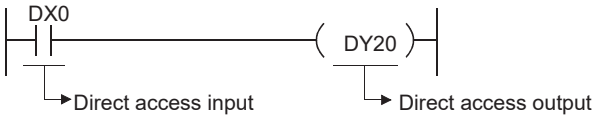
Processing details

- Refreshes only the device being scanned during a scan, and functions to fetch input from external sources or to output data to an output module.
- Fetching of input from or sending output to an external source is conducted in batch only after the execution of the END instruction, so it is not possible to output a pulse signal to an outside source during the execution of a scan. When the I/O refresh instruction is executed, the inputs (X) or outputs (Y) of the corresponding device numbers are refreshed forcibly midway through program execution. Therefore, a pulse signal can be output to an external source during a scan.
- Use direct access inputs (DX) or direct access outputs (DY) to refresh inputs (X) or outputs (Y) in 1-point units.

[Program based on the RFS instruction]



[Program based on direct access input and direct access output]



Operation error

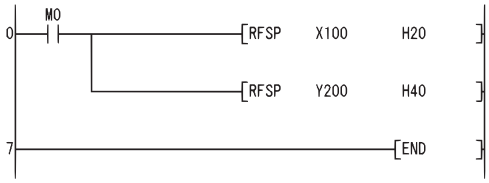
- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4101	The points specified in n exceed those of the proximate I/O.	○	○	○	○	○	—

Program example

- The following program refreshes X100 to X11F and Y200 to Y23F when M0 goes ON.

[Ladder Mode]



[List Mode]

Step	Instruction	Device	
0	LD	M0	
1	RFSP	X100	H20
4	RFSP	Y200	H40
7	END		

6.8 Other Convenient Instructions

Counter 1-phase input up or down

UDCNT1

Basic | High performance | Process | Redundant | Universal | LCPU

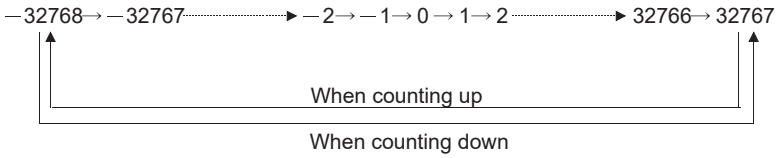
(S): (S)+0: Input number for count input (bits)
(S)+1: For setting count up/down (bits)
· OFF ··· Count up (add numbers when counting)
· ON ··· Count down (subtract numbers when counting)
(D): Number of the counter to be enabled to start counting with the UDCNT1 instruction (Device name)
n: Value to set (BIN 16 bits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(S)	○ (Only X)*1	—	—	—					—
(D)	—	△*2(Only C)	—	—					—
n	△*2	△*2	△*2	○					—

*1 Only the X device can be used for (S). However, the X device can be used only in the range of number of I/O points (the number of accessible points to actual I/O modules).
*2 Local devices and the file registers set for individual programs cannot be used.

Processing details

- When the input designated at (S) goes from OFF to ON, the present value of the counter designated at (D) will be updated.
- The direction of the count is determined by the ON/OFF status of the input designated by (S)+1.
  - OFF: Count up (counts by adding to the present value)
  - ON: Count down (counts by subtracting from the present value)
- When the count is going up, the counter contact designated at (D) goes ON when the present value becomes identical with the setting value designated by n. However, the present value count will continue even when the contact of the counter designated at (D) goes ON.
- When the count is going down, the counter for the contact specified by (D) turns off when the current value reaches the set value - 1.
- The counter specified by (D) is a ring counter. If it is counting up when the present value is 32767, the present value will become -32768. Further, if it is counting down when the present value is -32768, the present value will become 32767. The count processing performed on the present value is as shown below:



- The UDCNT1 instruction triggers counting when the execution command is turned OFF→ON and suspends counting when the execution command is turned ON→OFF. When the execution command is turned OFF→ON again, the counting resumes from the suspended value.
- The RST instruction clears the present value of the counter designated at (D) and turns the contact OFF.

Point

- With the UDCNT1 instruction, the argument device data is registered in the work area of the CPU module and counting operation is processed as a system interrupt. (The device data registered in the work area is cleared by turning the execution command OFF, or turning the STOP/RUN switch STOP→RUN.) For this reason, the pulses that can be counted must have longer ON and OFF times than the interrupt interval of the CPU module. The interrupt interval of the CPU module is 1ms.
- The set value cannot be changed during counting directed by the UDCNT1 instruction (while the execution command is ON). To change the set value, turn OFF the execution command.
- Counters designated by the UDCNT1 instruction cannot be used by any other instruction. If they are used by other instructions, they will not be capable of returning an accurate count.
- The UDCNT1 instruction can be used as many as 6 times within all the programs being executed. The seventh and the subsequent UDCNT1 instructions are not processed.

Operation error

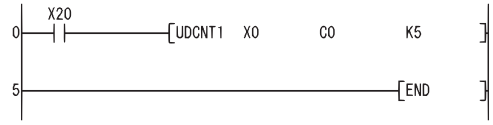
- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4101	The device specified by (S) exceeds the range of the corresponding device.	—	○	○	—	○	○

Program example

- This program uses C0 (Up/Down counter) to count the number of times X0 goes from OFF to ON after X20 has gone ON.

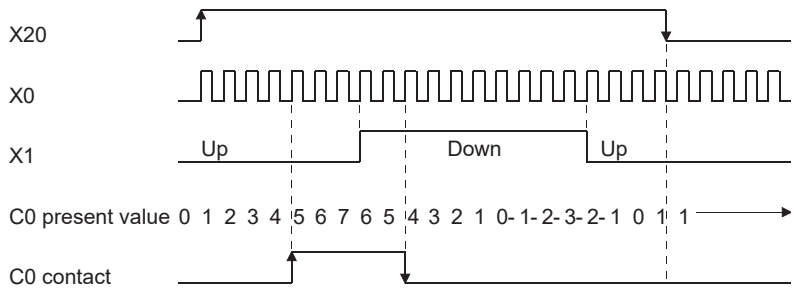
[Ladder Mode]



[List Mode]

Step	Instruction	Device		
0	LD	X20		
1	UDCNT1	X0	C0	K5
5	END			

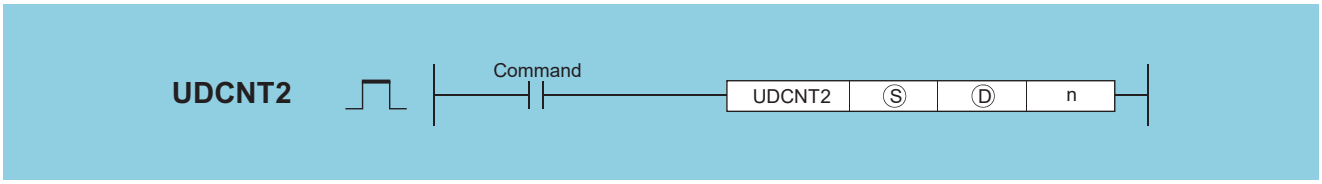
[Operation]



Counter 2-phase input up or down

UDCNT2

Basic | High performance | Process | Redundant | Universal | LCPU



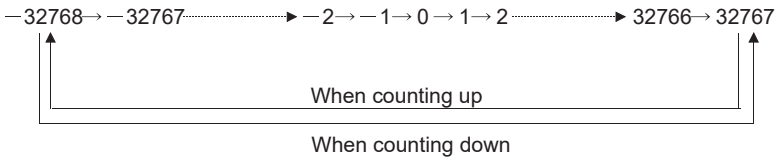
(S): (S)+0: Input number for count input (A phase pulse) (bits)
(S)+1: Input number for count input (B phase pulse) (bits)
(D): Number of the counter to be enabled to start counting with the UDCNT2 instruction (Device name)
n: Value to set (BIN 16 bits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(S)	○ (Only X)*1	—	—	—					—
(D)	—	△*2(Only C)	—	—					—
n	△*2	△*2	△*2	○					—

*1 Only the X device can be used for (S). However, the X device can be used only in the range of number of I/O points (the number of accessible points to actual I/O modules).
*2 Local devices and the file registers set for individual programs cannot be used.

Processing details

- The present value of the counter designated by (D) is updated depending on the status of the input designated by (S) (A phase pulse) and the status of the input designated by (S)+1 (B phase pulse).
- Direction of the count is determined in the following manner:
  - When (S) is ON, if (S)+1 goes from OFF to ON, count up operation is performed (values are added to the present value of the counter).
  - When (S) is ON, if (S)+1 goes from ON to OFF, count down operation is performed (values are subtracted from the present value of the counter).
  - No count operation is performed if (S) is OFF.
- When the count is going up, the counter contact designated at (D) goes ON when the present value becomes identical with the setting value designated by n. However, the present value count will continue even when the contact of the counter designated at (D) goes ON.
- When the count is going down, the counter for the contact specified by (D) turns off when the current value reaches the set value - 1.
- The counter specified by (D) is a ring counter. If it is counting up when the present value is 32767, the present value will become -32768. Further, if it is counting down when the present value is -32768, the present value will become 32767. The count processing performed on the present value is as shown below:



- Count processing conducted according to the UDCNT2 instruction begins when the count command goes from OFF to ON, and is suspended when it goes from ON to OFF. When the execution command is turned OFF to ON again, the counting resumes from the suspended value.
- The RST instruction clears the present value of the counter designated at (D) and turns the contact OFF.

Point

- With the UDCNT2 instruction, the argument device data is registered in the work area of the CPU module and counting operation is processed as a system interrupt. (The device data registered in the work area is cleared by turning the execution command OFF, or turning the STOP/RUN switch STOP→RUN.) For this reason, the pulses that can be counted must have longer ON and OFF times than the interrupt interval of the CPU module. The interrupt interval of the CPU module is 1ms.
- The set value cannot be changed during counting directed by the UDCNT2 instruction (while the execution command is ON). To change the set value, turn OFF the execution command.
- Counters designated by the UDCNT2 instruction cannot be used by any other instruction. If they are used by other instructions, they will not be capable of returning an accurate count.
- The UDCNT2 instruction can be used as many as 5 times within all the programs being executed. The sixth and the subsequent UDCNT2 instructions are not processed.

Operation error

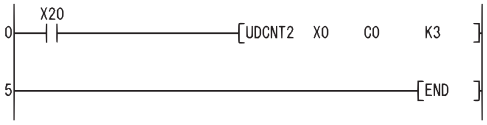
- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4101	The device specified by (S) exceeds the range of the corresponding device.	—	○	○	—	○	○

Program example

- The following program performs a count operation as instructed by C0 (count up or down) on the status of X0 and X1 after X20 has gone ON.

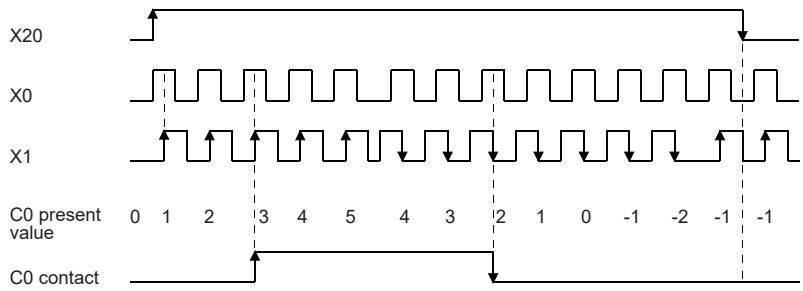
[Ladder Mode]



[List Mode]

Step	Instruction	Device		
0	LD	X20		
1	UDCNT2	X0	C0	K3
5	END			

[Operation]



Teaching timer

TTMR

Basic | High performance | Process | Redundant | Universal | LCPU

(D): (D)+0: The device where measurement value is stored (BIN 16 bit)
(D)+1: For CPU module system use (BIN 16 bit)
n: Measurement value multiplier (BIN 16 bits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(D)	—	○		—					—
n	—	○		○					—

Processing details

- Measures the time while the execution command is ON, and stores the multiplied value of the measured time by the multiplier specified by n at the device specified by (D).
- Clears the device specified by (D)+0 or (D)+1 when the execution command is turned OFF→ON.
- The multipliers that can be specified by n and the measurement units are as shown below:

n	Multiplier	Measurement unit
0	1	In units of seconds
1	10	In increments of 100ms
2	100	

Point

- Time measurements are conducted when the TTMR instruction is executed. Using the JMP or similar instruction to jump the TTMR instruction will make it impossible to get an accurate measurement.
- Do not change the multiplier designated by n while the TTMR instruction is being executed. Changing this multiplier will result in an inaccurate value being returned.
- The TTMR instruction can also be used in low speed execution type programs.
- The device designated by (D)+1 is used by the system of the CPU module, so users should not change its value. If users do change this value, the value stored in the device designated by (D) will no longer be accurate.

- No processing is performed when the value specified by "n" is other than 0 to 2.

Operation error

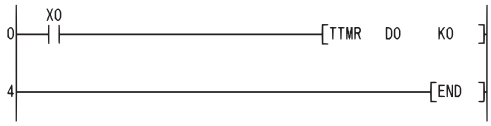
- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4101	The device specified by (D) exceeds the range of the corresponding device.	—	○	○	—	○	○

Program example

- The following program stores the amount of time that X0 is ON at D0.

[Ladder Mode]



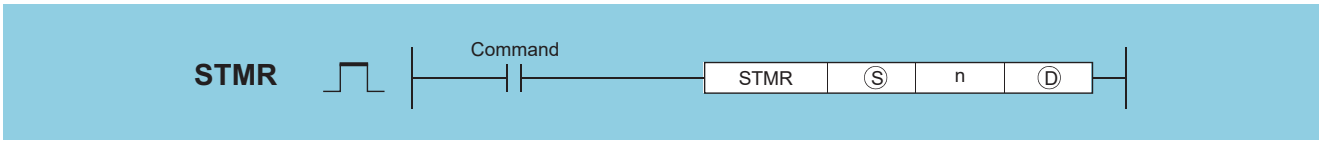
[List Mode]

Step	Instruction	Device	
0	LD	X0	
1	TTMR	D0	K0
4	END		

Special function timer

STMR

Basic | High performance | Process | Redundant | Universal | LCPU



(S): Timer number (word)
n: Value to set (BIN 16 bits).
(D): (D)+0: Off delay timer output (bits)
(D)+1: One shot timer output after OFF (bits)
(D)+2: One shot timer output after ON (bits)
(D)+3: ON delay and Off delay timer output (bits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(S)	—	△*1	—	—					—
n	○	○	○	○					—
(D)	○	—	—	—					—

*1 Can be used only by timer (T) data

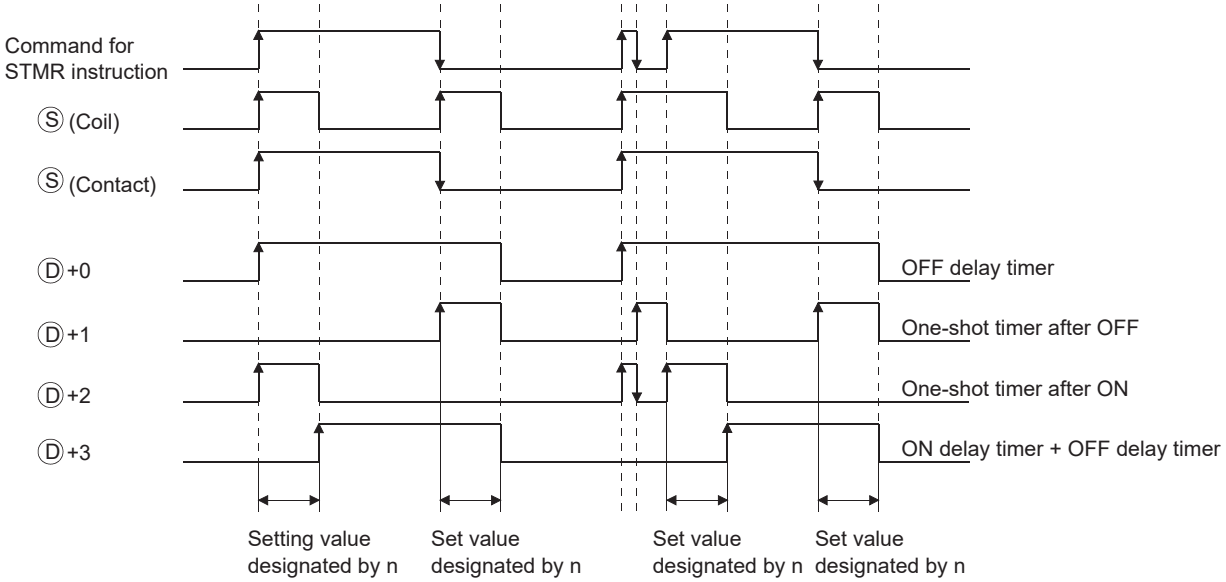
Processing details

- The STMR instruction uses the 4 points from the device designated by (D) to perform four types of timer output.

Item	Description
OFF delay timer output ((D)+0)	Goes ON at the rising edge of the command for the STMR instruction, and after the falling edge of the command, goes OFF when the amount of time designated by n has passed.
One shot timer output after OFF ((D)+1)	Goes ON at the falling edge of the command for the STMR instruction, and goes OFF when the amount of time designated by n has passed.
One shot timer output after ON ((D)+2)	Goes ON at the rising edge of the command for the STMR instruction, and goes OFF either when the amount of time designated by n has passed, or when the command for the STMR instruction goes OFF.
ON delay timer output ((D)+3)	Goes ON at the falling edge of the timer coil, and after the falling edge of the command for the STMR instruction, goes OFF when the amount of time designated by n has passed.

- The timer coil designated by (S) turns ON at the rising edge and falling edge of the command for the STMR instruction, and starts measurement of the present value.
  - The timer coil measures to the point where the value reaches the set value designated by n, then enters a time up state and goes OFF.
  - If the command for the STMR instruction goes OFF before the timer coil reaches the time up state, it will remain ON. Timer measurement is continued at this time. When the STRM instruction command goes ON once again, the present value will be cleared to 0 and measurement will begin once again.

- The timer contact goes ON at the rising edge of the command for the STMR instruction, and after the falling edge is reached, the timer coil goes OFF at the falling edge of the STMR instruction command. The timer contact is used by the CPU module system, and cannot be used by the user.



- Measurement of the present value of the timer specified by the STMR instruction is executed regardless of the command ON/OFF status of the STMR instruction. If the STMR instruction is jumped with the JMP or similar instruction, it will not be possible to get accurate measurement.
- Measurement unit for the timer designated by (D) is identical to the low speed timer.
- A value between 0 to 32767 can be set for n. No operation if n is other than 0 to 32767.
- The timer designated by (S) cannot be used by the OUT instruction. If the STMR instruction and the OUT instruction use the same timer number, accurate operation will not be conducted.

Operation error

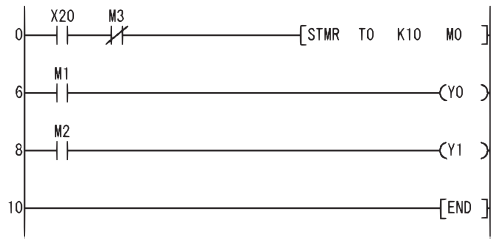
- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4101	The device specified by (D) exceeds the range of the corresponding device.	—	○	○	—	○	○

Program example

• The following program turns Y0 and Y1 ON and OFF once each second (flicker) when X20 is ON. (Uses 100ms timer)

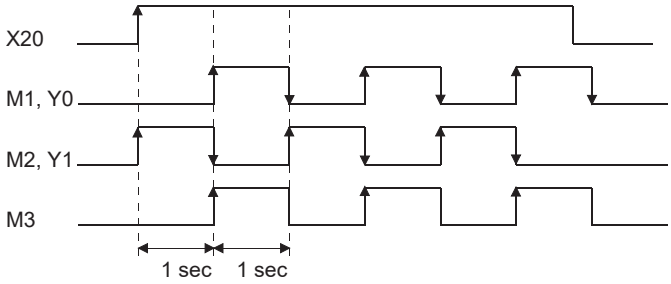
[Ladder Mode]



[List Mode]

Step	Instruction	Device		
0	LD	X20		
1	ANI	M3		
2	STMR	T0	K10	M0
6	LD	M1		
7	OUT	Y0		
8	LD	M2		
9	OUT	Y1		
10	END			

[Timing Chart]



Precautions

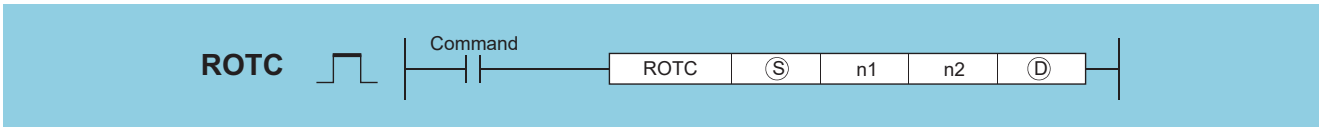
Note that the STMR instruction operates when the instruction is used within the range written data by the online program change.

For details, refer to the User's Manual (Function Explanation, Program Fundamentals) for the CPU module used.

Rotary table shortest direction control

ROTC

Basic | High performance | Process | Redundant | Universal | LCPU



(S): (S)+0: Measures the number of table rotations (for system use) (BIN 16 bits)
(S)+1: Call station number (BIN 16 bits)
(S)+2: Call item number (BIN 16 bits)
n1: Number of divisions of table (2 to 32767) (BIN 16 bits)
n2: Number of low-speed sections (value from 0 to less than n1) (BIN 16 bits)
(D): (D)+0: A phase input signal (bits)
(D)+1: B phase input signal (bits)
(D)+2: 0 point detection input signal (bits)
(D)+3: High speed forward rotation output signal (for system use) (bits)
(D)+4: Low speed forward rotation output signal (for system use) (bits)
(D)+5: Stop output signal (for system use) (bits)
(D)+6: Low speed reverse rotation output signal (for system use) (bits)
(D)+7: High speed reverse rotation output signal (for system use) (bits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(S)	—	○		—					—
n1	○	○		○					—
n2	○	○		○					—
(D)	○	—		—					—

Processing details

- This control functions to enable shortest direction control of the rotary table to the position of the station number designated by (S)+1 in order to remove or deposit an item whose number has been designated by (S)+2 on a rotary table with equal divisions of the value designated by n1.
- The item number and station number are controlled as items allocated by counterclockwise rotation.
- The system uses (S)+0 as a counter to instruct it as to what item is at which number counting from station number 0. Do not rewrite the sequence program data. Accurate controls will not be possible in cases where users have rewritten the data.
- The value of n2 should be less than the number of table divisions specified by n1.
- (D)+0 and (D)+1 are A and B phase input signals that are used to detect whether the direction of the rotary table rotation is forward or reverse. The direction of rotation is judged by whether the B phase pulse is at its rising or falling edge when the A phase pulse is ON:
  - When the B phase is at the rising edge: Forward rotation (clockwise rotation)
  - When the B phase is at the falling edge: Reverse rotation (counterclockwise rotation)
- (D)+2 is the 0 point detection output signal that goes ON when item number 0 has arrived at the No. 0 station. When the device designated by (D)+2 goes ON while the ROTC instruction is being executed, (S)+0 is cleared. It is best to perform this clear operation first, then to begin shortest direction control with the ROTC instruction.
- The data from (D)+3 to (D)+7 consists of output signals needed to control the table's operation. The output signal of one of the devices from (D)+3 to (D)+7 will go ON in response to the execution results of the ROTC instruction.
- If the command for the ROTC instruction is OFF, clears all (D)+3 to (D)+7 without performing shortest direction control.
- The ROTC instruction can be used only one time in all programs where it is executed. Attempts to use it more than one time will result in inaccurate operations.
- No processing is performed when the value of (S)+0 to (S)+2, or the value of n2 is greater than n1.

Operation error

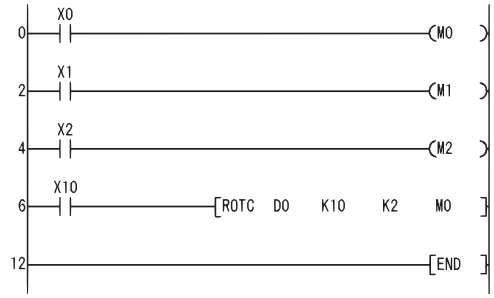
- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4101	The device specified by (S) or (D) exceeds the range of the corresponding device.	—	○	○	—	○	○

Program example

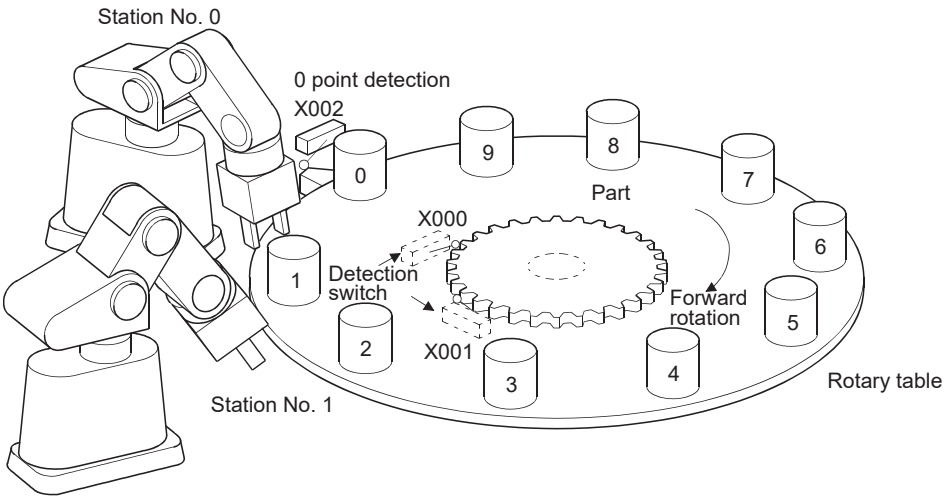
- The following program deposits the item at section D2 on a 10-division rotary table at the station at section D1, and the two sections ahead and behind this determine the rotation direction and control speed of the motor when the table is being rotated at low speed.

[Ladder Mode]



[List Mode]

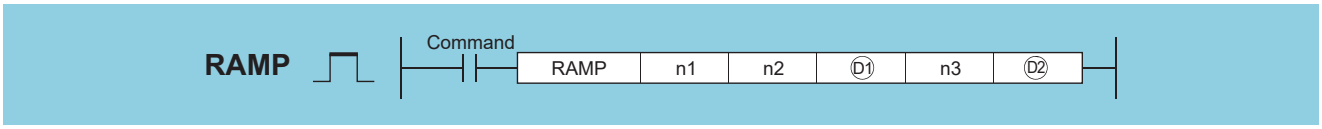
Step	Instruction	Device
0	LD	X0
1	OUT	M0
2	LD	X1
3	OUT	M1
4	LD	X2
5	OUT	M2
6	LD	X10
7	ROTC	D0 K10 K2 M0
12	END	



Ramp signal

RAMP

Basic | High performance | Process | Redundant | Universal | LCPU



n1: Initial value (BIN 16 bits)
n2: Final value (BIN 16 bits)
(D1): (D1)+0: Present value (BIN 16 bits)
(D1)+1: Number of executions (BIN 16 bits)
n3: Number of shifts (BIN 16 bits)
(D2): (D2)+0: Completion device (bits)
(D2)+1: Bit for selecting data retaining at completion (bit)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
n1	○	○						○	—
n2	○	○						○	—
(D1)	○	○						—	—
n3	○	○						○	—
(D2)	○	—						—	—

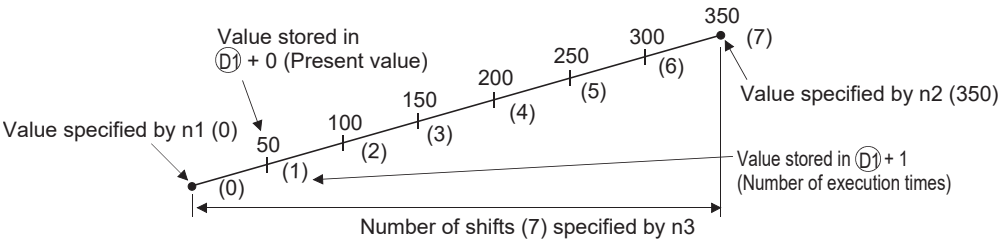
Processing details

- When the execution command is ON, the following processing is executed.
  - Shifts from the value specified by n1 to the value specified by n2 in the number of times specified by n3.
  - For n3, designate the number of scans (number of shifts) required for shift from n1 to n2.
  - No operation if other than 0<n3<32768.
  - The system uses (D1)+1 to store the number of times the instruction has been executed.
  - The value of one variation (one scan) is obtained by the expression below:

$$\text{Value of one variation (one scan)} = \frac{(\text{Value specified by n2}) - (\text{Value specified by n1})}{(\text{Value specified by n3})}$$

Ex.

0 is varied to 350 in seven scans as shown below.



When the calculated one variation is indivisible, compensation is made to achieve the value specified in n2 by the number of shifts specified in n3. Hence, a linear ramp may not be made.

- If the scan is performed for the number of moves specified by n3, the complete device specified by (D2) +0 is turned ON. The ON/OFF status of the completion device and the contents of (D1)+0 are determined by the ON/OFF status of the device designated by (D2)+1.
  - When (D2)+1 is OFF, +0 will go OFF at the next scan, and the RAMP instruction will begin a new move operation from the value currently at (D2)+0.
  - When (D2)+1 is ON, (D2)+0 will remain ON, and the contents of (D1)+0 will not change.
- When the command is turned OFF during the execution of this instruction, the contents of (D1)+0 will not change following this. When the command goes ON again, the RAMP instruction will begin a new move from the present value at +0.
- Do not change the specified values in n1 and n2 before the completion device specified in (D2)+0 turns ON. Since the same expression is used every scan to calculate the value stored in (D1)+1, changing n1/n2 may cause a sudden variation.

Operation error

- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4101	The device specified by (D1) or (D2) exceeds the range of the corresponding device.	—	○	○	—	○	○

Precautions

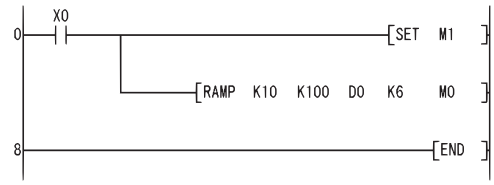
- When the digit specification of bit device is made to (D1), the digit specification of bit device can only be used when the following condition is met.

Specification of digits: K8

Program example

- The following program changes the contents of D0 from 10 to 100 in a total of 6 scans, and saves the contents of D0 when the move has been completed.

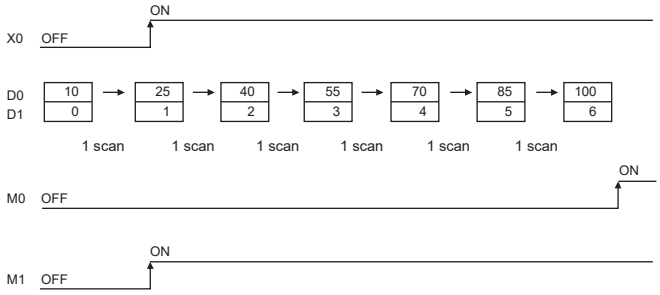
[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	X0
1	SET	M1
2	RAMP	K10 K100 D0 K6 M0
8	END	

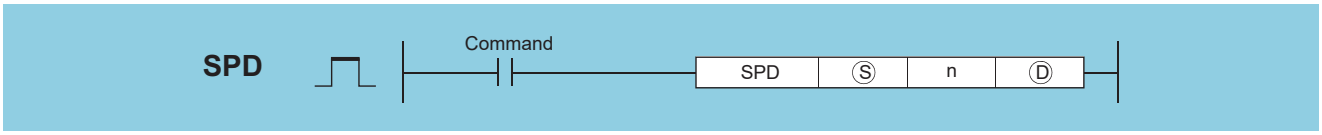
[Timing Chart]



Pulse density measurement

SPD

Basic | High performance | Process | Redundant | Universal | LCPU



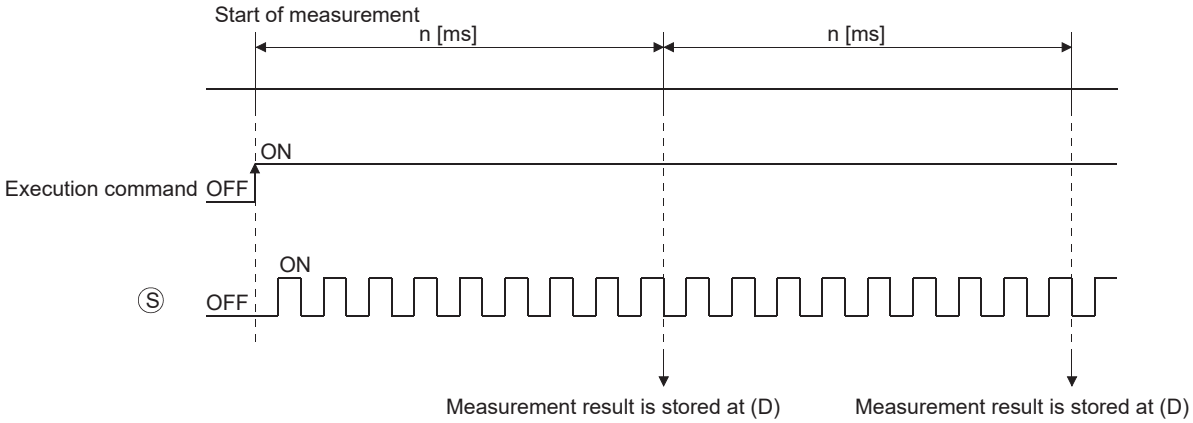
(S): Number for the device to which pulses are input (bits)
n: Measurement time or the number for the device where the measurement time is stored (unit: ms) (BIN 16 bits)
(D): Devices where the measurement results will be stored (BIN 16 bits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(S)	○ (Only X)	—		—					—
n	△*1	△*1		○					—
(D)	—	△*1		—					—

*1 Only the input (X) can be used. Note, however, that the input (X) can be used only within the range of the number of I/O points (the number of points that can access I/O modules).
*2 Local devices and the file registers set for individual programs cannot be used.

Processing details

- The number of turning OFF→ON input of the device specified by (S) is counted for just the amount of time specified by n, and the count results are stored in the device specified by (D).



- When measurement directed by the SPD instruction has been completed, measurement is done again from 0. Turn OFF the execution command to stop the measurement directed by the SPD instruction.

Point

- With the SPD instruction, the argument device data is registered in the work area of the CPU module and counting operation is processed as a system interrupt. (The device data registered in the work area is cleared by turning the execution command OFF, or turning the STOP/RUN switch STOP→RUN.) For this reason, the pulses that can be counted must have longer ON and OFF times than the interrupt interval of the CPU module. The interrupt interval of the CPU module is 1ms.
- The instruction is not processed when n = 0.
- The SPD instruction can be used as many as 6 times within all the programs being executed. The seventh and the subsequent SPD instructions are not processed.
- While the measurement is in execution (while the command input is ON) by the SPD instruction, the setting value cannot be changed. Turn OFF the command input before changing the setting value.

Operation error

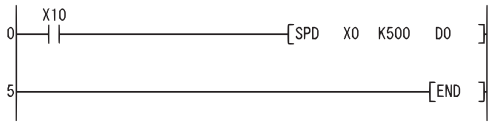
- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4101	The device specified by (S) exceeds the range of the corresponding device.	—	○	○	—	○	○

Program example

- The following program measures the pulses input to X0 for a period of 500ms when X10 goes ON, and stores the result at D0.

[Ladder Mode]



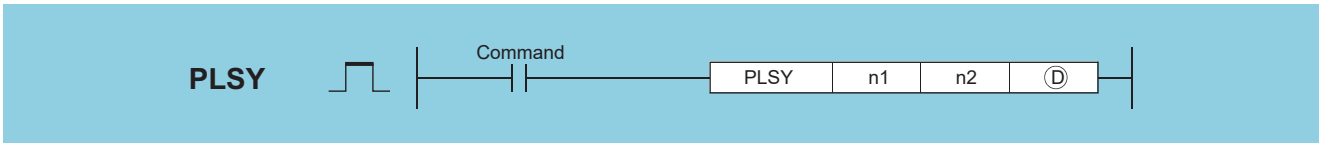
[List Mode]

Step	Instruction	Device		
0	LD	X10		
1	SPD	X0	K500	D0
5	END			

Fixed cycle pulse output

PLSY

Basic | High performance | Process | Redundant | Universal | LCPU



n1: Frequency or the number of the device where frequency is stored (BIN 16 bits)
n2: Outputs count or the number of the device where the outputs count is stored (BIN 16 bits)
(D): Number of the device to which pulses are output (bits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
n1	○	○							—
n2	○	○							—
(D)	△*1	—							—

*1 Only the output (Y) can be used. Note, however, that the output (Y) can be used only within the range of the number of I/O points (the number of points that can access I/O modules).

Processing details

- Outputs a pulse at a frequency designated by n1 the number of times designated by n2, to the output module with the output signal (Y) designated by (D).
- Frequencies between 1 to 100Hz can be designated by n1. If n1 is other than 1 to 100Hz, the PLSY instruction will not be executed.
- The number of outputs that can be designated by n2 is between 0 to 65535 (0000H to FFFFH). If n2 is set to "0", pulses are continuously output.
- Only an output number corresponding to the output module can be designated for pulse output at (D).
- Pulse output commences with the command rising edge of the PLSY instruction. Pulse output is suspended when the PLSY instruction command goes OFF.

Point

- With the PLSY instruction, the argument device data is registered in the work area of the CPU module and counting operation is processed as a system interrupt. (The device data registered in the work area is cleared by turning the execution command OFF, or turning the STOP/RUN switch STOP→RUN.) For this reason, the pulses that can be output must have longer ON and OFF times than the interrupt interval of the CPU module. The interrupt interval of the CPU module is 1ms.
- Do not change the argument for the PLSY instruction during pulse output directed by the PLSY instruction (while the execution command is ON). To change the argument, turn OFF the execution command.
- The PLSY instruction can be used only once in all programs executed by the CPU module. The second and the subsequent PLSY instructions are not processed.

Operation error

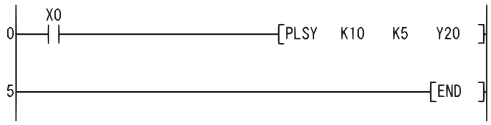
- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4101	The device specified by (D) exceeds the range of the corresponding device.	—	○	○	—	○	○

Program example

• The following program outputs a 10Hz pulse 5 times to Y20 when X0 is ON.

[Ladder Mode]



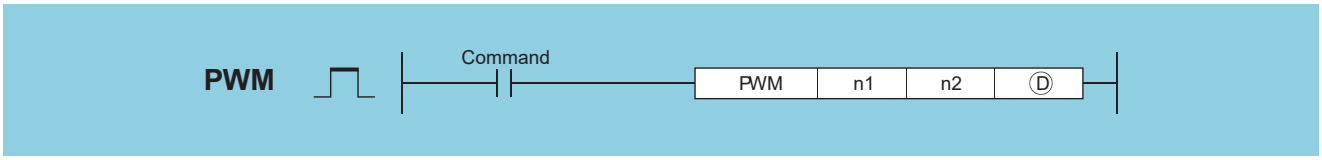
[List Mode]

Step	Instruction	Device		
0	LD	X0		
1	PLSY	K10	K5	Y20
5	END			

Pulse width modulation

PWM

Basic | High performance | Process | Redundant | Universal | LCPU



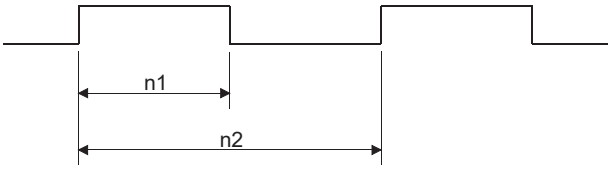
n1: ON time or the number for the device where the ON time is stored (unit: ms) (BIN 16 bits)
n2: Frequency or the number for the device where the frequency is stored (unit: ms) (BIN 16 bits)
(D): Number of the device to which pulses are output (bits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
n1	○	○							—
n2	○	○							—
(D)	△*1	—							—

*1 Only the output (Y) can be used. Note, however, that the output (Y) can be used only within the range of the number of I/O points (the number of points that can access I/O modules).

Processing details

- Outputs the pulse of the cycle set by n2, for the amount of time ON designated by n1, to the output module designated by (D).



- The setting ranges for n1 and n2 are shown below:

CPU Module Type Name	Setting Range for n1 and n2 [ms]*2
High Performance model QCPU, Process CPU Universal model QCPU, LCPU	1 to 65535 (0001H to FFFFH)

*2 The value specified by n1 should be less than the value specified by n2.

Point

- With the PWM instruction, the argument device data is registered in the work area of the CPU module and counting operation is processed as a system interrupt. (The device data registered in the work area is cleared by turning the execution command OFF, or turning the STOP/RUN switch STOP→RUN.) The interrupt interval (interrupt interval of n1, n2) of the CPU module is 1ms. For this reason, the PWM instruction can be used only once within all the programs being executed by the CPU module.
- When both n1 and n2 are 0, when n1 ≥ n2, or when the PWM instruction is executed twice or more, the instruction is not processed.
- Do not change the argument for the PWM instruction during pulse output directed by the PWM instruction (while the execution command is ON). To change the argument, turn OFF the execution command.

Operation error

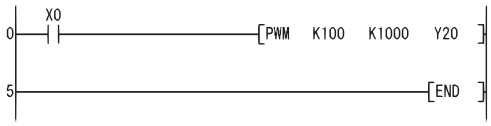
- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4101	The device specified by (D) exceeds the range of the corresponding device.	—	○	○	—	○	○

Program example

• The following program outputs a 100ms pulse once each second to Y20 when X0 is ON.

[Ladder Mode]



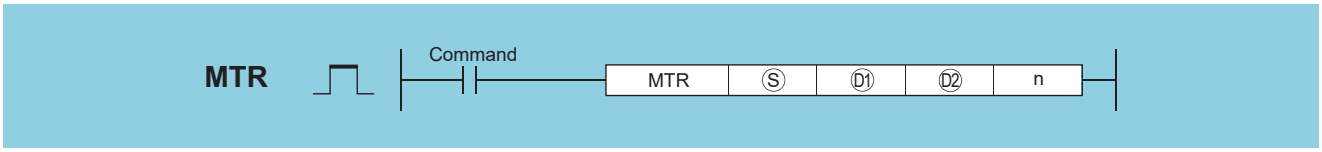
[List Mode]

Step	Instruction	Device
0	LD	X0
1	PWM	K100 K1000 Y20
5	END	

Matrix input

MTR

Basic | High performance | Process | Redundant | Universal | LCPU



(S): Head input device (bits)
(D1): Head output device (bits)
(D2): Head number of the devices where matrix input data will be stored (bits)
n: Number of input rows (BIN 16 bit)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(S)	○ (Only X)	—							—
(D1)	○ (Only Y)	—							—
(D2)	○	—							—
n	○	○							—

Processing details

- This instruction reads the input from 16 points × n-rows starting from the input number designated by (S), then stores fetched input data to area starting from the device designated by (D2).
- One row (16 points) can be fetched in 1 scan.
- Fetching from the first to the n th row is repeated.
- The first through the 16th points store the first row of data and the next 16 points store the second row of data at the devices following the device designated by (D2). For this reason, the space of 16 × n points from the device designated by (D2) are occupied by the MTR instruction.
- (D1) is the output needed to select the row which will be fetched, and the system automatically turns it ON and OFF. It uses the n points from the device designated by (D1).
- Only device numbers divisible by 16 can be designated for (S), (D1) and (D2).
- For n, a value in the range from 2 to 8 can be assigned.
- No processing is performed in the following cases.
  - The device number designated by (S), (D1), or (D2) is not divisible by 16.
  - The device designated by (S) is outside the actual input range.
  - The device designated by (D1) is outside the actual output range.
  - The space 16 × n points following the device designated by (D2) exceeds the relevant device range.
  - The value for n is not between 2 and 8.

Operation error

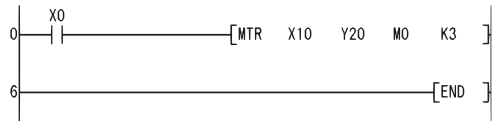
- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4101	The device other than the input (X) was specified at (S). The device other than the output (Y) was specified at (D1).	—	○	○	—	○	○

Program example

- The following program fetches, when X0 is turned ON, the 16 points×3 matrix starting from X10, and stores the matrix into the area starting from M0.

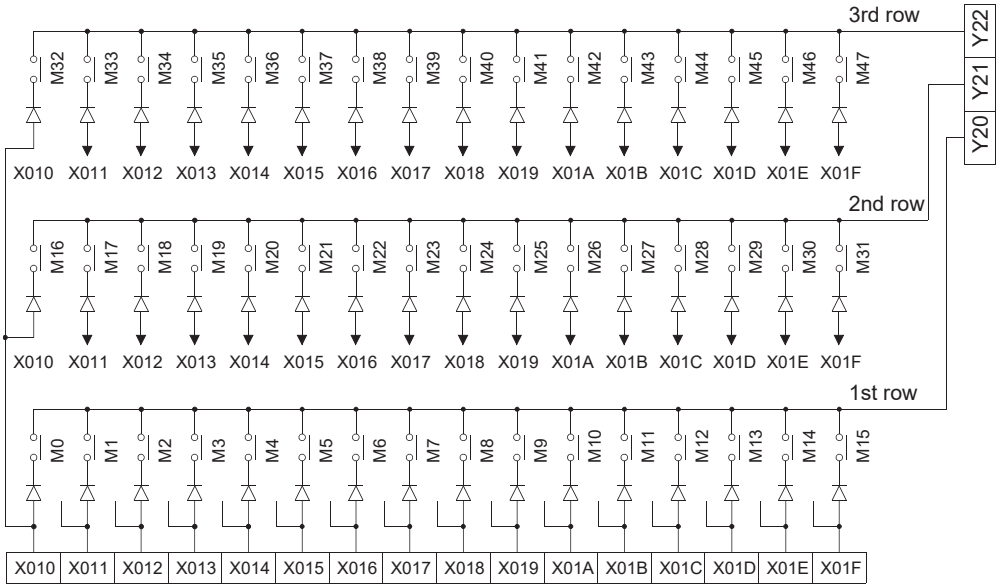
[Ladder Mode]



[List Mode]

Step	Instruction	Device			
0	LD	X0			
1	MTR	X10	Y20	M0	K3
6	END				

[Operation]



Precautions

- Note that the MTR instruction directly operates on actual input and output. The output (D1) that had been turned ON by the MTR instruction does not turn OFF when the MTR command turns OFF. Turn OFF the specified output (D1) in the sequence program.
- The MTR instruction execution interval must be longer than the total of response time of input and output modules. If the set interval is shorter than the value indicated above, an input cannot be read correctly. If the scan time in a sequence program is short, select the constant scan and set the scan time longer than the total of response time.

7 APPLICATION INSTRUCTIONS

7.1 Logical Operation Instructions

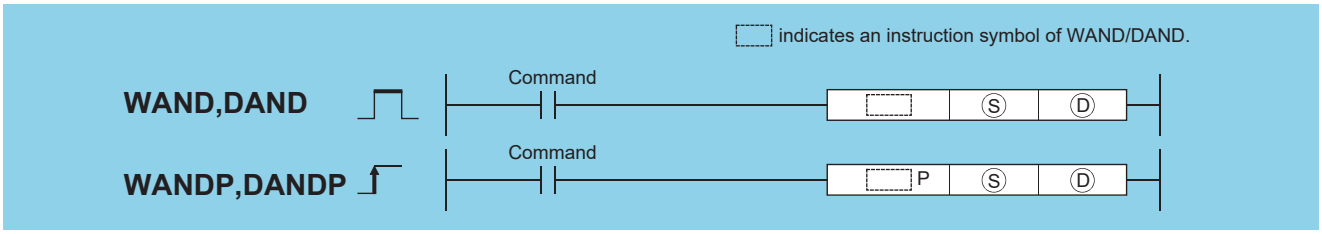
The logical operation instructions perform logical sum, logical product or other logical operations in 1-bit units.

Category	Processing details	Formula for operation	Example		
			A	B	Y
Logical product (AND)	Becomes 1 only when both input A and input B are 1; otherwise, is 0	$Y = A \cdot B$	0	0	0
			0	1	0
			1	0	0
			1	1	1
Logical sum (OR)	Becomes 0 only when both input A and input B are 0; otherwise, is 1	$Y = A + B$	0	0	0
			0	1	1
			1	0	1
			1	1	1
Exclusive OR (XOR)	Becomes 0 if input A and input B are equal; otherwise, is 1	$Y = \overline{A} \cdot B + A \cdot \overline{B}$	0	0	0
			0	1	1
			1	0	1
			1	1	0
NON exclusive logical sum (XNR)	Becomes 1 if input A and input B are equal; otherwise, is 0	$Y = (\overline{A} + B)(A + \overline{B})$	0	0	1
			0	1	0
			1	0	0
			1	1	1

Logical products with 16-bit data, logical products with 32-bit data

WAND(P), DAND(P) [When two data are set]

Basic | High performance | Process | Redundant | Universal | LCPU



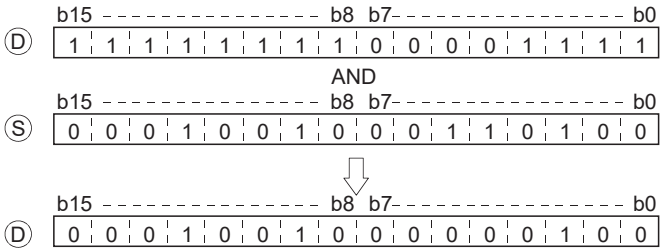
(S): Data for a logical product operation or the head number of the devices where the data is stored (BIN 16/32 bits)
(D): Head number of the devices where the logical product operation result will be stored (BIN 16/32 bits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(S)	○							○	—
(D)	○							—	—

Processing details

■WAND

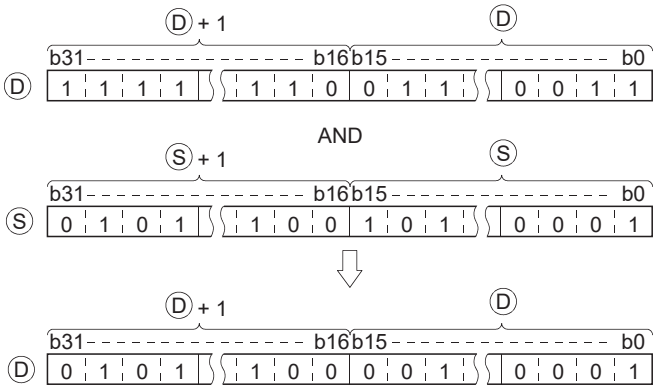
- A logical product operation is conducted for each bit of the 16-bit data of the device designated at (D) and the 16-bit data of the device designated at (S), and the results are stored in the device designated at (D).



- For bit devices, the bit devices after the points designated by digit specification are regarded as "0" in the operation.

■DAND

- Conducts a logical product operation on each bit of the 32-bit data for the device designated by (S1) and the 32-bit data for the device designated by (S2), and stores the results at the device designated by (D).



- For bit devices, the bit devices after the points designated by digit specification are regarded as "0" in the operation.

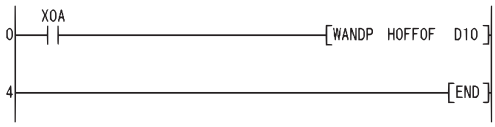
Operation error

- There is no operation error in the WAND(P) or DAND(P) instruction.

Program example

- The following program masks the digit in the 10s place of the 4-digit BCD value at D10 (second digit from the end) to 0 when XA is turned ON.

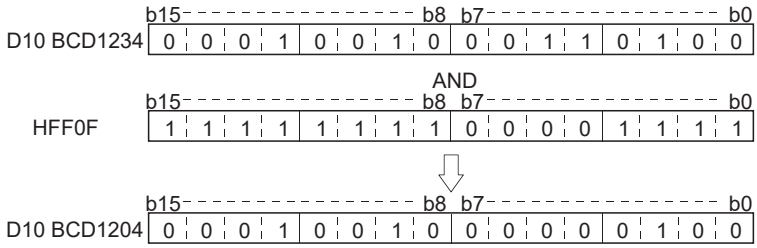
[Ladder Mode]



[List Mode]

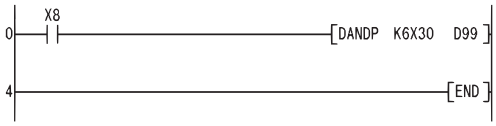
Step	Instruction	Device
0	LD	X0A
1	WANDP	HOFF0F D10
4	END	

[Operation]



- The following program performs a logical product operation on the data at D99 and D100, and the 24-bit data between X30 and X47 when X8 is ON, and stores the results at D99 and D100.

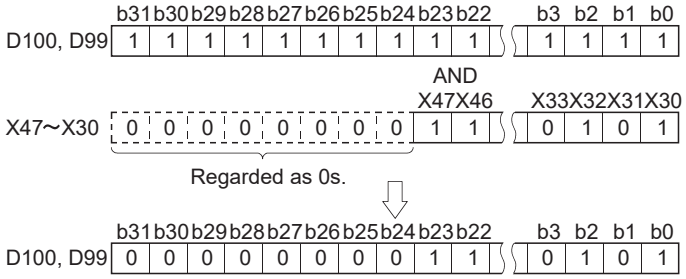
[Ladder Mode]



[List Mode]

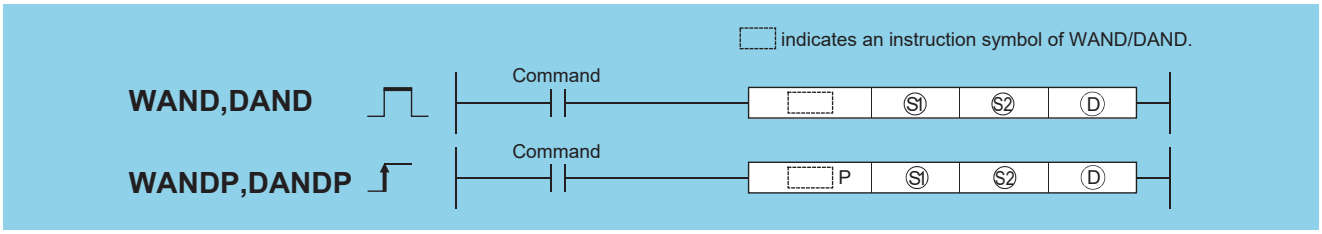
Step	Instruction	Device
0	LD	X8
1	DANDP	K6X30 D99
4	END	

[Operation]



7

WAND(P), DAND(P) [When three data are set]



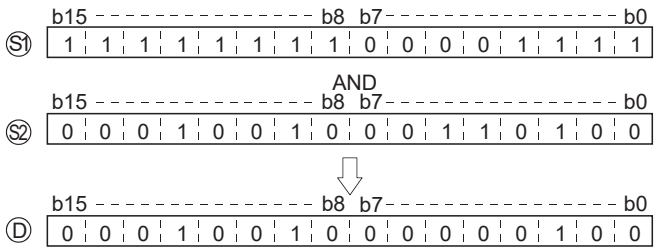
(S1), (S2) : Data for a logical product operation or the head number of the devices where the data is stored (BIN 16/32 bits)
(D): Head number of the devices where the logical product operation result will be stored (BIN 16/32 bits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(S1)	○							○	—
(S2)	○							○	—
(D)	○							—	—

Processing details

■WAND

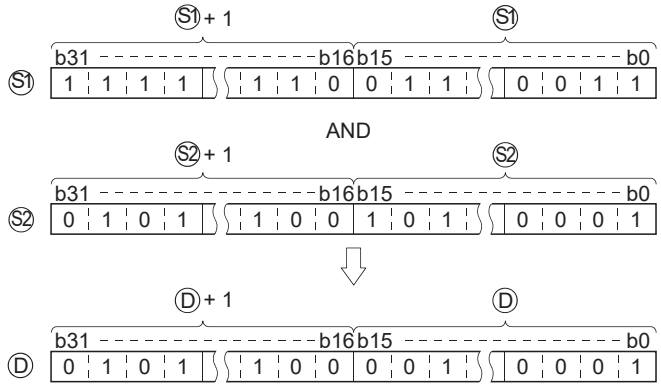
- A logical product operation is conducted for each bit of the 16-bit data of the device designated at (S1) and the 16-bit data of the device designated at (D).



- For bit devices, the bit devices after the points designated by digit specification are regarded as "0" in the operation.

■DAND

- Conducts a logical product operation on each bit of the 32-bit data for the device designated by (S1) and the 32-bit data for the device designated by (D), and stores the results at the device designated by (D).



- For bit devices, the bit devices after the points designated by digit specification are regarded as "0" in the operation.

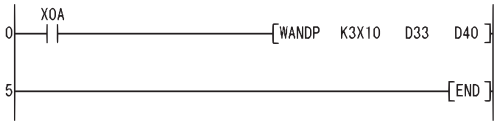
Operation error

- There is no operation error in the WAND(P) or DAND(P) instruction.

Program example

- The following program performs a logical product operation on the data from X10 to X1B and the data at D33 when XA is ON, and stores the results at D40.

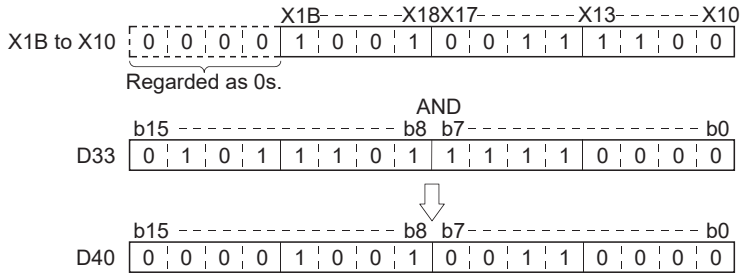
[Ladder Mode]



[List Mode]

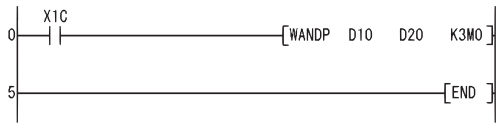
Step	Instruction	Device		
0	LD	X0A		
1	WANDP	K3X10	D33	D40
5	END			

[Operation]



- The following program performs a logical product operation on the data at D10 and at D20 when X1C is ON, and stores the results from M0 to M11.

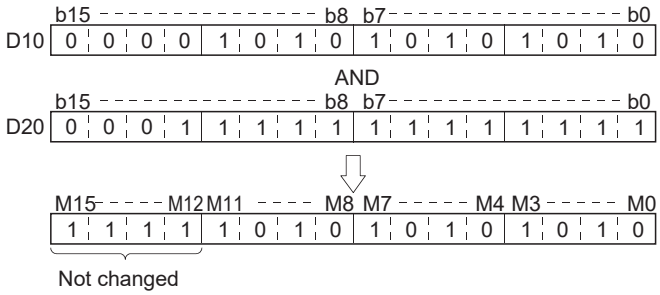
[Ladder Mode]



[List Mode]

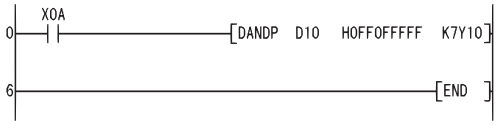
Step	Instruction	Device		
0	LD	X1C		
1	WANDP	D10	D20	K3M0
5	END			

[Operation]



• The following program masks the digit in the hundred-thousands place of the 8-digit BCD value at D10 and D11 (sixth digit from the end) to 0 when XA is ON, and outputs the results to from Y10 to Y2B.

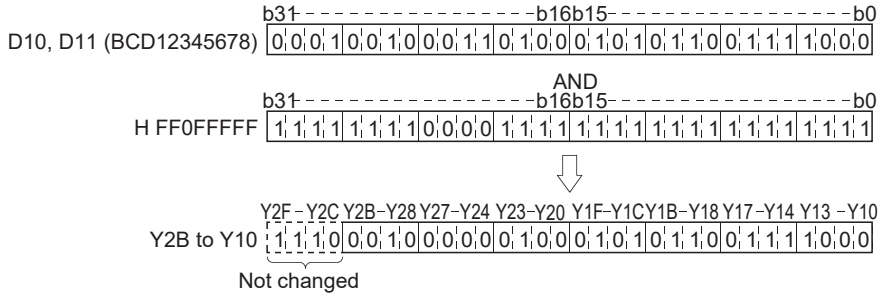
[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	X0A
1	DANDP	D10 H0FF0FFFFF K7Y10
6	END	

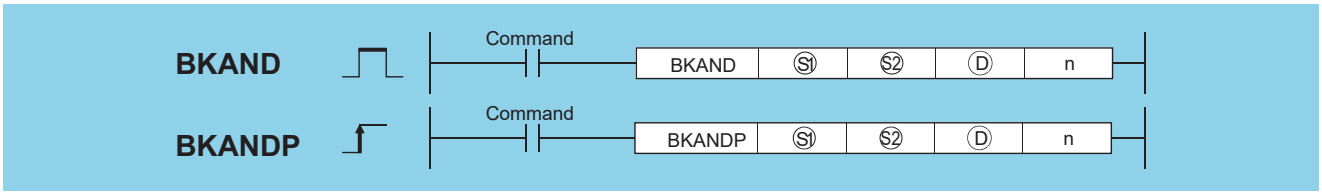
[Operation]



Block logical products

BKAND(P)

Basic | High performance | Process | Redundant | Universal | LCPU



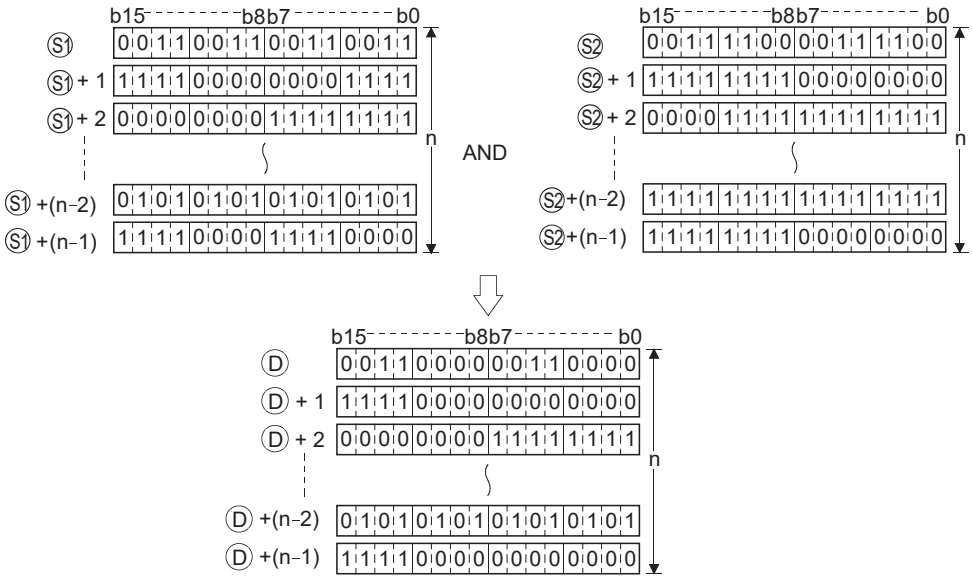
(S1)*1: Head number of the devices where data on which a logical operation will be conducted is stored (BIN 16 bits)
(S2)*1: Data for a logical operation or head number of the devices where the data for the logical operation is stored (BIN 16 bits)
(D)*1: Head number of the devices where the operation result will be stored (BIN 16 bits)
n: Number of operation data blocks (BIN 16 bits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(S1)*1	—	○		—				—	—
(S2)*1	—	○		—				○	—
(D)*1	—	○		—				—	—
n	○	○		○				○	—

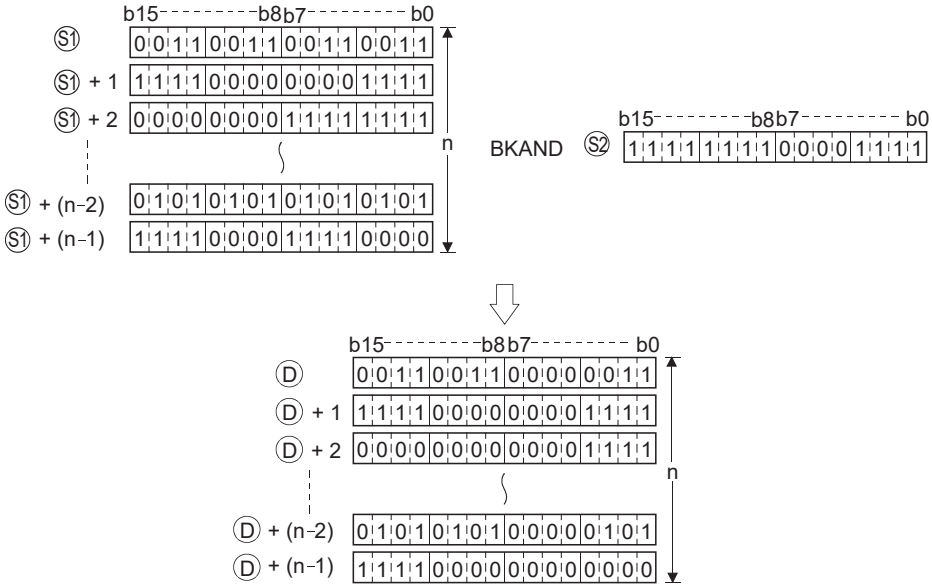
*1 The same device number can be specified for (S1) and (D) or (S2) and (D).

Processing details

- Performs a logical product operation on the data located in the n points from the device designated by (S1), and the data located in the n points from the device designated by (S2), and stores the results into the area starting from the device designated by (D).



• The constant designated by (S2) can be between -32768 and 32767 (BIN 16-bit data).



Operation error

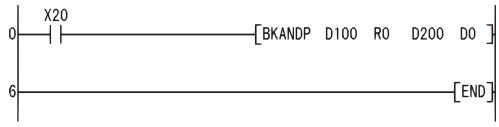
• In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4101	The points specified in n exceed those of the corresponding device specified in (S1), (S2), or (D). The ranges of devices starting from the one specified in (S1) and (D) overlap by n points (except when the same device is specified in (S1) and (D)). The ranges of devices starting from the one specified in (S2) and (D) overlap by n points (except when the same device is specified in (S2) and (D)).	○	○	○	○	○	○

Program example

• The following program performs a logical product operation on the data stored at D100 to D102 and the data stored at R0 to R2 when X20 is turned ON, and stores the operation result into the area starting from D200.

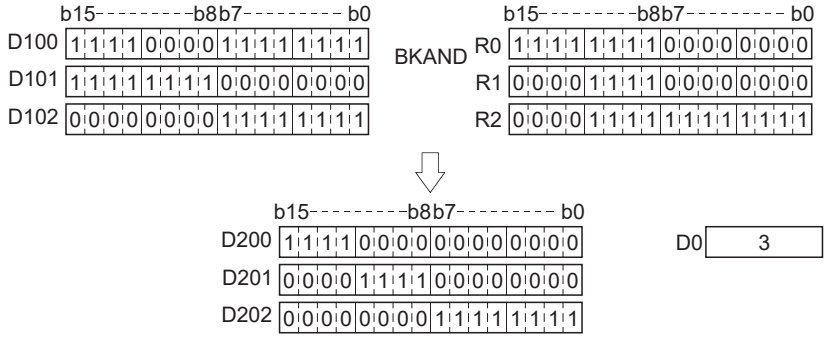
[Ladder Mode]



[List Mode]

Step	Instruction	Device			
0	LD	X20			
1	BKANDP	D100	R0	D200	D0
6	END				

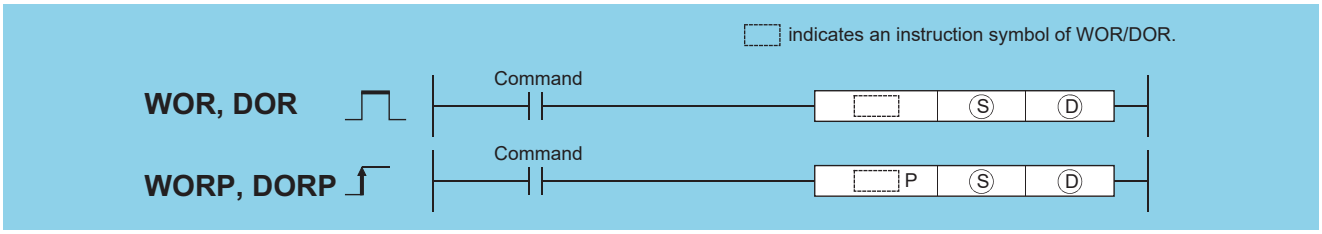
[Operation]



Logical sums of 16-bit data, logical sums of 32-bit data

WOR(P), DOR(P) [When two data are set]

Basic | High performance | Process | Redundant | Universal | LCPU



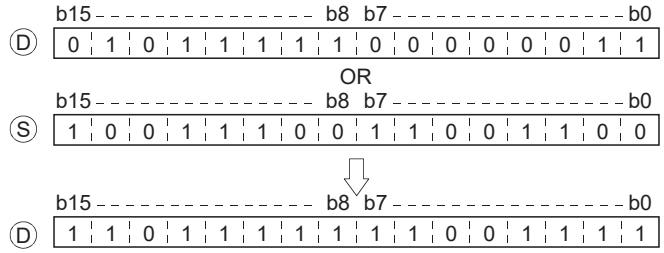
(S): Data for a logical sum operation or head number of the devices where the data is stored (BIN 16/32 bits)
(D): Head number of the devices where the logical sum operation result will be stored (BIN 16/32 bits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(S)	○							○	—
(D)	○							—	—

Processing details

■WOR

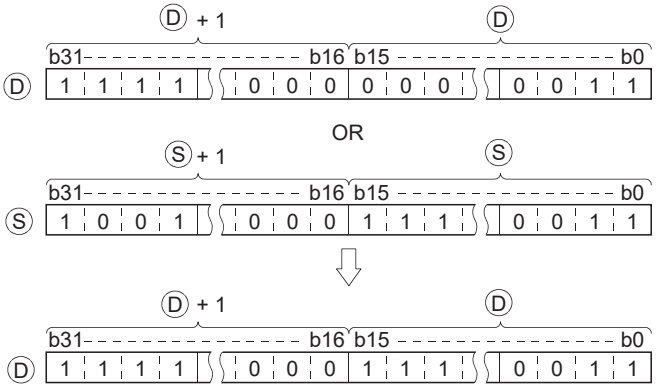
- Conducts a logical sum operation on each bit of the 16-bit data of the device designated by (D) and the 16-bit data of the device designated by (S), and stores the results at the device designated by (D).



- For bit devices, the bit devices after the points designated by digit specification are regarded as "0" in the operation.

■DOR

- Conducts a logical sum operation on each bit of the 32-bit data of the device designated by (D) and the 32-bit data of the device designated by (S), and stores the results at the device designated by (D).



- For bit devices, the bit devices after the points designated by digit specification are regarded as "0" in the operation.

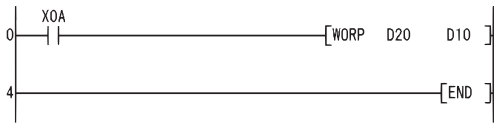
Operation error

- There is no operation error in the WOR(P) or DOR(P) instruction.

Program example

- The following program performs a logical sum operation on the data at D10 and D20 when XA is turned ON, and stores the results at D10.

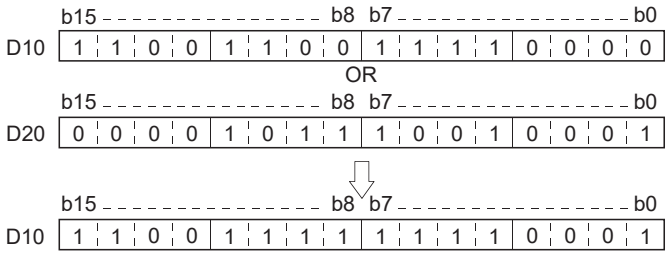
[Ladder Mode]



[List Mode]

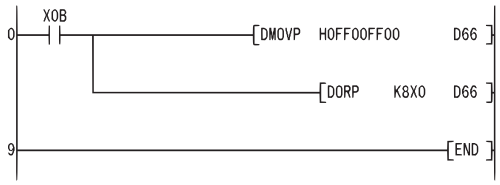
Step	Instruction	Device	
0	LD	XOA	
1	WORP	D20	D10
4	END		

[Operation]



- The following program performs a logical sum operation on the 32-bit data from X0 to X1F, and on the hexadecimal value FF00FF00H when XB is turned ON, and stores the results at D66 and D67.

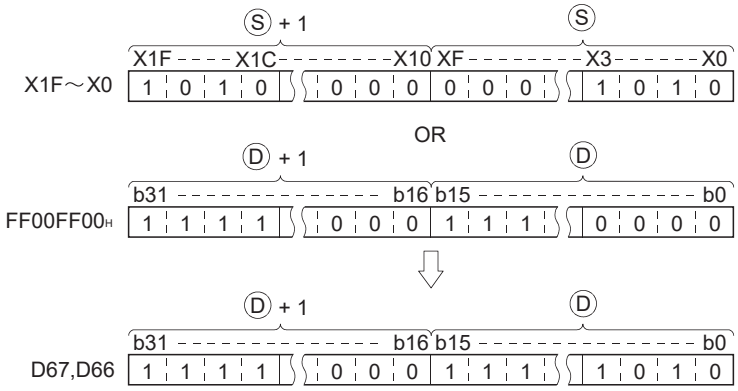
[Ladder Mode]



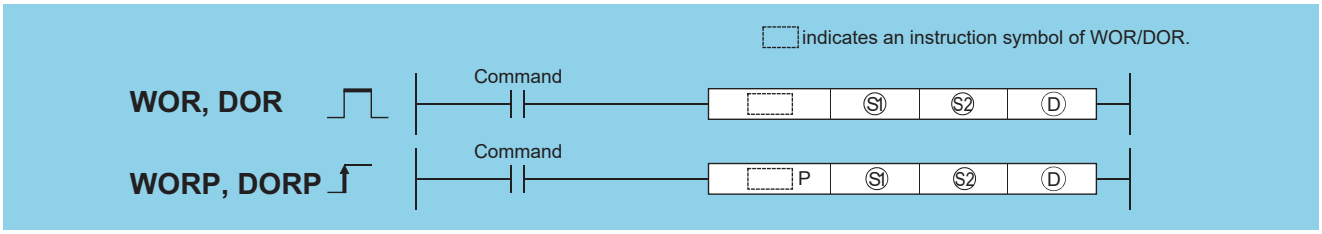
[List Mode]

Step	Instruction	Device	
0	LD	XOB	
1	DMOVP	HOFFOOFFOO	D66
4	DORP	K8XO	D66
9	END		

[Operation]



WOR(P), DOR(P) [When three data are set]



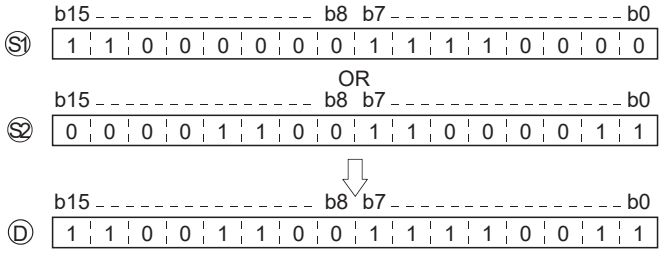
(S1), (S2): Data for a logical sum operation or head number of the devices where the data is stored (BIN 16/32 bits)
(D): Head number of the devices where the logical sum operation result will be stored (BIN 16/32 bits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(S1)	○							○	—
(S2)	○							○	—
(D)	○							—	—

Processing details

■WOR

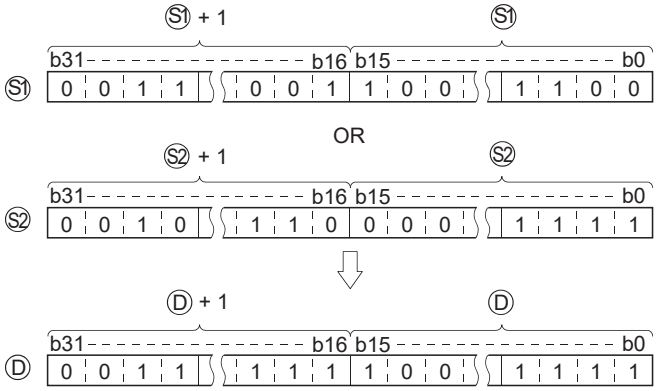
- Conducts a logical sum operation on each bit of the 16-bit data of the device designated by (S1) and the 16-bit data of the device designated by (S2), and stores the results at the device designated by (D).



- For bit devices, the bit devices after the points designated by digit specification are regarded as "0" in the operation.

■DOR

- Conducts a logical sum operation on each bit of the 32-bit data of the device designated by (S1) and the 32-bit data of the device designated by (D).



- For bit devices, the bit devices after the points designated by digit specification are regarded as "0" in the operation.

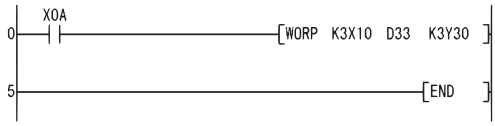
Operation error

- There is no operation error in the WOR(P) or DOR(P) instruction.

Program example

- The following program performs a logical sum operation on the data from X10 to X1B, and the data at D33, and stores the result at Y30 to Y3B when XA is ON.

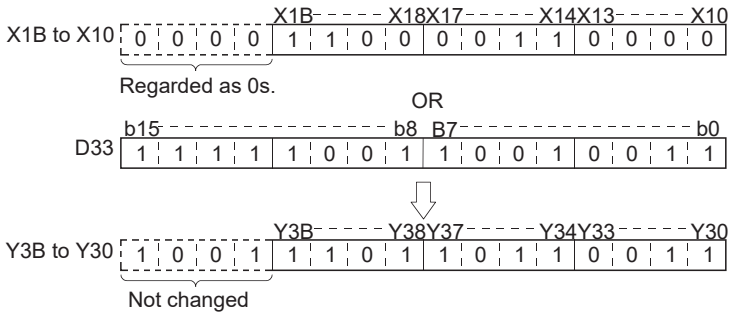
[Ladder Mode]



[List Mode]

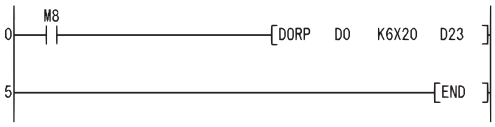
Step	Instruction	Device		
0	LD	XOA		
1	WORP	K3X10	D33	K3Y30
5	END			

[Operation]



- The following program performs a logical sum operation on the 32-bit data at D0 and D1, and the 24-bit data from X20 to X37, and stores the results at D23 and D24 when M8 is ON.

[Ladder Mode]



[List Mode]

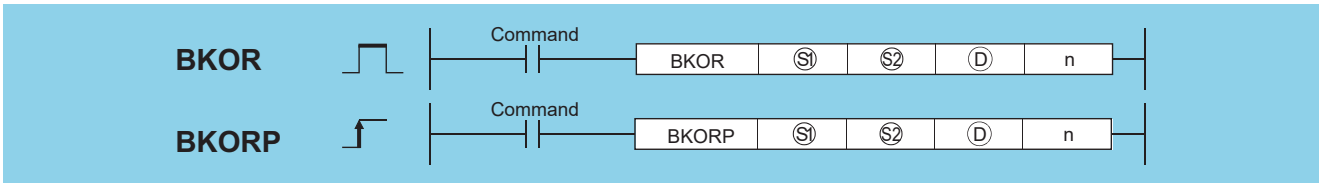
Step	Instruction	Device		
0	LD	M8		
1	DORP	D0	K6X20	D23
5	END			

[Operation]

Block logical sum operations

BKOR(P)

Basic | High performance | Process | Redundant | Universal | LCPU



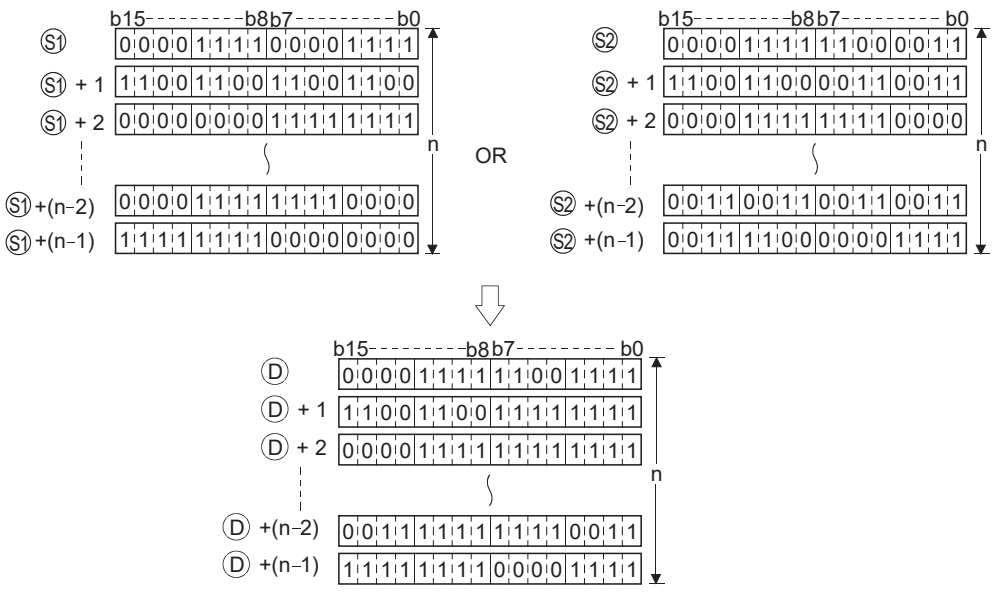
(S1)*1: Head number of the devices where data on which a logical operation will be conducted is stored (BIN 16 bits)
(S2)*1: Data for a logical operation or head number of the devices where the data for the logical operation is stored (BIN 16 bits)
(D)*1: Head number of the devices where the operation result will be stored (BIN 16 bits)
n: Number of operation data blocks (BIN 16 bits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(S1)*1	—	○		—				—	—
(S2)*1	—	○		—				○	—
(D)*1	—	○		—				—	—
n	○	○		○				○	—

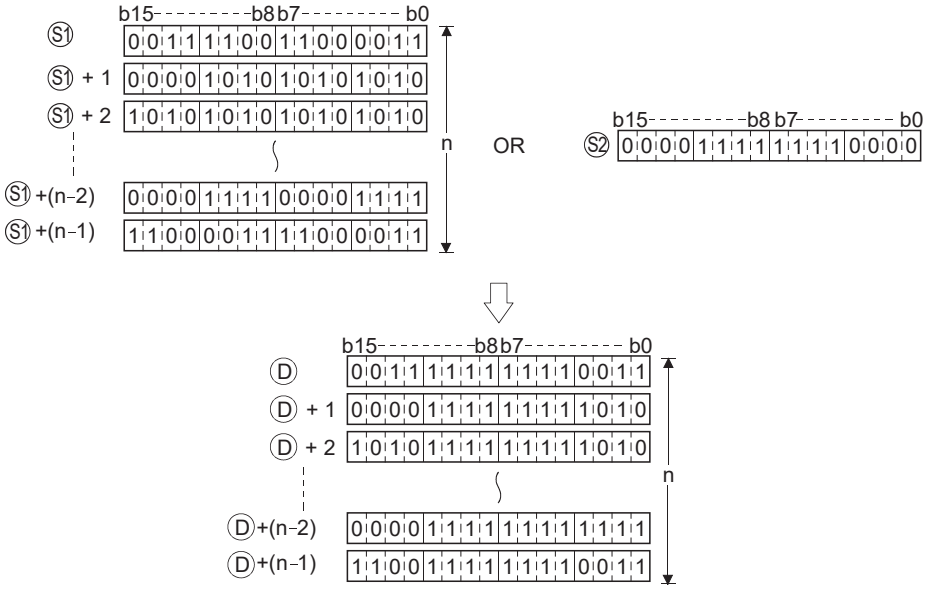
*1 The same device number can be specified for (S1) and (D) or (S2) and (D).

Processing details

- Performs a logical sum operation on the data located in the n points from the device designated by (S1), and the data located in the n points from the device designated by (S2), and stores the results into the area starting from the device designated by (D).



• The constant designated by (S2) can be between -32768 and 32767 (BIN 16-bit data).



Operation error

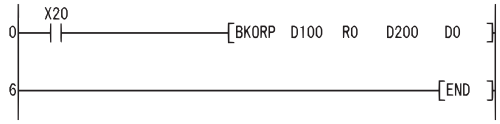
• In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4101	The points specified in n exceed those of the corresponding device specified in (S1), (S2), or (D). The ranges of devices starting from the one specified in (S1) and (D) overlap by n points (except when the same device is specified in (S1) and (D)). The ranges of devices starting from the one specified in (S2) and (D) overlap by n points (except when the same device is specified in (S2) and (D)).	○	○	○	○	○	○

Program example

• The following program performs a logical sum operation on the data stored at D100 to D102 and the data stored at R0 to R2 when X20 is turned ON, and stores the operation result into the area starting from D200.

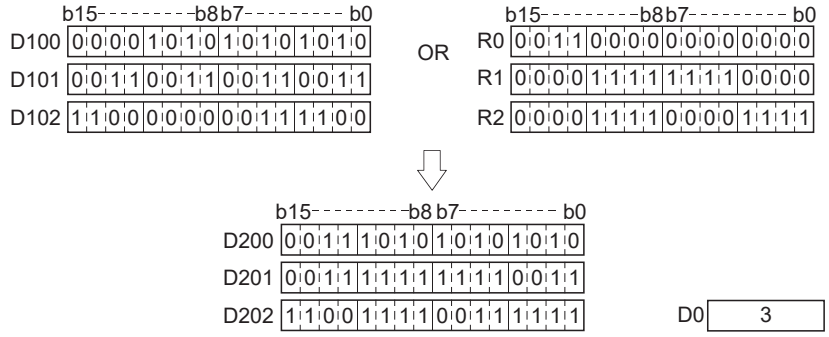
[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	X20
1	BKORP	D100 R0 D200 D0
6	END	

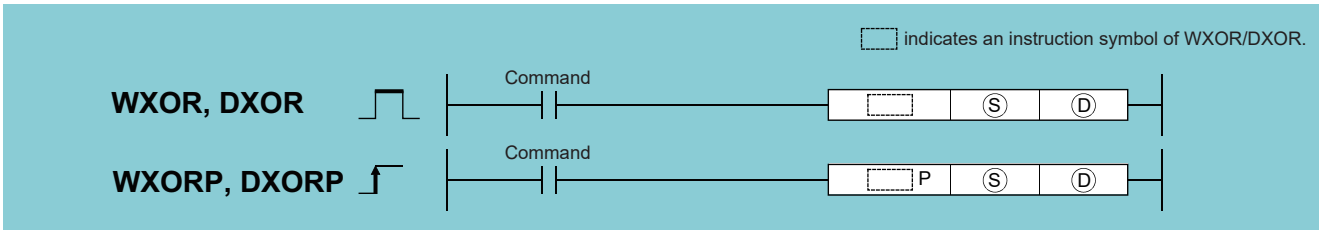
[Operation]



16-bit exclusive OR operations, 32-bit exclusive OR operations

WXOR(P), DXOR(P) [When two data are set]

Basic | High performance | Process | Redundant | Universal | LCPU



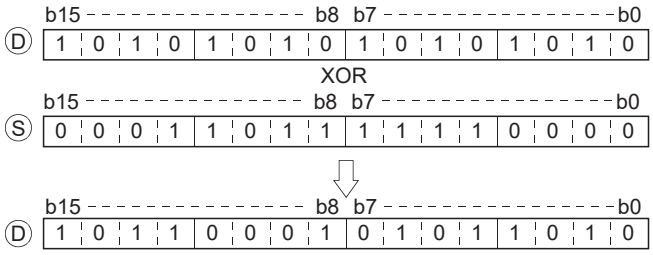
(S): Data for an exclusive OR operation or head number of the devices where the data is stored (BIN 16/32 bits)
(D): Head number of the devices where the exclusive OR operation result will be stored (BIN 16/32 bits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(S)	○							○	—
(D)	○							—	—

Processing details

■WXOR

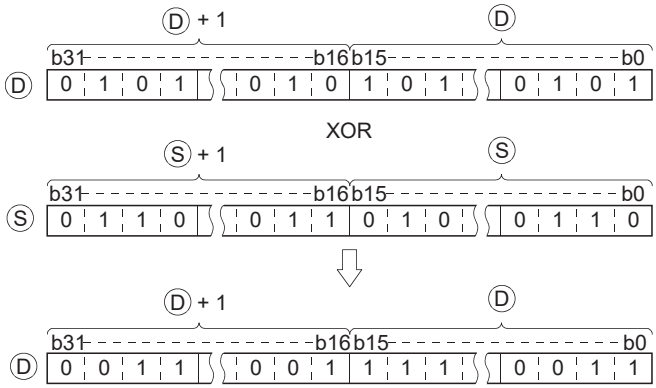
- Conducts an exclusive OR operation on each bit of the 16-bit data of the device designated by (D) and the 16-bit data of the device designated by (S), and stores the results at the device designated by (D).



- For bit devices, the bit devices after the points designated by digit specification are regarded as "0" in the operation.

■DXOR

- Conducts an exclusive OR operation on each bit of the 32-bit data of the device designated by (D) and the 32-bit data of the device designated by (S), and stores the results at the device designated by (D).



- For bit devices, the bit devices after the points designated by digit specification are regarded as "0" in the operation.

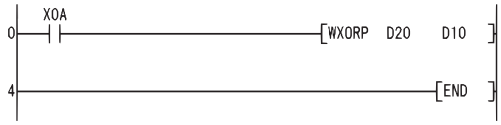
Operation error

- There is no operation error in the WXOR(P) or DXOR(P) instruction.

Program example

- The following program performs an exclusive OR operation on the data at D10 and D20 when XA is ON, and stores the result at D10.

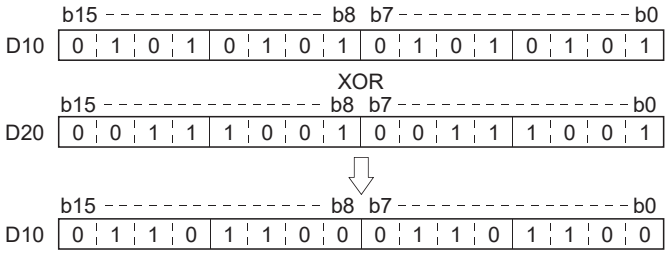
[Ladder Mode]



[List Mode]

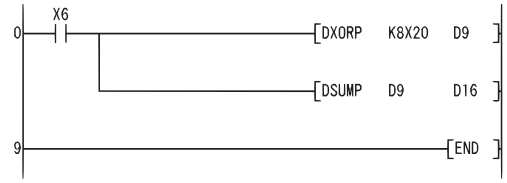
Step	Instruction	Device	
0	LD	XOA	
1	WXORP	D20	D10
4	END		

[Operation]



- The following program compares the bit pattern of the 32-bit data from X20 to X3F with the bit pattern of the data at D9 and D10 when X6 is ON, and stores the number of differing bits at D16. *1

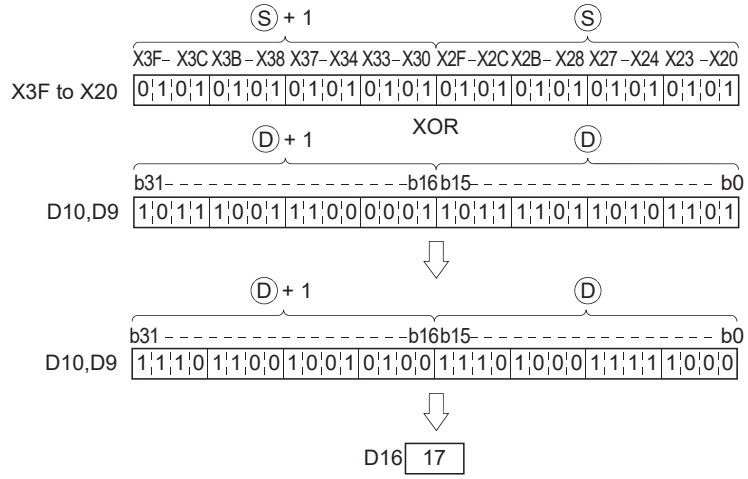
[Ladder Mode]



[List Mode]

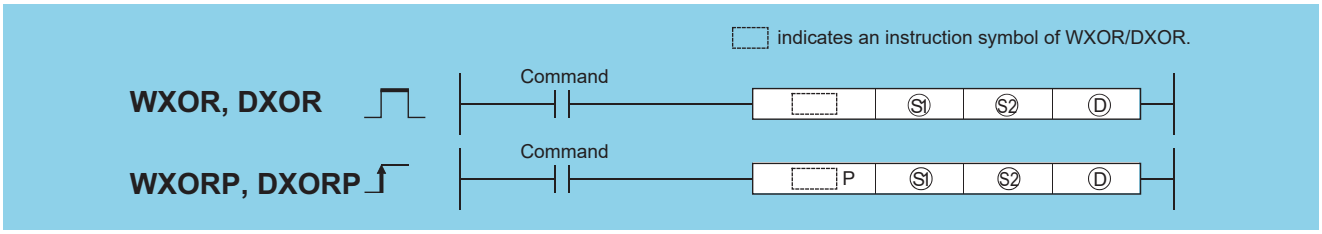
Step	Instruction	Device	
0	LD	X6	
1	DXORP	K8X20	D9
6	DSUMP	D9	D16
9	END		

[Operation]



*1 See Page 423 16-bit data bit check, 32-bit data check for more information on the DSUMP instruction.

WXOR(P), DXOR(P) [When three data are set]



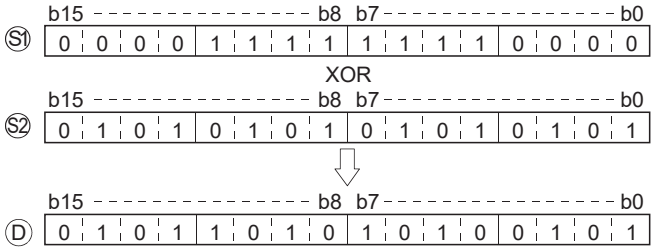
(S1), (S2): Data for an exclusive OR operation or head number of the devices where the data is stored (BIN 16/32 bits)
(D): Head number of the devices where the exclusive OR operation result will be stored (BIN 16/32 bits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(S1)	○							○	—
(S2)	○							○	—
(D)	○							—	—

Processing details

■WXOR

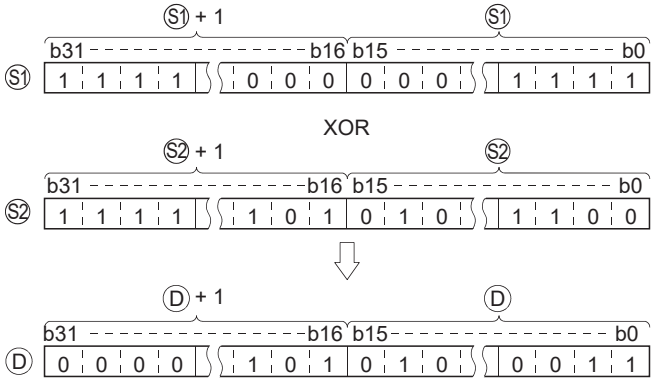
- Conducts an exclusive OR operation on each bit of the 16-bit data of the device designated by (S1) and the 16-bit data of the device designated by (S2), and stores the results at the device designated by (D).



- For bit devices, the bit devices after the points designated by digit specification are regarded as "0" in the operation.

■DXOR

- Conducts an exclusive OR operation on each bit of the 32-bit data of the device designated by (S1) and the 32-bit data of the device designated by (S2), and stores the results at the device designated by (D).



- For bit devices, the bit devices after the points designated by digit specification are regarded as "0" in the operation.

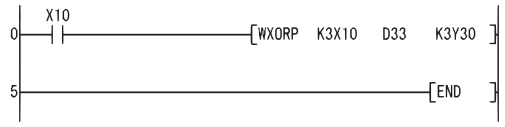
Operation error

- There is no operation error in the WXOR(P) or DXOR(P) instruction.

Program example

- The following program conducts an exclusive OR operation on the data from X10 to X1B and the data at D33 when X10 is ON, and outputs the result to Y30 to Y3B.

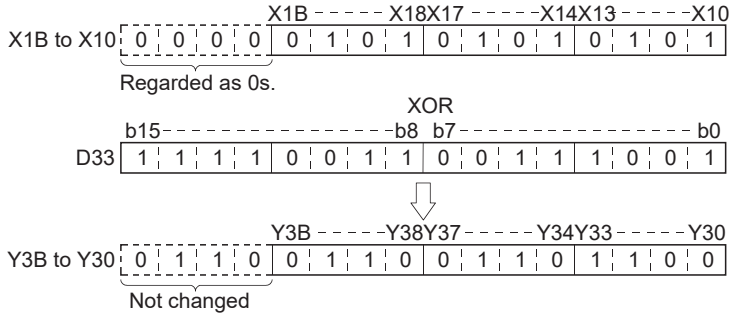
[Ladder Mode]



[List Mode]

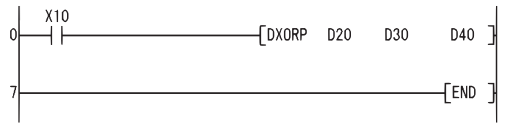
Step	Instruction	Device		
0	LD	X10		
1	WXORP	K3X10	D33	K3Y30
5	END			

[Operation]



- The following program conducts an exclusive OR operation on the data at D20 and D21, and the data at D30 and D31 when X10 is turned ON, and stores the results at D40 and D41.

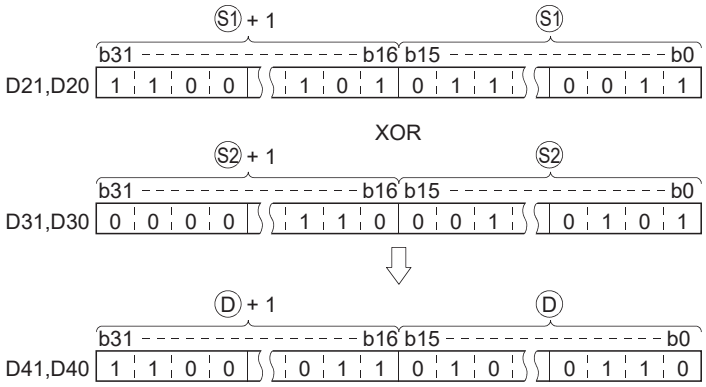
[Ladder Mode]



[List Mode]

Step	Instruction	Device		
0	LD	X10		
1	DXORP	D20	D30	D40
7	END			

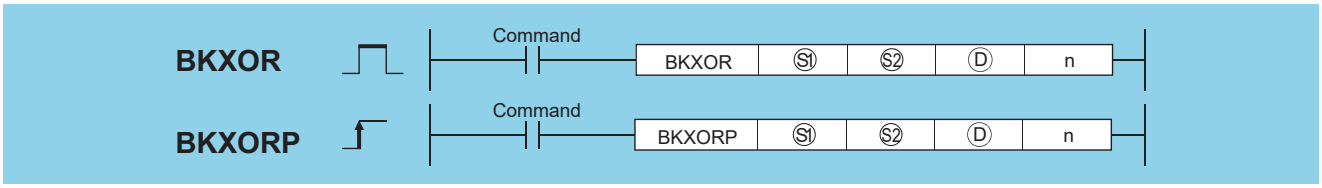
[Operation]



Block exclusive OR operations

BKXOR(P)

Basic | High performance | Process | Redundant | Universal | LCPU



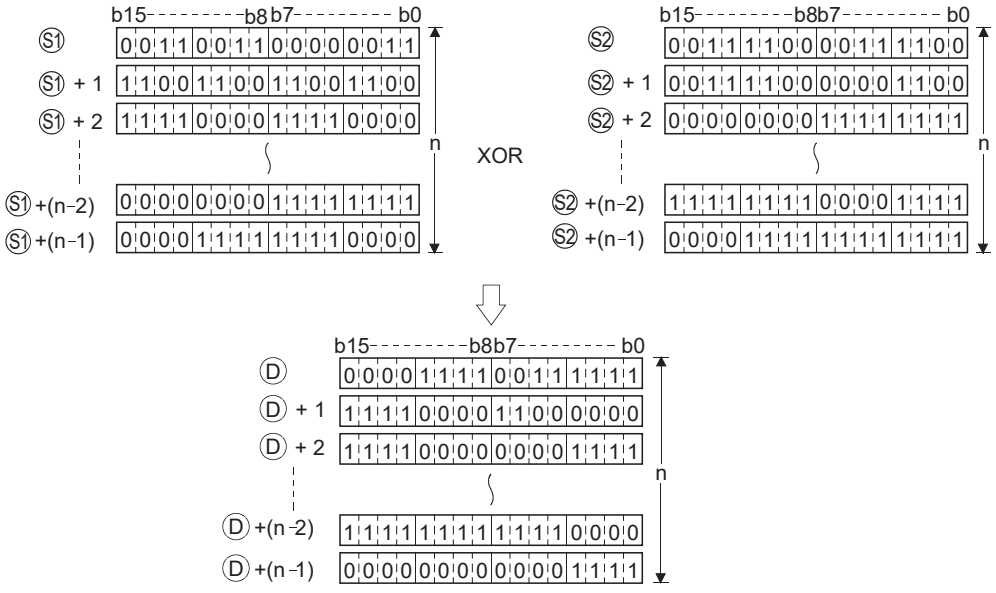
(S1)*1: Head number of the devices where data on which a logical operation will be conducted is stored (BIN 16 bits)
(S2)*1: Data for a logical operation or head number of the devices where the data for the logical operation is stored (BIN 16 bits)
(D)*1: Head number of the devices where the operation result will be stored (BIN 16 bits)
n: Number of operation data blocks (BIN 16 bits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(S1)*1	—	○		—				—	—
(S2)*1	—	○		—				○	—
(D)*1	—	○		—				—	—
n	○	○		○				○	—

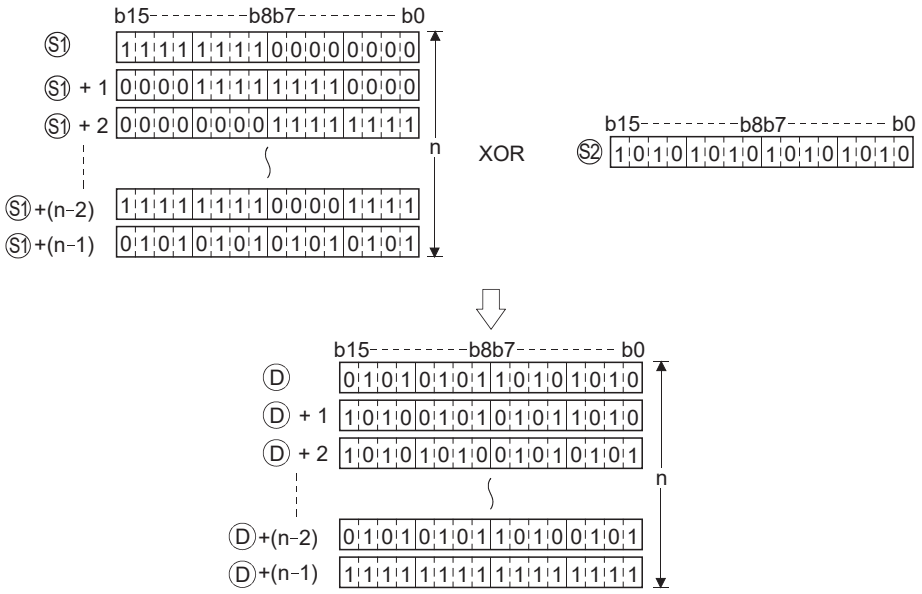
*1 The same device number can be specified for (S1) and (D) or (S2) and (D).

Processing details

- Performs an exclusive OR operation on the data located in the n points from the device designated by (S1), and the data located in the n points from the device designated by (S2), and stores the results into the area starting from the device designated by (D).



• The constant designated by (S2) can be between -32768 and 32767 (BIN 16-bit data).



Operation error

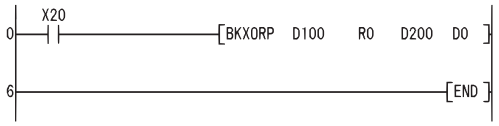
• In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4101	The points specified in n exceed those of the corresponding device specified in (S1), (S2), or (D). The ranges of devices starting from the one specified in (S1) and (D) overlap by n points (except when the same device is specified in (S1) and (D)). The ranges of devices starting from the one specified in (S2) and (D) overlap by n points (except when the same device is specified in (S2) and (D)).	○	○	○	○	○	○

Program example

• The following program performs an exclusive OR operation on the data stored at D100 to D102 and the data stored at R0 to R2 when X20 is turned ON, and stores the operation result into the area starting from D200.

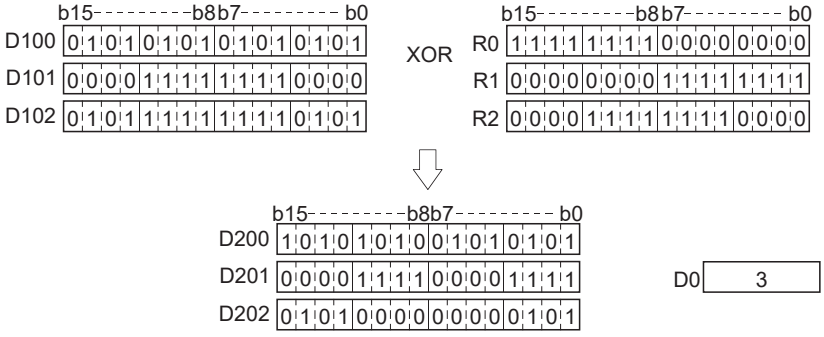
[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	X20
1	BKXORP	D100 R0 D200 D0
6	END	

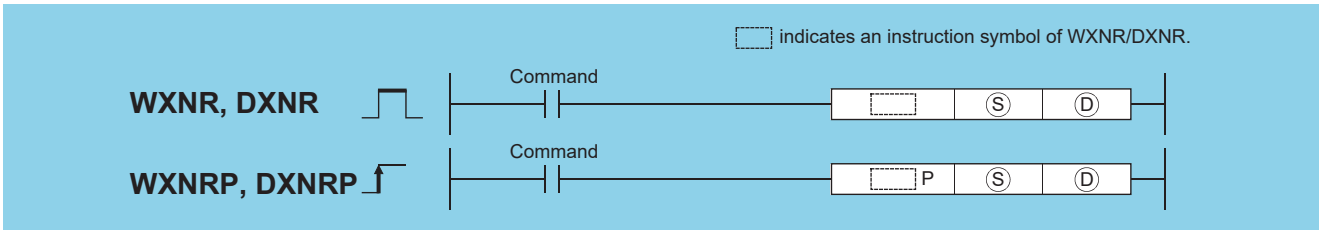
[Operation]



16-bit data exclusive NOR operations, 32-bit data exclusive NOR operations

WXNR(P), DXNR(P) [When two data are set]

Basic | High performance | Process | Redundant | Universal | LCPU



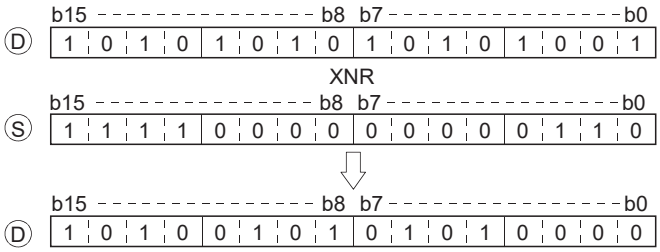
(S): Data for an exclusive NOR operation or head number of the devices where the data is stored (BIN 16/32 bits)
(D): Head number of the devices where the exclusive NOR operation result will be stored (BIN 16/32 bits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(S)	○							○	—
(D)	○							—	—

Processing details

■WXNR

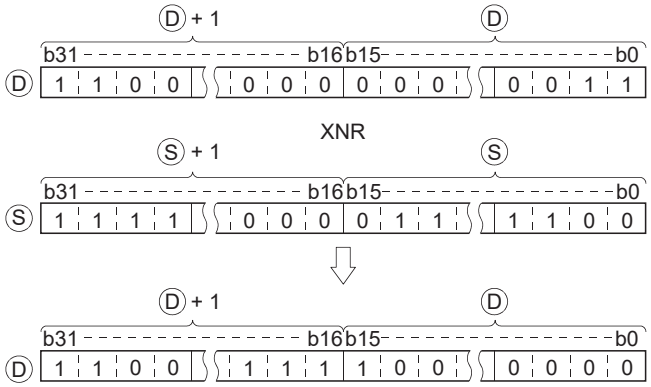
- Conducts an exclusive NOR operation on the 16-bit data of the device designated by (D) and the 16-bit data of the device designated by (S), and stores the results at the device designated by (D).



- For bit devices, the bit devices after the points designated by digit specification are regarded as "0" in the operation.

■DXNR

- Conducts an exclusive NOR operation on the 32-bit data of the device designated by (D) and the 32-bit data of the device designated by (S), and stores the results at the device designated by (D).



- For bit devices, the bit devices after the points designated by digit specification are regarded as "0" in the operation.

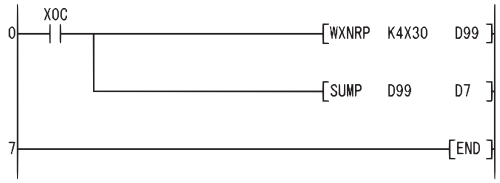
Operation error

- There is no operation error in the WXNR(P) or DXNR(P) instruction.

Program example

- The following program compares the bit patterns of the 16-bit data located from X30 to X3F with the bit patterns of the 16-bit data at D99 when XC is ON, and stores the number of identical bit patterns at D7. *1

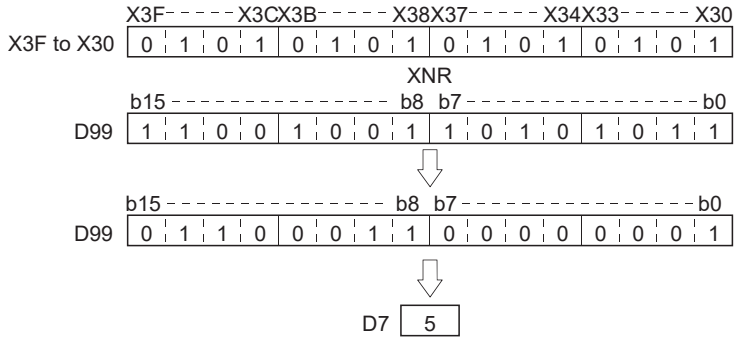
[Ladder Mode]



[List Mode]

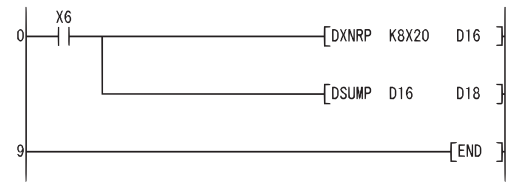
Step	Instruction	Device	
0	LD	X0C	
1	WXNRP	K4X30	D99
4	SUMP	D99	D7
7	END		

[Operation]



- The following program compares the bit patterns of the 32-bit data located from X20 to X3F with the bit patterns of the data at D16 and D17 when X6 is ON, and stores the number of identical bit patterns at D18. *1

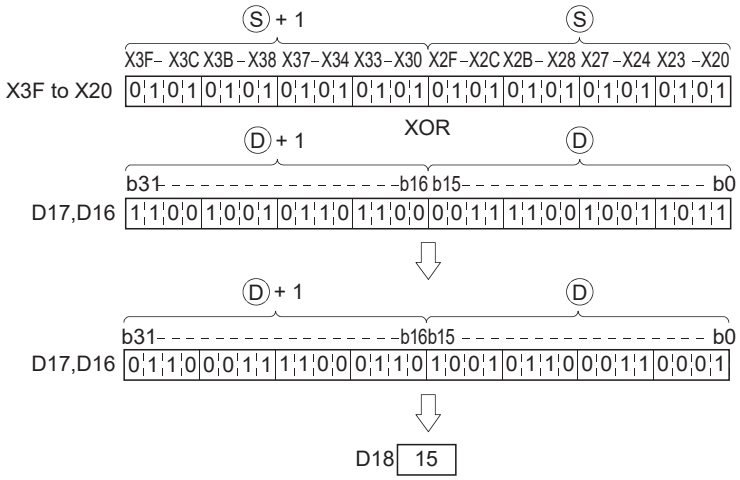
[Ladder Mode]



[List Mode]

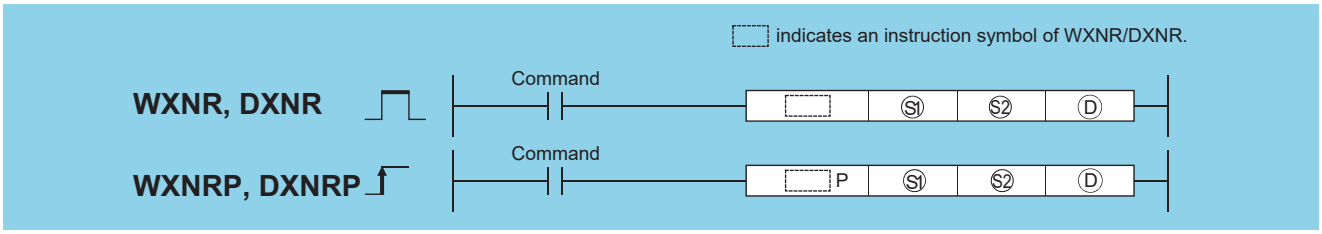
Step	Instruction	Device	
0	LD	X6	
1	DXNRP	K8X20	D16
6	DSUMP	D16	D18
9	END		

[Operation]



*1 See Page 423 16-bit data bit check, 32-bit data check for more information on the SUMP/DSUMP instructions.

WXNR(P), DXNR(P) [When three data are set]



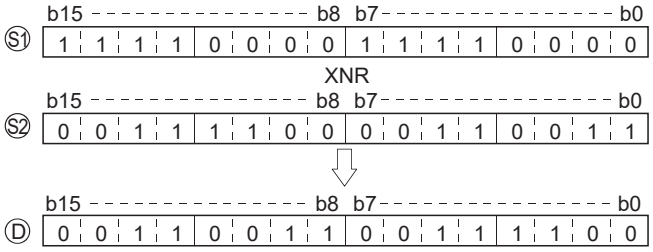
(S1), (S2): Data for an exclusive NOR operation or head number of the devices where the data is stored (BIN 16/32 bits)
(D): Head number of the devices where the exclusive NOR operation result will be stored (BIN 16/32 bits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(S1)	○							○	—
(S2)	○							○	—
(D)	○							—	—

Processing details

■WXNR

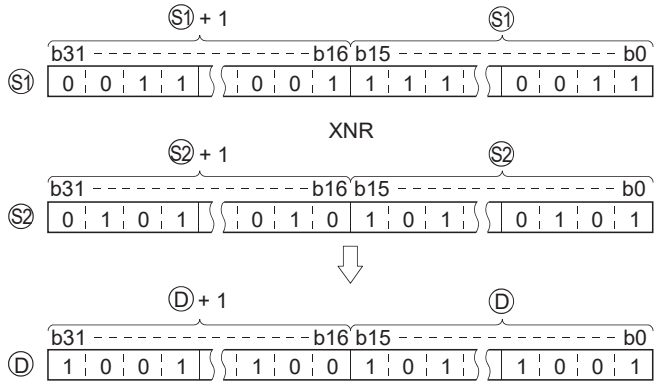
- Conducts an exclusive NOR operation on the 16-bit data of the device designated by (S1) and the 16-bit data of the device designated by (S2), and stores the results at the device designated by (D).



- For bit devices, the bit devices after the points designated by digit specification are regarded as "0" in the operation.

■DXNR

- Conducts an exclusive NOR operation on the 32-bit data of the device designated by (S1) and the 32-bit data of the device designated by (D).



- For bit devices, the bit devices after the points designated by digit specification are regarded as "0" in the operation.

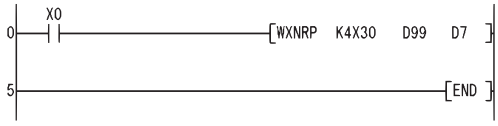
Operation error

- There is no operation error in the WXNR(P) or DXNR(P) instruction.

Program example

- The following program performs an exclusive NOR operation on the 16-bit data from X30 to X3F and the data at D99 when X0 is turned ON, and stores the results to D7.

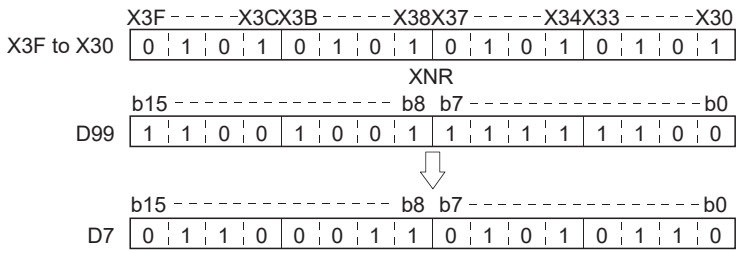
[Ladder Mode]



[List Mode]

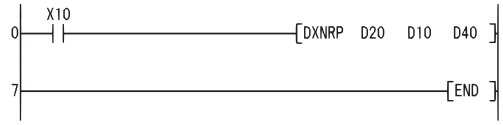
Step	Instruction	Device		
0	LD	X0		
1	WXNRP	K4X30	D99	D7
5	END			

[Operation]



- The following program performs an exclusive NOR operation on the 32-bit data at D20 and D21 and the data at D10 and D11 when X10 is turned ON, and stores the result to D40 and D41.

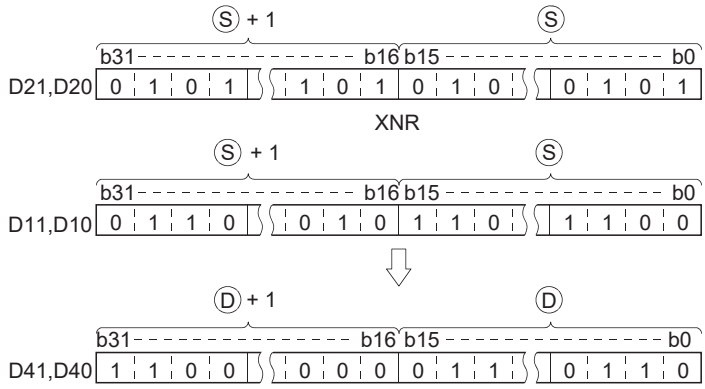
[Ladder Mode]



[List Mode]

Step	Instruction	Device		
0	LD	X10		
1	DXNRP	D20	D10	D40
7	END			

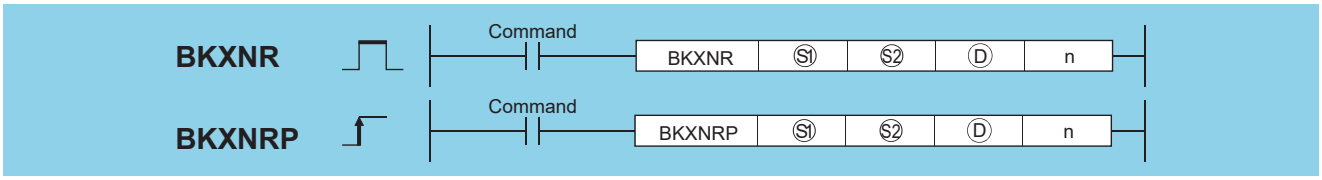
[Operation]



Block exclusive NOR operations

BKXNR(P)

Basic | High performance | Process | Redundant | Universal | LCPU



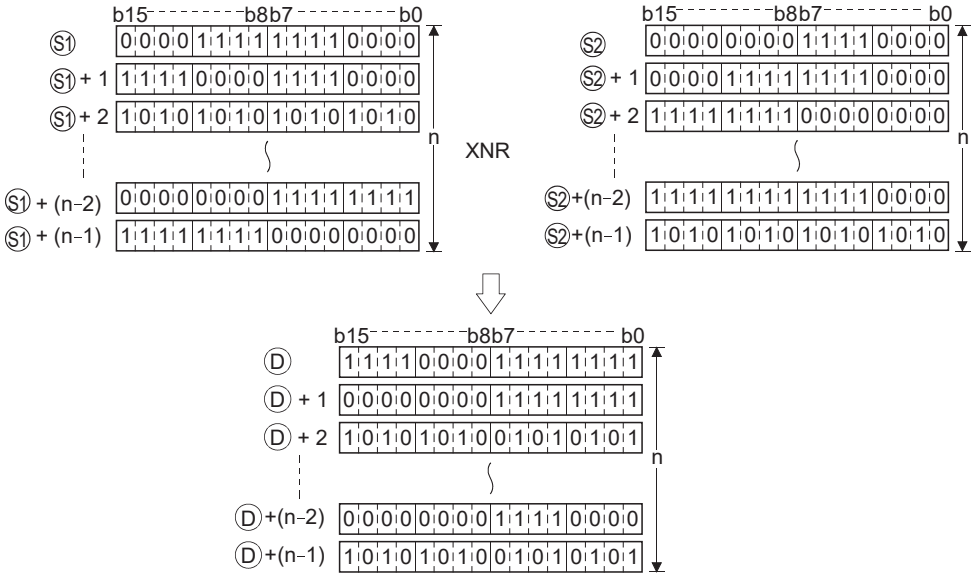
(S1)*1: Head number of the devices where data on which a logical operation will be conducted is stored (BIN 16 bits)
(S2)*1: Data for a logical operation or head number of the devices where the data for the logical operation is stored (BIN 16 bits)
(D)*1: Head number of the devices where the operation result will be stored (BIN 16 bits)
n: Number of operation data blocks (BIN 16 bits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(S1)*1	—	○		—				—	—
(S2)*1	—	○		—				○	—
(D)*1	—	○		—				—	—
n	○	○		○				○	—

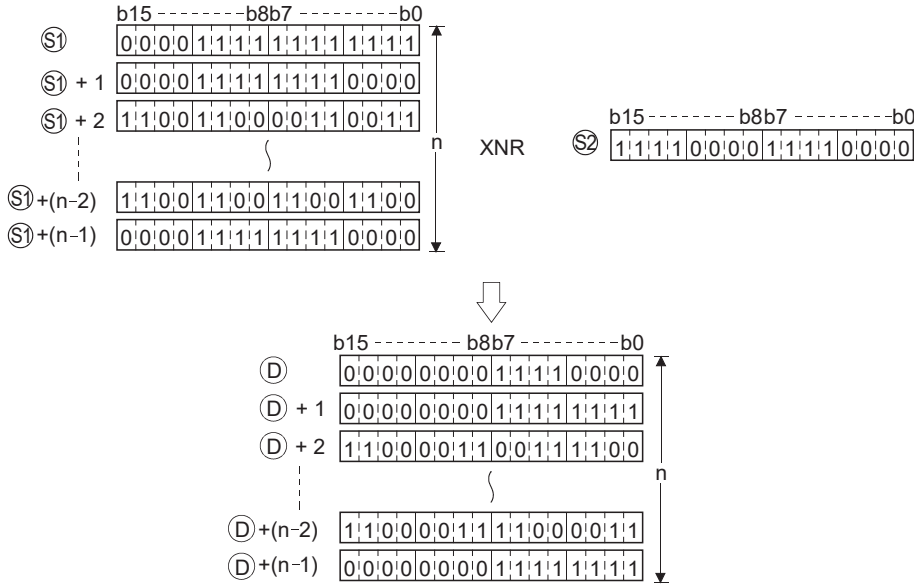
*1 The same device number can be specified for (S1) and (D) or (S2) and (D).

Processing details

- Performs an exclusive NOR operation on the data located in the n points from the device designated by (S1), and the data located in the n points from the device designated by (S2), and stores the results into the area starting from the device designated by (D).



• The constant designated by (S2) can be between -32768 and 32767 (BIN 16-bit data).



Operation error

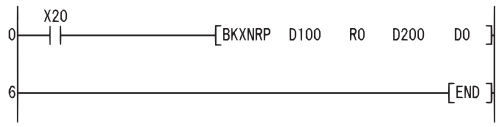
• In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4101	The points specified in n exceed those of the corresponding device specified in (S1), (S2), or (D). The ranges of devices starting from the one specified in (S1) and (D) overlap by n points (except when the same device is specified in (S1) and (D)). The ranges of devices starting from the one specified in (S2) and (D) overlap by n points (except when the same device is specified in (S2) and (D)).	○	○	○	○	○	○

Program example

• The following program performs an exclusive NOR operation on the data stored at D100 to D102 and the data stored at R0 to R2 when X20 is turned ON, and stores the operation result into the area starting from D200.

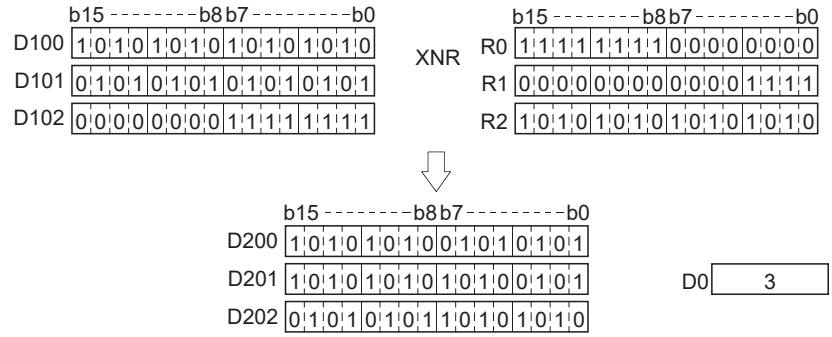
[Ladder Mode]



[List Mode]

Step	Instruction	Device			
0	LD	X20			
1	BKXNRP	D100	R0	D200	D0
6	END				

[Operation]

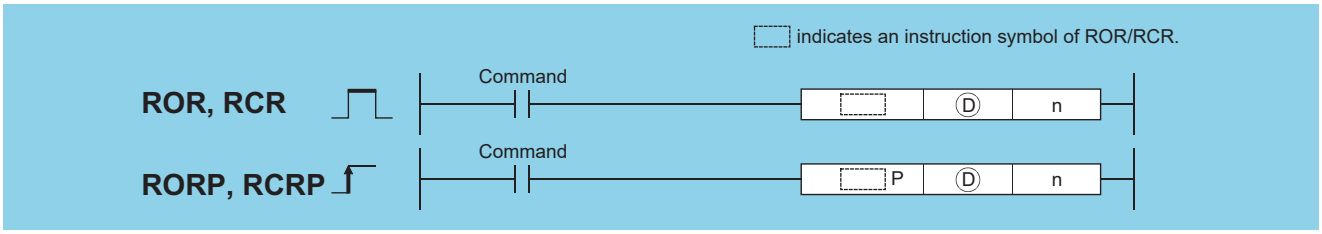


7.2 Rotation Instructions

Right rotation of 16-bit data

ROR(P), RCR(P)

Basic | High performance | Process | Redundant | Universal | LCPU



(D): Head number of the devices to rotate (BIN 16 bits)
n: Number of rotations (0 to 15) (BIN 16 bits)

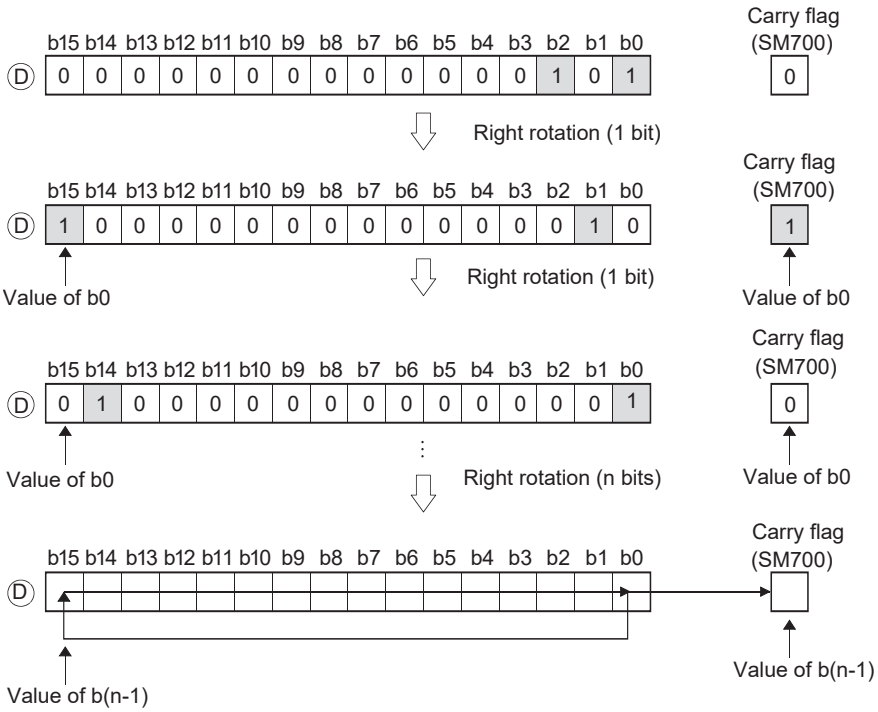
Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(D)	○							—	—
n	○							○	—

7

Processing details

■ROR

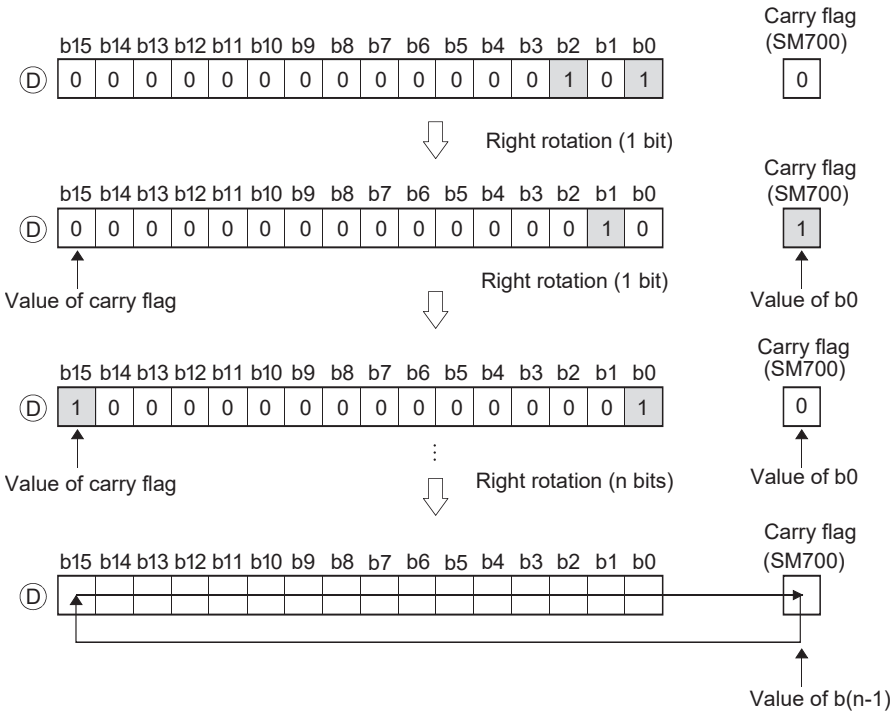
- Rotates 16-bit data of the device designated by (D), not including the carry flag, n-bits to the right. The carry flag is ON or OFF depending on the status prior to the execution of the ROR instruction.



- When a bit device is designated for (D), a rotation is performed within the device range specified by digit specification. The number of bits by which a rotation is executed is the remainder of n / (specified number of bits). For example, when n = 15 and (specified number of bits) = 12 bits, the remainder of 15 / 12 = 1 is "3", and the data is rotated 3 bits.
- Specify any of 0 to 15 as n. If the value specified as n is 16 or greater, the remainder of n / 16 is used for rotation. For example, when n = 18, the contents are rotated two bits to the right since the remainder of 18 / 16 = 1 is "2".

■RCR

- Rotates 16-bit data of the device designated by (D), including the carry flag, n-bits to the right. The carry flag is ON or OFF depending on the status prior to the execution of the ROR instruction.



- When a bit device is designated for (D), a rotation is performed within the device range specified by digit specification. The number of bits by which a rotation is executed is the remainder of n / (specified number of bits). For example, when n = 15 and (specified number of bits) = 12 bits, the remainder of 15 / 12 = 1 is "3", and the data is rotated 3 bits.
- Specify any of 0 to 15 as n. If the value specified as n is 16 or greater, the remainder of n / 16 is used for rotation. For example, when n = 18, the contents are rotated two bits to the right since the remainder of 18 / 16 = 1 is "2".

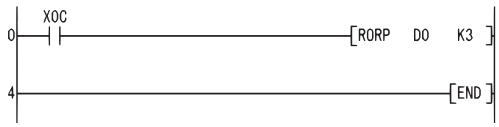
Operation error

- There is no operation error in the ROR(P) or RCR(P) instruction.

Program example

• The following program rotates the contents of D0, not including the carry flag, 3 bits to the right when XC is turned ON.

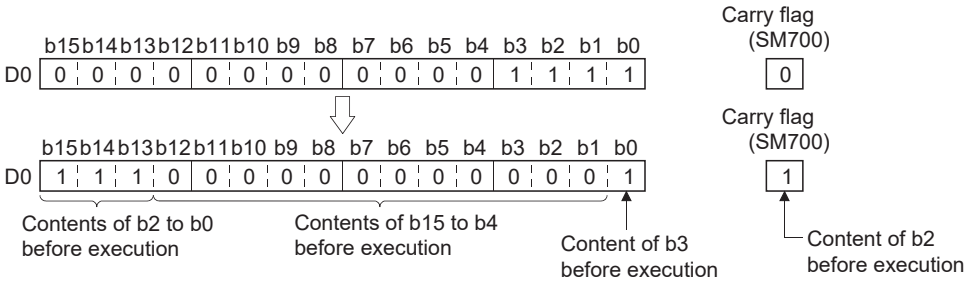
[Ladder Mode]



[List Mode]

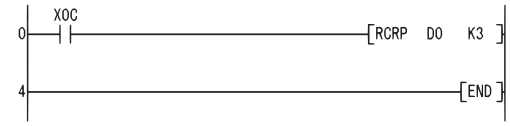
Step	Instruction	Device	
0	LD	XOC	
1	RORP	D0	K3
4	END		

[Operation]



• The following program rotates the contents of D0, including the carry flag, 3 bits to the right when XC is turned ON.

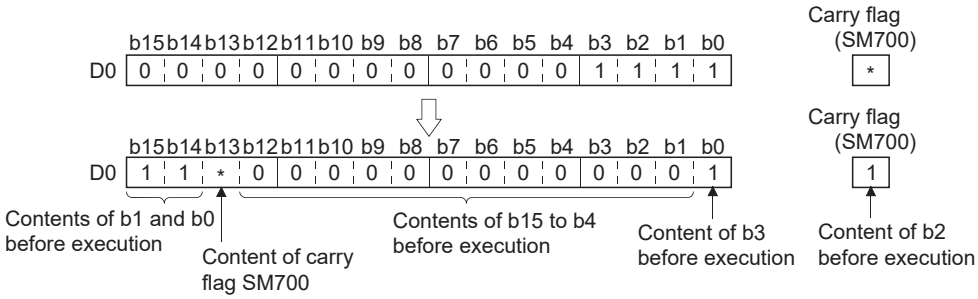
[Ladder Mode]



[List Mode]

Step	Instruction	Device	
0	LD	XOC	
1	RCRP	D0	K3
4	END		

[Operation]

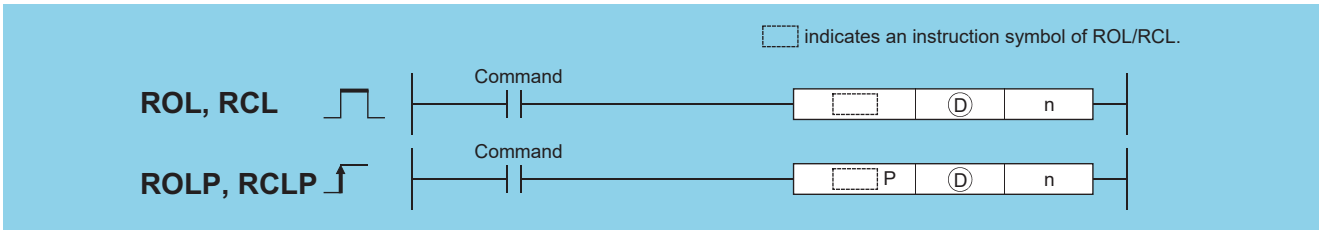


* ON/OFF status of the carry flag depends on its status before the execution of ROR.

Left rotation of 16-bit data

ROL(P), RCL(P)

Basic | High performance | Process | Redundant | Universal | LCPU



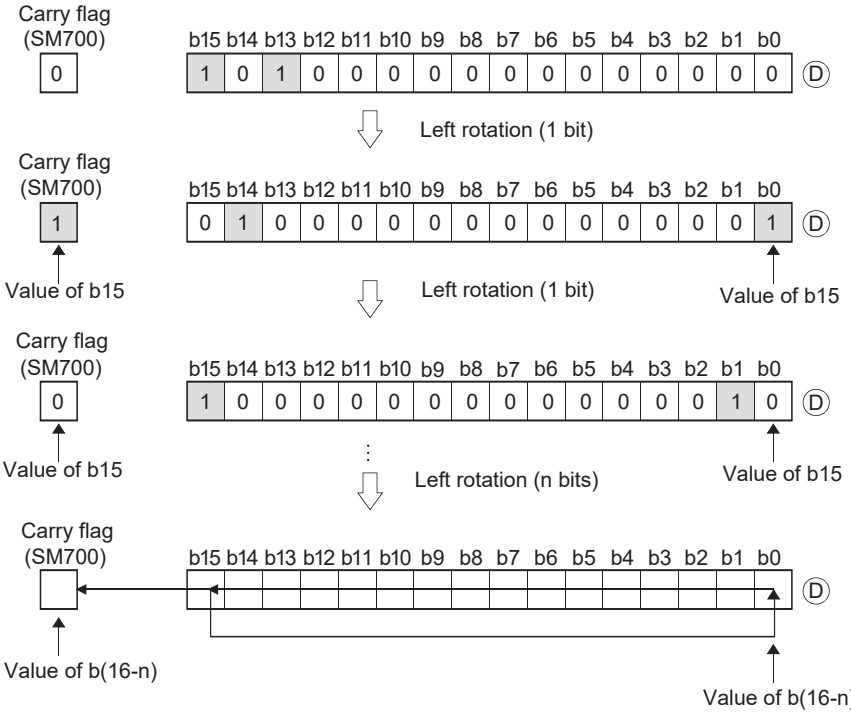
(D): Head number of the devices to rotate (BIN 16 bits)
n: Number of rotations (0 to 15) (BIN 16 bits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(D)	○							—	—
n	○							○	—

Processing details

■ROL

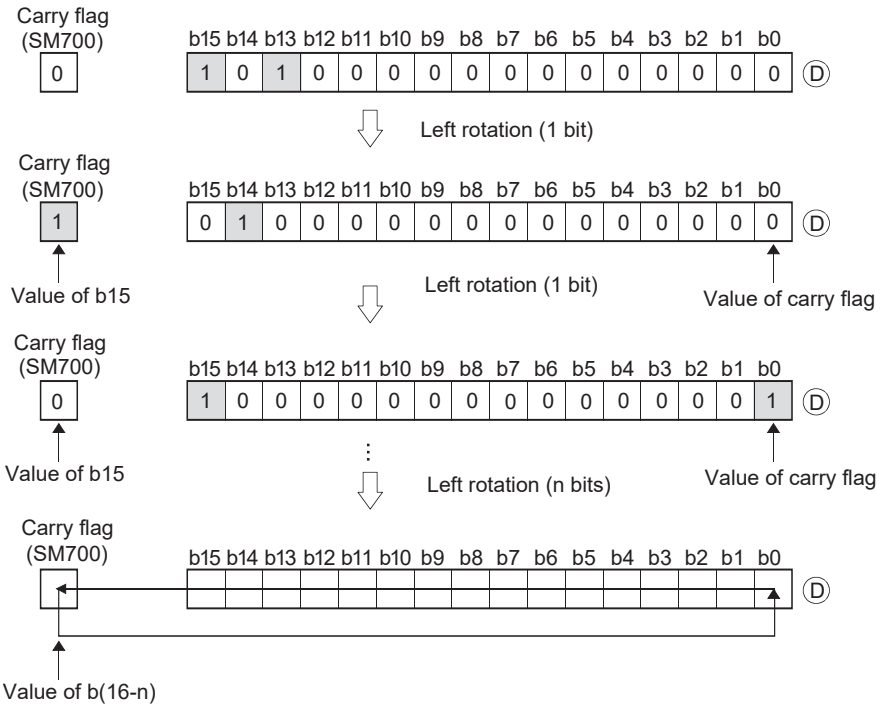
- Rotates the 16-bit data of the device designated at (D), not including the carry flag, n-bits to the left. The carry flag turns ON or OFF depending on its status prior to the execution of ROL instruction.



- When a bit device is designated for (D), a rotation is performed within the device range specified by digit specification. The number of bits by which a rotation is executed is the remainder of n / (specified number of bits). For example, when n = 15 and (specified number of bits) = 12 bits, the remainder of 15 / 12 = 1 is "3", and the data is rotated 3 bits.
- Specify any of 0 to 15 as n. If the value specified as n is 16 or greater, the remainder of n / 16 is used for rotation. For example, when n = 18, the data is rotated 2 bits to the left since the remainder of 18 / 16 = 1 is "2".

■RCL

- Rotates the 16-bit data of the device designated by (D), including the carry flag, n-bits to the left. The carry flag turns ON or OFF depending on its status prior to the execution of RCL instruction.



- When a bit device is designated for (D), a rotation is performed within the device range specified by digit specification. The number of bits by which a rotation is executed is the remainder of n / (specified number of bits). For example, when n = 15 and (specified number of bits) = 12 bits, the remainder of 15 / 12 = 1 is "3", and the data is rotated 3 bits.
- Specify any of 0 to 15 as n. If the value specified as n is 16 or greater, the remainder of n / 16 is used for rotation. For example, when n = 18, the data is rotated 2 bits to the left since the remainder of 18 / 16 = 1 is "2".

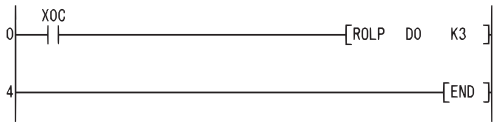
Operation error

- There is no operation error in the ROL(P) or RCL(P) instruction.

Program example

• The following program rotates the contents of D0, not including the carry flag, 3 bits to the left when XC is turned ON.

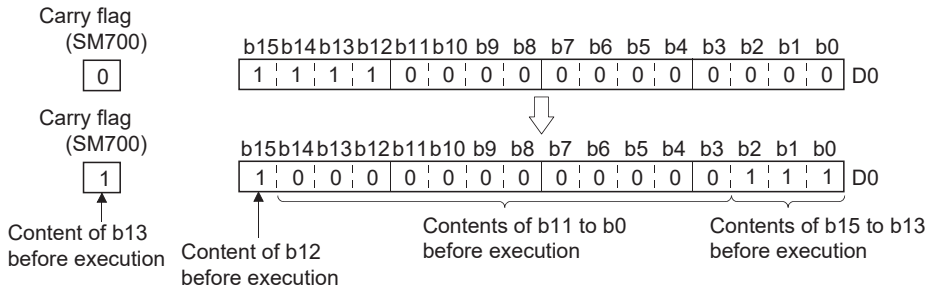
[Ladder Mode]



[List Mode]

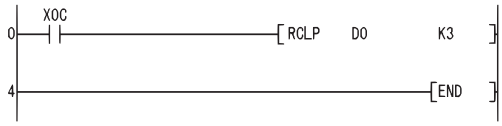
Step	Instruction	Device	
0	LD	XOC	
1	ROLP	D0	K3
4	END		

[Operation]



• The following program rotates the contents of D0, including the carry flag, 3 bits to the left when XC is turned ON.

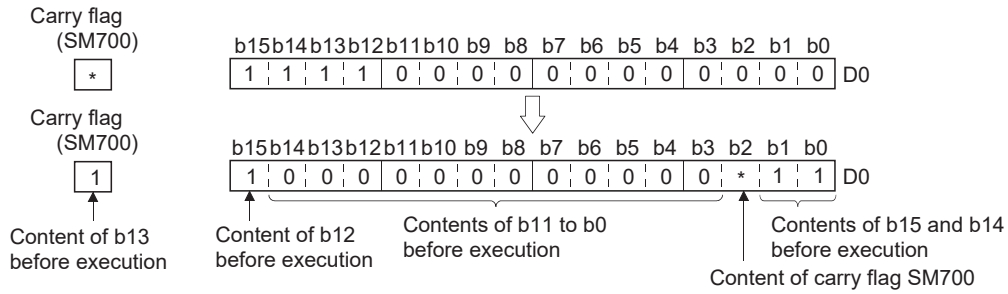
[Ladder Mode]



[List Mode]

Step	Instruction	Device	
0	LD	XOC	
1	RCLP	D0	K3
4	END		

[Operation]

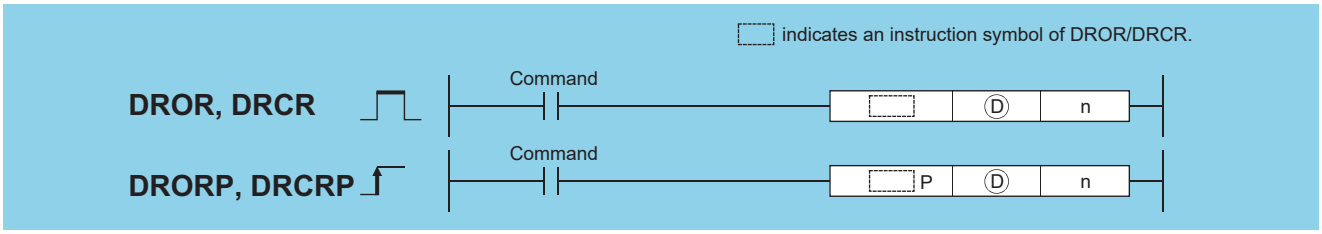


* ON/OFF status of the carry flag depends on its status before the execution of RCL.

Right rotation of 32-bit data

DROR(P), DRCR(P)

Basic | High performance | Process | Redundant | Universal | LCPU



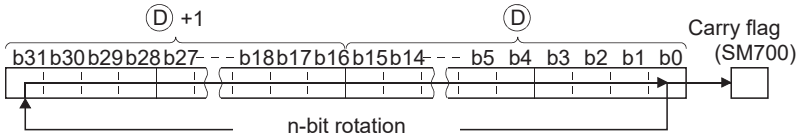
(D): Head number of the devices to rotate (BIN 32 bits)
n: Number of rotations (0 to 31) (BIN 16 bits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(D)	○							—	—
n	○							○	—

Processing details

■DROR

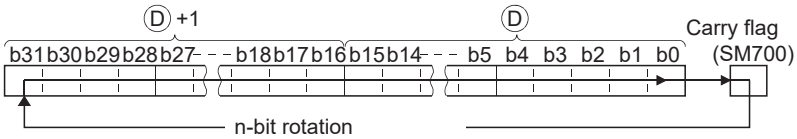
- The 32-bit data of the device designated at (D), not including the carry flag, is rotated n-bits to the right. The carry flag turns ON or OFF depending on its status prior to the execution of the DROR instruction.



- When a bit device is designated for (D), a rotation is performed within the device range specified by digit specification. The number of bits by which a rotation is executed is the remainder of n / (specified number of bits). For example, when n = 31 and (specified number of bits) = 24 bits, the remainder of 31 / 24 = 1 is "7", and the data is rotated 7 bits.
- Specify any of 0 to 31 as n. If the value specified as n is 32 or greater, the remainder of n / 32 is used for rotation. For example, when n = 34, the contents are rotated two bits to the right since the remainder of 34 / 32 = 1 is "2".

■DRCR

- Rotates 32-bit data of the device designated by (D), including the carry flag, n-bits to the right. The carry flag goes ON or OFF depending on its status prior to the execution of the DRCR instruction.



- When a bit device is designated for (D), a rotation is performed within the device range specified by digit specification. The number of bits by which a rotation is executed is the remainder of n / (specified number of bits). For example, when n = 31 and (specified number of bits) = 24 bits, the remainder of 31 / 24 = 1 is "7", and the data is rotated 7 bits.
- Specify any of 0 to 31 as n. If the value specified as n is 32 or greater, the remainder of n / 32 is used for rotation. For example, when n = 34, the contents are rotated two bits to the right since the remainder of 34 / 32 = 1 is "2".

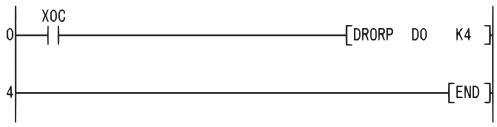
Operation error

- There is no operation error in the DROR(P) or DRCR(P) instruction.

Program example

- The following program rotates the contents of D0 and D1, not including the carry flag, 4 bits to the right when XC is ON.

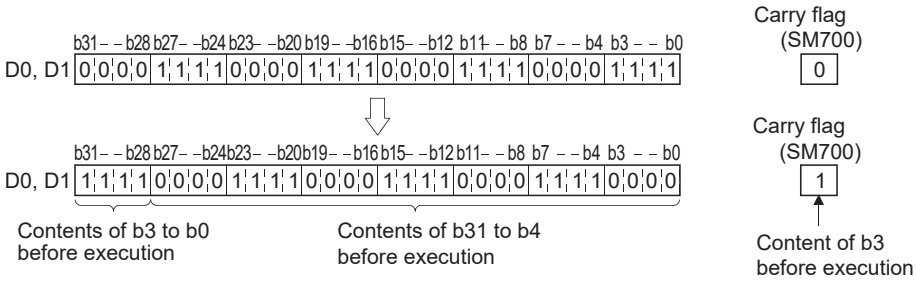
[Ladder Mode]



[List Mode]

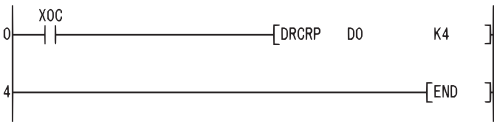
Step	Instruction	Device	
0	LD	XOC	
1	DRORP	D0	K4
4	END		

[Operation]



- The following program rotates the contents of D0 and D1, including the carry flag, 4 bits to the right when XC is ON.

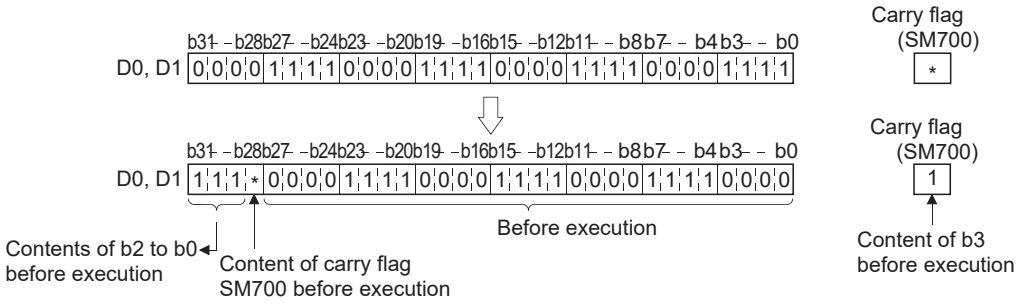
[Ladder Mode]



[List Mode]

Step	Instruction	Device	
0	LD	XOC	
1	DRCRP	D0	K4
4	END		

[Operation]

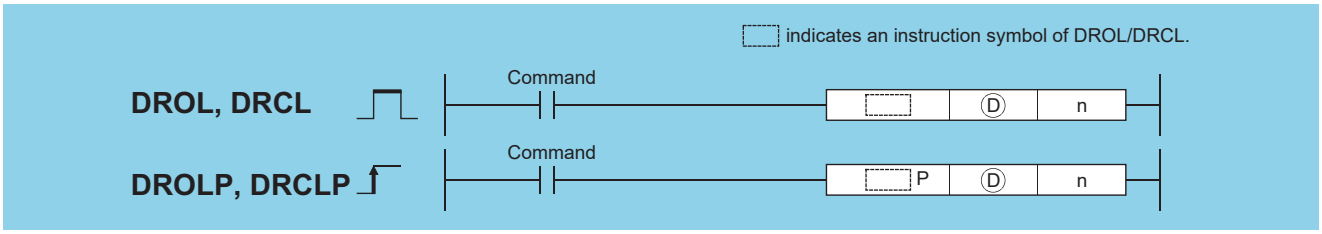


* : ON/OFF status of the carry flag depends on its status before the execution of DRCR.

Left rotation of 32-bit data

DROL(P), DRCL(P)

Basic | High performance | Process | Redundant | Universal | LCPU



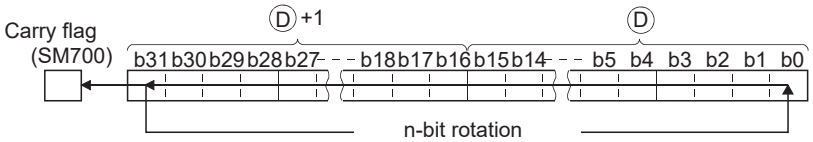
(D): Head number of the devices to rotate (BIN 32 bits)
n: Number of rotations (0 to 31) (BIN 16 bits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(D)	○							—	—
n	○							○	—

Processing details

■DROL

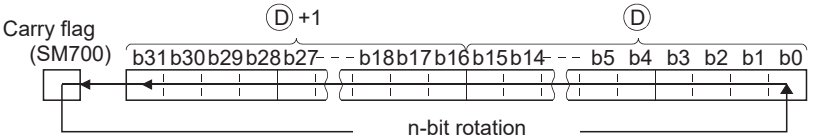
- The 32-bit data of the device designated at (D), not including the carry flag, is rotated n-bits to the left. The carry flag turns ON or OFF depending on its status prior to the execution of the DROL instruction.



- When a bit device is designated for (D), a rotation is performed within the device range specified by digit specification. The number of bits by which a rotation is executed is the remainder of n / (specified number of bits). For example, when n = 31 and (specified number of bits) = 24 bits, the remainder of 31 / 24 = 1 is "7", and the data is rotated 7 bits.
- Specify any of 0 to 31 as n. If the value specified as n is 32 or greater, the remainder of n / 32 is used for rotation. For example, when n = 34, the data is rotated 2 bits to the left since the remainder of 34 / 32 = 1 is "2".

■DRCL

- Rotates 32-bit data of the device designated by (D), including the carry flag, n-bits to the left. The carry flag turns ON or OFF depending on its status prior to the execution of the DRCL instruction.



- When a bit device is designated for (D), a rotation is performed within the device range specified by digit specification. The number of bits by which a rotation is executed is the remainder of n / (specified number of bits). For example, when n = 31 and (specified number of bits) = 24 bits, the remainder of 31 / 24 = 1 is "7", and the data is rotated 7 bits.
- Specify any of 0 to 31 as n. If the value specified as n is 32 or greater, the remainder of n / 32 is used for rotation. For example, when n = 34, the data is rotated 2 bits to the left since the remainder of 34 / 32 = 1 is "2".

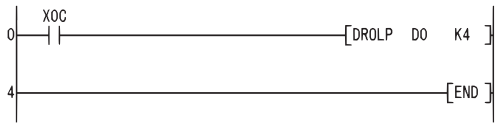
Operation error

- There is no operation error in the DROL(P) or DRCL(P) instruction.

Program example

- The following program rotates the contents of D0 and D1, not including the carry flag, 4 bits to the left when XC is ON.

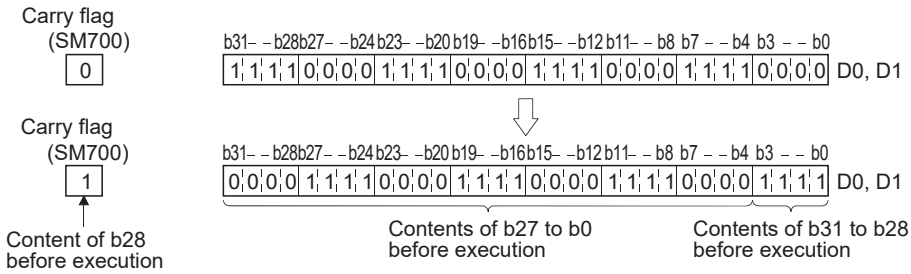
[Ladder Mode]



[List Mode]

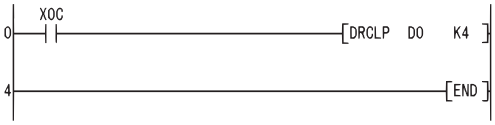
Step	Instruction	Device	
0	LD	XOC	
1	DROLP	D0	K4
4	END		

[Operation]



- The following program rotates the contents of D0 and D1, including the carry flag, 4 bits to the left when XC is ON.

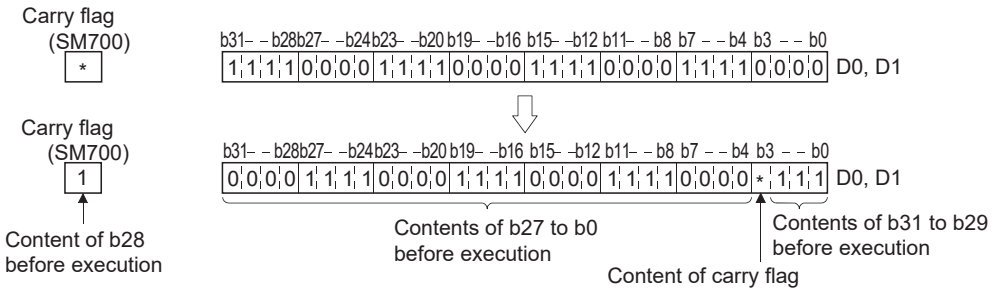
[Ladder Mode]



[List Mode]

Step	Instruction	Device	
0	LD	XOC	
1	DRCLP	D0	K4
4	END		

[Operation]



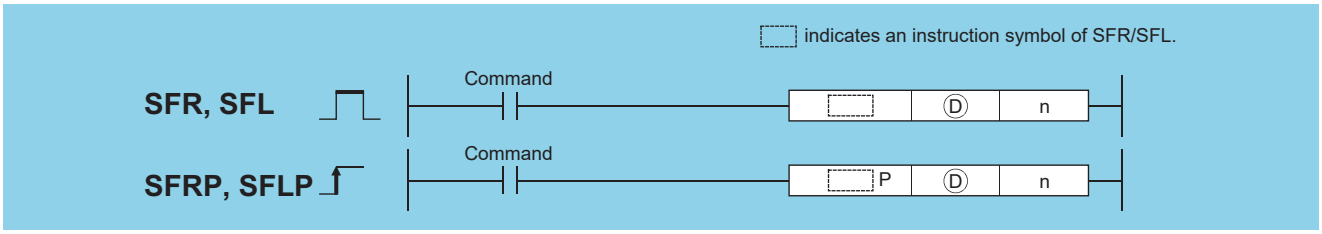
* : ON/OFF status of the carry flag depends on its status before the execution of DRCL.

7.3 Shift Instructions

n-bit shift to right of 16-bit data, n-bit shift to left of 16-bit data

SFR(P), SFL(P)

Basic | High performance | Process | Redundant | Universal | LCPU



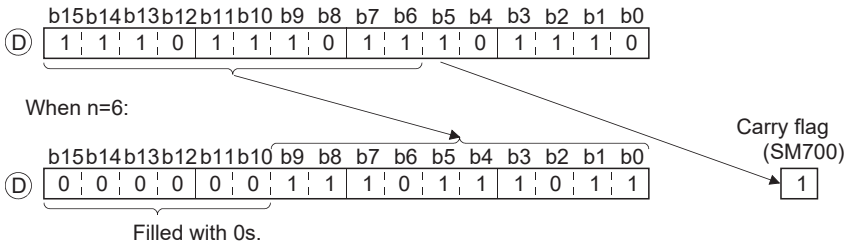
(D): Head number of the devices where shift data is stored (BIN 16 bits)
n: Number of shifts (0 to 15) (BIN 16 bits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(D)	○							—	—
n	○							○	—

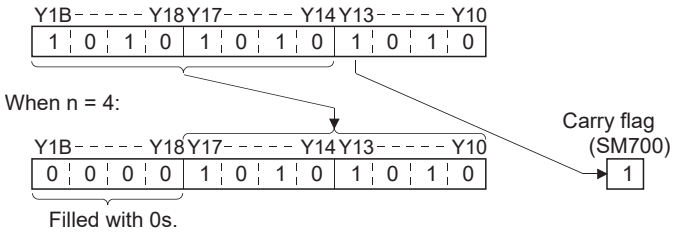
Processing details

■SFR

- Causes a shift to the right by n bits of the 16-bit data from the device designated at (D). The n bits from the upper bit are filled with 0s.



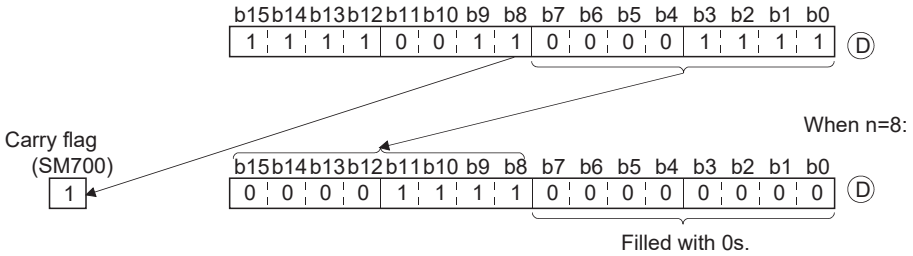
- When a bit device is designated for (D), a right shift is executed within the device range specified by digit specification. The number of bits by which a shift is executed is the remainder of n / (specified number of bits). For example, when n = 15 and (specified number of bits) = 8 bits, the remainder of 15 / 8 = 1 is "7", and the data is shifted 7 bits.



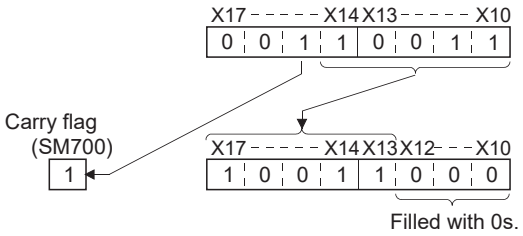
- Specify any of 0 to 15 as n. If the value specified as n is 16 or greater, the remainder of n / 16 is used for a shift to the right. For example, when n = 18, the data is shifted 2 bits to the right since the remainder of 18 / 16 =1 is 2.

■SFL

- Shifts 16-bit data at device designated by (D) n bits to the left. Bits starting from the lowest bit to n bit are filled with 0s.



- When a bit device is designated for (D), a left shift is executed within the device range specified by digit specification. The number of bits by which a shift is executed is the remainder of n / (specified number of bits). For example, when n = 15 and (specified number of bits) = 8 bits, the remainder of 15 / 8 = 1 is "7", and the data is shifted 7 bits.



- Specify any of 0 to 15 as n. If the value specified as n is 16 or greater, the remainder of n / 16 is used for a shift to the left. For example, when n = 18, the data is shifted 2 bits to the left since the remainder of 18 / 16 = 1 is "2".

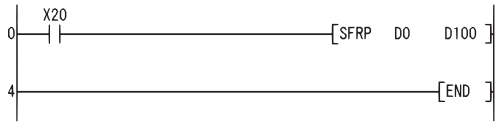
Operation error

- There is no operation error in the SFR(P) or SFL(P) instruction.

Program example

• The following program shifts the data of D0 to the right by the number of bits designated by D100 when X20 is turned ON.

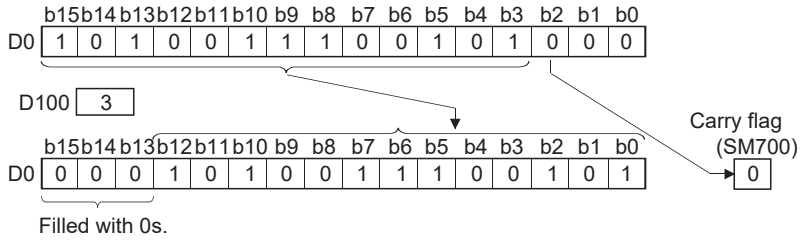
[Ladder Mode]



[List Mode]

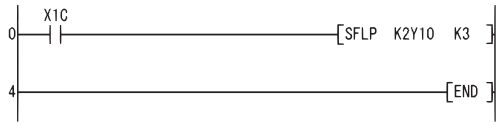
Step	Instruction	Device	
0	LD	X20	
1	SFRP	D0	D100
4	END		

[Operation]



• The following program shifts the contents of X10 to X17 3 bits to the left when X1C is ON.

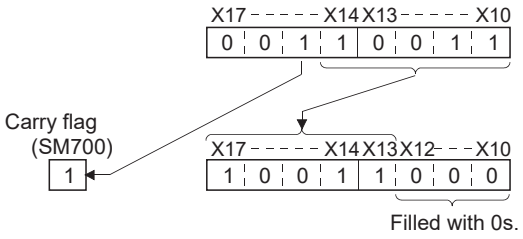
[Ladder Mode]



[List Mode]

Step	Instruction	Device	
0	LD	X1C	
1	SFLP	K2Y10	K3
4	END		

[Operation]

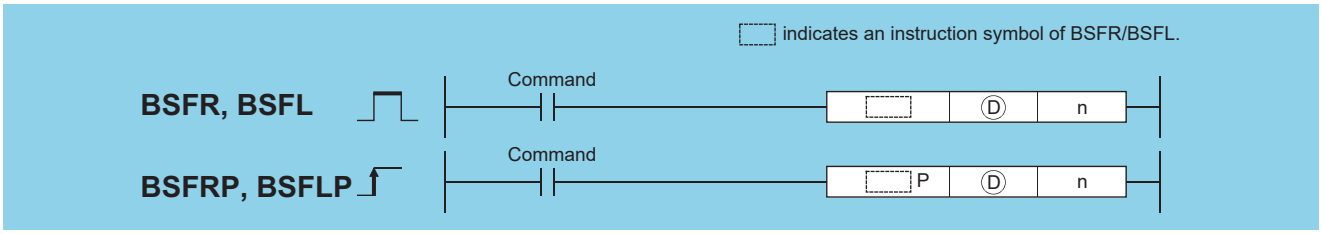


7

1-bit shift to right of n-bit data, 1-bit shift to left of n-bit data

BSFR(P), BSFL(P)

Basic | High performance | Process | Redundant | Universal | LCPU



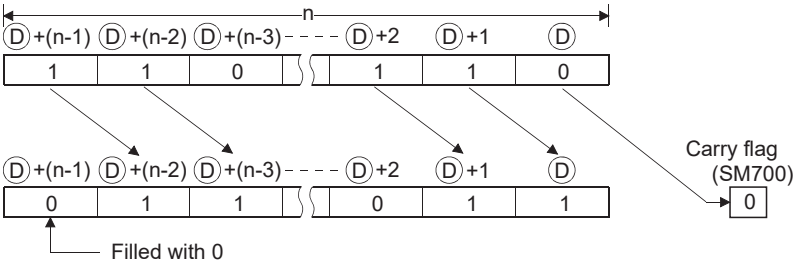
(D): Head number of the devices to be shifted (bits)
n: Number of devices to which shift is executed (BIN 16 bits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(D)	○		—						—
n	○		○						—

Processing details

■BSFR

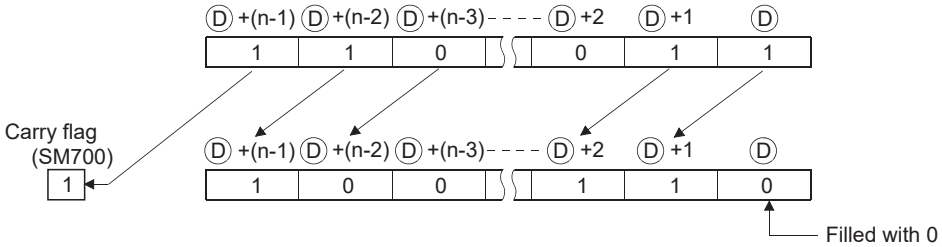
• Shifts the data in n points from the device designated by (D) to the right by one bit.



• The device designated by (D)+(n-1) is filled with 0.

■BSFL

• Shifts the data in n points from the device designated by (D) to the left by one bit.



• The device designated by (D) is filled with 0.

Operation error

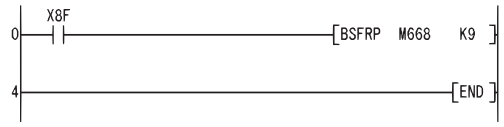
• In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4101	The points specified in n exceed those of the corresponding device specified in (D).	○	○	○	○	○	○

Program example

• The following program shifts the data at M668 to M676 to the right when X8F is turned ON.

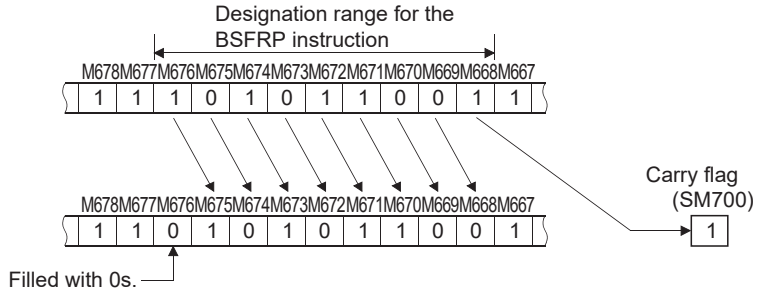
[Ladder Mode]



[List Mode]

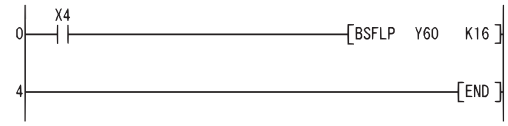
Step	Instruction	Device	
0	LD	X8F	
1	BSFRP	M668	K9
4	END		

[Operation]



• The following program shifts the data at Y60 to Y6F to the left when X4 is turned ON.

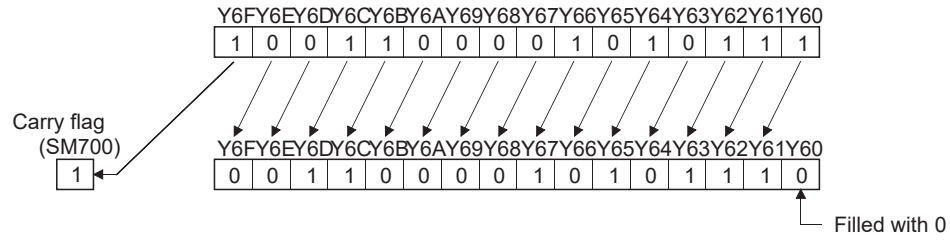
[Ladder Mode]



[List Mode]

Step	Instruction	Device	
0	LD	X4	
1	BSFLP	Y60	K16
4	END		

[Operation]

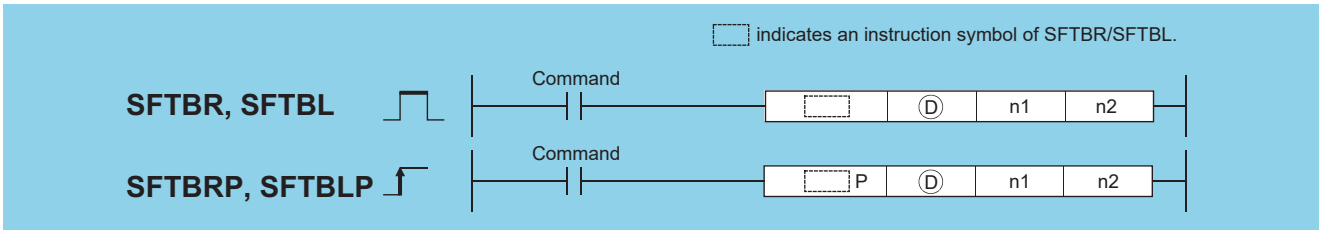


n-bit shift to right of n-bit data, n-bit shift to left of n-bit data

SFTBR(P), SFTBL(P)

Basic | High performance | Process | Redundant | Ver. Universal | LCPU

- QnU(D)(H)CPU, QnUDE(H)CPU: the serial number (first five digits) is "10102" or later
- Q00UJCPU, Q00UCPU, Q01UCPU, QnUDVCPU, QnUDPVCPU: Supported



(D): Head number of the devices to be shifted (bits)
n1: Number of bits to be shifted (BIN 16 bits)
n2: Number of shifts (BIN 16 bits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(D)	○*1	○	○	—					—
n1	○	○	○	○					—
n2	○	○	○	○					—

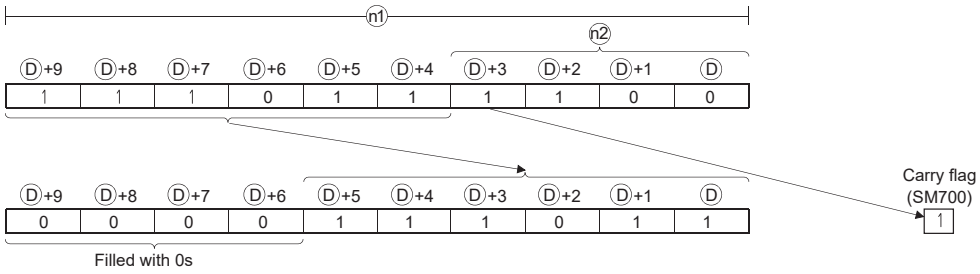
*1 T, ST, and C devices are not available.

Processing details

■SFTBR(P)

- This instruction shifts the n1 bits data in the devices starting from the device specified by (D) to the right by n2 bits.

n1=10, n2=4

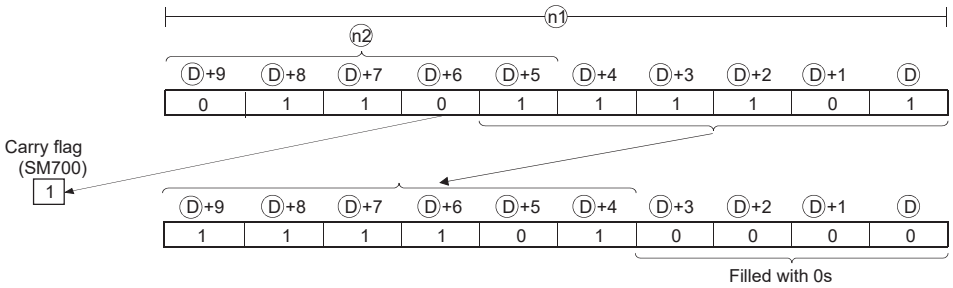


- n1 and n2 are specified under the condition that n1 is larger than n2. If the value of n2 is equal to or larger than the value of n1, the remainder of n2 / n1 (n2 divided by n1) is used for a shift. However, if the remainder of n2 / n1 is 0, the instruction will be not processed.
- This instruction specifies n1 ranged from 1 to 64.
- Bits starting from the highest bit to n2th bit are filled with 0s. If the value of n2 is larger than the value of n1, the remainder of n2 / n1 will be 0.
- If the value specified by n1 or n2 is 0, the instruction will be not processed.

■SFTBL(P)

• This instruction shifts the n1 bits data in the devices starting from the device specified by (D) to the left by n2 bits.

n1=10, n2=4



• n1 and n2 are specified under the condition that n1 is larger than n2. If the value of n2 is equal to or larger than the value of n1, the remainder of n2 / n1 (n2 divided by n1) is used for a shift. However, if the remainder of n2 / n1 is 0, the instruction will be not processed.

• This instruction specifies n1 ranged from 1 to 64.

• Bits starting from the lowest bit to n2th bit are filled with 0s. If the value of n2 is larger than the value of n1, the remainder of n2 / n1 will be 0.

• If the value specified by n1 or n2 is 0, the instruction will be not processed.

Operation error

• In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4100	The value specified in n1 is other than 0 to 64. The value in n2 is negative.	—	—	—	—	○	○
4101	The points specified in n1 exceed those of the device specified in (D).	—	—	—	—	○	○

Program example

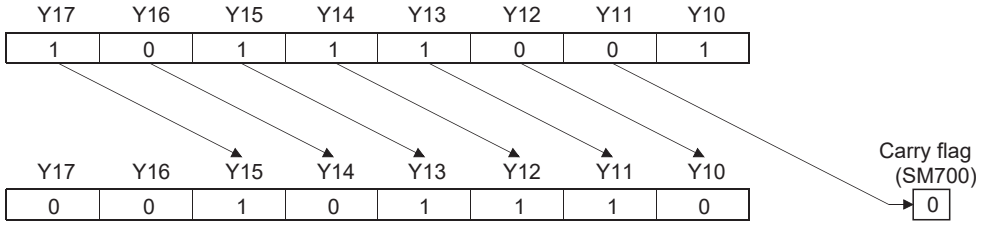
• The following program shifts the data of Y10 to Y17 (8 bits) specified by (D) to the right by 2 bits (n2), when M0 is turned on.

[Ladder Mode]

[List Mode]

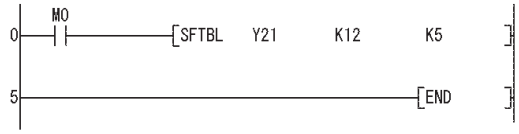
Step	Instruction	Device		
0	LD	M0		
1	SFTBR	Y10	K8	K2
5	END			

[Operation]



• The following program shifts the data of Y21 to Y2C (12 bits) specified by (D) to the left by 5 bits (n2), when M0 is turned on.

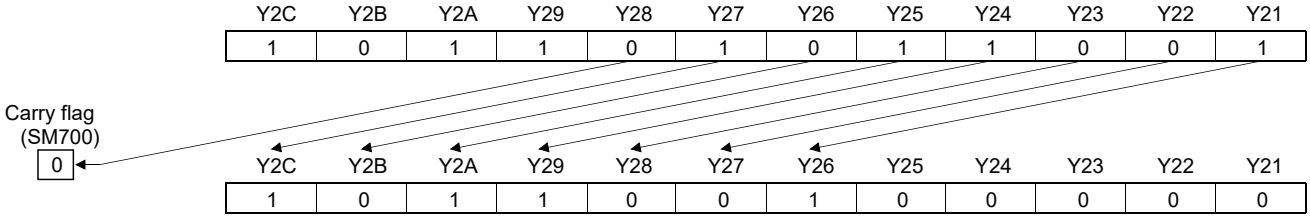
[Ladder Mode]



[List Mode]

Step	Instruction	Device		
0	LD	M0		
1	SFTBL	Y21	K12	K5
5	END			

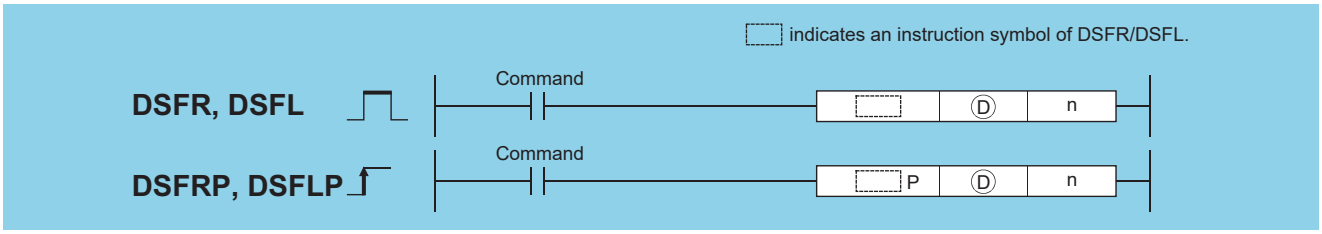
[Operation]



1-word shift to right of n-word data, 1-word shift to left of n-word data

DSFR(P), DSFL(P)

Basic | High performance | Process | Redundant | Universal | LCPU



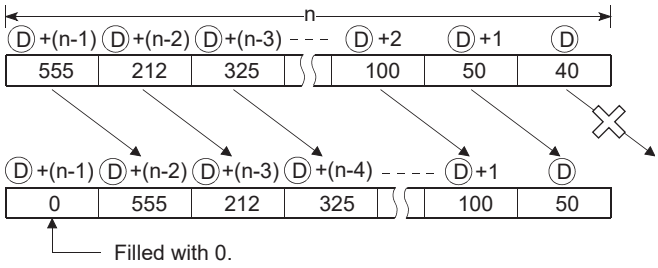
(D): Head number of the devices to be shifted (BIN 16 bits)
n: Number of devices to which shift is executed (BIN 16 bits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(D)	—	○		—					—
n	○	○		○					—

Processing details

■DSFR

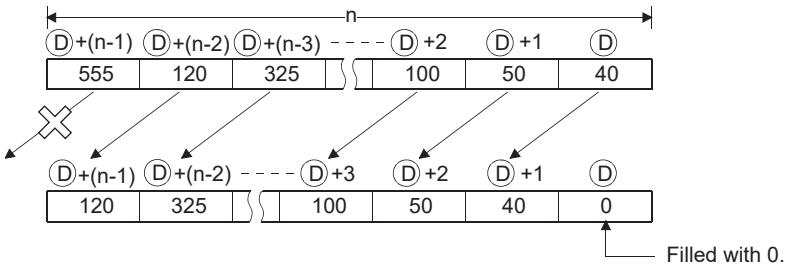
• Shifts data n points from device designated by (D) 1-word to the right.



• The device designated by (D)+(n-1) is filled with 0.

■DSFL

• Shifts data n points from device designated by (D) 1-word to the left.



• The device designated by (D) is filled with 0.

Operation error

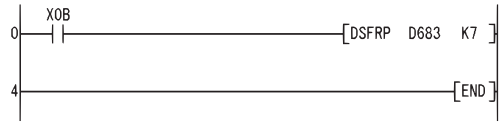
• In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4101	The points specified in n exceed those of the corresponding device specified in (D).	○	○	○	○	○	○

Program example

• The following program shifts the contents of D683 to D689 to the right when XB is turned ON.

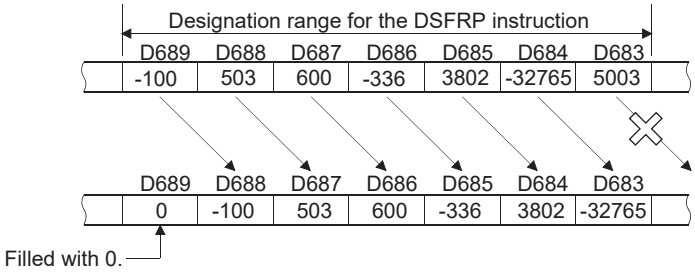
[Ladder Mode]



[List Mode]

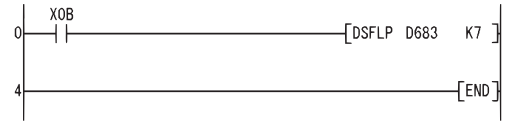
Step	Instruction	Device	
0	LD	XOB	
1	DSFRP	D683	K7
4	END		

[Operation]



• The following program shifts the contents of D683 to D689 to the left when XB is turned ON.

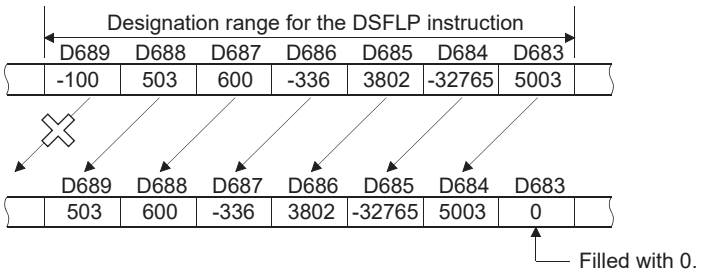
[Ladder Mode]



[List Mode]

Step	Instruction	Device	
0	LD	XOB	
1	DSFLP	D683	K7
4	END		

[Operation]

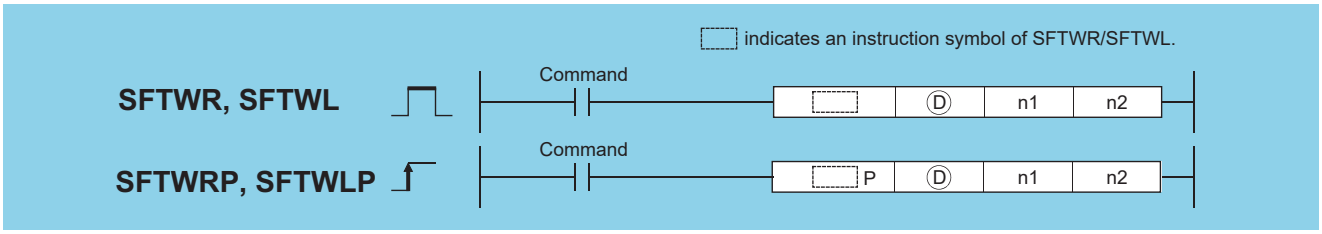


n-word shift to right of n-word data, n-word shift to left of n-word data

SFTWR(P), SFTWL(P)

Basic (×) | High performance (×) | Process (×) | Redundant (×) | Universal (Ver.) | LCPU

- QnU(D)(H)CPU, QnUDE(H)CPU: the serial number (first five digits) is "10102" or later
- Q00UJCPU, Q00UCPU, Q01UCPU, QnUDVCPU, QnUDPVCPU: Supported



(D): Head number of the devices to be shifted (BIN 16 bits)
n1: Number of words to be shifted (BIN 16 bits)
n2: Number of shifts (BIN 16 bits)

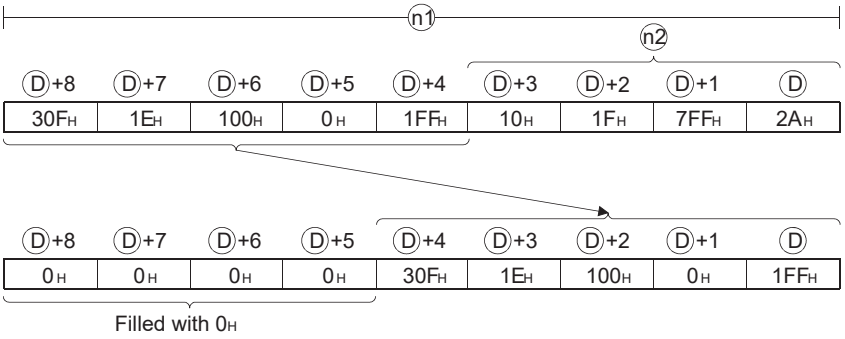
Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(D)	○	○	○	—					—
n1	—	○	○	○					—
n2	○	○	○	○					—

Processing details

■SFTWR(P)

- This instruction shifts n1 words data in the devices starting from the device specified by (D) to the right by n2 words.

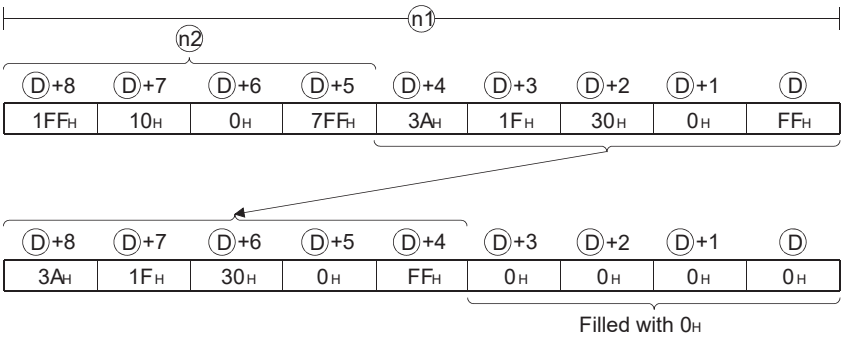
n1=9, n2=4



- The n2 words data in the devices starting from the highest device are filled with 0s.
- If the value specified by n1 or n2 is 0, the instruction will be not processed.
- If the value of n2 is equal to or larger than the value of n1, the n1 words data in the devices starting from the device specified by (D) will be filled with 0s.

■SFTWL(P)

• This instruction shifts the n1 words data in the devices starting from the device specified by (D) to the left by n2 words.
n1=9, n2=4



• The n2 words in the devices starting from the lowest device are filled with 0s.

• If the value specified by n1 or n2 is 0, the instruction will be not processed.

• If the value of n2 is equal to or greater than the value of n1, the n1 words devices starting from the device specified by (D) will be filled with 0s.

Operation error

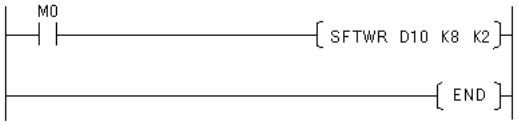
• In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4100	The value in n1 or n2 is negative.	—	—	—	—	○	○
4101	The points specified in n1 exceed those of the device specified in (D).	—	—	—	—	○	○

Program example

- The following program shifts the 8 words (n1) data stored in the devices starting from D10 specified by (D) to the right by 2 words (n2), when M0 is turned on.

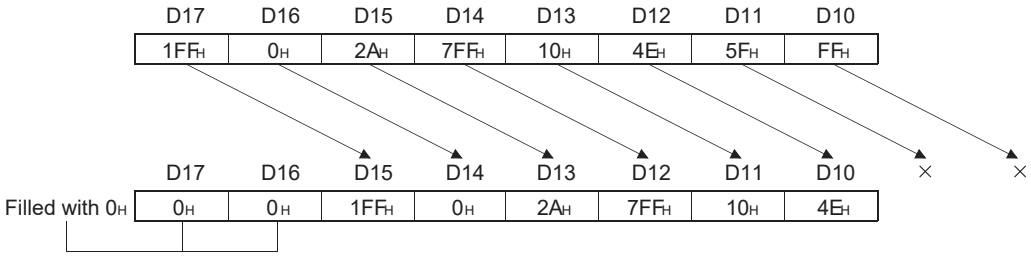
[Ladder Mode]



[List Mode]

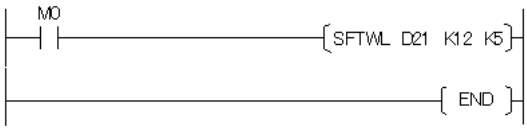
Step	Instruction	Device
0	LD	M0
1	SFTWR	D10 K8 K2
5	END	

[Operation]



- The following program shifts the 12 words (n1) data in the devices starting from D21 specified by (D) to the left by 5 words (n2), when M0 is turned on.

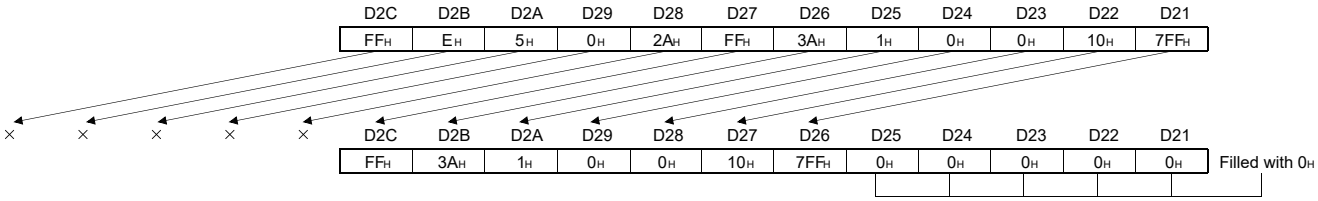
[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	M0
1	SFTWL	D21 K12 K5
5	END	

[Operation]

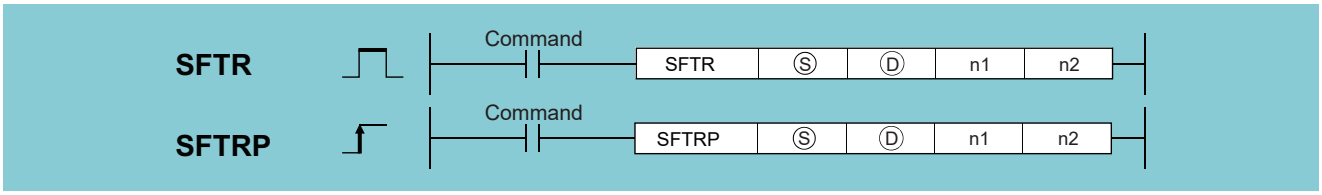


Bit shift right

SFTR(P)

Basic | High performance | Process | Redundant | Ver. Universal | Ver. LCPU

• QnUDVCPU, QnUDPVCPU, LCPU: The serial number (first five digits) is "16112" or later.



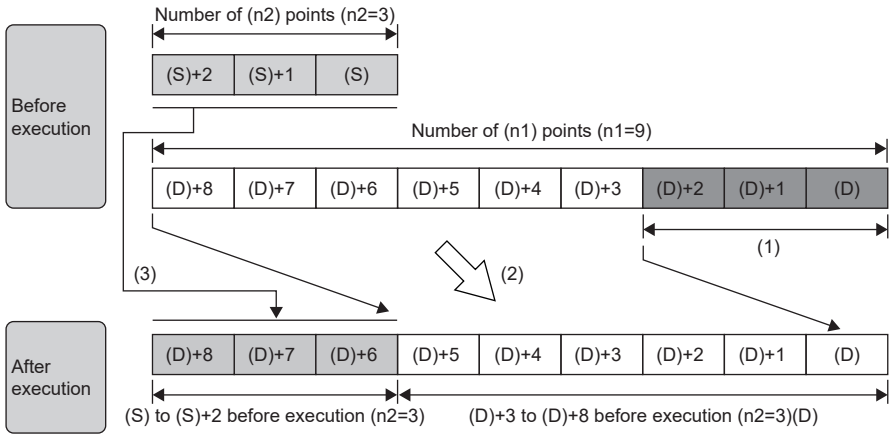
(S): Start number (bit) of the bit devices to be stored in empty data after shifting to the right
(D): Start number of bit device to shift to the right (bit)
n1: Bit data length of shift data (n2 ≤ n1 ≤ 1024) (BIN 16 bits)
n2: Number of bit points to shift to the right (n2 ≤ n1 ≤ 1024)*1 (BIN 16 bits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(S)	○	—	○	—				—	—
(D)	○	—	○	—				—	—
n1	—	○	○	—				○	—
n2	—	○	○	—				○	—

*1 Do not set a negative value in the number of bit points (n2) to be shifted to the right.

Processing details

• This instruction shifts bit data for (n2) bits to the right within the data range of (n1) bits starting from (D). After the shift, the function transfers (n2)-bit data from (S) to the (n2) bits from ((D)+(n1))-(n2).



(1) Overflow (deletion data)
(2) n2 bits shift to the right (n2=3)
(3) Coping n2 points

Operation error

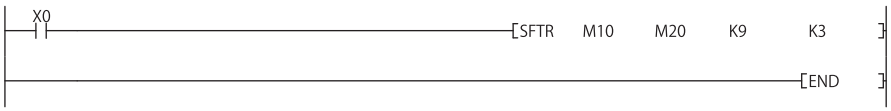
- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4100	n1 is other than 0 to 1024. n2 is a negative value.	—	—	—	—	○	○
4101	n1 is smaller than n2. The value in the device specified by the instruction exceeds the setting range. The device range for the number of n2 points from (S) overlaps that for the number of n1 points from (D).	—	—	—	—	○	○

Program example

- The sample program shifts the 9 bits of data starting from M20 3 bits to the right and transfers 3 bits of data from M10 to the 3 bits from M26.

[Ladder Mode]

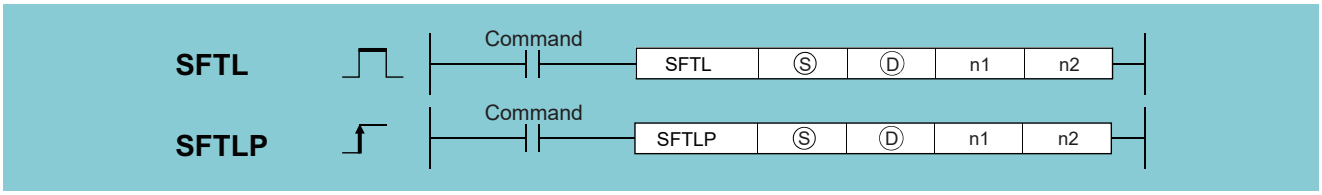


Bit shift left

SFTL(P)

Basic | High performance | Process | Redundant | Ver. Universal | Ver. LCPU

• QnUDVCPU, QnUDPVCPU, LCPU: The serial number (first five digits) is "16112" or later.



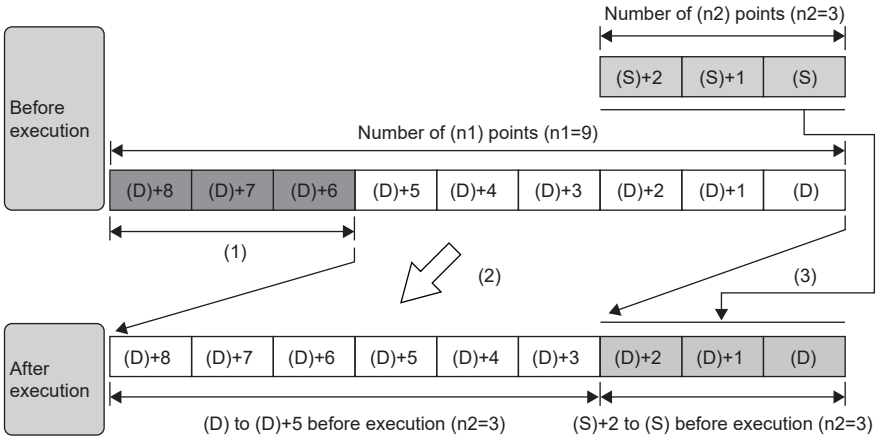
(S): Start number (bit) of the bit devices to be stored in empty data after shifting to the left
(D): Start number of bit device to shift to the left (bit)
n1: Bit data length of shift data (n2 ≤ n1 ≤ 1024) (BIN 16 bits)
n2: Number of bit points to shift to the left (n2 ≤ n1 ≤ 1024)*1 (BIN 16 bits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(S)	○	—	○	—				—	—
(D)	○	—	○	—				—	—
n1	—	○	○	—				○	—
n2	—	○	○	—				○	—

*1 Do not set a negative value in the number of bit points (n2) to be shifted to the left.

Processing details

• This instruction shifts bit data for (n2) bits to the left within the data range of (n1) bits starting from (D). After the shift, the function transfers (n2)-bit data from (S) to the (n2) bits from (D).



(1) Overflow (deletion data)
(2) n2 bits shift to the left (n2=3)
(3) Coping n2 points

Operation error

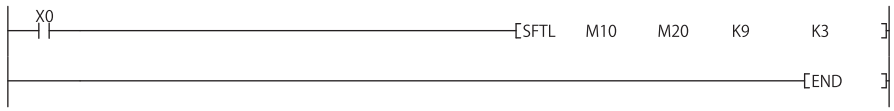
- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4100	n1 is other than 0 to 1024. n2 is a negative value.	—	—	—	—	○	○
4101	n1 is smaller than n2. The value in the device specified by the instruction exceeds the setting range. The device range for the number of n2 points from (S) overlaps that for the number of n1 points from (D).	—	—	—	—	○	○

Program example

- The sample program shifts the 9 bits of data starting from M20 3 bits to the left and transfers 3 bits of data from M10 to the 3 bits from M20.

[Ladder Mode]

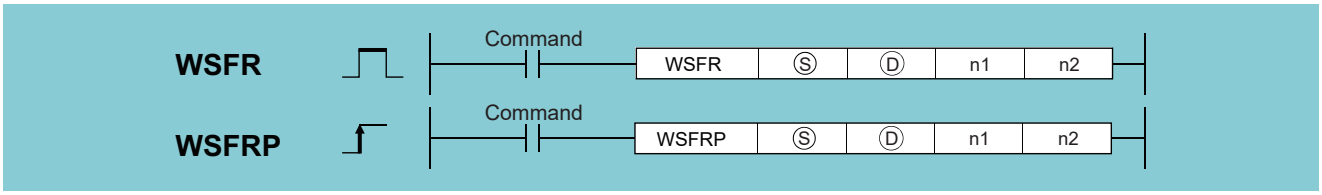


Word shift right

WSFR(P)

Basic | High performance | Process | Redundant | Ver. Universal | Ver. LCPU

• QnUDVCPU, QnUDPVCPU, LCPU: The serial number (first five digits) is "16112" or later.



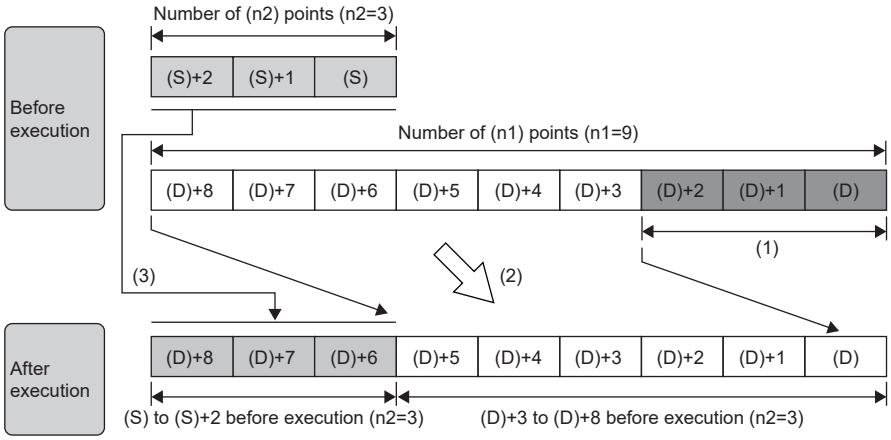
(S): Start number of the word devices to be stored in empty data after shifting to the right (BIN 16 bits)
(D): Start number of word device to shift to the right (BIN 16 bits)
n1: Word data length of shift data (n2 ≤ n1 ≤ 512) (BIN 16 bits)
n2: Number of word points to shift to the right (n2 ≤ n1 ≤ 512)*1 (BIN 16 bits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(S)	—	○	○	—				—	—
(D)	—	○	○	—				—	—
n1	—	○	○	—				○	—
n2	—	○	○	—				○	—

*1 Do not set a negative value in the number of word points (n2) to be shifted to the right.

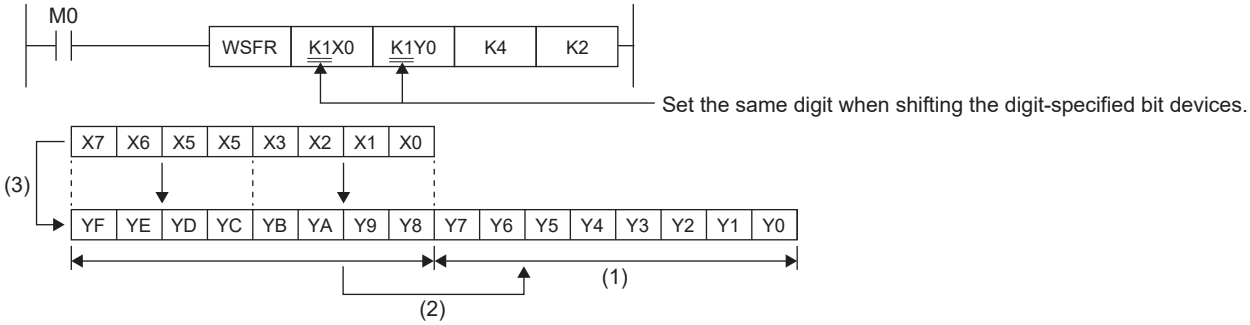
Processing details

• This instruction shifts word data for (n2) words to the right within the data range of (n1) words starting from (D). After the shift, the function transfers (n2)-word data from (S) to the (n2) words from ((D)+(n1))-(n2).



(1) Overflow (deletion data)
(2) n2 words shift to the right (n2=3)
(3) Coping n2 words

• An example of operation where digit specification is used for shift data (D) and the data (S) to be stored in the shift data is shown below.



(1) Overflow (deletion data)
(2) n2 words shift to the right (n2=2)
(3) Coping n2 words

Operation error

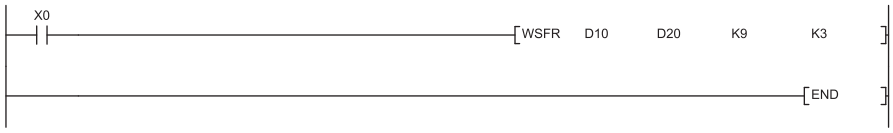
• In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4100	n1 is other than 0 to 512. n2 is a negative value.	—	—	—	—	○	○
4101	n1 is smaller than n2. The value in the device specified by the instruction exceeds the setting range. The device range for the number of n2 points from (S) overlaps that for the number of n1 points from (D).	—	—	—	—	○	○

Program example

• The sample program shifts the 9 words of data starting from D20 3 words to the right and transfers 3 words of data from D10 to the 3 words from D26.

[Ladder Mode]

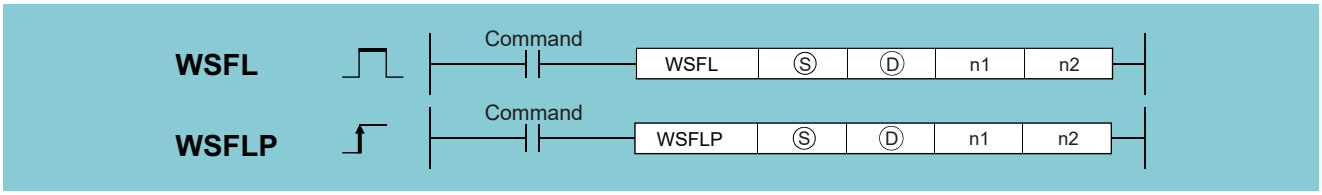


Word shift left

WSFL(P)

Basic | High performance | Process | Redundant | Ver. Universal | Ver. LCPU

• QnUDVCPU, QnUDPVCPU, LCPU: The serial number (first five digits) is "16112" or later.



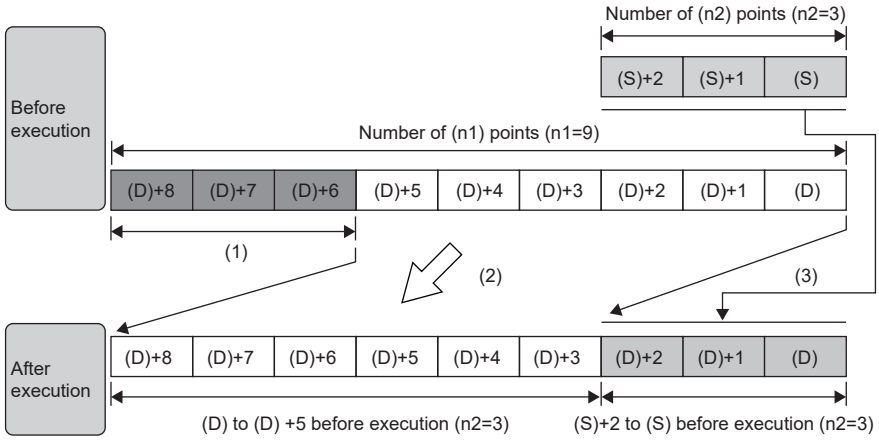
(S): Start number of the word devices to be stored in empty data after shifting to the left (BIN 16 bits)
(D): Start number of word device to shift to the left (BIN 16 bits)
n1: Word data length of shift data (n2 ≤ n1 ≤ 512) (BIN 16 bits)
n2: Number of word points to shift to the left (n2 ≤ n1 ≤ 512)*1 (BIN 16 bits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(S)	—	○	○	—				—	—
(D)	—	○	○	—				—	—
n1	—	○	○	—				○	—
n2	—	○	○	—				○	—

*1 Do not set a negative value in the number of word points (n2) to be shifted to the left.

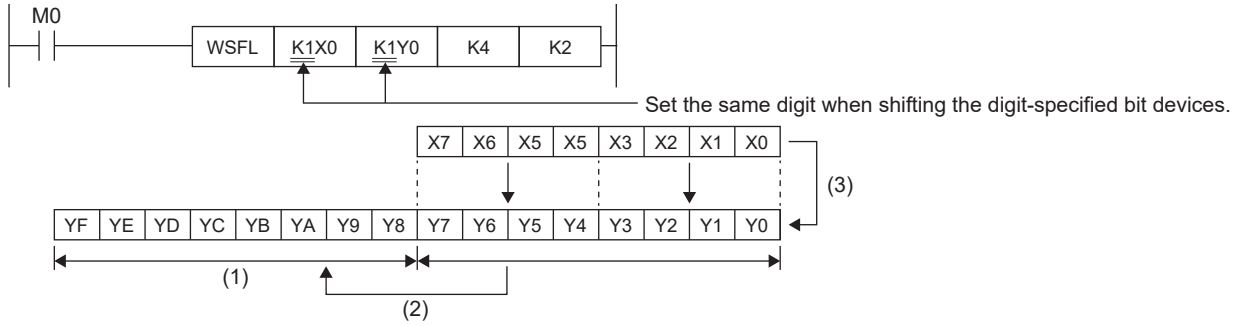
Processing details

• This instruction shifts word data for (n2) words to the left within the data range of (n1) words starting from (D). After the shift, the function transfers (n2)-word data from (S) to the (n2) words from (D).



(1) Overflow (deletion data)
(2) n2 words shift to the left (n2=3)
(3) Coping n2 words

• An example of operation where digit specification is used for shift data (D) and the data (S) to be stored in the shift data is shown below.



(1) Overflow (deletion data)
(2) n2 words shift to the left (n2=2)
(3) Coping n2 words

Operation error

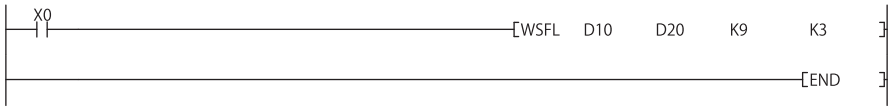
• In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4100	n1 is other than 0 to 512. n2 is a negative value.	—	—	—	—	○	○
4101	n1 is smaller than n2. The value in the device specified by the instruction exceeds the setting range. The device range for the number of n2 points from (S) overlaps that for the number of n1 points from (D).	—	—	—	—	○	○

Program example

• The sample program shifts the 9 words of data starting from D20 3 words to the left and transfers 3 words of data from D10 to the 3 words from D20.

[Ladder Mode]

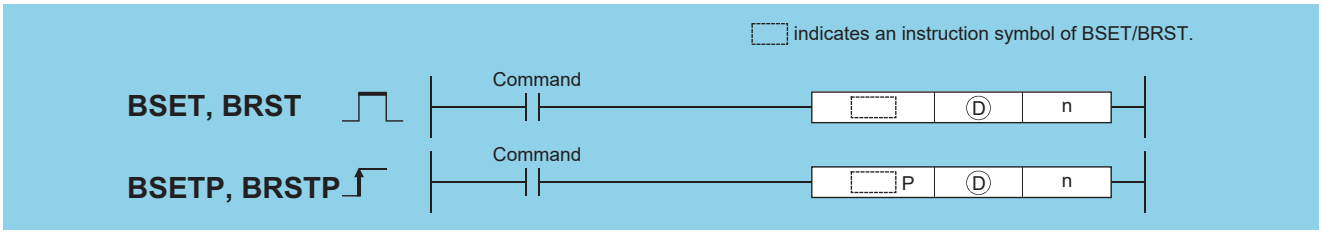


7.4 Bit Processing Instructions

Bit set for word devices, bit reset for word devices

BSET(P), BRST(P)

Basic | High performance | Process | Redundant | Universal | LCPU



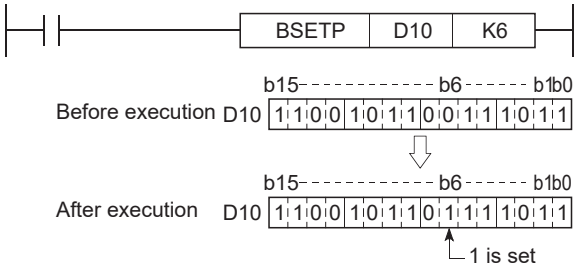
(D): Device whose bits are set/reset (BIN 16 bits)
n: Position of the bit to be set/reset (0 to 15) (BIN 16 bits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(D)	○							—	—
n	○							○	—

Processing details

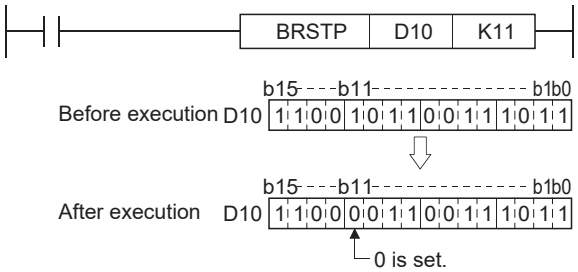
■BSET

- Sets (sets "1" at) the nth bit in the word device designated at (D).
- If n exceeds "15", bit set/reset is performed with the lower 4 bits of the data.



■BRST

- Resets the nth bit of a word device designated by (D) to 0.
- If n exceeds "15", bit set/reset is performed with the lower 4 bits of the data.



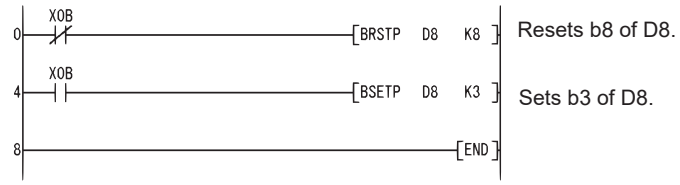
Operation error

- There is no operation error in the BSET(P) or BRST(P) instruction.

Program example

- The following program resets the 8th bit of D8 (b8) to 0 when XB is OFF, and sets the 3rd bit of D8 (b3) to 1 when XB is ON.

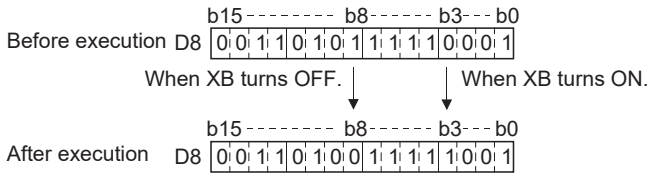
[Ladder Mode]



[List Mode]

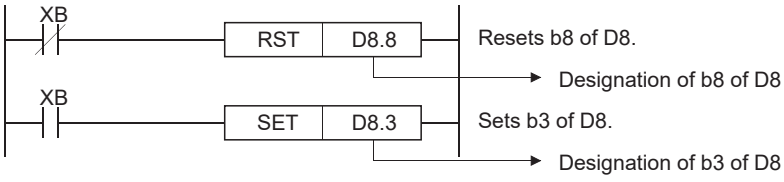
Step	Instruction	Device	
0	LD I	XOB	
1	BRSTP	D8	K8
4	LD	XOB	
5	BSETP	D8	K3
8	END		

[Operation]



Point

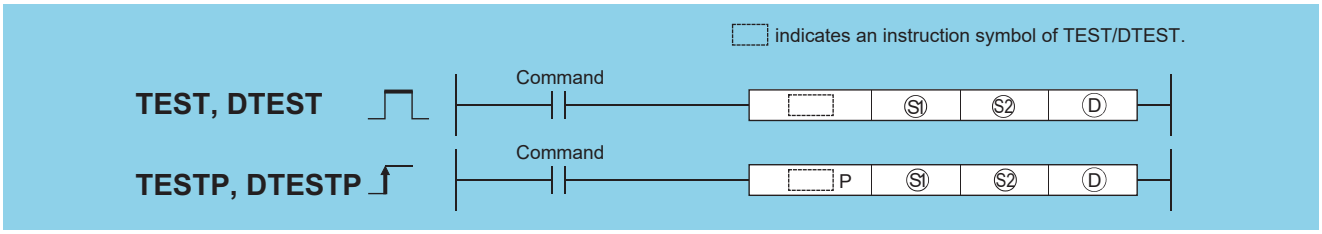
Bit set or reset of word devices can also be conducted by bit designation of word devices.
For the bit specification for word devices, refer to the User's Manual (Function Explanation, Program Fundamentals) for the CPU module used.
The processing of program example would be conducted as shown below if bit designation of a word device had been used:



Bit tests

TEST(P), DTEST(P)

Basic | High performance | Process | Redundant | Universal | LCPU



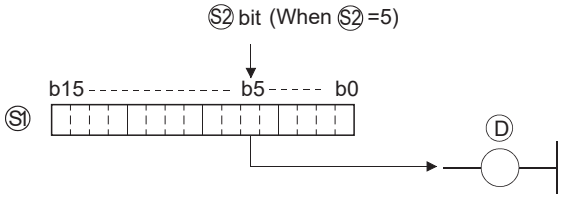
(S1): Number of the device where bit data to be extracted is stored (BIN 16/32 bits)
(S2): Location of the bit data to be extracted (0 to 15 (TEST)/0 to 31 (DTEST)) (BIN 16 bits)
(D): Number of the bit device where the extracted data will be stored (bits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(S1)	○						○	—	—
(S2)	○						○	○	—
(D)	○ (Other than T, ST, C)						—	—	—

Processing details

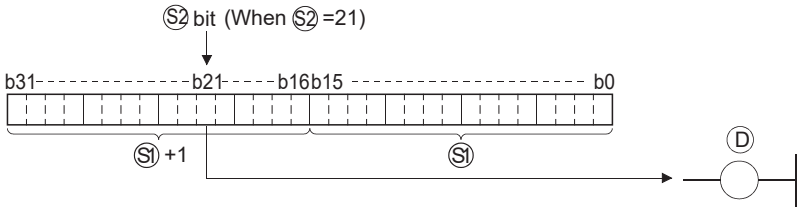
■TEST

- Fetches bit data at the location designated by (S2) within the word device designated by (S1), and writes it to the bit device designated by (D).
- The bit device designated by (D) is OFF when the relevant bit is "0" and ON when it is "1".
- The position designated by (S2) indicates the position of an individual bit in a 1-word data block (0 to 15). When 16 or more is designated at (S2), the target is the bit data at the position indicated by the remainder of n / 16. For example, when n = 18, the target is the data at b2 since the remainder of 18 / 16 =1 is "2".



■DTEST

- Fetches bit data at the location designated by (S2) within the 2-word device designated by (S1), or (S1)+1, and writes it to the bit device designated by (D).
- The bit device designated by (D) is OFF when the relevant bit is "0" and ON when it is "1".
- The position designated by (S2) indicates the position of an individual bit in a 2-word data block (0 to 31). When 32 or more is designated at (S2), the target is the bit data at the position indicated by the remainder of n / 32. For example, when n = 34, the target is the data at b2 since the remainder of 34 / 32 =1 is "2".



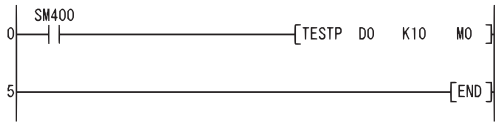
Operation error

- There is no operation error in the TEST(P) or DTEST(P) instruction.

Program example

- The following program turns M0 ON or OFF based on the status of the 10th bit in the 1-word data block (D0).

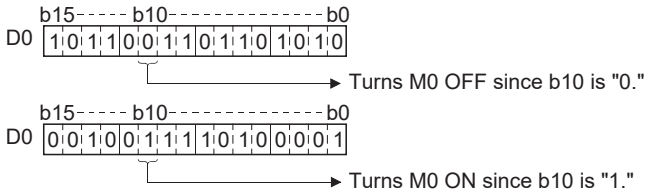
[Ladder Mode]



[List Mode]

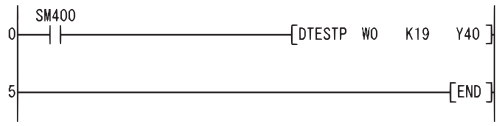
Step	Instruction	Device		
0	LD	SM400		
1	TESTP	D0	K10	M0
5	END			

[Operation]



- The following program turns Y40 ON or OFF, depending on the status of the 19th bit of the 2-word data (W0 and W1).

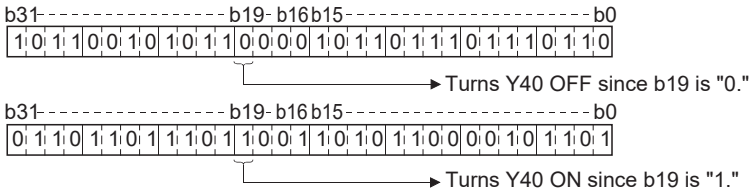
[Ladder Mode]



[List Mode]

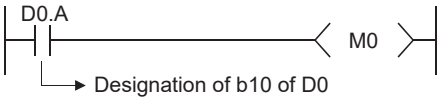
Step	Instruction	Device		
0	LD	SM400		
1	DTESTP	W0	K19	Y40
5	END			

[Operation]



Point

Programs using the bit test instruction can be rewritten as programs using bit designation of word devices. The program example of the TESTP instruction would be conducted as shown below if bit designation of a word device had been used:

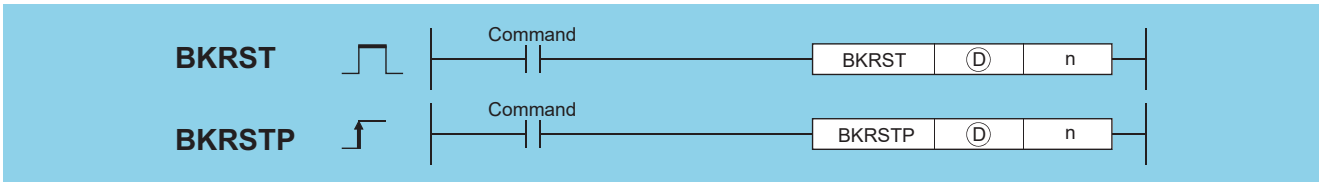


M0 turns ON/OFF depending on the ON/OFF status of b10 of D0 (D0.A).

Batch reset of bit devices

BKRST(P)

Basic | High performance | Process | Redundant | Universal | LCPU



(D): Head number of the devices to be reset (bits)
n: Number of the devices to be reset (BIN 16 bits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(D)	○			—					—
n	○			○					—

Processing details

- Resets bit device n-points from the bit device designated by (D).

Device	Status
Annunciator (F)	• Turns device n-points from annunciator (F) number designated by (D) OFF. • Deletes annunciator number turned OFF from SD64 to SD79 and compresses remaining data forward. • Stores number of annunciators stored from SD64 to SD79 at SD63.
Timer (T) Counter (C)	• Sets the current value n-points from timer (T) or counter c designated by (C) to 0, and turns coil contact OFF.
Bit devices other than the above	• Turns OFF coil or contact n-points from the device designated by (D).

- If the designated device is OFF, the device status will not change.

Operation error

- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4101	The points specified in n exceed those of the corresponding device specified in (D).	○	○	○	○	○	○

Program example

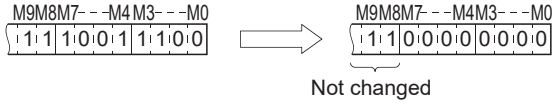
- The following program turns OFF devices from M0 to M7 when X0 is turned ON.

[Ladder Mode]

[List Mode]

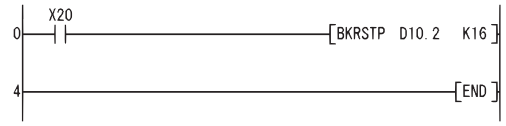
Step	Instruction	Device	
0	LD	X0	
1	BKRSTP	M0	K8
4	END		

[Operation]



- The following program sets data from 2nd bit (b2) of D10 to 1st bit (b1) of D11 to 0 when X20 is turned ON.

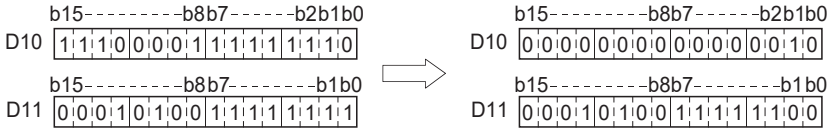
[Ladder Mode]



[List Mode]

Step	Instruction	Device	
0	LD	X20	
1	BKRSTP	D10.2	K16
4	END		

[Operation]

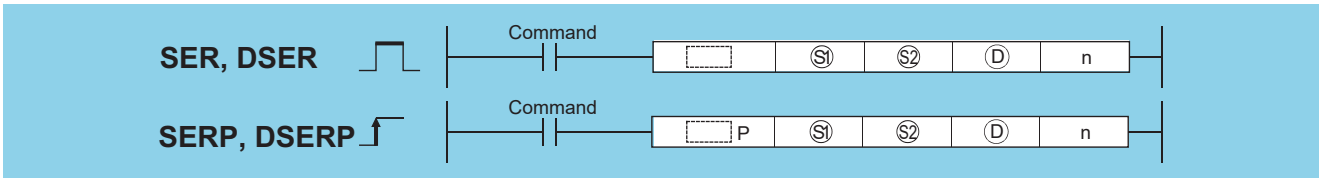


7.5 Data Processing Instructions

16-bit data search, 32-bit data search

SER(P), DSER(P)

Basic | High performance | Process | Redundant | Universal | LCPU



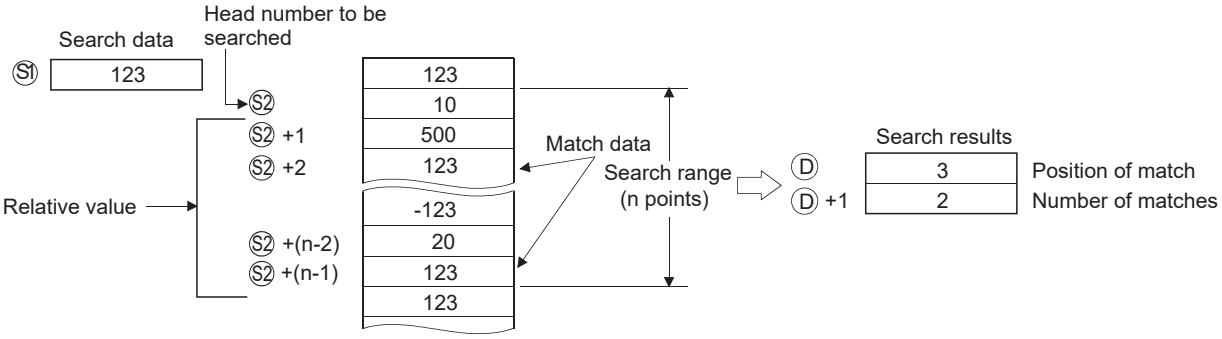
(S1): Search data or head number of the devices where the search data is stored (BIN 16/32 bits)
(S2): Data to be searched or head number of the devices where the data to be searched is stored (BIN 16 bits)
(D): Head number of the devices where the search result will be stored (BIN 16 bits)
n: Number of searches (BIN 16 bits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(S1)	○	○		○	○			○	—
(S2)	—	○		—	—			—	—
(D)	—	○		—	○			—	—
n	○	○		○	○			○	—

Processing details

■SER

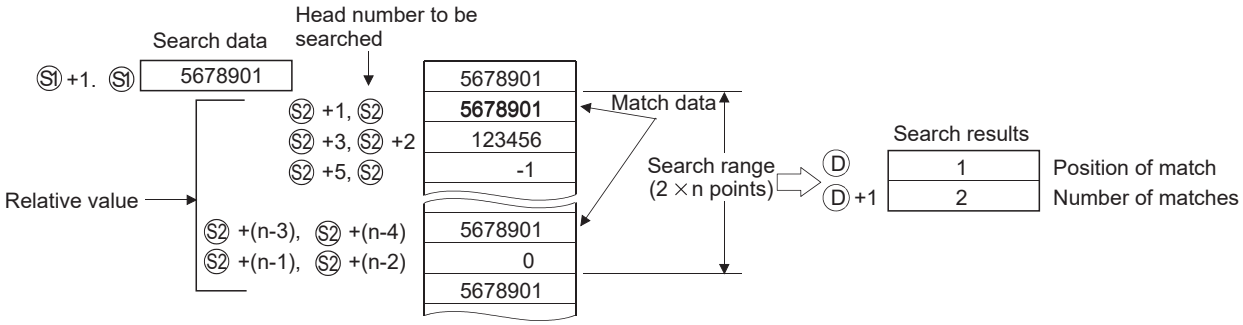
- Searches n points from the 16-bit data of the device designated by (S2), regarding 16-bit data of the device designated by (S1) as a keyword. Then, the number of matches with the keyword is stored at the device designated by (D)+1, and the first matched device number (in the relative number from (S2)) is stored at the device designated by (D).



- No processing is conducted if n is 0 or a negative value.
- If no matches are found in the search, the devices designated at (D) and (D)+1 become "0".

■DSER

- Searches n points from the device designated by (S2) in 32-bit units (2 × n points in 16-bit units) regarding 32-bit data of the device designated by (S1) +1 and (S1) as a keyword. Then, the number of matches with the keyword is stored at the device designated by (D)+1, and the first matched device number (in the relative number from (S2)) is stored at the device designated by (D).

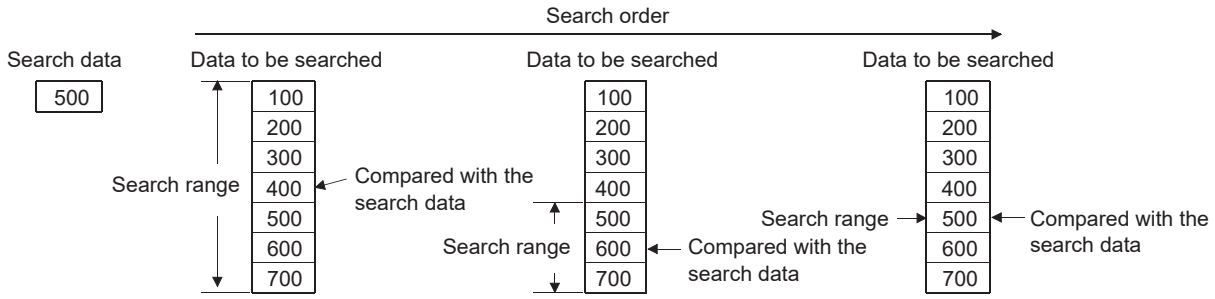


- No processing is conducted if n is 0 or a negative value.
- If no matches are found in the search, the devices designated at (D) and (D)+1 become "0".

Point

If the data to be searched using the SER/DSER instruction is sorted in the ascending order, searches can be accelerated by the use of the binary search method, which is activated by turning SM702*1 ON.
However, correct search results are not obtained if SM702 is turned ON when the data to be searched is not sorted in the ascending order.

*1 SM702 is the special relay for setting the search method.
- SM702 OFF: Sequential search method (linear search method)
(Comparison with the search data starts from the beginning of the data to be searched.)
- SM702 ON: Binary search method
(Obtains the center value of the sorted array and decides if the obtained value is larger or smaller than the search value, then, chooses the area for search between the larger and smaller value divisions. By repeating this process, the area for search is narrowed down.)



Operation error

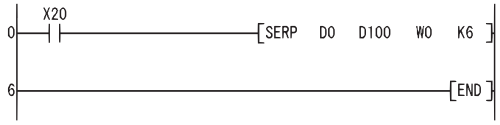
- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4101	The range of n exceeds that of the device specified in (S2)	○	○	○	○	○	○
	The device range specified in (D) exceeds the range of the corresponding device.	—	—	—	—	○	○

Program example

- The following program searches D100 to D105 for the contents of D0 when X20 is ON, and stores the search results at W0 and W1.

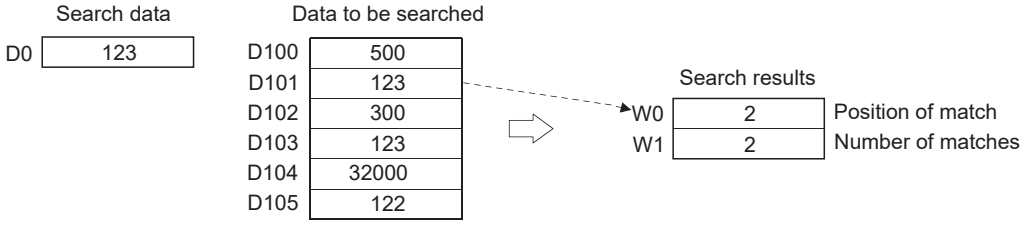
[Ladder Mode]



[List Mode]

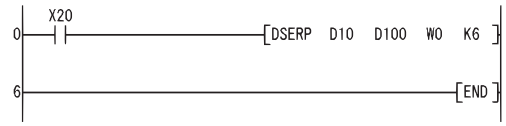
Step	Instruction	Device
0	LD	X20
1	SERP	D0 D100 W0 K6
6	END	

[Operation]



- The following program searches D100 to D111 for the contents of D11 and D10 when X20 is ON, and stores the search results at W0 and W1.

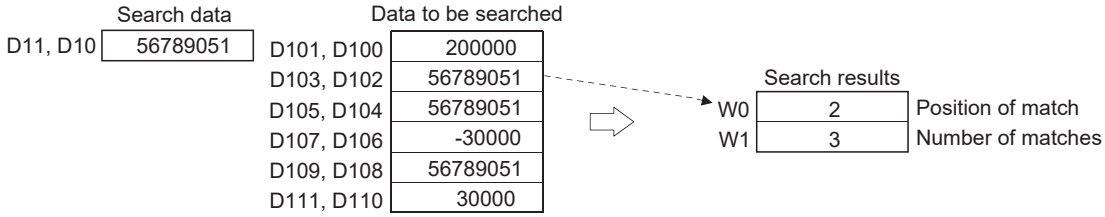
[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	X20
1	DSERP	D10 D100 W0 K6
6	END	

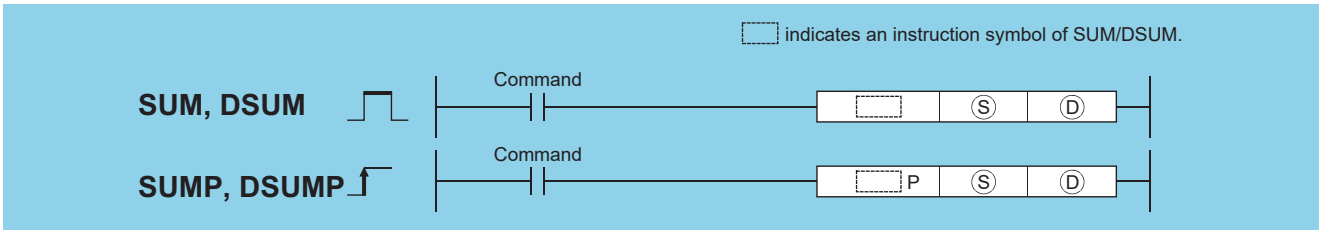
[Operation]



16-bit data bit check, 32-bit data check

SUM(P), DSUM(P)

Basic | High performance | Process | Redundant | Universal | LCPU



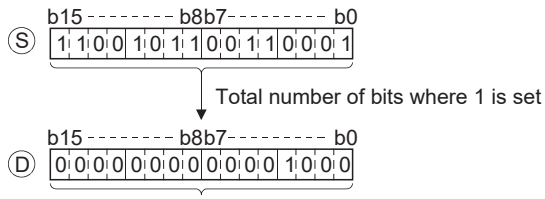
(S): Head number of the devices where the total number of bits of "1" is counted (BIN 16/32 bits)
(D): Head number of the devices where the total number of the bits will be stored (BIN 16/32 bits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(S)	○							○	—
(D)	○							—	—

Processing details

■SUM

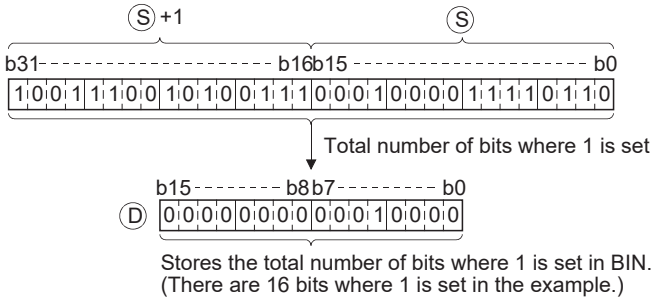
- From the 16-bit data in the device designated by (S), stores the total number of bits where 1 is set, in the device designated by (D).



Stores the total number of bits where 1 is set in BIN.
(There are 8 bits where 1 is set in the example.)

■DSUM

- From the 32-bit data in the device designated by (S), stores the total number of bits where 1 is set, in the device designated by (D).



Stores the total number of bits where 1 is set in BIN.
(There are 16 bits where 1 is set in the example.)

Operation error

- There is no operation error in the SUM(P) or DSUM(P) instruction.

Program example

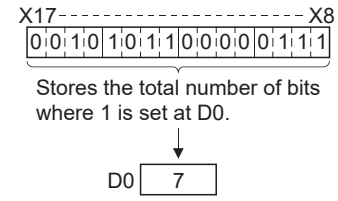
• The following program stores the number of bits which are ON from X8 to X17 into D0 when X10 is turned ON.

[Ladder Mode]

[List Mode]

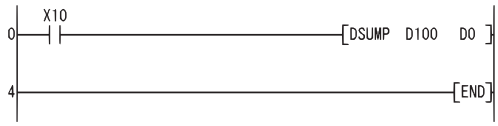
Step	Instruction	Device	
0	LD	X10	
1	SUMP	K4X8	D0
4	END		

[Operation]



• The following program stores the number of bits which are ON in D100 and D101 into D0 when X10 is turned ON.

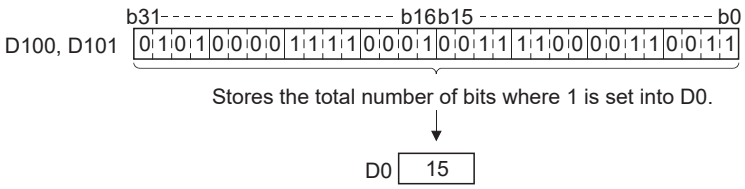
[Ladder Mode]



[List Mode]

Step	Instruction	Device	
0	LD	X10	
1	DSUMP	D100	D0
4	END		

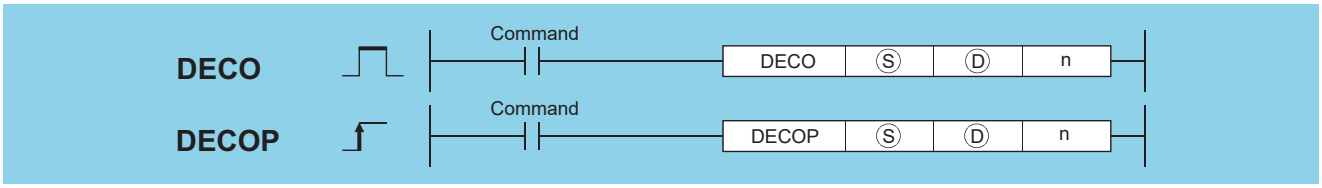
[Operation]



Decoding from 8 to 256 bits

DECO(P)

Basic | High performance | Process | Redundant | Universal | LCPU

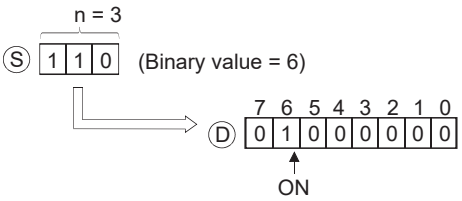


(S): Data to be decoded or the number of the device where the data to be decoded is stored (BIN 16 bits)
(D): Head number of the devices where the decoding result will be stored (Device name)
n: Valid bit length (1 to 8), 0: No processing (BIN 16 bits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(S)	○			○				○	—
(D)	○			—				—	—
n	○			○				○	—

Processing details

- Turns ON the bit position of (D), which corresponds to the binary value designated by the lower n bits at (S).



- The value of n can be designated between 1 and 8.
- No processing is conducted if n = 0, and there are no changes in the details of the device designated at (D).
- Bit devices are treated as 1 bit, and word devices as 16 bits.

Operation error

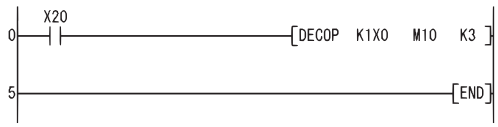
- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4100	The value of n is other than 0 to 8.	○	○	○	○	○	○
4101	The range 2^n bits from (D) exceeds the range of the corresponding device. The range n bits from (S) exceeds each setting area of the corresponding device.	○	○	○	○	○	○

Program example

• The following program decodes the 3 bits from X0 and stores the results at M10 when X20 is ON.

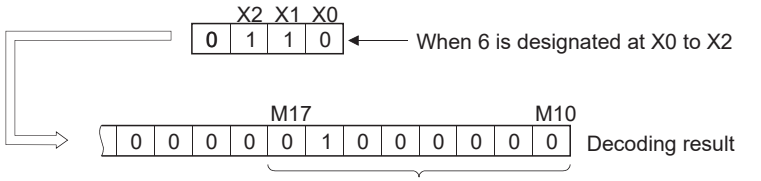
[Ladder Mode]



[List Mode]

Step	Instruction	Device		
0	LD	X20		
1	DECOP	K1X0	M10	K3
5	END			

[Operation]

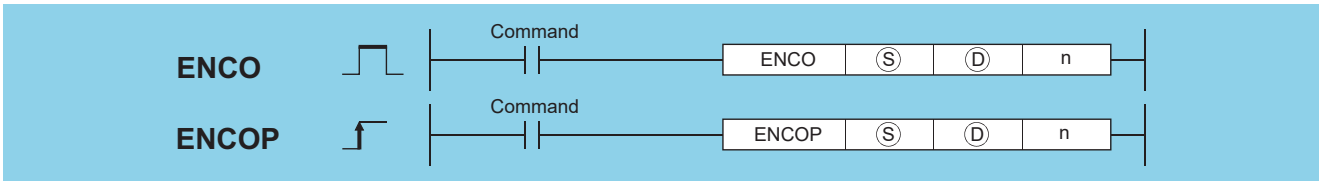


If 3 bits are designated as significant bits, 8 points are occupied.

Encoding from 256 to 8 bits

ENCO(P)

Basic | High performance | Process | Redundant | Universal | LCPU

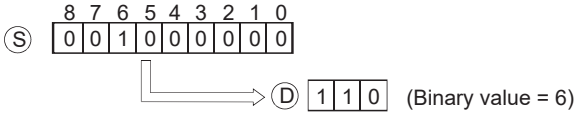


(S): Head number of the device where the data to be encoded is stored (Device name)
(D): Number of the device where the encoding result will be stored (BIN 16 bits)
n: Valid bit length (1 to 8), 0: No processing (BIN 16 bits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(S)	○			—				—	—
(D)	○			○				—	—
n	○			○				○	—

Processing details

- Stores the binary value corresponding to the bits which are "1" included in the 2^n-bit data of (S) to (D).



- The value of n can be designated at between 1 and 8.
- If n=0, there will be no operation, and the contents of (D) will not change.
- Bit devices are treated as 1 bit, and word devices as 16 bits.
- If more than 1 bit is at 1, processing will be conducted at the upper bit location.

Operation error

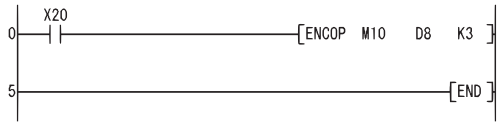
- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4100	The value of n is other than 0 to 8. All data 2^n bits from (S) is "0".	○	○	○	○	○	○
4101	The range 2^n bits from (S) exceeds the range of the corresponding device.	○	○	○	○	○	○

Program example

• The following program encodes the 3 bits from M10 when X20 is ON, and stores the results at D8.

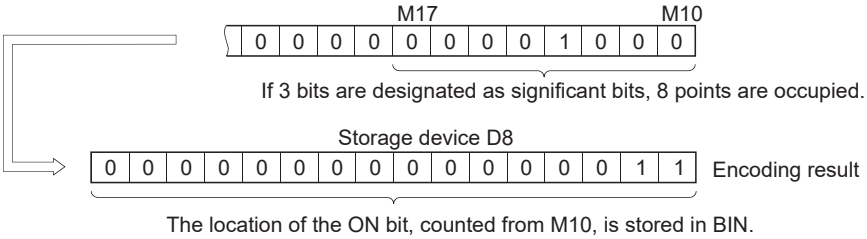
[Ladder Mode]



[List Mode]

Step	Instruction	Device		
0	LD	X20		
1	ENCOP	M10	D8	K3
5	END			

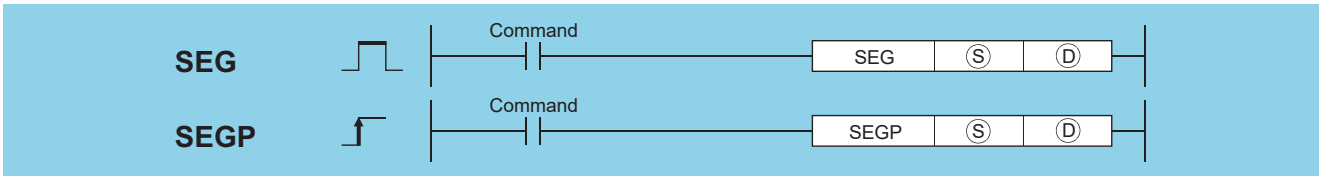
[Operation]



7-segment decode

SEG(P)

Basic | High performance | Process | Redundant | Universal | LCPU

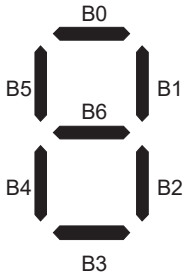


(S): Data to be decoded or head number of the devices where the data to be decoded is stored (BIN 16 bits)
(D): Head number of the devices where the decoding result will be stored (BIN 16 bits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(S)	○							○	—
(D)	○							—	—

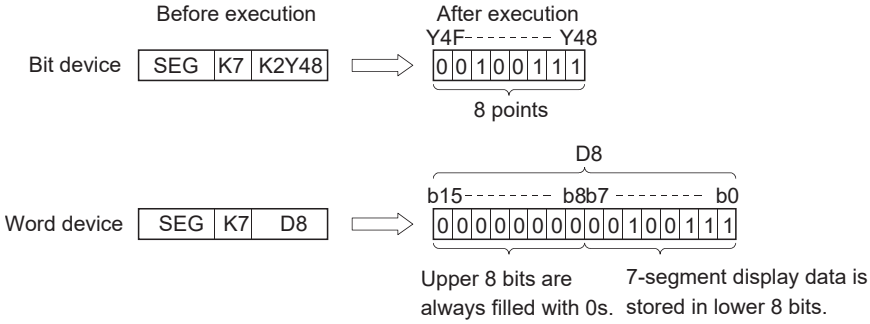
Processing details

- Decodes the data from 0 to F designated by the lower 4 bits of (S) to 7-segment display data, and stores at (D).

(S)		Configuration of 7 Segments	(D)								Display data
Hexadecimal	Bit pattern		B7	B6	B5	B4	B3	B2	B1	B0*1	
0	0000		0	0	1	1	1	1	1	1	
1	0001		0	0	0	0	0	1	1	0	
2	0010		0	1	0	1	1	0	1	1	
3	0011		0	1	0	0	1	1	1	1	
4	0100		0	1	1	0	0	1	1	0	
5	0101		0	1	1	0	1	1	0	1	
6	0110		0	1	1	1	1	1	0	1	
7	0111		0	0	1	0	0	1	1	1	
8	1000		0	1	1	1	1	1	1	1	
9	1001		0	1	1	0	1	1	1	1	
A	1010		0	1	1	1	0	1	1	1	
B	1011		0	1	1	1	1	1	0	0	
C	1100		0	0	1	1	1	0	0	1	
D	1101		0	1	0	1	1	1	1	0	
E	1110		0	1	1	1	1	0	0	1	
F	1111		0	1	1	1	0	0	0	1	

*1 Head number of bit device or lowest bit of word device

- If (D) is a bit device, indicates the head number of the devices storing the 7-segment display data; if it is a word device, indicates the number of the device storing the data.



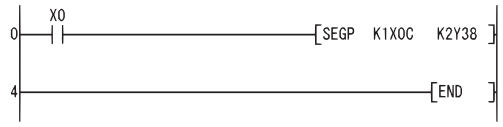
Operation error

- There is no operation error in the SEG(P) instruction.

Program example

- The following program converts the data from XC to XF to 7-segment display data and outputs it to Y38 to Y3F when X0 is turned ON.

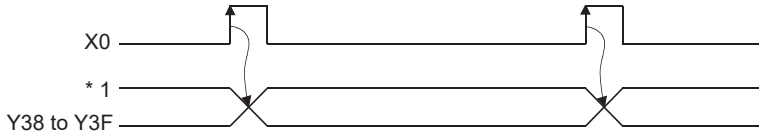
[Ladder Mode]



[List Mode]

Step	Instruction	Device	
0	LD	X0	
1	SEGP	K1X0C	K2Y38
4	END		

[Timing Chart]

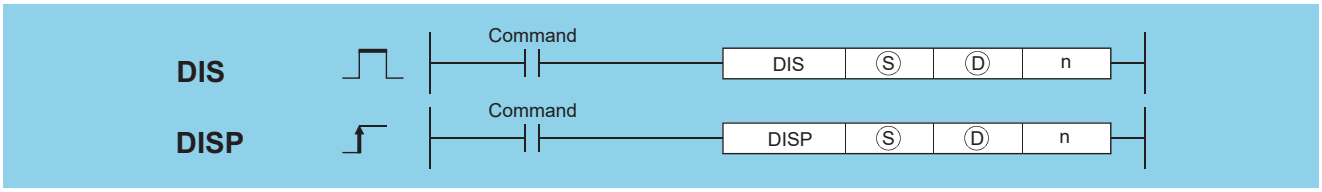


*1: The data Y38 to Y3F will not change until the next data is output.

4-bit dissociation of 16-bit data

DIS(P)

Basic | High performance | Process | Redundant | Universal | LCPU

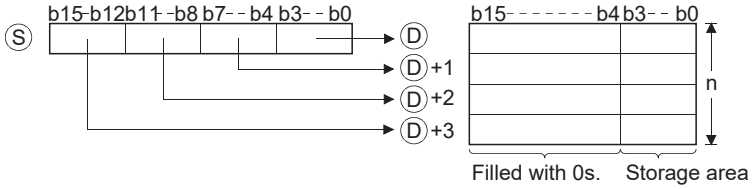


(S): Head number of the devices where data to be dissociated is stored (BIN 16 bits)
(D): Head number of the devices where the dissociated data will be stored (BIN 16 bits)
n: Number of dissociations (1 to 4), 0: No processing (BIN 16 bits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(S)	○	○		○					—
(D)	—	○		—					—
n	○	○		○					—

Processing details

- Stores the lower n-digits (1 digit is 4 bits) of the 16-bit data designated by (S) at the lower 4 bits n-points from the device designated by (D).



- The upper 12 bits n-points from the device designated by (S) become 0.
- The value of n can be designated at between 1 and 4.
- If n=0, there will be no processing, and the contents n-points from (D) will not change.

Operation error

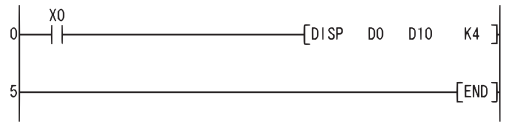
- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4100	The value of n is other than 0 to 4.	○	○	○	○	○	○
4101	The range n-points from (D) exceeds the range of the corresponding device.	○	○	○	○	○	○

Program example

• The following program dissociates the 16-bit data from D0 into 4-bit groups, and stores from D10 to D13 when X0 is ON.

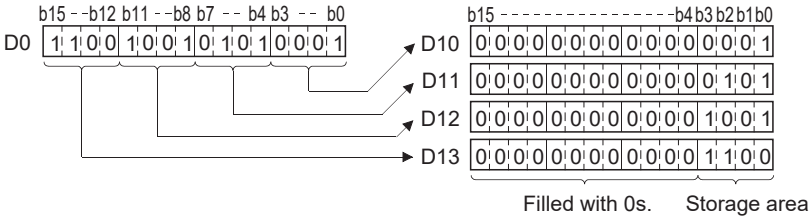
[Ladder Mode]



[List Mode]

Step	Instruction	Device		
0	LD	X0		
1	DISP	D0	D10	K4
5	END			

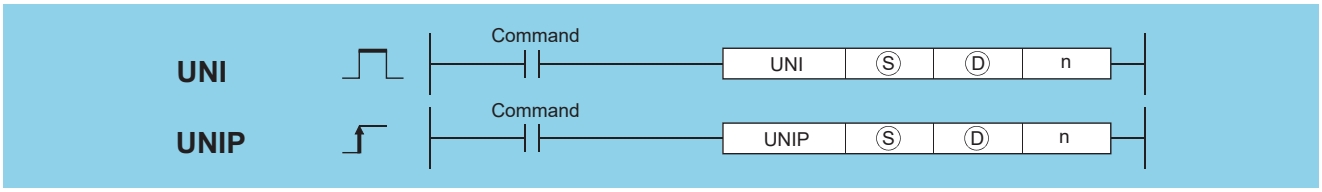
[Operation]



4-bit linking of 16-bit data

UNI(P)

Basic | High performance | Process | Redundant | Universal | LCPU

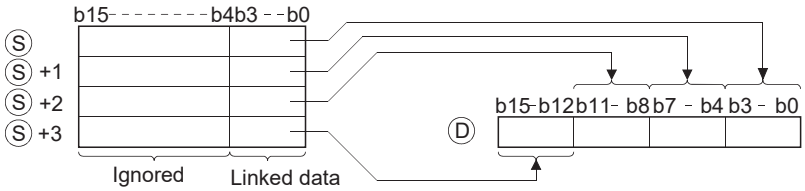


(S): Head number of the devices where data to be linked is stored (BIN 16 bits)
(D): Head number of the devices where the linked data will be stored (BIN 16 bits)
n: Number of links (1 to 4), 0: No processing (BIN 16 bits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(S)	—	○		—				—	—
(D)	○	○		○				—	—
n	○	○		○				○	—

Processing details

- Links lower 4 bits of 16-bit data n-points from device designated by (S) to 16-bit device designated by (D).



- The bits of the upper (4-n) digits of the device designated by (D) become 0.
- The value of n can be designated at between 1 and 4.
- If n=0, there will be no processing, and the contents of device (D) will not change.

Operation error

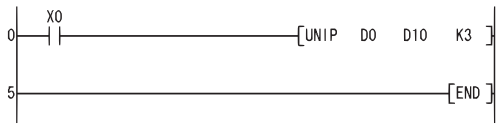
- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4100	The value of n is other than 0 to 4.	○	○	○	○	○	○
4101	The range n-points from (S) exceeds the range of the corresponding device.	○	○	○	○	○	○

Program example

• The following program links the lower 4 bits of D0 to D2 when X0 is ON, and stores them at D10.

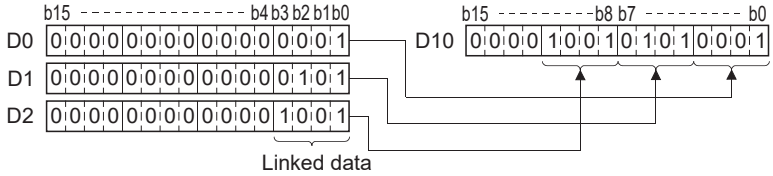
[Ladder Mode]



[List Mode]

Step	Instruction	Device		
0	LD	X0		
1	UNIP	D0	D10	K3
5	END			

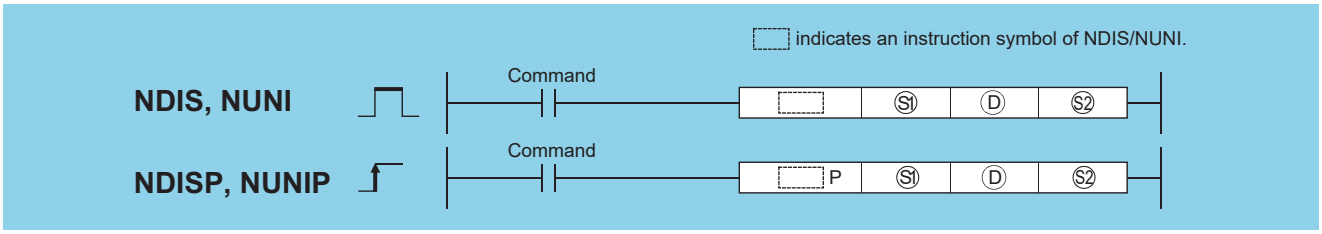
[Operation]



Dissociation of random data, linking of random data

NDIS(P), NUNI(P)

Basic | High performance | Process | Redundant | Universal | LCPU



(S1): Head number of the devices where data to be dissociated/linked is stored (BIN 16 bits)
(D): Head number of the devices where the dissociated/linked data will be stored (BIN 16 bits)
(S2): Head number of the devices where the units of dissociation/linking will be stored (BIN 16 bits)

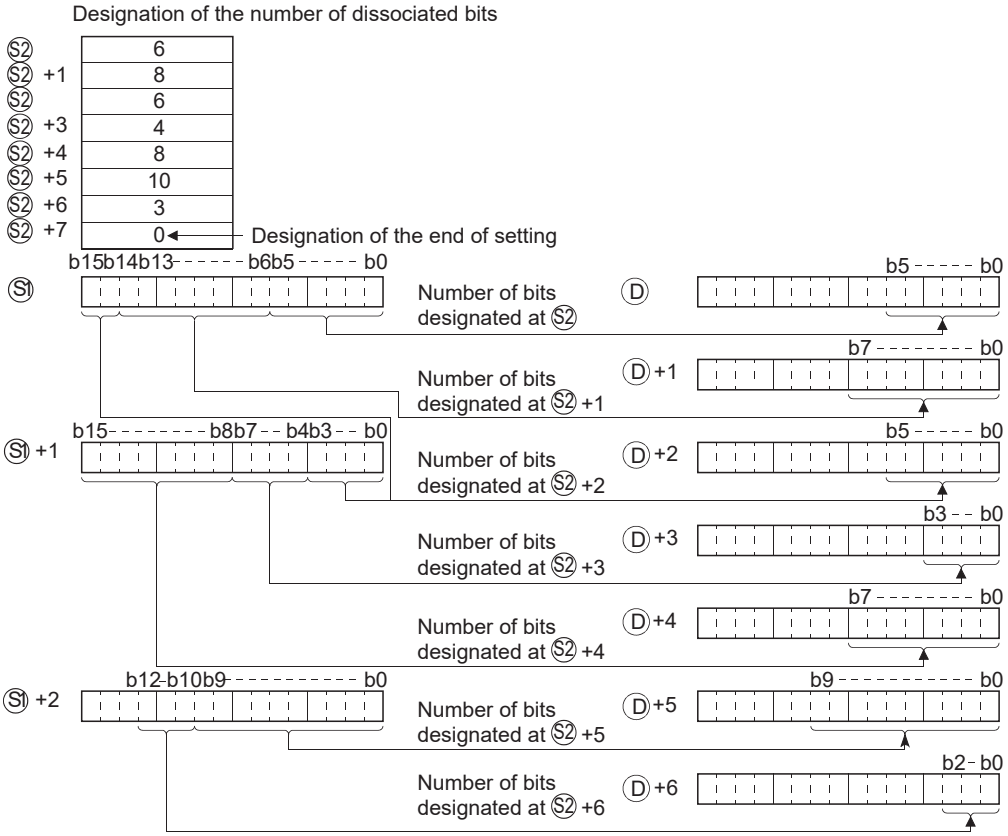
Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant	Others
	Bit	Word		Bit	Word				
(S1)	—	○		—					
(D)	—	○		—					
(S2)	—	○		—					

7

Processing details

■NDIS

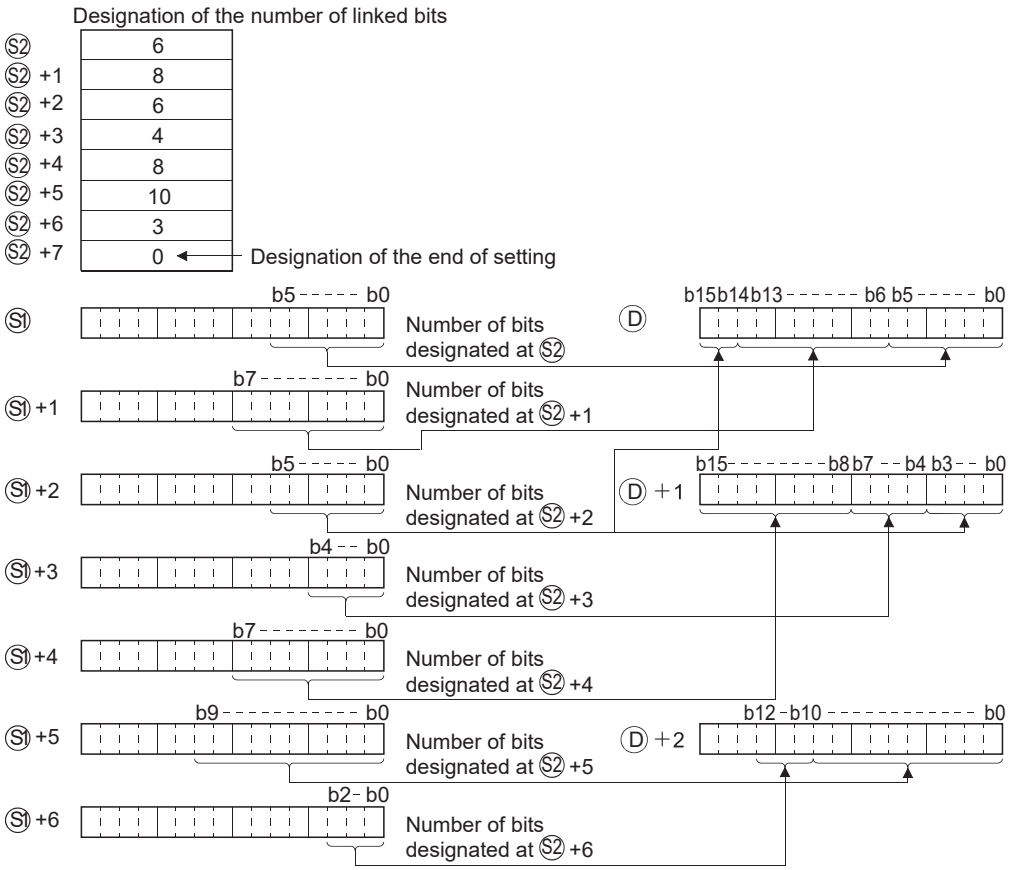
- Dissociates data stored in device numbers starting from that designated at (S1) into the number of individual bits designated at (S2), and stores this data in device numbers starting from that designated at (D).



- The number of dissociated bits designated at (S2) can be designated within a range of 1 to 16 bits.
- Bits from the device number designated at (S2) to the device number where "0" is stored are processed as dissociated bits.
- Do not overlap the device range for data to be dissociated ((S1) to end range of (S1)) with the device range which stores the dissociated data ((D) to end range of (D)). If overlapped, the correct operation result may not be obtained.
- Do not specify the same device number for (S1), (S2), and (D). If the same device is specified for (S1), (S2), and (D), the operation does not work correctly.

■NUNI

- Links individual bits of data stored into the area starting from the device number designated by (S1) in the number of bits specified by (S2), and stores them following the device number designated by (D).



- The number of bits to be linked as designated by (S2) can be within a range of from 1 to 16.
- Processing will be performed on the number of bits to be linked from the device number designated by (S2) to the device number storing "0".
- Do not overlap the device range for data to be linked ((S1) to end range of (S1)) with the device range which stores the linked data ((D) to end range of (D)). If overlapped, the correct operation result may not be obtained.
- Do not overlap the device numbers to be designated at (S1), (S2), and (D). If overlapped, correct operation is not possible.

Operation error

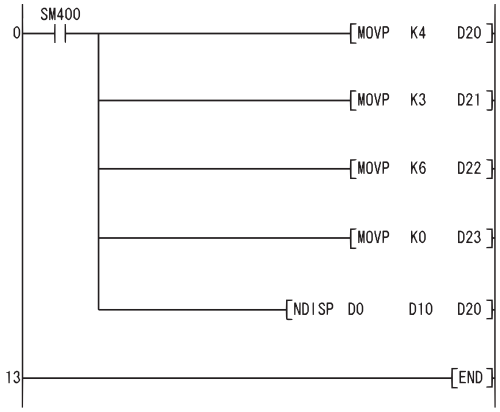
- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4100	The number of dissociated or linked bits specified by (S2) has not been set within the range from 1 to 16 bits.	○	○	○	○	○	○
4101	The device number of the device specified by (S1) or (D) based on the number of dissociated or linked bits specified by (S2) is greater than the last device number of each device.	○	○	○	○	○	○

Program example

- The following program dissociates data of 4, 3, and 6 bits respectively from the lower bits of D0, and stores them from D10 to D12.

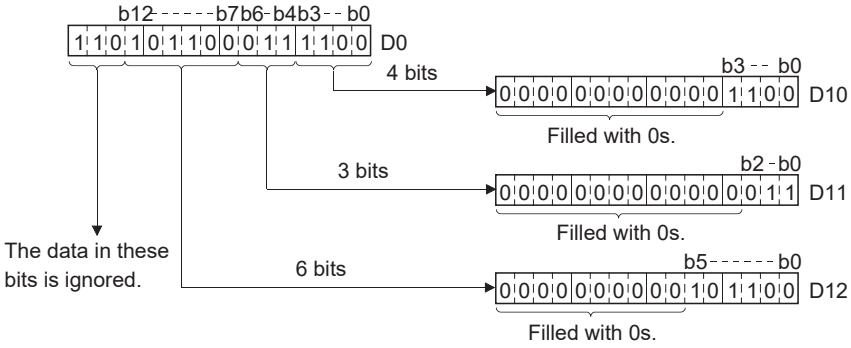
[Ladder Mode]



[List Mode]

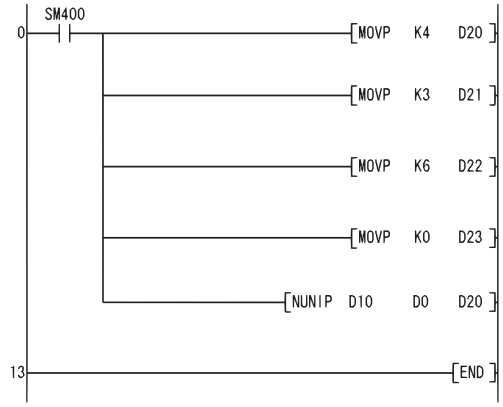
Step	Instruction	Device		
0	LD	SM400		
1	MOVP	K4	D20	
3	MOVP	K3	D21	
5	MOVP	K6	D22	
7	MOVP	K0	D23	
9	NDISP	D0	D10	D20
13	END			

[Operation]



• The following program links the lower 4 bits of data from D10, the lower 3 bits of data from D11, and the lower 6 bits of data from D12, and stores at D0.

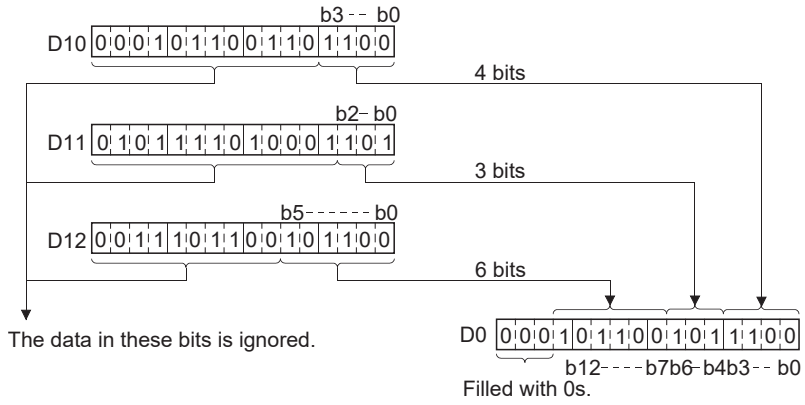
[Ladder Mode]



[List Mode]

Step	Instruction	Device		
0	LD	SM400		
1	MOVP	K4	D20	
3	MOVP	K3	D21	
5	MOVP	K6	D22	
7	MOVP	K0	D23	
9	NUNIP	D10	D0	D20
13	END			

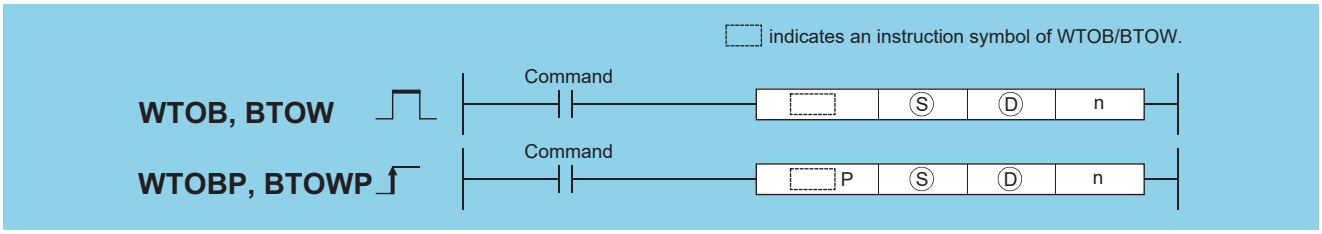
[Operation]



Data dissociation in byte units, data linking in byte units

WTOB(P), BTOW(P)

Basic | High performance | Process | Redundant | Universal | LCPU



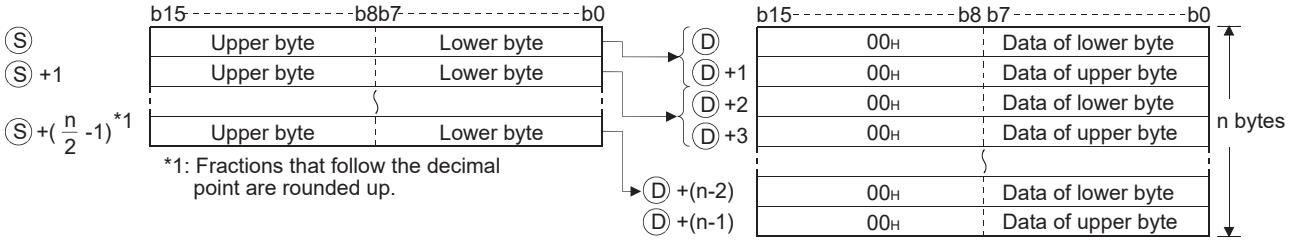
(S): Head number of the devices where data to be dissociated/linked in byte units is stored (BIN 16 bits)
(D): Head number of the devices where the result of dissociated/linking in byte units will be stored (BIN 16 bits)
n: Number of byte data to be dissociated/linked (BIN 16 bits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(S)	—	○		—					—
(D)	—	○		—					—
n	○	○		○					—

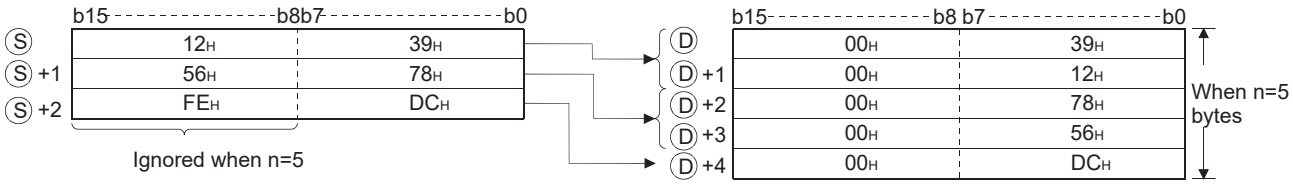
Processing details

■WTOB

- Dissociates n-bytes of the 16-bit data stored into the area starting from the device number designated by (S), and stores them following the device designated by (D).

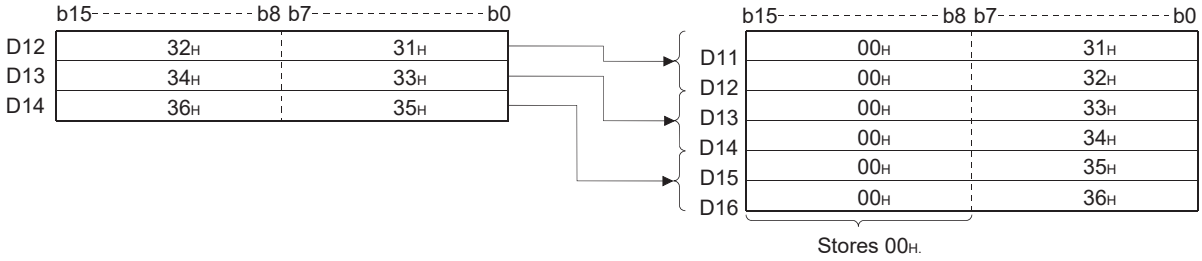


For example, if n=5, data through the lower 8 bits of (S) to ((S)+2) would be stored from ((D) to (D)+4).



- Setting the number of bytes with n automatically determines the range of the 16-bit data designated by (S) and the range of the devices to store the byte data designated by (D).
- No processing will be conducted when the number of bytes designated by n is "0".

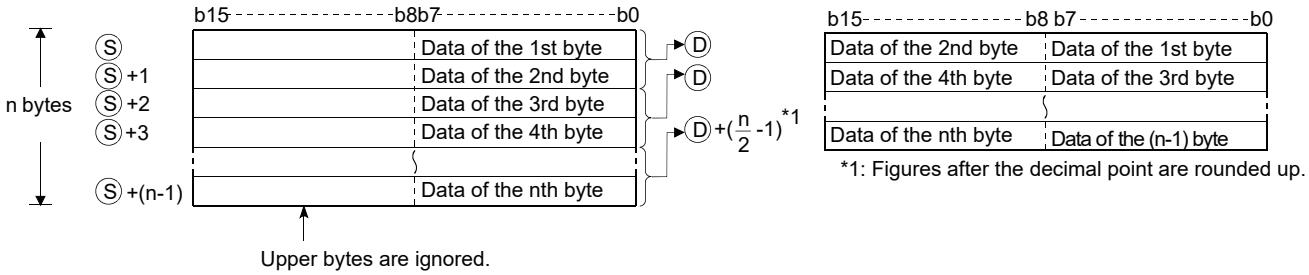
• The "00H" code will automatically be stored at the upper 8 bits of the byte storage device designated by (D).



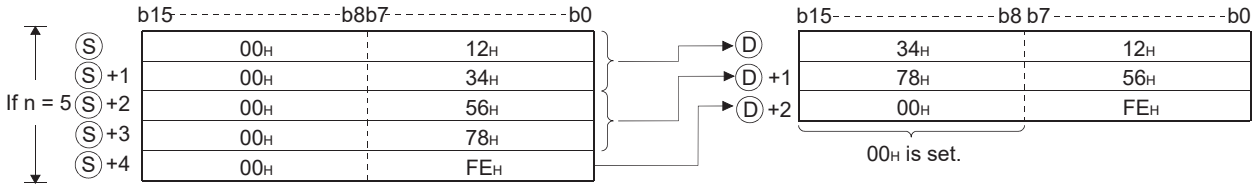
• Even though the range of devices with the data to be dissociated ((S) to (S)+(n ÷ 2 - 1)) is the same as the range of devices for storing dissociated data ((D) to (D)+(n-1)), the instruction operates correctly.

■BTOW

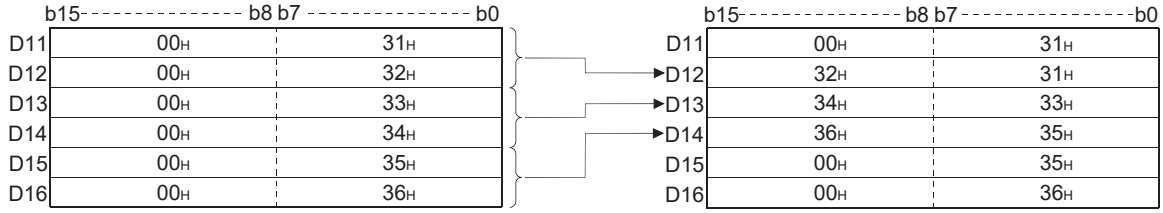
• Links the lower 8 bits of the 16-bit data in n words stored in the area starting from the device designated by (S) in 1-word units and stores it into the area starting from the device designated by (D). The upper 8 bits of n-word data stored in the area starting from the device designated by (S) will be ignored. Further, if n is an odd number, 0 is stored at the upper 8 bits of the device where the nth byte data is stored.



For example, if n=5, the lower 8 bits of data from (S) to ((S)+4) are linked and stored at (D) to ((D)+2).



• Setting the number of bytes with n automatically determines the range of the byte data designated by (S) and the range of the devices to store the linked data designated by (D).
• No processing will be conducted when the number of bytes designated by n is "0".
• The upper 8 bits of the byte storage device designated by (S) are ignored, and the lower 8 bits are used.
• Linking is correctly processed even when the device range ((S) to (S)+(n-1)) where the data to be linked is stored overlaps with the device range ((D) to (D)+(n ÷ 2 - 1)) where the linked data will be stored. For example, the following will take place in a case where the lower 8 bits of D11 to D16 are to be stored at D12 to D14:



Operation error

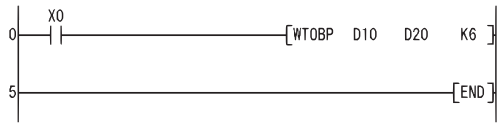
• In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4101	The range of the values in n exceeds that of the device specified by (S). The range of the values in n exceeds that of the device specified by (D).	○	○	○	○	○	○

Program example

• The following program dissociates the data at D10 to D12 in byte units and stores it at D20 to D25 when X0 is turned ON.

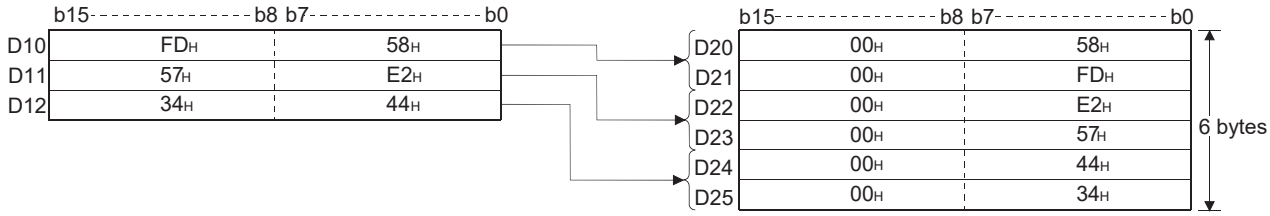
[Ladder Mode]



[List Mode]

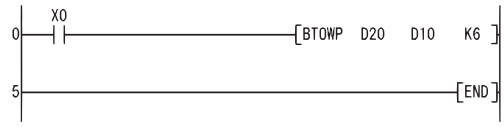
Step	Instruction	Device		
0	LD	X0		
1	WTOBP	D10	D20	K6
5	END			

[Operation]



• The following program links the lower 8 bits of data from D20 through D25 and stores the result at D10 to D12 when X0 is turned ON.

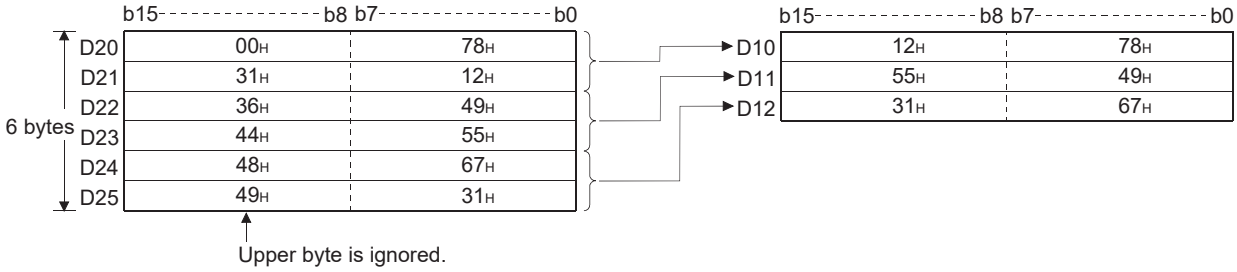
[Ladder Mode]



[List Mode]

Step	Instruction	Device		
0	LD	X0		
1	BTOWP	D20	D10	K6
5	END			

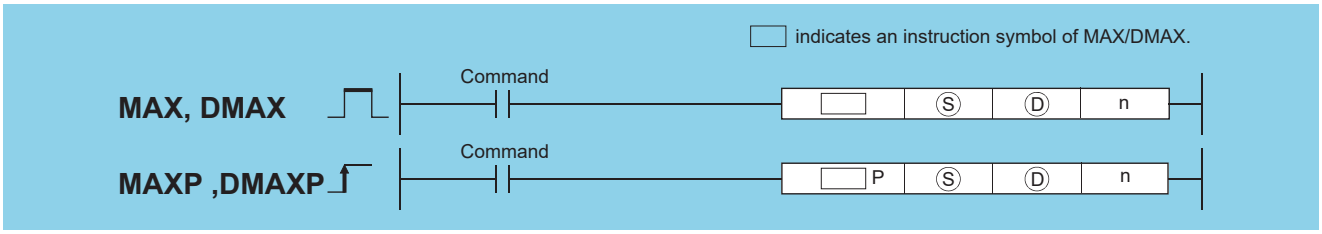
[Operation]



Maximum value search for 16-bit data, maximum value search for 32-bit data

MAX(P), DMAX(P)

Basic | High performance | Process | Redundant | Universal | LCPU



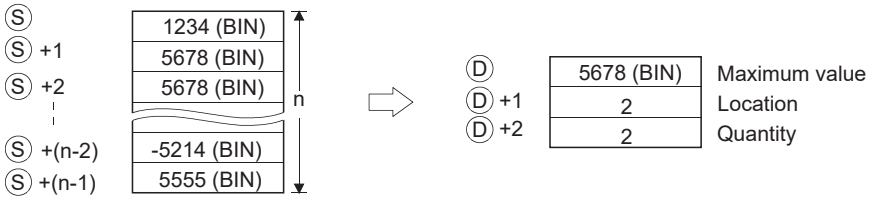
(S): Start device where the data to be searched is stored (BIN 16/32 bits)
(D): Start device where the maximum value search result will be stored (BIN 16/32 bits)
n: Number of data blocks to be searched (BIN 16 bits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(S)	—	○		—					—
(D)	—	○		—					—
n	○	○		○					—

Processing details

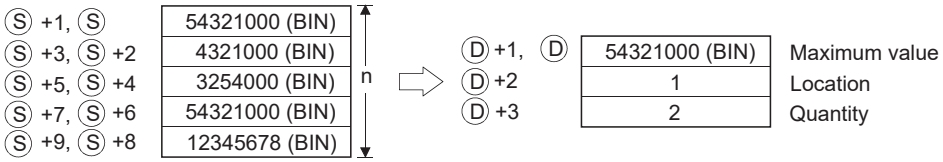
■MAX

- Searches in the n points of 16-bit BIN data, from the device designated by (S), for the maximum value and stores the searched maximum value at the device designated by (D).
- Starts the search from the device designated by (S) and stores the location, specified in the number of points counted from (S), of the device where the maximum value is found first at (D)+1 and stores the number of the found minimum values at (D)+2.



■DMAX

- Searches in the n points of 32-bit BIN data, from the device designated by (S), for the maximum value and stores the searched maximum value at the device designated by (D) and (D)+1.
- Starts the search from the device designated by (S) and stores the location, specified in the number of points counted from (S), of the device where the maximum value is found first at (D)+2 and stores the number of the found minimum values at (D)+3.



Operation error

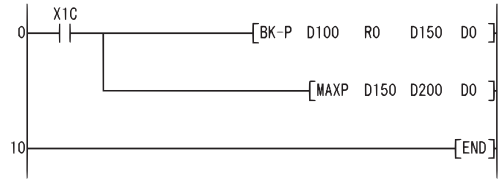
- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4101	The points specified in n exceed those of the corresponding device specified in (S).	○	○	○	○	○	○
4101	The points of the device specified in (D) exceed those of the corresponding device.	—	—	—	—	○	○

Program example

- The following program subtracts, when X1C is turned ON, the data stored at D100 to D103 from the data stored at R0 to R3, and searches in the results of subtraction for the maximum value, then, stores it at D200 to D202.

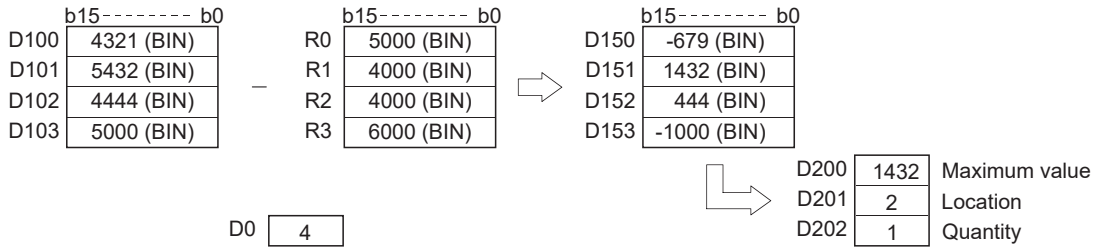
[Ladder Mode]



[List Mode]

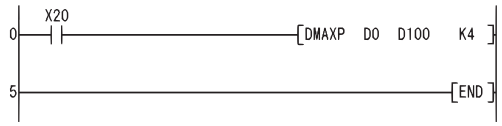
Step	Instruction	Device			
0	LD	X1C			
1	BK-P	D100	R0	D150	D0
6	MAXP	D150	D200	D0	
10	END				

[Operation]



- The following program searches for the maximum value from the 32-bit data at D0 to D7, and stores it at D100 to D103 when X20 is turned ON.

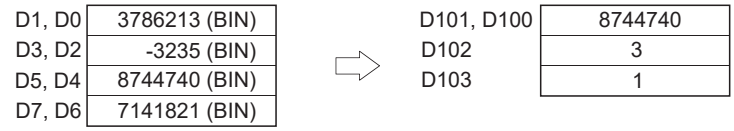
[Ladder Mode]



[List Mode]

Step	Instruction	Device		
0	LD	X20		
1	DMAXP	D0	D100	K4
5	END			

[Operation]



Minimum value search for 16-bit data, minimum value search for 32-bit data

MIN(P), DMIN(P)

Basic | High performance | Process | Redundant | Universal | LCPU

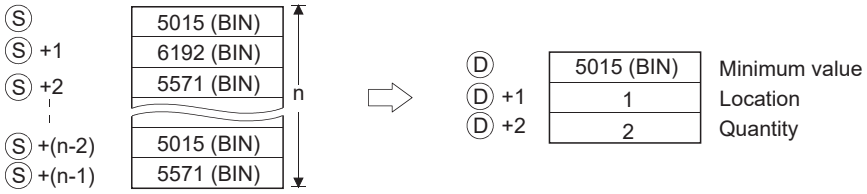
(S): Start device where the data to be searched is stored (BIN 16/32 bits)
(D): Start device where the minimum value search result will be stored (BIN 16/32 bits)
n: Number of data blocks to be searched (BIN 16 bits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(S)	—	○		—					—
(D)	—	○		—					—
n	○	○		○					—

Processing details

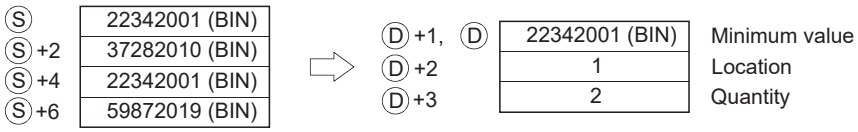
■MIN

- Searches in the n points of 16-bit BIN data, from the device designated by (S), for the minimum value and stores searched minimum value at the device designated by (D).
- Starts the search from the device designated by (S) and stores the location, specified in the number of points counted from (S), of the device where the minimum value is found first at (D)+1 and stores the number of the found minimum values at (D)+2.



■DMIN

- Searches in the n points of 32-bit BIN data, from the device designated by (S), for the minimum value and stores searched minimum value at the devices designated by (D) and (D)+1.
- Starts the search from the device designated by (S) and stores the location, specified in the number of points counted from (S), of the device where the minimum value is found first at (D)+2 and stores the number of the found minimum values at (D)+3.



Operation error

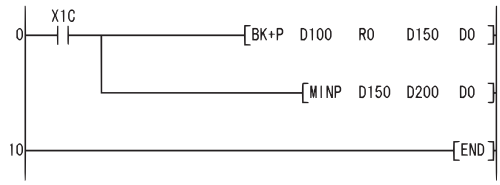
- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4101	The points specified in n exceed those of the corresponding device specified in (S).	○	○	○	○	○	○
	The device specified by (D) exceeds the range of the corresponding device.	—	—	—	—	○	○

Program example

- The following program adds, when X1C is turned ON, the data stored at D100 to D103 and the data stored at R0 to R3, and searches in the results of addition for the minimum value, then, stores it at D200 to D202.

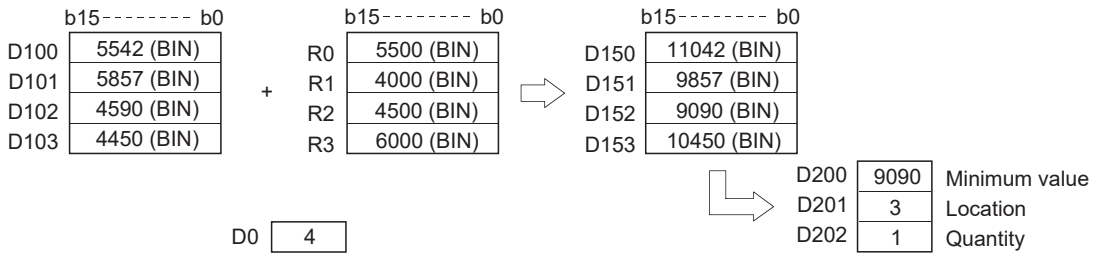
[Ladder Mode]



[List Mode]

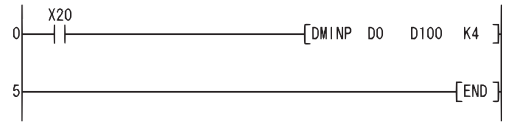
Step	Instruction	Device			
0	LD	X1C			
1	BK+P	D100	R0	D150	D0
6	MINP	D150	D200	D0	
10	END				

[Operation]



- The following program, when X20 is turned ON, searches for the minimum value from the 32-bit data contained from D0 to D7, and stores it from D100 to D103.

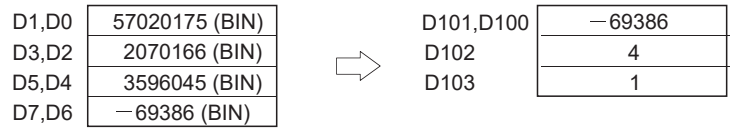
[Ladder Mode]



[List Mode]

Step	Instruction	Device		
0	LD	X20		
1	DMINP	D0	D100	K4
5	END			

[Operation]

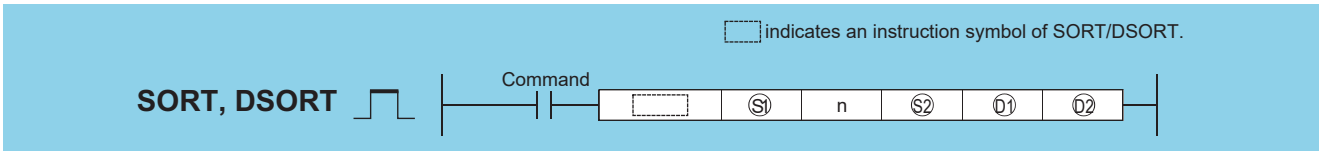


7

BIN 16-bit data sort operations, BIN 32-bit data sort operations

SORT, DSORT

Basic | High performance | Process | Redundant | Universal | LCPU



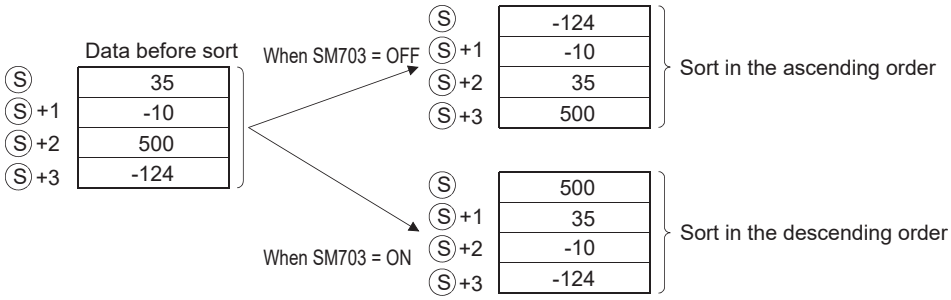
(S1): Head device number in the table to be sorted (BIN 16/32 bits)
n: Number of data blocks to be sorted (BIN 16 bits)
(S2): Number of data blocks to be compared in one sort operation (BIN 16 bits)
(D1): Number of the bit device to be turned ON at the completion of the sort operation (bits)
(D2): Device reserved for the system (BIN 16 bits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(S1)	—	○		—					—
n	○	○		○					—
(S2)	○	○		○					—
(D1)	○	○ (Other than T, ST, C)		—					—
(D2)	—	○		—					—

Processing details

■SORT

- Sorts (rearranges data) BIN 16-bit data n points from (S1) in ascending or descending order. Sort order is designated by the ON/OFF status of SM703:
  - When SM703 is OFF: Ascending order sort
  - When SM703 is ON: Descending order sort



- Several scans are required for sorts performed by the SORT instruction. The number of scans executed until completion is the value obtained by dividing the maximum number of times executed until the completion of the sort by the number of data blocks compared at one execution designated by (S2). (Decimal fractions are rounded up.) When the value of (S2) is increased, the number of scans until completion of the sort is reduced, but the amount of time per scan is lengthened.

• The maximum number of executions until completion of the sort should be calculated according to the following equation:

The maximum number of executions until completion = (n) × (n - 1) / 2 [times executed]

Ex.

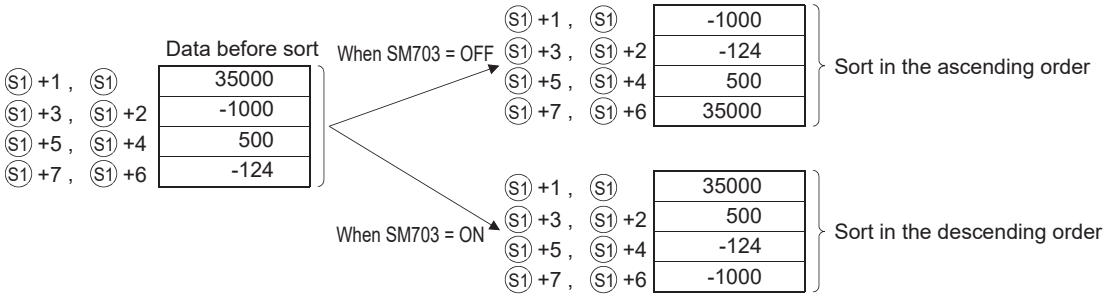
When n=10, the number of executions is obtained as 10 × (10 - 1) / 2=45 [times executed].

If (S2)=2, then the number of scans until the completion of sort is calculated as 45/2=22.5 → 23 [scans].

- The device designated by (D1) (the completion device) is turned OFF by the execution of the SORT instruction, and turned ON when the sort is completed. Because the device designated by (D1) is maintained in the ON state after the completion of the sort, the user must turn it OFF if required.
- The 2 points from the device designated by (D2) are used by the system during the execution of the SORT instruction. These 2 points from the device designated by (D2) should therefore not be used by the user. Changing these points may cause an error code to be returned (Error code: 4100).
- If the value of n is changed during the execution of the SORT instruction, the sort will be conducted in accordance with the number of sort data blocks after the change.
- If the execution command is turned OFF during the execution of the SORT instruction, the sort is suspended. The sort resumes from the beginning when the execution command is turned ON again.
- To execute another sort operation immediately after the completion of the previous sort, turn OFF the execution command once, then turn it ON.

■DSORT

- Sorts (rearranges data) BIN 32-bit data n points from (S1) in ascending or descending order. Sort order is designated by the ON/OFF status of SM703:
  - When SM703 is OFF: Ascending order sort
  - When SM703 is ON: Descending order sort



- Several scans are required for sorts performed by the DSORT instruction. The number of scans executed until completion is the value obtained by dividing the maximum number of times executed until the completion of the sort by the number of data blocks compared at one execution designated by (S2). (Decimal fractions are rounded up.) When the value of (S2) is increased, the number of scans until completion of the sort is reduced, but the amount of time per scan is lengthened.
- The maximum number of executions until completion of the sort should be calculated according to the following equation:

The maximum number of executions until completion = (n) × (n - 1) / 2 [times executed]

Ex.

When n=10, the number of executions is obtained as 10 × (10 - 1) / 2=45 [times executed].

If (S2)=2, then the number of scans until the completion of sort is calculated as 45/2=22.5 → 23 [scans].

- The device designated by (D1) (the completion device) is turned OFF by the execution of the SORT instruction, and turned ON when the sort is completed. Because the device designated by (D1) is maintained in the ON state after the completion of the sort, the user must turn it OFF if required.
- The 2 points from the device designated by (D2) are used by the system during the execution of a DSORT instruction. These 2 points from the device designated by (D2) should therefore not be used by the user. Changing these points may cause an error code to be returned (Error code: 4100).
- If the value of n is changed during the execution of the SORT instruction, the sort will be conducted in accordance with the number of sort data blocks after the change.
- If the execution command is turned OFF during the execution of the SORT instruction, the sort is suspended. The sort resumes from the beginning when the execution command is turned ON again.
- To execute another sort operation immediately after the completion of the previous sort, turn OFF the execution command once, then turn it ON.

Operation error

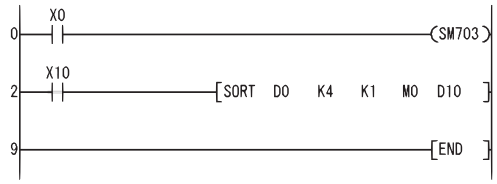
• In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4100	(S2) is 0 or a negative value. The values of (D2) used in the system at the 2nd scan or later are more than n. The values of (D2) used in the system at the 2nd scan or later are (D2)<(D2)+1.	○	○	○	○	○	○
4101	The range from (S1) to (S1)+n (including (S1)) overlaps the range from (D2) to (D2)+1.	○	○	○	○	○	○
	For the SORT(P) instruction, the range of the device specified by (S1) exceeds the range from (S1) to (S1)+n (including (S1)).	○	○	○	○	○	○
	For the DSORT(P) instruction, the range of the device specified by (S1) exceeds the range from (S1) to (S1)+(2×n) (including (S1)).	○	○	○	○	○	○

Program example

• The following program sorts the BIN 16-bit data from D0 to D3 in the ascending/descending order when X10 is turned ON.

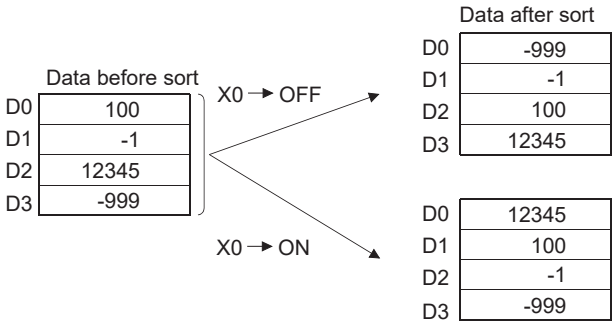
[Ladder Mode]



[List Mode]

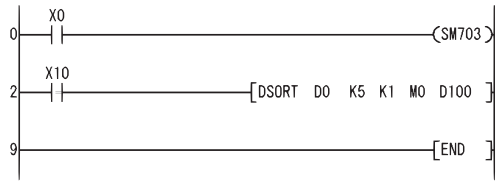
Step	Instruction	Device
0	LD	X0
1	OUT	SM703
2	LD	X10
3	SORT	D0 K4 K1 M0 D10
9	END	

[Operation]



• The following program sorts the BIN 32-bit data from D0 to D9 in ascending/descending order when X10 is turned ON.

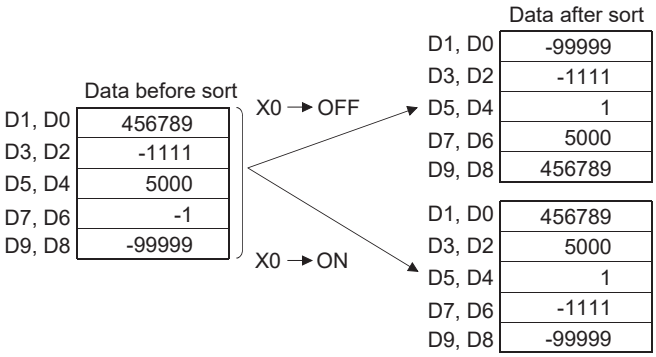
[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	X0
1	OUT	SM703
2	LD	X10
3	DSORT	D0 K5 K1 M0 D100
9	END	

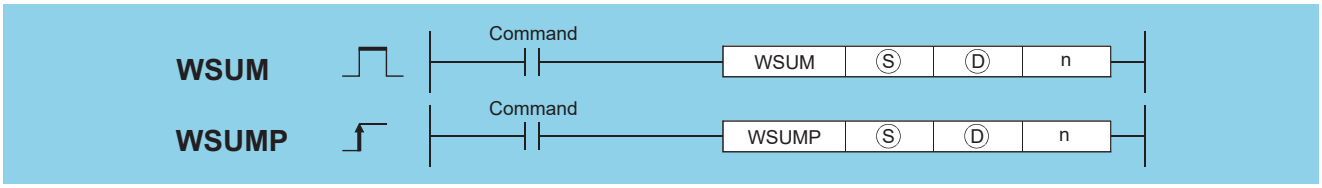
[Operation]



Calculation of totals for 16-bit data

WSUM(P)

Basic | High performance | Process | Redundant | Universal | LCPU



(S): Head number of the devices where data to be summed are stored (BIN 16 bits)
(D): Head number of the devices where the sum will be stored (BIN 32 bits)
n: Number of data blocks (BIN 16 bits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(S)	—	○		—				—	—
(D)	○	○		○				—	—
n	○	○		○				○	—

Processing details

• Adds all 16-bit BIN data for n blocks from the device designated at (S), and stores it in the device designated at (D).

Ⓢ	4444 (BIN)
Ⓢ + 1	3333 (BIN)
Ⓢ + 2	1234 (BIN)
Ⓢ + 3	−5426 (BIN)
Ⓢ + 4	329 (BIN)
Ⓢ + 5	10000 (BIN)

n

⇨ Ⓓ + 1, Ⓓ 13914 (BIN)

Operation error

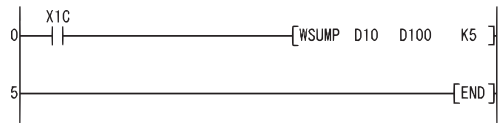
• In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4101	The points specified in n exceed those of the corresponding device specified in (S).	○	○	○	○	○	○

Program example

• The following program adds the 16-bit BIN data from D10 to D14, and stores it in D100 and D101 when X1C is turned ON.

[Ladder Mode]



[List Mode]

Step	Instruction	Device		
0	LD	X1C		
1	WSUMP	D10	D100	K5
5	END			

[Operation]

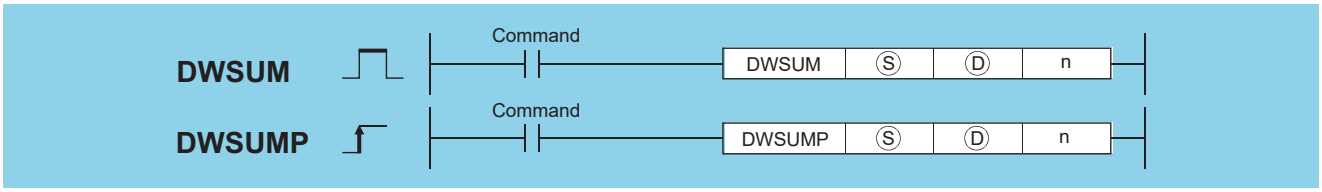
D10	4500 (BIN)
D11	2500 (BIN)
D12	−3276 (BIN)
D13	6780 (BIN)
D14	4444 (BIN)

⇨ D101,D100 14948 (BIN)

Calculation of totals for 32-bit data

DWSUM(P)

Basic | High performance | Process | Redundant | Universal | LCPU



(S): Head number of the devices where data to be summed are stored (BIN 32 bits)
(D): Head number of the devices where the sum will be stored (BIN 64 bits)
n: Number of data blocks (BIN 16 bits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(S)	—	○		—				—	—
(D)	—	○		—				—	—
n	○	○		○				○	—

Processing details

- Adds all 32-bit BIN data stored in n points of devices starting from the one designated by (S), and stores the result to 4 points of devices (4 words) starting from the one designated by (D).

Ⓢ + 1, Ⓢ	32767000 (BIN)
Ⓢ + 3, Ⓢ + 2	6000 (BIN)
Ⓢ + 5, Ⓢ + 4	35392000 (BIN)
Ⓢ + 7, Ⓢ + 6	−11870000 (BIN)
Ⓢ + 9, Ⓢ + 8	12345000 (BIN)

n

⇨ Ⓓ + 3 to Ⓓ 68640000 (BIN)

Operation error

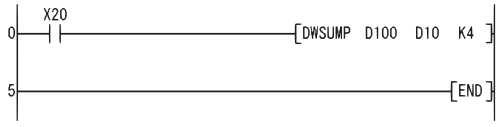
- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4101	The points specified in n exceed those of the corresponding device specified in (S).	○	○	○	○	○	○
	The device specified in (D) exceeds the range of the corresponding device.	—	—	—	—	○	○

Program example

- The following program adds the 32-bit BIN data at D100 to D107, and stores the result at D10 and D13 when X20 is turned ON.

[Ladder Mode]



[List Mode]

Step	Instruction	Device		
0	LD	X20		
1	DWSUMP	D100	D10	K4
5	END			

[Operation]

D101,D100	11245600 (BIN)
D103,D102	27543200 (BIN)
D105,D104	558800 (BIN)
D107,D106	−15675000 (BIN)

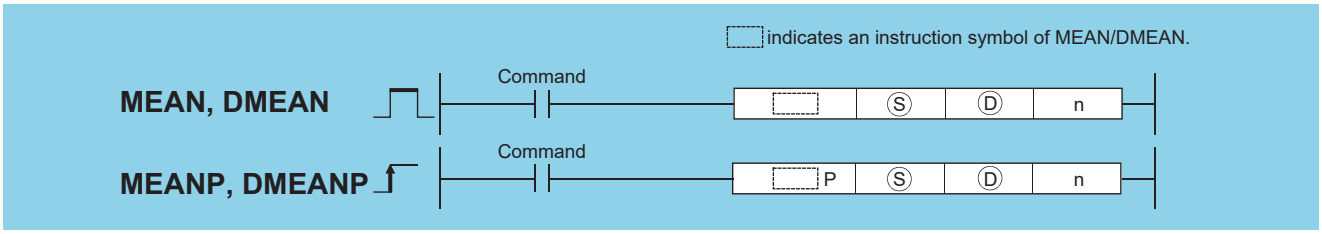
⇨ D13 to D10 23672600 (BIN)

Calculation of averages for 16-bit data, calculation of averages for 32-bit data

MEAN(P), DMEAN(P)

Basic (×) | High performance (×) | Process (×) | Redundant (×) | Universal (Ver.) | LCPU

- QnU(D)(H)CPU, QnUDE(H)CPU: the serial number (first five digits) is "10102" or later
- Q00UJCPU, Q00UCPU, Q01UCPU, QnUDVCPU, QnUDPVCPU: Supported



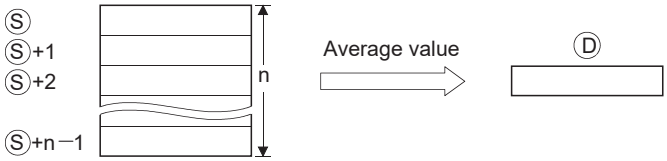
(S): Head number of the devices where the data to be averaged are stored (BIN16/32 bits)
(D): Head number of the devices where the average will be stored (BIN 16/32 bits)
n: Number of data or number of the devices where the number of data are stored (Setting range: 1 to 32767) (BIN 16 bits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(S)	—	○	○	—				—	—
(D)	—	○	○	—				—	—
n	○	○	○	○				○	—

Processing details

■MEAN(P)

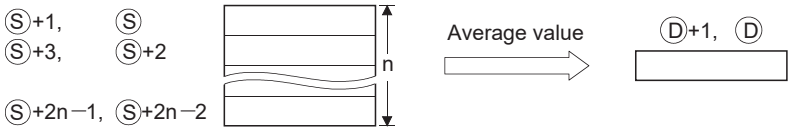
- This instruction calculates the mean of 16-bit BIN data stored in n-point devices starting from the device specified by (S), and then stores the result into the device specified by (D).



- If the value calculated is not integer, this instruction will drop the number of decimal places.
- If the value specified by n is 0, the instruction will be not processed.

■DMEAN(P)

- This instruction calculates the mean of 32-bit BIN data stored in n-point devices starting from the device specified by (S), and then stores the result into the device specified by (D).



- If the value calculated is not integer, this instruction will drop the number of decimal places.
- If the value specified by n is 0, the instruction will be not processed.

Operation error

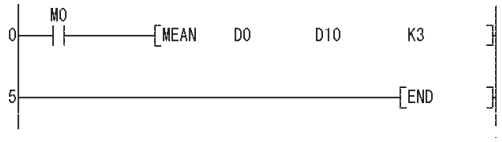
• In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4100	The value specified in n is other than 0 to 32767.	—	—	—	—	○	○
4101	The points specified in n exceed those of the corresponding device specified in (S).	—	—	—	—	○	○

Program example

• The following program stores the average value of 16-bit data stored from D0 to D2 into D10, when M0 is turned on.

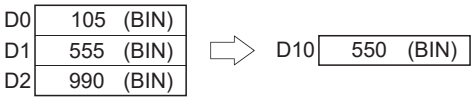
[Ladder Mode]



[List Mode]

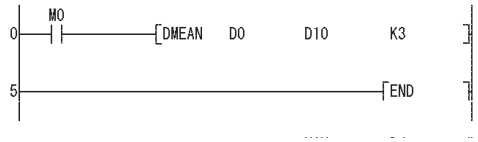
Step	Instruction	Device
0	LD	M0
1	MEAN	D0 D10 K3
5	END	

[Operation]



• The following program stores the average value of 32-bit data stored from D0 to D5 into D10 and D11, when M0 is turned on.

[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	M0
1	DMEAN	D0 D10 K3
5	END	

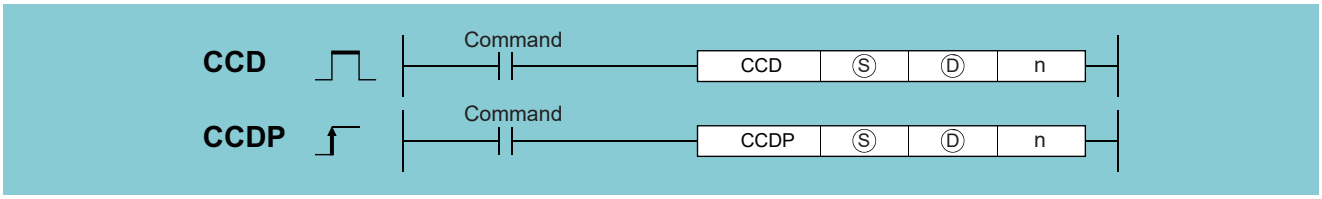
[Operation]

Check code

CCD(P)

Basic | High performance | Process | Redundant | Ver. Universal | Ver. LCPU

• QnUDVCPU, QnUDPVCPU, LCPU: The serial number (first five digits) is "16112" or later.



(S): Start number of target word device (BIN 16 bits)
(D): Start number of word device in which calculated data is to be stored (BIN 16 bits)
n: Number of data blocks (setting range: 1 to 256) (BIN 16 bits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(S)	—	○	○	—				—	—
(D)	—	○	○	—				—	—
n	—	○	○	—				○	—

Processing details

• Performs addition of the data stored in the devices specified by (S) to (S)+(n)-1 and calculates the horizontal parity, and stores the added data in the device specified by (D) and the horizontal parity in the device specified by (D)+1. These instructions support two modes, 16-bit conversion mode and 8-bit conversion mode, used for calculation. The two conversion modes can be selected by SM772 ON/OFF. The operation in each conversion mode is outlined below.

■16-bit conversion mode (when SM772 is OFF)

Eight-bit data each above and below the n points of data starting from (S) is added, and the added data and horizontal parity data are stored in the devices (D) and (D)+1 respectively. An example when n = 6 is shown below.

• Calculation of addition data value

In 16-bit conversion mode, added data is obtained by adding 6 bytes at the shaded positions in the following figure. Here, added data is "0315H" and accordingly "0315H" is stored in the device specified by (D).

Device	Decimal	Hexadecimal	
		Upper bits	Lower bits
D0	24932	61H	64H
D1	4219	10H	7BH
D2	-1333	FAH	CBH
D3	-1	FFH	FFH
D4	32761	7FH	F9H
D5	10000	27H	10H

• Calculation of horizontal parity value

In 16-bit conversion mode, the shaded part in the following figure becomes the calculation target of the horizontal parity. The number of ON (1) bits is calculated to determine the parity value which becomes ON (1) when the number of ON (1) bits is finally odd or OFF (0) when it is finally even, and the horizontal parity value is stored in the device specified by (S)+1.

Device	b7	b6	b5	b4	b3	b2	b1	b0	
Upper 8 bits of D0	0	1	1	0	0	0	0	1	61H
Lower 8 bits of D0	0	1	1	0	0	1	0	0	64H
Upper 8 bits of D1	0	0	0	1	0	0	0	0	10H
Lower 8 bits of D1	0	1	1	1	1	0	1	1	7BH
Upper 8 bits of D2	1	1	1	1	1	0	1	0	7AH
Lower 8 bits of D2	1	1	0	0	1	0	1	1	CBH
Horizontal parity	0	1	0	1	1	1	1	1	5FH

⇦ This value is stored in (D)+1.

■8-bit conversion mode (when SM772 is ON)

• For the n points of data (lower 8 bits only) starting from (S), the added data and horizontal parity data are stored in the devices (D) and (D)+1 respectively. An example when n = 6 is shown below.

• Calculation of addition data value

In 8-bit conversion mode, added data is obtained by adding 6 bytes at the shaded positions in the following figure. Here, added data is "03B2H" and accordingly "03B2H" is stored in the device specified by (D).

Device	Decimal	Hexadecimal	
		Upper bits	Lower bits
D0	24932	61H	64H
D1	4219	10H	7BH
D2	-1333	FAH	CBH
D3	-1	FFH	FFH
D4	32761	7FH	F9H
D5	10000	27H	10H

• Calculation of horizontal parity value

In 8-bit conversion mode, the shaded part in the following figure becomes the calculation target of the horizontal parity. The number of ON (1) bits is calculated to determine the parity value which becomes ON (1) when the number of ON (1) bits is finally odd or OFF (0) when it is finally even, and the horizontal parity value is stored in the device specified by (S)+1.

Device	b7	b6	b5	b4	b3	b2	b1	b0	
Lower 8 bits of D0	0	1	1	0	0	1	0	0	64H
Lower 8 bits of D1	0	1	1	1	1	0	1	1	7BH
Lower 8 bits of D2	1	1	0	0	1	0	1	1	CBH
Lower 8 bits of D3	1	1	1	1	1	1	1	1	FFH
Lower 8 bits of D4	1	1	1	1	1	0	0	1	F9H
Lower 8 bits of D5	0	0	0	1	0	0	0	0	10H
Horizontal parity	1	1	0	0	0	0	1	0	C2H

⇦ This value is stored in (D)+1.

Operation error

- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4101	The value in the device specified by the instruction exceeds the setting range.	—	—	—	—	○	○

Program example

- The sample program calculates the added data and horizontal parity of six points of data starting from D0 and stores the results in the devices specified by D100 and D101.

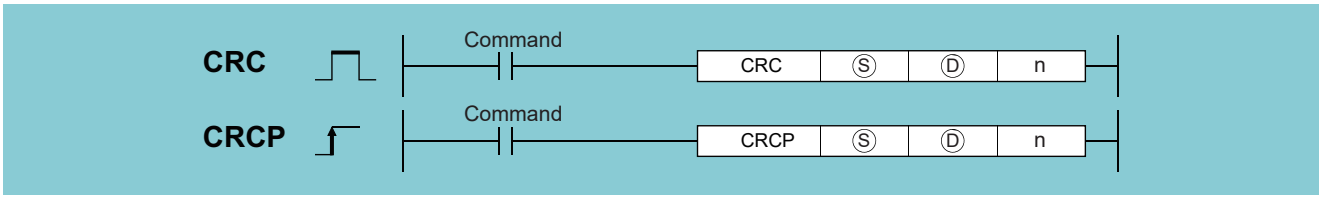
[Ladder Mode]

CRC operation

CRC(P)

Basic | High performance | Process | Redundant | Ver. Universal | Ver. LCPU

• QnUDVCPU, QnUDPVCPU, LCPU: The serial number (first five digits) is "16112" or later.



(S): Start number of the devices where the target data of CRC value generation is stored (BIN 16 bits)
(D): Number of the device where the CRC value generated is stored (BIN 16 bits)
n: Number of 8-bit data blocks for which the CRC value is to be determined or the number for the device where the number of 8-bit data blocks is stored (setting range: 1 to 256) (BIN 16 bits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(S)	—	○	○	—	—	—	—	—	—
(D)	—	○	○	—	—	—	—	—	—
n	—	○	○	—	—	—	—	○	—

7

Processing details

• Generates the CRC value of n points of 8-bit data (byte units) starting from the device specified by (S) and stores it in the device specified by (D). "$X^{16}+X^{15}+X^2+1$" is used as the generator polynomial to generate the CRC value. These instructions support two modes, 16-bit conversion mode and 8-bit conversion mode, used for calculation of the CRC value. The two conversion modes can be selected by SM772 ON/OFF. The operation in each conversion mode is described below.

■16-bit conversion mode (when SM772 is OFF)

CRC operation is performed for the upper 8 bits (byte units) and lower 8 bits (byte units) in the device specified by (S). The operation result is stored in the 16 bits of one point of the device specified by (D). The CRC value when n = 6 is shown below. In 16-bit conversion mode, the 6 bytes at the shaded positions in the following figure are the operation target. Here, the CRC value is "A57BH" and accordingly "A57BH" is stored in the device specified by (D).

Device	Decimal	Hexadecimal	
		Upper bits	Lower bits
D0	24932	61H	64H
D1	4219	10H	7BH
D2	-1333	FAH	CBH
D3	-1	FFH	FFH
D4	32761	7FH	F9H
D5	10000	27H	10H

■8-bit conversion mode (when SM772 is ON)

CRC operation is performed for the lower 8 bits (lower byte) of the data in the device specified by (S). The operation result is stored using the two points from the device specified by (D): lower 8 bits (byte units) are stored in the device specified by (D) and upper 8 bits (byte units) are stored in the device specified by (D)+1. In 8-bit conversion mode, the 6 bytes at the shaded positions in the following figure are the operation target. Here, the CRC value is "BDA1H" and accordingly "A1H" is stored in the device specified by (D) and "BDH" is stored in the device specified by (D)+1.

Device	Decimal	Hexadecimal	
		Upper bits	Lower bits
D0	24932	61H	64H
D1	4219	10H	7BH
D2	-1333	FAH	CBH
D3	-1	FFH	FFH
D4	32761	7FH	F9H
D5	10000	27H	10H

Operation error

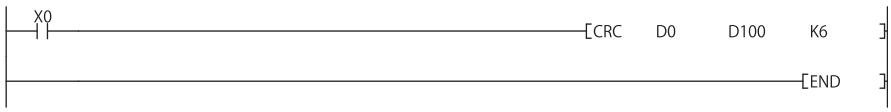
- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4100	A digit other than 4 is specified in the digit-specified bit device in (S), (D).	—	—	—	—	○	○
4101	The value in the device specified by the instruction exceeds the setting range. (S)+n-1, (D)+1 is out of the device range.	—	—	—	—	○	○

Program example

- The sample program calculates the CRC value of six points of data in the devices starting from D0 and stores the result in the devices specified by D100.

[Ladder Mode]

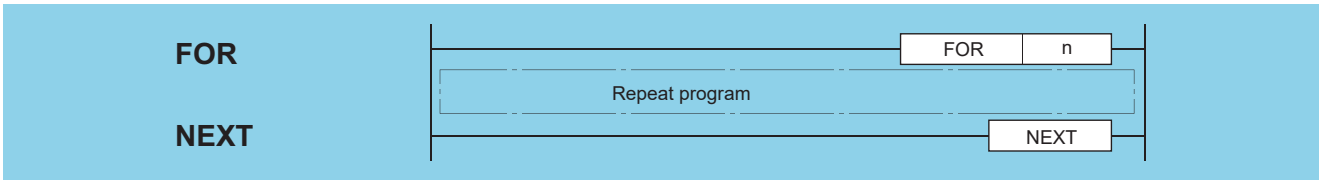


7.6 Structure Creation Instructions

FOR to NEXT instruction loop

FOR, NEXT

Basic | High performance | Process | Redundant | Universal | LCPU

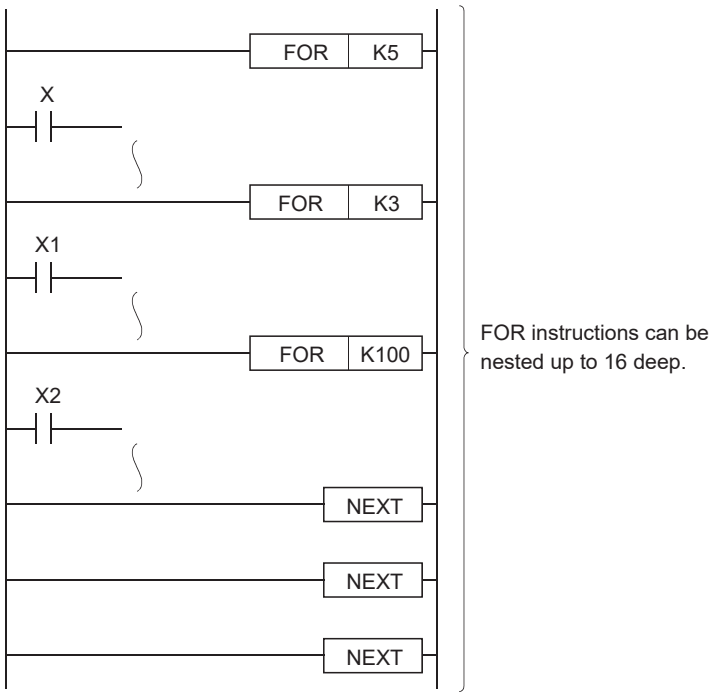


n: Number of repetitions of FOR to NEXT loop (1 to 32767) (BIN 16 bits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
n	○								—

Processing details

- When the processing in the FOR to NEXT loop is executed n-times without conditions, the step following the NEXT instruction will be executed.
- The value of n can be designated at between 1 and 32767. If it is designated from -32768 to 0, the processing which is executed when n=1 will be performed.
- If you do not desire to execute the processing called for within the FOR to NEXT loop, use the CJ or SCJ instruction to jump.
- FOR instructions can be nested up to 16 deep.



Operation error

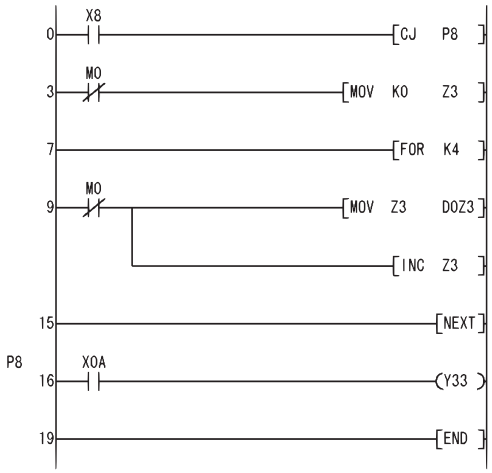
• In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4200	After the FOR instruction was executed, the END, FEND, or GOEND instruction was executed prior to the NEXT instruction. The STOP instruction is in process between the FOR and the NEXT instructions.	○	○	○	○	○	○
4201	The NEXT instruction was executed prior to the FOR instruction.	○	○	○	○	○	○
4202	The 17th FOR instruction was executed when the FOR instruction has been nested.	○	○	○	○	○	○

Program example

• The following program executes the FOR to NEXT loop when X8 is OFF, and does not execute it when X8 is ON.

[Ladder Mode]

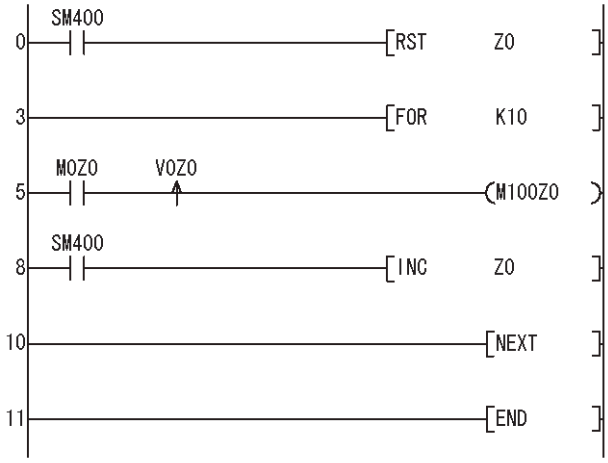


[List Mode]

Step	Instruction	Device	
0	LD	X8	
1	CJ	P8	
3	LDI	M0	
4	MOV	K0	Z3
7	FOR	K4	
9	LDI	M0	
10	MOV	Z3	DOZ3
13	INC	Z3	
15	NEXT		
16	P8		
17	LD	X0A	
18	OUT	Y33	
19	END		

Point

- To force an end to the repetitious execution of the FOR to NEXT loop during the execution of the loop, insert a BREAK instruction. See Page 462 Forced end of FOR to NEXT instruction loop for details concerning the use of the BREAK instruction.
- Use the EGP/EGF instruction to perform the pulse operation of an index-modified program between the FOR and NEXT instructions. Note, however, that rise and fall instructions are not available on the operation output side. Refer to Page 148 Pulse conversion of edge relay operation results for details of the EGP/EGF instruction. The program samples are shown below:

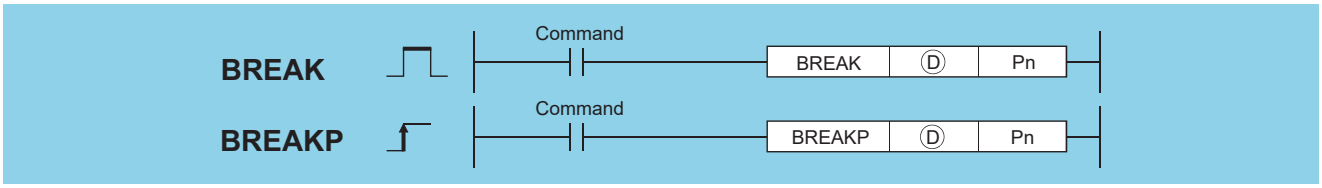


- Branching into a FOR to NEXT loop using a JMP or other branch instruction from the outside of the FOR to NEXT loop is not possible.

Forced end of FOR to NEXT instruction loop

BREAK(P)

Basic | High performance | Process | Redundant | Universal | LCPU

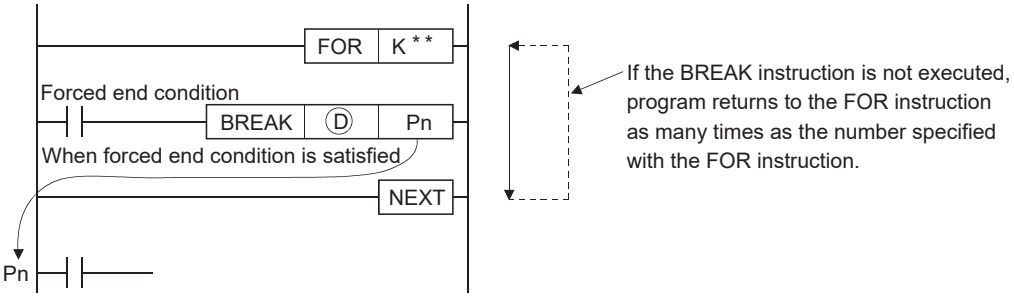


(D): Number of the device where the remaining number of loops will be stored (BIN 16 bits)
Pn: Number of the pointer (device name (pointer)) where the program is branched at the forced end of a loop.

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant	Others
	Bit	Word		Bit	Word				P
(D)	○							—	—
Pn	—							—	○

Processing details

- Forces an end to a FOR to NEXT instruction loop and shifts the operation to the pointer specified by Pn. Only a pointer within the same program file can be assigned to Pn. If a pointer of the other program file is used, an operation error will be returned.



- The remaining number of the FOR to NEXT instruction loop times is stored at (D). Note that the remaining number includes the operation when the BREAK instruction is executed.
- The BREAK instruction can be used only during the execution of a FOR to NEXT instruction loop.
- The BREAK instruction can be used only when there is only one level of nesting. When an end is forced to the multiple nesting levels, execute the same number of BREAK instructions for the nesting levels.

Operation error

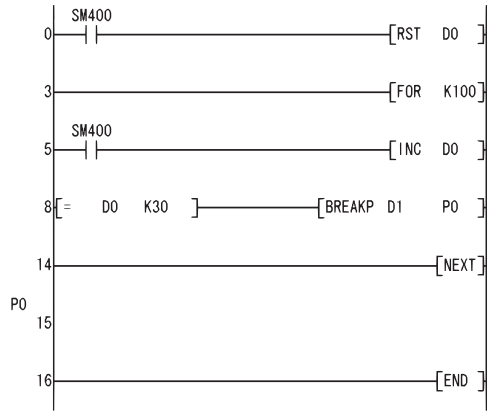
- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4203	The BREAK instruction is used in a case other than with the FOR to NEXT instruction loop.	○	○	○	○	○	○
4210	The jump destination for the pointer specified by Pn does not exist. The pointer of another program file is specified for Pn.	○	○	○	○	○	○

Program example

- The following program forces the FOR to NEXT loop to end when the value of D0 reaches 30 (when the FOR to NEXT loop has been executed 30 times). (The value 71 is stored at D1 when the BREAK instruction is executed.)

[Ladder Mode]



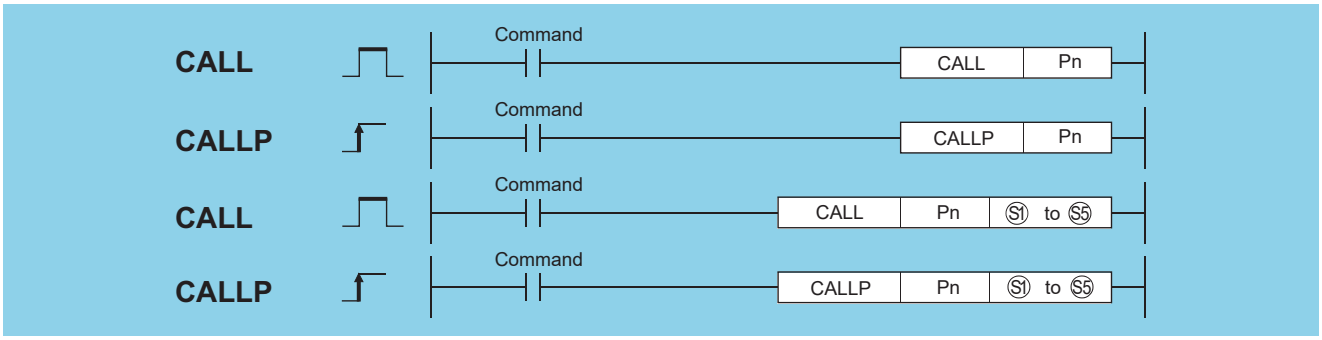
[List Mode]

Step	Instruction	Device	
0	LD	SM400	
1	RST	D0	
3	FOR	K100	
5	LD	SM400	
6	INC	D0	
8	LD=	D0	K30
11	BREAKP	D1	P0
14	NEXT		
15	P0		
16	END		

Subroutine program calls

CALL(P)

Basic | High performance | Process | Redundant | Universal | LCPU

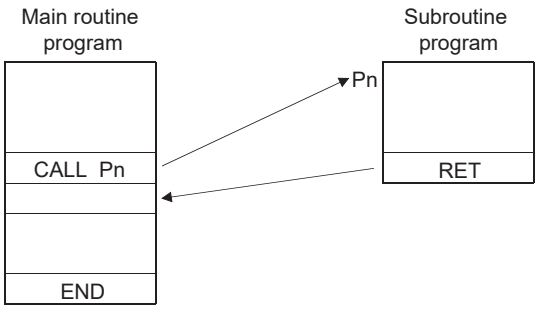


Pn: Head pointer number of a subroutine program (Device name)
(S1) to (S5): Number of the device to be passed as an argument to a subroutine program (bits, BIN 16 bits, BIN 32 bits)

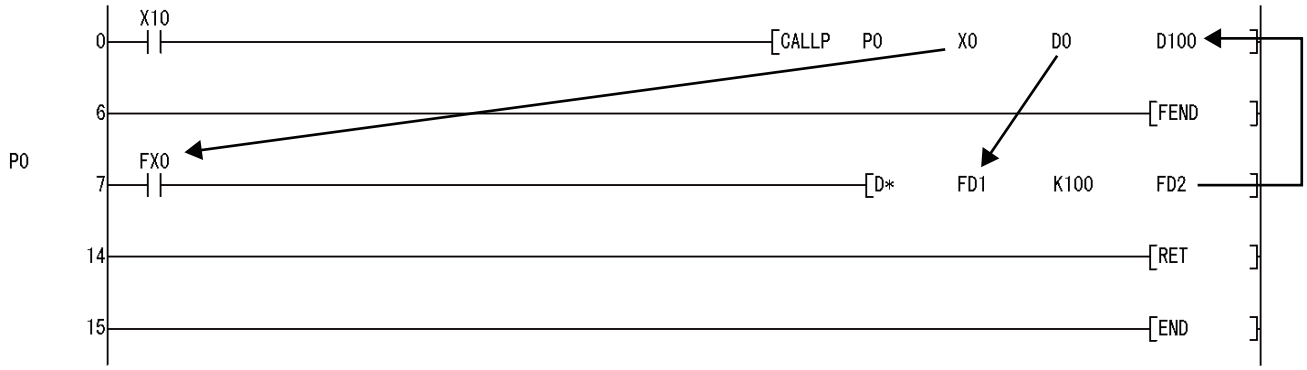
Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others P
	Bit	Word		Bit	Word				
Pn	—	—		—					○
(S1) to (S5)	○ (Other than F)	○		○					—

Processing details

- When the CALL (P) instruction is executed, executes the subroutine program of the program specified by Pn.
- The CALL (P) instruction can execute subroutine programs specified by a pointer within the same program file and subroutine programs specified by a common pointer.



• When function devices (FX, FY, FD) are used by a subroutine program, specify a device with (S1) to (S5) corresponding to the function device. The contents to the devices specified by (S1) to (S5) are as indicated below.

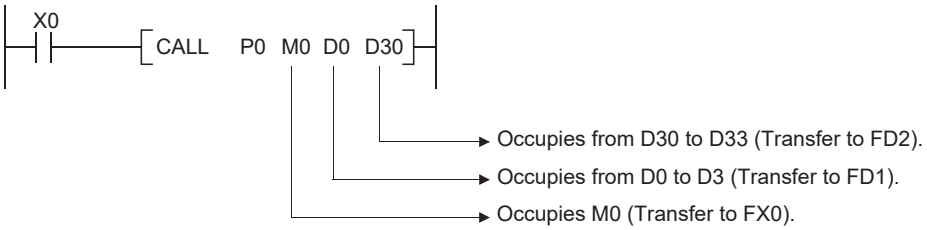


- Prior to execution of the subroutine program, bit data is transmitted to FX, and word data is transmitted to FD.
- After the execution of the subroutine program, the contents of FY and FD are transmitted to the corresponding devices.
- The function devices FX and FY are processed in units of bits. The function device FD is processed in units of 4 words. The size of the data to be dealt with will differ depending on the device specified in the argument. The device specified as a function device should be secured for the data size. An error will occur if it cannot be secured for the data size.

Function devices	Device	Data size	Remark
• FX • FY	Bit device	1 point	—
	When bit designation is made for word device	1 bit	
• FD	When digit designation of a bit device is used*1	4 words	The data size varies depending on the instruction to be used.
	Word device	4 words	

*1 An error will not occur even when the device number specified by (S1) to (S5) is not a multiple of 16 at the digit designation of the bit device.

[Main routine program]

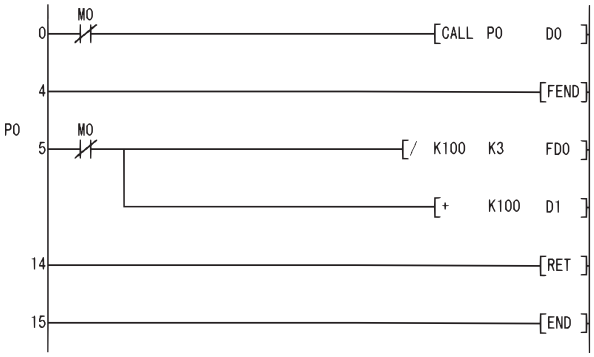


- (S1) to (S5) can be used with the CALL (P) instruction.
- The number of function devices to be used by a subroutine program must be identical to the number of arguments in the CALL (P) instruction. Also, the types of the function device and CALL (P) argument used should be identical.
- The device numbers specified as the arguments of the CALL(P) instruction should not overlap. If they do overlap, it will not be possible to obtain accurate calculations.
- The device used in the argument of the CALL (P) instruction should not be used in a subroutine program. If used, it will not be possible to obtain accurate calculations. (☞ Page 466 Incorrect operation example)
- When the device, either timer or counter, is used in the argument of the CALL(P) instruction, only the current value is transmitted/received.

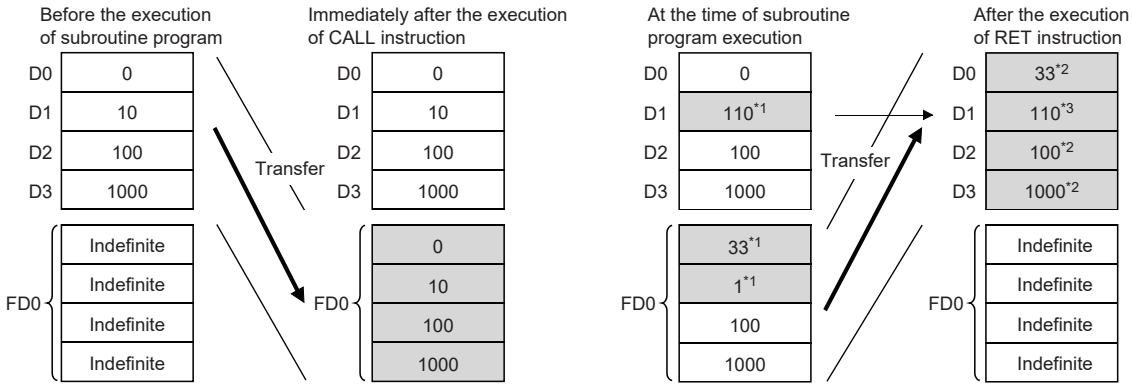
■Incorrect operation example

The following example shows the operation performed when D0 is specified for FD0 in the subroutine program and D1 is used in the subroutine program.

[Program example]



[Operation performed after subroutine program execution]



*1 Stores the execution result of the subroutine program.

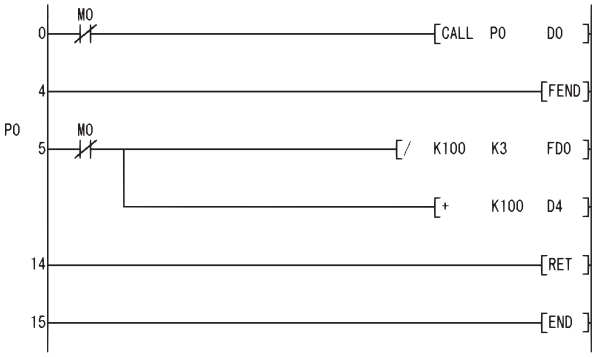
*2 Replaced by the value of the function device.

*3 D1 does not reflect the value of the function device.

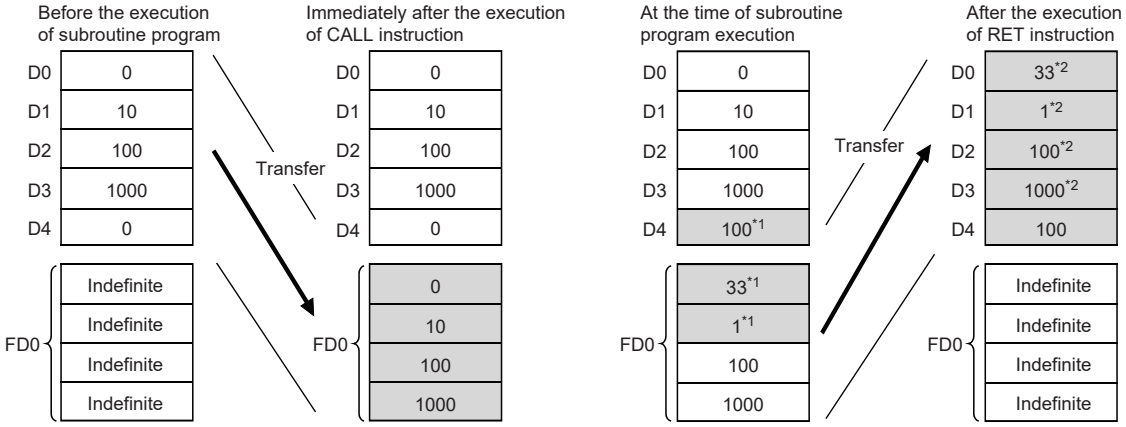
■Correct operation example

The following example shows the operation performed when D0 is specified for FD0 in the subroutine program and D4 is used in the subroutine program.

[Program example]



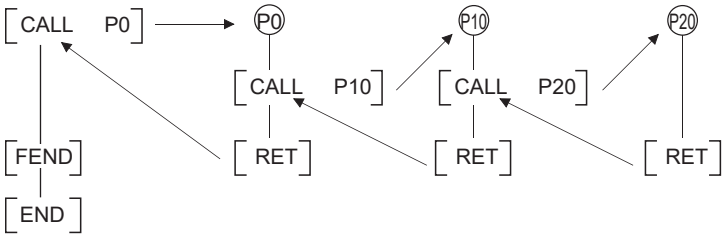
[Operation performed after subroutine program execution]



*1 Stores the execution result of the subroutine program.

*2 Replaced by the value of the function device.

- Up to 16 nesting levels are possible with the CALL(P) instruction. However, this 16 levels is the total number of levels in the CALL(P), FCALL(P), ECALL(P), EFCALL(P), and XCALL instructions.



- Devices which are turned ON within subroutine programs will be latched even if the subroutine program is not executed. Devices which are turned ON during the execution of a subroutine program can be turned OFF by the execution of the FCALL(P) instruction.

Operation error

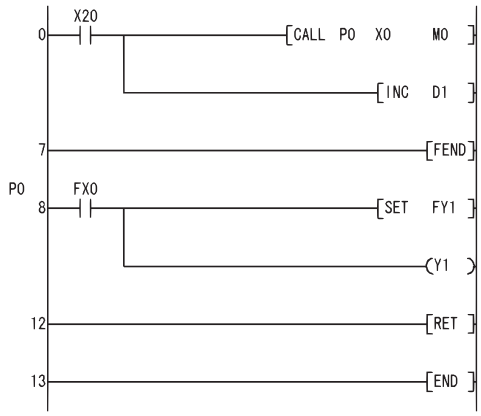
- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4101	The device specified for the argument cannot be secured for the data size.	○	○	○	○	○	○
4210	There is no subroutine program for the pointer specified in the CALL (P) instruction.	○	○	○	○	○	○
4211	After the CALL (P) instruction was executed, the END, FEND, GOEND, or STOP instruction was executed prior to the RET instruction.	○	○	○	○	○	○
4212	The RET instruction was executed prior to the CALL (P) instruction.	○	○	○	○	○	○
4213	The 17th nesting level is executed.	○	○	○	○	○	○

Program example

- The following program executes a subroutine program with argument when X20 is turned ON.

[Ladder Mode]



[List Mode]

Step	Instruction	Device		
0	LD	X20		
1	CALL	P0	X0	M0
5	INC	D1		
7	FEND			
8	P0			
9	LD	FX0		
10	SET	FY1		
11	OUT	Y1		
12	RET			
13	END			

Return from subroutine programs

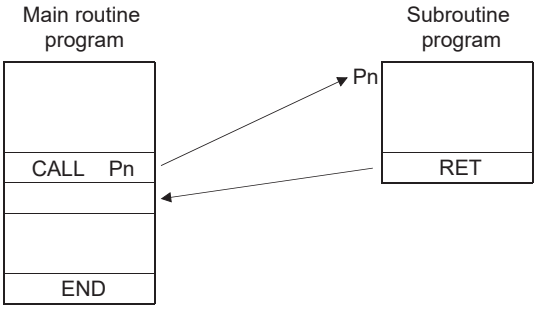
RET

Basic | High performance | Process | Redundant | Universal | LCPU

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant	Others
	Bit	Word		Bit	Word				
—	—								

Processing details

- Indicates end of subroutine program
- When the RET instruction is executed, returns to the step following the CALL (P), FCALL (P), ECALL (P), EFCALL (P) or XCALL instruction which called the subroutine program.



Operation error

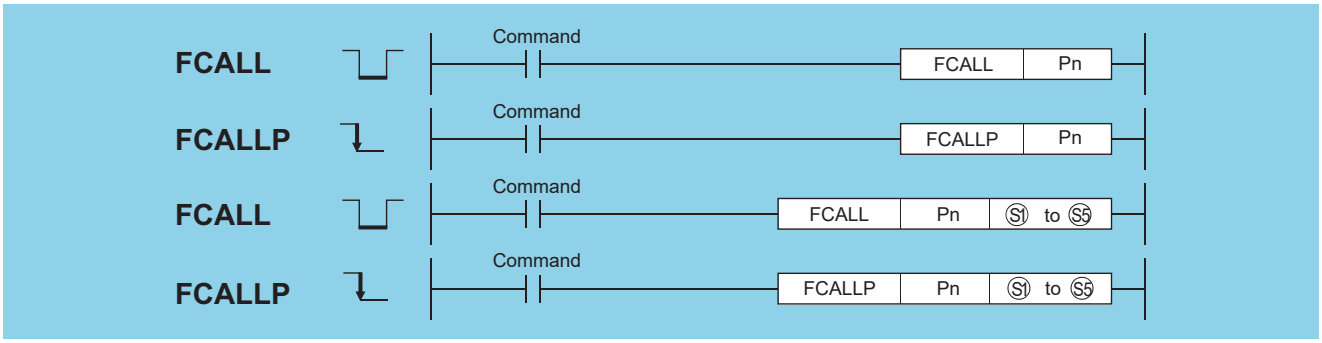
- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4211	After the CALL(P), FCALL (P), ECALL (P), EFCALL (P) or XCALL instruction was executed, an END, FEND, GOEND, or STOP instruction was executed prior to the RET instruction.	○	○	○	○	○	○
4212	The RET instruction was executed prior to the CALL (P), FCALL (P), ECALL (P), EFCALL (P) or XCALL instruction.	○	○	○	○	○	○

Subroutine program output OFF calls

FCALL(P)

Basic | High performance | Process | Redundant | Universal | LCPU

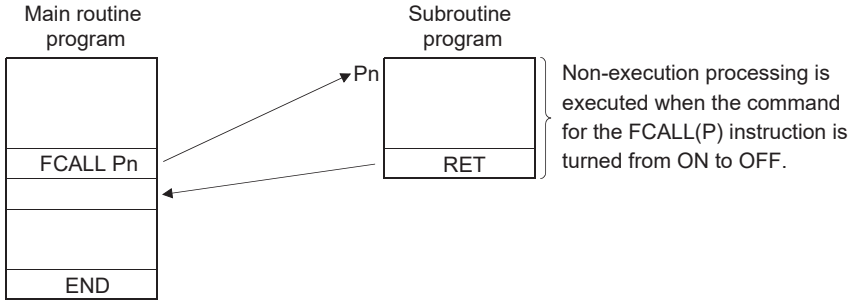


Pn: Head pointer number of a subroutine program (Device name)
(S1) to (S5): Number of the device to be passed as an argument to a subroutine program (bits, BIN 16 bits, BIN 32 bits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant	Others
	Bit	Word		Bit	Word				P
Pn	—	—		—					○
(S1) to (S5)	○ (Other than F)	○		○					—

Processing details

- When FCALL(P) is executed, the non-execution processing of the subroutine program of the pointer designated by Pn is performed.
- The FCALL (P) instruction can execute subroutine programs designated by a pointer within the same program file, and subroutine programs designated by common pointers.
  - Non-execution processing is identical to the processing that is conducted when the condition contacts for the individual coil instructions are in the OFF state.

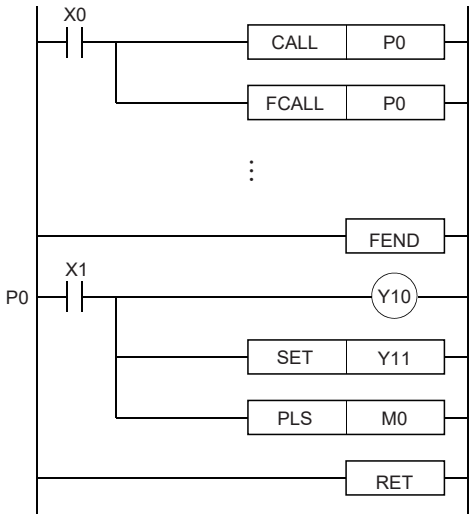


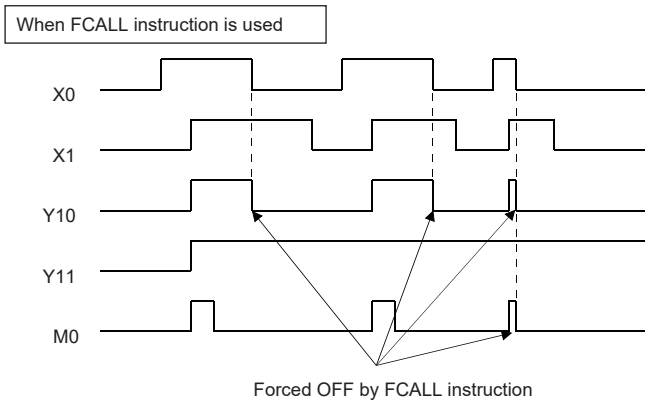
  - The operation results for the individual coil instructions following non-execution processing will be as follows, regardless of the ON/OFF status of the individual contacts:

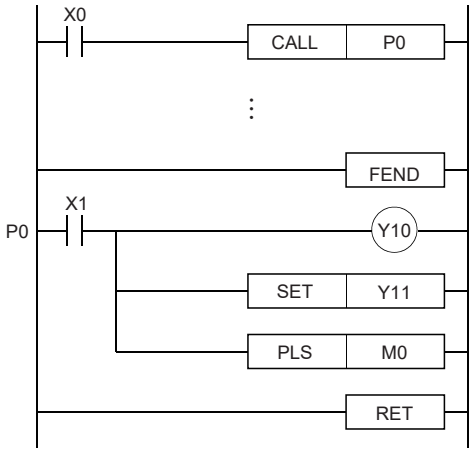
Item	Operation results
OUT instruction	Forced OFF
SET instruction RST instruction SFT instruction Basic instructions Application instructions	Maintains status
PLS instruction Pulse generation instruction (□P)	Processing identical to when condition contacts are OFF
Present value of low speed/high speed timers	0
Present value of retentive timer Present value of counter	Preserves

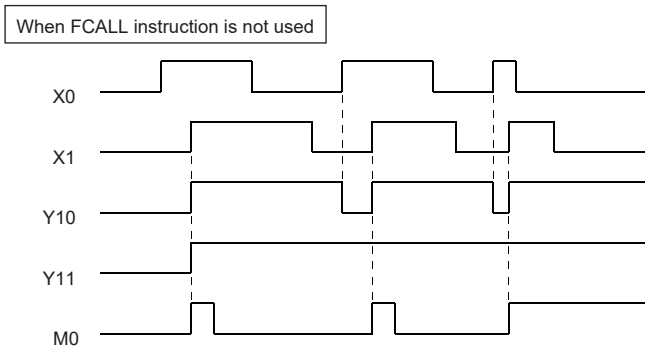
- The FCALL (P) instruction is used in conjunction with the CALL(P) instruction.

- If the FCALL (P) instruction is used in conjunction with the CALL(P) instruction, non-execution processing of a subroutine program is performed when the execution command is turned OFF, enabling forcible turning OFF of the OUT instruction and the PLS instruction (including □P instructions). In case the FCALL (P) instruction is not used in conjunction with the CALL(P) instruction, non-execution processing of a subroutine program is not performed even if the execution command is turned OFF. Therefore, output status of the individual coil instructions remains unchanged.

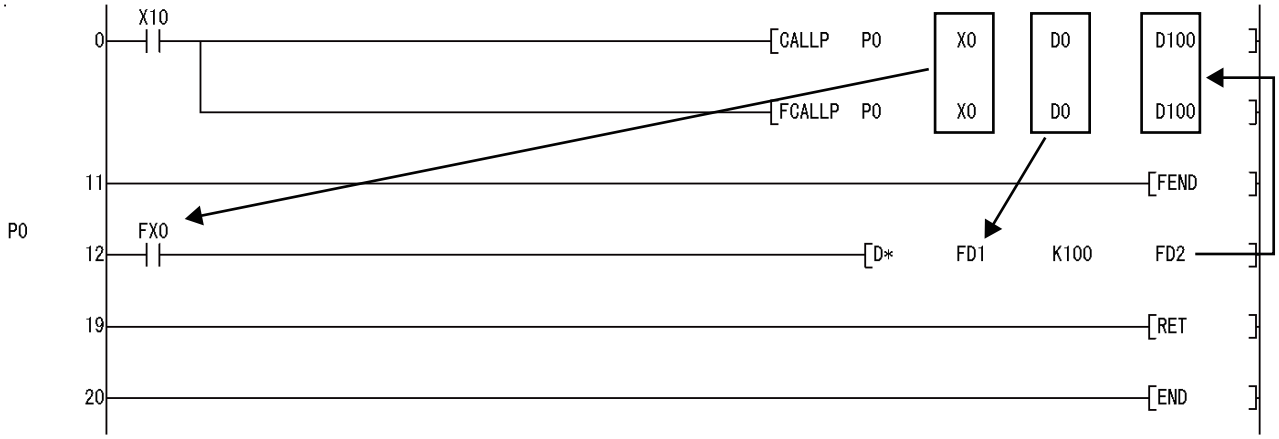








• When function devices (FX, FY, FD) are used by a subroutine program, specify a device with (S1) to (S5) corresponding to the function device. The contents to the devices specified by (S1) to (S5) are as indicated below.

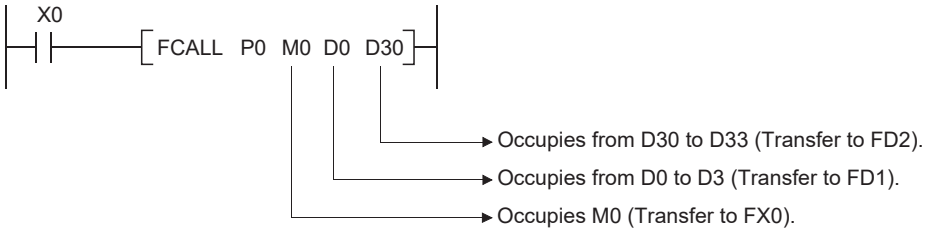


• Prior to execution of the subroutine program, bit data is transmitted to FX, and word data is transmitted to FD.
• After the execution of the subroutine program, the contents of FY and FD are transmitted to the corresponding devices.
• The function devices FX and FY are processed in units of bits. The function device FD is processed in units of 4 words. The size of the data to be dealt with will differ depending on the device specified in the argument. The device specified as a function device should be secured for the data size. An error will occur if it cannot be secured for the data size.

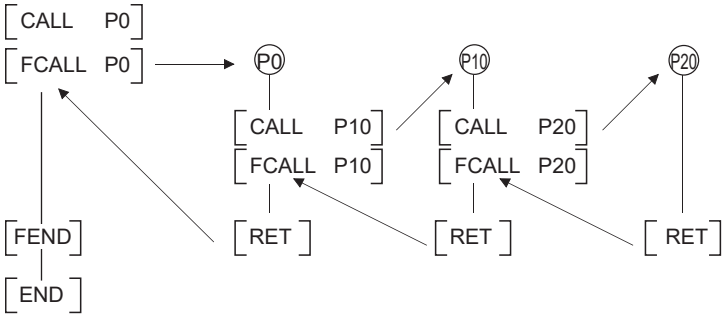
Function devices	Device	Data size	Remark
• FX • FY	Bit device	1 point	—
	When bit designation has been made for word device	1 bit	
• FD	When digit designation of a bit device is used*1	4 words	The upper 2 words of FD become 0.
	Word device	4 words	—

*1 An error will not occur even when the device number specified by (S1) to (S5) is not a multiple of 16 at the digit designation of the bit device.

[Main routine program]



• The FCALL (P) instruction can use from (S1) to (S5).
• Up to 16 nesting levels are possible with the FCALL(P) instruction. However, this 16 levels is the total number of levels in the CALL(P), FCALL(P), ECALL(P), EFCALL(P), and XCALL instructions.



Operation error

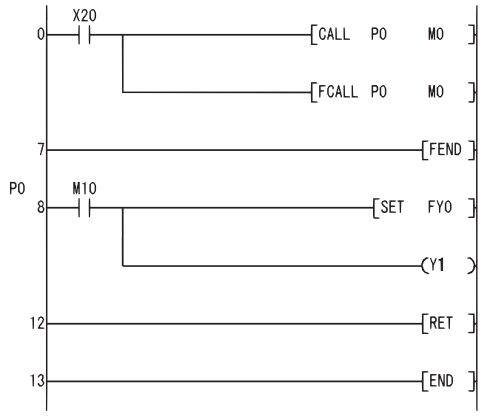
- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4101	The device specified for the argument cannot be secured for the data size.	○	○	○	○	○	○
4210	The subroutine program of the pointer designated by the FCALL (P) instruction does not exist.	○	○	○	○	○	○
4211	After the FCALL(P) instruction was executed, the END, FEND, GOEND, or STOP instruction was executed prior to the RET instruction.	○	○	○	○	○	○
4212	The RET instruction was executed prior to the FCALL (P) instruction.	○	○	○	○	○	○
4213	The 17th nesting level is executed.	○	○	○	○	○	○

Program example

- The following program executes a subroutine program with argument when X20 is turned ON, and forces non-execution processing when X20 is turned from ON to OFF.

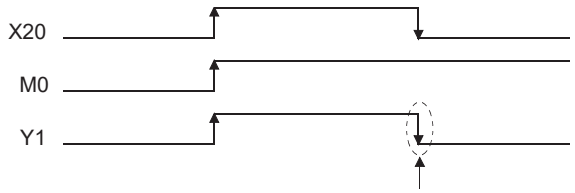
[Ladder Mode]



[List Mode]

Step	Instruction	Device	
0	LD	X20	
1	CALL	P0	M0
4	FCALL	P0	M0
7	FEND		
8	P0		
9	LD	M10	
10	SET	FY0	
11	OUT	Y1	
12	RET		
13	END		

[Timing Chart]

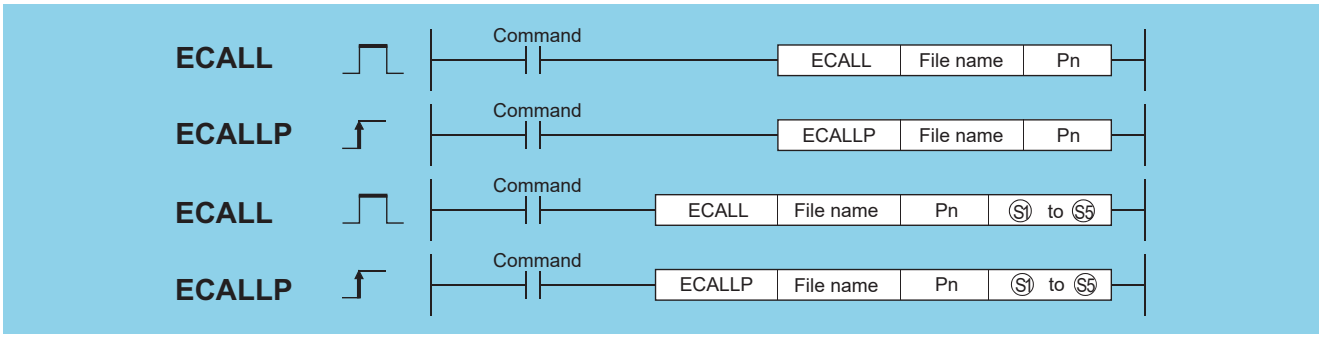


Forced OFF by FCALL instruction

Subroutine calls between program files

ECALL(P)

Basic | High performance | Process | Redundant | Universal | LCPU

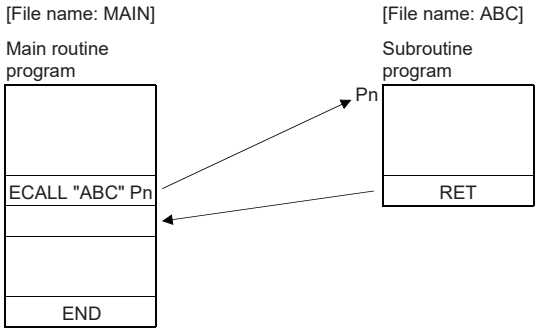


File name: Name of the program file to be called (character string)
Pn: Head pointer number of a subroutine program (Device name)
(S1) to (S5): Number of the device to be passed as an argument to a subroutine program (bits, BIN 16 bits, BIN 32 bits)

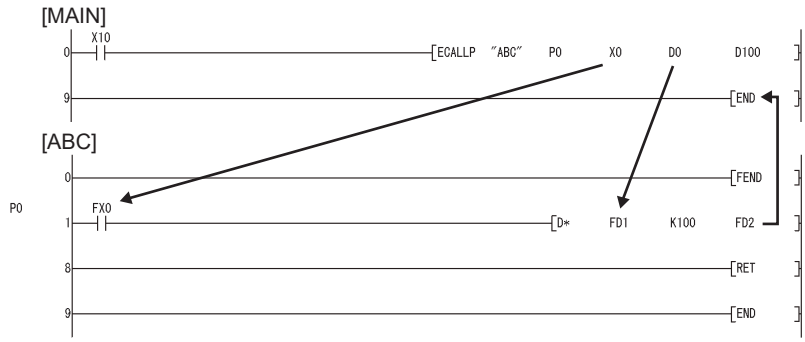
Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant		Others	
	Bit	Word		Bit	Word			K, H	$	P	
File name	—	○		—					○	—	
Pn	—	—		—					—	○	
(S1) to (S5)	○ (Other than F)	○		○					—	—	

Processing details

- Executes the subroutine program of the pointer designated by Pn in the designated program file name when the ECALL (P) instruction is executed. The ECALL(P) instruction can be used to call a subroutine program that uses a local pointer from a different program file.



- Only the file name of a program file stored in the drive 0 (program memory/internal RAM) can be designated for a file name.
- It is not necessary to designate the extension (.QPG) with the file name. (Only .QPG files will be acted on.)
- When function devices (FX, FY, FD) are used by a subroutine program, specify a device corresponding to the function device with (S1) to (S5). The contents of the devices specified by (S1) to (S5) are as indicated below.

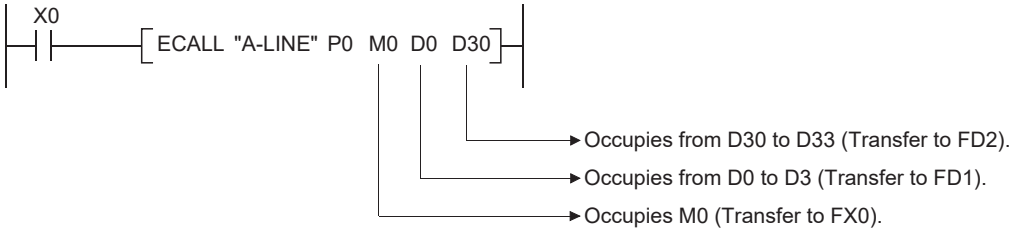


- Prior to execution of the subroutine program, bit data is transmitted to FX, and word data is transmitted to FD.
- After the execution of the subroutine program, the contents of FY and FD are transmitted to the corresponding devices.
- The function devices FX and FY are processed in units of bits. The function device FD is processed in units of 4 words. The size of the data to be dealt with will differ depending on the device specified in the argument. The device specified as a function device should be secured for the data size. An error will occur if it cannot be secured for the data size.

Function devices	Device	Data size	Remark
• FX • FY	Bit device	1 point	—
	When bit designation has been made for word device	1 bit	
• FD	When digit designation of a bit device is used*1	4 words	The data size varies depending on the instruction to be used.
	Word device	4 words	

*1 An error will not occur even when the device number specified by (S1) to (S5) is not a multiple of 16 at the digit designation of the bit device.

[Main routine program]



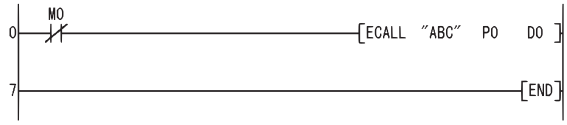
- From (S1) to (S5) can be used by the ECALL instruction.
- The device used in the argument of the ECALL instruction should not be used in a subroutine program. If used, it will not be possible to obtain accurate calculations. (☞ Page 476 Incorrect operation example)

■Incorrect operation example

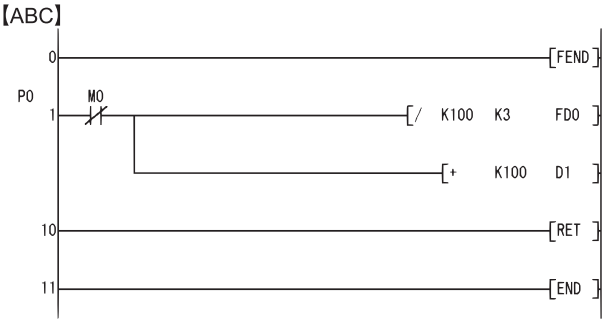
The following example shows the operation performed when D0 is specified for FD0 in the subroutine program and D1 is used in the subroutine program.

[Program example]

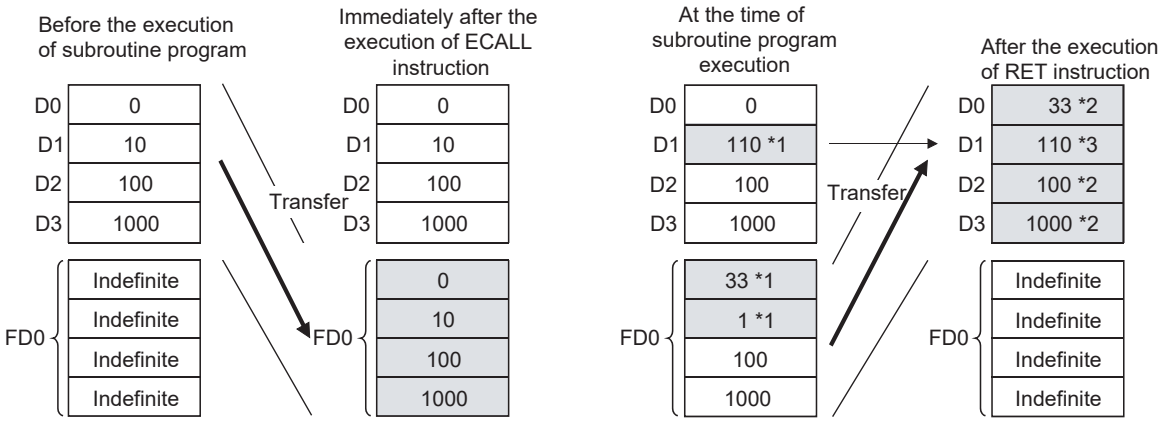
【MAIN】



【ABC】



[Operation performed after subroutine program execution]



*1 Stores the execution result of the subroutine program.

*2 Replaced by the value of the function device.

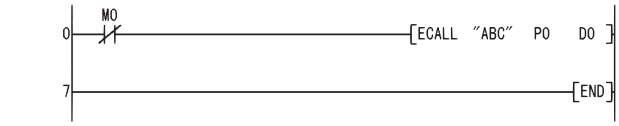
*3 D1 does not reflect the value of the function device.

■Correct operation example

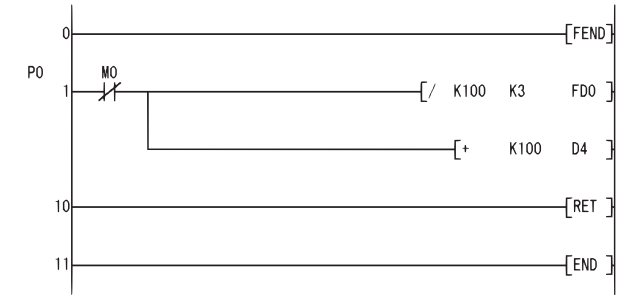
The following example shows the operation performed when D0 is specified for FD0 in the subroutine program and D4 is used in the subroutine program.

[Program example]

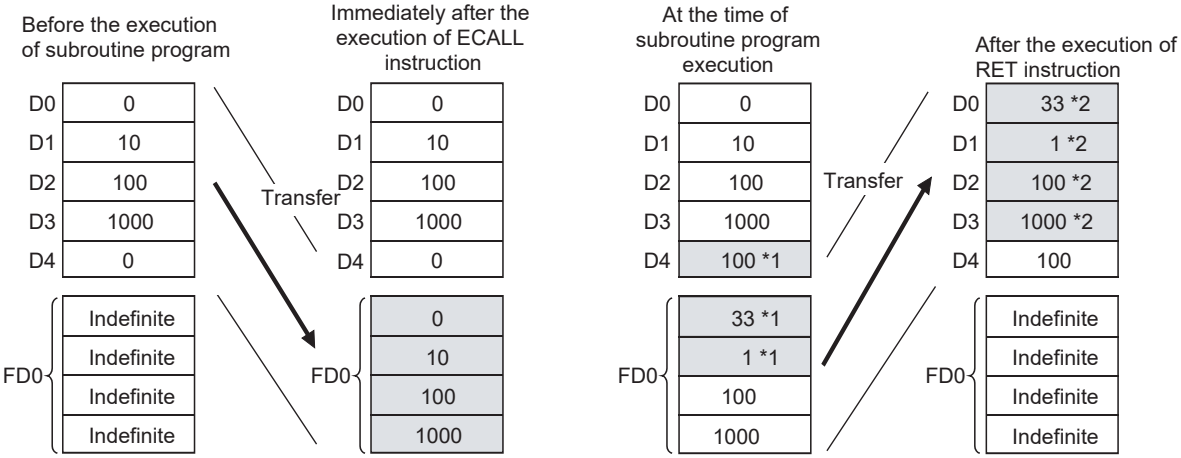
[MAIN]



[ABC]



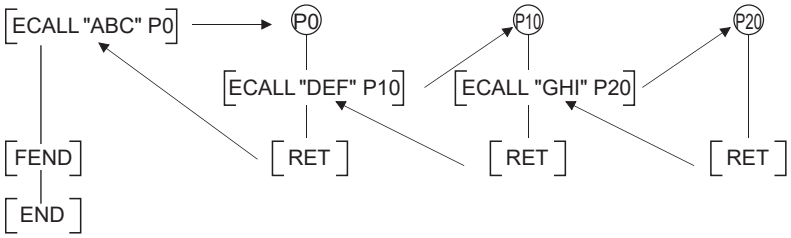
[Operation performed after subroutine program execution]



*1 Stores the execution result of the subroutine program.

*2 Replaced by the value of the function device.

- The device numbers specified as the arguments of the ECALL(P) instruction should not overlap. If they do overlap, it will not be possible to obtain accurate calculations.
- Up to 16 levels of nesting can be used with the ECALL(P) instruction. However, this 16 levels is the total number of levels in the CALL(P), FCALL(P), ECALL(P), EFCALL(P), and XCALL instructions.



- Devices which are turned ON within subroutine programs will be latched even if the subroutine program is not executed. Devices turned ON during the execution of a subroutine program can be turned OFF by the EFCALL(P) instruction.

Operation error

• In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

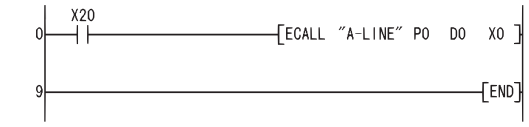
Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
2410	The specified file does not exist.	○	○	○	○	○	○
2411	The specified file cannot be executed.	○	○	○	○	○	○
4101	The device specified for the argument cannot be secured for the data size.	○	○	○	○	○	○
4210	The subroutine program of the pointer specified by the ECALL (P) instruction does not exist.	○	○	○	○	○	○
4211	After the ECALL (P) instruction was executed, the END, FEND, GOEND, or STOP instruction was executed prior to the RET instruction.	○	○	○	○	○	○
4212	The RET instruction was executed prior to the ECALL (P) instruction.	○	○	○	○	○	○
4213	The 17th nesting level is executed.	○	○	○	○	○	○

Program example

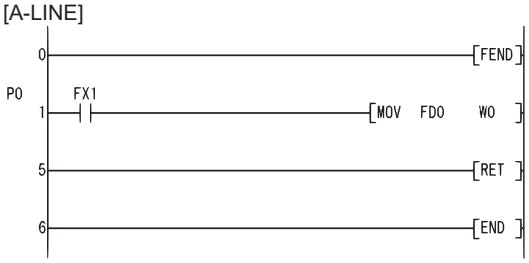
• The following program executes program block P0 of the program A-LINE when X20 is turned ON.

[Ladder Mode]

[MAIN]



[A-LINE]



[List Mode]

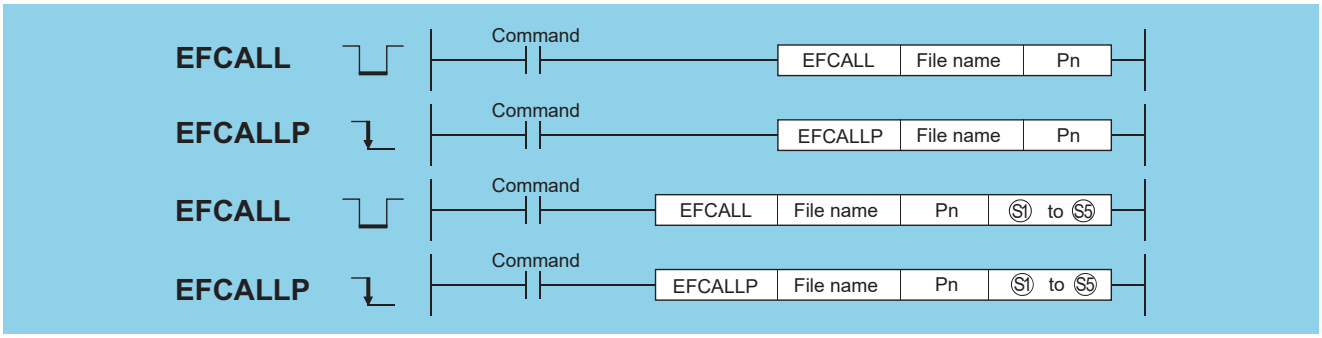
Step	Instruction	Device
0	LD	X20
1	ECALL	"A-LINE" P0 D0 X0
9	END	

Step	Instruction	Device
0	FEND	
1	P0	
2	LD	FX1
3	MOV	FD0 W0
5	RET	
6	END	

Subroutine output OFF calls between program files

EFCALL(P)

Basic | High performance | Process | Redundant | Universal | LCPU

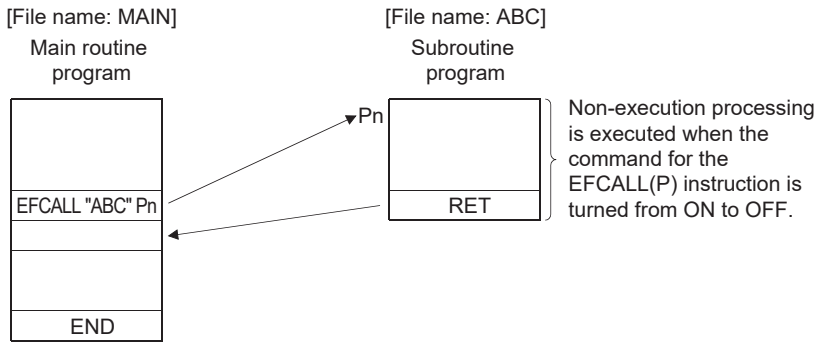


File name: Name of the program file to be called (character string)
Pn: Head pointer number of a subroutine program (Device name)
(S1) to (S5): Number of the device to be passed as an argument to a subroutine program (bits, BIN 16 bits, BIN 32 bits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant		Others	
	Bit	Word		Bit	Word			K, H	$	P	
File name	—	○		—					○	—	
Pn	—	—		—					—	○	
(S1) to (S5)	○ (Other than F)	○		○					—	—	

Processing details

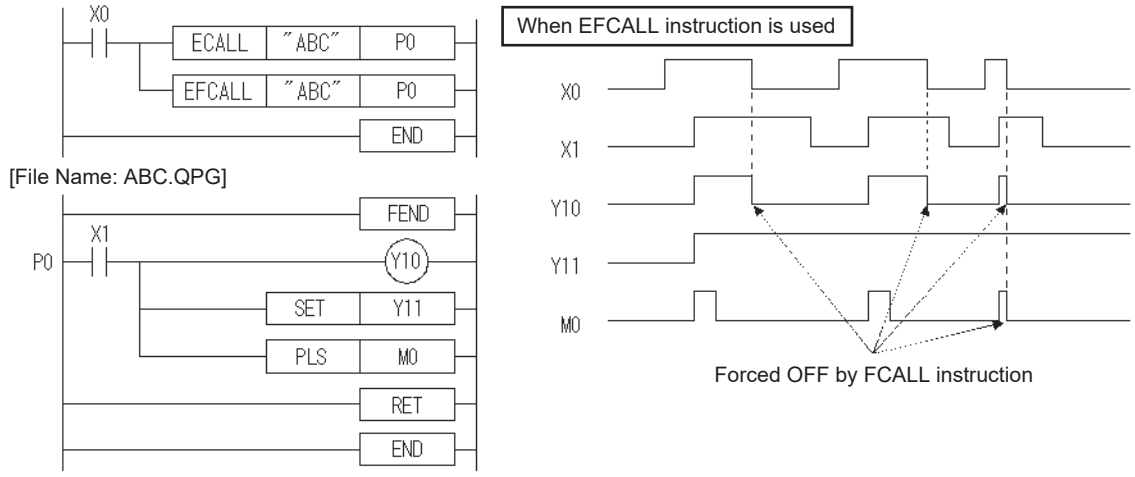
- When the EFCALL(P) instruction is executed, the non-execution processing of the subroutine program of the pointer designated by Pn is performed.
- The EFCALL (P) can also be used to call a subroutine program that uses a local pointer from a different program file.
  - Non-execution processing is identical to the processing that is conducted when the condition contacts for the individual coil instructions are in the OFF state.

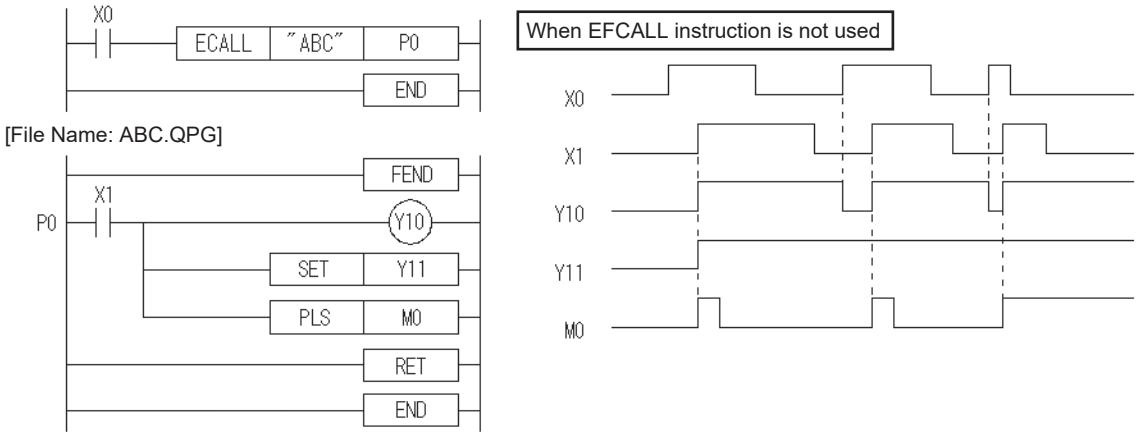


- The operation results for the individual coil instructions following non-execution processing will be as follows, regardless of the ON/OFF status of the individual contacts:

Item	Operation results
OUT instruction	Forced OFF
SET instruction RST instruction SFT instruction Basic instructions Application instructions	Maintains status
PLS instruction Pulse generation instruction (□P)	Processing identical to when condition contacts are OFF
Present value of low speed/high speed timers	0
Present value of retentive timer Present value of counter	Preserves

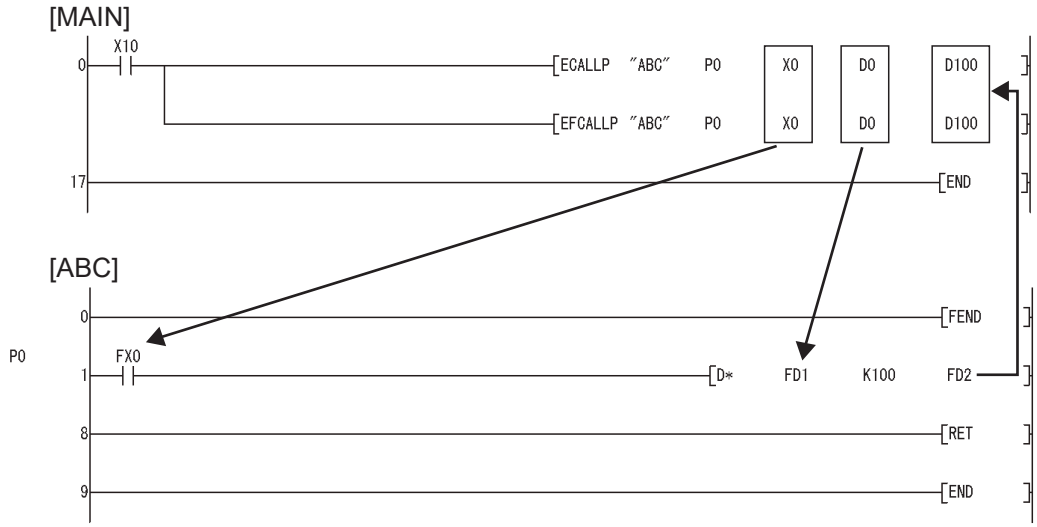
- The EFCALL (P) instruction is used in combination with the ECALL (P) instruction.
- If the EFCALL(P) instruction is used in conjunction with the ECALL(P) instruction, non-execution processing of a subroutine program is performed when the execution command is turned OFF, enabling forcible turning OFF of the OUT instruction and the PLS instruction (including □P instructions). In case the EFCALL(P) instruction is not used in conjunction with the ECALL(P) instruction, non-execution processing of a subroutine program is not performed even if the execution command is turned OFF. Therefore, output status of the individual coil instructions remains unchanged.





- Only the file name of a program file stored in the drive 0 (program memory/internal RAM) can be designated for a file name.
- It is not necessary to designate the extension (.QPG) with the file name. (Only .QPG files will be acted on.)

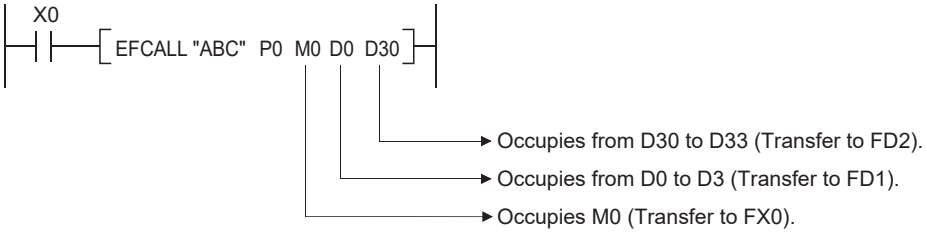
- When function devices (FX, FY, FD) are used by a subroutine program, specify a device corresponding to the function device with (S1) to (S5).



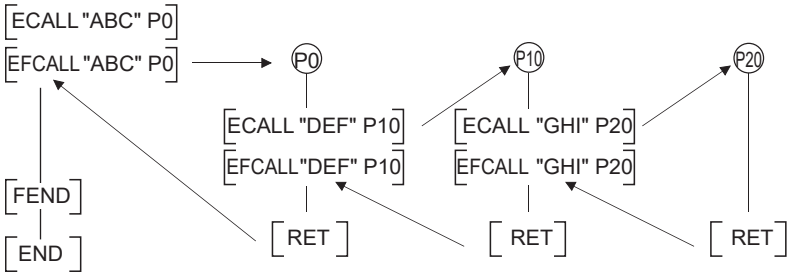
- Prior to execution of the subroutine program, bit data is transmitted to FX, and word data is transmitted to FD.
- After the execution of the subroutine program, the contents of FY and FD are transmitted to the corresponding devices.
- The function devices FX and FY are processed in units of bits. The function device FD is processed in units of 4 words. The size of the data to be dealt with will differ depending on the device specified in the argument. The device specified as a function device should be secured for the data size. An error will occur if it cannot be secured for the data size.

Function devices	Device	Data size	Remark
• FX • FY	Bit device	1 point	—
	When bit designation has been made for word device	1 bit	
• FD	When digit designation of a bit device is used*1	4 words	The upper 2 words of FD become 0.
	Word device	4 words	—

*1 An error will not occur even when the device number specified by (S1) to (S5) is not a multiple of 16 at the digit designation of the bit device.
[Main routine program]



- (S1) to (S5) can be used with the EFCALL (P) instruction.
- The number of function devices used by subroutine programs must be identical to the number of arguments used by the EFCALL (P) instruction. Further, the function devices should be identical to the types of arguments used by the EFCALL (P) instruction.
- Up to 16 levels of nesting can be used with the EFCALL (P) instruction. However, this 16 levels is the total number of levels in the CALL(P), FCALL(P), ECALL(P), EFCALL(P), and XCALL instructions.



Operation error

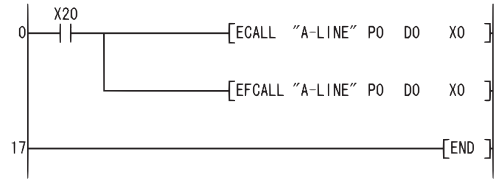
- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
2411	The specified file cannot be executed.	○	○	○	○	○	○
4101	The device specified for the argument cannot be secured for the data size.	○	○	○	○	○	○
4210	The subroutine program of the pointer specified by the ECALL (P) instruction does not exist. The specified file does not exist.	○	○	○	○	○	○
4211	After the EFCALL (P) instruction was executed, the END, FEND, GOEND, or STOP instruction was executed prior to the RET instruction.	○	○	○	○	○	○
4212	The RET instruction was executed prior to the EFCALL (P) instruction.	○	○	○	○	○	○
4213	The 17th nesting level is executed.	○	○	○	○	○	○

Program example

- The following program executes a subroutine program with argument when X0 is ON, and forces non-execution processing when X20 is turned from ON to OFF.

[Ladder Mode]



[List Mode]

Step	Instruction	Device		
0	LD	X20		
1	ECALL	"A-LINE" P0	D0	X0
9	EFCALL	"A-LINE" P0	D0	X0
17	END			

Subroutine program calls

XCALL

Ver. Basic | High performance | Process | Redundant | Universal | LCPU

- Basic model QCPU: The serial number (first five digits) is "04122" or later.

Pn: Head pointer number of a subroutine program (Device name)
(S1) to (S5): Number of the device to be passed as an argument to a subroutine program (bits, BIN 16 bits, BIN 32 bits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others P
	Bit	Word		Bit	Word				
P	—	—		—					○
(S1) to (S5)	○ (Other than F)	○		○					—

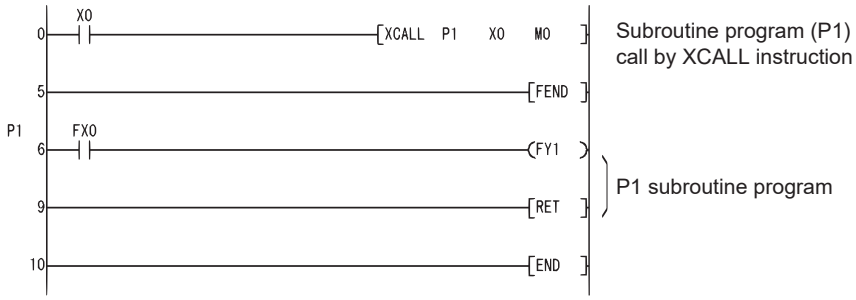
Processing details

- XCALL instruction executes the subroutine program and performs non-execution processing of the subroutine program.
  - Execution of subroutine program: Executes each coil instruction according to ON/OFF status of the condition contacts.
  - Non-execution of subroutine program: Performs the same processing for each coil instruction as when the condition contacts are OFF status.
- The operation results for the individual coil instructions following non-execution processing will be as follows, regardless of the ON/OFF status of the individual contacts:

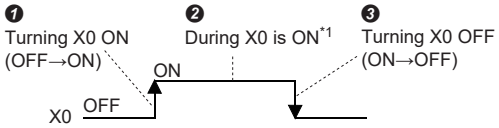
Item	Operation results
OUT instruction	Forced OFF
SET instruction RST instruction SFT instruction Basic instructions Application instructions	Maintains status
PLS instruction Pulse generation instruction (□P)	Processing identical to when condition contacts are OFF
Present value of low speed/high speed timers	0
Present value of retentive timer Present value of counter	Preserves

• Operation of XCALL instruction varies according to the CPU module type. The following program example shows the operation of XCALL instruction for each CPU module.

[Program example]



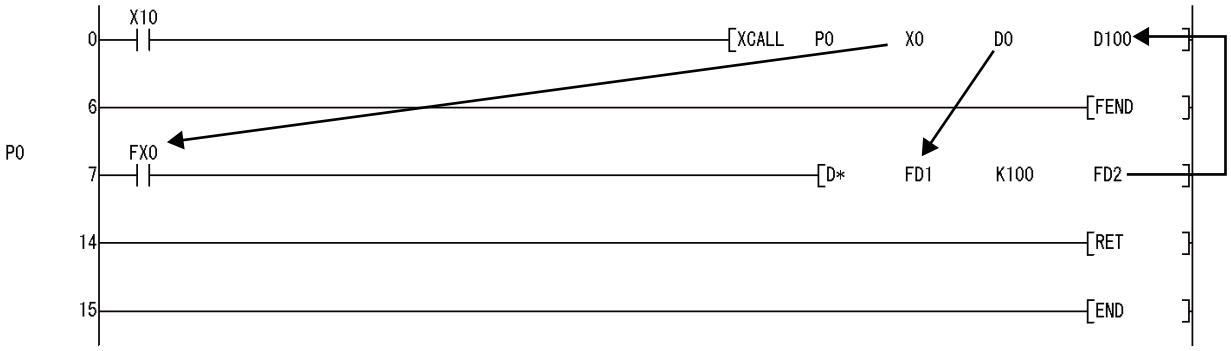
[ON/OFF timing of X0]



*1 Time during X0 is ON (❷) does not include the time when turning X0 ON (❶).

Component	Operation of XCALL instruction
• Process CPU (serial No. of first five digits: 07031 or earlier) • High performance model QCPU (serial No. of first five digits: 06081 or earlier)	❶ When X0 is turned ON: Without process (Do not execute subroutine program of "P1".) ❷ During X0 is ON: Execute subroutine program of "P1". ❸ When X0 is turned OFF: Perform "Non-execution processing" of subroutine program of "P1".
• High performance model QCPU (serial No. of first five digits: 06082 or later) • Process CPU (serial No. of first five digits: 07032 or later)	❶ Using SM734 (XCALL instruction executing condition designation) to select operation when X0 is turned ON. • When SM734 is OFF: Without process (Do not execute subroutine program of "P1".) • When SM734 is ON: Execute subroutine program of "P1". ❷ During X0 is ON: Execute subroutine program of "P1". ❸ When X0 is turned OFF: Perform "Non-execution processing" of subroutine program of "P1".
• Redundant CPU • Basic model QCPU • Universal model QCPU • LCPU	❶ When X0 is turned ON: Execute subroutine program of "P1". ❷ During X0 is ON: Execute subroutine program of "P1". ❸ When X0 is turned OFF: Perform "Non-execution processing" of subroutine program of "P1".

- When function devices (FX, FY, FD) are used by a subroutine program, specify a device with (S1) to (S5) corresponding to the function device. The contents to the devices specified by (S1) to (S5) are as indicated below.

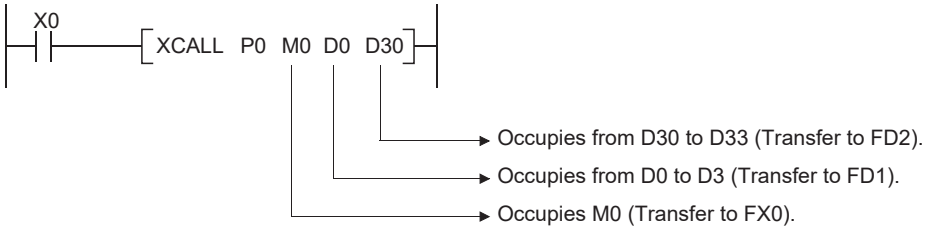


- Prior to execution of the subroutine program, bit data is transmitted to FX, and word data is transmitted to FD.
- After the execution of the subroutine program, the contents of FY and FD are transmitted to the corresponding devices.
- The function devices FX and FY are processed in units of bits. The function device FD is processed in units of 4 words. The size of the data to be dealt with will differ depending on the device specified in the argument. The device specified as a function device should be secured for the data size. An error will occur if it cannot be secured for the data size.

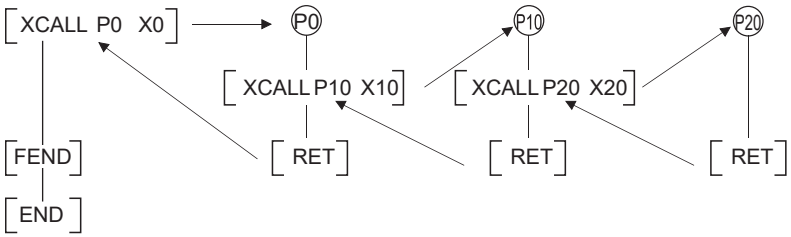
Function devices	Device	Data size	Remark
• FX • FY	Bit device	1 point	—
	When bit designation is made for word device	1 bit	
• FD	When digit designation of a bit device is used*2	4 words	The data size varies depending on the instruction to be used.
	Word device	4 words	

*2 An error will not occur even when the device number specified by (S1) to (S5) is not a multiple of 16 at the digit designation of the bit device.

[Main routine program]



- (S1) to (S5) can be used by the XCALL instruction.
- The number of function devices used by a subroutine program must be identical to the number of arguments in the XCALL instruction. Also, the function device and the type of XCALL argument should be identical.
- The device numbers specified as the arguments of the XCALL instruction should not overlap. If they do overlap, it will not be possible to obtain accurate calculations.
- Up to 16 nesting levels can be used with the XCALL instruction. However, this 16 levels is the total number of levels in the CALL(P), FCALL(P), ECALL(P), EFCALL(P), and XCALL instructions.

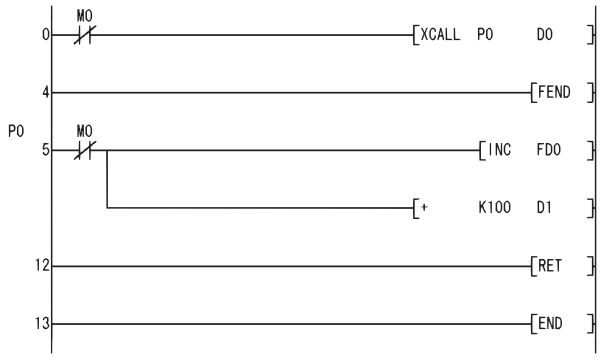


• The device used for the argument of the XCALL instruction must not be used in a subroutine program. If used, it will not be possible to perform correct calculations. (☞ Page 486 Incorrect operation example)

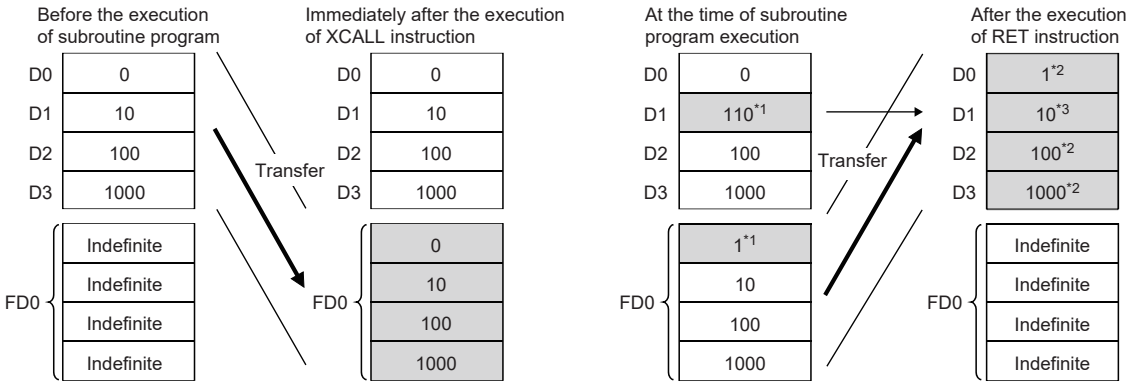
■Incorrect operation example

The following example shows the operation performed when D0 is specified for FD0 in the subroutine program and D1 is used in the subroutine program.

[Program example]



[Operation performed after subroutine program execution]



*1 Stores the execution result of the subroutine program.

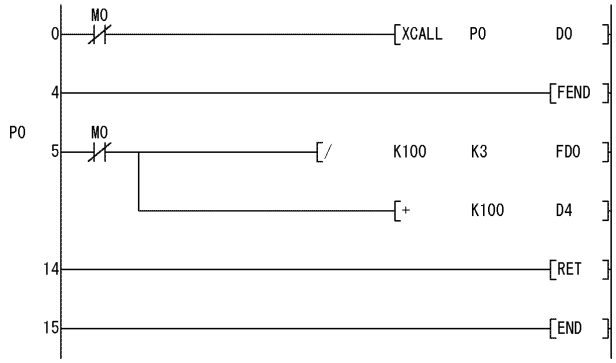
*2 Replaced by the value of the function device.

*3 D1 does not reflect the operation result in the subroutine program.

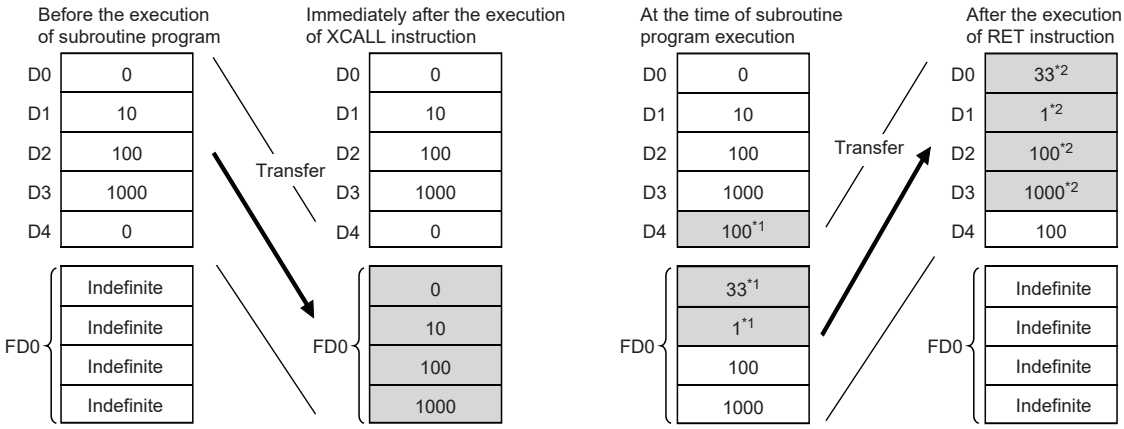
■Correct operation example

The following example shows the operation performed when D0 is specified for FD0 in the subroutine program and D4 is used in the subroutine program.

[Program example]



[Operation performed after subroutine program execution]



*1 Stores the execution result of the subroutine program.

*2 Replaced by the value of the function device.

Operation error

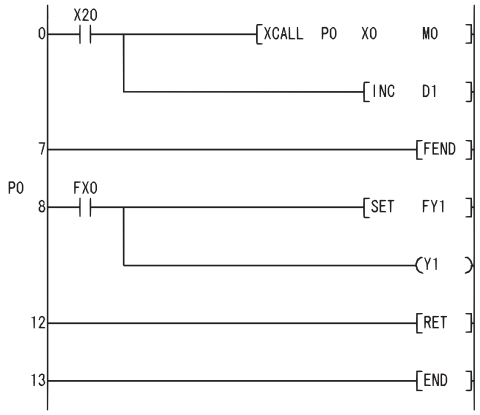
- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4101	The device specified for the argument cannot be secured for the data size.	○	○	○	○	○	○
4210	There is no subroutine program for the pointer specified in the XCALL (P) instruction.	○	○	○	○	○	○
4211	After the XCALL (P) instruction was executed, the END, FEND, GOEND, or STOP instruction was executed prior to the RET instruction.	○	○	○	○	○	○
4212	The RET instruction was executed prior the XCALL (P) instruction.	○	○	○	○	○	○
4213	The 17th nesting level is executed.	○	○	○	○	○	○

Program example

- The following program executes a subroutine program with argument when X20 is turned ON.

[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	X20
1	XCALL	P0 X0 M0
5	INC	D1
7	FEND	
8	P0	
9	LD	FX0
10	SET	FY1
11	OUT	Y1
12	RET	
13	END	

Refresh

COM

Basic | High performance | Process | Redundant (×) | Universal (×) | LCPU (×)

Refer to Page 491 Select refresh (COM) for the COM instruction of the following CPU modules.

- Basic model QCPU of serial No. 04122 or later
- High Performance model QCPU of serial No. 04012 or later
- Process CPU of serial No. 07032 or later
- Redundant CPU
- Universal model QCPU
- LCPU

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant	Others
	Bit	Word		Bit	Word				
—	—								

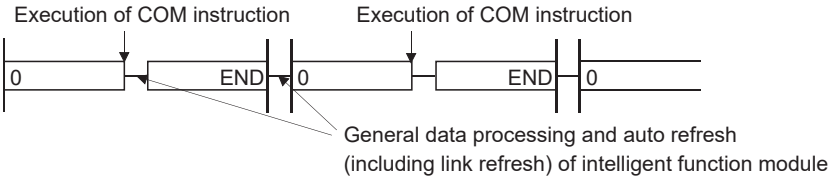
Processing details

- Use the COM instruction for the following purposes.
  - To reduce the time required to send/receive data to/from the remote I/O stations.
  - To ensure data communication with a CPU module on another station when two CPU modules perform operations using different scan times.
- The processing of the COM instruction differs depending on the status (ON or OFF) of the special relay SM775.
  - SM775 is OFF: Performs both auto refresh and communication with external devices.*1*2
  - SM775 is ON: Performs only communication with external devices.*1

*1 The following processing is performed in communication with external devices.
- Monitor processing of other station
- Read processing by the serial communications module of the buffer memory of another intelligent function module

*2 The auto refresh includes the following processing:
- Refresh of MELSECNET/10, MELSECNET/H
- CC-Link refresh
- Auto refresh of intelligent function modules

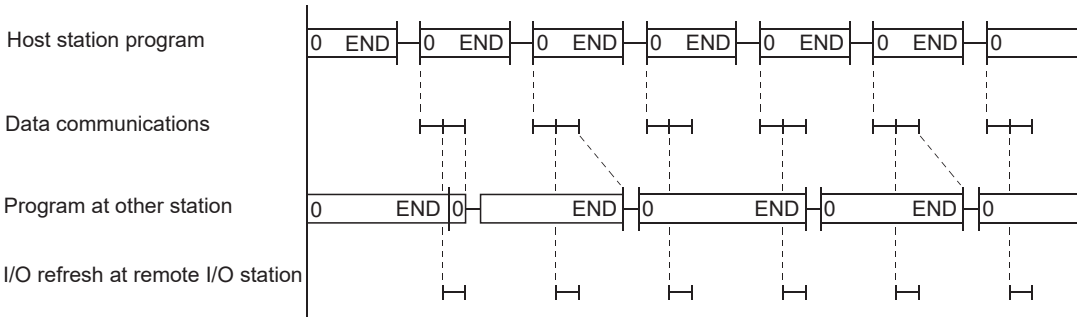
- At the point of the execution of the COM instruction, the CPU module temporarily stops the processing of the sequence program, and performs the same operation as ordinary data processing as well as auto refresh of intelligent function modules (including link refreshes) at the END processing. However, the low speed cyclic refresh of MELSECNET/10 or MELSECNET/H is not performed.



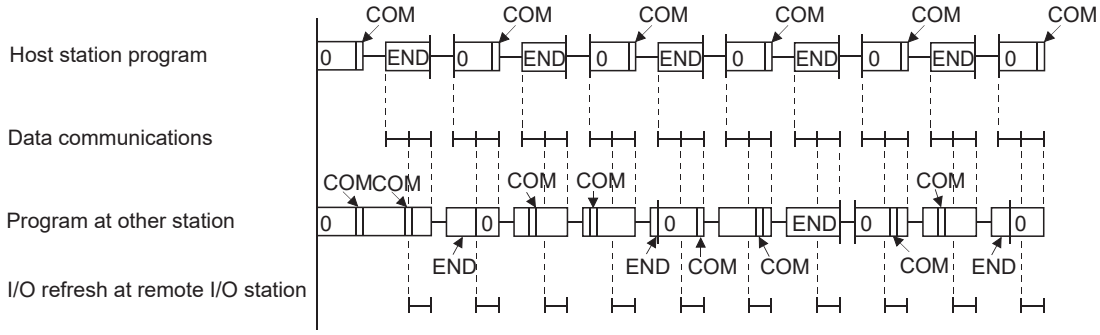
- The COM instruction can be used in a sequence program any number of times. Note, however, that the scan time of the sequence program will increase by the time taken for communication with external devices and auto refresh (including link refresh) of intelligent function modules.

• Data communications using the COM instruction

(1) Example of data communications when COM instruction is not used



(2) Example of data communications when COM instruction has been used

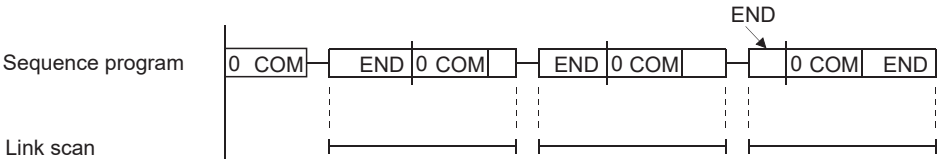


- When the COM instruction is used at the host station, it is possible to increase the number of data communication repetitions with the remote I/O station unconditionally, as shown in (2) above, and thus to speed up data communications.
- In cases where the remote station scan time is longer than the scan time of the host station, the COM instruction used at the remote station side can avoid the occurrence of timing failure in which the data cannot be fetched, as shown in (1).
- When the COM instruction has been used at the other station, a link refresh will be performed each time that station receives a command from the host station.

· Step 0 ~COM instruction
· COM instruction ~COM instruction
· COM instruction ~END instruction

Link refresh can be performed once in each of these intervals.

- If the scan time from the linked station is longer than the sequence program scan time at the host station, designating the COM instruction at the host station will not increase the speed of data communications.



Point

The programs in which the COM instruction cannot be used are shown below:

- Low-speed execution type programs
- Interrupt programs
- Fixed scan execution type programs

Operation error

- There is no operation error in the COM instruction.

Select refresh (COM)

COM

Ver. Basic | Ver. High performance | Ver. Process | Redundant | Universal | LCPU

- Basic model QCPU: The serial number (first five digits) is "04122" or later.
- High Performance model QCPU: The serial number (first five digits) is "04012" or later.
- Process CPU: The serial number (first five digits) is "07032" or later.

Refer to Page 489 Refresh for the COM instruction of the following CPU modules.

- Basic model QCPU of serial No. 04121 or later
- High Performance model QCPU of serial No. 04011 or later
- Process CPU of serial No. 07031 or later

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant	Others
	Bit	Word		Bit	Word				
—	—								

Point

When switching the execution status of the select refresh by turning ON/OFF an execution condition, use the CCOM(P) instruction. (☞ Page 494 Select refresh (CCOM(P)))

Processing details

- The COM instruction is used to perform I/O refresh at any timing during execution of a sequence program.
- The following processing can be performed with the COM instruction.

Processing item	QCPU	LCPU
I/O refresh	○	○
CC-Link refresh	○	○
CC-Link IE Controller Network refresh	○	×
CC-Link IE Field Network refresh	○*1	○*2
CC-Link IE Field Network Basic refresh	○*4	○*5
MELSECNET/H refresh	○	×
Auto refresh of intelligent function modules	○	○
Auto refresh using QCPU standard area of multiple CPU system	○	×
Reading input/output data of all modules other than the multiple CPU system group	○	×
Auto refresh using the multiple CPU high speed transmission area of multiple CPU system	○	×
Communication with display unit	×	○*3
Service processing (communication with programming tool, GOT, or other external devices)	○	○

*1 Products with the first five digits of the serial No. "12012" or higher are applicable.
*2 Products with the first five digits of the serial No. "13012" or higher are applicable.
*3 Built-in Ethernet port LCPUs are applicable.
*4 QnUDVCPUs and QnUDPVCPUs with the first five digits of the serial No. "18112" or higher are applicable.
*5 Built-in Ethernet port LCPUs with the first five digits of the serial No. "18112" or higher are applicable.

Point

The following processing is also performed during service processing.

- Monitor processing of other station
- Read of another intelligent function module buffer memory by the serial communication module

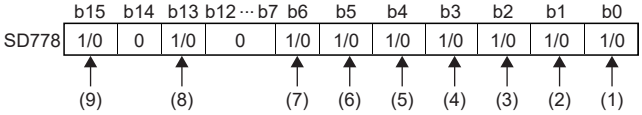
- All the processing items except I/O refresh are performed when SM775 is turned OFF.
- Selecting a processing item
  - Select a processing item in SD778 and turn ON SM775. The following table shows processing that can be specified in SD778 when SM775 is turned ON.

Processing item	QCPU		LCPU	
	When SM775 is OFF	When SM775 is ON	When SM775 is ON	When SM775 is ON
I/O refresh	Not executed	Whether to be executed or not can be selected.	Not executed	Whether to be executed or not can be selected.
CC-Link refresh	Executed		Executed	
CC-Link IE Controller Network refresh			—	—
CC-Link IE Field Network refresh			Executed	Whether to be executed or not can be selected.
CC-Link IE Field Network Basic refresh				
MELSECNET/H refresh			—	—
Auto refresh of intelligent function modules			Executed	Whether to be executed or not can be selected.
Auto refresh using QCPU standard area of multiple CPU system			—	—
Reading input/output data of all modules other than the multiple CPU system group			—	—
Auto refresh using the multiple CPU high speed transmission area of multiple CPU system			—	—
Communication with display unit	—	—	Executed	Whether to be executed or not can be selected.
Service processing (communication with programming tool, GOT, or other external devices)	Executed	Whether to be executed or not can be selected.		

  - Set an execution status for each processing in SD778. Set an execution status for each bit of SD778 as shown below.

[QCPU]

Bit of SD778	Executed	Not executed
b0 to b6, b13	1	0
b15	0	1



(1) b0: I/O refresh
(2) b1: CC-Link refresh
(3) b2: CC-Link IE Controller Network, MELSECNET/H refresh
(4) b3: Auto refresh of intelligent function module
(5) b4: Auto refresh using QCPU standard area of multiple CPU system, reading inputs/outputs from the outside of the multiple CPU system group
(6) b5: Auto refresh using the multiple CPU high speed transmission area of multiple CPU system
(7) b6: CC-Link IE Field Network refresh
(8) b13: CC-Link IE Field Network Basic refresh
(9) b15: Service processing (communication with programming tool, GOT, or other external devices)

Ex.

To make only the send/receive processing with the remote I/O station faster, designate MELSECNET/H refresh only. (Set only b2 and b15 of SD778 to 1 (SD778: 8004H).)

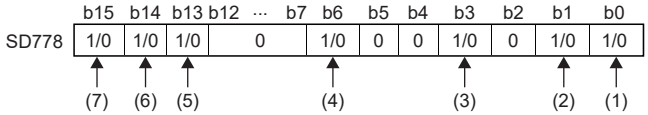
Point

Refresh between the multiple CPUs by the COM instruction is performed under the following condition.

- Receiving operation from other CPUs: When b4 of SD778 (auto refresh in the CPU shared memory) is 1.
- Sending operation from host CPU: When b15 of SD778 (execution status of service processing) is 0.

[LCPU]

Bit of SD778	Executed	Not executed
b0, b1, b3, b6, b13, b14	1	0
b15	0	1

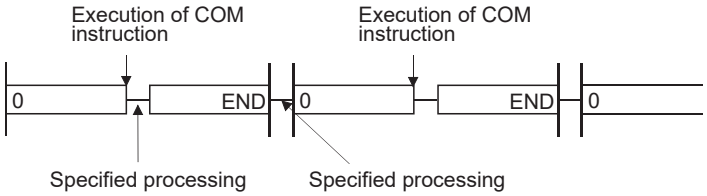


(1) b0: I/O refresh
(2) b1: CC-Link refresh
(3) b3: Auto refresh of intelligent function module
(4) b6: CC-Link IE Field Network refresh
(5) b13: CC-Link IE Field Network Basic refresh
(6) b14: Communication with display unit
(7) b15: Service processing (communication with programming tool, GOT, or other external devices)

Ex.

To speed up processing of the display unit only, specify communication with the display unit only. (Write "1" to bits b14 and b15 of SD778 (SD778: C000H).)

- At the point of the execution of the COM instruction, the CPU module temporarily stops the processing of the sequence program, and performs specified processing.



- The COM instruction can be used in a sequence program any number of times. However, note that the scan time of the sequence program will be lengthened by the time taken for the processing selected in SD778.
- Only with the Universal model QCPU and LCPU, interruption is enabled during the execution of the COM instruction. However, note that the data can be separated if the refresh data is used by an interrupt program etc.
- With the Built-in Ethernet port QCPU and Built-in Ethernet port LCPU, processing time may be increased if the service process was executed by the COM instruction while the built-in Ethernet ports are in Ethernet connection.

Point

The programs in which the COM instruction cannot be used are shown below:

- Low-speed execution type programs
- Interrupt programs
- Fixed scan execution type programs

For the redundant CPU, there are restrictions on use of the COM instruction. Refer to the manual below for details.

QnPRHCPU User's Manual (Redundant System)

Operation error

- There is no operation error in the COM instruction.

Select refresh (CCOM(P))

CCOM(P)

Basic | High performance | Process | Redundant | Ver. Universal | LCPU

- QnU(D)(H)CPU, QnUDE(H)CPU: the serial number (first five digits) is "10102" or later
- Q00UJCPU, Q00UCPU, Q01UCPU, QnUDVCPU, QnUDPVCPU: Supported

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant	Others
	Bit	Word		Bit	Word				
—	—								

Point

When executing the select refresh without an execution condition, use the COM instruction. (☞ Page 491 Select refresh (COM))

Processing details

- Refer to Page 491 Select refresh (COM).

Operation error

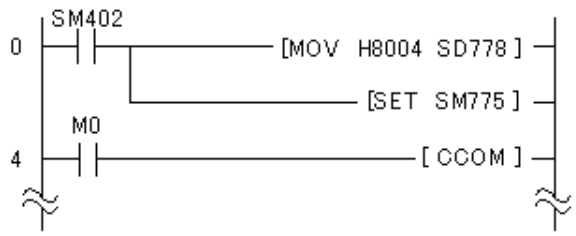
- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4100	The CCOM(P) instruction was executed in the QnU(D)(H)CPU whose serial number (first five digits) is "10101" or earlier.	—	—	—	—	○	—

Program example

- Turning on M0 enables the program to execute the select refresh, while turning off M0 disables the program to execute the select refresh.

[Ladder Mode]



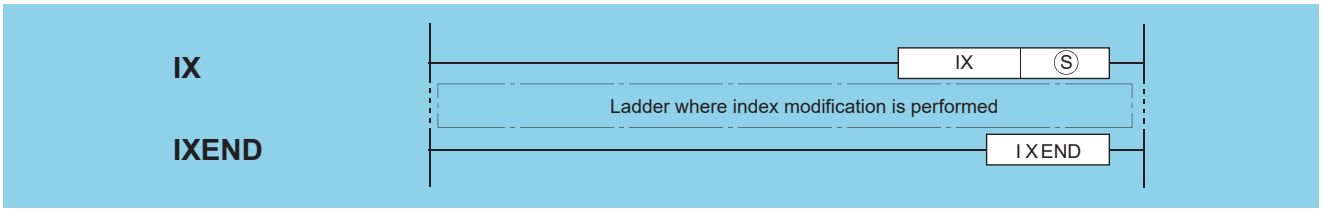
[List Mode]

Step	Instruction	Device
0	LD	SM402
1	MOV	H8004 SD778
3	SET	SM775
4	LD	M0
5	CCOM	

Index modification of entire ladder

IX, IXEND

Basic | High performance | Process | Redundant | Universal (×) | LCPU (×)



(S): Head number of the devices where index modification data is stored (BIN 16 bits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant	Others
	Bit	Word		Bit	Word				
(S)	—	○		—					

Processing details

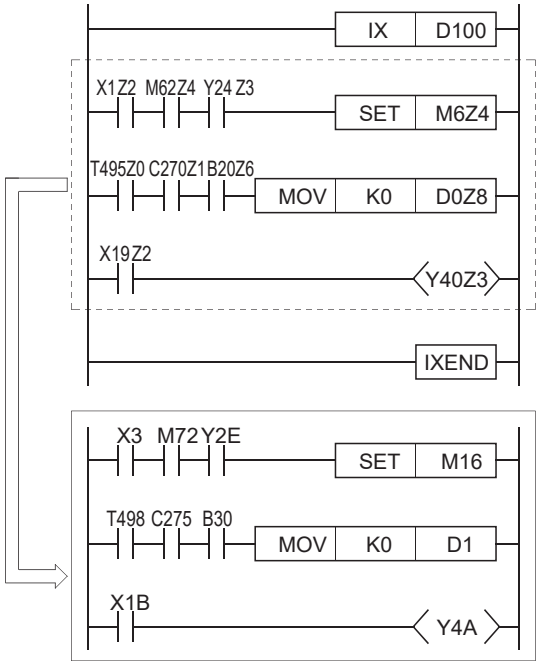
- Performs index modification on all devices in the ladder up to the IXEND instruction after the IX instruction, using the index modification value specified in the index modification table. Refer to Page 498 Designation of modification values in index modification of entire ladders for how to configure an index modification table. The configuration of the index modification table and the corresponding index register numbers are as shown below:

	Device name	Index register number
Ⓢ	Modification value of timer (T)	Z0
Ⓢ + 1	Modification value of counter (C)	Z1
Ⓢ + 2	Modification value of input (X)	Z2
Ⓢ + 3	Modification value of output (Y)	Z3
Ⓢ + 4	Modification value of internal relay (M)	Z4
Ⓢ + 5	Modification value of latch relay (L)	Z5
Ⓢ + 6	Modification value of link relay (B)	Z6
Ⓢ + 7	Modification value of edge relay (V)	Z7

	Device name	Index register number	
Ⓢ + 8	Modification value of data register (D)	Z8	
Ⓢ + 9	Modification value of link register (W)	Z9	
Ⓢ + 10	Modification value of file register (R)	Z10	*1
Ⓢ + 11	Modification value of buffer register I/O No. (U)	Z11	
Ⓢ + 12	Modification value of buffer register (G)	Z12	
Ⓢ + 13	Modification value of link direct device network No. (J)	Z13	
Ⓢ + 14	Modification value of file register (ZR)	Z14	
Ⓢ + 15	Modification value of pointer (P)	Z15	

*1 When using a basic model QCPU, index registers with numbers from Z10 onward cannot be used.

- Index modification for device numbers is accomplished in the manner as below: By setting a modification value to each of the devices, the set modification values are added to all the device numbers of the devices used in the ladder between the IX and IXEND instructions. The program is executed using the index modified device numbers.

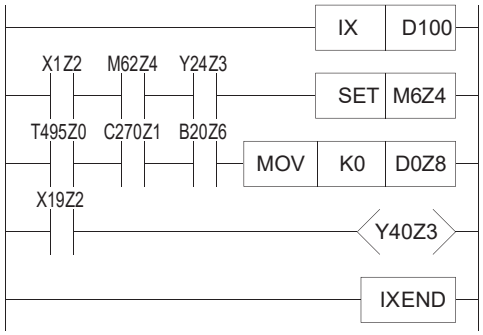


Modification value

D100	8	T	(Z0)
D101	5	C	(Z1)
D102	2	X	(Z2)
D103	10	Y	(Z3)
D104	10	M	(Z4)
D105	20	L	(Z5)
D106	16	B	(Z6)
D107	20	V	(Z7)
D108	1	D	(Z8)

· Value "2" is added to X1 and X9. → Processed as X3 and X1B, respectively.
· Value "10 (A_H)" is added to Y24 and Y40. → Processed as Y2E and Y4A, respectively.
· Value "10" is added to M6 and M62. → Processed as M16 and M72, respectively.
· Value "16 (10_H)" is added to B20. → Processed as B30.
· Value "8" is added to T495. → Processed as T498.
· Value "5" is added to C270. → Processed as C275.
· Value "1" is added to D0. → Processed as D1.

- Instructions such as the PLS, PLF, and □P instructions, which are executed only once when input conditions have been established, cannot be index modified by using the IX to IXEND instruction loop.
- In cases where adding the modification value causes the device number to exceed the device range, accurate processing will not be conducted.
- Do not execute the IX or IXEND instructions during online program changes of sequence programs (write during RUN). Accurate processing will not be conducted if this happens.
- Modification values are preset for random word devices as BIN values, and the initial device number for which modification values have been set is designated by (S).
- Do not execute a scan execution type program and an interrupt program simultaneously between the IX and IXEND instructions.
- Whether the program will be expanded or a user needs to create the program is depending on your GPP function software package. The index register should be added to the index modification ladder established with the IX and IXEND instructions. *2



*2 The value of Zn is returned to the previous Zn value before the execution of the IX instruction after the IXEND instruction has been executed.

Point

- When using the IX and IXEND instructions in both a normal sequence program and an interrupt sequence program, establish the interlock to avoid simultaneous execution. The interlock assumes the area between the IX and IXEND instructions in the normal sequence program as DI, disabling the interruption.
- The IXDEV and IXSET instructions can be used to specify modification values. For details, refer to Page 498 Designation of modification values in index modification of entire ladders.

Operation error

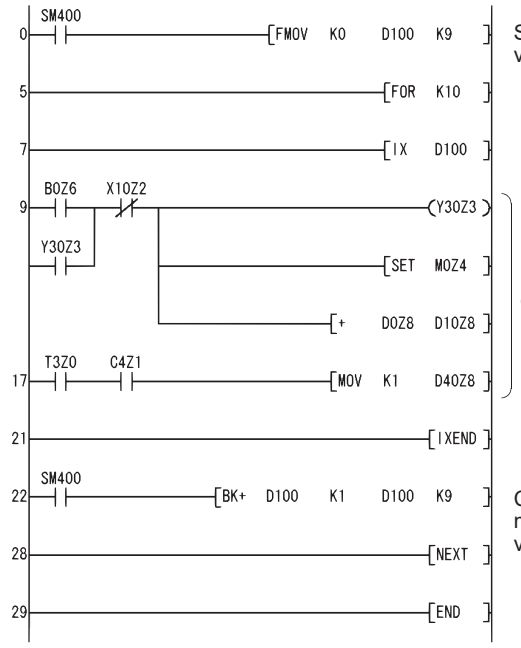
• In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4231	The IX and IXEND instructions are not used as a pair. After the IX instruction was executed, the END, FEND, GOEND, or STOP instruction was executed prior to the IXEND instruction.	○	○	○	○	—	—

Program example

• The following program executes the same ladder 10 times, while changing device numbers.

[Ladder Mode]



[List Mode]

Step	Instruction	Device			
0	LD	SM400			
1	FMOV	K0	D100	K9	
5	FOR	K10			
7	IX	D100			
9	LD	B0Z6			
10	OR	Y30Z3			
11	ANI	X10Z2			
12	OUT	Y30Z3			
13	SET	M0Z4			
14	+	D0Z8	D10Z8		
17	LD	T3Z0			
18	AND	C4Z1			
19	MOV	K1	D40Z8		
21	IXEND				
22	LD	SM400			
23	BK+	D100	K1	D100	K9
28	NEXT				
29	END				

[Operation]

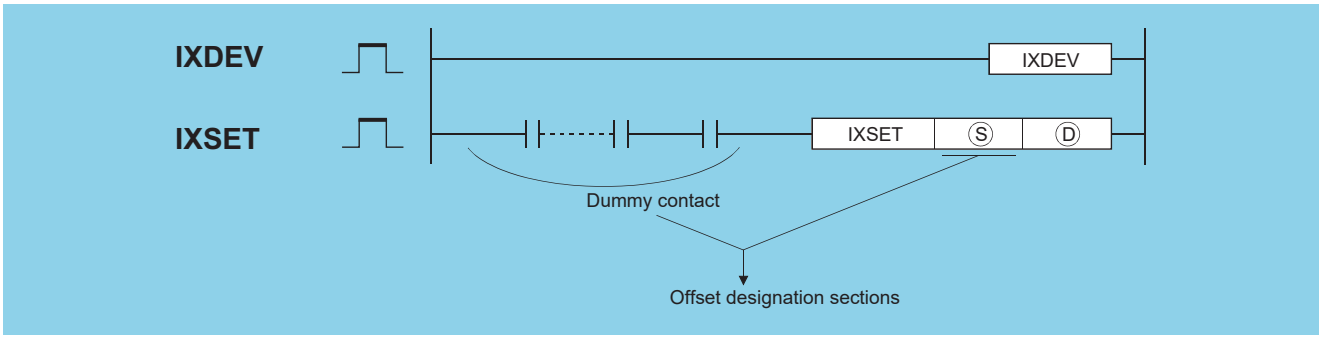
	Modification value
D100	Modification value of T
D101	Modification value of C
D102	Modification value of X
D103	Modification value of Y
D104	Modification value of M
D105	Modification value of L
D106	Modification value of B
D107	Modification value of V
D108	Modification value of D

1st time	2nd time	3rd time	10th time
B0 →	B1 →	B2 - - - - - - →	B9
X10 →	X11 →	X12 - - - - - - →	X19
Y30 →	Y31 →	Y32 - - - - - - →	Y39
M0 →	M1 →	M2 - - - - - - →	M9
D0 →	D1 →	D2 - - - - - - →	D9
D10 →	D11 →	D12 - - - - - - →	D19
T3 →	T4 →	T5 - - - - - - →	T12
C4 →	C5 →	C6 - - - - - - →	C13
D40 →	D41 →	D42 - - - - - - →	D49

Designation of modification values in index modification of entire ladders

IXDEV, IXSET

Basic | High performance | Process | Redundant | ~~Universal~~ | ~~LCPU~~



(S): Head number of the devices where index modification data is stored (pointer only) P□ (Pointer)
(D): Head number of the devices where index modification data will be stored (except a pointer) (BIN 16 bits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant	Others
	Bit	Word		Bit	Word				P
(S)	—	—		—					○
(D)	—	○		—					—

Processing details

- The IXDEV and IXSET instructions are used to configure an index modification table used in the IX and IXEND instructions.
- The device offset value designated at the offset designation area is set at the index modification table designated by (D).
- The value 0 will be entered if no designation is made.
- Word devices are also indicated by contact (word device bit designation). Data register 10 (D10) is designated with D10.0. (Any value from 0 to F can be used for the bit number.)
- Designation is made according to the method described below. *1 (The symbol □ is where the offset value will be. The notation XX indicates random selection.)

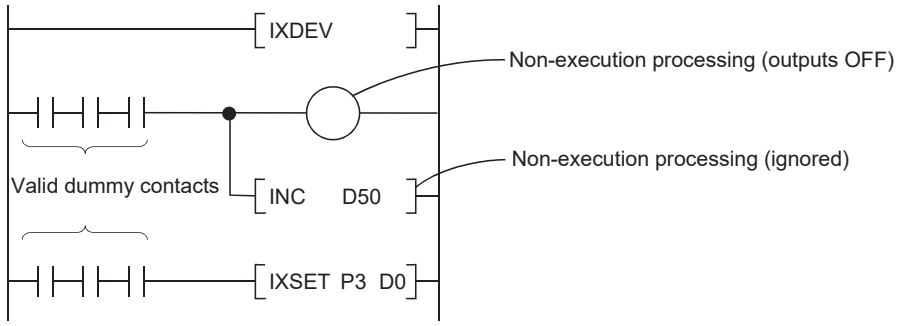
Device	T	C	X	Y	M	L	V	B	D
Designation method	T□	C□	X□	Y□	M□	L□	V□	B□	D□.XX

Device	W	R	U/G	J	ZR	P
Designation method	W□.XX	R□.XX	U□\G□.XX	J□\B□ *2	ZR□.XX	IXSET S D *3

*1 When using a basic model QCPU, the devices R, U/G, J, ZR and P cannot be used.
*2 Devices following J□\ designate B, W, X, or Y, and the offset value is also set in correspondence with this.
*3 When using a basic model QCPU, specify a dummy device number.
(S) is P□.

- If two offsets for two identical types of device have been set in the offset designation area, the last value set will be valid.
- The IXDEV and IXSET instructions should be treated as a pair.
- Any value from 0 to 32767 is valid for ZR. (The offset value will be the remainder of the quotient of the designated device number divided by 32768.)
- The dummy contacts in the offset specifying part are valid for only LD and AND located within the range of the IXDEV-IXSET instructions. The IXDEV-IXSET instructions will not be executed if other instructions are described.

Ex.



Operation error

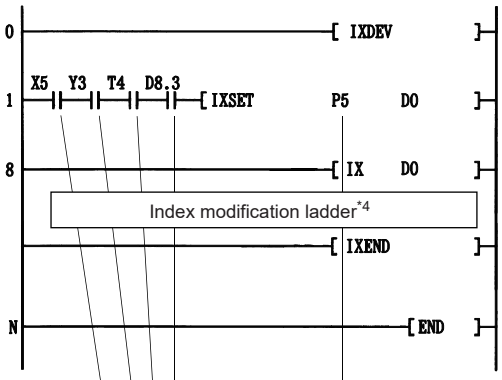
- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4231	The IXDEV and IXSET instructions are not used as a pair.	○	○	○	○	○	○

Program example

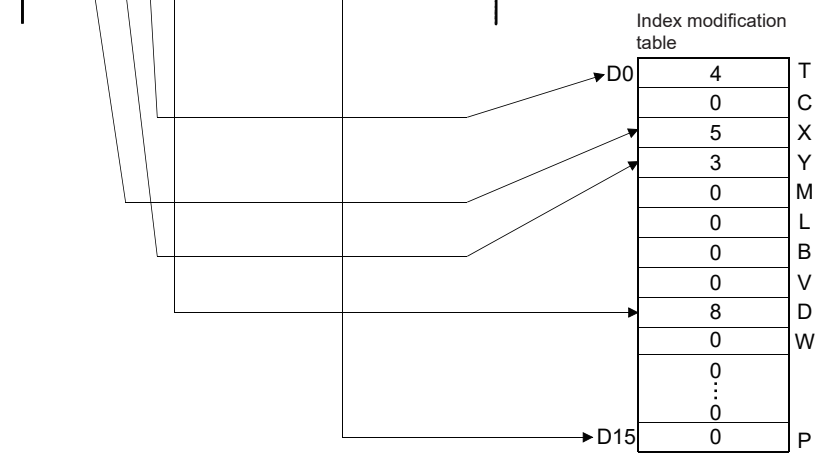
- The following program changes the modification values for input (X), output (Y), data register (D) and pointer (P). When using a basic model QCPU, the devices R, U/G, J, ZR and P cannot be used.

[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	IXDEV	
1	LD	X5
2	AND	Y3
3	AND	T4
4	AND	D8.3
5	IXSET	P5
		D0
8	IX	D0
⋮	IXEND	
N	END	



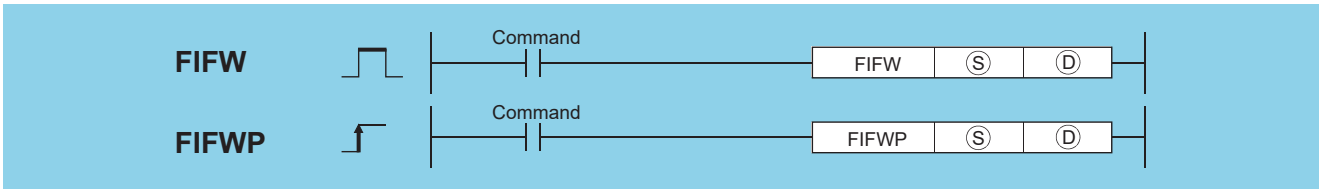
*4 Refer to Page 495 Index modification of entire ladder for index modification using the IX to IXEND instructions.

7.7 Data Table Operation Instructions

Writing data to the data table

FIFW(P)

Basic | High performance | Process | Redundant | Universal | LCPU

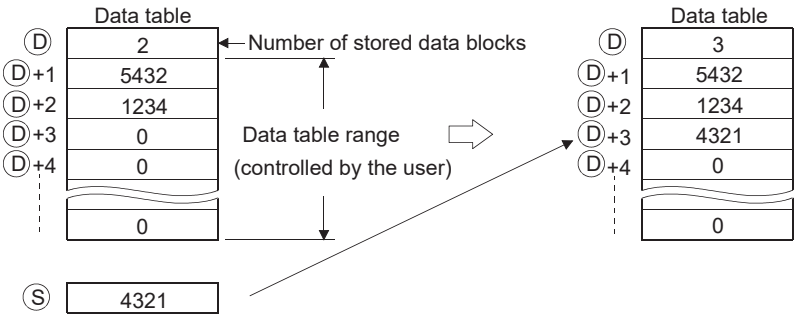


(S): Data to be written into the table or the number of the device where the data is stored (BIN 16 bits)
(D): Head number of the table (BIN 16 bits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(S)	○	○		○				○	—
(D)	—	○		—				—	—

Processing details

- Stores the 16-bit data designated by (S) in the data table designated by (D). The number of data blocks stored in the table is stored at (D), and the data designated by (S) is stored in sequence from (D)+1.



- When the FIFW(P) instruction is executed for the first time, device values specified by (D) should be cleared.
- The number of data blocks to be written in the data table and the data table range should be controlled by the user.

Operation error

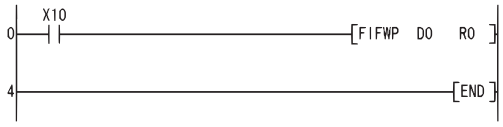
- In any of the following cases, an operation error occurs, the error flag (SM0) turns on, and an error code is stored in SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4100	The FIFW(P) instruction was executed when the value of (D) was FFFFH.	○	○	○	○	○	○
4101	The data table range has exceeded the range of the corresponding devices at execution of the FIFW(P) instruction.	○	○	○	○	○	○

Program example

- The following program stores the data at D0 to the data table following R0 when X10 is turned ON.

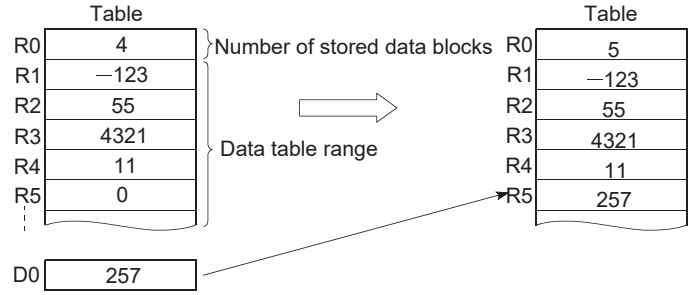
[Ladder Mode]



[List Mode]

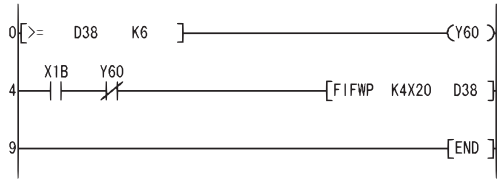
Step	Instruction	Device	
0	LD	X10	
1	FIFWP	D0	R0
4	END		

[Operation]



- The following program stores the data at X20 to X2F to data table of D38 to D44 table when X1B is turned ON, and, if there are more than 6 data blocks to be stored, turns Y60 ON and disables the FIFW instruction.

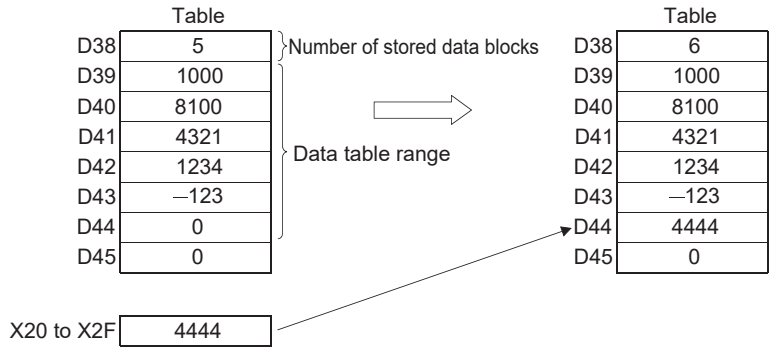
[Ladder Mode]



[List Mode]

Step	Instruction	Device	
0	LD>=	D38	K6
3	OUT	Y60	
4	LD	X1B	
5	ANI	Y60	
6	FIFWP	K4X20	D38
9	END		

[Operation]



Reading oldest data from tables

FIFR(P)

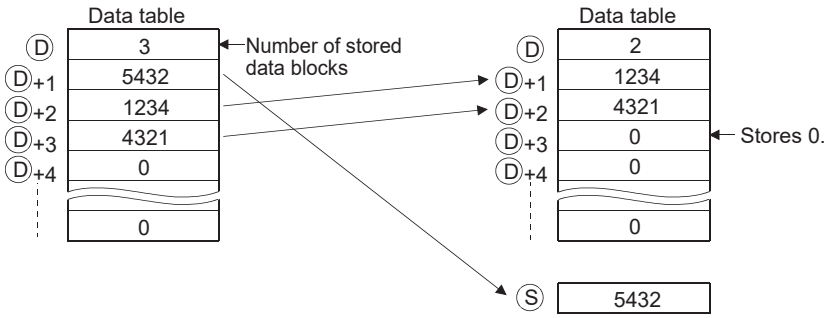
Basic | High performance | Process | Redundant | Universal | LCPU

(S): Head number of the devices where the data read from the table will be stored (BIN 16 bits)
(D): Head number of the table (BIN 16 bits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant	Others
	Bit	Word		Bit	Word				
(S)	○	○		○				—	
(D)	—	○		—				—	

Processing details

- Stores the oldest data ((D)+1) input to the table designated by (D) at the device designated by (S). After execution of the FIFR(P) instruction, the data in the data table is all compressed up by one block.



- Provide an interlock circuit not to execute the FIFR(P) instruction when the value stored in (D) is 0.

Operation error

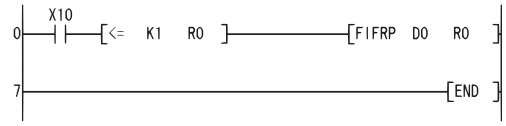
- In any of the following cases, an operation error occurs, the error flag (SM0) turns on, and an error code is stored in SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4100	The FIFR(P) instruction was executed when the value of (D) was 0.	○	○	○	○	○	○
4101	The data table range has exceeded the range of the corresponding devices at execution of the FIFR(P) instruction.	○	○	○	○	○	○

Program example

- The following program stores the R1 data from the table R0 to R7 at D0 when X10 is turned ON.

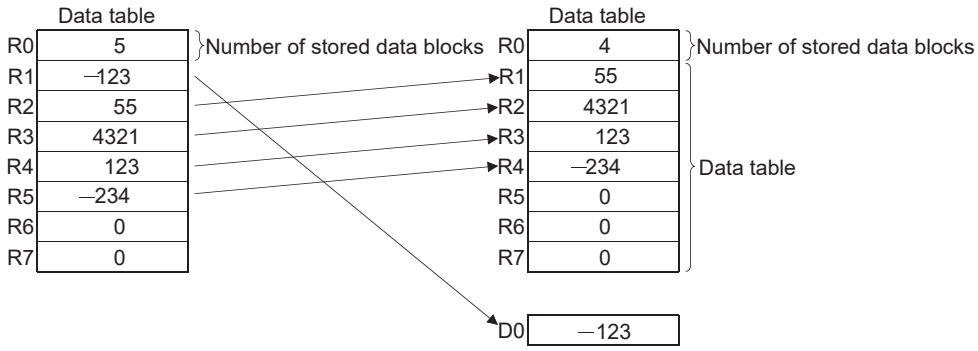
[Ladder Mode]



[List Mode]

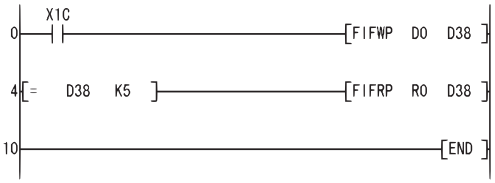
Step	Instruction	Device	
0	LD	X10	
1	AND<=	K1	R0
4	FIFRP	D0	R0
7	END		

[Operation]



- The following program stores the data at D0 in the data table D38 to D43, and, when the table stores 5 data, stores the data at D39 of the data table in R0, when X1C is turned ON.

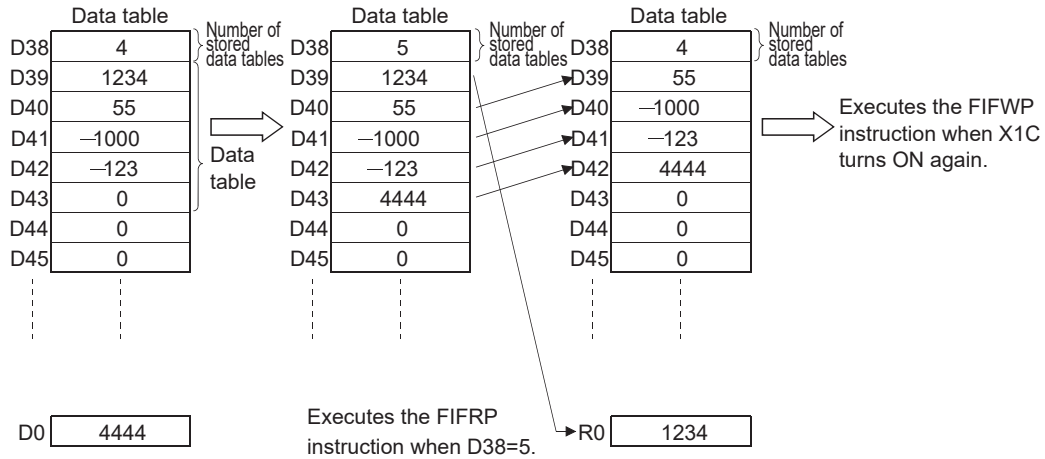
[Ladder Mode]



[List Mode]

Step	Instruction	Device	
0	LD	X1C	
1	FIFWP	D0	D38
4	LD=	D38	K5
7	FIFRP	R0	D38
10	END		

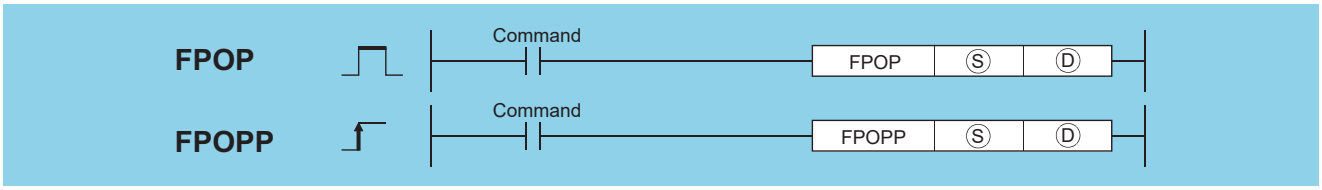
[Operation]



Reading newest data from data tables

FPOP(P)

Basic | High performance | Process | Redundant | Universal | LCPU

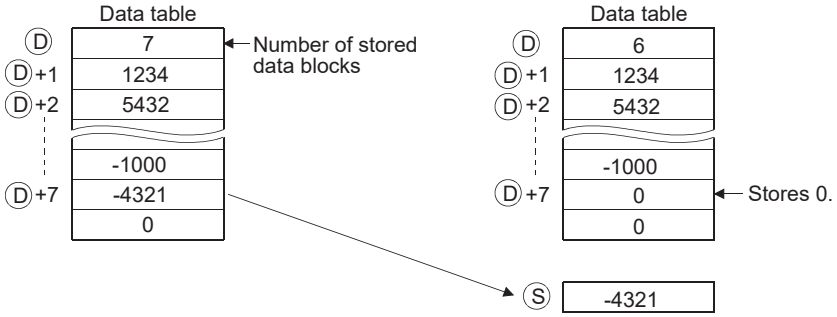


(S): Head number of the devices where the data read from the table will be stored (BIN 16 bits)
(D): Head number of the table (BIN 16 bits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant	Others
	Bit	Word		Bit	Word				
(S)	○	○		○				—	
(D)	—	○		—				—	

Processing details

- Stores the newest data input to the table designated by (D) at the device designated by (S). After the execution of the FPOP instruction, the device storing the data read by the FPOP instruction is reset to 0.



- Perform interlock to avoid executing the FPOP instruction when the value stored at (D) is 0.

Operation error

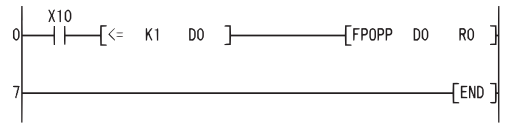
- In any of the following cases, an operation error occurs, the error flag (SM0) turns on, and an error code is stored in SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4100	The FPOP instruction was executed when the value of (D) was 0.	○	○	○	○	○	○
4101	The data table range exceeded the range of the corresponding device at the execution of the FPOP instruction.	○	○	○	○	○	○

Program example

• The following program stores the data stored last in the data table R0 to R7 at D0 when X10 is turned ON.

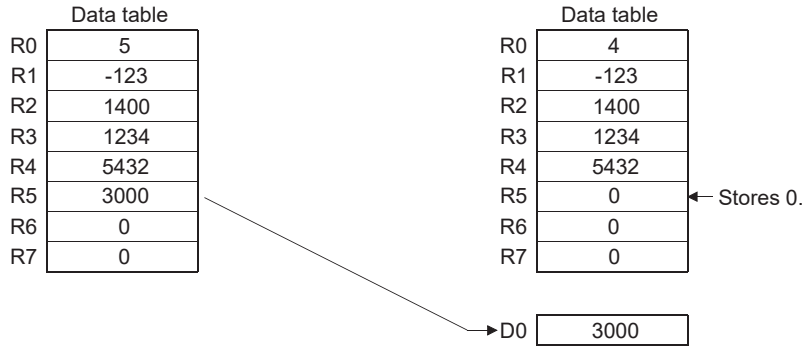
[Ladder Mode]



[List Mode]

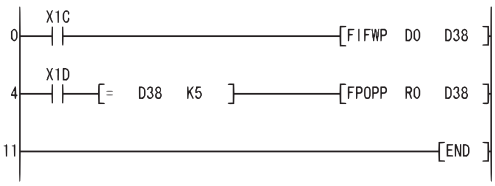
Step	Instruction	Device	
0	LD	X10	
1	AND< =	K1	D0
4	FPOPP	D0	R0
7	END		

[Operation]



• The following program stores the data at D0 in the data table D38 to D43 when X1C is turned ON, and when the number of data stores in the table reaches 5, turns X1D ON, and stores the data stored last in the data table to R0.

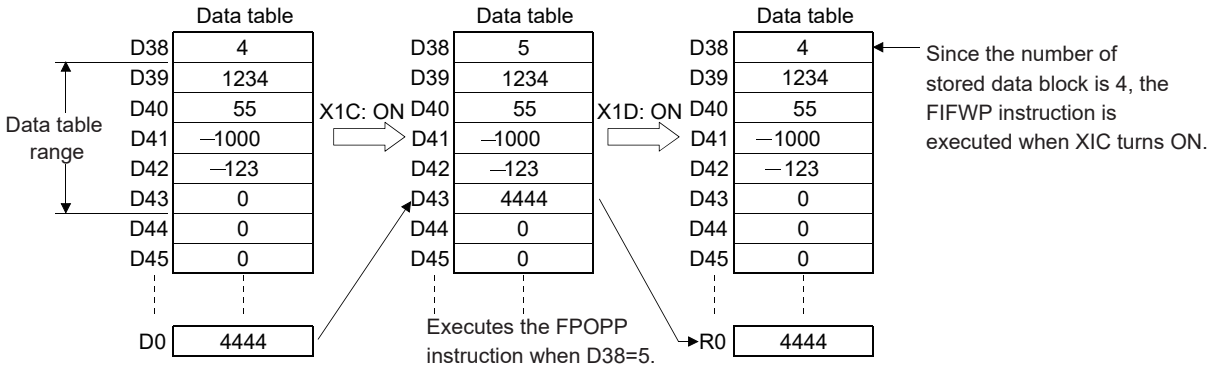
[Ladder Mode]



[List Mode]

Step	Instruction	Device	
0	LD	X1C	
1	FIFWP	D0	D38
4	LD	X1D	
5	AND=	D38	K5
8	FPOPP	R0	D38
11	END		

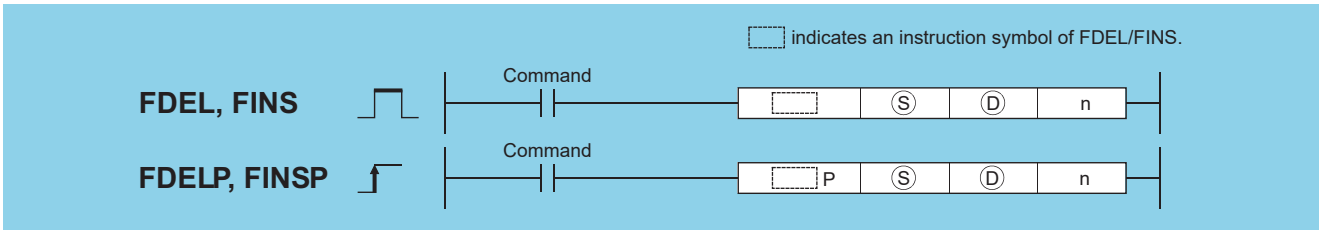
[Operation]



Deletion of data from data tables, insertion of data in data tables

FDEL(P), FINS(P)

Basic | High performance | Process | Redundant | Universal | LCPU



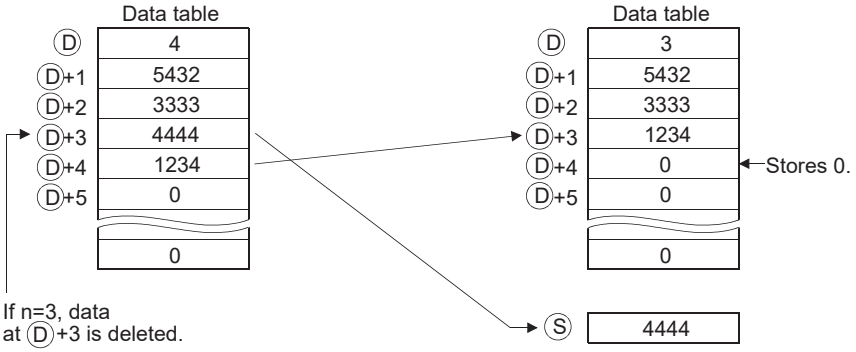
(S): Head number of the devices where data to be inserted is stored (BIN 16 bits), head number of the devices where the data to be deleted will be stored (BIN 16 bits)
(D): Head number of the table (BIN 16 bits)
n: Location on the table where data is inserted/deleted (BIN 16 bits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(S)	○	○		○				—	—
(D)	—	○		—				—	—
n	○	○		○				○	—

Processing details

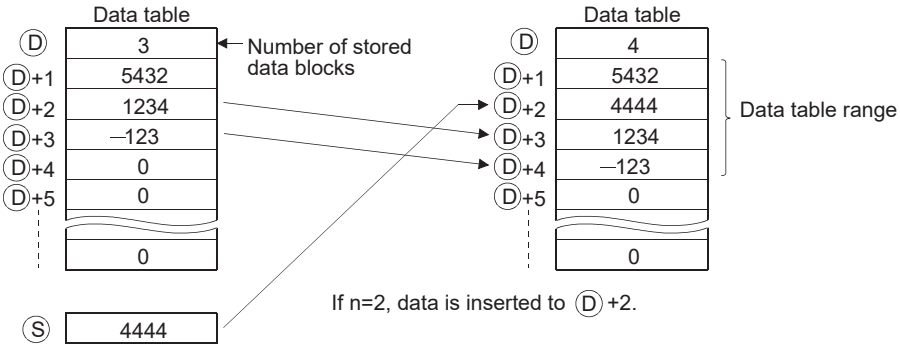
■FDEL

- Deletes the nth block of data from the data table designated by (D), and stores it at the device designated by (S). After the execution of the FDEL instruction, the data in the table following the deleted block is compressed forward by one block.



■FINS

- Inserts the 16-bit data designated by (S) at the nth block of the data table designated by (D). After the execution of the FINS instruction, the data in the table following the inserted block is all dropped one position.



Operation error

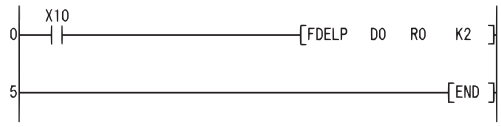
• In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4100	The FDEL or FINS instruction was executed when n=0. The FDEL instruction was executed when the value of (D) was 0. The FINS instruction was executed when the value of (D) was FFFFH.	○	○	○	○	○	○
4101	The Nth position from (D) is larger than the number of data storage at the execution of the FDEL instruction. The Nth position from (D) is larger than the "number of data storage + 1" at the execution of the FINS instruction. The value of n in the case of the FDEL, FINS instruction exceeds the device range of the table (D). The data table range exceededs the range of the corresponding device at execution of the FDEL or FINS instruction.	○	○	○	○	○	○

Program example

• The following program deletes the second data from the table R0 to R7 and stores the deleted data at D0 when X10 is turned ON.

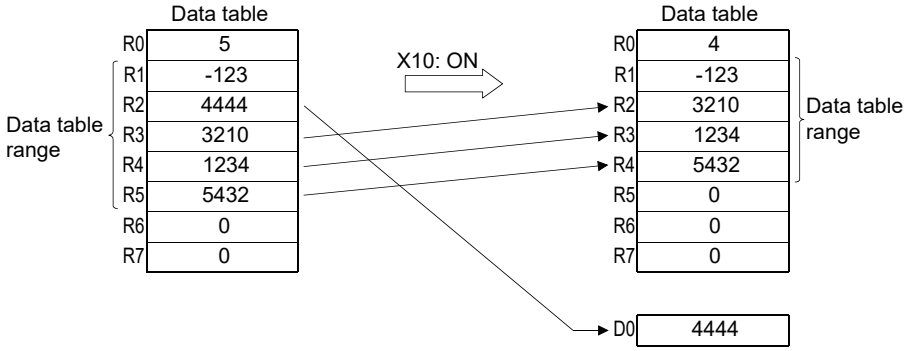
[Ladder Mode]



[List Mode]

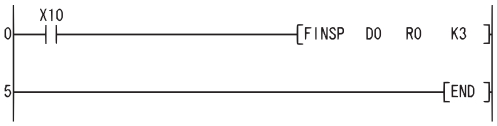
Step	Instruction	Device		
0	LD	X10		
1	FDELP	D0	R0	K2
5	END			

[Operation]



• The following program inserts the data at D0 into the third position at the table R0 to R7 when X10 is turned ON.

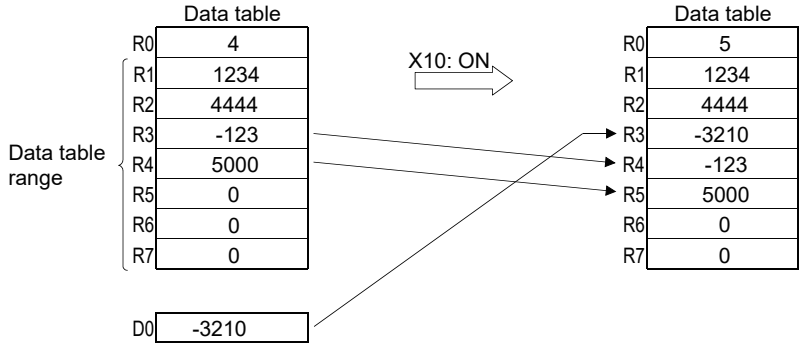
[Ladder Mode]



[List Mode]

Step	Instruction	Device		
0	LD	X10		
1	FINSP	D0	R0	K3
5	END			

[Operation]

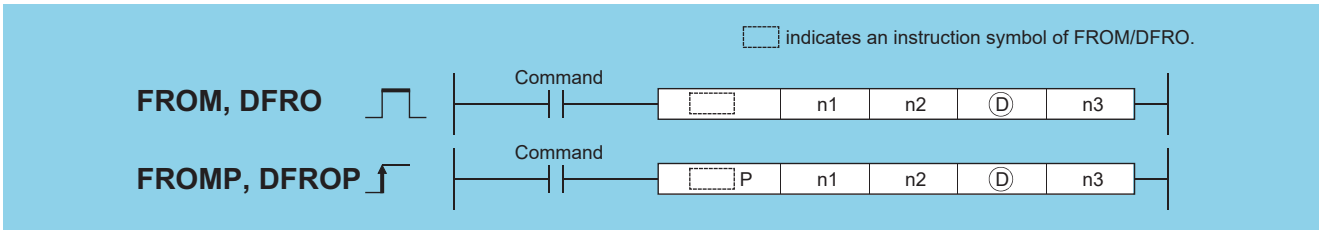


7.8 Buffer Memory Access Instructions

Reading 1-word data from the intelligent function module, reading 2-word data from the intelligent function module

FROM(P), DFRO(P)

Basic | High performance | Process | Redundant | Universal | LCPU



n1: Head I/O number of an intelligent function module*1 (BIN 16 bits)
n2: Head address of the buffer memory where data to be read is stored (BIN 16 bits)
(D): Head number of the devices where the read data will be stored (BIN 16/32 bits)
n3: Number of read data (BIN 16 bits)

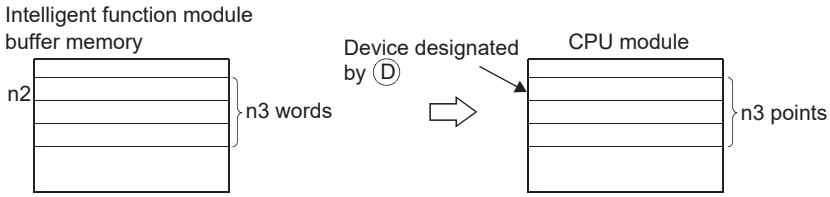
Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others U
	Bit	Word		Bit	Word				
n1	○			○					○
n2	○			○					—
(D)	○			—					—
n3	○			○					—

*1 Specified with the upper three digits when the head I/O number is expressed in 4 hexadecimal digits.

Processing details

■FROM

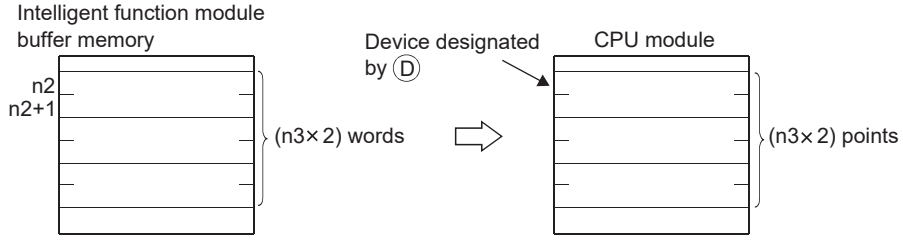
- Reads the data in n3 words from the buffer memory address designated by n2 of the intelligent function module designated by n1, and stores the data into the area starting from the device designated by (D).



- An instruction which has been executed will result in non-processing if it fails to access the target module because the module is faulty or busy in processing.

■DFRO

- Reads the data in (n3 × 2) words from the buffer memory address designated by n2 of the the intelligent function module designated by n1, and stores the data into the area starting from the device designated by (D).



- An instruction which has been executed will result in non-processing if it fails to access the target module because the module is faulty or busy in processing.

Point

Data read from intelligent function modules is also possible with the use of an intelligent function module device.

For intelligent function module device, refer to the User's Manual (Function Explanation, Program Fundamentals) for the CPU module used.

Operation error

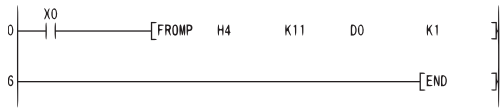
- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
1402	An error has been detected in an intelligent function module at the execution of the instruction.	○	○	○	○	○	○
1412	There has been no exchange of signals with an intelligent function module at the execution of the instruction.	○	○	○	○	○	○
2110	The I/O number specified in n1 is not for the intelligent function module.	○	○	○	○	○	○
4101	The range of n3 points (2 × n3 points for the DFRO) from the device specified in (D) exceeds the specified device range. The address specified in n2 is outside the buffer memory range.	○	○	○	○	○	○

Program example

- The following program reads CH1 digital output value of the Q68ADV at I/O numbers 040 to 04F to D0 when X0 is turned on (reads data by one word from the buffer memory address 11).

[Ladder Mode]

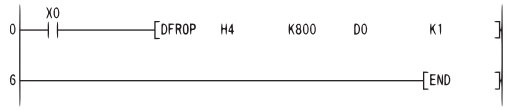


[List Mode]

Step	Instruction	Device			
0	LD	X0			
1	FROMP	H4	K11	D0	K1
6	END				

- The following program reads the current feed value of axis 1 of the QD75P4 at I/O numbers 040 to 05F to D0 and D1 when X0 is turned on (reads data by two words from the buffer memory address 800).

[Ladder Mode]



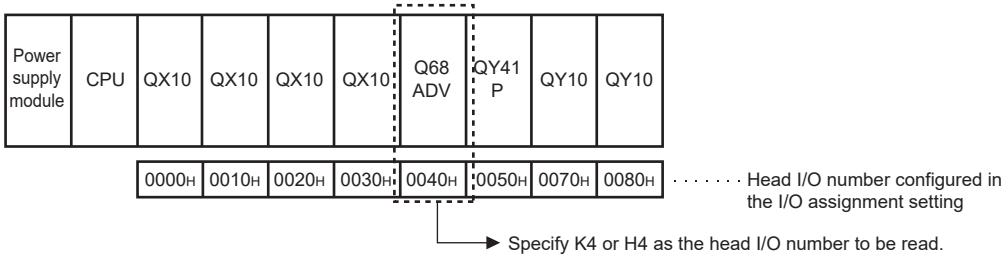
[List Mode]

Step	Instruction	Device			
0	LD	X0			
1	DFROP	H4	K800	D0	K1
6	END				

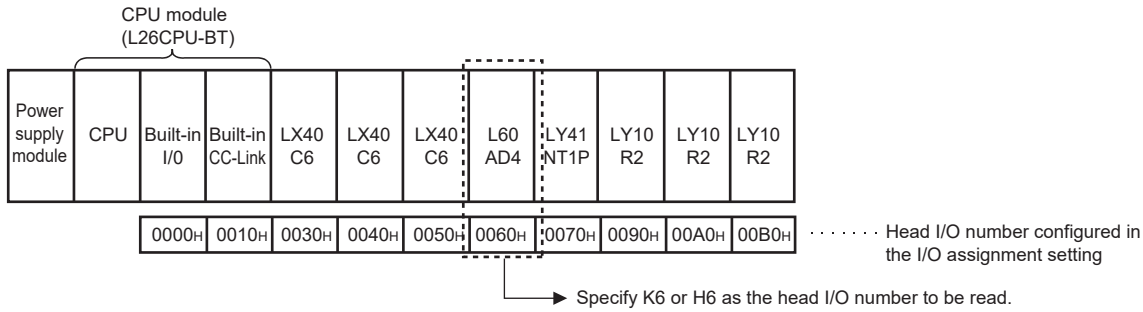
Point

- The value of n1 is specified by the upper 3 digits of hexadecimal 4 digits which represent the head I/O number of an intelligent function module.

[QCPU]



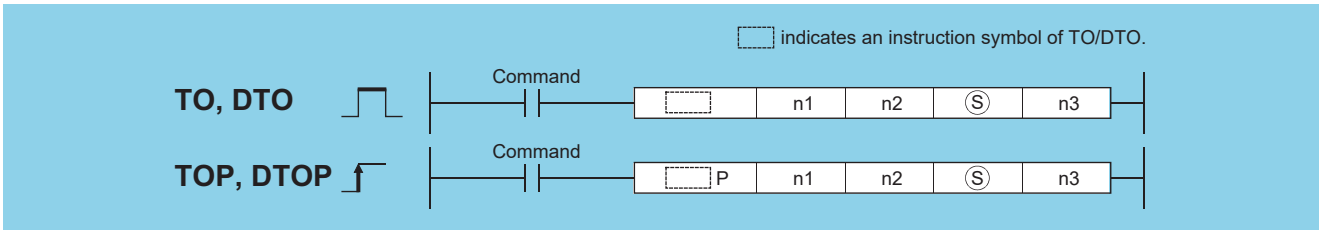
[LCPU]



Writing 1-word data to the intelligent function module, writing 2-word data to the intelligent function module

TO(P), DTO(P)

Basic | High performance | Process | Redundant | Universal | LCPU



n1: Head I/O number of an intelligent function module*1 (BIN 16 bits)
n2: Head address of the area where data is written (BIN 16 bits)
(S): Data to be written or head number of the devices where the data to be written is stored (BIN 16/32 bits)
n3: Number of data blocks to be written (BIN 16 bits)

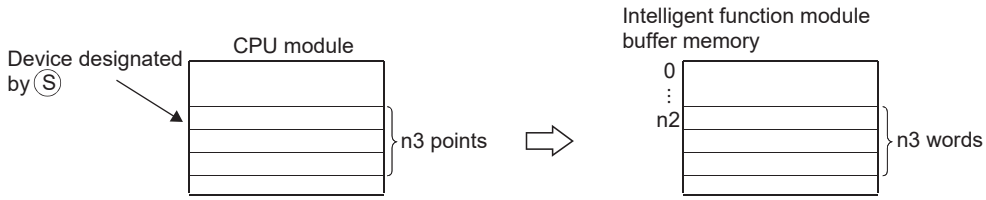
Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others U
	Bit	Word		Bit	Word				
n1	○			○				○	○
n2	○			○				○	—
(S)	○			—				○	—
n3	○			○				○	—

*1 Specified with the upper three digits when the head I/O number is expressed in 4 hexadecimal digits.

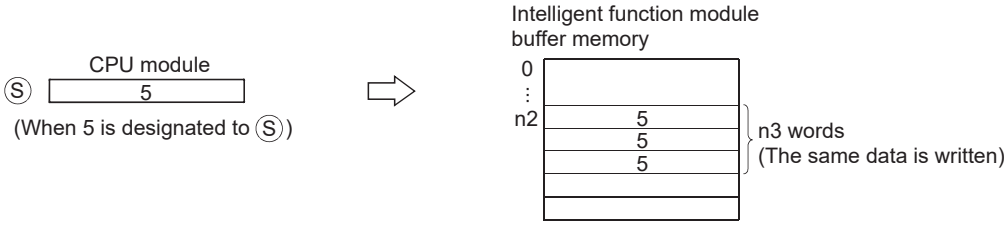
Processing details

■TO

- Writes the data stored in n3 points starting from the device designated by (S) into the area starting from buffer memory address designated by n2 of the intelligent function module designated by n1.



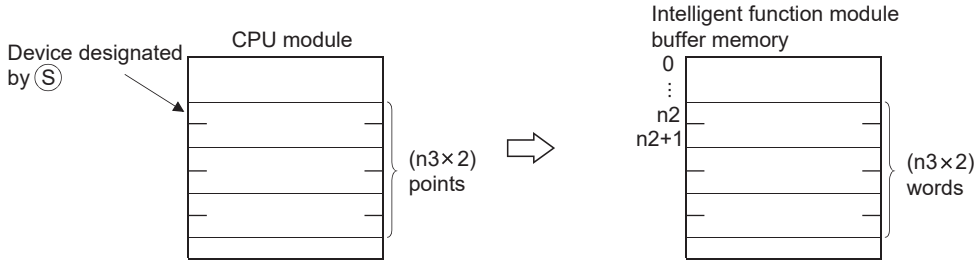
- When a constant is designated to (S), writes the same data (value designated to (S)) to the area of n3 words starting from the specified buffer memory. ((S) can be designated in the following range: -32768 to 32767 or 0H to FFFFH.)



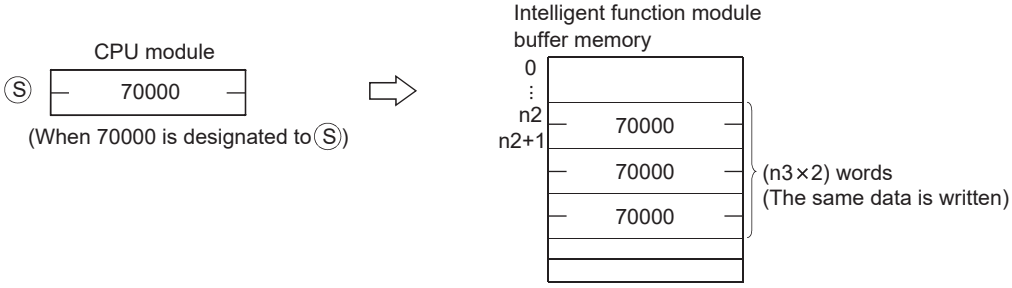
- An instruction which has been executed will result in non-processing if it fails to access the target module because the module is faulty or busy in processing.

■DTO

- Writes the data stored in n3 × 2 points starting from the device designated by (S) into the area starting from buffer memory address designated by n2 of the intelligent function module designated by n1.



- When a constant is designated to (S), writes the same data (value designated to (S)) to the area of n3 × 2 words starting from the specified buffer memory. ((S) can be designated in the following range: -2147483648 to 2147483647 or 0H to FFFFFFFFH.)



- An instruction which has been executed will result in non-processing if it fails to access the target module because the module is faulty or busy in processing.

Point

Data write to intelligent function modules is also possible with the use of an intelligent function module device. For intelligent function module device, refer to the User's Manual (Function Explanation, Program Fundamentals) for the CPU module used.

Operation error

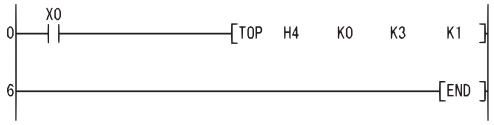
- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
1402	An error has been detected in an intelligent function module at the execution of the instruction.	○	○	○	○	○	○
1412	There has been no exchange of signals with an intelligent function module at the execution of the instruction.	○	○	○	○	○	○
2110	The I/O number specified in n1 is not for the intelligent function module.	○	○	○	○	○	○
4101	The range of n3 points (2 × n3 points for the DTO) from the device specified in (S) exceeds the specified device range. The address specified in n2 is outside the buffer memory range.	○	○	○	○	○	○

Program example

- The following program sets "A/D conversion disabled" to the CH1 and CH2 of the Q68ADV at I/O numbers 040 to 04F when X0 is turned on (writes "3" to the buffer memory address 0).

[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	X0
1	TOP	H4 K0 K3 K1
6	END	

- The following program zeroes the positioning address/movement amount of axis 1 of the QD75P4 at I/O numbers 040 to 05F when X0 is turned on (writes 0 to the buffer memory addresses 2006 and 2007).

[Ladder Mode]

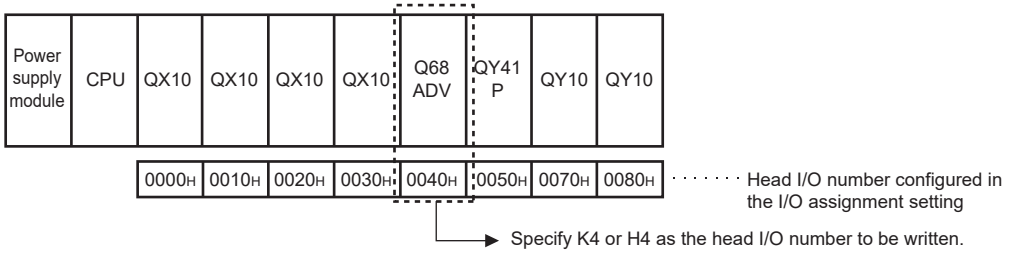
[List Mode]

Step	Instruction	Device
0	LD	X0
1	DTOP	U4 K2006 K0 K1
7	END	

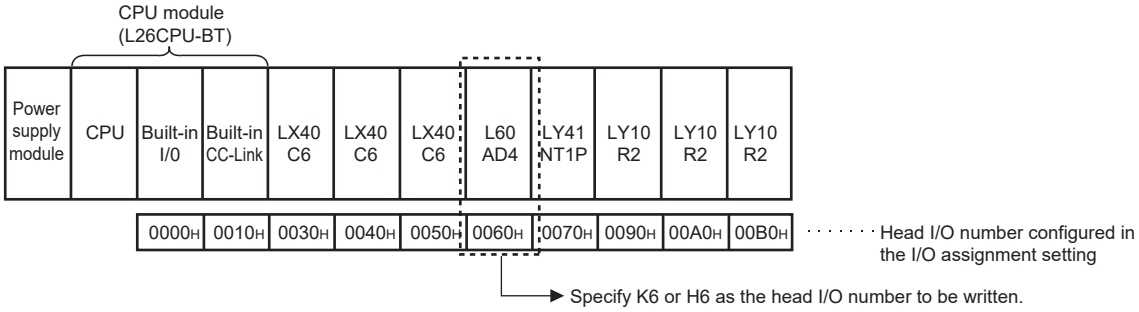
Point

- The value of n1 is specified by the upper 3 digits of hexadecimal 4 digits which represent the head I/O number of an intelligent function module.

[QCPU]



[LCPU]

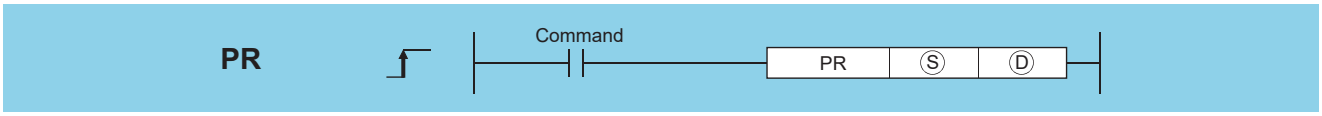


7.9 Display Instructions

Print ASCII code

PR

Basic | High performance | Process | Redundant | Universal | LCPU



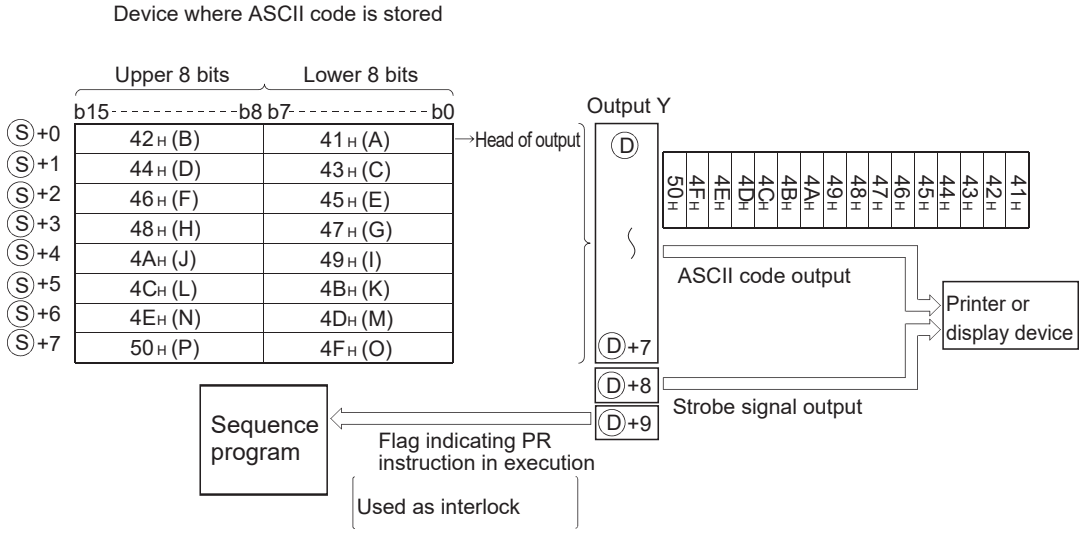
(S): ASCII code or head number of the devices where the ASCII code is stored (character string)
(D): Head number of the output module to which the ASCII code will be output (bits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant $	Others
	Bit	Word		Bit	Word				
(S)	—	△*1		—			○	○	—
(D)	○ (Only Y)	—		—			○	—	—

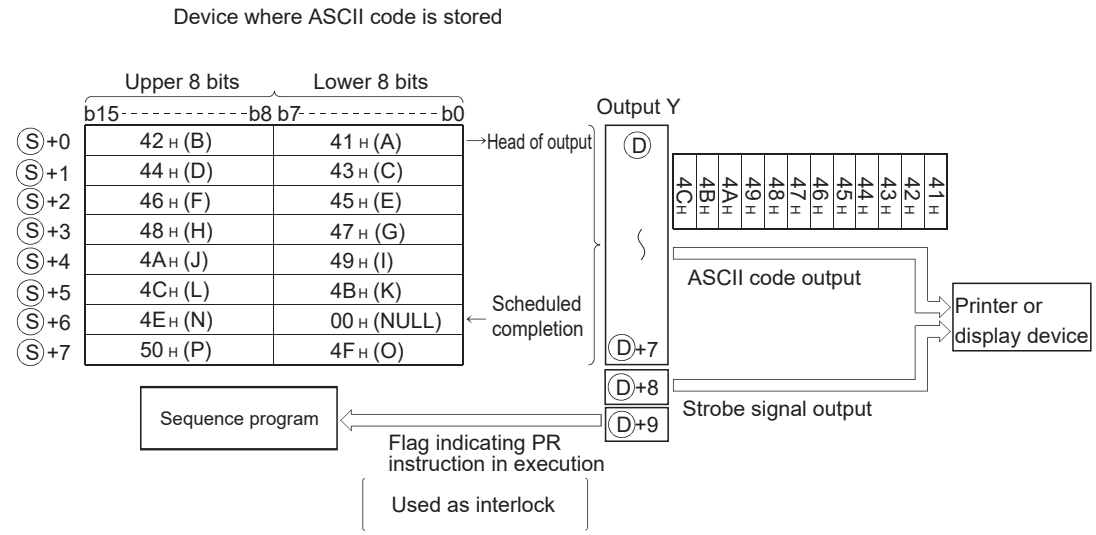
*1 Local devices and the file registers set for individual programs cannot be used.

Processing details

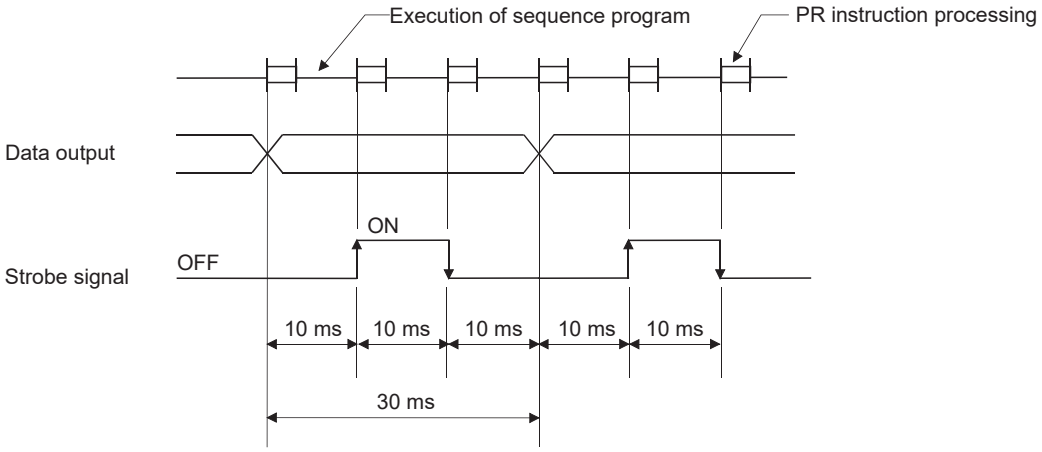
- Outputs ASCII code stored in the device specified by (S) or ASCII code stored in the area starting from the device number to an output module specified by (D). The number of characters output differs according to the ON/OFF status of SM701 (number of output characters selection).
  - If SM701 is ON, characters 8 points (16 characters) from the device designated by (S) will be the target of the operation.



• When SM701 is off, everything from the device specified by (S) to the NULL code "00H" will be the target of the operation.



- The number of points used by the output module is 10 points from the Y address designated by (D).
- Output signals from the output module are transmitted at the rate of 30ms per character. For this reason, the time required to the completion of the transmission of the designated number of characters (n) will be 30ms × n (ms). At 10ms interrupt intervals, the PR instruction executes data output, strobe signal ON, and strobe signal OFF. The other instructions are executed continuously during a period between the above processings.



- In addition to the ASCII code, the output module also outputs a strobe signal (10ms ON, 20ms OFF) from the (D)+8 device.
- Following the execution of the PR instruction, the PR instruction execution flag ((D)+9 device) remains ON until the completion of the transmission of the designated number of characters.
- The PR and PRC instructions can be used multiple times, but it is preferable to establish an interlock with the PR instruction execution flag ((D)+9 device) so that they will not be ON simultaneously.
- If the contents of the device in which ASCII codes are stored changes during the ASCII code output, the modified data after change will be output.

Operation error

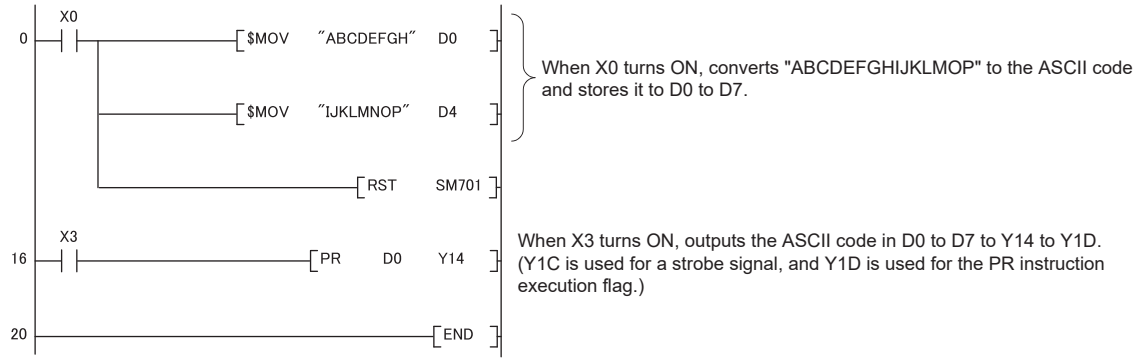
- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4101	When SM701 is off, there is no NULL code "00H" within the device range specified by (S).	—	○	○	—	—	—

Program example

- The following program converts the string "ABCDEFGHIJKLMNOP" to ASCII code when X0 is turned ON and stores it from D0 to D7, and then outputs the ASCII code at D0 to D7 to Y14 to Y1D when X3 is turned ON (when SM701 is OFF).

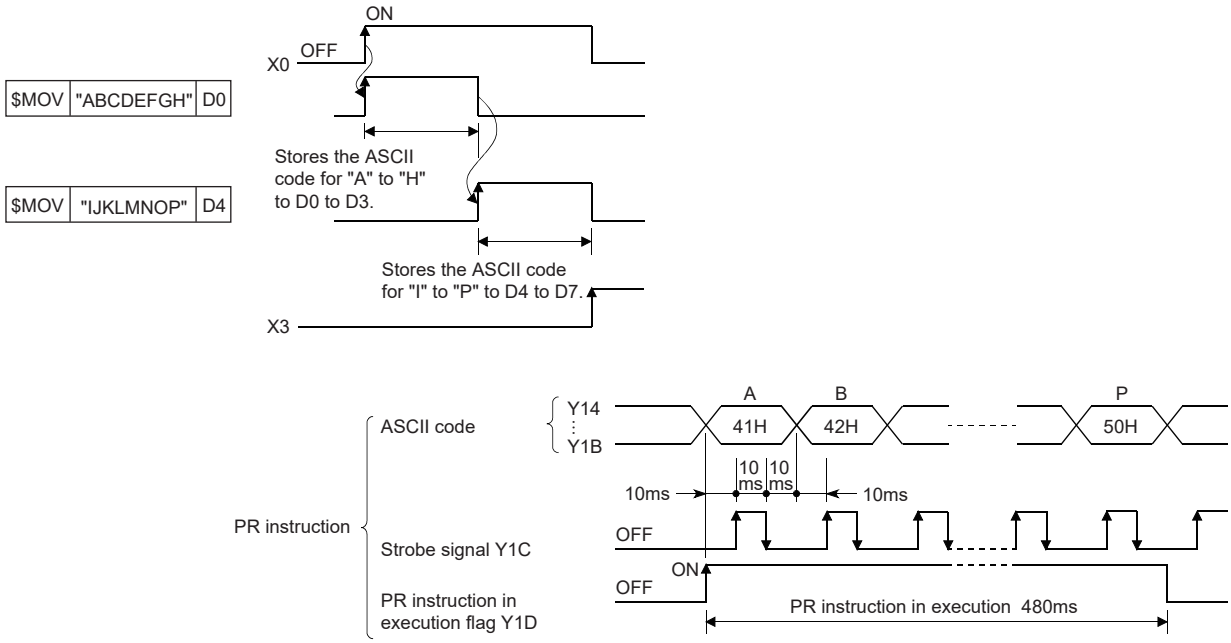
[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	X0
1	$MOV	"ABCDEFGH" D0
8	$MOV	"IJKLMNOP" D4
15	RST	SM701
16	LD	X3
17	PR	D0 Y14
20	END	

[Timing Chart]



Print comment

PRC

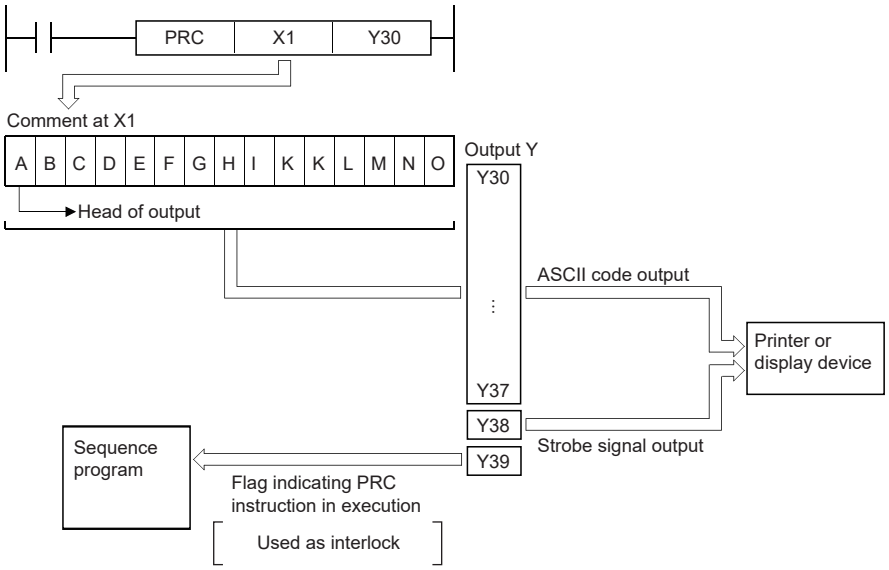
Basic | High performance | Process | Redundant | Universal | LCPU

(S): Head number of the device which prints the comment (Device name)
(D): Head number of the output module which outputs the comment (bits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant	Others P, I, J, U
	Bit	Word		Bit	Word				
(S)	○	○		○			—	—	○
(D)	○ (Only Y)	—		—			—	—	—

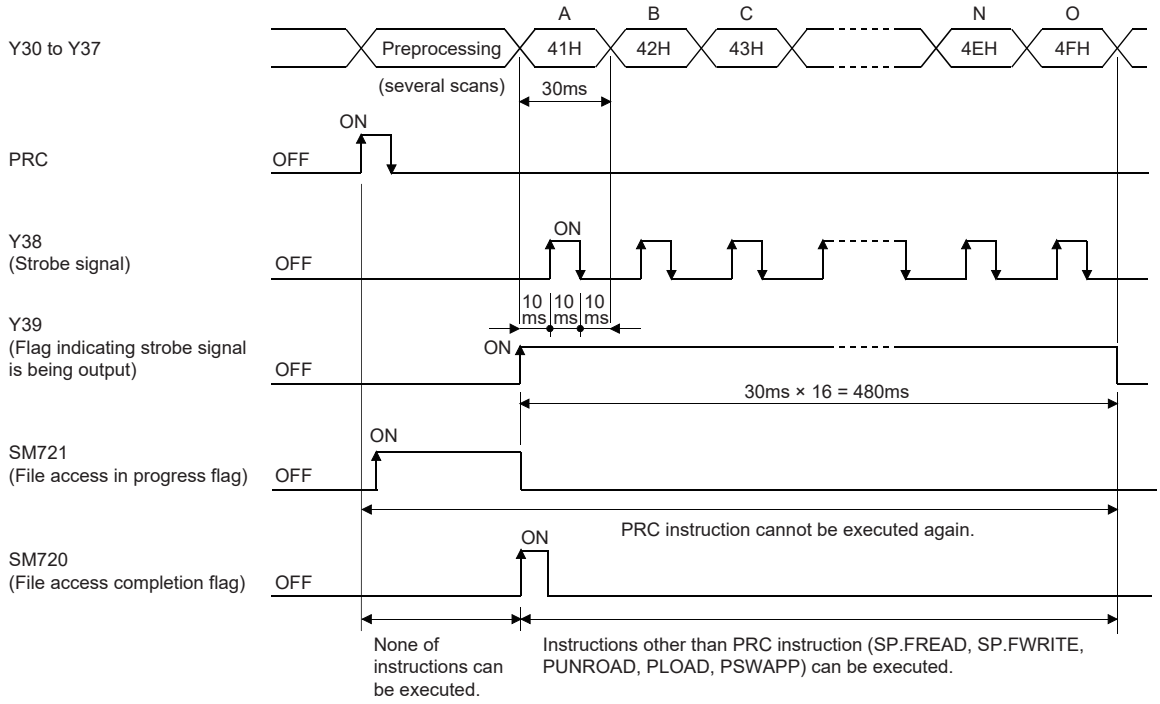
Processing details

- Outputs comment (ASCII code) at device designated by (S) to output module designated by (D). The number of characters output differs according to the ON/OFF status of SM701.
  - When SM701 is OFF: Comment is 32 characters
  - When SM701 is ON: Comment is the upper 16 characters
- The number of points used by the output module is 10 points from the Y address designated by (D).

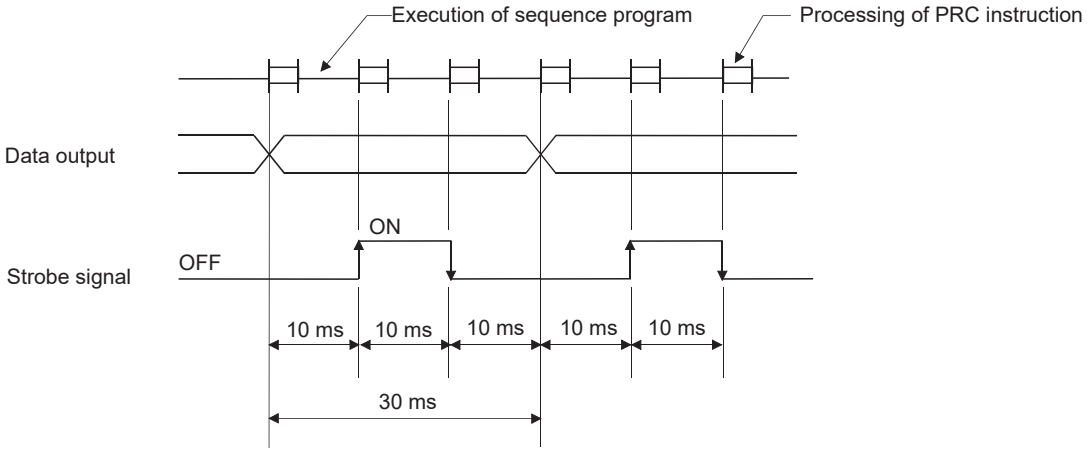


7

[Timing Chart]



- Output signals from the output module are transmitted at the rate of 30ms per character. For this reason, the time required to the completion of the transmission of the designated number of characters will be 30ms × n (ms). At 10ms interrupt intervals, the PRC instruction executes data output, strobe signal ON, and strobe signal OFF. The other instructions are executed continuously during a period between the above processings.



- In addition to the ASCII code, the output module also outputs a strobe signal (10ms ON, 20ms OFF) from the (D)+8 device.
- Following the execution of the PRC instruction, the PRC instruction execution flag ((D)+9 device) remains ON until the completion of the transmission of the designated number of characters.
- The PRC instruction can be used multiple times, but it is preferable to establish an interlock with the PRC instruction execution flag ((D)+9 device) so that they will not be ON simultaneously.
- If no comments have been registered at the device designated by (S), processing will not be performed.
- When a comment is read, SM720 turns ON for one scan after the instruction is completed. SM721 turns ON during the execution of the instruction. The PRC instruction cannot be executed while SM721 is ON. If the attempt is made, no processing is performed.

Point

- For device comments used with the PRC instruction, use comment files stored in the standard ROM or memory card. Comment files stored in the program memory cannot be used.
- The comment file used by the PRC instruction is set at the "PLC File Setting" option in the PLC parameter dialog box. If no comment file has been set for use by the PLC file setting, it will not be possible to output device comments with the PRC instruction.
- Do not execute the PRC instruction during an interrupt program. Otherwise, malfunction may occur.
- If a device which is not set a comment is specified, an error occurs. However, when the comment range for the device is set in the "Device Comment Detail Setting" of the "Online Data Operation" window using a programming tool, the error does not occur even if the comment is not set within the setting range.

Operation error

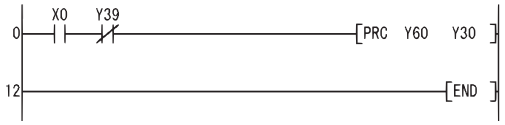
- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4100	The PRC instruction is executed while a comment is written during RUN.	—	○	○	—	—	—
	A device which is not set a comment is specified and the PRC instruction is executed.	—	○	○	—	—	—

Program example

- Program which outputs the comment of Y60 to Y30 to Y39 when X0 is turned ON.

[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	X0
1	ANI	Y39
2	PRC	Y60 Y30
5		<:y60="aa">
12	END	

Error display and annunciator reset

LEDR

Basic | High performance | Process | Redundant | Universal | LCPU

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant	Others
	Bit	Word		Bit	Word				
—	—								

Processing details

- Resets the self-diagnosis error display so that annunciator display or operation can be continued. With one execution of this instruction, either error display or annunciator is reset.

■Operation when self-diagnosis error is generated

- If the self-diagnosis error is one which allows continued operation

If there is an indication of the self-diagnosis error which the CPU module can continue the operation, reset the "ERR." LED in front of the CPU module.

It will be necessary to reset SM0, SM1, and SD0 at the user program, because they are not reset automatically.

Since the cause of the error displayed at this time has a higher priority over annunciator, no action for resetting the annunciator is taken.

- When a battery error is generated.

If the LEDR instruction is executed after the battery is changed, the "BAT." LED in front of the CPU module is reset.

SM51 is also turned OFF at this time.

■Operations when an annunciator (F) is ON.

When the LEDR instruction is executed, the following operations are performed.

- "USER" LED flickers, and is turned OFF
- The annunciators (F) stored in SD62 and SD64 are reset, and the F numbers for SD65 to SD79 are moved up.
- The data newly stored at SD64 is transmitted to SD62.
- The data at SD63 is decremented by 1. However, if SD63 is 0, it remains 0.

• The defaults for the error item numbers set in special registers SD207 to SD209 and order of priority are given in the table below:

Priority	Factor number (Hexadecimal)	Description	Remark
1	1	AC DOWN SINGLE PS.DOWN SINGLE PS.ERROR	Power supply cut Redundant base unit power supply voltage drop (QCPU only) Redundant power supply module fault (QCPU only)
2	2	UNIT VERIFY ERR. FUSE BREAK OFF SP. UNIT ERROR	I/O module verify error (QCPU only) Blown fuse (QCPU only) Special function module verify error (QCPU only) Intelligent function module verification error
		SP. UNIT DOWN	Intelligent function module error (LCPU only)
3	3	OPERATION ERROR LINK PARA.ERROR SFCP OPE. ERROR SFCP EXE. ERROR REMOTE PASS.FAIL SNTP OPE.ERROR	Operation Errors Link parameter error (QCPU only) SFC instruction operation error (QCPU only) SFC program execution error (QCPU only) Remote password error (LCPU only) SNTP error (LCPU only)
4	4	ICM.OPE ERROR FILE OPE. ERROR EXTEND INST. ERROR OPE. MODE DIFF. CAN'T EXE.MODE TRK.TRANS. ERR. TRK.SIZE ERROR TRK.DISCONNECT FLASH ROM ERROR	Memory card operation error File access error Extend instruction error (QCPU only) Operation status, switch mismatch (QCPU only) Current mode-time function execution disabled (QCPU only) Tracking data transmission error (QCPU only) Tracking capacity excess error (QCPU only) Tracking cable not connected, failure (QCPU only) Flash ROM access count exceeded error (LCPU only)
5	5	PRG.TIME OVER	Constant scan setting time over error Low speed execution monitoring time over error (QCPU only)
6	6	CHK instruction	—
7	7	Annunciator	—
8	8	LED instruction	—
9	9	BATTERY ERR.	—
10	A	Clock data	—
11	B	CAN'T SWITCH STANDBY SYS.DOWN MEM.COPY EXE.	System switching error (QCPU only) Standby system not started/stop error (QCPU only) Memory copy function executed (QCPU only)
12	C	DISPLAY ERROR	Display unit error (LCPU only)

• If the highest priority is given to the annunciator, it can be reset with priority by the LEDR instruction. (Basic Model QCPU, High Performance model QCPU, Process CPU, and Redundant CPU)

Operation error

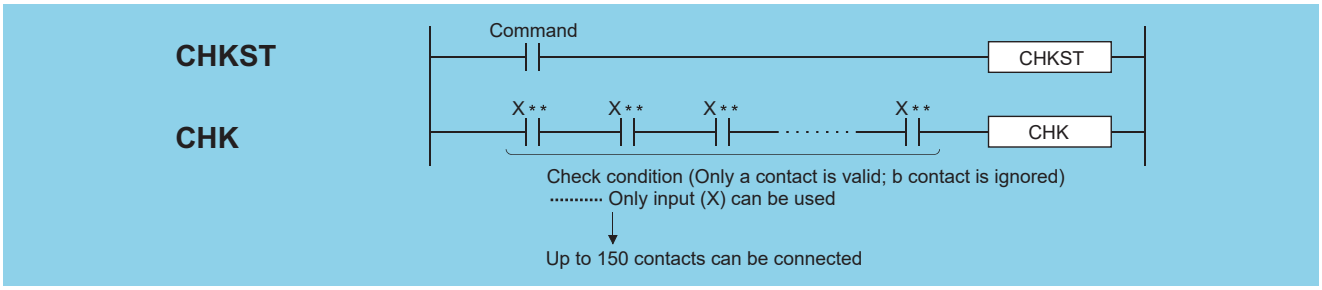
• There is no operation error using the LEDR instruction.

7.10 Debugging and Failure Diagnosis Instructions

Special format failure check

CHKST, CHK

Basic | High performance | Process | Redundant | Universal | LCPU



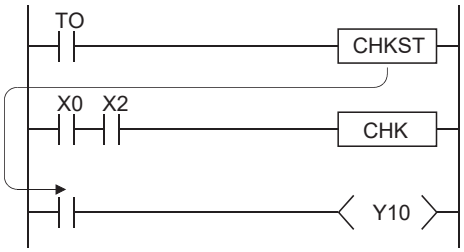
Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant	Others
	Bit	Word		Bit	Word				
—	—								

Processing details

■CHKST

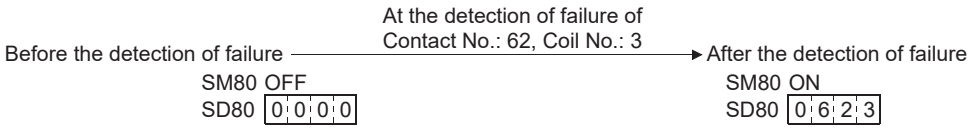
- The CHKST instruction is the instruction that starts the CHK instruction.
- If the command for the CHKST instruction is OFF, execution jumps from the CHK instruction to the next instruction.
- If the command for the CHKST instruction is ON, the CHK instruction is executed.

When T0 is OFF, program jumps to the instruction next to the CHK instruction.

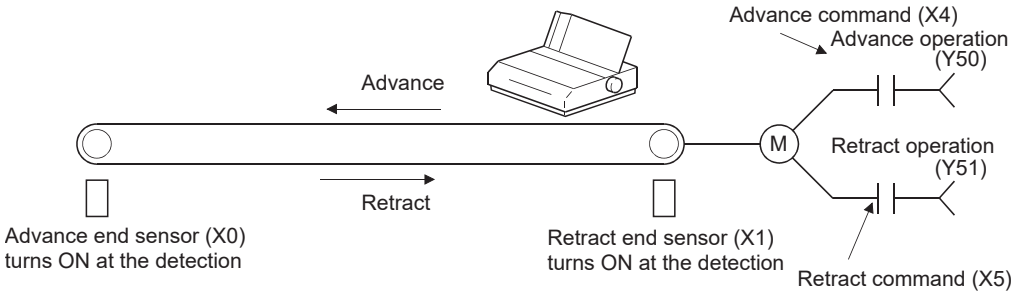


■CHK

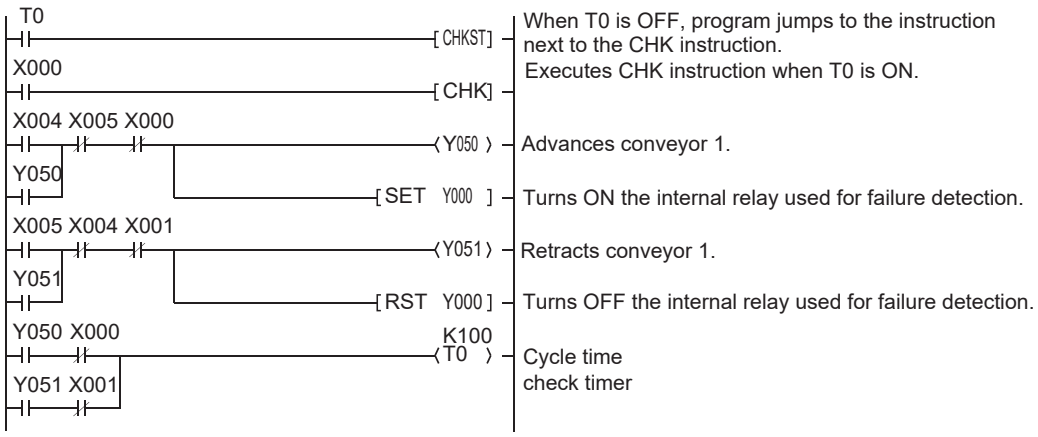
- The CHK instruction is the instruction used for the bidirectional operation to confirm the nature of the system failure. When the CHK instruction is executed, a failure diagnosis check is conducted with the designated check conditions, and if a failure is detected, SM80 is turned ON, and the failure number is stored at SD80 as a BCD value.
- The error code "9010" will be returned if a failure is detected. The contact number where the failure was discovered is stored at the upper 3 digits of SD80, and the coil number where the failure was detected is stored at the lower 1 digit of SD80. (☞ Point)



- The contact instruction prior to the CHK instruction does not control the execution of the CHK instruction, but rather sets the check conditions.



- A ladder such as the one shown below can be created to perform a cycle time over check for the system shown above:



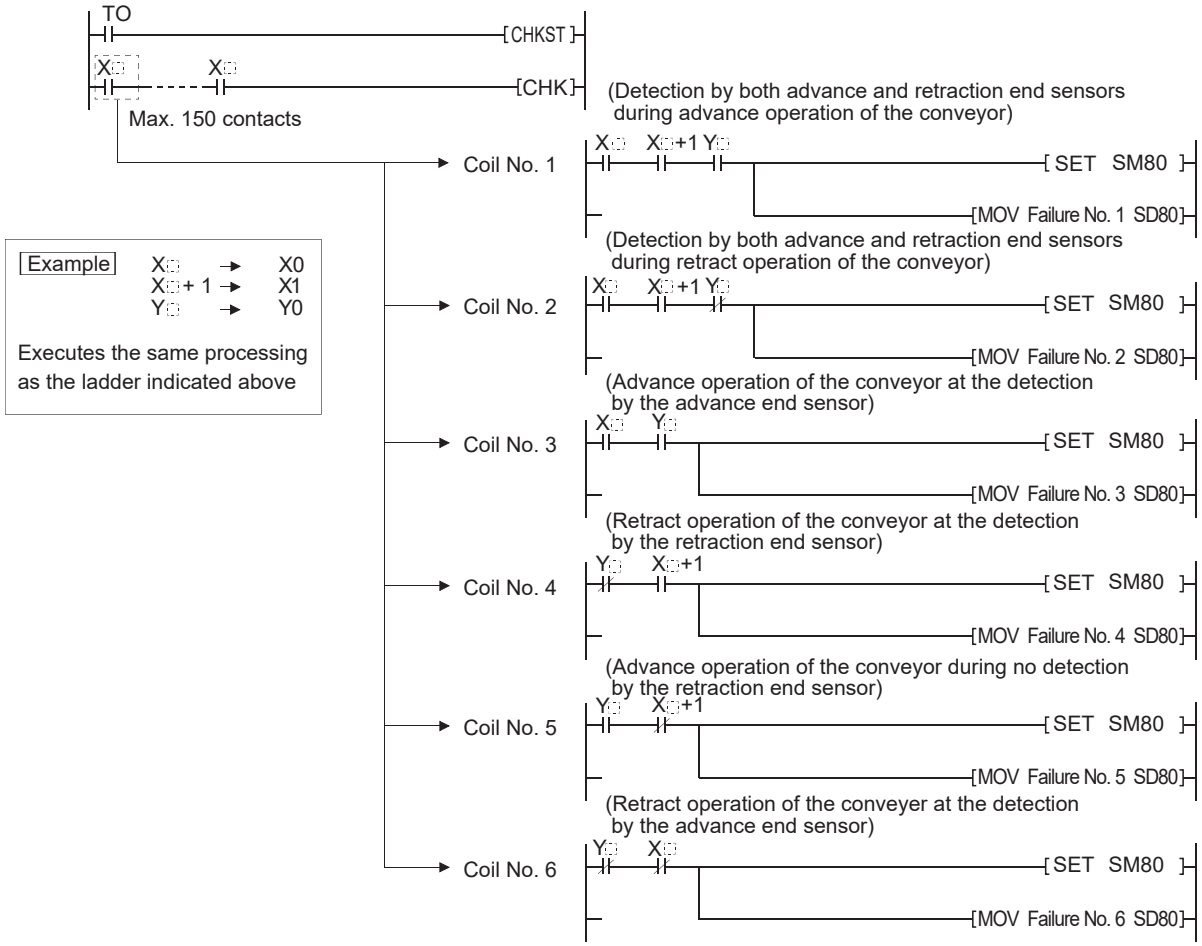
- The following points should be taken into consideration when creating a ladder for use with the CHK instruction:
  - The contact numbers for the advance edge detection sensor and the retract edge detection sensor (X□) must always be continuous. Further, the contact number (X□) for the advance edge detection sensor should be lower than that for the retract edge.
  - Controls for the advance edge detection sensor contact number (X□) and output with the identical number (Y□)*1 are as follows:

When advance operation is in progress: turn ON
When retract operation is in progress: turn OFF

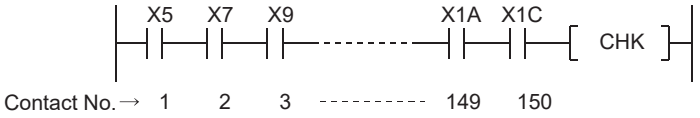
*1 Output (Y□) is treated as an internal relay, and cannot be output to an external device.

Point

- Depending on the designated contact, the CHK instruction undergoes processing identical to that shown for the ladder below:

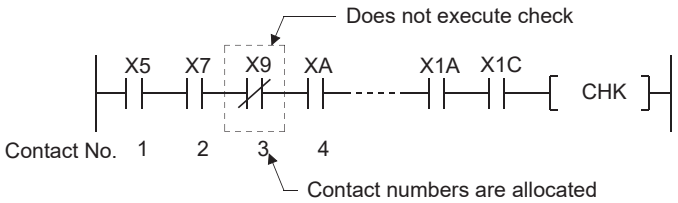


- Numbers 1 to 150 from the vertical bus on the left side have been allocated as contact numbers during failure detection.



- Reset SM80 and SD80 prior to forcing the execution of the CHK instruction. After the execution of the CHK instruction, it cannot be performed once again until SM80 and SD80 have been reset. (The contents of SM80 and SD80 will be preserved until reset by user.)
- A CHKST instruction must be placed before the CHK instruction. An error will be returned if an instruction other than the LD, LDI, AND or ANI instruction is used between the CHK instruction and the CHKST instruction. (Error code: 4235)
- The CHK instruction can be written at any step of the program. However, there is a limit in the number of uses of the CHK instruction. An error will be returned if the CHK instruction is used exceeding the number of uses specified below. (Error code: 4235)
  - Can be used up to two places in all program files being executed.
  - Can be used only one place in a single program file.

- Place LD and AND instructions prior to the CHK instruction to establish a check condition. Check conditions cannot be set using other contact instructions. If a check condition has been set with LDI or ANI, the processing for the check condition they specify will not be conducted. However, contact numbers during failure detection can also be allocated to the LDI and ANI instructions.



- The failure detection method differs according to whether SM710 is ON or OFF.
  - If SM710 is OFF, checks will be conducted of coil numbers 1 to 6 for each contact successively. When the CHK instruction is executed, checks will be in order from coil No. 1 of contact No. 1, through coil No. 6, then move on to contact No. 2 and check the coils in order from No. 1. The CHK instruction will be completed when coil No. 6 from contact No. n has been checked.
  - If SM710 is ON, checks will be conducted of contact numbers 1 through n, in coil number order. When the CHK instruction is executed, checks will begin with the ladder for coil No. 1, in order from contact No. 1 until contact No. n, then move on to the coil No. 2 ladder and begin from contact No. 1. The CHK instruction will be completed when a check has been made through contact No. n of coil No. 6.
- If more than one failure is detected, the number of the first failure detected will be stored. Failure numbers detected after this will be ignored.
- The CHK instruction cannot be used by a low speed execution type program. If a low speed execution type program has been set in a program file containing the CHK instruction, an operation error will be returned, and the CPU module operation will be suspended.

Operation error

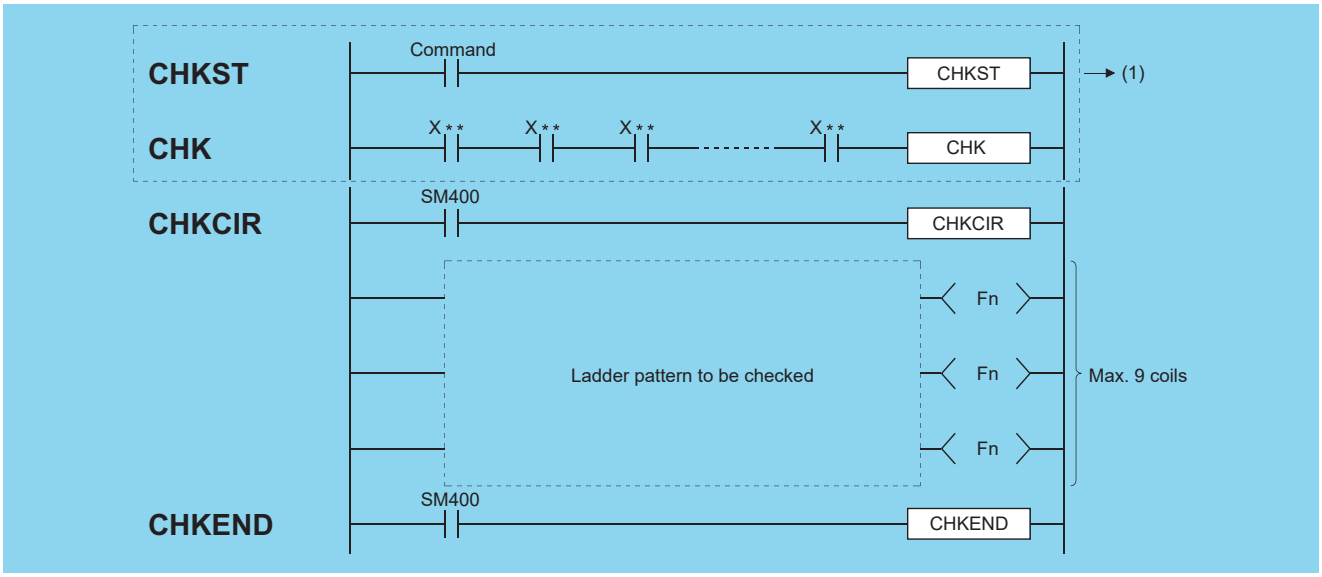
- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4235	There is a parallel ladder. There is an NOP instruction. There are more than 150 contact instructions. A CHK instruction is not executed after the CHKST instruction. The CHK instruction is executed when no CHKST instruction has been executed. The CHKST and CHK instruction are used in a low speed execution type program. There is an instruction other than the LD, LDI, AND or ANI instruction between the CHK instruction and the CHKST instruction. The CHK instruction is used on three or more points in all of the program files being executed. The CHK instruction is used on two more points in a single program file.	—	○	○	○	—	—

Changing check format of CHK

CHKCIR, CHKEND

Basic | High performance | Process | Redundant | Universal | LCPU



(1) ☞ Page 522 Special format failure check

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant	Others
	Bit	Word		Bit	Word				
—	—								

Processing details

- The check ladder pattern that will be used in the CHK instruction can be updated to any format desired. The actual failure checks are conducted with the CHKST and CHK instructions.
- Failure checks are conducted according to the check conditions designated by the CHK instruction and the ladder pattern described between the CHKCIR and CHKEND instructions.

Point

- Refer to Page 522 Special format failure check for more information on the CHKST and CHK instructions.
- To change the check format of the CHK instruction using the CHKCIR to CHKEND instructions, the user should create a ladder with index modification (Z0).

• The device numbers indicated at check conditions (X2 and X8 in the figure below) will assume index modification values for the individual device numbers (with the exception of annunciators (F)) described in the ladder patterns.

Ex.

X10 within the dotted lines in the figure below would be as follows:

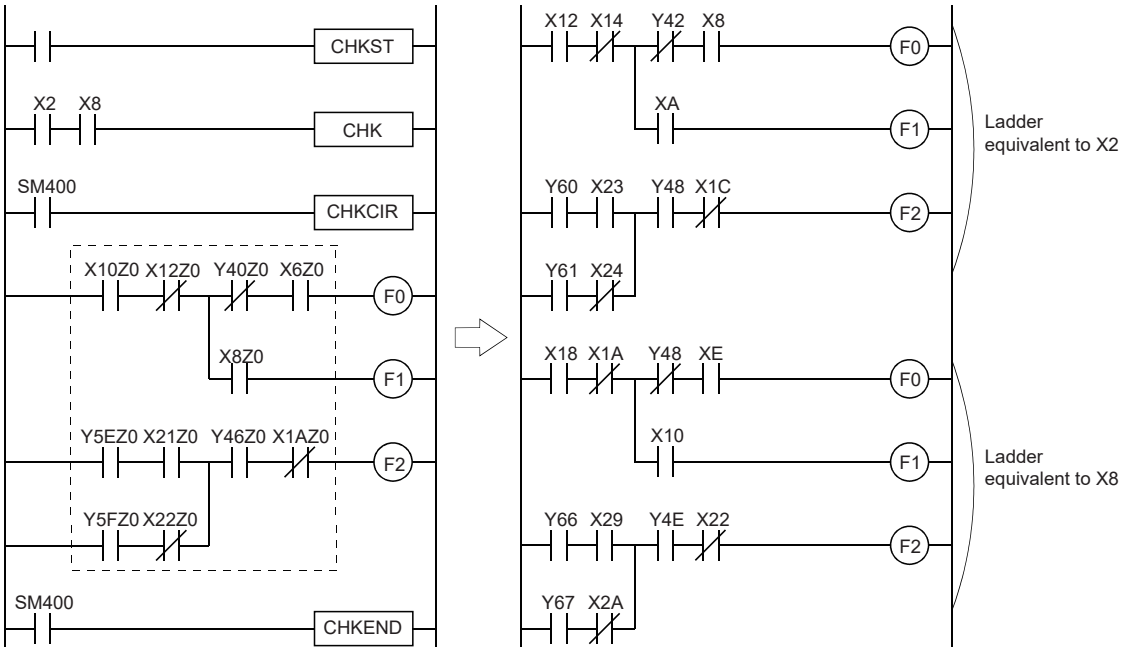
When the check condition corresponds to X2, the processing is performed by X12.

When the check condition corresponds to X8, the processing is performed by X18.

However, the order in which failure detection is executed differs depending on whether SM710 is ON or OFF.

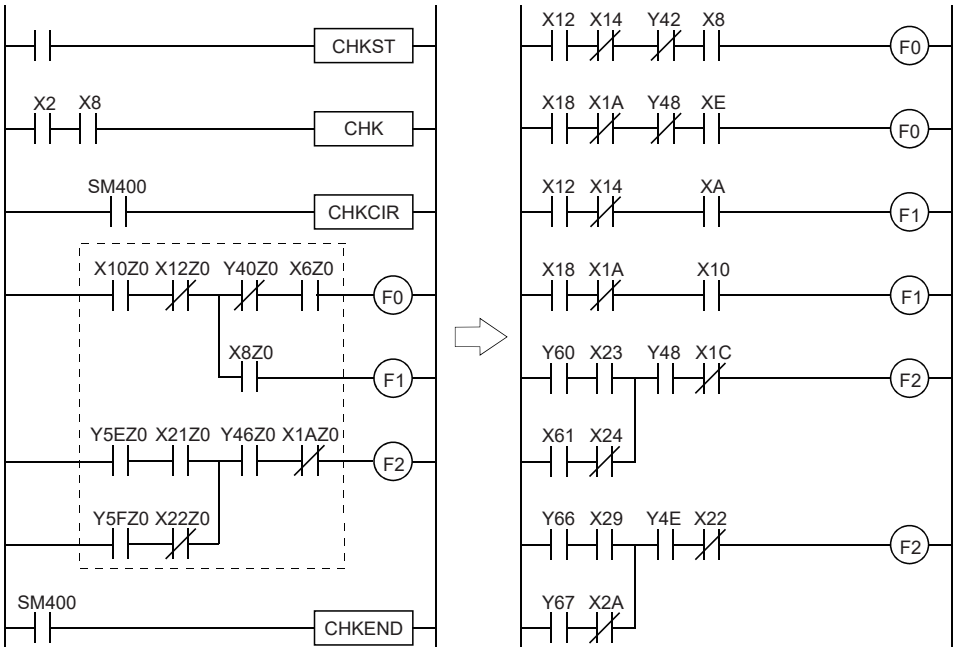
• If SM710 is OFF, checks will be conducted of coil numbers 1 through the end for each contact successively.

[Ladder specified by CHKCIR to CHKEND] [Order of check by CPU module]



• If SM710 is ON, checks will be conducted of contact numbers 1 through the end, in coil number order.

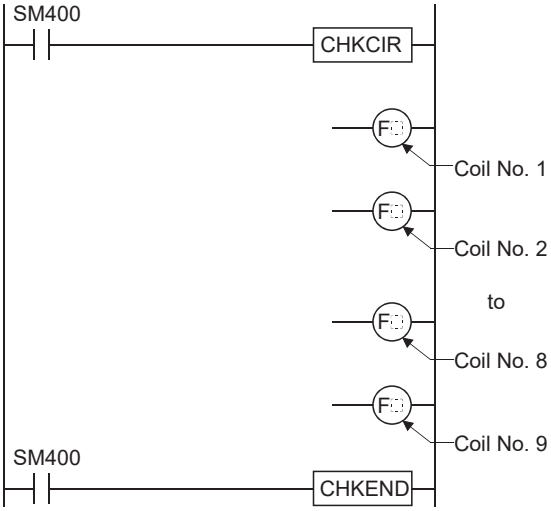
[Ladder specified by CHKCIR to CHKEND] [Order of check by CPU module]



- Failure checks check the ON/OFF status of OUT F☐ by using the ladder pattern in the various check conditions. In all check conditions, SM80 will be turned ON if even one of the OUT F☐ is ON in a ladder pattern. Further, the error numbers (contact numbers and coil numbers) corresponding to the OUT F☐ which were found to be ON will be stored from SD80 in BCD order.
- The instructions that can be used in ladder patterns are as follows:

Type	Instruction
Contact	LD, LDI, AND, ANI, OR, ORI, ANB, ORB, MPS, MPP, MRD, and comparative operation instructions
Coil	OUT F☐

- The following devices can be used for ladder pattern contacts: Input (X) and Output (Y).
- Only annunciators (F) can be used in ladder pattern coils. However, since annunciators (F) are used as a dummy, any value can be set for an annunciator (F). Further, they can overlap with no difficulties.
- ON/OFF controls can be performed without error if an annunciator (F) used during the execution of the CHK instruction has the same number as an annunciator (F) used in some other context than the CHK instruction. They will be treated differently during the CHK instruction than they are in the different context.
- The annunciators (F) used in the CHK instruction do not actually turn ON/OFF. Even when they are monitored from an external device, the ON/OFF status cannot be checked.
- A ladder pattern can be created up to 256 steps. Further, OUT F☐ can use up to 9 coils.
- Coil numbers for ladders designated with the CHKCIR through CHKEND instructions are allocated coil numbers from 1 to 9, from top to bottom.



- The CHKCIR and CHKEND instructions can be written at any step in the program desired. It can be used in up to two locations in all program files being executed. However, the CHKCIR and CHKEND instructions cannot be used in more than 1 location in a single program file.
- The CHKCIR and CHKEND instructions cannot be used in low speed execution type programs. If a program file in which the CHKCIR or CHKEND instruction is described is set as a low speed execution type program, an operation error will occur, and the High Performance model QCPU/Process CPU/Redundant CPU operation will be suspended.

Operation error

• In any of the following cases, an operation error occurs, the error flag (SM0) turns on, and an error code is stored in SD0.

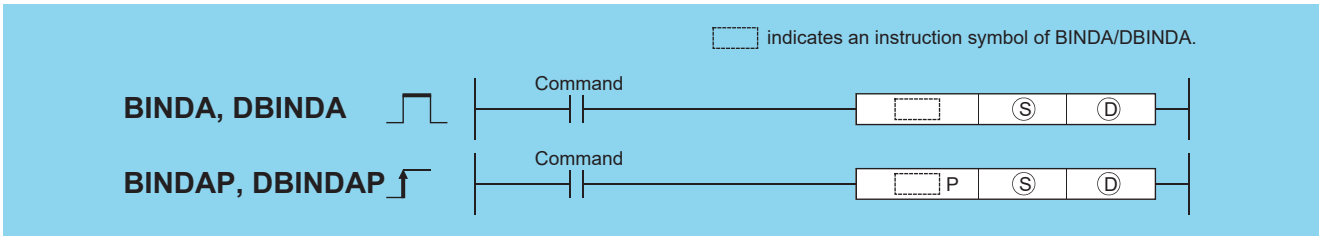
Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4230	The CHKEND instruction is not executed after the CHKCIR instruction. The CHKEND instruction is executed when no CHKCIR instruction has been executed.	—	○	○	○	—	—
4235	The CHKCIR or CHKEND instruction appears three or more times in all program files. The CHKCIR or CHKEND instruction appears two or more times in a single program file. The CHKST and CHK instruction are used in a low speed execution type program. There are 10 or more F devices in a ladder pattern. The ladder pattern has 257 or more steps. The device has been encountered which cannot be used in a ladder pattern. Index modification has been conducted on the ladder pattern device.	—	○	○	○	—	—

7.11 Character String Processing Instructions

Conversion from BIN 16-bit data to decimal ASCII, conversion from BIN 32-bit data to decimal ASCII

BINDA(P), DBINDA(P)

Basic | High performance | Process | Redundant | Universal | LCPU



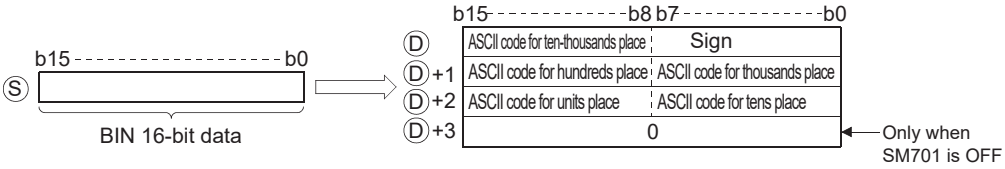
(S): BIN data to be converted to ASCII (BIN 16/32 bits)
(D): Head number of the devices where the conversion result will be stored (character string)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(S)	○	○		○					—
(D)	—	○		—					—

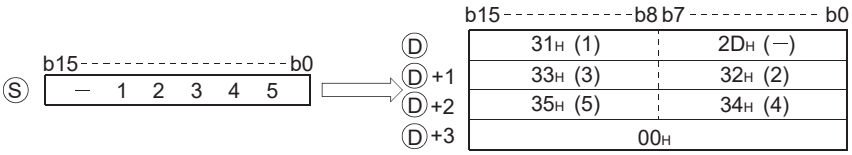
Processing details

■BINDA

- Converts the individual digit numbers of decimal notation of the BIN 16-bit data designated by (S) into ASCII codes, and stores the results into the area starting from the device designated by (D).



For example, if -12345 has been designated at (S), the following will be stored from (D) onward:



- The BIN data designated at (S) can be in the range from -32768 to 32767.
- The operation results stored at (D) are as follows:
  - The sign "20H" will be stored if the BIN data is positive, and the sign "2DH" will be stored if it is negative.
  - The sign "20H" will be stored for the leading zeros of effective digits. (Zero suppression is conducted.)

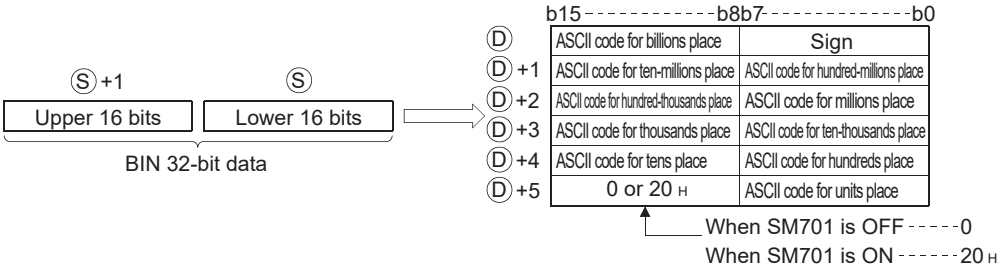
0 0 3 2 5
Number of significant digits
20H is set

- The storage of data at devices specified by (D)+3 differs depending on the ON/OFF status of SM701 (output number of characters conversion signal).

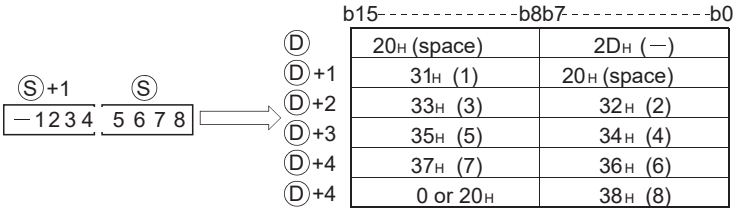
When SM701 is OFF: Stores "0"
When SM701 is ON: Does not change

■DBINDA

- Converts the individual digit numbers of decimal notation of the BIN 32-bit data designated by (S) into ASCII codes, and stores the results into the area starting from the device designated by (D).



For example, if the value -12345678 has been designated by (S), the following would be stored into the area starting from (D):



- BIN data designated by (S) can be between -2147483648 to 2147483647.
- The operations results stored at (D) will be stored in the following way:
  - The sign "20H" will be stored if the BIN data is positive, and the sign "2DH" will be stored if it is negative.
  - The sign "20H" will be stored for the leading zeros of effective digits. (Zero suppression is conducted.)

00120345 60

20H Number of significant digits

- The data stored at the upper 8 bits of the device designated by (D)+5 differs depending on the ON/OFF status of SM701 (number of characters to output select signal).

When SM701 is OFF: Stores "0"
When SM701 is ON: Stores "20H"

Operation error

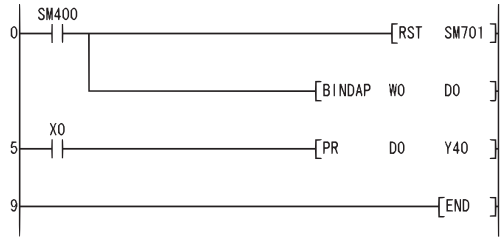
- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4101	The range of the device specified in (D) exceeds the range of the corresponding device.	—	—	—	—	○	○

Program example

- The following example program uses the PR instruction to output the 16-bit BIN data W0 value by decimal to Y40 to Y48 as ASCII.

[Ladder Mode]



[List Mode]

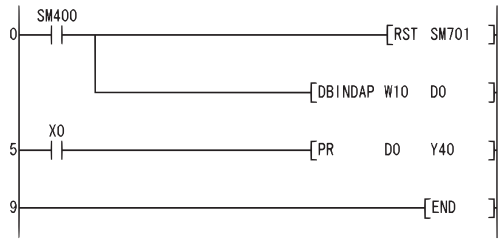
Step	Instruction	Device	
0	LD	SM400	
1	RST	SM701	
2	BINDAP	W0	D0
5	LD	X0	
6	PR	D0	Y40
9	END		

[Operation]

Conducts ASCII output of Y40 to Y48 by using the PR instruction when X0 goes ON.
Because SM701 is OFF, the PR instruction will output ASCII code until 00H is encountered.

- The following program uses the PR instruction to output the decimal value of the 32-bit BIN data at W10 and W11 in ASCII code to Y40 to Y48.

[Ladder Mode]

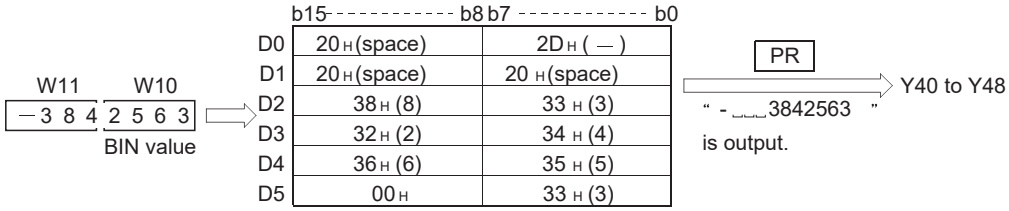


[List Mode]

Step	Instruction	Device	
0	LD	SM400	
1	RST	SM701	
2	DBINDAP	W10	D0
5	LD	X0	
6	PR	D0	Y40
9	END		

[Operation]

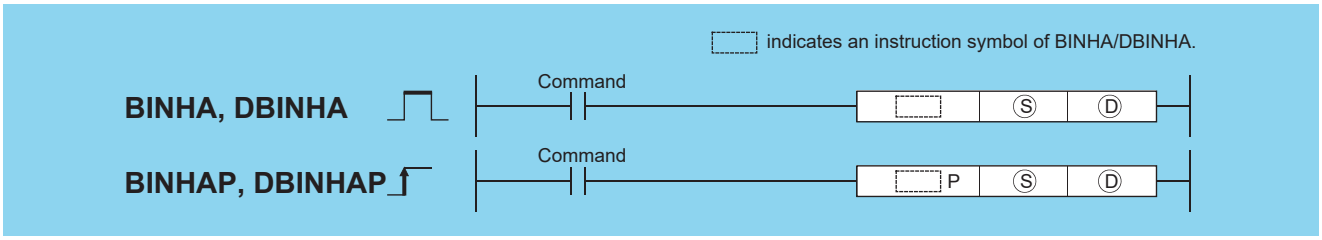
Conducts ASCII output of Y40 to Y48 by using the PR instruction when X0 goes ON.
Because SM701 is OFF, the PR instruction will output ASCII code until 00H is encountered.



Conversion from BIN 16-bit data to hexadecimal ASCII, conversion from BIN 32-bit data to hexadecimal ASCII

BINHA(P), DBINHA(P)

Basic | High performance | Process | Redundant | Universal | LCPU



(S): BIN data to be converted to ASCII (BIN 16/32 bits)
(D): Head number of the devices where the conversion result will be stored (character string)

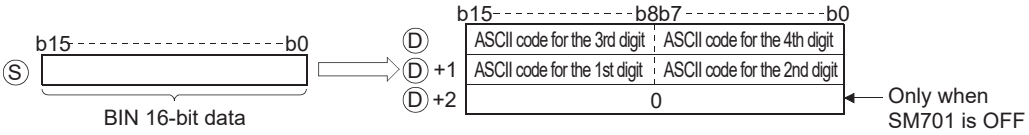
Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(S)	○	○		○					—
(D)	—	○		—					—

7

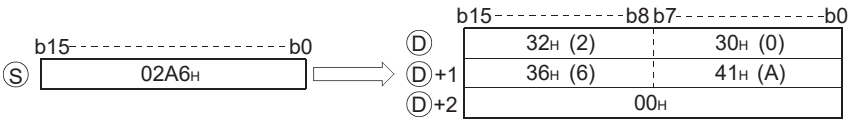
Processing details

■BINHA

- Converts the individual digit numbers of hexadecimal notation of the BIN 16-bit data designated by (S) into ASCII codes, and stores the results into the area starting from the device designated by (D).



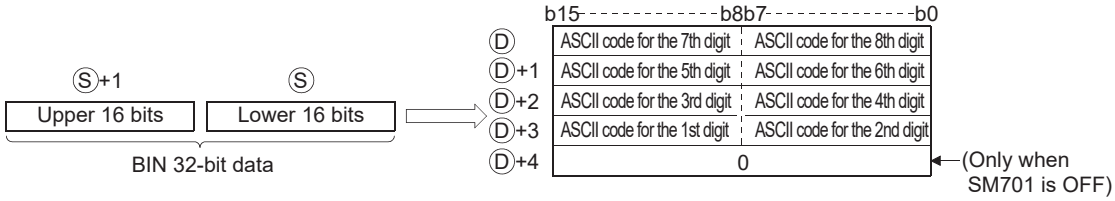
For example, if the value 02A6H is designated by (S), the operation result would be stored following (D) in the following manner:



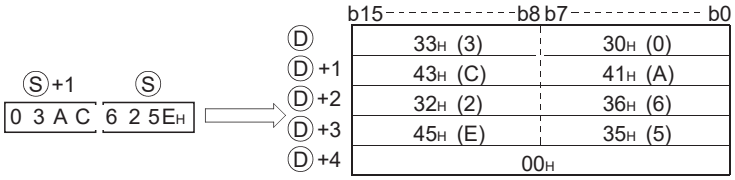
- The BIN data designated by (S) can be in the range from 0H to FFFFH.
- The operation results stored at (D) are processed as 4-digit hexadecimal values. For this reason, zeros which are significant digits on the left side of the value are processed as "0". (No zero suppression is conducted.)
- The data to be stored at the device designated by (D)+2 differs depending on the ON/OFF status of SM701 (number of characters to output select signal).
  - When SM701 is OFF: Stores "0"
  - When SM701 is ON: Does not change

■DBINHA

- Converts the individual digit numbers of hexadecimal notation of the BIN 32-bit data designated by (S) into ASCII codes, and stores the results into the area starting from the device designated by (D).



For example, if the value 03AC625EH has been designated by (S), it would be stored following (D) in the following manner:



- The BIN data designated by (S) can be in the range from 0H to FFFFFFFFH.
- The operation results stored at (D) are processed as 8-digit hexadecimal values. For this reason, zeros which are significant digits on the left side of the value are processed as "0". (No zero suppression is conducted.)
- The data to be stored at the device designated by (D)+2 differs depending on the ON/OFF status of SM701 (number of characters to output select signal).
  - When SM701 is OFF: Stores "0"
  - When SM701 is ON: Does not change

Operation error

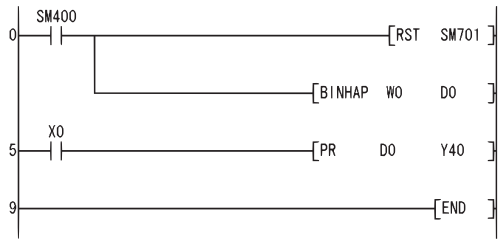
- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4101	The range of the device specified in (D) exceeds the range of the corresponding device.	—	—	—	—	○	○

Program example

- The following program uses the PR instruction to output the hexadecimal value of the 16-bit BIN data at W0 in ASCII code to Y40 to Y48.

[Ladder Mode]

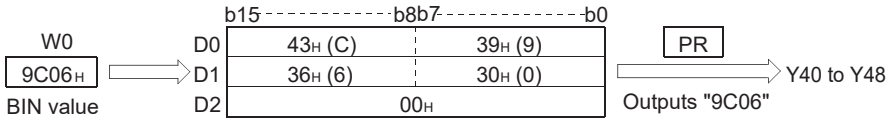


[List Mode]

Step	Instruction	Device	
0	LD	SM400	
1	RST	SM701	
2	BINHAP	W0	D0
5	LD	X0	
6	PR	D0	Y40
9	END		

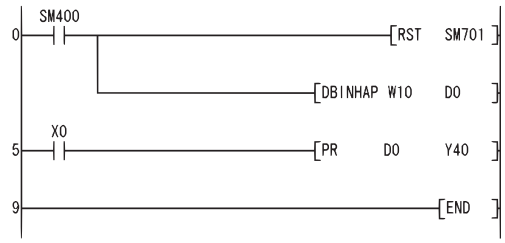
[Operation]

Conducts ASCII output of Y40 to Y48 by using the PR instruction when X0 goes ON.
Because SM701 is OFF, the PR instruction will output ASCII code until 00H is encountered.



- The following program uses the PR instruction to output the hexadecimal value of the 32-bit BIN data at W10 and W11 to Y40 to Y48.

[Ladder Mode]

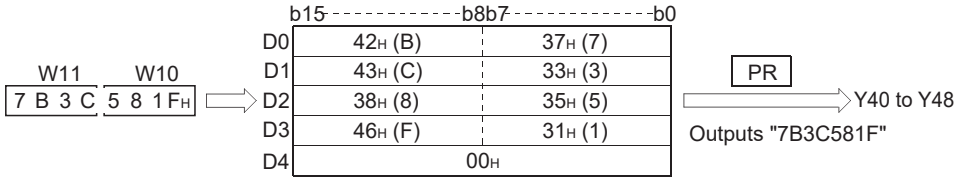


[List Mode]

Step	Instruction	Device	
0	LD	SM400	
1	RST	SM701	
2	DBINHAP	W10	D0
5	LD	X0	
6	PR	D0	Y40
9	END		

[Operation]

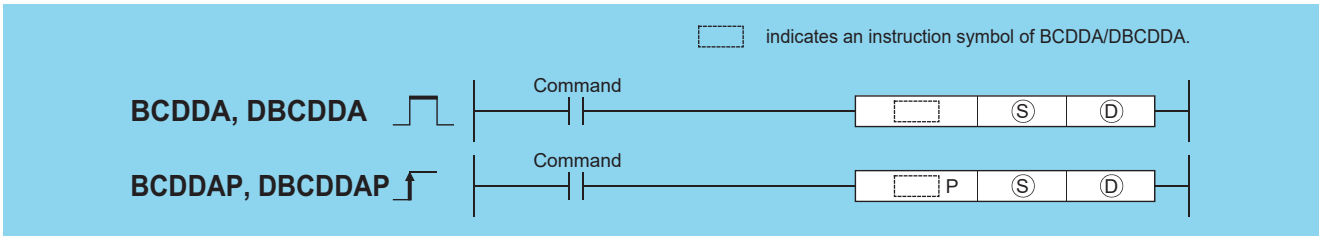
Conducts ASCII output of Y40 to Y48 by using the PR instruction when X0 goes ON.
Because SM701 is OFF, the PR instruction will output ASCII code until 00H is encountered.



Conversion from BCD 4-digit data to decimal ASCII data, conversion from BCD 8-digit data to decimal ASCII data

BCDDA(P), DBCDDA(P)

Basic | High performance | Process | Redundant | Universal | LCPU



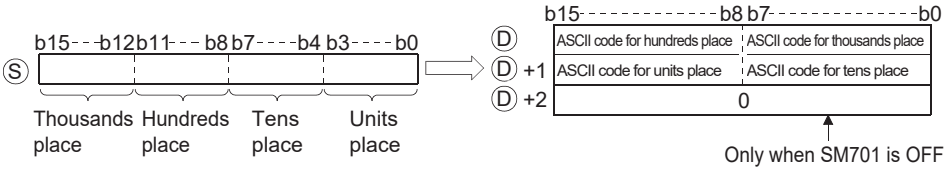
(S): BCD data to be converted to ASCII (BCD 4 digits/8 digits)
(D): Head number of the devices where the conversion result will be stored (character string)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(S)	○	○		○					—
(D)	—	○		—					—

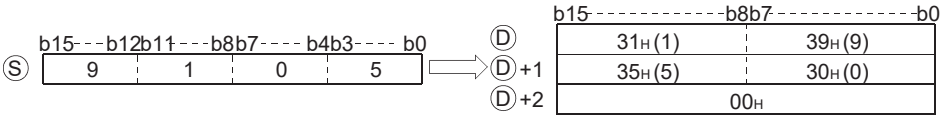
Processing details

■BCDDA

- Converts the individual digit numbers of hexadecimal notation of the BCD 4-digit data designated by (S) into ASCII codes, and stores the results into the area starting from the device designated by (D).



For example, when "9105" is designated for (S), the results of the operation are stored into the area starting from (D) in the following manner:

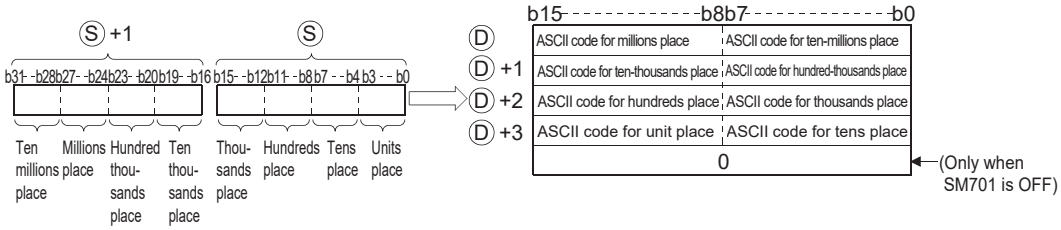


- The BCD data designated by (S) can be in the range of from 0 to 9999.
- The results of calculation stored in the device (D). All zeros on the left side of the "Number of significant digits" are zero-suppressed.

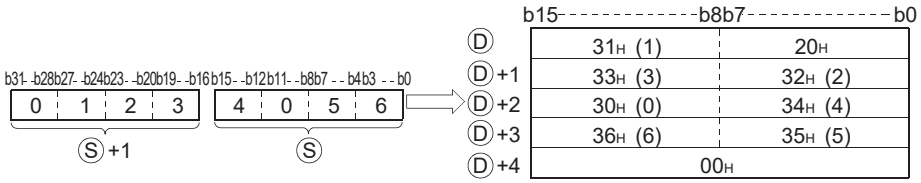
- The data to be stored at the device designated by (D)+2 differs depending on the ON/OFF status of SM701 (number of characters to output select signal).
  - When SM701 is OFF: Stores "0"
  - When SM701 is ON: Does not change

■DBCDDA

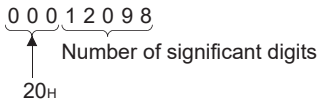
- Converts the individual digit numbers of hexadecimal notation of the BCD 8-digit data designated by (S) into ASCII codes, and stores the results into the area starting from the device designated by (D).



For example, if the value 01234056 is designated by (S), the operation result would be stored following (D) in the following manner:



- The BCD data designated by (S) can be in the range of from 0 to 99999999.
- The results of calculation stored in the device (D). All zeros on the left side of the "Number of significant digits" are zero-suppressed.



- The data to be stored at the device designated by (D)+4 differs depending on the ON/OFF status of SM701 (number of characters to output select signal).
  - When SM701 is OFF: Stores "0"
  - When SM701 is ON: Does not change

Operation error

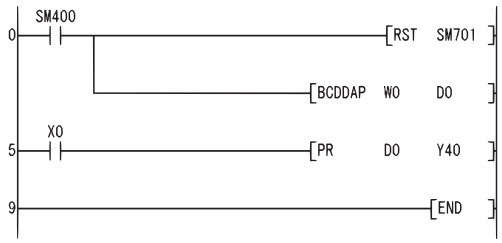
- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4100	During the operation of the BCDDA instruction, the data of (S) is other than 0 to 9999. During the operation of the DBCDDA instruction, the data of (S) is other than 0 to 99999999.	—	○	○	○	○	○
4101	The range of the device specified in (D) exceeds the range of the corresponding device.	—	—	—	—	○	○

Program example

- The following program uses the PR instruction to convert BCD 4-digit data (the value at W0) to decimal, and outputs it in ASCII format to Y40 to Y48.

[Ladder Mode]

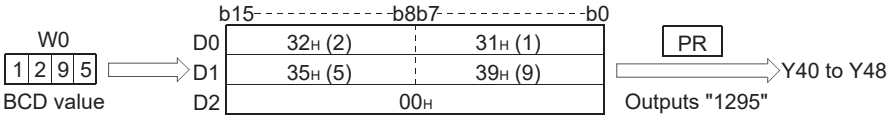


[List Mode]

Step	Instruction	Device	
0	LD	SM400	
1	RST	SM701	
2	BCDDAP	W0	D0
5	LD	X0	
6	PR	D0	Y40
9	END		

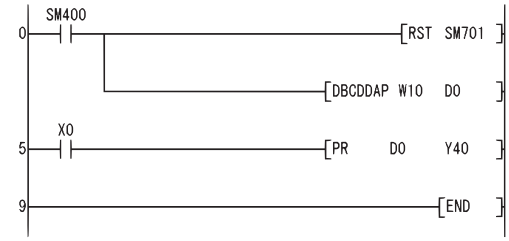
[Operation]

Conducts ASCII output of Y40 to Y48 by using the PR instruction when X0 goes ON.
Because SM701 is OFF, the PR instruction will output ASCII code until 00H is encountered.



- The following program uses the PR instruction to convert BCD 8-digit data (the values at W10 and W11) to decimal, and outputs it in ASCII format to Y40 to 48.

[Ladder Mode]

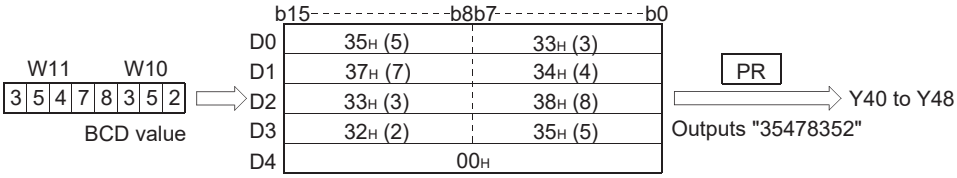


[List Mode]

Step	Instruction	Device	
0	LD	SM400	
1	RST	SM701	
2	DBCDDAP	W10	D0
5	LD	X0	
6	PR	D0	Y40
9	END		

[Operation]

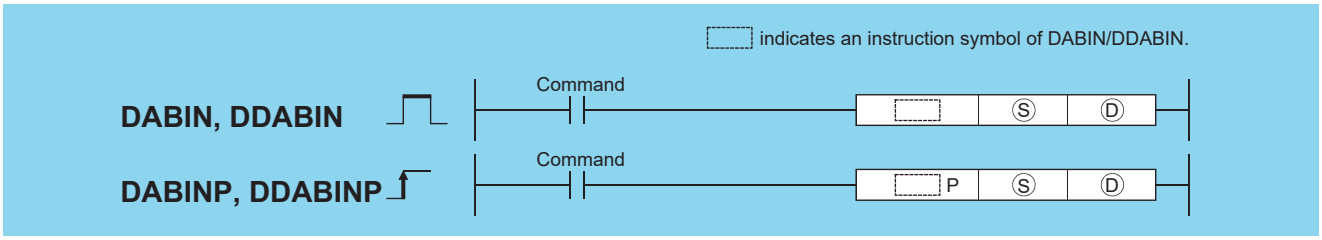
Conducts ASCII output of Y40 to Y48 by using the PR instruction when X0 goes ON.
Because SM701 is OFF, the PR instruction will output ASCII code until 00H is encountered.



Conversion from decimal ASCII to BIN 16-bit data, conversion from decimal ASCII to BIN 32-bit data

DABIN(P), DDABIN(P)

Basic | High performance | Process | Redundant | Universal | LCPU



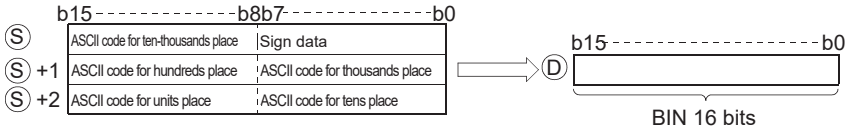
(S): ASCII data to be converted to BIN value or head number of the devices where the ASCII data is stored (character string)
(D): Head number of the devices where the conversion result will be stored (BIN 16/32 bits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant $	Others
	Bit	Word		Bit	Word				
(S)	—	○		—				○	—
(D)	○	○		○				—	—

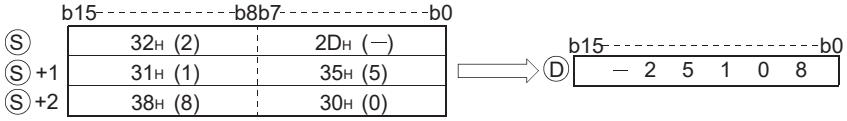
Processing details

■DABIN

- Converts decimal ASCII data stored into the area starting from the device number designated by (S) into BIN 16-bit data, and stores it in the device number designated by (D).



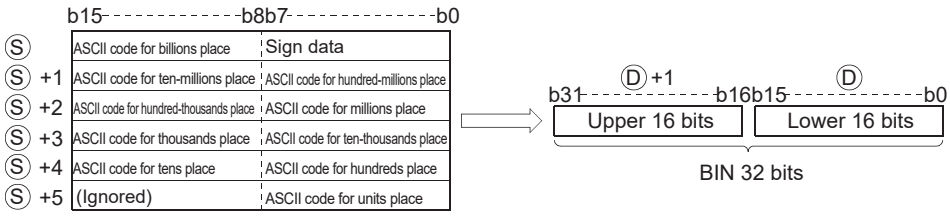
For example, if the ASCII code "-25108H" is specified for the area starting from (S), the conversion result is stored at (D) as shown below:



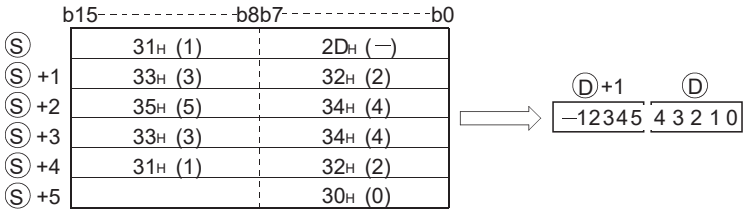
- The ASCII data designated by from (S) to (S)+2 can be in the range of from -32768 to 32767.
- The sign "20H" will be stored if the BIN data is positive, and the sign "2DH" will be stored if it is negative. (If other than "20H " and "2DH" is set, it will be processed as positive data.)
- ASCII code can be set for each position within the range from "30H" to "39H".
- If the ASCII code set for individual positions is "20H" or "00H," it will be processed as "30H".

■DDABIN

- Converts decimal ASCII data stored into the area starting from the device number designated by (S) into BIN 32-bit data, and stores it in the device number designated by (D).



For example, if the ASCII code of -1234543210H is designated for the area starting from (S), the operation result would be stored at (D)+1 and (D) in the following manner:



- The ASCII data designated by (S) to (S)+5 can be in the range of from -2147483648 to 2147483647. Further, data stored at the upper bytes of (S)+5 will be ignored.
- The sign "20H" will be stored if the BIN data is positive, and the sign "2DH" will be stored if it is negative. (If other than "20H " and "2DH" is set, it will be processed as positive data.)
- ASCII code can be set for each position within the range from "30H" to "39H".
- If the ASCII code set for individual positions is "20H" or "00H," it will be processed as "30H".

Operation error

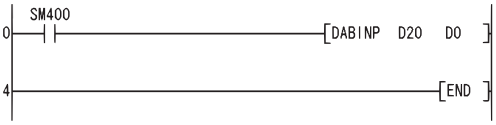
- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4100	The ASCII codes specified in (S) to (S) +5 other than "30H" to "39H", "20H", or "00H". The ASCII data specified in (S) to (S) +5 is outside the following ranges: • When the DABIN instruction is used: -32768 to 32767 • When the DDABIN instruction is used: -2147483648 to 2147483647	—	○	○	○	○	○
4101	The device specified in (S) exceeds the range of the corresponding device.	—	—	—	—	○	○

Program example

- The following program converts the decimal, 5-digit ASCII data and sign set at D20 through D22 to BIN values, and stores the result at D0.

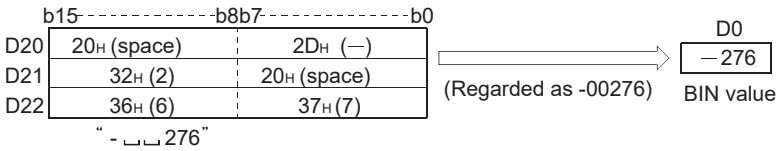
[Ladder Mode]



[List Mode]

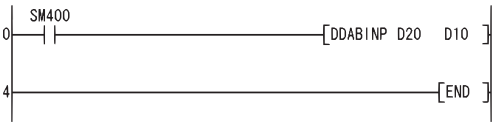
Step	Instruction	Device	
0	LD	SM400	
1	DABINP	D20	D0
4	END		

[Operation]



- The following program converts the decimal, 10-digit ASCII data and sign set at D20 through D25 to BIN values and stores the result at D10 and D11.

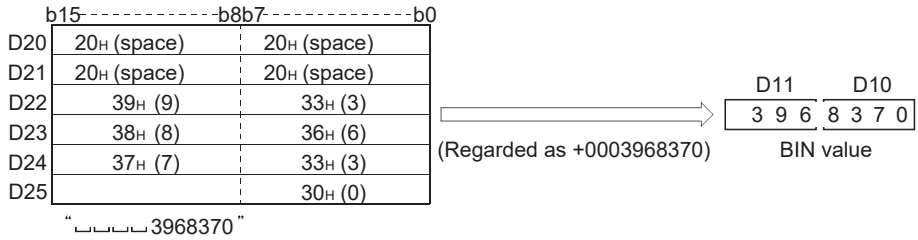
[Ladder Mode]



[List Mode]

Step	Instruction	Device	
0	LD	SM400	
1	DDABINP	D20	D10
4	END		

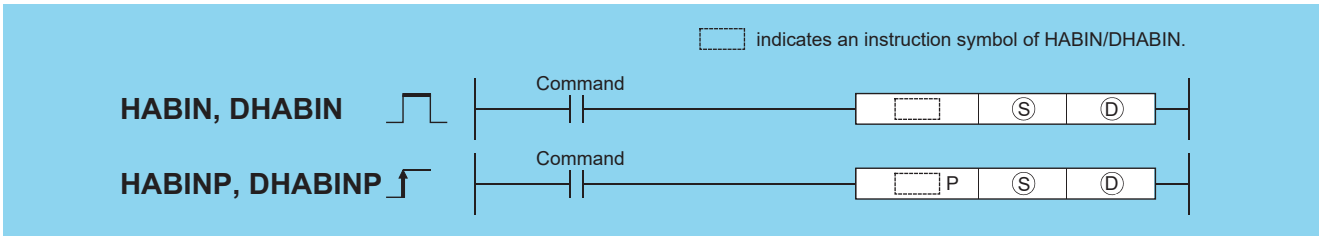
[Operation]



Conversion from hexadecimal ASCII to BIN 16-bit data, conversion from hexadecimal ASCII to BIN 32-bit data

HABIN(P), DHABIN(P)

Basic | High performance | Process | Redundant | Universal | LCPU



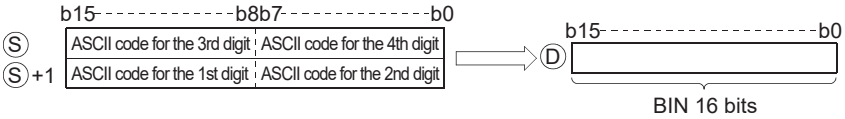
(S): ASCII data to be converted to BIN value or head number of the devices where the ASCII data is stored (character string)
(D): Head number of the devices where the conversion result will be stored (BIN 16/32 bits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant $	Others
	Bit	Word		Bit	Word				
(S)	—	○		—				○	—
(D)	○	○		○				—	—

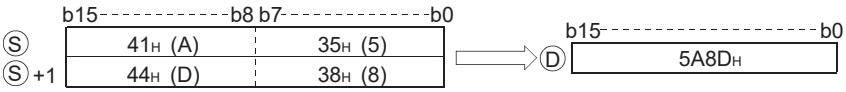
Processing details

■HABIN

- Converts hexadecimal ASCII data stored in the area starting from the device number designated by (S) into BIN 16-bit data, and stores it in the device number designated by (D).



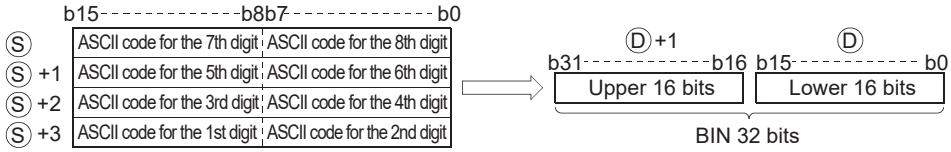
For example, if the ASCII code of 5A8DH is designated for the area starting from (S), the operation result would be stored at (D) in the following manner:



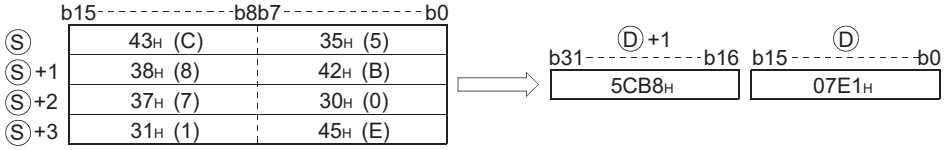
- The ASCII data designated by (S) to (S)+1 can be in the range of from 0000H to FFFFH.
- The ASCII codes can be in the range of "30H" to "39H" and from "41H" to "46H".

■DHABIN

- Converts hexadecimal ASCII data stored in the area starting from the device number designated by (S) into BIN 32-bit data, and stores it in the device number designated by (D).



For example, if the ASCII code of 5CB807E1H is designated for the area starting from (S), the operation result would be stored at (D)+1 and (D) in the following manner:



- The ASCII data designated by (S) to (S)+3 can be in the range of from 00000000H to FFFFFFFFH.
- The ASCII codes can be in the range of "30H" to "39H" and from "41H" to "46H".

Operation error

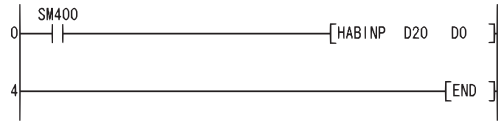
- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4100	The ASCII codes specified in (S) to (S) +3 are other than "30H" to "39H" and from "41H" to "46H".	—	○	○	○	○	○
4101	The range of the device specified in (S) exceeds the range of the corresponding device.	—	—	—	—	○	○

Program example

- The following program converts the hexadecimal, 4-digit ASCII data set at D20 and D21 to BIN data, and stores the result at D0.

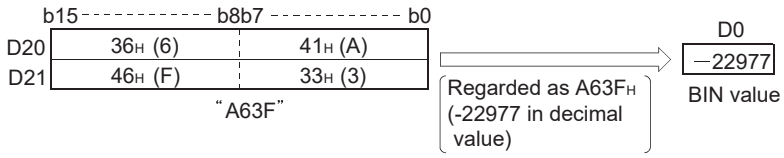
[Ladder Mode]



[List Mode]

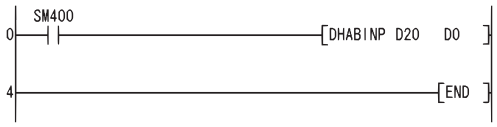
Step	Instruction	Device	
0	LD	SM400	
1	HABINP	D20	D0
4	END		

[Operation]



- The following program converts the hexadecimal, 8-digit ASCII data set at D20 to D23 to BIN values, and stores the result at D10 and D11.

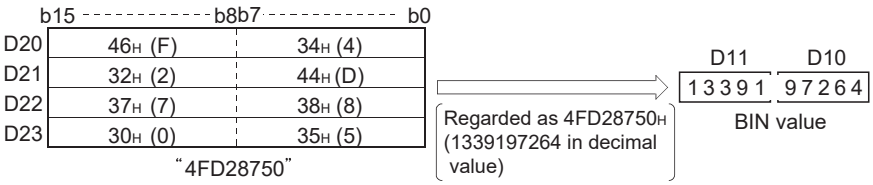
[Ladder Mode]



[List Mode]

Step	Instruction	Device	
0	LD	SM400	
1	DHABINP	D20	D0
4	END		

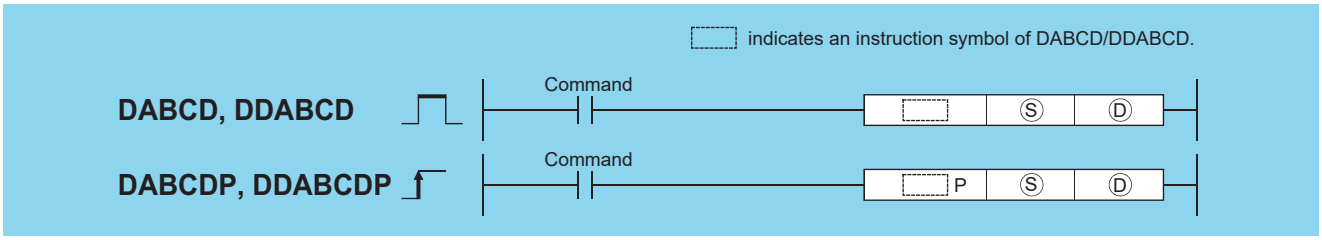
[Operation]



Conversion from decimal ASCII to BCD 4-digit data, conversion from decimal ASCII to BCD 8-digit data

DABCD(P), DDABCD(P)

Basic | High performance | Process | Redundant | Universal | LCPU



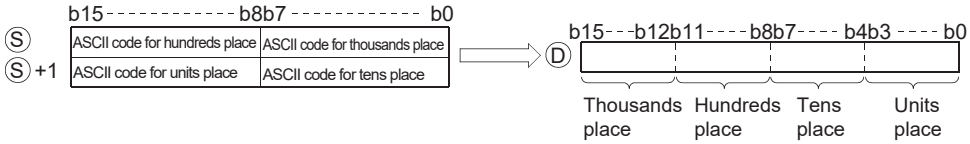
(S): ASCII data to be converted to BCD value or head number of the devices where the ASCII data is stored (character string)
(D): Head number of the devices where the conversion result will be stored (BCD 4 digits/8 digits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant $	Others
	Bit	Word		Bit	Word				
(S)	—	○		—				○	—
(D)	○	○		○				—	—

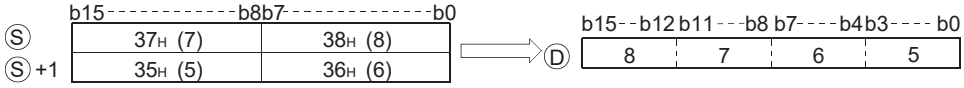
Processing details

■DABCD

- Converts decimal ASCII data stored in the area starting from device number designated by (S) into 4-digit BCD data, and stores at device number designated by (D).



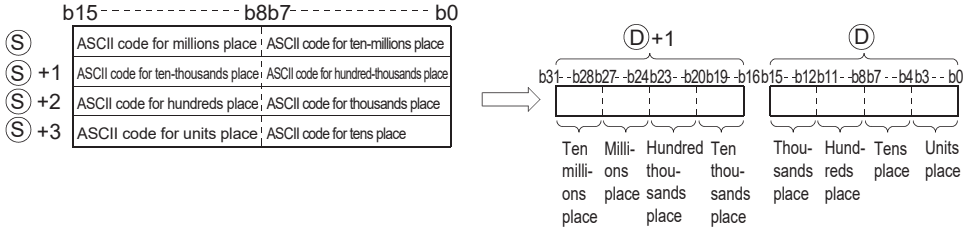
For example, if the ASCII code of 8765H is designated for the area starting from (S), the operation results would be stored at (D) in the following manner:



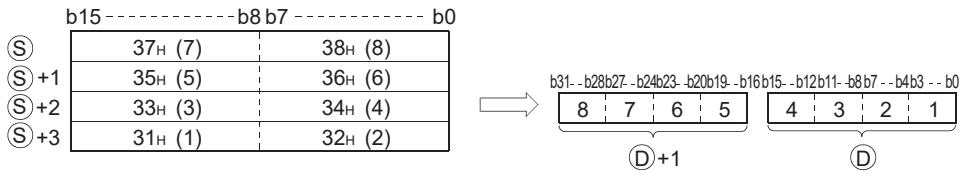
- The ASCII data designated by (S) to (S)+1 can be in the range of from 0 to 9999.
- The ASCII code set at each digit can be in the range of from "30H" to "39H".
- If ASCII code for individual digits is "20H" or "00H", it is processed as "30H".

■DDABCD

- Converts decimal ASCII data stored in the area starting from the device designated by (S) to 8-digit BCD data, and stores it into the area starting from the device designated by (D).



For example, if the ASCII code of 87654321H is designated for the area starting from (S), the operation results would be stored at (D)+1 and (D) in the following manner:



- The ASCII data designated by (S) to (S)+3 can be in the range of from 0 to 99999999.
- The ASCII code set at each digit can be in the range of from "30H" to "39H".
- If ASCII code for individual digits is from "20H" to "00H", it is processed as "30H".

Operation error

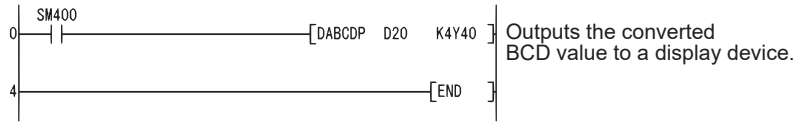
- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4100	A character other than 0 to 9 is put in the data of (S).	—	○	○	○	○	○
4101	The range of the device specified in (S) exceeds the range of the corresponding device.	—	—	—	—	○	○

Program example

- The following program converts the decimal ASCII data set from D20 to D22 to BCD 4-digit data, and outputs the results to Y40 to Y4F.

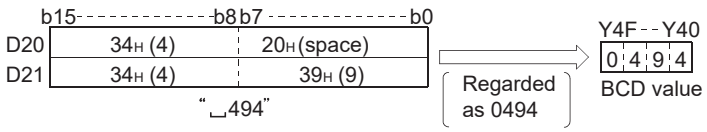
[Ladder Mode]



[List Mode]

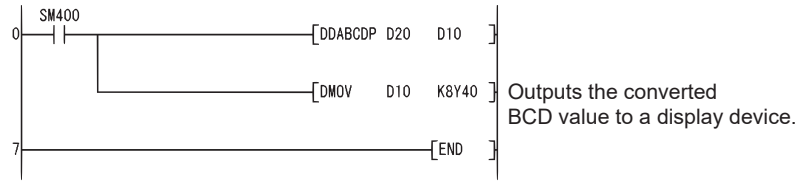
Step	Instruction	Device	
0	LD	SM400	
1	DABCDP	D20	K4Y40
4	END		

[Operation]



- The following program converts the decimal ASCII data set at D20 to D23 into 8-digit BCD data, stores the result at D10 and D11, and also outputs it to from Y40 to Y5F.

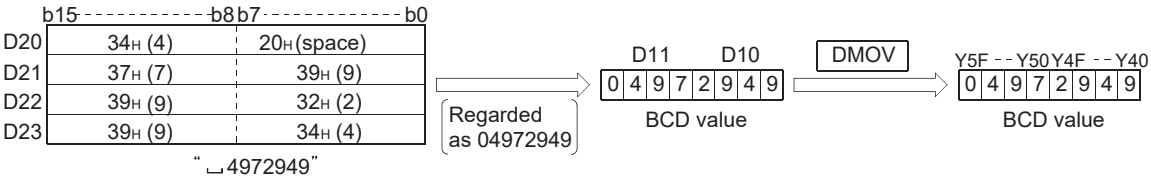
[Ladder Mode]



[List Mode]

Step	Instruction	Device	
0	LD	SM400	
1	DDABCDP	D20	D10
4	DMOV	D10	K8Y40
7	END		

[Operation]



Reading device comment data

COMRD(P)

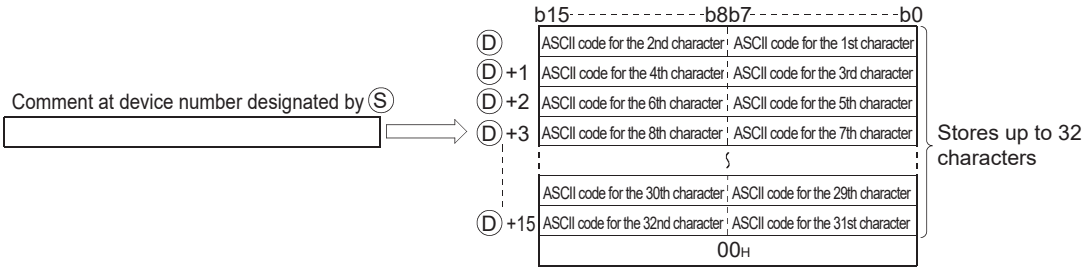
Basic | High performance | Process | Redundant | Universal | LCPU

(S): Head number of the devices where a comment to be read is stored (Device name)
(D): Head number of the devices where the read comment will be stored (character string)

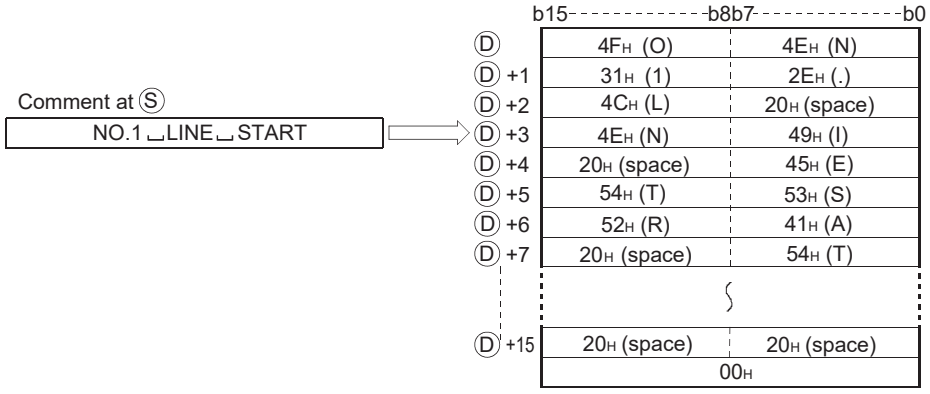
Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant	Others BL\S, BL, BL\TR, P, I, J, U
	Bit	Word		Bit	Word				
(S)	○	○		○			—		○
(D)	—	○		—			—		—

Processing details

- Reads the comment at the device number designated by (S), and stores it as ASCII code in the area starting from the device number designated by (D).



For example, if the comment for the device specified by (S) is "NO.1␣LINE␣START", the operation results would be stored in (D) and later devices as follows:



- If no comment has been registered for the device specified by (S) despite the fact that the comment range setting is made, all of the characters for the comment are processed as "20H" (space).
- The device number plus 1 where the final character of (D) is stored differs depending on the ON/OFF status of SM701 (number of characters to output select signal).
  - When SM701 is OFF: Does not change
  - When SM701 is ON: Stores "0"
- When a comment is read, SM720 turns ON for one scan after the instruction is completed. SM721 is turned ON during the execution of the instruction. While SM721 is ON, the COMRD(P) instruction cannot be executed. If the attempt is made, no processing is performed.

Point

- For device comments used with the COMRD(P) instruction, use comment files stored in the standard RAM, standard ROM, memory card, or SD memory card. Comment files stored in the program memory cannot be used.
- Set the comment file used for the COMRD(P) instruction in "PLC file setting" in the PLC parameter dialog box. If the comment file to be used is not set in the PLC file setting, device comments cannot be output with the COMRD(P) instruction. When a comment file is set in the "PLC File" tab of the PLC Parameter dialog box, but the file does not exist at power-on or reset, "FILESET ERROR" (error code: 2400) will occur.
- The COMRD(P) instruction cannot be executed during the interrupt program. No operation if executed.

Operation error

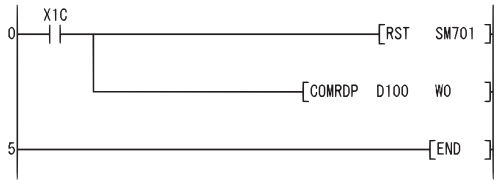
- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4100	The comment is not registered to the device number specified by (S).	—	○	○	○	○	○
4101	The device number specified by (D) is not a word device.	—	○	○	○	○	○
	The range of the device specified by (D) exceeds the range of the corresponding device.	—	—	—	—	○	○

Program example

- The following program stores the comments set at D100 into the area starting from W0 as ASCII when X1C is turned ON.

[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	X1C
1	RST	SM701
2	COMRDP	D100 W0
5	END	

[Operation]

Comment at D100

LINE␣A␣TARGET

	b15 ---- b8	b7 ---- b0
W0	49H (I)	4CH (L)
W1	45H (E)	4EH (N)
W2	41H (A)	20H (space)
W3	54H (T)	20H (space)
W4	52H (R)	41H (A)
W5	45H (E)	47H (G)
W6	20H (space)	54H (T)
W7	20H (space)	20H (space)
W15	20H (space)	20H (space)
W16	00H	

Precautions

- The processing completes after several scans.
- The COMRD(P)/PRC instruction is not executed if the start signal (execution command) of the COMRD(P)/PRC instruction is turned ON before completion of the instruction (while SM721 is ON). Execute the COMRD(P)/PRC instruction when SM721 is OFF.
- Two or more file comments cannot be accessed simultaneously.
- The following instructions cannot be executed simultaneously because they use SM721 in common.

Instruction Name	ON During Execution	ON for One Scan After Completion	ON after Abnormal Completion
SP.FREAD SP.FWRITE	SM721	Designated by instruction.	(Device designated by instruction) + 1
PRC COMRD		SM720	None

- For the High-speed Universal model QCPU, Universal model Process CPU and LCPU, when a comment file stored on an SD memory card is used, this instruction cannot be executed while SM606 (SD memory card forced disable instruction) is ON. If the instruction is executed, the command will be ignored.

Character string length detection

LEN(P)

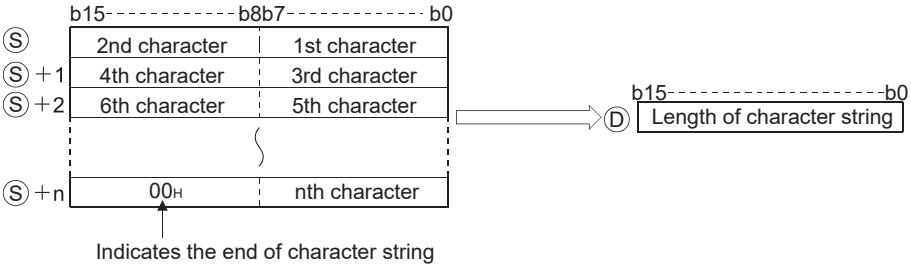
Basic | High performance | Process | Redundant | Universal | LCPU

(S): Character string or head number of the devices where the character string is stored (character string)
(D): Number of the device where the length of detected character string will be stored (BIN 16 bits)

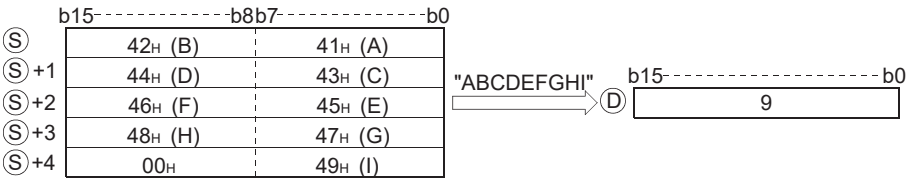
Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant $	Others
	Bit	Word		Bit	Word				
(S)	—	○		—				○	—
(D)	○	○		○				—	—

Processing details

- Detects length of character string designated by (S) and stores in the area starting from the device number designated by (D). Processes the data from the device number designated by (S) to the device number storing "00H" as a character string.



For example, when the value "ABCDEFGHI" is stored in the area starting from (S), the value 9 is stored at (D).



Operation error

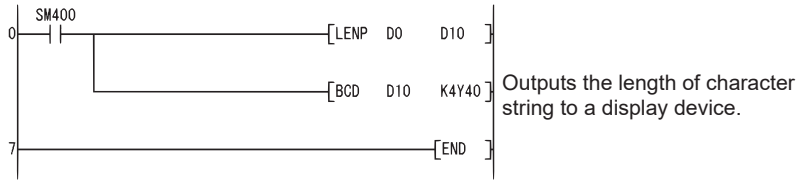
- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4101	There is no NULL code "00H" within the range of the corresponding devices starting from the device number specified by (S).	—	○	○	○	○	○

Program example

• The following program outputs the length of the character string from D0 to Y40 to Y4F as BCD 4-digit values.

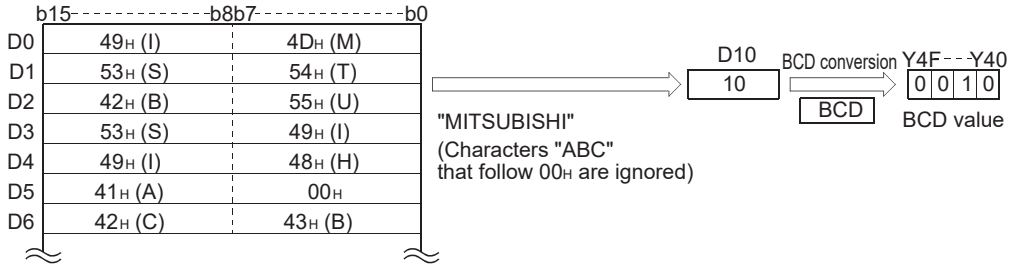
[Ladder Mode]



[List Mode]

Step	Instruction	Device	
0	LD	SM400	
1	LENP	D0	D10
4	BCD	D10	K4Y40
7	END		

[Operation]

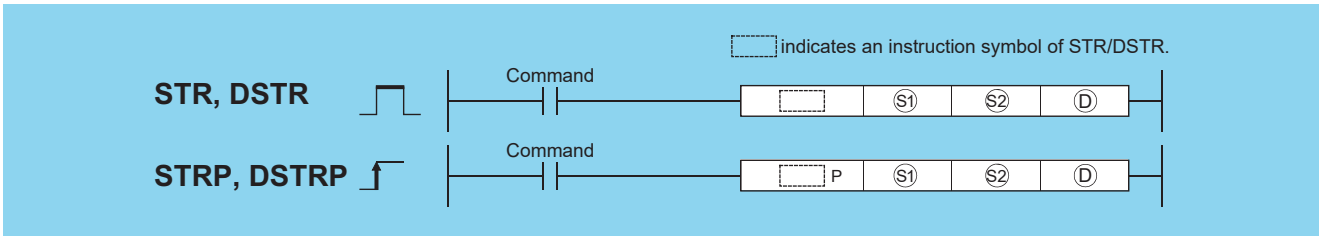


Conversion from BIN 16-bit data to character string, conversion from BIN 32-bit data to character string

STR(P), DSTR(P)

Ver. Basic | High performance | Process | Redundant | Universal | LCPU

• Basic model QCPU: The serial number (first five digits) is "04122" or later.



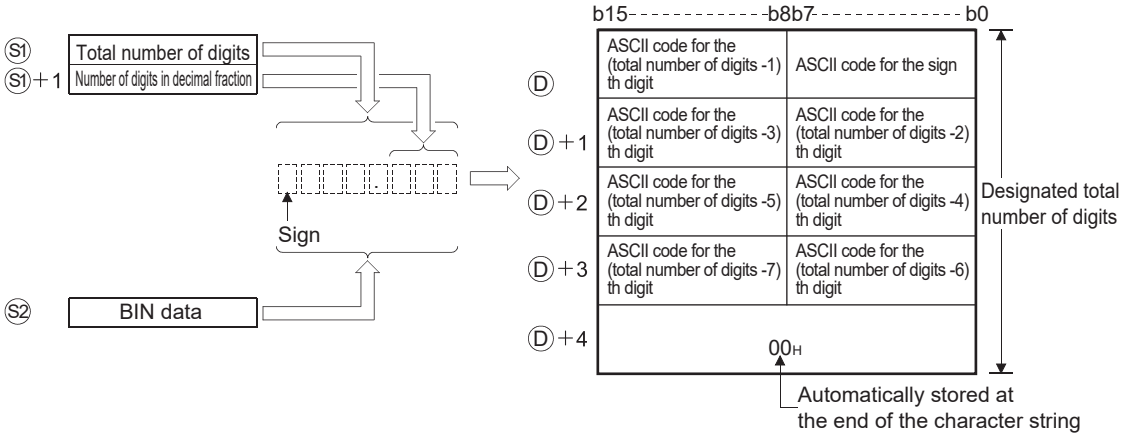
(S1): Head number of the devices where the digits numbers for the numerical value to be converted are stored (BIN 16 bits)
(S2): BIN data to be converted (BIN 16/32 bits)
(D): Head number of the devices where the converted character string will be stored (character string)

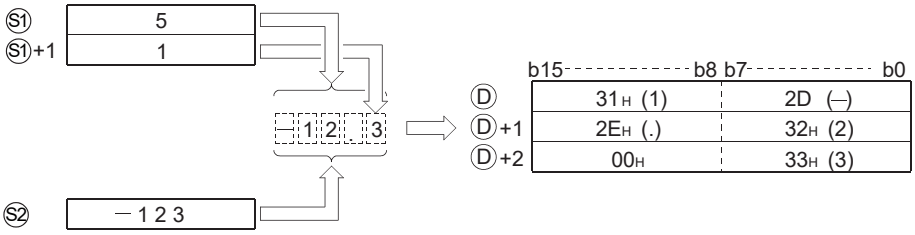
Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(S1)	○	○		○				—	—
(S2)	○	○		○				○	—
(D)	—	○		—				—	—

Processing details

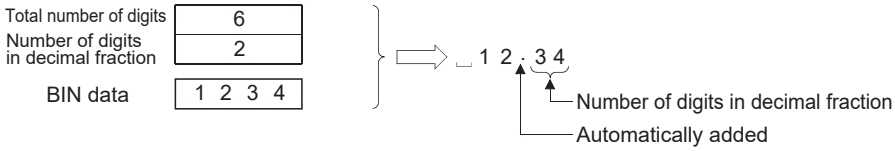
■STR

- Adds a decimal point to the BIN 16-bit data designated by (S2) at the location designated by (S1), converts the data to character string data, and stores it in the area starting from the device number designated by (D).

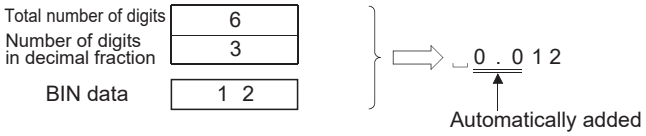




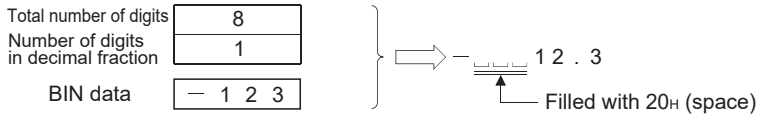
- The total number of digits that can be designated by (S1) is from 2 to 8.
- The number of digits that can be designated by (S1)+1 as a part of the decimal fraction is from 0 to 5. However, the number of digits following the decimal point must be smaller than or equal to the total number of digits minus 3.
- BIN data in the range between -32768 and 32767 can be designated at (S2).
- The converted character string data is stored at the area starting from the device number (D) as indicated below:
  - The sign "20H" (space) will be stored if the BIN data is positive, and the sign "2DH" (minus sign) will be stored if it is negative.
  - If the setting for the number of digits after the decimal fraction is anything other than "0", "2EH" (.) will automatically be stored at the position before the first of the specified number of digits. If the number of digits in the decimal fraction part of the number is "0", the ASCII code "2EH" (.) will not be stored.



  - If the total number of digits following the decimal fraction is greater than the number of BIN data digits, a zero will be added automatically and the number converted by shifting to the right, so that it would become "0.□□□□□".



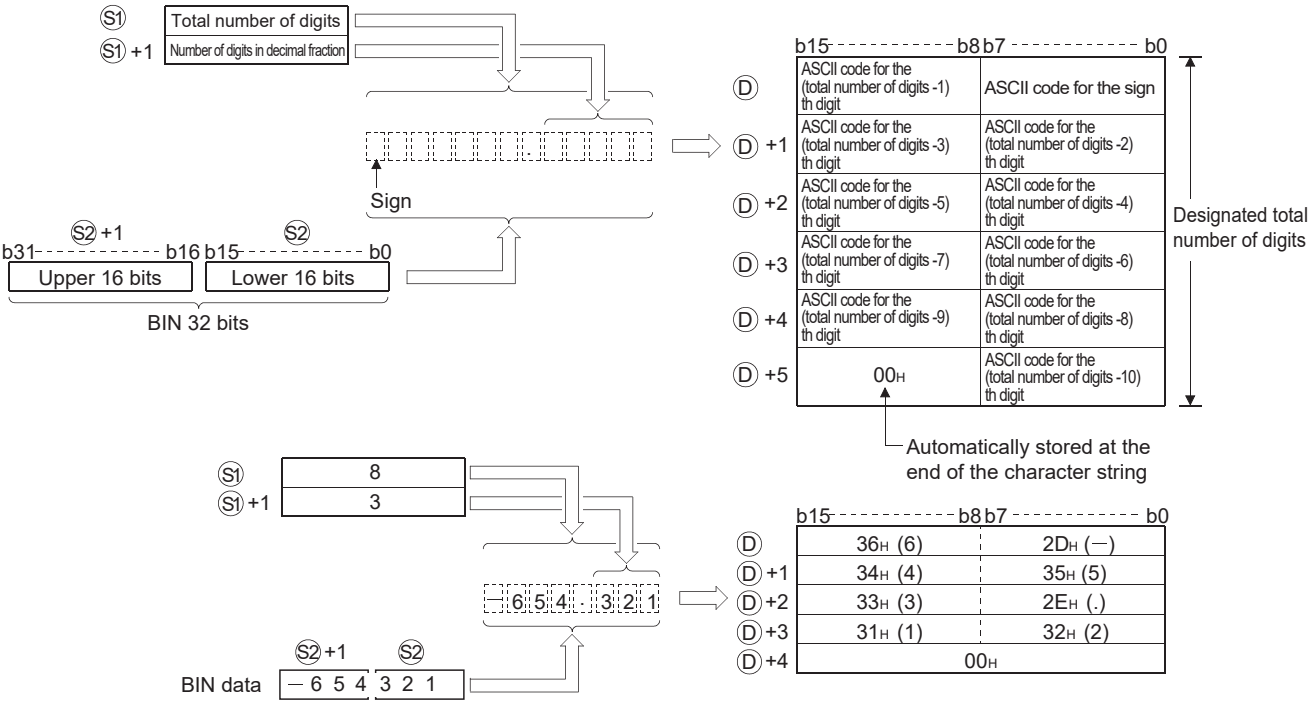
  - If the total number of digits excluding the sign and the decimal point is greater than the number of BIN data digits, "20H" (space) will be stored between the sign and the numeric value. If the number of BIN digits is greater, an error will be returned.



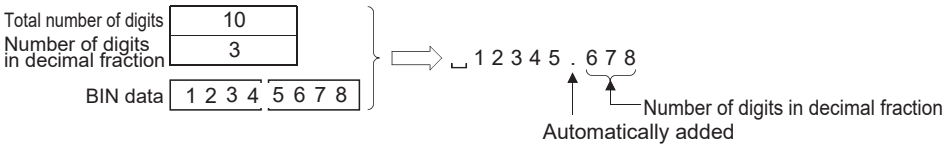
  - The value "00H" is automatically stored at the end of the converted character string.

■DSTR

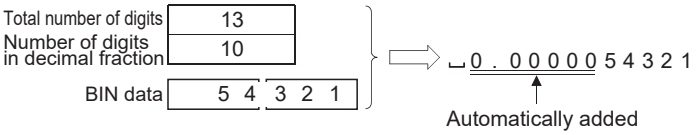
- Adds a decimal point to the BIN 32-bit data designated by (S2) at the location designated by (S1), converts the data to character string data, and stores it following the device number designated by (D).



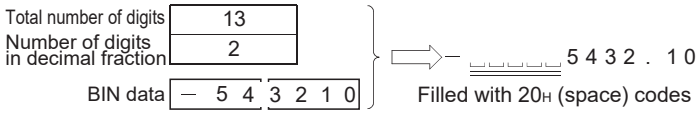
- The total number of digits that can be designated by (S1) is from 2 to 13.
- The number of digits that can be designated by (S1)+1 as a part of the decimal fraction is from 0 to 10. However, the number of digits following the decimal point must be smaller than or equal to the total number of digits minus 3.
- The BIN data that can be designated by (S1) and (S2)+1 is within the range of from -2147483648 to 2147483647.
- The converted character string data is stored at the area starting from the device number (D) as indicated below:
  - The sign "20H" (space) will be stored if the BIN data is positive, and the sign "2DH" (minus sign) will be stored if it is negative.
  - If the setting for the number of digits after the decimal fraction is anything other than "0", "2EH" (.) will automatically be stored at the position before the first of the specified number of digits. If the number of digits in the decimal fraction part of the number is "0", the ASCII code "2EH" (.) will not be stored.



  - If the total number of digits following the decimal fraction is greater than the number of BIN data digits, a zero will be added automatically and the number converted by shifting to the right, so that it would become "0.□□□□".



  - If the total number of digits excluding the sign and the decimal point is greater than the number of BIN data digits, "20H" (space) will be stored between the sign and the numeric value. If the number of BIN digits is greater, an error will be returned.



  - The value "00H" is automatically stored at the end of the converted character string.

Operation error

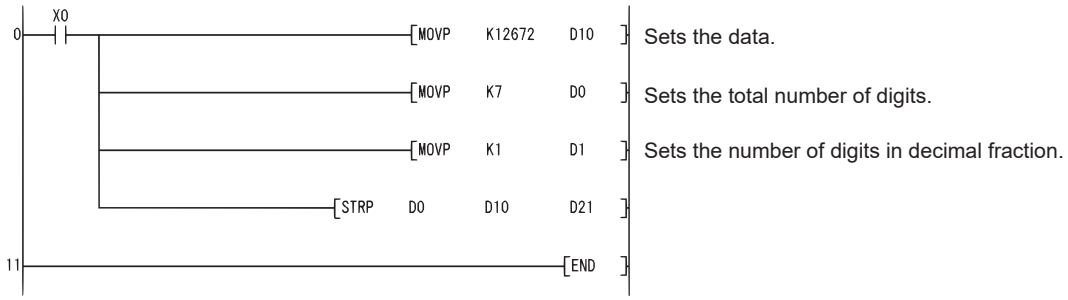
- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4100	The total number of digits specified by (S1) is outside the following ranges: • When the STR instruction is used: 2 to 8 • When the DSTR instruction is used: 2 to 13 The number of digits for a part of the decimal fraction specified by (S) +1 is outside the following ranges: • When the STR instruction is used: 0 to 5 • When the DSTR instruction is used: 0 to 10 The relationship between the total number of digits specified by (S1) and the number of digits in the decimal fraction specified by (S1) +1 is not as follows: • Total number of digits -3 ≥ Number of digits in the decimal fraction The number of digits specified by (S1) is smaller than the number of digits of the BIN data + 2 specified by (S2) ((Number of digits of (S1) < Number of digits of the BIN data at (S2) without a sign + number of digits of a sign (+ or -) + number of digits of decimal point (.))	○	○	○	○	○	○
4101	The range of the devices that store the character string specified in (D) exceeds the range of the corresponding device.	○	○	○	○	○	○

Program example

- The following program converts the BIN 16-bit data stored at D10 when X0 is turned ON in accordance with the digit designation of D0 and D1, and stores the result from D20 to D23.

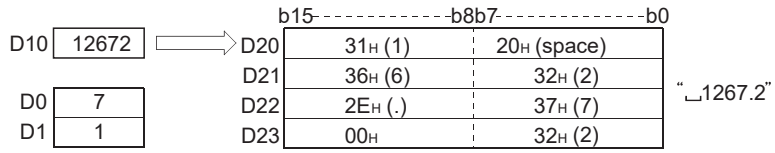
[Ladder Mode]



[List Mode]

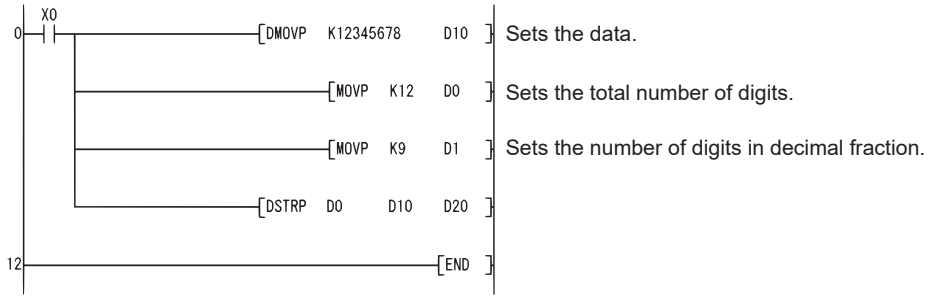
Step	Instruction	Device		
0	LD	X0		
1	MOVP	K12672	D10	
3	MOVP	K7	D0	
5	MOVP	K1	D1	
7	STRP	D0	D10	D21
11	END			

[Operation]



• The following program converts the BIN 32-bit data stored at D10 and D11 when X0 is turned ON in accordance with the digit designation of D0 and D1, and stores the result at from D20 to D26.

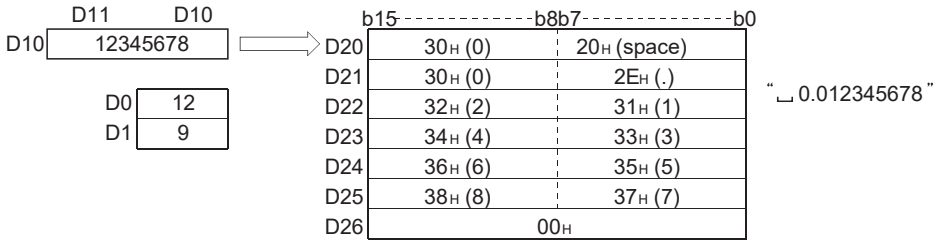
[Ladder Mode]



[List Mode]

Step	Instruction	Device		
0	LD	X0		
1	DMOVP	K12345678		D10
4	MOVP	K12	D0	
6	MOVP	K9	D1	
8	DSTRP	D0	D10	D20
12	END			

[Operation]

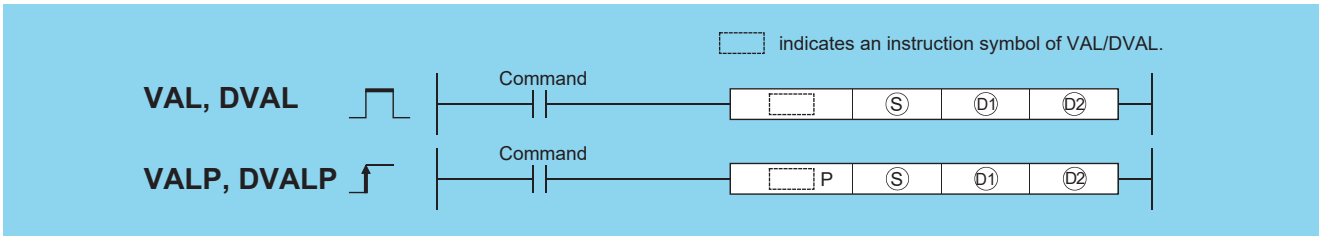


Conversion from character string to BIN 16-bit data, conversion from character string to BIN 32-bit data

VAL(P), DVAL(P)

Ver. Basic | High performance | Process | Redundant | Universal | LCPU

• Basic model QCPU: The serial number (first five digits) is "04122" or later. (Compatible GX Developer: Version 8.00A or later)



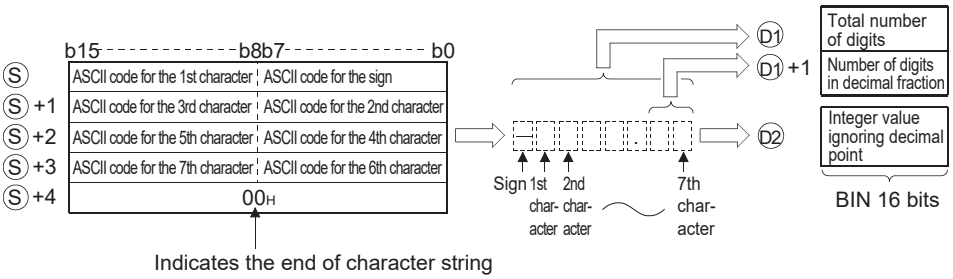
(S): Character string to be converted to BIN data or head number of the devices where the character string is stored (character string)
(D1): Head number of the devices where the number of digits of the converted BIN data will be stored (BIN 16 bits)
(D2): Head number of the devices where the converted BIN data will be stored (BIN 16/32 bits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant $	Others
	Bit	Word		Bit	Word				
(S)	—	○		—				○	—
(D1)	○	○		—				—	—
(D2)	○	○		○				—	—

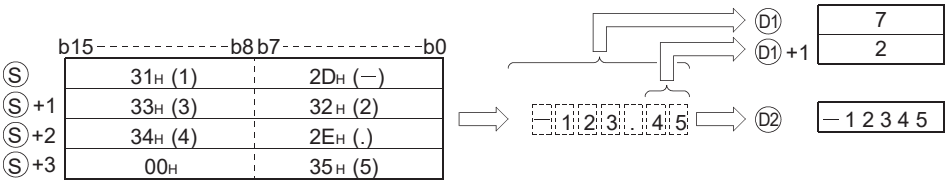
Processing details

■VAL

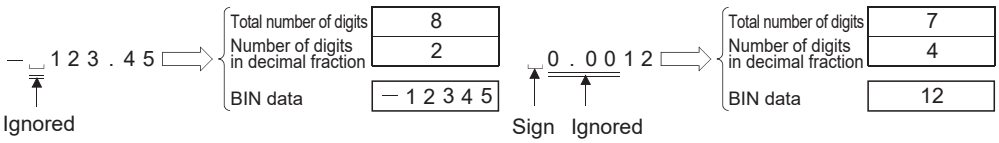
- Converts character strings stored in the device numbers starting from that designated at (S) to BIN 16-bit data, and stores the number of digits and BIN data in (D1) and (D2). For conversions from character strings to BIN, all data from the device number designated by (S) to the device number where "00H" is stored will be processed as character strings.



For example, if the character string "-123.45" is designated for the area starting from (S), the operation result would be stored at (D1) and (D2) in the following manner:



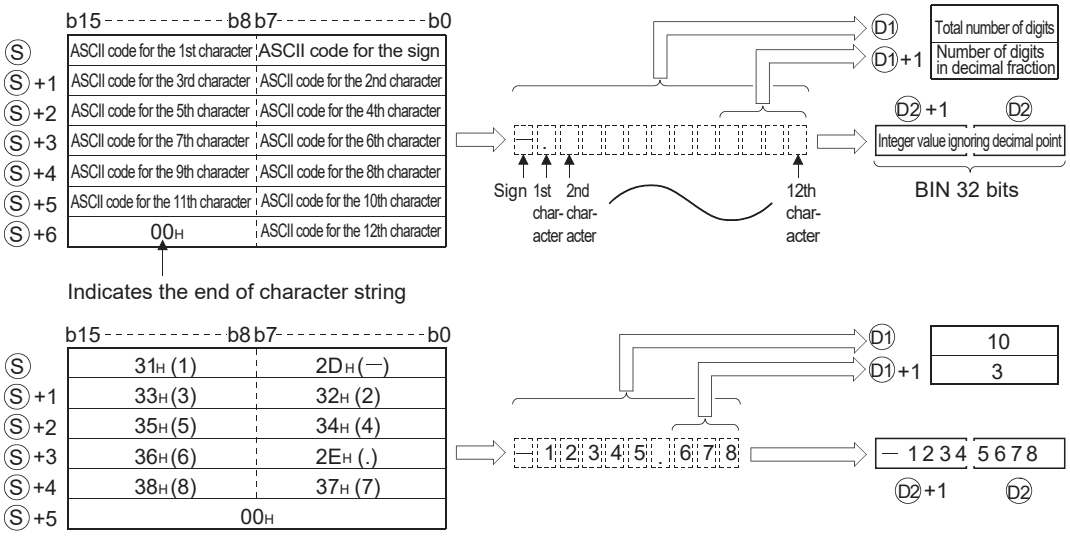
- The total number of characters that can be designated as a character string at (S) is from 2 to 8.
- From 0 to 5 characters from the character string designated at (S) can become the decimal fraction part. However, this number must not exceed the total number of digits minus 3.
- The range of the numerical character string that can be converted to BIN value is from -32768 to 32767, ignoring a decimal point. Numerical value character strings, excluding the sign and the decimal point, can be designated only within the range from "30H" to "39H". (Example: "-12345.6" → "-123456")
- The sign "20H" will be stored if the numerical value is positive, and the sign "2DH" will be stored if it is negative.
- "2EH" is set for the decimal point.
- The total number of digits stored at (D1) amounts to all characters expressing numerical values (including signs and decimal points). The characters following the decimal point stored at (D1)+1 include the number of characters from "2EH" (.) onward. The BIN data stored at (D2) is the character string ignoring the decimal point that has been converted to BIN data.
- In cases where the character string designated by (S) contains "20H" (space) or "30H" (0) between the sign and the first numerical value other than "0", these "20H" and "30H" are ignored in the conversion into a BIN value.



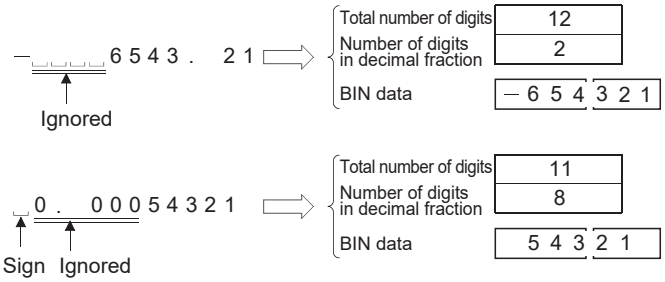
- Do not set "2EH" (.) as the final character of (S). (Example: "1234.") (High Performance model QCPU, Process CPU, Redundant CPU only)

■DVAL

- Converts the character string stored in the area starting from the device designated by (S) to BIN 32-bit data, and stores the digits numbers and BIN data in (D1) and (D2). For conversions from character strings to BIN, all data from the device number designated by (S) to the device number where "00H" is stored will be processed as character strings.



- The total number of characters in the character string indicated by (S) is from 2 to 13.
- From 0 to 10 characters in the character string indicated by (S) can be the decimal fraction part. However, this number must not exceed the total number of digits minus 3.
- The range of the numerical character string that can be converted to BIN value is from -2147483648 to 2147483647, excluding the decimal point. Numerical value character strings, excluding the sign and the decimal point, can be designated only within the range from "30H" to "39H".
- The sign "20H" will be stored if the numerical value is positive, and the sign "2DH" will be stored if it is negative.
- "2EH" is set for the decimal point.
- The total number of digits stored at (D1) amounts to all characters expressing numerical values (including signs and decimal points). The characters following the decimal point stored at (D1)+1 include the number of characters from "2EH" (.) onward. The BIN data stored at (D2) is the character string ignoring the decimal point that has been converted to BIN data.
- In cases where the character string designated by (S) contains "20H" (space) or "30H" (0) between the sign and the first numerical value other than "0", these "20H" and "30H" are ignored in the conversion into a BIN value.



- Do not set "2EH" (.) as the final character of (S). (Example: "1234.") (High Performance model QCPU, Process CPU, Redundant CPU only)

Operation error

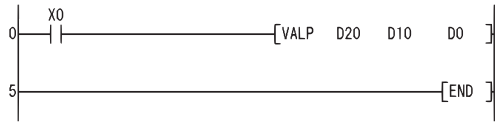
• In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4100	The number of characters in the character string specified by (S) is outside the following ranges: • When VAL instruction is in use: 2 to 8 • When DVAL instruction is in use: 2 to 13 The number of characters in the decimal fraction portion of the character string specified by (S) is outside the following ranges: • When the VAL instruction is in use: 0 to 5 • When the DVAL instruction is in use: 0 to 10 The total number of characters in the character string specified by (S) and the number of characters in the decimal fraction part stand in a relationship that is outside the following ranges: Total number of characters -3 ≥ Number of characters in the decimal fraction part An ASCII code other than "20H" or "2DH" has been set for the sign. An ASCII code other than "30H" to "39H" or "2EH" (decimal point) has been set as a digit for one of the individual numbers. There has been more than one decimal points set in the value. The converted BIN value exceeds the following ranges: • When the VAL instruction is in use: -32768 to 32767 • When the DVAL instruction is in use: -2147483648 to 2147483647	○	○	○	○	○	○
	"2EH" (.) has been set as the final character of (S).	—	○	○	○	—	—
4101	There is no NULL code "00H" within the range of the corresponding devices starting from the device number specified by (S).	○	○	○	○	○	○

Program example

- The following program reads the character string data stored from D20 to D22 as an integer, converts it to a BIN value, and stores it at D0 when X0 is ON.

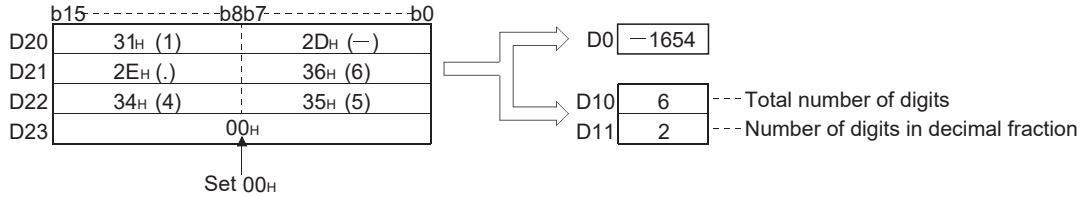
[Ladder Mode]



[List Mode]

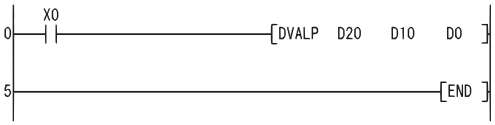
Step	Instruction	Device		
0	LD	X0		
1	VALP	D20	D10	D0
5	END			

[Operation]



- The following program reads the character string data stored from D20 to D24 as an integer, converts it to a BIN value, and stores it at D0 when X0 is ON.

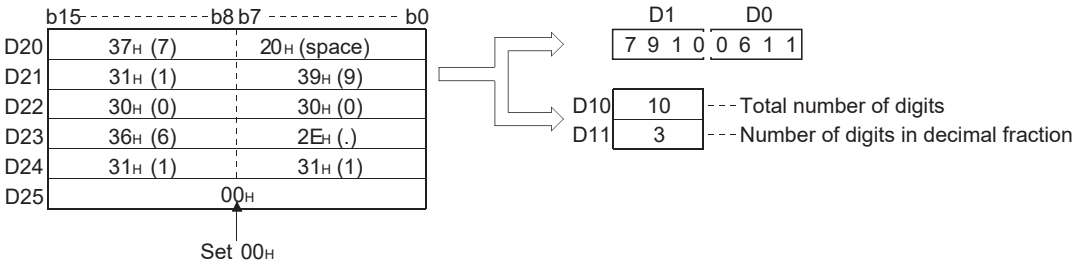
[Ladder Mode]



[List Mode]

Step	Instruction	Device		
0	LD	X0		
1	DVALP	D20	D10	D0
5	END			

[Operation]

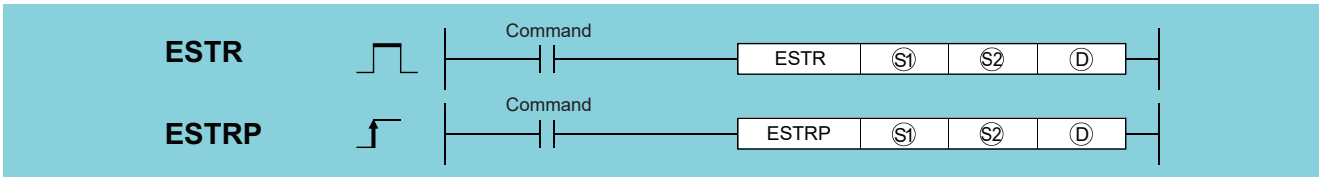


Conversion from floating-point data to character string data

ESTR(P)

Ver. Basic | High performance | Process | Redundant | Universal | LCPU

• Basic model QCPU: The serial number (first five digits) is "04122" or later.



(S1): 32-bit floating decimal point data to be converted or head number of the devices where the data is stored (real number)
(S2): Head number of the devices where display designation for the numerical value to be converted is stored (BIN 16 bits)
(D): Head number of the devices where the converted character string will be stored (character string)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant E	Others
	Bit	Word		Bit	Word				
(S1)	—	○		—	○		○*1	○	—
(S2)	—	○		—	—		—	—	—
(D)	—	○		—	—		—	—	—

*1 Applicable for the Universal model QCPU, LCPU.

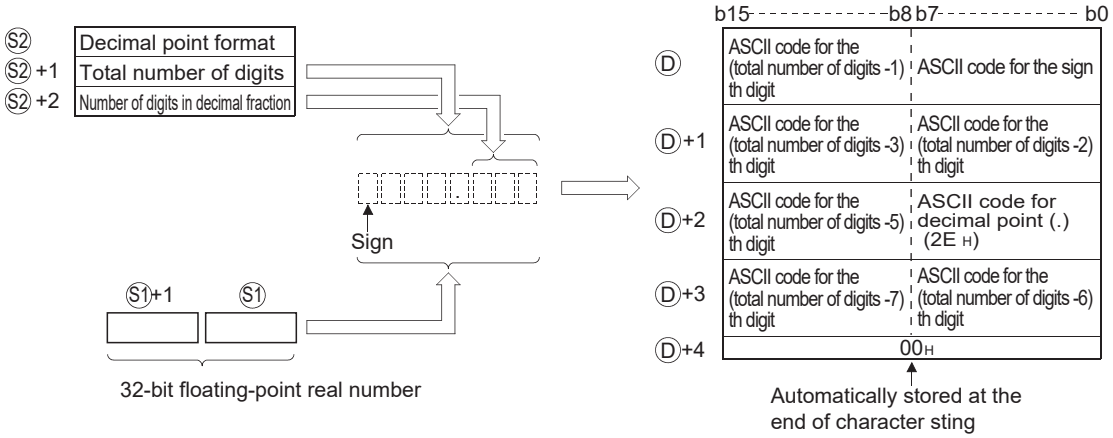
Processing details

• Converts the 32-bit floating decimal point data designated by (S1) to a character string according to the display designation specified by (S2), and stores the result into the area starting from the device number designated by (D).
• The post-conversion data differs depending on the display designation designated by (S2).

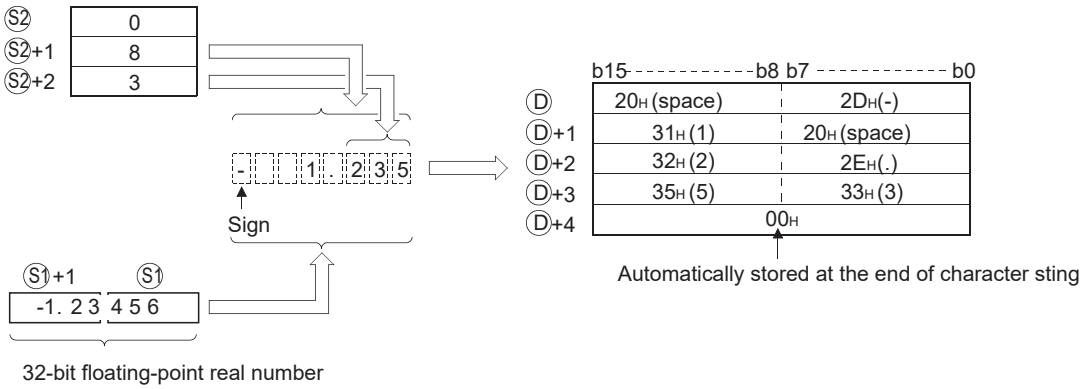
(S2)	0: Decimal point format 1: Exponent format	The converted data differs depending on the format selected at (S2).
(S2)+1	Total number of digits	··· Setting is possible in the range from 2 to 24.
(S2)+2	Number of digits in decimal fraction	

• When an input value is set using a programming tool, a rounding error may occur. For precautions, refer to Page 90 Precautions.

■When using decimal point format



For example, in a case where there are 8 digits in total, with 3 digits in the decimal fraction part, and the value designated is -1.23456, the operation result would be stored in the area starting from (D) in the following manner:

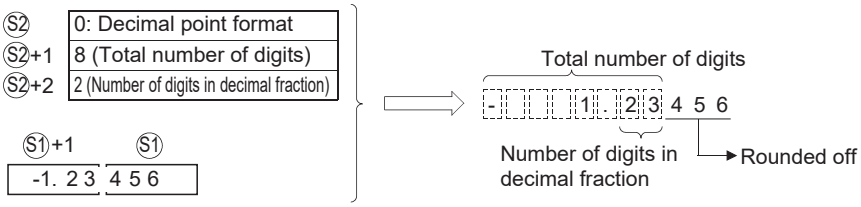


- The total number of digits that can be designated by (S2)+1 is as shown below:
  - When the number of decimal fraction digits is "0": Number of digits (max.: 24) ≥ 2
  - When the number of decimal fraction digits is other than "0": Number of digits (max.: 24) ≥ (Number of decimal fraction digits + 3)
- The number of digits of decimal fraction part that can be designated by (S2)+2 is from 0 to 7. However, the number of digits following the decimal point must be smaller than or equal to the total number of digits minus 3.

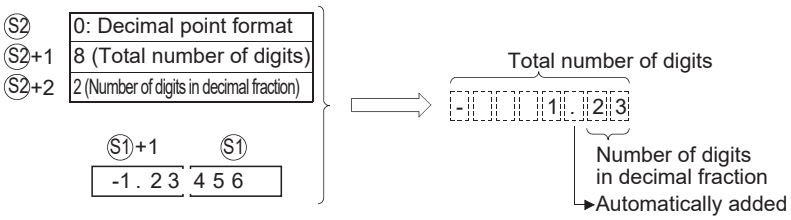
• The converted character string data is stored at the area starting from the device number (D) as indicated below:

• The sign "20H" (space) will be stored if the 32-bit floating decimal point type real number is positive, and the sign "2DH" (minus sign) will be stored if it is negative.

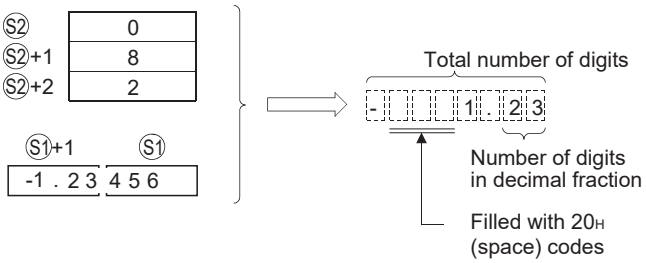
• If the decimal fraction part of a 32-bit floating point real number data is out of the range of the digits of decimal fraction part, the lower decimal values will be rounded off.



• If the number of digits following the decimal point has been set at any value other than "0", "2EH" (.) will automatically be stored at the position before the first of the specified number of digits. If the number of digits in the decimal fraction part of the number is "0", the ASCII code "2EH" (.) will not be stored.

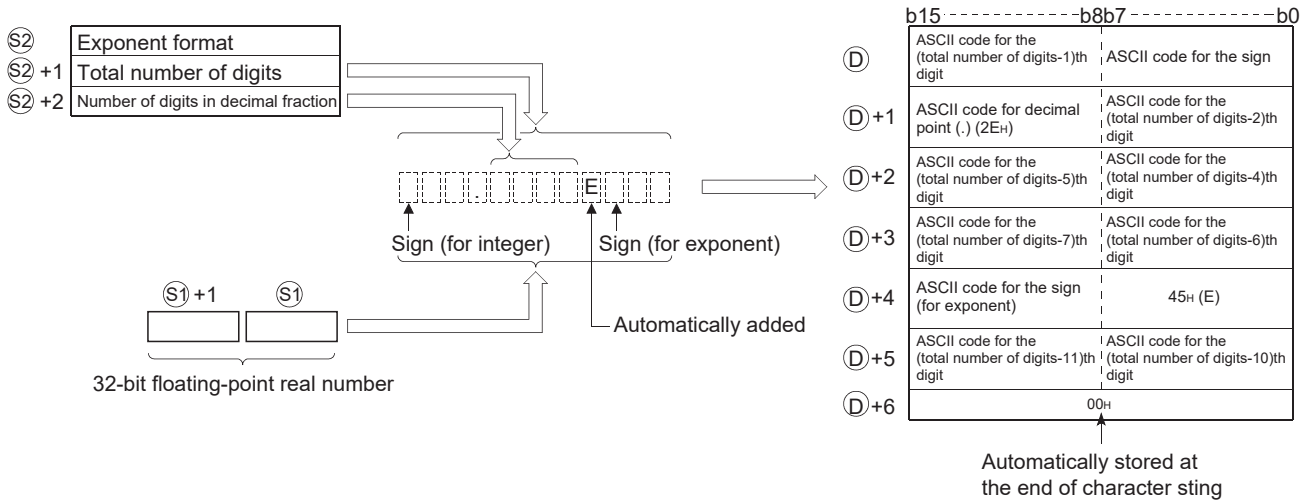


• If the total number of digits, excluding the sign, the decimal point and the decimal fraction part, is greater than the integer part of the 32-bit floating point type real number data, "20H (space)" will be stored between the sign and the integer part.

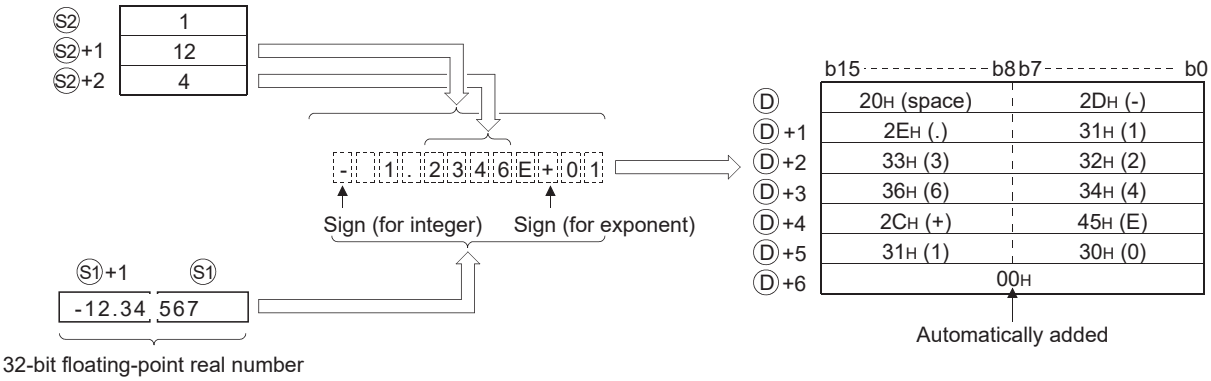


• The value "00H" is automatically stored at the end of the converted character string.

■When using exponent format

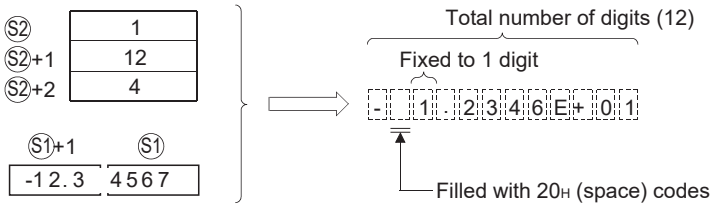


For example, in a case where there are 12 digits in total, with 4 digits in the decimal fraction part, and the value designated is -12.34567, the operation result would be stored in the area starting from (D) in the following manner:

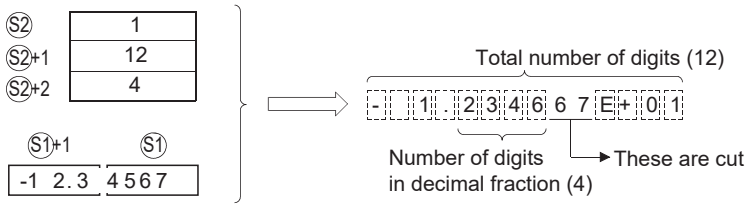


- The total number of digits that can be designated by (S2)+1 is as shown below:
  - When the number of decimal fraction digits is "0": Number of digits (max.: 24) ≥ 2
  - When the number of decimal fraction digits is other than "0": Number of digits (max.: 24) ≥ (Number of decimal fraction digits + 7)
- The number of digits of decimal fraction part that can be designated by (S2)+2 is from 0 to 7. However, the number of digits in the decimal fraction portion should be equal to or less than the total number of digits minus 7.

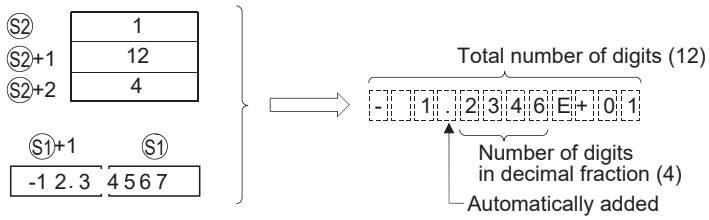
- The converted character string data is stored at the area starting from the device number (D) as indicated below:
  - If the 32-bit floating decimal point type real number data is positive in value, the sign before the integer will be stored as ASCII code "20H" (space), and if it is a negative value, the sign will be stored as "2DH" (-).
  - The integer portion is fixed to one digit. 20H (space) will be stored between the integer and the sign.



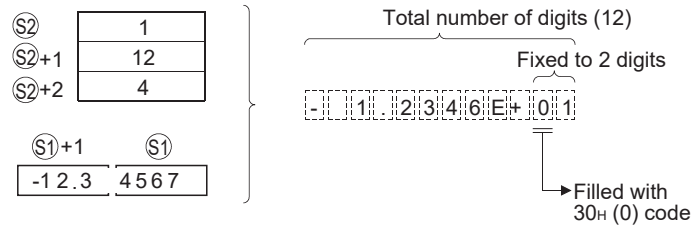
  - If the decimal fraction part of the 32-bit floating point real number data is out of the range of the digits of decimal fraction part, the lower decimal values will be rounded off.



  - If the number of digits of the decimal fraction part has been set at any value other than "0", "2EH" (.) will automatically be stored at the position before the first of the specified number of digits. If the number of digits in the decimal fraction part of the number is "0", the ASCII code "2EH" (.) will not be stored.



  - The ASCII code "2CH" (+) will be stored as the sign for the exponent portion of the value if the exponent is positive in value, and the code "2DH" (-) will be stored if the exponent is a negative value.
  - The exponent portion is fixed at 2 digits. If the exponent portion is only 1 digit, the ASCII code "30H" (0) will be stored between the sign and the exponent portion of the number.



  - The value "00H" is automatically stored at the end of the converted character string.

Operation error

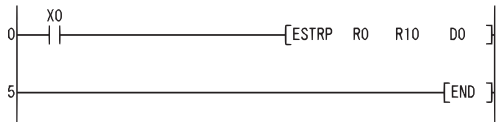
- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4100	The (S1) value is not within the following range: 0, $2^{-126} \leq \lvert (S1) \rvert < 2^{128}$ The format specified by (S2) is other than 0 and 1. The total number of digits specified by (S2) + 1 is outside the following ranges: [When using decimal point format] • When the number of decimal fraction digits is "0": Total number of digits ≥ 2 • When the number of decimal fraction digits is not "0": Total number of digits ≥ (Number of decimal fraction digits + 3) [When using exponent format] • When the number of decimal fraction digits is "0": Total number of digits ≥ 6 • When the number of decimal fraction digits is not "0": Total number of digits ≥ (Number of decimal fraction digits + 7) The number of digits for the decimal fraction portion specified by (S2) +2 is outside the following ranges: • When using the decimal point format: Number of decimal fraction digits ≤ (Total number of digits -3) • When using the exponent format: Number of decimal fraction digits ≤ (Total number of digits -7) The value in more than 24 digits was specified.	○	○	○	○	—	—
4101	The range of the devices that store the character string specified in (D) exceeds the range of the corresponding device.	○	○	○	○	○	○
	The range of the device specified by (S2) exceeds the range of the corresponding device.	—	—	—	—	○	○
4140	The (S1) value is not within the following range: 0, $2^{-126} \leq \lvert (S1) \rvert < 2^{128}$ The specified device value is -0, unnormalized number, nonnumeric, and ±∞.	—	—	—	—	○	○

Program example

- The following program converts the 32-bit floating point type real number data which had been stored at R0 and R1 in accordance with the conversion designation that is being stored at R10 to R12, and stores the result following D0 when X0 goes ON.

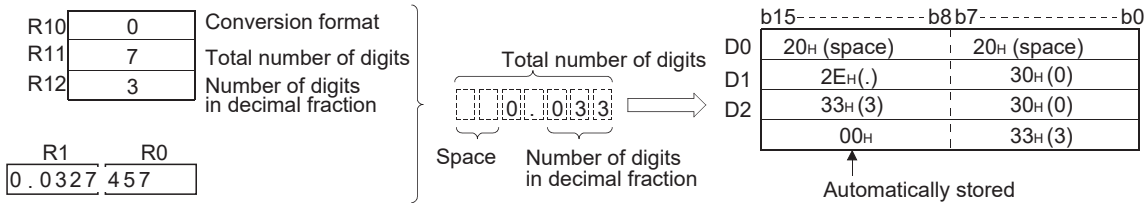
[Ladder Mode]



[List Mode]

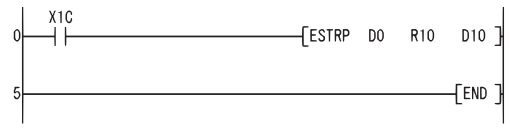
Step	Instruction	Device		
0	LD	X0		
1	ESTRP	R0	R10	D0
5	END			

[Operation]



- The following program converts the 32-bit floating decimal point type real number data which had been stored at D0 and D1 in accordance with the conversion designation that is being stored at R10 to R12, and stores the result following D10 when X1C goes ON.

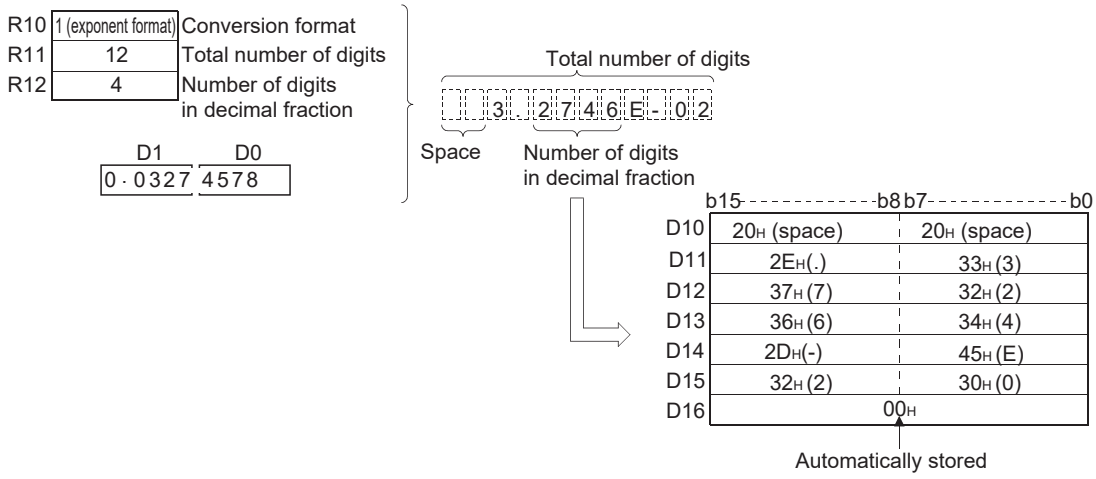
[Ladder Mode]



[List Mode]

Step	Instruction	Device		
0	LD	X1C		
1	ESTRP	D0	R10	D10
5	END			

[Operation]



Conversion from character string to floating-point data

EVAL(P)

Ver. Basic | High performance | Process | Redundant | Universal | LCPU

• Basic model QCPU: The serial number (first five digits) is "04122" or later.

(S): Character string data to be converted to 32-bit floating decimal point real number data or head number of the devices where the character string data is stored (character string)

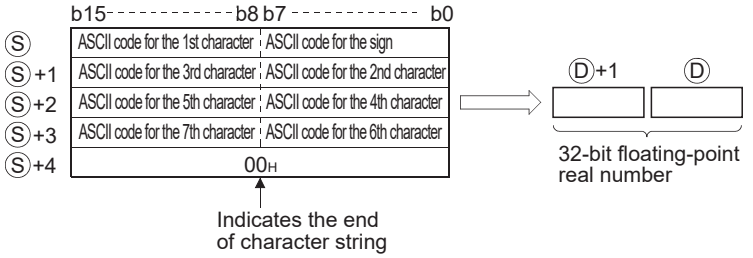
(D): Head number of the devices where the converted 32-bit floating decimal point real number data will be stored (real number)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant $	Others
	Bit	Word		Bit	Word				
(S)	—	○		—	—		—	○	—
(D)	—	○		—	○		○*1	—	—

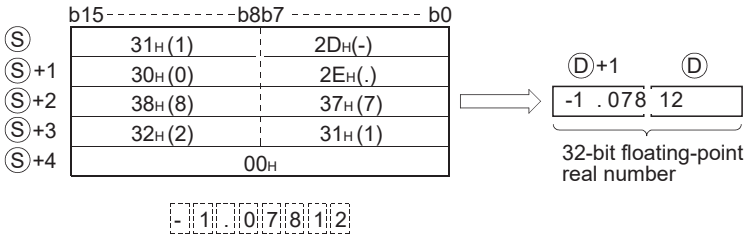
*1 Applicable for the Universal model QCPU, LCPU.

Processing details

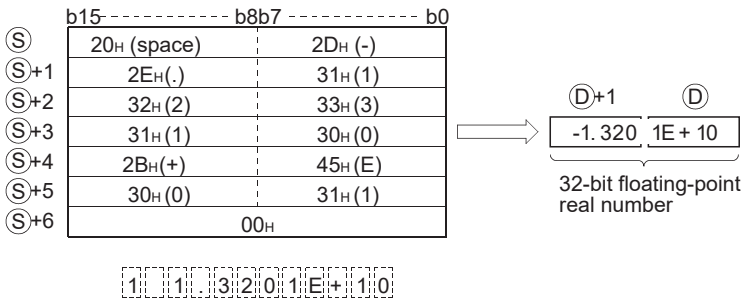
- Converts character string stored in the area starting from the device number designated by (S) to 32-bit floating point type real number, and stores result at device designated by (D).
- The designated character string can be converted to 32-bit floating point type real number data either in the decimal point format or the exponent format.



• When using decimal point format



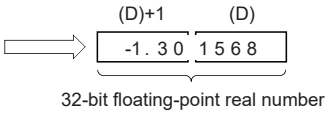
• When using exponent format



- Excluding the sign, decimal point, and exponent portion of the result, 6 digits of the character string designated by (S) to be converted to a 32-bit floating decimal point type real number will be effective.

• When using decimal point format (QnUDVCPU and QnUDPVCPU)

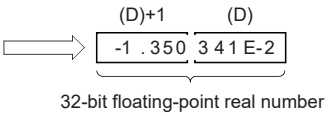
	b15 to b8		b7 to b0	
(S)	20H	(space)	2DH	(-)
(S)+1	31H	(1)	20H	(space)
(S)+2	33H	(3)	2EH	(.)
(S)+3	31H	(1)	30H	(0)
(S)+4	36H	(6)	35H	(5)
(S)+5	31H	(1)	38H	(8)
(S)+6	00H		32H	(2)



32-bit floating-point real number

• When using exponent format (QnUDVCPU and QnUDPVCPU)

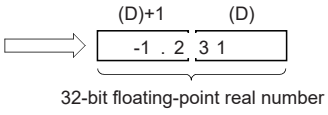
	b15 to b8		b7 to b0	
(S)	20H	(space)	2DH	(-)
(S)+1	2EH	(.)	31H	(1)
(S)+2	35H	(5)	33H	(3)
(S)+3	33H	(3)	30H	(0)
(S)+4	31H	(1)	34H	(4)
(S)+5	45H	(E)	32H	(2)
(S)+6	30H	(0)	2DH	(-)
(S)+7	00H		32H	(2)



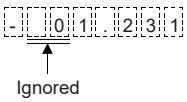
32-bit floating-point real number

- In the decimal point format, if "2BH" (+) is specified for the sign or if the designation of sign is omitted, conversion is made assuming a positive value. If "2DH" (-) is specified for the sign, the character string is converted assuming a negative value.
- In the exponent format, if "2BH" (+) is specified for the sign in the exponent portion or if the designation of sign is omitted, conversion is made assuming a positive value. If "2DH" (-) is specified for the sign in the exponent portion, the character string is converted assuming a negative value.
- In a case where the ASCII code "20H (space)" or "30H" (0) exists between numbers not including the initial zero in a character string specified by (S), it will be ignored when the conversion is done.

	b15 to b8		b7 to b0	
(S)	20H	(space)	2DH	(-)
(S)+1	31H	(1)	30H	(0)
(S)+2	32H	(2)	2EH	(.)
(S)+3	31H	(1)	33H	(3)
(S)+4	00H			

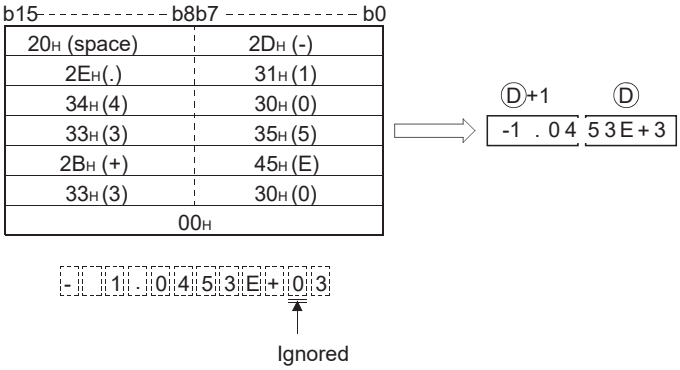


32-bit floating-point real number



Ignored

- In a case where the ASCII code "30H (0) " exists between the character "E" and a number in an exponent format character string, the "30H" would be ignored when the conversion is performed.



- If the "20H" (space) code is contained in the character string, the code is ignored in the conversion.
- Up to 24 characters can be set for a character string. The codes "20H" (space) and "30H" (0) contained in the character string are also counted as a character.

Operation error

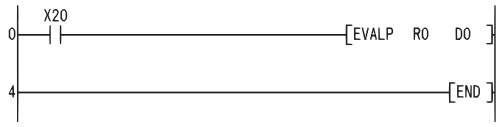
- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4100	The improper data as shown below, which cannot be converted into (S), have been set. • The characters other than "30H"(0) to "39H"(9) exist in the integral part or decimal part. • Two or more "2EH"(.) exist in the character string. • The characters other than "45H"(E), "65H" (e), "2BH" (+), "2DH" (-) exist in the character string. • The multiple exponents of "45H" (E), "65H" (e) exist in the character string. • Three digits or more numeric values are described on the exponents in the character string. • Multiple codes exist on the exponent of "2BH"(+), "2DH(-)" in the character string. • Multiple codes of "2BH"(+), "2DH(-)" exist in the positive number part of the character string in a decimal point format; in the mantissa part of the character string in an exponential format. The number of character at (S) or later exceeds 0 or 24 characters.	○	○	○	○	○	○
4101	There is no NULL code "00H" within the range of the corresponding devices starting from the device number specified by (S).	○	○	○	○	○	○
4141	The operation result exceeds the following range. (when an overflow occurs) | Data after conversion | < 2^{128}	—	—	—	—	○	○

Program example

- The following program converts the character string stored in the area starting from R0 to a 32-bit floating decimal point type real number, and stores the result at D0 and D1 when X20 is turned ON.

[Ladder Mode]

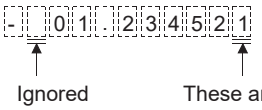


[List Mode]

Step	Instruction	Device	
0	LD	X20	
1	EVALP	R0	D0
4	END		

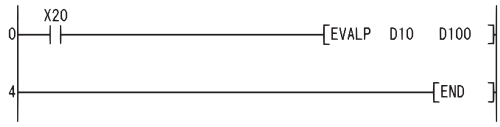
[Operation]

	b15 ---------- b8b7 ---------- b0	
R0	20_H (space)	$2D_H$ (-)
R1	31_H (1)	30_H (0)
R2	32_H (2)	$2E_H$ (.)
R3	34_H (4)	33_H (3)
R4	32_H (2)	35_H (5)
R5	00_H	31_H (1)



- The following program converts the character string stored in the area starting from D10 to a 32-bit floating decimal point type real number, and stores the result at D100 and D101 when X20 is turned ON.

[Ladder Mode]

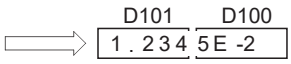


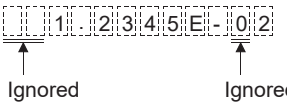
[List Mode]

Step	Instruction	Device	
0	LD	X20	
1	EVALP	D10	D100
4	END		

[Operation]

	b15 ---------- b8b7 ---------- b0	
D10	20_H (space)	20_H (space)
D11	$2E_H$ (.)	31_H (1)
D12	33_H (3)	32_H (2)
D13	35_H (5)	34_H (4)
D14	$2D_H$ (-)	45_H (E)
D15	32_H (2)	30_H (0)
D16	00	

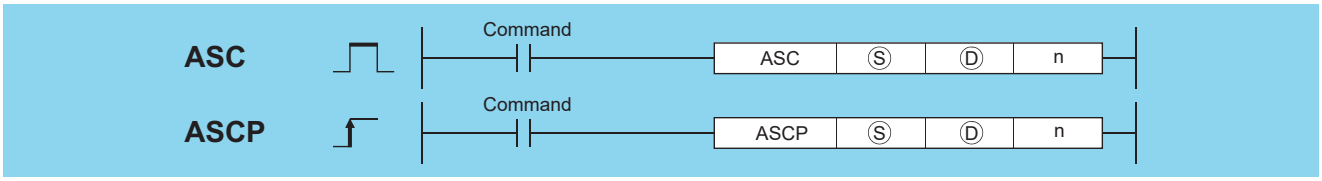




Conversion from hexadecimal BIN to ASCII

ASC(P)

Basic | High performance | Process | Redundant | Universal | LCPU

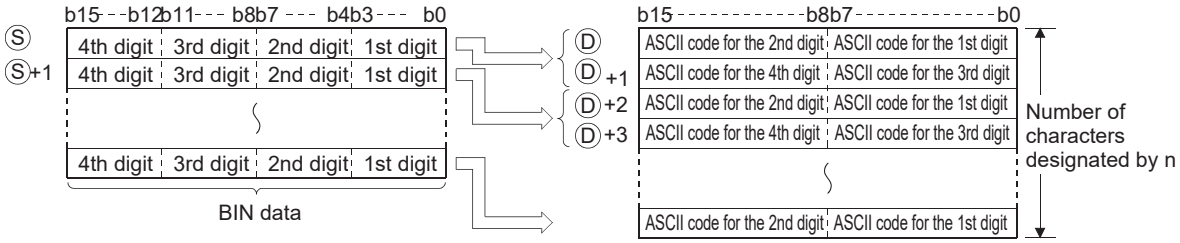


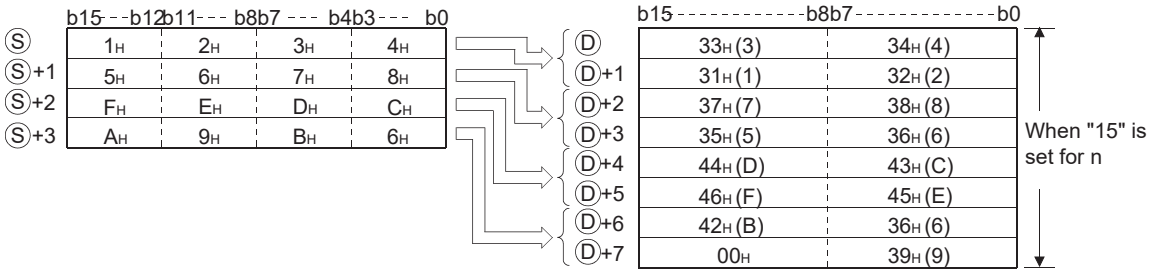
(S): Head number of the devices where BIN data to be converted to a character string is stored (BIN 16 bits)
(D): Head number of the devices where the converted character string will be stored (character string)
n: Number of characters to be stored (BIN 16 bits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(S)	—	○		—					—
(D)	—	○		—					—
n	○	○		○					—

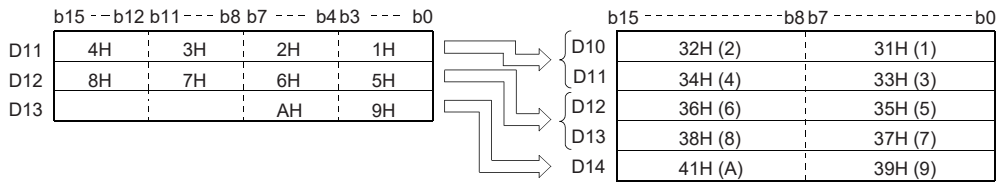
Processing details

- Converts the BIN 16-bit data stored in the area starting from the device designated by (S) to ASCII by treating the BIN data in hexadecimal representation. Then, stores the converted data into the area starting from the device designated by (D), for the number of characters specified by n.



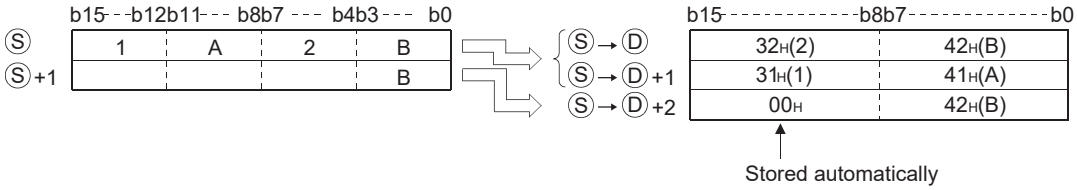


- The use of n to set the number of characters causes the BIN data range designated by (S) and the character string storage device range designated by (D) to be set automatically.
- Processing will be performed accurately even if the device range where BIN data to be converted is being stored overlaps with the device range where the converted ASCII data will be stored.



• If an odd number of characters has been designated by n, the ASCII code "00H" will be automatically stored in the upper 8 bits of the final device in the range where the character string is to be stored.

When 5 characters have been designated by n.



• If the number of characters designated by n is "0", conversion processing will not be conducted.

Operation error

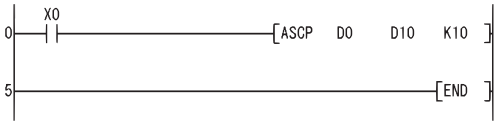
• In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4100	The character numbers designated by n are those other than 0 to 16383.	—	○	○	○	○	○
4101	The range for the number of characters designated by n following the device number designated by (S) exceeds the relevant device range. The range for the number of characters designated by n following the device number designated by (D) exceeds the relevant device range.	—	○	○	○	○	○

Program example

• The following program reads the BIN data being stored at D0 as hexadecimal values, converts them to a character string, and stores the result from D10 to D14 when X0 is turned ON.

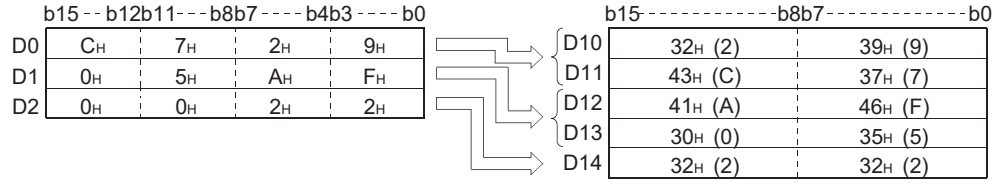
[Ladder Mode]



[List Mode]

Step	Instruction	Device		
0	LD	X0		
1	ASCP	D0	D10	K10
5	END			

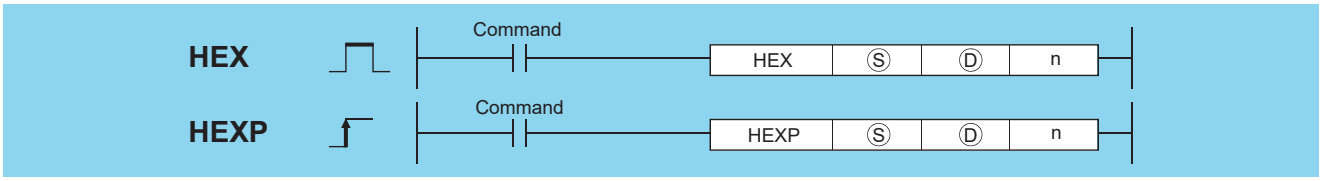
[Operation]



Conversion from ASCII to hexadecimal BIN

HEX(P)

Basic | High performance | Process | Redundant | Universal | LCPU

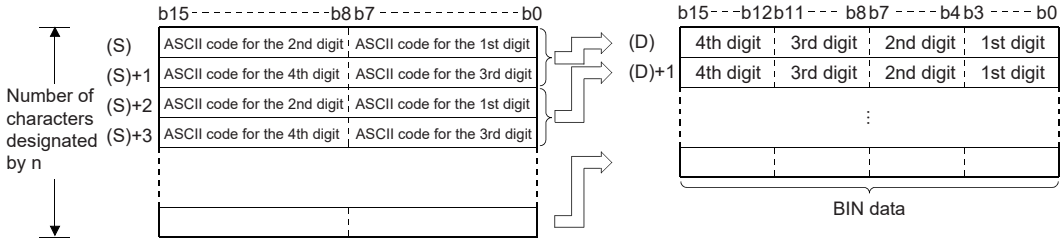


(S): Head number of the devices where a character string to be converted to BIN data is stored (character string)
(D): Head number of the devices where the converted BIN data will be stored (BIN 16 bits)
n: Number of characters to be stored (BIN 16 bits)

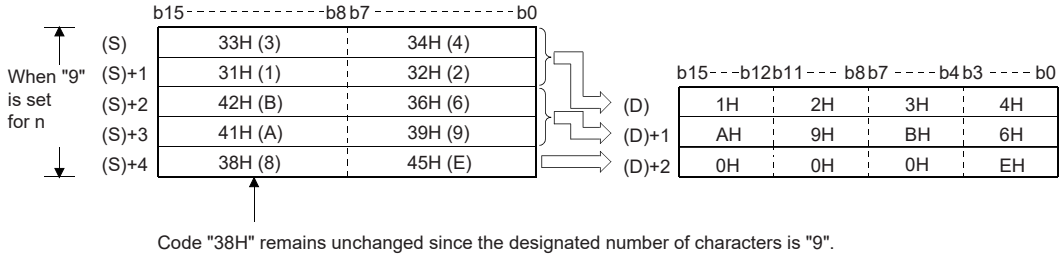
Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(S)	—	○		—					—
(D)	—	○		—					—
n	○	○		○					—

Processing details

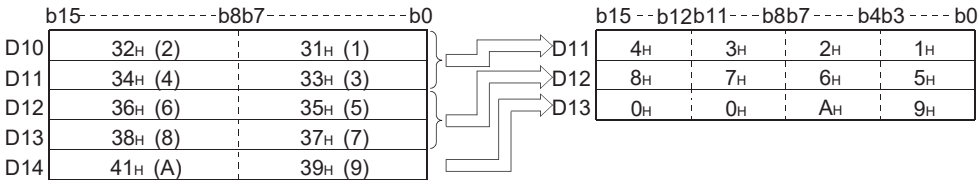
- Converts the number of characters of hexadecimal ASCII data designated by n stored in the area starting from the device number designated by (S) into BIN values and stores them in the area starting from the device number designated by (D).



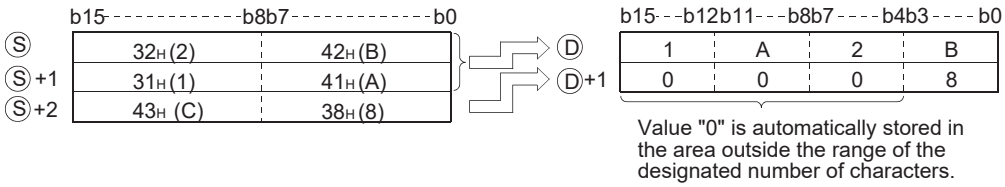
For example, if the number 9 has been designated by n, the operation would be as follows:



- When the number of characters is specified for n, the range of characters designated by (S) as well as the device range designated by (D) in which the BIN data will be stored are automatically decided.
- Accurate processing will be conducted even in cases where the range of devices where the ASCII code to be converted is being stored overlaps with the range of devices that will store the converted BIN data.



- If the number of characters designated by n is not divisible by 4, "0" will be automatically stored after the designated number of characters in the final device number of the devices which are storing the converted BIN values.



- If the number of characters designated by n is "0", conversion processing will not be conducted.
- ASCII code that can be designated by (S) includes from "30H" to "39H" and from "41H" to "46H".

Operation error

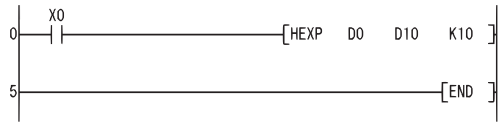
- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4100	Characters other than those outside the hexadecimal character string (characters that are not in the range between "30H" to "39H" and "41H" to "46H") have been set in the device specified by (S). The character numbers designated by n are those other than 0 to 16383.	—	○	○	○	○	○
4101	The range of the device specified by (S) exceeds the range from (S) to (S) + the number of characters specified in n (including (S)). The range of the device specified by (D) exceeds the range from (D) to (D) + the number of characters specified in n (including (D)). n is negative.	—	○	○	○	○	○

Program example

- The following program converts the character string being stored from D0 to D4 to BIN data and stores the result from D10 to D14 when X0 goes ON.

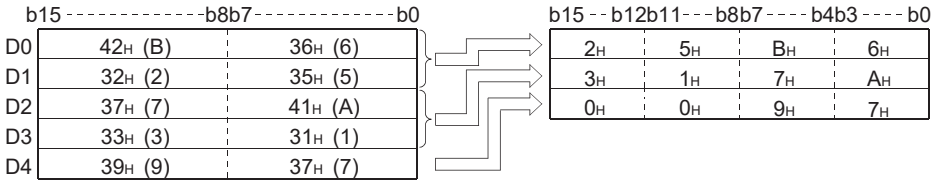
[Ladder Mode]



[List Mode]

Step	Instruction	Device		
0	LD	X0		
1	HEXP	D0	D10	K10
5	END			

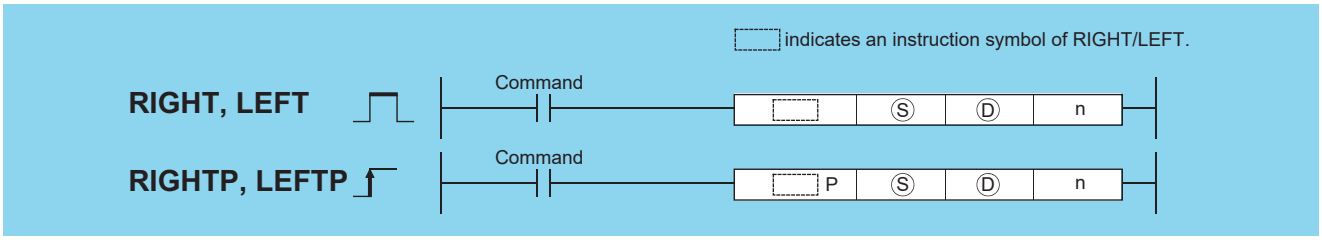
[Operation]



Extracting character string data from the right, extracting character string data from the left

RIGHT(P), LEFT(P)

Basic | High performance | Process | Redundant | Universal | LCPU



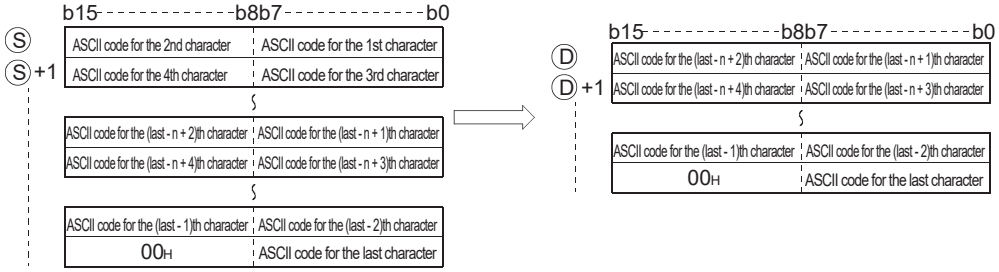
(S): Character string or head number of the devices where the character string is stored (character string)
(D): Head number of the devices where the character string consisting of n characters starting from the right or left of (S) will be stored (character string)
n: Number of characters to be extracted (BIN 16 bits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant		Others
	Bit	Word		Bit	Word			K, H	$	
(S)	—	○		—				—	○	—
(D)	—	○		—				—	—	—
n	○	○		○				○	—	—

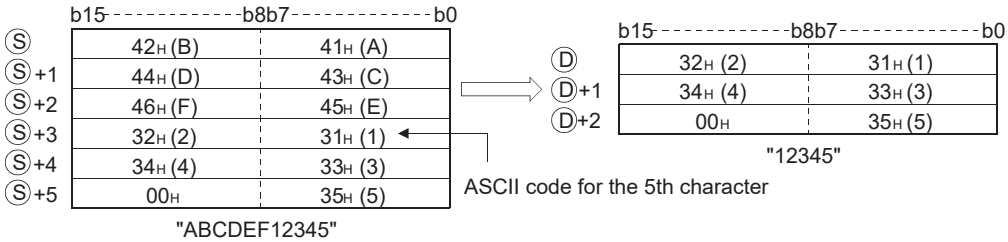
Processing details

■RIGHT

- Stores n number of characters from the right side of the character string (the end of the character string) being stored in devices starting from that whose number is designated by (S), in devices starting from that whose number is designated by (D).



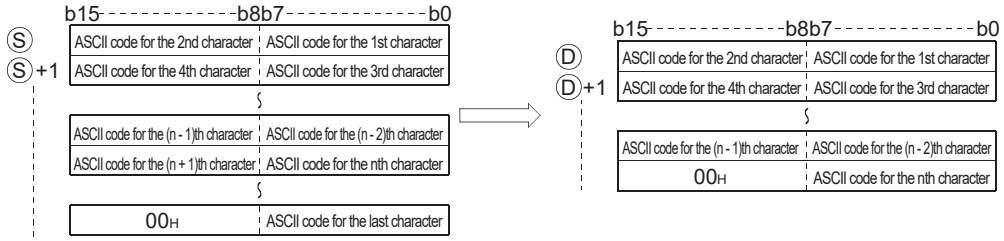
When n=5



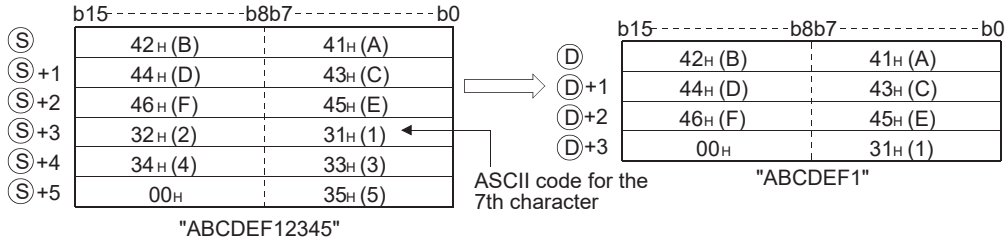
- The NULL code (00H) indicating the end of the character string is automatically added to the end of the character string. Refer to Page 92 Using character string data for the format of the character string data.
- If the number of characters designated by n is "0", the NULL code (00H) will be stored at (D).

■LEFT

- Stores n number of characters from the left side of the character string (the beginning of the character string) being stored in devices starting from that whose number is designated by (S), in devices starting from that whose number designated by (D).



When n=7



- The NULL code (00H) indicating the end of the character string is automatically added to the end of the character string. Refer to Page 92 Using character string data for the format of the character string data.
- If the number of characters designated by n is "0", the NULL code (00H) will be stored at (D).

Operation error

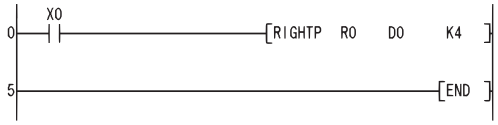
- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4100	The character number of character string designated by (S) is "0".	—	○	○	○	○	○
4101	The value of n exceeds the number of characters specified by (S). The range of the device specified by (D) exceeds the range from (D) to (D) + the number of characters specified in n (including (D)).	—	○	○	○	○	○

Program example

- The following program stores 4 characters of data from the rightmost of the character string stored in the area starting from R0, and stores it into the area starting from D0 when X0 is turned ON.

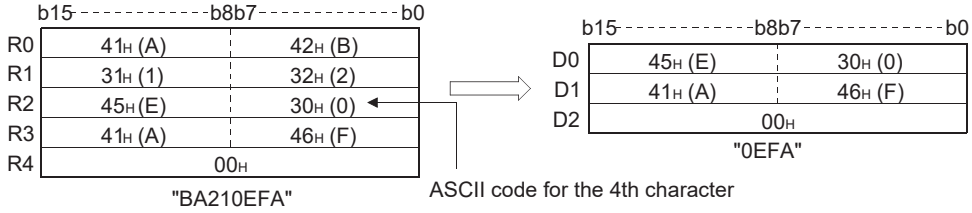
[Ladder Mode]



[List Mode]

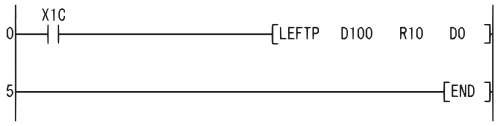
Step	Instruction	Device		
0	LD	X0		
1	RIGHTP	R0	D0	K4
5	END			

[Operation]



- The following program stores the number of characters corresponding to the value being stored in D0 from the left of the character string data being stored at D100 to the area starting from R10 when X1C is turned ON.

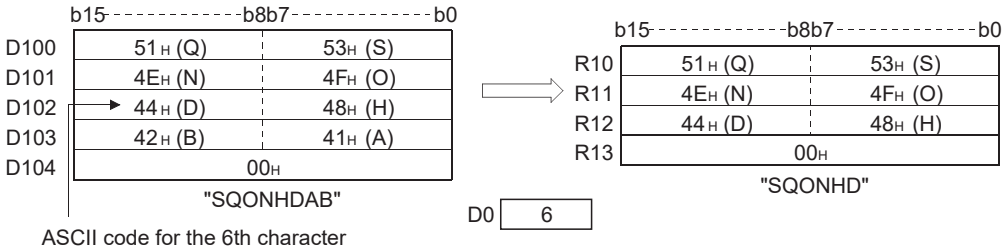
[Ladder Mode]



[List Mode]

Step	Instruction	Device		
0	LD	X1C		
1	LEFTP	D100	R10	D0
5	END			

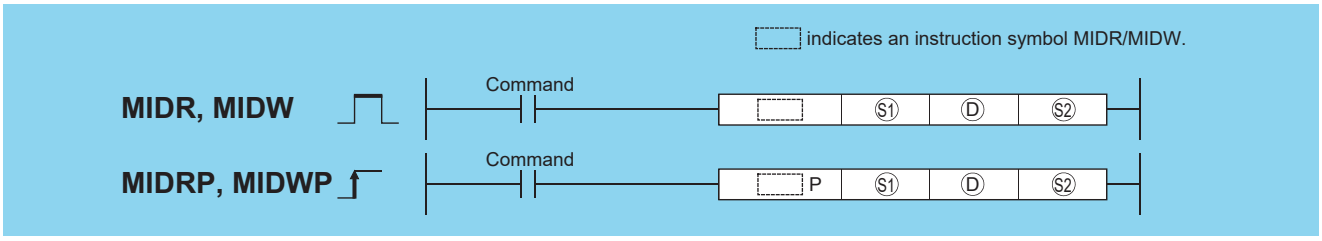
[Operation]



Random selection from character strings, random replacement in character strings

MIDR(P), MIDW(P)

Basic | High performance | Process | Redundant | Universal | LCPU



(S1): Character string or head number of the devices where the character string is stored (character string)
(D): Head number of the devices where a character string data obtained as the result of operation will be stored (character string)
(S2): Head number of the devices where the location of the first character and the number of characters will be stored (BIN 16 bits)
· (S2): Position of first character
· (S2) +1: Number of characters

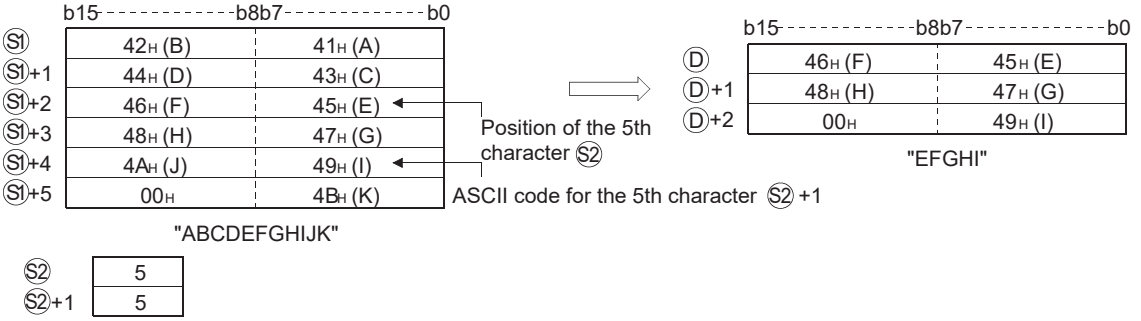
Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant $	Others
	Bit	Word		Bit	Word				
(S1)	—	○		—				○	—
(D)	—	○		—				—	—
(S2)	○	○		○				—	—

7

Processing details

■MIDR

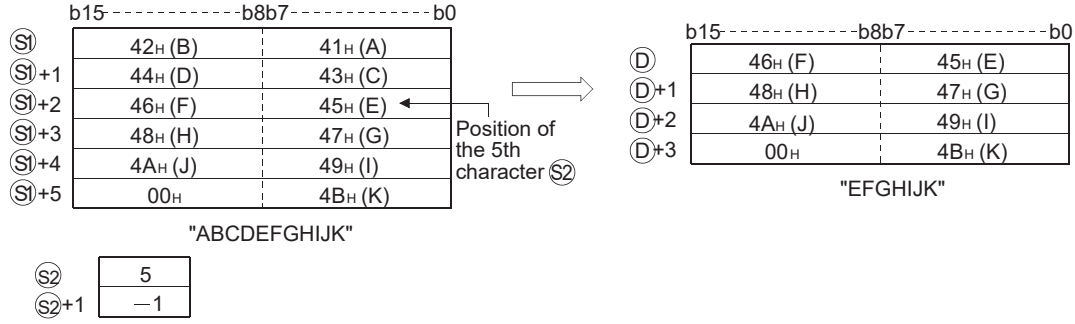
- Extracts the character string data of (S2)+1 characters, starting from the position designated by (S2), counted from the left end of the character string data designated by (S1), and stores the extracted data into the area starting from the device designated by (D).



S2	5
S2+1	5

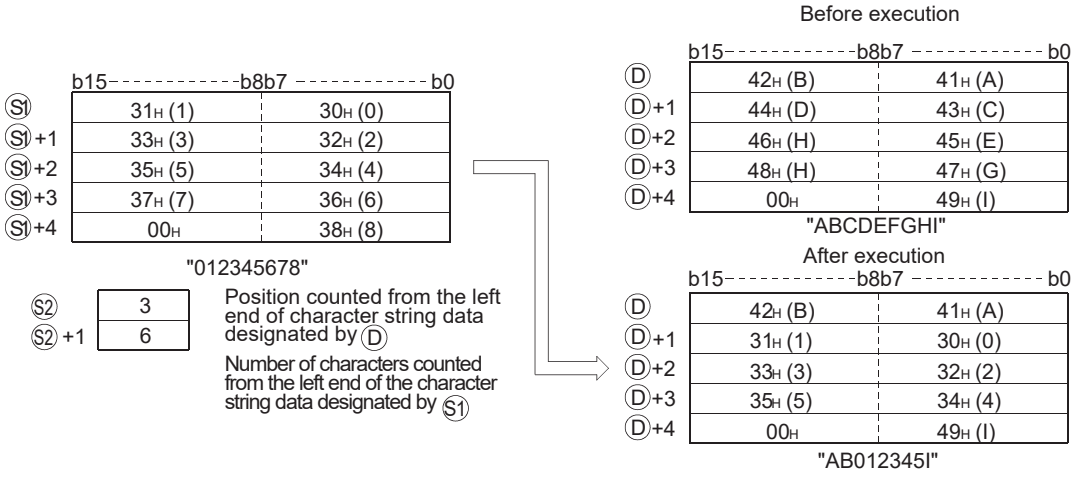
- The NULL code (00H) indicating the end of the character string is automatically added to the end of the character string. Refer to Page 92 Using character string data for the format of the character string data.
- If the number of characters designated by (S2)+1 is "0", the NULL code (00H) is stored at the start of (D).

• If the number of characters designated by (S2)+1 is "-1", stores the data up to the final character designated by (S) starting from the device designated by (D).



■MIDW

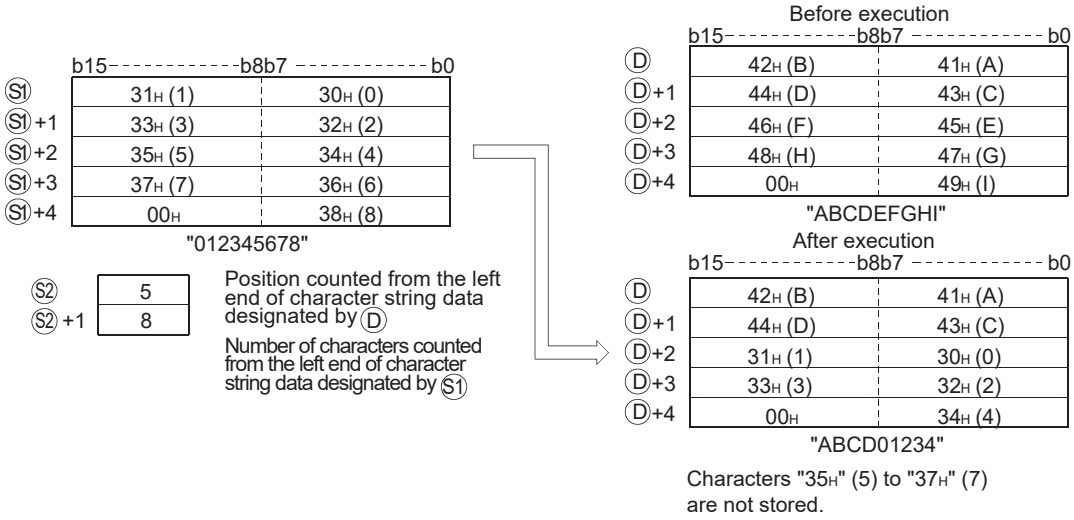
• Extracts the character string data of (S2)+1 characters, starting from the left end of the character string data designated by (S1), and stores the extracted data to the character string data designated by (D) in the area starting from the position designated by (S2) from the left end.



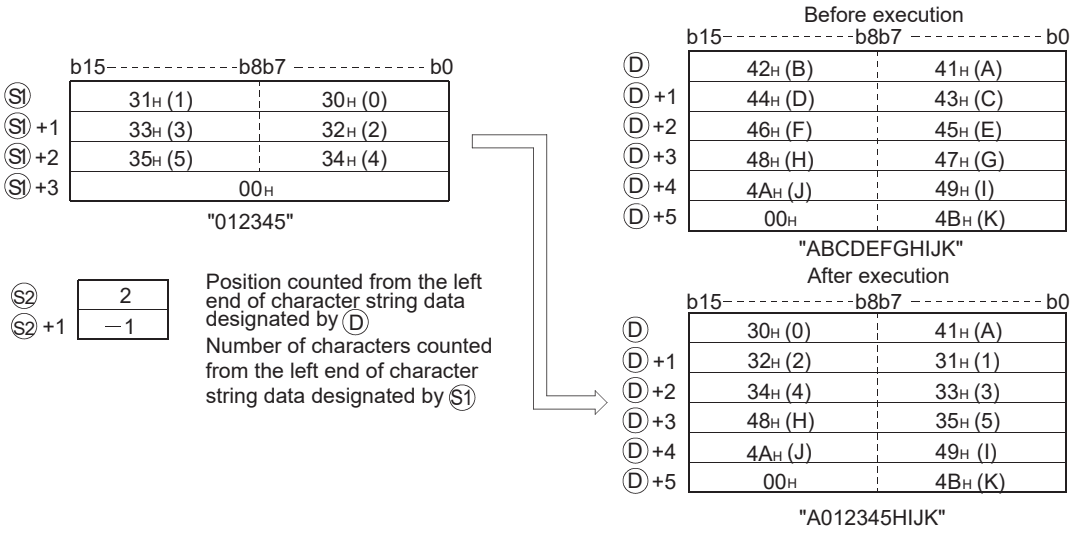
• The NULL code (00H) indicating the end of the character string is automatically added to the end of the character string. Refer to Page 92 Using character string data for the format of the character string data.

• If the number of characters designated by (S2)+1 is "0", the NULL code (00H) is stored at the start of (D).

• If the number of characters designated by (S2)+1 exceeds the final character from the character string data designated by (D), data will be stored up to the final character.



• If the number of characters designated by (S2)+1 is "-1", stores the data up to the final character designated by (S1) to the area starting from the device designated by (D).



Operation error

• In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

For MIDR instruction

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4101	The value of (S2) exceeds the number of characters specified by (S1). The amount of (S2) and (S2) +1 exceeds the number of characters of (S1). The (S2)+1 number of characters from position (D) exceeds the (D) device range. The (S2)+0 value is 0. The values of (S2)+1 are those other than the effective values (-1, 0, 1 or over). There is no NULL code "00H" within the range of the corresponding devices starting from the device number specified by (S1).	—	○	○	○	○	○

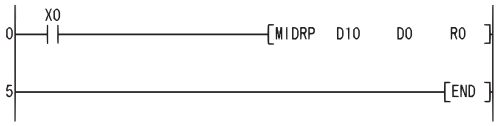
For MIDW instruction

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4101	The value of (S2) exceeds the number of characters specified by (D). The amount of (S2) and (S2) +1 exceeds the number of characters of (S1). The (S2)+1 value exceeds the number of characters for (S1). The (S2)+0 value is 0. The values of (S2)+1 are those other than the effective values (-1, 0, 1 or over). There is no NULL code "00H" within the range of the corresponding devices starting from the device number specified by (S1).	—	○	○	○	○	○

Program example

- The following program stores the 3rd character through the 6th character from the left of the character string stored in the area starting from D10 at devices starting from D0 when X0 is turned ON.

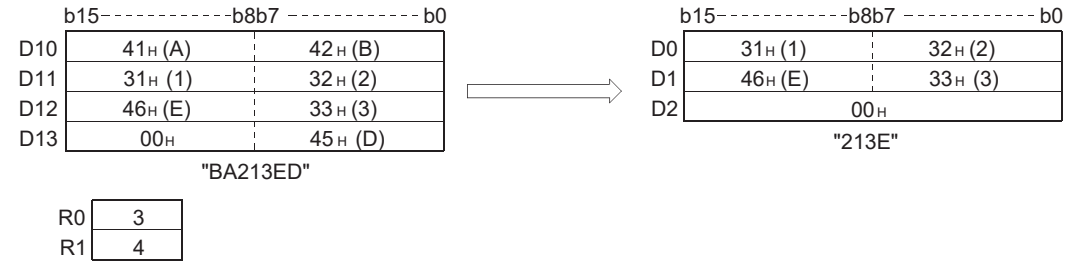
[Ladder Mode]



[List Mode]

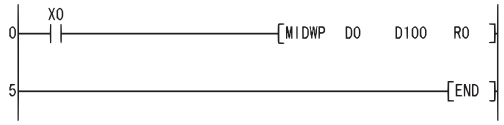
Step	Instruction	Device		
0	LD	X0		
1	MIDRP	D10	D0	R0
5	END			

[Operation]



- The following program stores 4 characters of the character string data stored in the area starting from D0 into the area starting from the 3rd character from the left of the character string data in the area starting from D100 when X0 is turned ON.

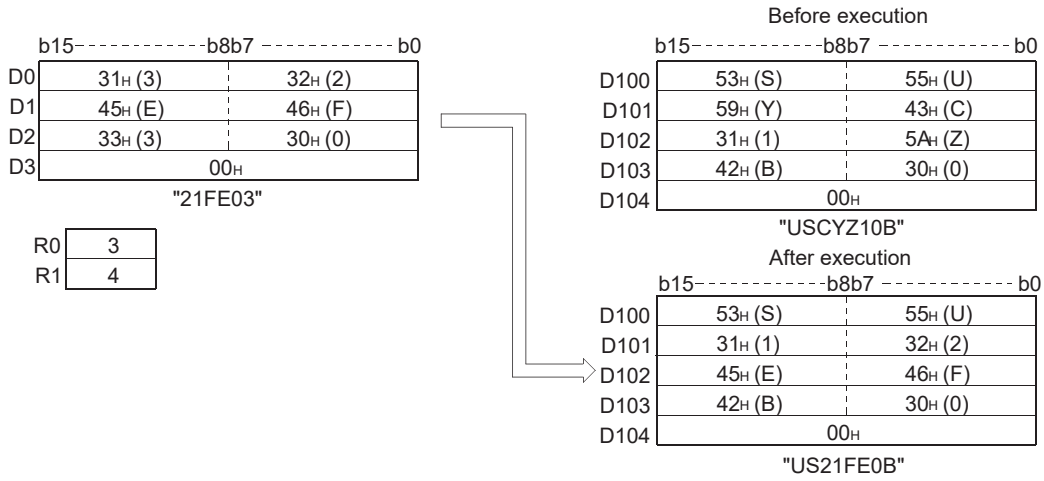
[Ladder Mode]



[List Mode]

Step	Instruction	Device		
0	LD	X0		
1	MIDWP	D0	D100	R0
5	END			

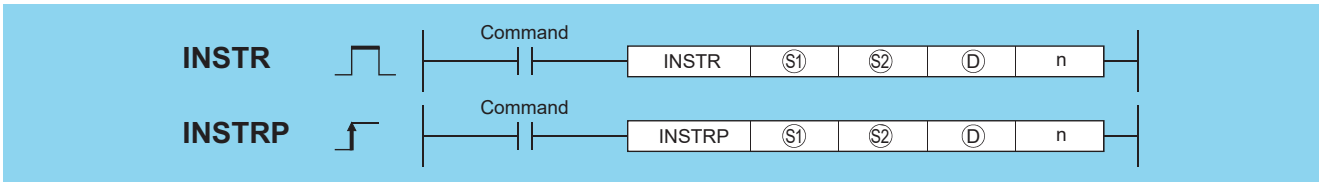
[Operation]



Character string search

INSTR(P)

Basic | High performance | Process | Redundant | Universal | LCPU



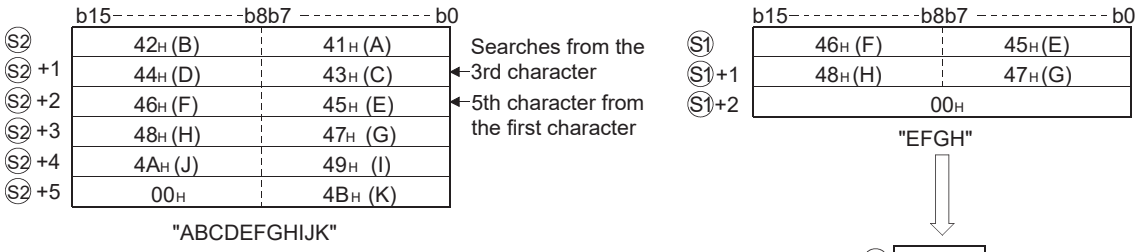
(S1): Character string to be searched or head number of the devices where the character string to be searched is stored (character string)
(S2): Character string in which a search is performed or head number of the devices where the character string is stored (character string)
(D): Head number of the devices where the result of search will be stored (BIN 16 bits)
n: Location to start the search (BIN 16 bits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant		Others
	Bit	Word		Bit	Word			K, H	$	
(S1)	—	○		—				—	○	—
(S2)	—	○		—				—	○	—
(D)	○	○		○				—	—	—
n	○	○		○				○	—	—

Processing details

- Searches for the character string data designated by (S1) in the area starting from the nth character from the left of the character string data designated by (S2) and stores the result of search at the device designated by (D). As the result of search, the location of match, counted in the number of characters from the first character of the character string data designated by (S2), is stored.

When n=3



- If there is no matching character string data, stores "0" at (D).

Operation error

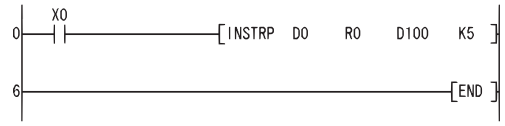
- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4100	The value of n exceeds the number of characters for (S2). The number of character specified by (S1) is 0. n is negative or 0.	—	○	○	○	○	○
4100	There is no NULL code "00H" within the range of the corresponding devices starting from the devices specified by (S1) and (S2).	—	○	○	○	—	—
4101	There is no NULL code "00H" within the range of the corresponding devices starting from the devices specified by (S1) and (S2).	—	—	—	—	○	○

Program example

- The following program searches from the 5th character from the left of the character string data stored in devices starting from R0 for the character string data in devices starting from D0, and stores the results at D100 when X0 goes ON.

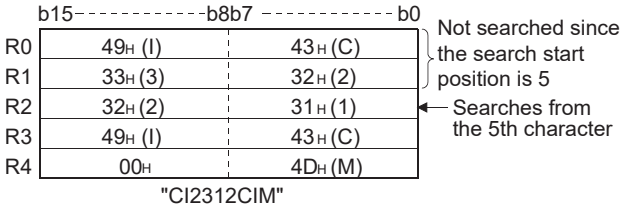
[Ladder Mode]

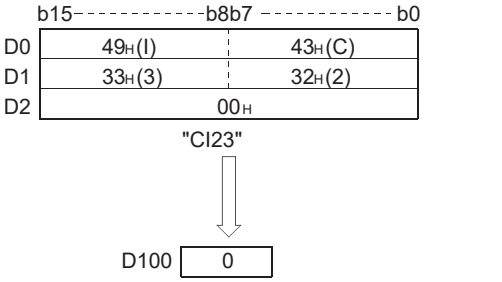


[List Mode]

Step	Instruction	Device			
0	LD	X0			
1	INSTRP	D0	R0	D100	K5
6	END				

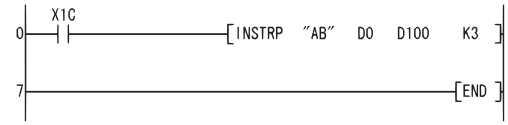
[Operation]





- The following program searches from the 3rd character from the left of the character string data being stored in devices starting from D0 for the character string data "AB", and stores the results of the search at D100 when X1C goes ON.

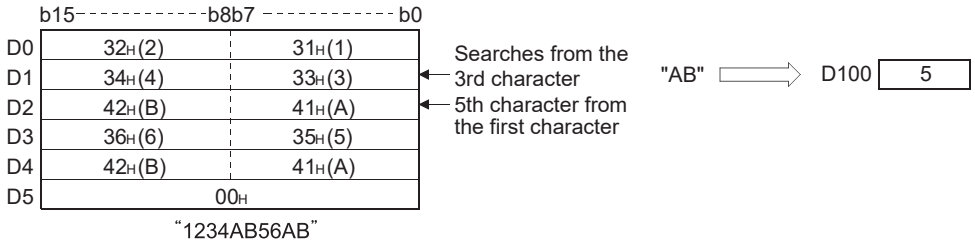
[Ladder Mode]



[List Mode]

Step	Instruction	Device			
0	LD	X1C			
1	INSTRP	"AB"	D0	D100	K3
7	END				

[Operation]

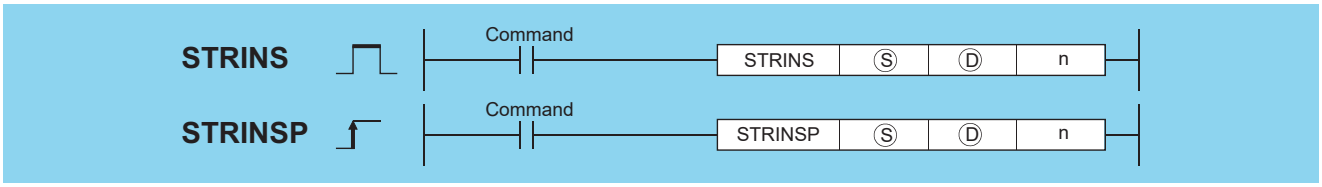


Insertion of character string

STRINS(P)

Basic | High performance | Process | Redundant | Ver. Universal | LCPU

- QnU(D)(H)CPU, QnUDE(H)CPU: the serial number (first five digits) is "10102" or later
- Q00UJCPU, Q00UCPU, Q01UCPU, QnUDVCPU, QnUDPVCPU: Supported



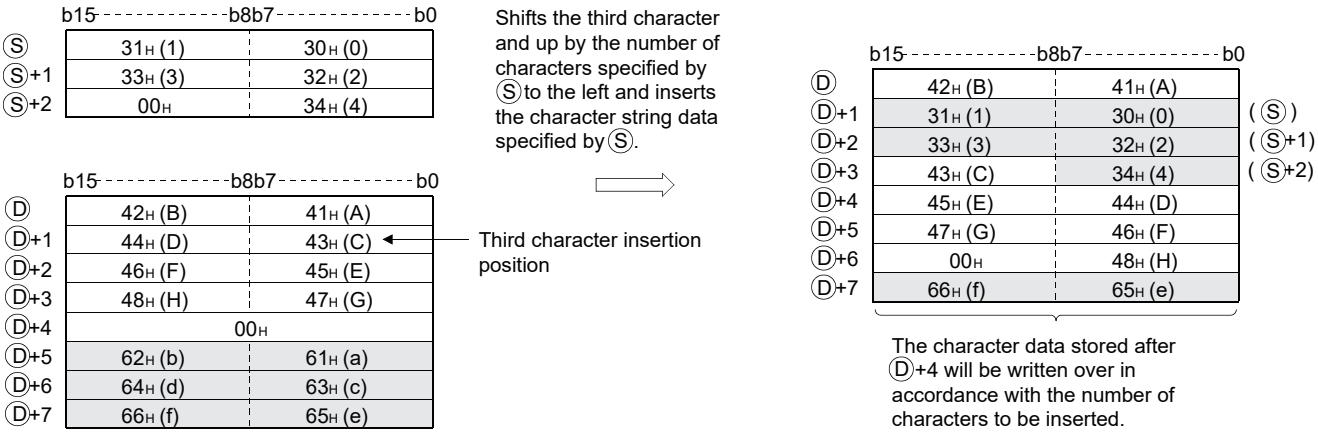
(S): Character string to be inserted or head number (character string) of the devices where insert character strings are stored
(D): Head number (character string) of the devices where insert character strings are stored
n: Insert position (Setting range: 1 ≤ n ≤ 16383) (BIN 16 bits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant		Others
	Bit	Word		Bit	Word			K, H	$	
(S)	—	○		—				—	○	—
(D)	—	○		—				—	—	—
n	○	○		○				○	—	—

Processing details

- This instruction inserts the character string data specified by (S) to the nth device (insert position) from the initial character string data stored in the devices specified by (D).

Insert position: n=3



- This instruction stores the NULL code (00H) into the device (1 word) that positions after the last device where the character string data are stored, if the character string ((S)+(D)) value is even after the insertion.
- This instruction stores the NULL code (00H) into the last device (high 8 bits) where the character string data are stored, if the character string ((S)+(D)) value is odd after the insertion.
- This instruction links the device, where the character string data are stored, specified by (S) with the last device specified by (D), if n is specified by the number of devices specified by (D) plus one.

Operation error

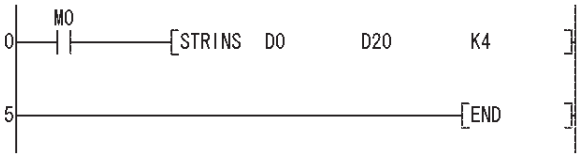
- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4100	The number of characters in the devices specified by (S), (D), or the devices specified by ((S)+(D)) after the insertion exceeds 16383 characters. The value specified in n is not within the specified range. (1≤n≤16383) The value specified in n exceeds the number of characters of the character string (D)+1.	—	—	—	—	○	○
4101	The devices, that store character strings, specified by (S) overlaps with even one of the devices specified by (D). The range of the devices specified by ((S)+(D)) in which character strings data have been inserted exceeds the specified device range. There is no NULL code "00H" within the range of the corresponding devices starting from the device numbers specified by (S) and (D). The device where the character has been inserted is the same as the device storing the character strings.	—	—	—	—	○	○

Program example

- The following program inserts the character string data stored in the device D0 and up to the fourth device from the initial character string data stored in D20 and up, when M0 is turned on.

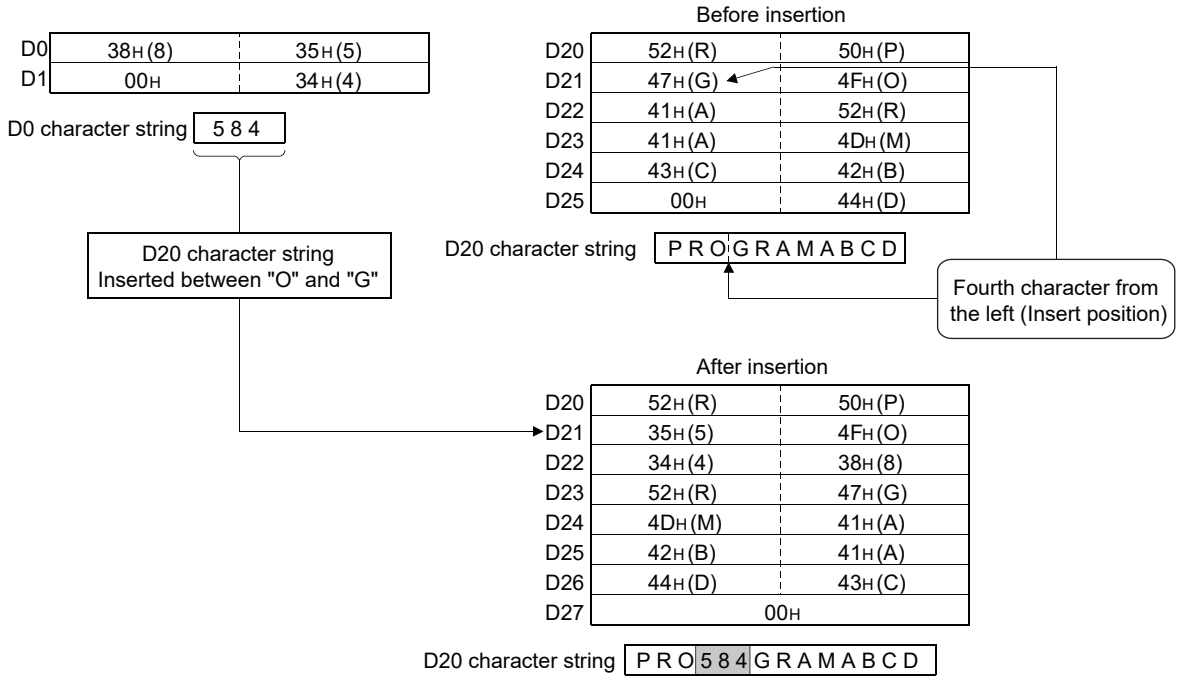
[Ladder Mode]



[List Mode]

Step	Instruction	Device		
0	LD	M0		
1	STRINS	D0	D20	K4
5	END			

[Operation]

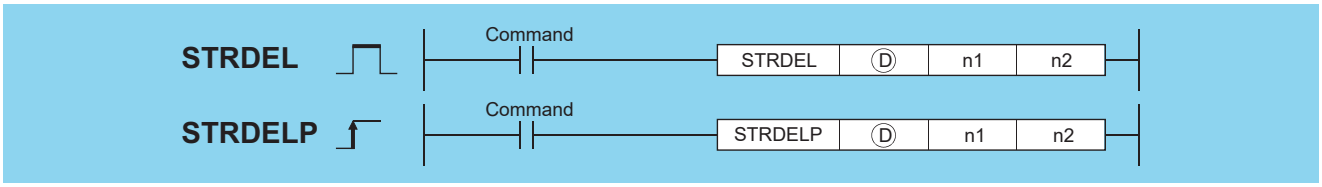


Deletion of character string

STRDEL(P)

Basic | High performance | Process | Redundant | Ver. Universal | LCPU

- QnU(D)(H)CPU, QnUDE(H)CPU: the serial number (first five digits) is "10102" or later
- Q00UJCPU, Q00UCPU, Q01UCPU, QnUDVCPU, QnUDPVCPU: Supported



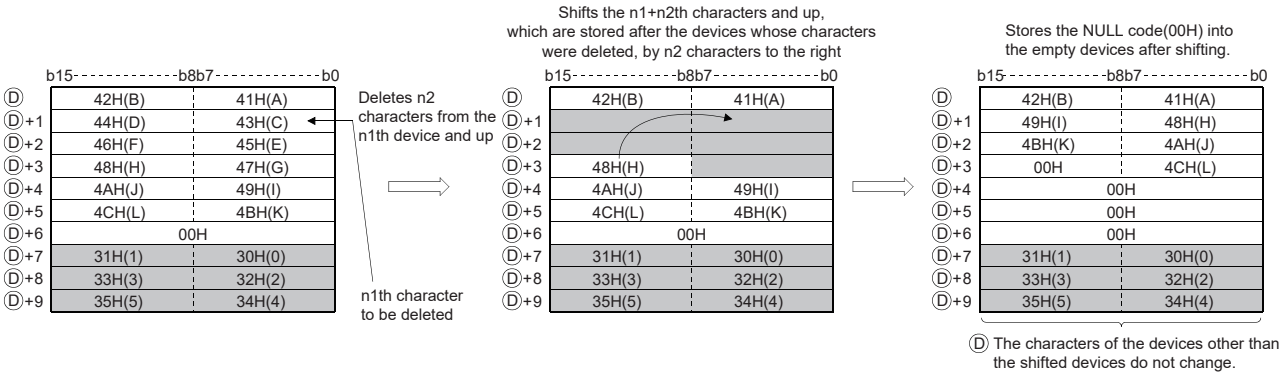
(D): Head number (character string) of the devices where character strings to be deleted are stored
n1: Deletion start position (Setting range 1 ≤ n1 ≤ 16383) (BIN 16 bits)
n2: Number of characters to be deleted (Setting range 1 ≤ n2 ≤ 16384-n1) (BIN 16 bits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(D)	—	○		—				—	—
n1	—	○		○				○	—
n2	○	○		○				○	—

Processing details

- This instruction deletes n2 characters data in the devices specified by (D) starting from the device (insert position) specified by n1.

Device position where character string data to be deleted: n1=3
Number of characters to be deleted: n2=5



- This instruction stores the NULL code (00H) into the device (one word) that positions after the last device that stores the character string data when the character string data specified by (D) is even, after the characters are deleted.
- This instruction stores the NULL code (00H) into the last device (high 8 bits) that stores the character string data when the character string data specified by (D) is odd, after the characters are deleted.
- This instruction shifts the characters stored in the devices that position after the deleted devices by n2 characters to the right, and then stores the NULL code (00H) into the empty device.

Operation error

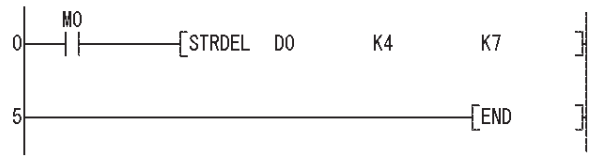
- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4100	The number of characters in the devices specified by (D) exceeds 16383. The value specified by n1 is not within the range. (1≤n1≤16383) The value specified by n1 exceeds the number of characters in the devices specified by (D). The value specified in n2 exceeds the number of characters between n1 and the last character in (D). The value specified in n2 is negative.	—	—	—	—	○	○

Program example

- The following program deletes the fourth to the seventh characters in the character string data stored in the devices D0 and up, when M0 is turned on.

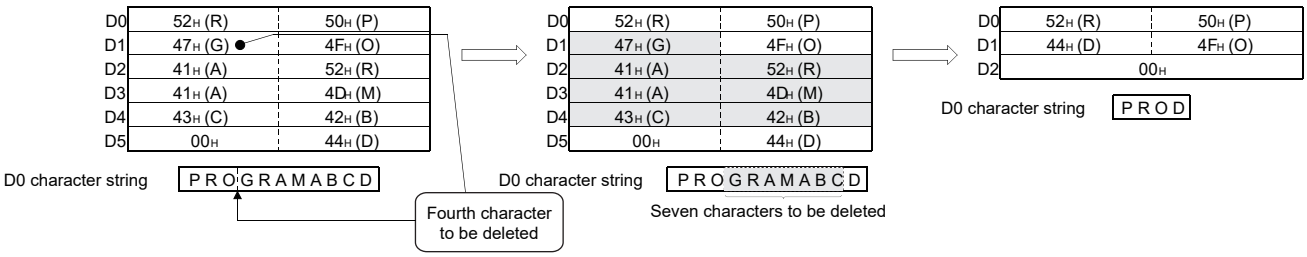
[Ladder Mode]



[List Mode]

Step	Instruction	Device		
0	LD	M0		
1	STRDEL	D0	K4	K7
5	END			

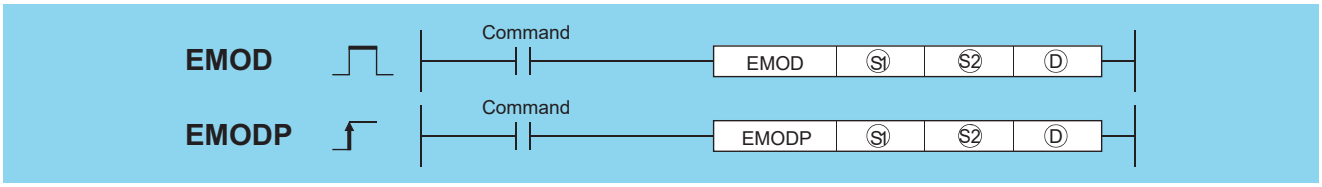
[Operation]



Floating-point data to BCD

EMOD(P)

Basic | High performance | Process | Redundant | Universal | LCPU



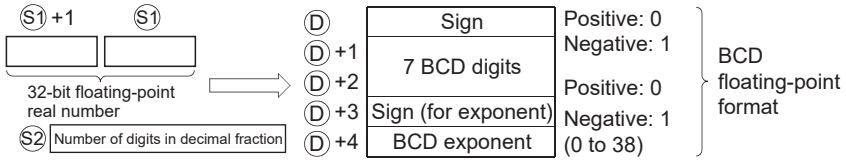
(S1): 32-bit floating decimal point real number data or head number of the devices where the floating decimal point real number data is stored (real number)
(S2): Decimal fraction digits data (BIN 16 bits)
(D): Head number of the devices where the data after break down into BCD will be stored (BIN 16 bits)

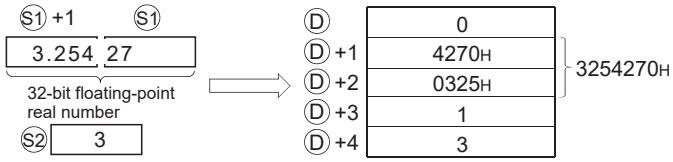
Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant		Others
	Bit	Word		Bit	Word			K, H	E	
(S1)	—	○		—	○		○*1	—	○	—
(S2)	○	○		○	○		○	○	—	—
(D)	—	○		—	—		—	—	—	—

*1 Applicable for the Universal model QCPU, LCPU.

Processing details

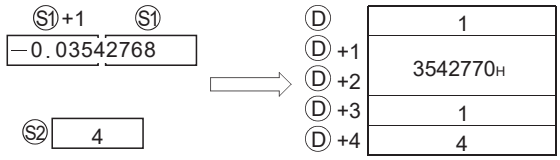
- Dissociate the 32-bit floating decimal point data designated by (S1) into BCD type floating point format based on the decimal fraction digits specified by (S2), and stores the result into the area starting from the device designated by (D).

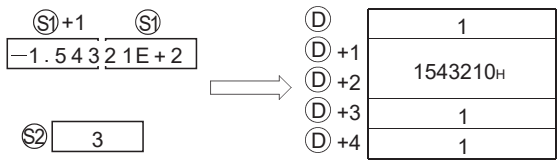




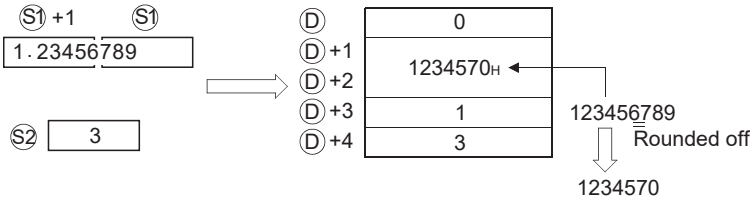
- (S2) specifies the decimal fraction digits of the 32-bit floating decimal point real number data of (S1). In the example above, a decimal fraction digit is designated as shown below:

3.25427
(S2)=3





• The 7th digit of the significant digits being stored at (D)+1 and (D)+2 is rounded off to make a 6-digit number.



• When an input value is set using a programming tool, a rounding error may occur. For precautions, refer to Page 90 Precautions.

Operation error

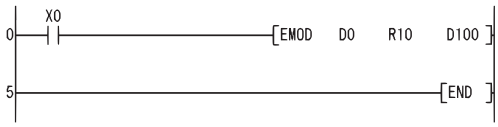
• In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4100	The decimal fraction digit specified by (S2) is not within the range between 0 and 7. The 32-bit floating point real number specified by (S1) is not within the following range: 0, $2^{-126} \leq \lvert \text{Device} \rvert < 2^{128}$	—	○	○	○	○	○
4101	The range of the device specified by (D) exceeds that of the corresponding device.	—	○	○	○	○	○
	The range of the device specified by (D) exceeds that of the corresponding device.	—	—	—	—	○	○
4140	The specified device value is -0, unnormalized number, nonnumeric, and ±∞.	—	—	—	—	○	○

Program example

• The following program breaks down the 32-bit floating decimal point type real number data stored at D0 and D1 into BCD according to the decimal fraction digits as designated by R10, and stores the results into the area starting from D100 when X0 is turned ON.

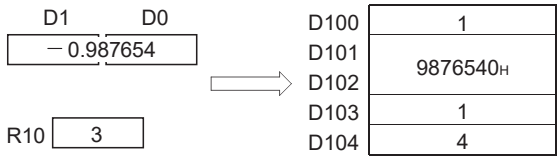
[Ladder Mode]



[List Mode]

Step	Instruction	Device		
0	LD	X0		
1	EMOD	D0	R10	D100
5	END			

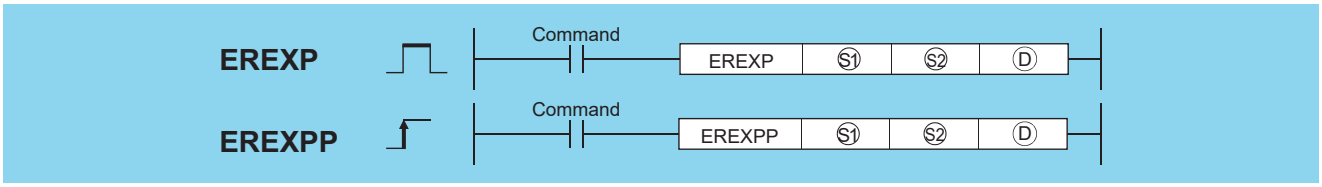
[Operation]



From BCD format data to floating-point data

EREXP(P)

Basic | High performance | Process | Redundant | Universal | LCPU



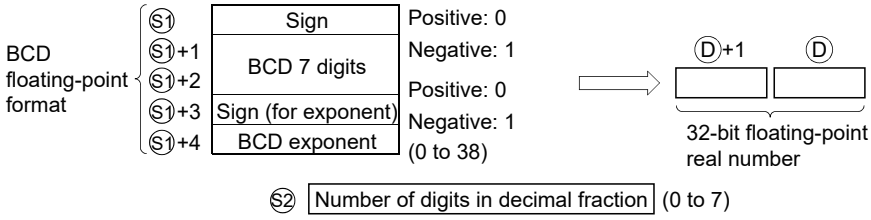
(S1): Head number of the devices where BCD type floating point format data is stored (BIN 16 bits)
(S2): Decimal fraction digits data (BIN 16 bits)
(D): The device where the converted 32-bit floating point real number data will be stored (real number)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(S1)	—	○		—	—		—	—	—
(S2)	○	○		○	○		○	○	—
(D)	—	○		—	○		○*1	—	—

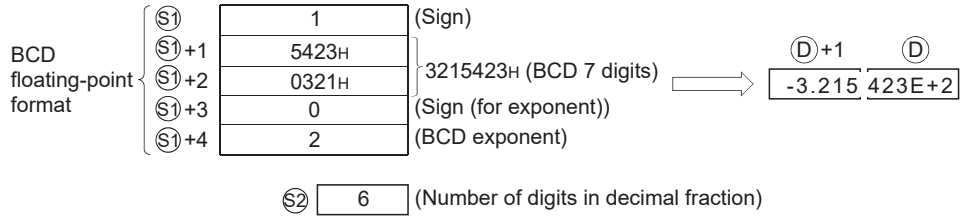
*1 Applicable for the Universal model QCPU, LCPU.

Processing details

- Converts the BCD type floating point data designated by (S1) to the 32-bit floating decimal point real number data according to the decimal fraction digits specified by (S2), and stores the result into the area starting from the device designated by (D).



- The sign at (S1) and the sign for the exponent at (S1) +3 is set at 0 for a positive value and at 1 for a negative value.
- 0 to 38 can be set for the BCD exponent of (S1)+4.
- 0 to 7 can be set for the decimal fraction digits of (S2).



Operation error

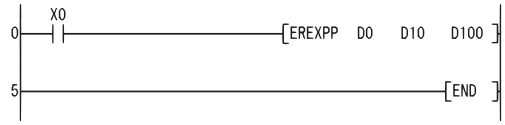
- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4100	The data format in the device specified by (S1) is not 0 or 1. A value other than 0 to 9 exists in the each digit of (S1)+1 and (S1)+2. The data format in the device specified by (S1)+3 is not 0 or 1. The exponent data in the device specified by (S1)+4 is not within the range from 0 to 38. The decimal fraction digit specified by (S2) is not within the range between 0 and 7.	—	○	○	○	○	○
4101	The range of the device specified by (S1) exceeds that of the corresponding device.	—	—	—	—	○	○

Program example

- The following program converts the BCD type floating decimal point format data being stored in devices starting from D0 to 32-bit floating decimal point type real number data based on the decimal fraction digit being stored at D10, and stores the result at D100 and D101 when X0 goes ON.

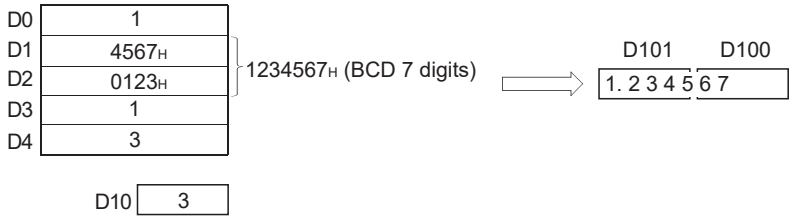
[Ladder Mode]



[List Mode]

Step	Instruction	Device		
0	LD	X0		
1	EREXPP	D0	D10	D100
5	END			

[Operation]



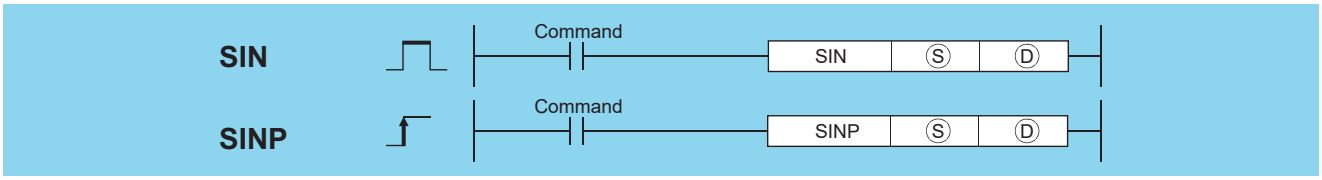
7.12 Special Function Instructions

SIN operation on floating-point data (single precision)

SIN(P)

Ver. Basic | High performance | Process | Redundant | Universal | LCPU

• Basic model QCPU: The serial number (first five digits) is "04122" or later.



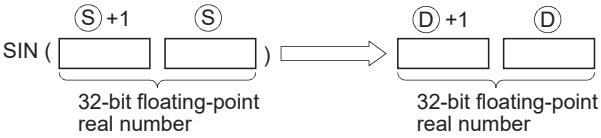
(S): Angle data of which the SIN (sine) value is obtained or head number of the devices where the angle data is stored (real number)
(D): Head number of the devices where the operation result will be stored (real number)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant E	Others
	Bit	Word		Bit	Word				
(S)	—	○		—	○		○*1	○	—
(D)	—	○		—	○		○*1	—	—

*1 Applicable for the Universal model QCPU, LCPU.

Processing details

• Returns the SIN (sine) value of the angle designated at (S) and stores the operation result in the device number designated at (D).



• Angles designated at (S) are set in radian units (degrees × π / 180). For conversion between degrees and radian values, see the RAD and DEG instructions.

• When an input value is set using a programming tool, a rounding error may occur. For precautions, refer to Page 90 Precautions.

Operation error

• In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

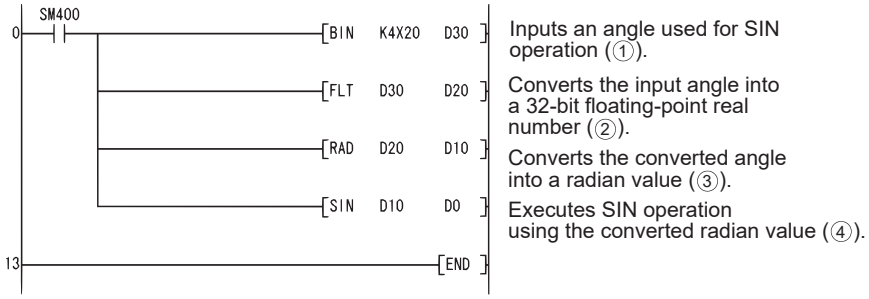
Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4100	The specified device value is -0.*2	○	○	○	○	—	—
	An error occurred during the operation.	—	—	○	○	—	—
4140	The specified device value is -0, unnormalized number, nonnumeric, and ±∞.	—	—	—	—	○	○
4141	The operation result exceeds the following range. (when an overflow occurs) |Operation result| < 2^{128}	—	—	—	—	○	○
	An error occurred during operation.						

*2 There are CPU modules that will not result in an operation error if -0 is specified. For details, refer to Page 88 Using single/double-precision real number data.

Program example

- The following program conducts a SIN operation on the angles stored in the four BCD digits from X20 to X2F and stores the results at D0 and D1 as 32-bit floating decimal point type real numbers.

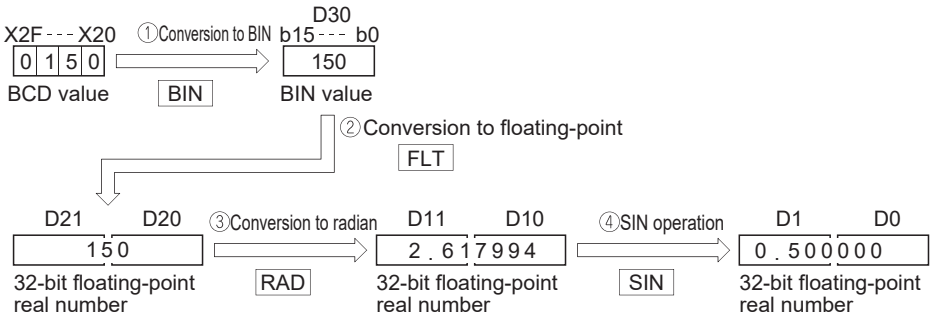
[Ladder Mode]



[List Mode]

Step	Instruction	Device	
0	LD	SM400	
1	BIN	K4X20	D30
4	FLT	D30	D20
7	RAD	D20	D10
10	SIN	D10	D0
13	END		

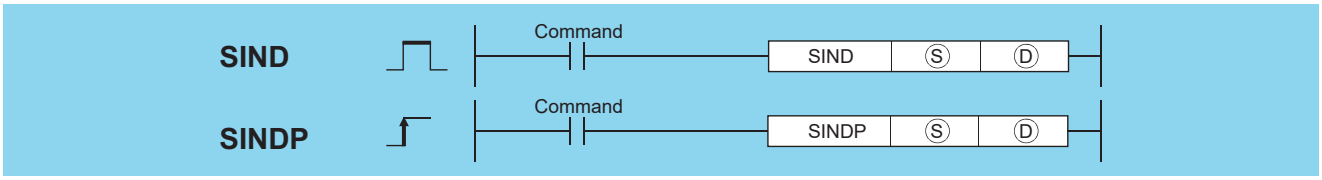
[Operations involved when X20 to X2F designate a value of 150]



SIN operation on floating-point data (double precision)

SIND(P)

Basic | High performance | Process | Redundant | Universal | LCPU

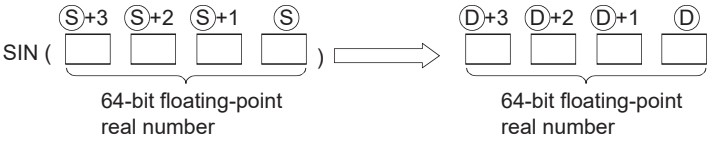


(S): Angle data of which the SIN (sine) value is obtained or head number of the devices where the angle data is stored (real number)
(D): Head number of the devices where the operation result will be stored (real number)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant E	Others
	Bit	Word		Bit	Word				
(S)	—	○		—				○	—
(D)	—	○		—				—	—

Processing details

- The SIN (sine) value of the angle specified by (S) is calculated and its result is stored into the device specified by (D).



- Angles designated at (S) are set in radian units (degrees × π / 180). For conversion between degrees and radian values, see the RADD and DEGD instructions.
- When the operation results in -0 or an underflow, the result is processed as 0.
- When an input value is set using a programming tool, a rounding error may occur. For precautions, refer to Page 90 Precautions.

Operation error

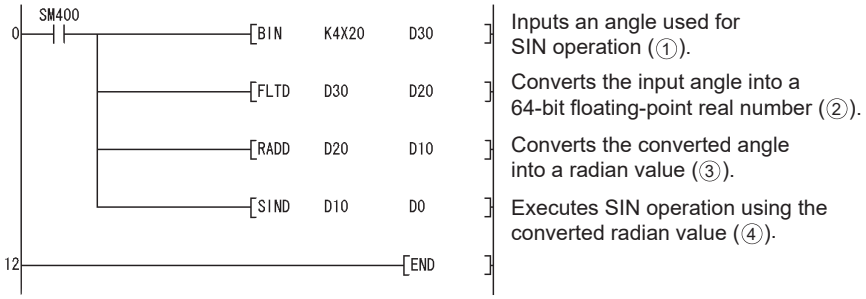
- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4140	The specified device value is not within the following range: 0, $2^{-1022} \leq$ | Specified device value | $< 2^{1024}$ The value of the specified device is -0, a subnormal number, NaN (not a number), or ±∞.	—	—	—	—	○	○
4141	The operation result exceeds the following range. (when an overflow occurs) |Operation result| $< 2^{1024}$	—	—	—	—	○	○
	An error occurred during operation.						

Program example

- The following program conducts a SIN operation on the angles stored in the four BCD digits from X20 to X2F and stores the results at D0 to D3 as 64-bit floating decimal point type real numbers.

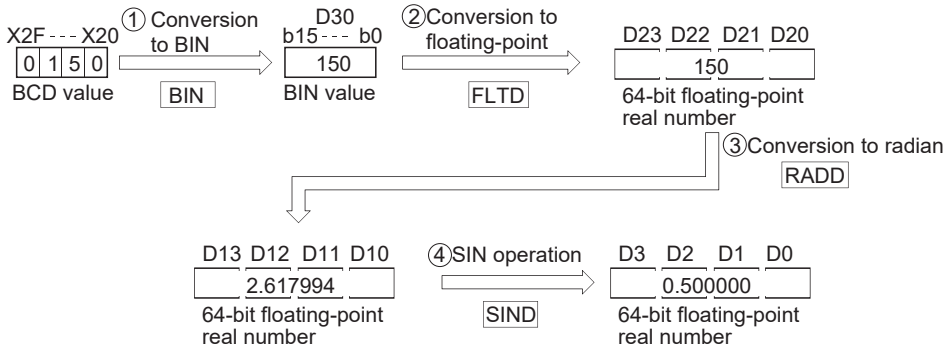
[Ladder Mode]



[List Mode]

Step	Instruction	Device	
0	LD	SM400	
1	BIN	K4X20	D30
3	FLTD	D30	D20
6	RADD	D20	D10
9	SIND	D10	D0
12	END		

[Operations involved when X20 to X2F designate a value of 150]

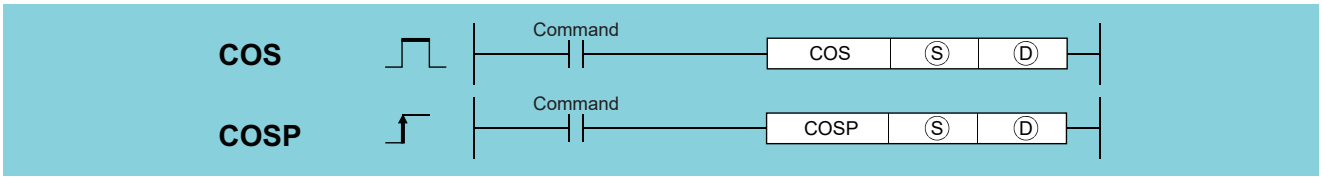


COS operation on floating-point data (single precision)

COS(P)

Ver. Basic | High performance | Process | Redundant | Universal | LCPU

• Basic model QCPU: The serial number (first five digits) is "04122" or later.



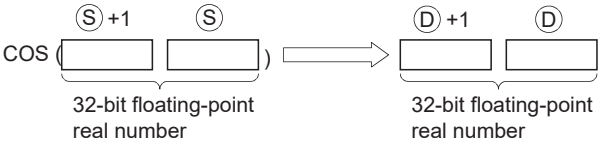
(S): Angle data of which the COS (cosine) value is obtained or head number of the devices where the angle data is stored (real number)
(D): Head number of the devices where the operation result will be stored (real number)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant E	Others
	Bit	Word		Bit	Word				
(S)	—	○		—	○		○*1	○	—
(D)	—	○		—	○		○*1	—	—

*1 Applicable for the Universal model QCPU, LCPU.

Processing details

• Returns the COS (cosine) value of the angle designated by (S) and stores operation result at device number designated by (D).



• Angles designated at (S) are set in radian units (degrees × π ÷ 180). For conversion between degrees and radian values, see the RAD and DEG instructions.

• When an input value is set using a programming tool, a rounding error may occur. For precautions, refer to Page 90 Precautions.

Operation error

• In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

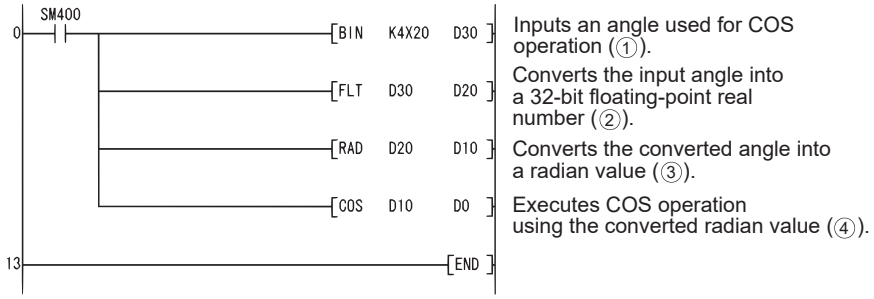
Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4100	The specified device value is -0.*2	○	○	○	○	—	—
	An error occurred during the operation.	—	—	○	○	—	—
4140	The specified device value is -0, unnormalized number, nonnumeric, and ±∞.	—	—	—	—	○	○
4141	The operation result exceeds the following range. (when an overflow occurs) |Operation result| < 2^{128}	—	—	—	—	○	○
	An error occurred during operation.						

*2 There are CPU modules that will not result in an operation error if -0 is specified. For details, refer to Page 88 Using single/double-precision real number data.

Program example

- The following program performs a COS operation on the angle data designated by the 4 BCD digits from X20 to X2F, and stores results as 32-bit floating decimal point type real numbers at D0 and D1.

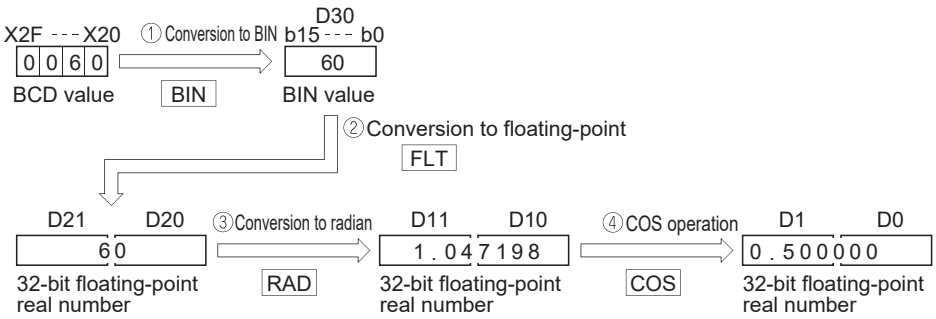
[Ladder Mode]



[List Mode]

Step	Instruction	Device	
0	LD	SM400	
1	BIN	K4X20	D30
4	FLT	D30	D20
7	RAD	D20	D10
10	COS	D10	D0
13	END		

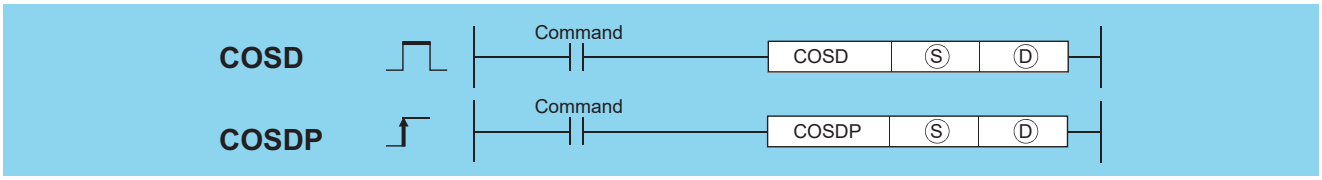
[Operations involved when X20 to X2F designate a value of 60]



COS operation on floating-point data (double precision)

COSD(P)

Basic | High performance | Process | Redundant | Universal | LCPU

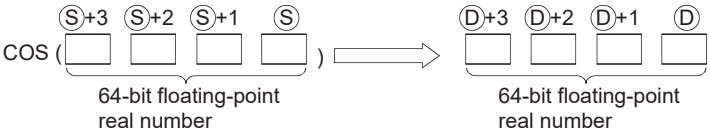


(S): Angle data of which the COS (cosine) value is obtained or head number of the devices where the angle data is stored (real number)
(D): Head number of the devices where the operation result will be stored (real number)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant E	Others
	Bit	Word		Bit	Word				
(S)	—	○		—				○	—
(D)	—	○		—				—	—

Processing details

- The COS (cosine) value of the angle specified by (S) is calculated and its result is stored into the device specified by (D).



- Angles designated at (S) are set in radian units (degrees × π / 180). For conversion between degrees and radian values, see the RADD and DEGD instructions.
- When the operation results in -0 or an underflow, the result is processed as 0.
- When an input value is set using a programming tool, a rounding error may occur. For precautions, refer to Page 90 Precautions.

Operation error

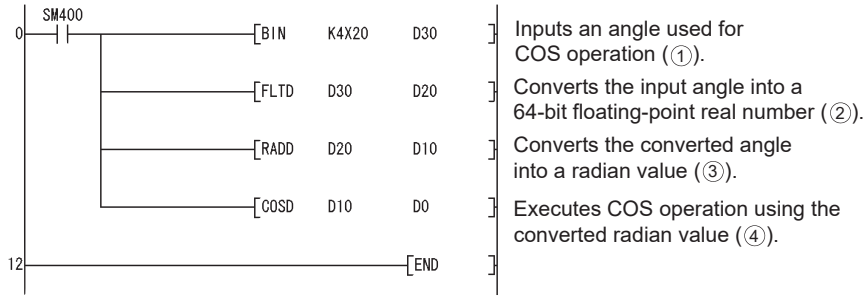
- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4140	The specified device value is not within the following range: 0, $2^{-1022} \leq$ | Specified device value | $< 2^{1024}$ The value of the specified device is -0, a subnormal number, NaN (not a number), or ±∞.	—	—	—	—	○	○
4141	The operation result exceeds the following range. (when an overflow occurs) |Operation result| $< 2^{1024}$	—	—	—	—	○	○
	An error occurred during operation.						

Program example

- The following program performs a COS operation on the angle data designated by the 4 BCD digits from X20 to X2F, and stores results as 64-bit floating decimal point type real numbers at D0 to D3.

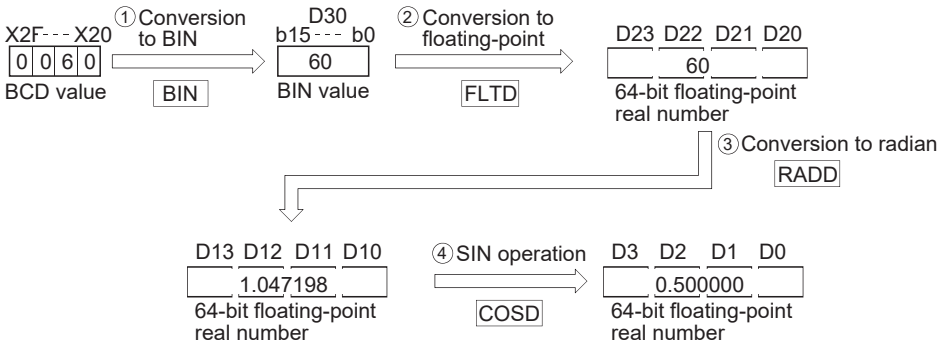
[Ladder Mode]



[List Mode]

Step	Instruction	Device	
0	LD	SM400	
1	BIN	K4X20	D30
3	FLTD	D30	D20
6	RADD	D20	D10
9	COSD	D10	D0
12	END		

[Operations involved when X20 to X2F designate a value of 60]

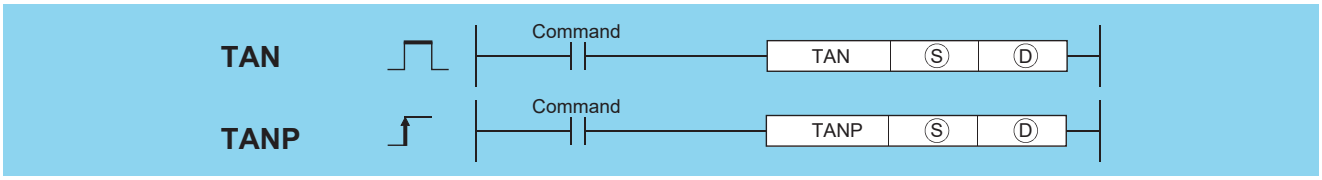


TAN operation on floating-point data (single precision)

TAN(P)

Ver. Basic | High performance | Process | Redundant | Universal | LCPU

• Basic model QCPU: The serial number (first five digits) is "04122" or later.



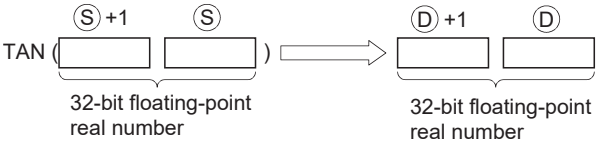
(S): Angle data of which the TAN (tangent) value is obtained or head number of the devices where the angle data is stored (real number)
(D): Head number of the devices where the operation result will be stored (real number)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant E	Others
	Bit	Word		Bit	Word				
(S)	—	○		—	○		○*1	○	—
(D)	—	○		—	○		○*1	—	—

*1 Applicable for the Universal model QCPU, LCPU.

Processing details

• Returns the tangent (TAN) value of the angle data designated by (S), and stores operation result in device designated by (D).



• Angles designated at (S) are set in radian units (degrees × π / 180). For conversion between degrees and radian values, see the RAD and DEG instructions.
• When angles designated by (S) are π/2 radians, or (3/2)π radians, an operation error will be generated in the calculation of the radian value, so care must be taken to avoid such errors.
• When an input value is set using a programming tool, a rounding error may occur. For precautions, refer to Page 90 Precautions.

Operation error

• In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

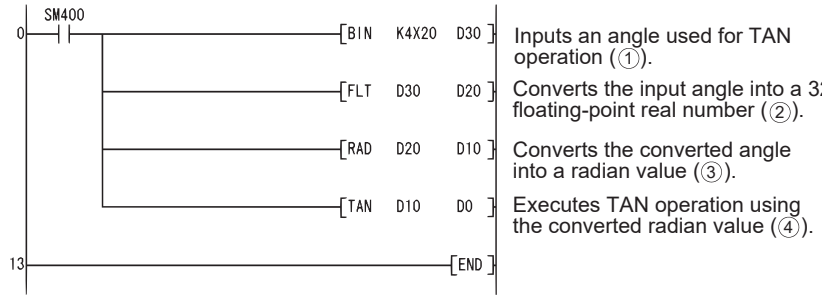
Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4100	The specified device value is not within the following range: 0, $2^{-126} \leq$ | Specified device value | $< 2^{128}$ The specified device value is -0.*2	○	○	○	○	—	—
	An error occurred during the operation.	—	—	○	○	—	—
4140	The specified device value is -0, unnormalized number, nonnumeric, and ±∞.	—	—	—	—	○	○
4141	The operation result exceeds the following range. (when an overflow occurs) |Operation result| $< 2^{128}$	—	—	—	—	○	○
	An error occurred during operation.						

*2 There are CPU modules that will not result in an operation error if -0 is specified. For details, refer to Page 88 Using single/double-precision real number data.

Program example

- The following program performs a TAN operation on the angle data set by the 4 BCD digits from X20 to X2F, and stores the results as 32-bit floating decimal point type real numbers at D0 and D1.

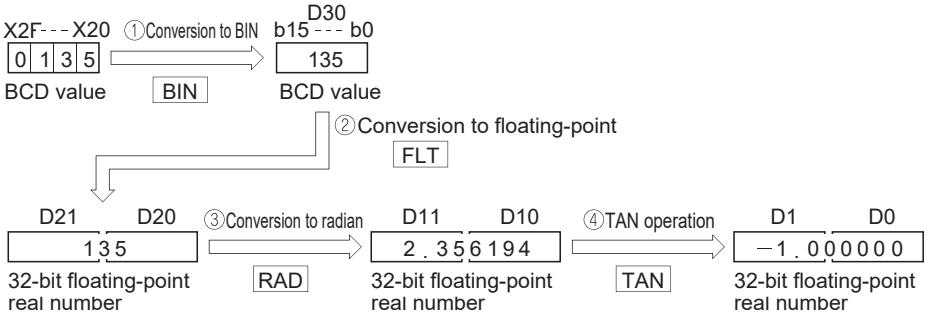
[Ladder Mode]



[List Mode]

Step	Instruction	Device	
0	LD	SM400	
1	BIN	K4X20	D30
4	FLT	D30	D20
7	RAD	D20	D10
10	TAN	D10	D0
13	END		

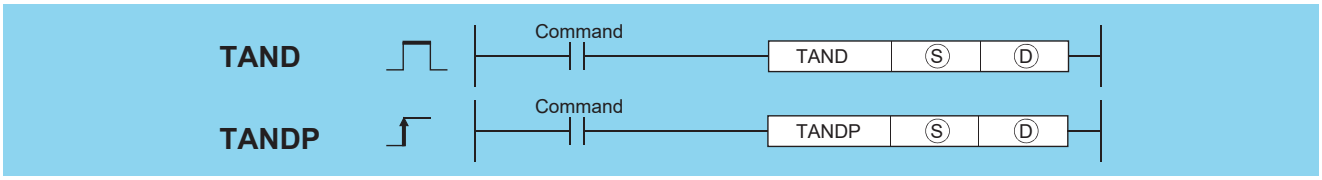
[Operations involved when X20 to X2F designate a value of 135]



TAN operation on floating-point data (double precision)

TAND(P)

Basic | High performance | Process | Redundant | Universal | LCPU

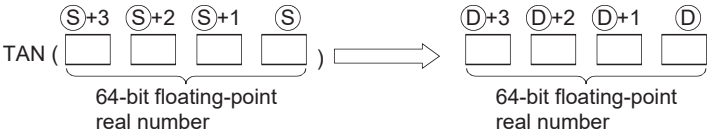


(S): Angle data of which the TAN (tangent) value is obtained or head number of the devices where the angle data is stored (real number)
(D): Head number of the devices where the operation result will be stored (real number)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant E	Others
	Bit	Word		Bit	Word				
(S)	—	○		—				○	—
(D)	—	○		—				—	—

Processing details

- The TAN (tangent) value of the angle specified by (S) is calculated and its result is stored into the device specified by (D).



- Angles designated at (S) are set in radian units (degrees × π ÷ 180). For conversion between degrees and radian values, see the RADD and DEGD instructions.
- When angles designated by (S) are π/2 radians, or (3/2)π radians, an operation error will be generated in the calculation of the radian value, so care must be taken to avoid such errors.
- When the operation results in -0 or an underflow, the result is processed as 0.
- When an input value is set using a programming tool, a rounding error may occur. For precautions, refer to Page 90 Precautions.

Operation error

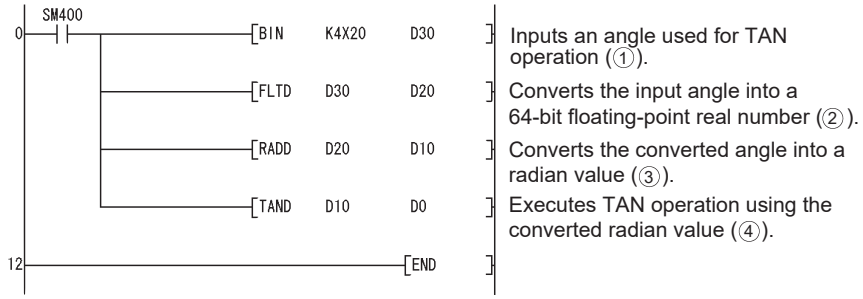
- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4140	The specified device value is not within the following range: 0, $2^{-1022} \leq$ | Specified device value | $< 2^{1024}$ The value of the specified device is -0, a subnormal number, NaN (not a number), or ±∞.	—	—	—	—	○	○
4141	The operation result exceeds the following range. (when an overflow occurs) |Operation result| $< 2^{1024}$	—	—	—	—	○	○
	An error occurred during operation.						

Program example

- The following program performs a TAN operation on the angle data set by the 4 BCD digits from X20 to X2F, and stores the results as 64-bit floating decimal point type real numbers at D0 to D3.

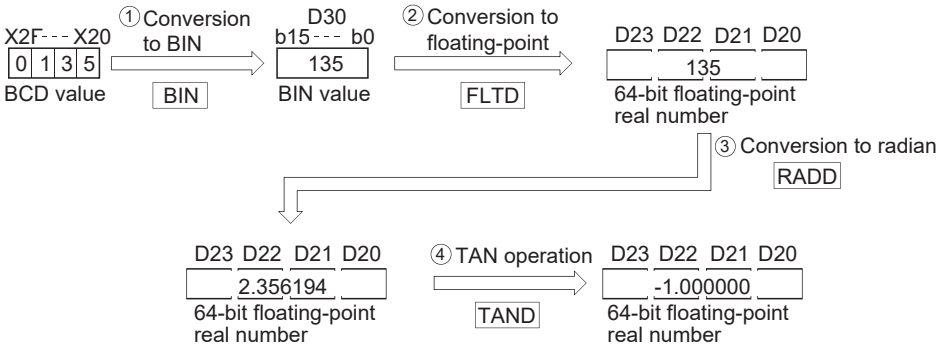
[Ladder Mode]



[List Mode]

Step	Instruction	Device	
0	LD	SM400	
1	BIN	K4X20	D30
3	FLTD	D30	D20
6	RADD	D20	D10
9	TAND	D10	D0
12	END		

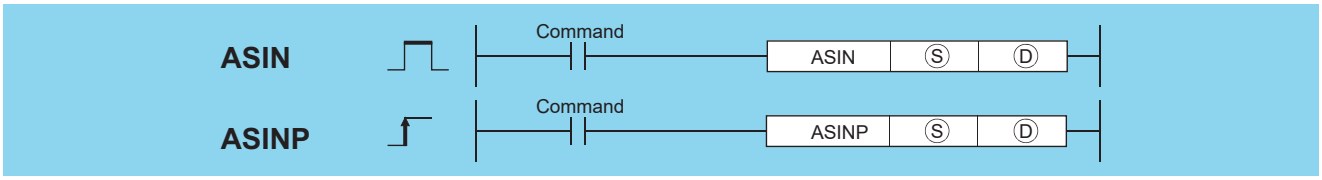
[Operations involved when X20 to X2F designate a value of 135]



Arc sine operation on floating-point data (single precision)

ASIN(P)

Basic | High performance | Process | Redundant | Universal | LCPU



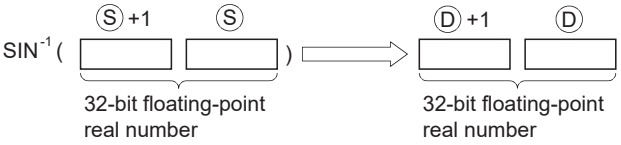
(S): SIN value of which the SIN^{-1} (inverse sine) value is obtained or head number of the devices where the SIN value is stored (real number)
(D): Head number of the devices where the operation result will be stored (real number)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant E	Others
	Bit	Word		Bit	Word				
(S)	—	○		—	○		○*1	○	—
(D)	—	○		—	○		○*1	—	—

*1 Applicable for the Universal model QCPU, LCPU.

Processing details

- Returns the SIN^{-1} angle of the SIN value designated by (S), and stores operation results at word device designated by (D).



- The SIN value designated by (S) can be in the range from -1.0 to 1.0.
- The angle (operation result) stored at (D) is stored in radian units. For more information on the conversion between radian and angle data, see description of RAD and DEG instructions.
- When an input value is set using a programming tool, a rounding error may occur. For precautions, refer to Page 90 Precautions.

Operation error

- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

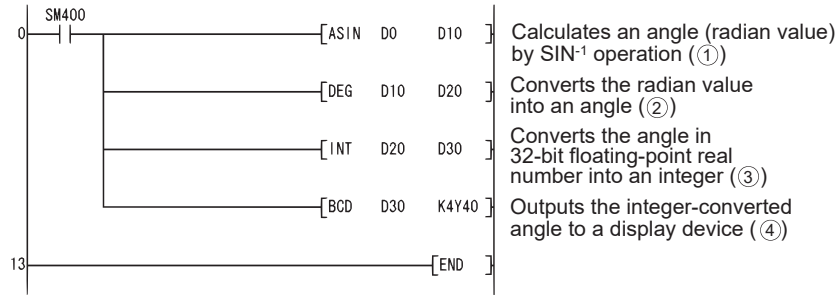
Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4100	The value specified by (S) is not within the range between -1.0 and 1.0.	—	○	○	○	○	○
	The specified device value is -0.*2	—	○	○	○	—	—
4140	The specified device value is not within the following range: 0, $2^{-126} \leq$ | Specified device value | $< 2^{128}$ The specified device value is -0, unnormalized number, nonnumeric, and ±∞.	—	—	—	—	○	○
4141	The operation result exceeds the following range. (when an overflow occurs) |Operation result| $< 2^{128}$	—	—	—	—	○	○

*2 There are CPU modules that will not result in an operation error if -0 is specified. For details, refer to Page 88 Using single/double-precision real number data.

Program example

- The following program seeks the inverse sine of the 32-bit floating decimal point real number at D0 and D1, and outputs the angle to the 4 BCD digits at Y40 to Y4F.

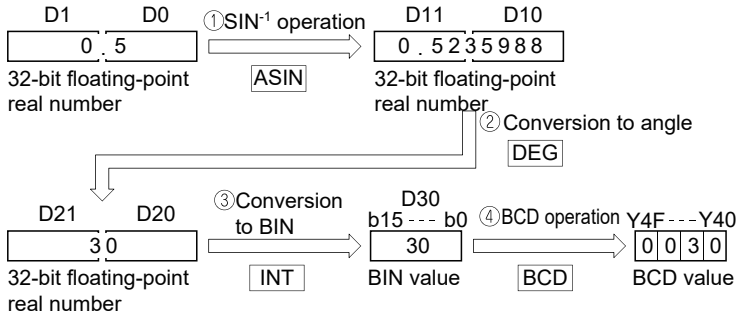
[Ladder Mode]



[List Mode]

Step	Instruction	Device	
0	LD	SM400	
1	ASIN	D0	D10
4	DEG	D10	D20
7	INT	D20	D30
10	BCD	D30	K4Y40
13	END		

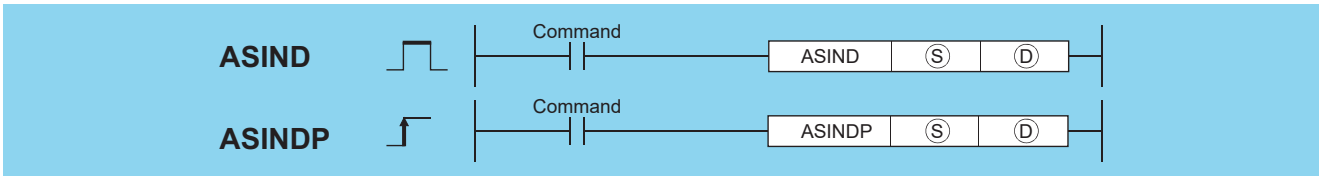
[Operations involved when D0 and D1 value is 0.5]



Arc sine operation on floating-point data (double precision)

ASIND(P)

Basic | High performance | Process | Redundant | Universal | LCPU

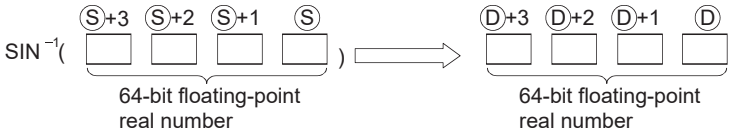


(S): SIN value of which the SIN^{-1} (inverse sine) value is obtained or head number of the devices where the SIN value is stored (real number)
(D): Head number of the devices where the operation result will be stored (real number)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant E	Others
	Bit	Word		Bit	Word				
(S)	—	○		—				○	—
(D)	—	○		—				—	—

Processing details

- The angle is calculated from the SIN (sine) value specified by (S) is and its result is stored into the device specified by (D).



- The SIN value designated by (S) can be in the range from -1.0 to 1.0.
- The angle (operation result) stored at (D) is stored in radian units. For more information on the conversion between radian and angle data, see description of RADD and DEGD instructions.
- When the operation results in -0 or an underflow, the result is processed as 0.
- When an input value is set using a programming tool, a rounding error may occur. For precautions, refer to Page 90 Precautions.

Operation error

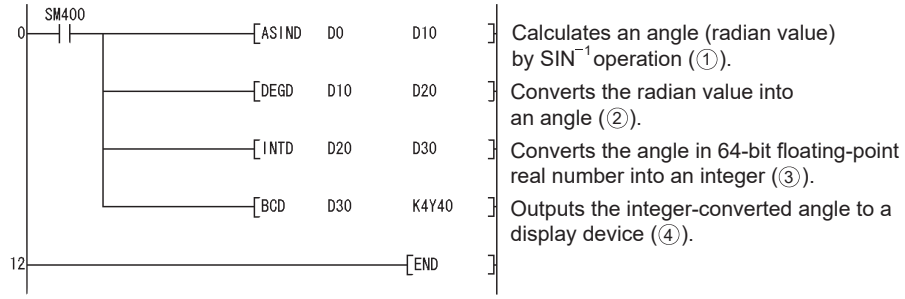
- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4100	The value specified by (S) is within the double-precision floating-point range and not within the range between -1.0 and 1.0.	—	—	—	—	○	○
4140	The specified device value is not within the following range: 0, $2^{-1022} \leq$ | Specified device value | $< 2^{1024}$ The value of the specified device is -0, a subnormal number, NaN (not a number), or ±∞.	—	—	—	—	○	○
4141	The operation result exceeds the following range. (when an overflow occurs) |Operation result| $< 2^{1024}$	—	—	—	—	○	○

Program example

- The following program seeks the inverse sine of the 64-bit floating decimal point real number at D0 to D3, and outputs the angle to the 4 BCD digits at Y40 to Y4F.

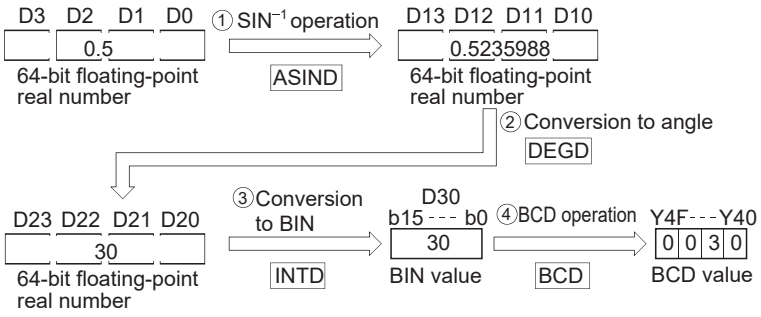
[Ladder Mode]



[List Mode]

Step	Instruction	Device	
0	LD	SM400	
1	ASIND	D0	D10
4	DEGD	D10	D20
7	INTD	D20	D30
10	BCD	D30	K4Y40
12	END		

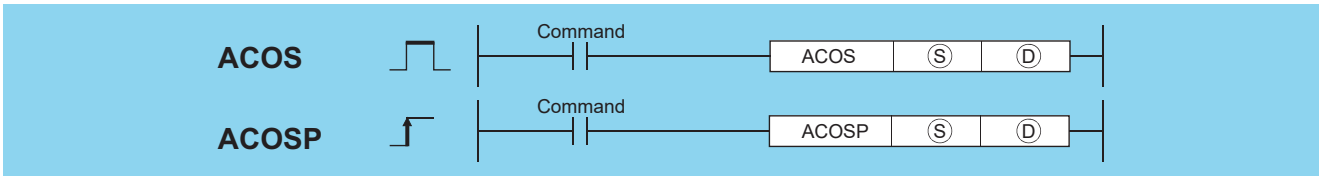
[Operations involved when the D0 to D3 value is 0.5]



Arc cosine operation on floating-point data (single precision)

ACOS(P)

Basic | High performance | Process | Redundant | Universal | LCPU



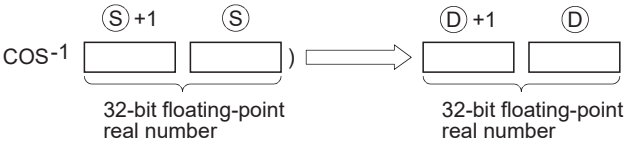
(S): COS value of which the COS^{-1} (inverse cosine) value is obtained or head number of the devices where the COS value is stored (real number)
(D): Head number of the devices where the operation result will be stored (real number)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant E	Others
	Bit	Word		Bit	Word				
(S)	—	○		—	○		○*1	○	—
(D)	—	○		—	○		○*1	—	—

*1 Applicable for the Universal model QCPU, LCPU.

Processing details

- Returns the COS^{-1} angle of the COS value designated by (S), and stores operation results at word device designated by (D).



- The COS value designated by (S) can be in the range of from -1.0 to 1.0.
- The angle (operation result) stored at (D) is stored in radian units. For more information on the conversion between radian and angle data, see description of RAD and DEG instructions.
- When an input value is set using a programming tool, a rounding error may occur. For precautions, refer to Page 90 Precautions.

Operation error

- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

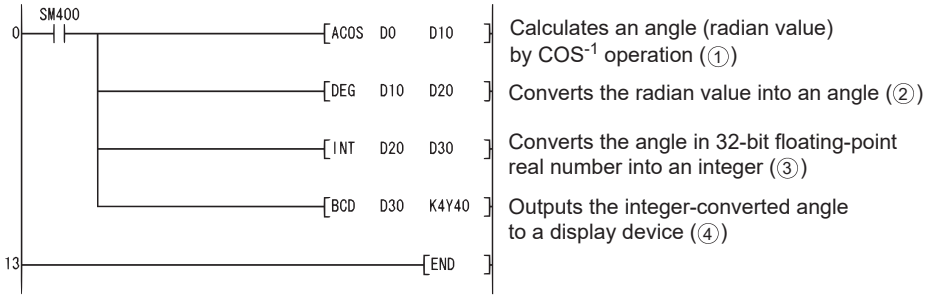
Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4100	The value specified in (S) is not within the range between -1.0 and 1.0.	—	○	○	○	○	○
	The specified device value is -0.*2	—	○	○	○	—	—
4140	The specified device value is not within the following range: 0, $2^{-126} \leq$ | Specified device value | $< 2^{128}$ The specified device value is -0, unnormalized number, nonnumeric, and ±∞.	—	—	—	—	○	○
4141	The operation result exceeds the following range. (when an overflow occurs) |Operation result| $< 2^{128}$	—	—	—	—	○	○

*2 There are CPU modules that will not result in an operation error if -0 is specified. For details, refer to Page 88 Using single/double-precision real number data.

Program example

- The following program seeks the inverse cosine of the 32-bit floating decimal point real number at D0 and D1, and outputs the angle to the 4 BCD digits at Y40 to Y4F.

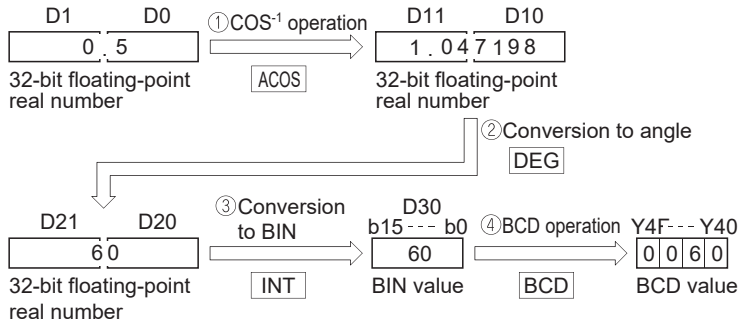
[Ladder Mode]



[List Mode]

Step	Instruction	Device	
0	LD	SM400	
1	ACOS	D0	D10
4	DEG	D10	D20
7	INT	D20	D30
10	BCD	D30	K4Y40
13	END		

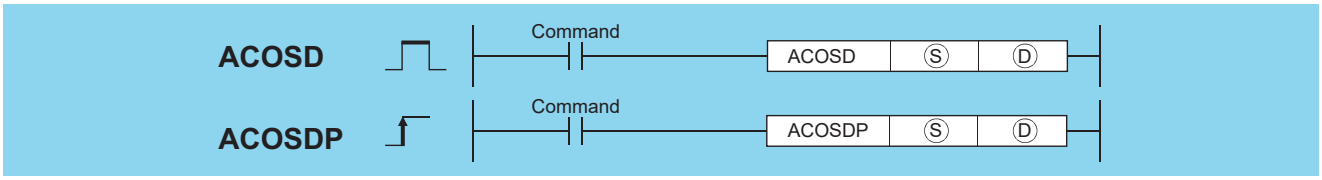
[Operations involved when the D0 and D1 value is 0.5]



Arc cosine operation on floating-point data (double precision)

ACOSD(P)

Basic | High performance | Process | Redundant | Universal | LCPU

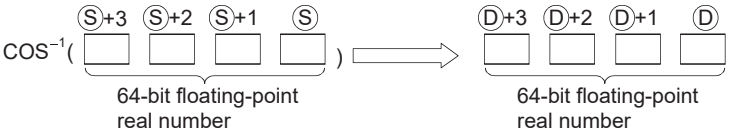


(S): COS value of which the COS^{-1} (inverse cosine) value is obtained or head number of the devices where the COS value is stored (real number)
(D): Head number of the devices where the operation result will be stored (real number)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant E	Others
	Bit	Word		Bit	Word				
(S)	—	○		—				○	—
(D)	—	○		—				—	—

Processing details

- The angle is calculated from the COS (cosine) value specified by (S) and its result is stored into the device specified by (D).



- The COS value designated by (S) can be in the range of from -1.0 to 1.0.
- The angle (operation result) stored at (D) is stored in radian units. For more information on the conversion between radian and angle data, see description of RADD and DEGD instructions.
- When the operation results in -0 or an underflow, the result is processed as 0.
- When an input value is set using a programming tool, a rounding error may occur. For precautions, refer to Page 90 Precautions.

Operation error

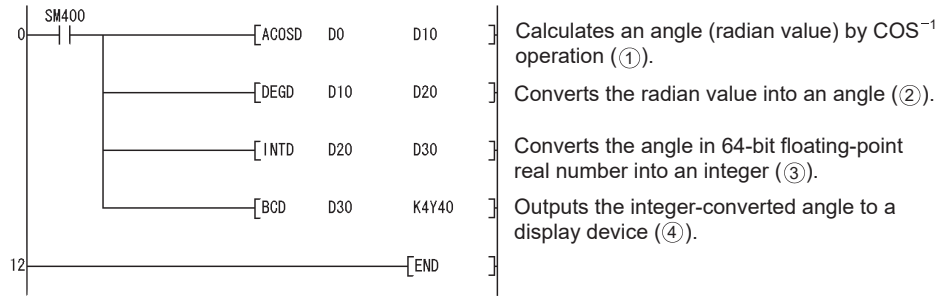
- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4100	The value specified by (S) is within the double-precision floating-point range and not within the range from -1.0 and 1.0.	—	—	—	—	○	○
4140	The specified device value is not within the following range: 0, $2^{-1022} \leq$ | Specified device value | $< 2^{1024}$ The value of the specified device is -0, a subnormal number, NaN (not a number), or ±∞.	—	—	—	—	○	○
4141	The operation result exceeds the following range. (when an overflow occurs) |Operation result| $< 2^{1024}$	—	—	—	—	○	○

Program example

- The following program seeks the inverse cosine of the 64-bit floating decimal point real number at D0 to D3, and outputs the angle to the 4 BCD digits at Y40 to Y4F.

[Ladder Mode]



Calculates an angle (radian value) by COS^{-1} operation (①).

Converts the radian value into an angle (②).

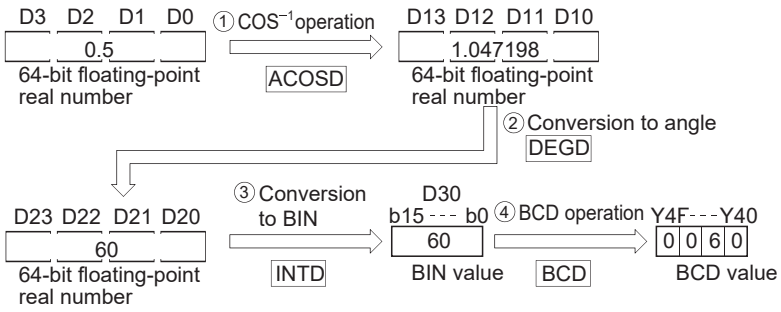
Converts the angle in 64-bit floating-point real number into an integer (③).

Outputs the integer-converted angle to a display device (④).

[List Mode]

Step	Instruction	Device	
0	LD	SM400	
1	ACOSD	D0	D10
4	DEGD	D10	D20
7	INTD	D20	D30
10	BCD	D30	K4Y40
12	END		

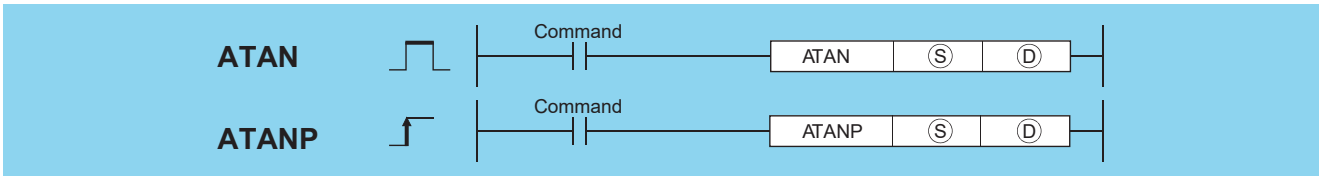
[Operations involved when the D0 to D3 value is 0.5]



Arc tangent operation on floating-point data (single precision)

ATAN(P)

Basic | High performance | Process | Redundant | Universal | LCPU



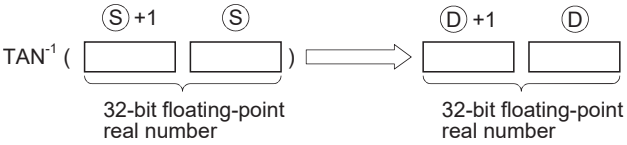
(S): TAN value of which the TAN^{-1} (inverse tangent) value is obtained or head number of the devices where the TAN value is stored (real number)
(D): Head number of the devices where the operation result will be stored (real number)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant E	Others
	Bit	Word		Bit	Word				
(S)	—	○		—	○		○*1	○	—
(D)	—	○		—	○		○*1	—	—

*1 Applicable for the Universal model QCPU, LCPU.

Processing details

- Returns the TAN^{-1} angle of the TAN value designated by (S), and stores operation results at word device designated by (D).



- The angle (operation result) stored at (D) is stored in radian units. For more information on the conversion between radian and angle data, see description of RAD and DEG instructions.
- When an input value is set using a programming tool, a rounding error may occur. For precautions, refer to Page 90 Precautions.

Operation error

- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

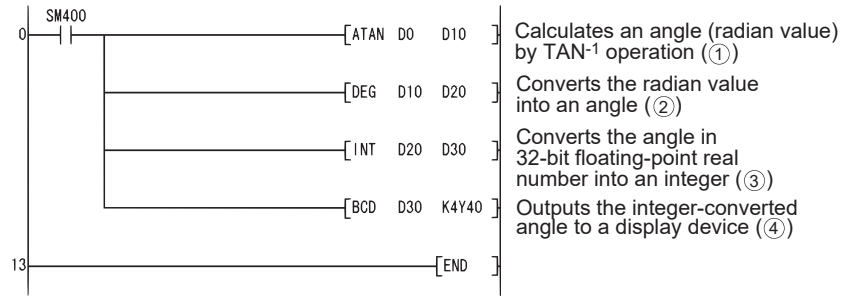
Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4100	The specified device value is -0.*2	—	○	○	○	—	—
4140	The specified device value is not within the following range: 0, $2^{-126} \leq$ | Specified device value | $< 2^{128}$ The specified device value is -0, unnormalized number, nonnumeric, and ±∞.	—	—	—	—	○	○
4141	The operation result exceeds the following range. (when an overflow occurs) |Operation result| $< 2^{128}$	—	—	—	—	○	○

*2 There are CPU modules that will not result in an operation error if -0 is specified. For details, refer to Page 88 Using single/double-precision real number data.

Program example

- The following program seeks the inverse tangent of the 32-bit floating decimal point real number at D0 and D1, and outputs the angle to the 4 BCD digits at Y40 to Y4F.

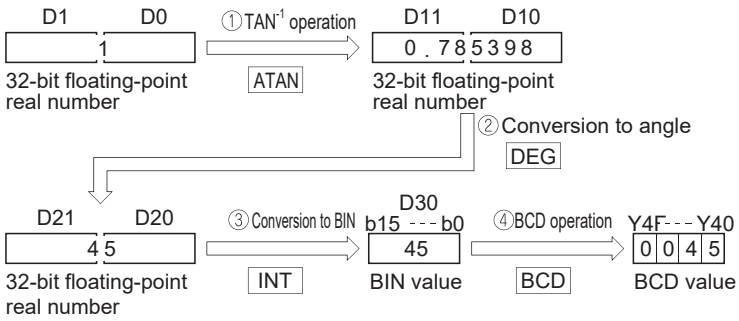
[Ladder Mode]



[List Mode]

Step	Instruction	Device	
0	LD	SM400	
1	ATAN	D0	D10
4	DEG	D10	D20
7	INT	D20	D30
10	BCD	D30	K4Y40
13	END		

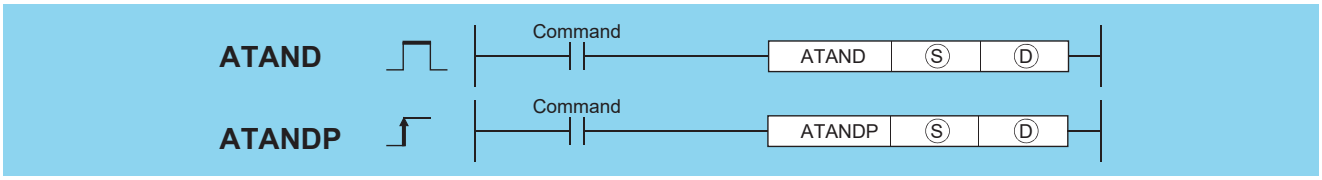
[Operations involved when D0 and D1 value is 1]



Arc tangent operation on floating-point data (double precision)

ATAND(P)

Basic | High performance | Process | Redundant | Universal | LCPU

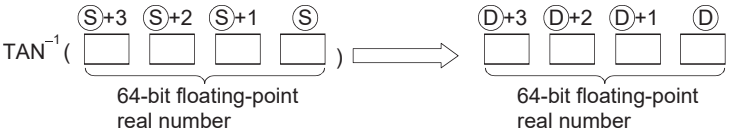


(S): TAN value of which the TAN^{-1} (inverse tangent) value is obtained or head number of the devices where the TAN value is stored (real number)
(D): Head number of the devices where the operation result will be stored (real number)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant E	Others
	Bit	Word		Bit	Word				
(S)	—	○		—				○	—
(D)	—	○		—				—	—

Processing details

- The angle is calculated from the TAN (tangent) value specified by (S) and its result is stored into the device specified by (D).



- The angle (operation result) stored at (D) is stored in radian units. For more information on the conversion between radian and angle data, see description of RADD and DEGD instructions.
- When the operation results in -0 or an underflow, the result is processed as 0.
- When an input value is set using a programming tool, a rounding error may occur. For precautions, refer to Page 90 Precautions.

Operation error

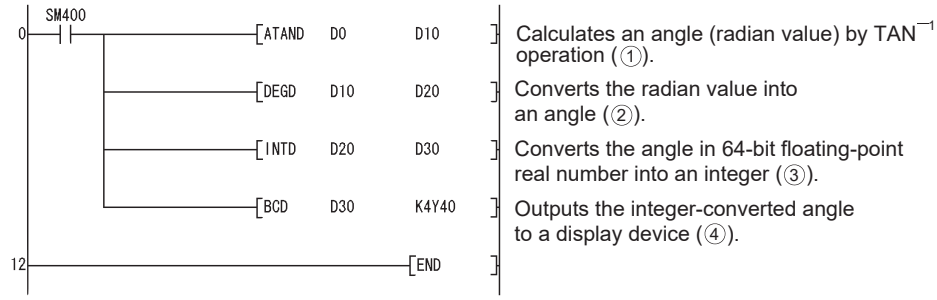
- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4140	The specified device value is not within the following range: 0, $2^{-1022} \leq$ | Specified device value | $< 2^{1024}$ The value of the specified device is -0, a subnormal number, NaN (not a number), or ±∞.	—	—	—	—	○	○
4141	The operation result exceeds the following range. (when an overflow occurs) |Operation result| $< 2^{1024}$	—	—	—	—	○	○

Program example

- The following program seeks the inverse tangent of the 64-bit floating decimal point real number at D0 to D3, and outputs the angle to the 4 BCD digits at Y40 to Y4F.

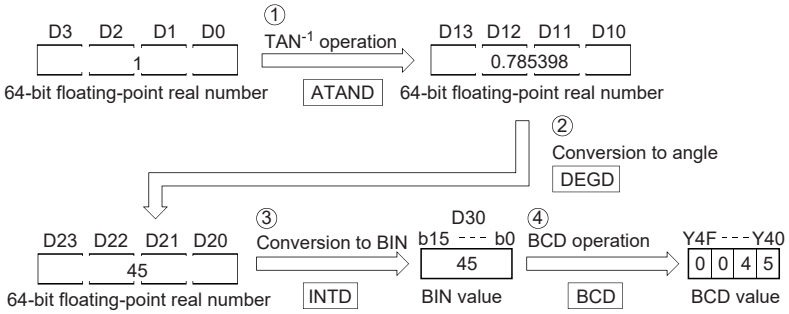
[Ladder Mode]



[List Mode]

Step	Instruction	Device	
0	LD	SM400	
1	ATAND	D0	D10
4	DEGD	D10	D20
7	INTD	D20	D30
10	BCD	D30	K4Y40
12	END		

[Operations involved when the D0 to D3 value is 1]



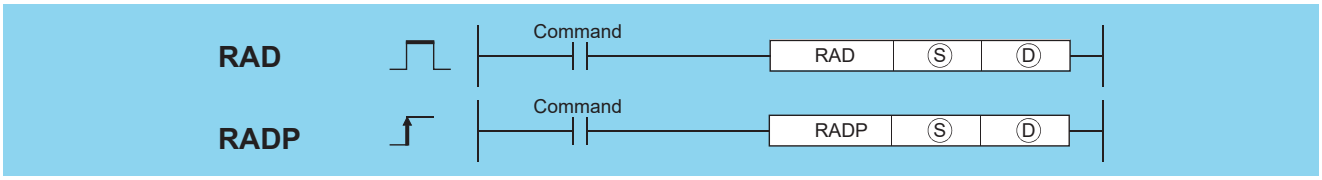
7

Conversion from floating-point angle to radian (single precision)

RAD(P)

Ver. Basic | High performance | Process | Redundant | Universal | LCPU

• Basic model QCPU: The serial number (first five digits) is "04122" or later.



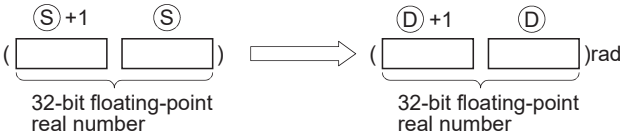
(S): Angle to be converted to radian units or head number of the devices where the angle is stored (real number)
(D): Head number of the devices where the value converted in radian units will be stored (real number)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant E	Others
	Bit	Word		Bit	Word				
(S)	—	○		—	○		○*1	○	—
(D)	—	○		—	○		○*1	—	—

*1 Applicable for the Universal model QCPU, LCPU.

Processing details

• Converts units of angle size from angle units designated by (S) to radian units, and stores result at device number designated by (D).



• Conversion from degree to radian units is performed according to the following equation:

$$\text{Radian unit} = \text{Degree unit} \times \frac{\pi}{180}$$

• When an input value is set using a programming tool, a rounding error may occur. For precautions, refer to Page 90 Precautions.

Operation error

• In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

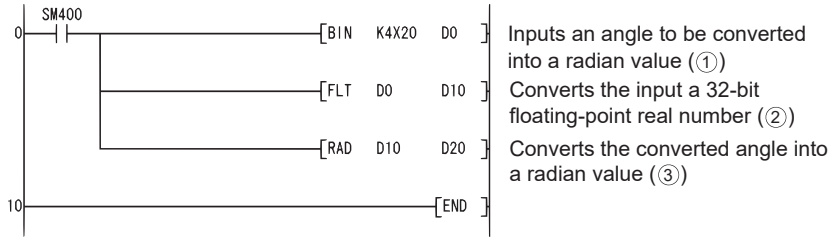
Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4100	The specified device value is -0.*2	○	○	○	○	—	—
4140	The specified device value is not within the following range: 0, $2^{-126} \leq$ | Specified device value | $< 2^{128}$ The specified device value is -0, unnormalized number, nonnumeric, and ±∞.	—	—	—	—	○	○
4141	The operation result exceeds the following range. (when an overflow occurs) |Operation result| $< 2^{128}$	—	—	—	—	○	○

*2 There are CPU modules that will not result in an operation error if -0 is specified. For details, refer to Page 88 Using single/double-precision real number data.

Program example

- The following program converts the angle set by the 4 BCD digits at X20 to X2F to radians, and stores results as 32-bit floating decimal point type real number at D20 and D21.

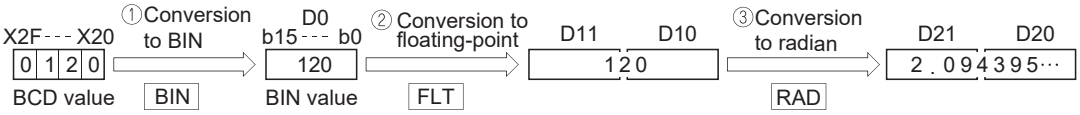
[Ladder Mode]



[List Mode]

Step	Instruction	Device	
0	LD	SM400	
1	BIN	K4X20	D0
4	FLT	D0	D10
7	RAD	D10	D20
10	END		

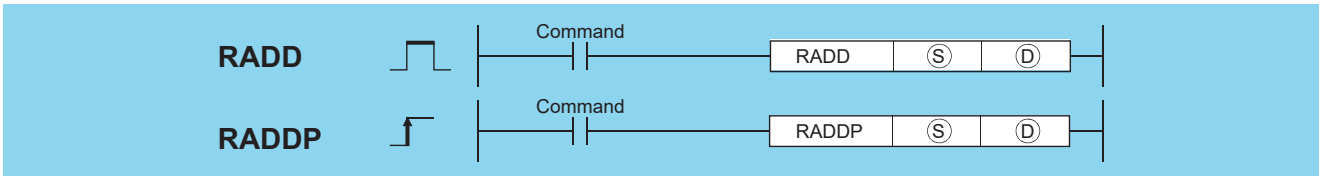
[Operations involved when X20 to X2F designate a value of 120]



Conversion from floating-point angle to radian (double precision)

RADD(P)

Basic | High performance | Process | Redundant | Universal | LCPU

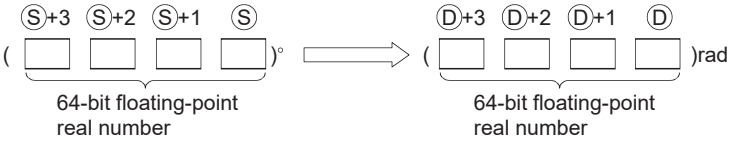


(S): Angle to be converted to radian units or head number of the devices where the angle is stored (real number)
(D): Head number of the devices where the value converted in radian units will be stored (real number)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant E	Others
	Bit	Word		Bit	Word				
(S)	—	○		—				○	—
(D)	—	○		—				—	—

Processing details

- The unit expressing the size of an angle is converted into the radian unit from the degree unit specified by (S), and its result is stored into the device specified by (D).



- Conversion from degree to radian units is performed according to the following equation:

$$\text{Radian unit} = \text{Degree unit} \times \frac{\pi}{180}$$

- When the operation results in -0 or an underflow, the result is processed as 0.
- When an input value is set using a programming tool, a rounding error may occur. For precautions, refer to Page 90 Precautions.

Operation error

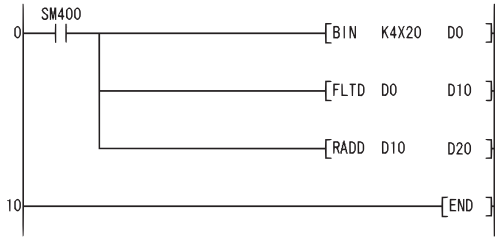
- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4140	The specified device value is not within the following range: 0, $2^{-1022} \leq$ | Specified device value | $< 2^{1024}$ The value of the specified device is -0, a subnormal number, NaN (not a number), or ±∞.	—	—	—	—	○	○
4141	The operation result exceeds the following range. (when an overflow occurs) |Operation result| $< 2^{1024}$	—	—	—	—	○	○

Program example

- The following program converts the angle set by the 4 BCD digits at X20 to X2F to radians, and stores results as 64-bit floating decimal point type real number at D20 to D23.

[Ladder Mode]

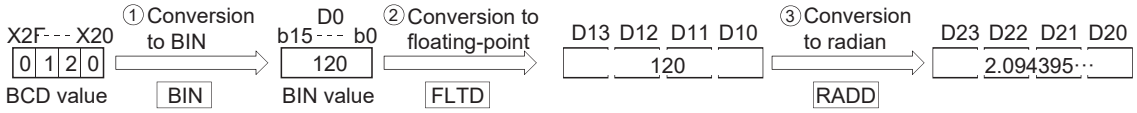


Inputs an angle to be converted into a radian value (①).
Converts the input angle into a 64-bit floating-point real number (②).
Converts the converted angle into a radian value (③).

[List Mode]

Step	Instruction	Device	
0	LD	SM400	
1	BIN	K4X20	D0
3	FLTD	D0	D10
6	RADD	D10	D20
9	END		

[Operations involved when X20 to X2F designate a value of 120]

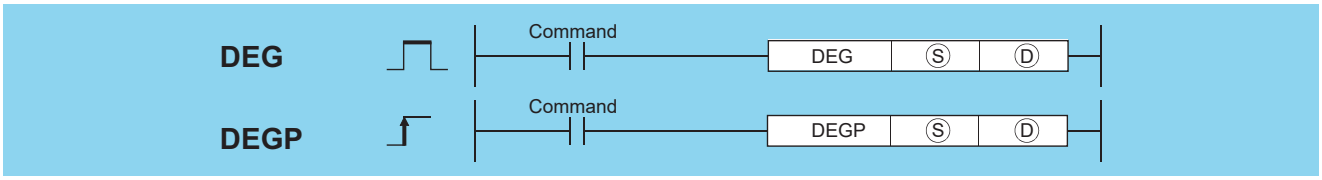


Conversion from floating-point radian to angle (single precision)

DEG(P)

Ver. Basic | High performance | Process | Redundant | Universal | LCPU

• Basic model QCPU: The serial number (first five digits) is "04122" or later.



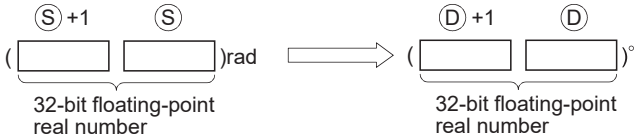
(S): Radian angle to be converted to degrees or head number of the devices where the radian angle is stored (real number)
(D): Head number of the devices where the value converted in degrees will be stored (real number)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant E	Others
	Bit	Word		Bit	Word				
(S)	—	○		—	○		○*1	○	—
(D)	—	○		—	○		○*1	—	—

*1 Applicable for the Universal model QCPU, LCPU.

Processing details

• Converts units of angle size from radian units designated by (S) to angles, and stores result at device number designated by (D).



• The conversion from radians to angles is performed according to the following equation:

$$\text{Degree unit} = \text{Radian unit} \times \frac{180}{\pi}$$

• When an input value is set using a programming tool, a rounding error may occur. For precautions, refer to Page 90 Precautions.

Operation error

• In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

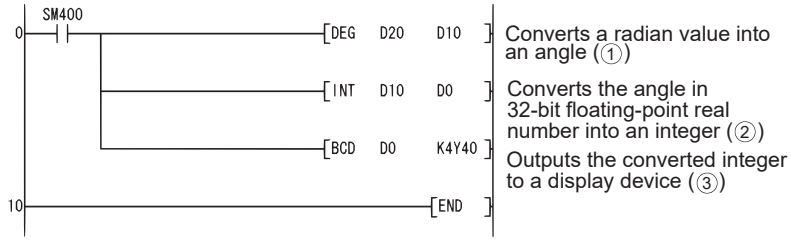
Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4100	The specified device value is -0.*2	○	○	○	○	—	—
4140	The specified device value is -0, unnormalized number, nonnumeric, and ±∞.	—	—	—	—	○	○
4141	The operation result exceeds the following range. (when an overflow occurs) |Operation result| < 2^{128}	—	—	—	—	○	○

*2 There are CPU modules that will not result in an operation error if -0 is specified. For details, refer to Page 88 Using single/double-precision real number data.

Program example

- The following program converts the radian value set with 32-bit floating decimal point type real number at D20 and D21 to angles, and stores the result as a BCD value at Y40 to Y4F.

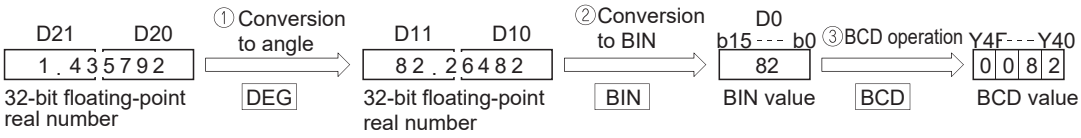
[Ladder Mode]



[List Mode]

Step	Instruction	Device	
0	LD	SM400	
1	DEG	D20	D10
4	INT	D10	D0
7	BCD	D0	K4Y40
10	END		

[Operations involved when the values at D20 and D21 are 1.435792]

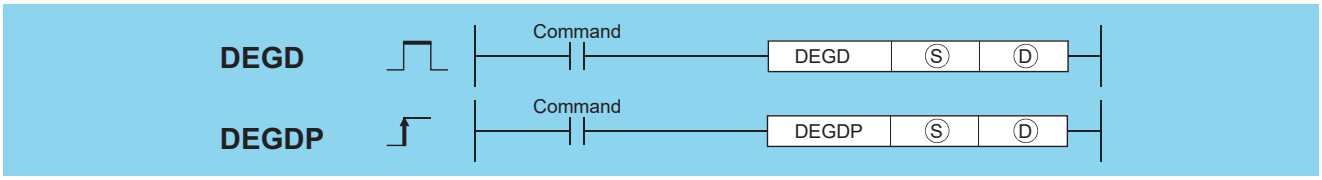


7

Conversion from floating-point radian to angle (double precision)

DEGD(P)

Basic | High performance | Process | Redundant | Universal | LCPU

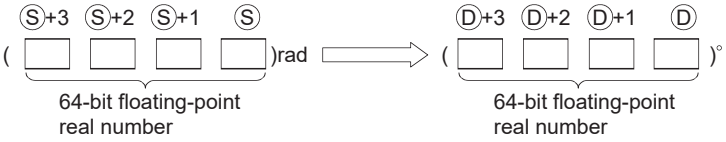


(S): Radian angle to be converted to degrees or head number of the devices where the radian angle is stored (real number)
(D): Head number of the devices where the value converted in degrees will be stored (real number)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant E	Others
	Bit	Word		Bit	Word				
(S)	—	○		—				○	—
(D)	—	○		—				—	—

Processing details

- The unit expressing the size of an angle is converted into the degree unit from the radian unit specified by (S), and its result is stored into the device specified by (D).



- The conversion from radians to angles is performed according to the following equation:

$$\text{Degree unit} = \text{Radian unit} \times \frac{180}{\pi}$$

- When the operation results in -0 or an underflow, the result is processed as 0.
- When an input value is set using a programming tool, a rounding error may occur. For precautions, refer to Page 90 Precautions.

Operation error

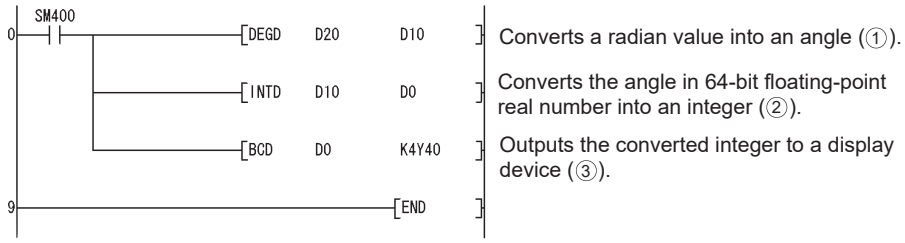
- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4140	The specified device value is not within the following range: 0, $2^{-1022} \leq$ | Specified device value | $< 2^{1024}$ The value of the specified device is -0, a subnormal number, NaN (not a number), or ±∞.	—	—	—	—	○	○
4141	The operation result exceeds the following range. (when an overflow occurs) |Operation result| $< 2^{1024}$	—	—	—	—	○	○

Program example

- The following program converts the radian value set with 64-bit floating decimal point type real number at D20 to D23 to angles, and stores the result as a BCD value at Y40 to Y4F.

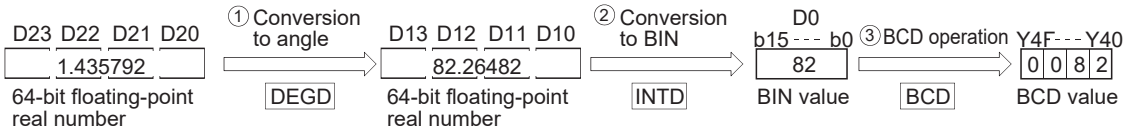
[Ladder Mode]



[List Mode]

Step	Instruction	Device	
0	LD	SM400	
1	DEGD	D20	D10
4	INTD	D10	D0
7	BCD	D0	K4Y40
9	END		

[Operations involved when the values at D20 to D23 are 1.435792]



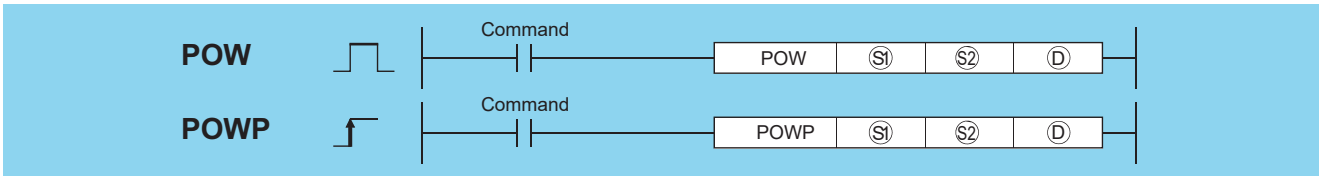
7

Exponentiation operation on floating-point data (single precision)

POW(P)

Basic | High performance | Process | Redundant | Ver. Universal | LCPU

- QnU(D)(H)CPU, QnUDE(H)CPU: the serial number (first five digits) is "10102" or later
- Q00UJCPU, Q00UCPU, Q01UCPU, QnUDVCPU, QnUDPVCPU: Supported



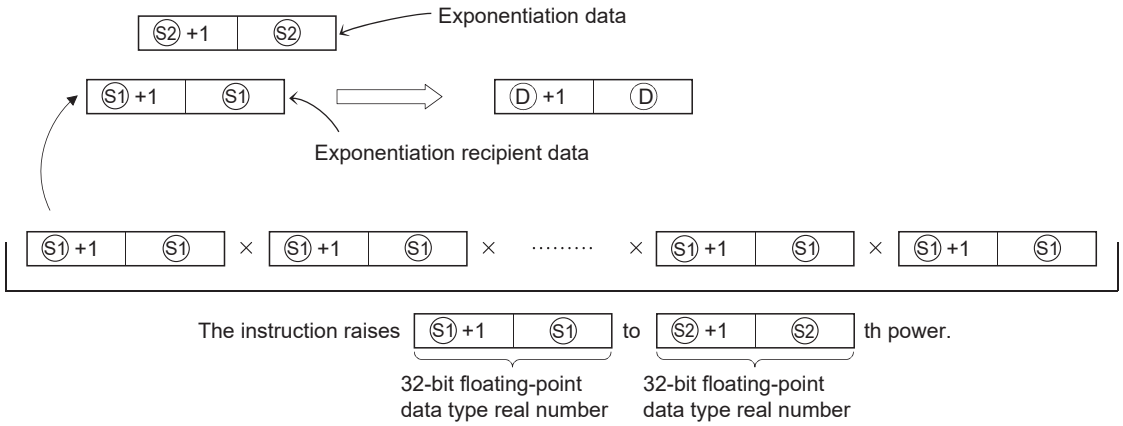
(S1): Exponentiation recipient data or head number of the devices where the exponentiation recipient data are stored (real number)
(S2): Exponentiation data or head number of the devices where the data are stored (real number)
(D): Head number of the devices where the operation result will be stored (real number)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant E	Others
	Bit	Word		Bit	Word				
(S1)	—	○		—	○		○	△*1	—
(S2)	—	○		—	○		○	△*1	—
(D)	—	○		—	○		○	—	—

*1 Available only for real number

Processing details

- This instruction raises the 32-bit floating-point data type real number specified by (S1) to the number nth specified by (S2) power, and then stores the operation result into the device specified by (D).



- The following shows the values to be specified by and stored into (S1) or (S2).

$0, 2^{-126} \leq |$ Set values (Storage values) $| < 2^{128}$

- If the value resulted from the operation is -0 or an underflow occurs, the result will be processed as 0.
- When an input value is set using a programming tool, a rounding error may occur. For precautions, refer to Page 90 Precautions.

Operation error

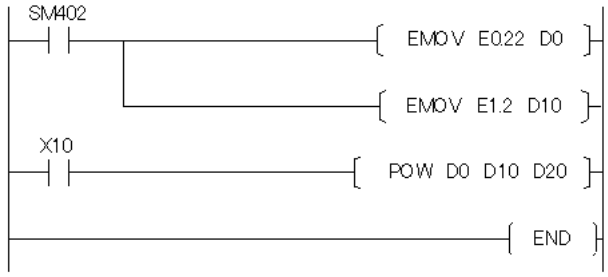
- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4140	The specified device value is not within the following range: 0, $2^{-126} \leq$ | Specified device value | $< 2^{128}$ The specified device value is -0, unnormalized number, nonnumeric, and ±∞.	—	—	—	—	○	○
4141	The operation result exceeds the following range. (when an overflow occurs) |Operation result| $< 2^{128}$	—	—	—	—	○	○

Program example

- The following program raises the 32-bit floating-point data type real number data specified by D0 and D1 to the data specified by (D10 and D11)th power, when X10 is turned on. Then the program stores the operation result into D20 and D21.

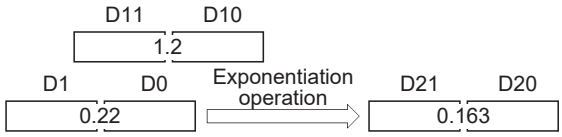
[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	SM402
1	EMOV	E0.22 D0
4	EMOV	E1.2 D10
7	LD	X10
8	POW	D0 D10 D20
12	END	

[Operation]

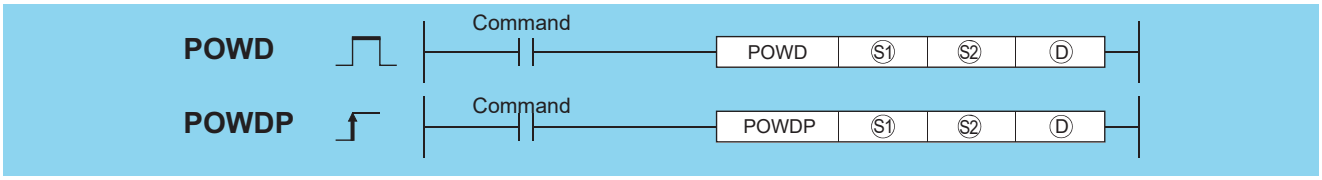


Exponentiation operation on floating-point data (double precision)

POWD(P)

Basic (×) | High performance (×) | Process (×) | Redundant (×) | Ver. Universal | LCPU

- QnU(D)(H)CPU, QnUDE(H)CPU: the serial number (first five digits) is "10102" or later
- Q00UJCPU, Q00UCPU, Q01UCPU, QnUDVCPU, QnUDPVCPU: Supported



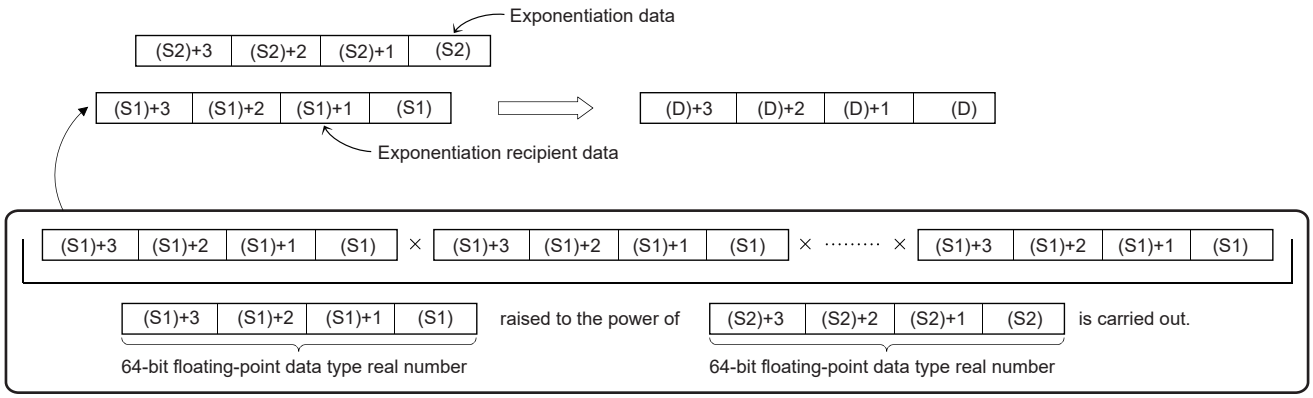
(S1): Exponentiation recipient data or head number of the devices where the exponentiation recipient data are stored (real number)
(S2): Exponentiation data or head number of the devices where the data are stored (real number)
(D): Head number of the devices where the operation result will be stored (real number)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant E	Others
	Bit	Word		Bit	Word				
(S1)	—	○		—				△*1	—
(S2)	—	○		—				△*1	—
(D)	—	○		—				—	—

*1 Available only for real number

Processing details

- This instruction raises the 64-bit floating-point data type real number specified by (S1) to the number nth specified by (S2) power, and then stores the operation result into the device specified by (D).



- The following shows the values to be specified by and stored into (S1) or (S2).

$0, 2^{-1022} \leq |$ Set values (Storage values) $| < 2^{1024}$

- If the value resulted from the operation is -0 or an underflow occurs, the result will be processed as 0.
- When an input value is set using a programming tool, a rounding error may occur. For precautions, refer to Page 90 Precautions.

Operation error

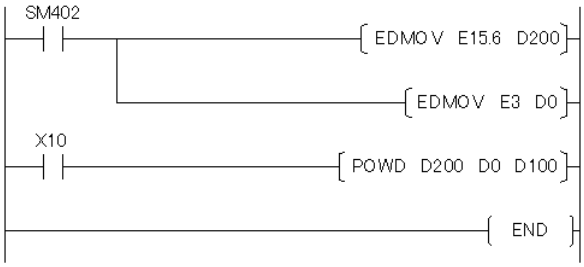
- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4140	The specified device value is not within the following range: 0, $2^{-1022} \leq$ | Specified device value | $< 2^{1024}$ The value of the specified device is -0, a subnormal number, NaN (not a number), or $\pm\infty$.	—	—	—	—	○	○
4141	The operation result exceeds the following range. (when an overflow occurs) |Operation result| $< 2^{1024}$	—	—	—	—	○	○

Program example

- The following program raises the 64-bit floating-point data type real number specified by D200 to D203 to the number nth specified by D0 to D3 power, when X10 is turned on. Then the program stores the operation result into D100 to D103.

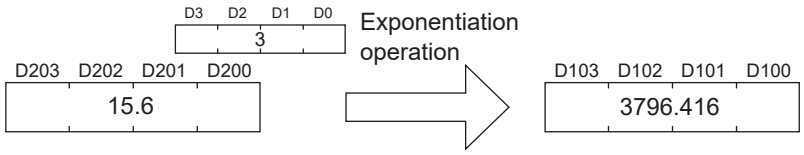
[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	SM402
1	EDMOV	E15.6 D200
4	EDMOV	E3 D0
7	LD	X10
8	POWD	D200 D0 D100
12	END	

[Operation]

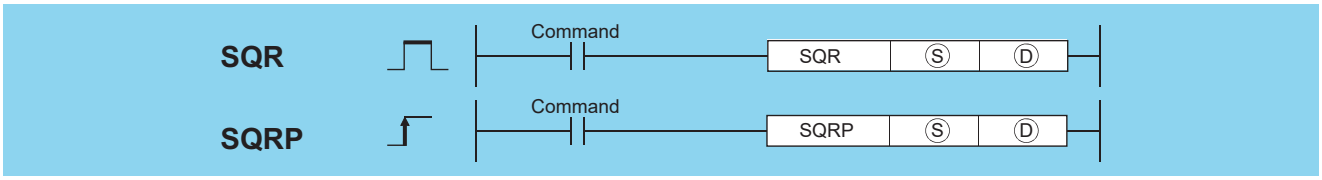


Square root operation for floating-point data (single precision)

SQR(P)

Ver. Basic | High performance | Process | Redundant | Universal | LCPU

• Basic model QCPU: The serial number (first five digits) is "04122" or later.



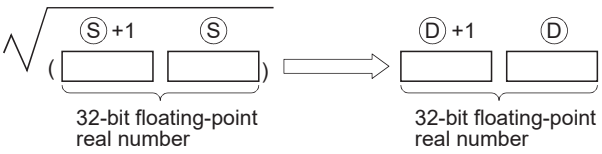
(S): Data of which the square root is obtained or head number of the devices where the data is stored (real number)
(D): Head number of the devices where the operation result will be stored (real number)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant E	Others
	Bit	Word		Bit	Word				
(S)	—	○		—	○		○*1	○	—
(D)	—	○		—	○		○*1	—	—

*1 Applicable for the Universal model QCPU, LCPU.

Processing details

• Returns the square root of the value designated at (S), and stores the operation result in the device number designated at (D).



• Only positive values can be specified by (S). (Operation cannot be performed on negative numbers.)
• When an input value is set using a programming tool, a rounding error may occur. For precautions, refer to Page 90 Precautions.

Operation error

• In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

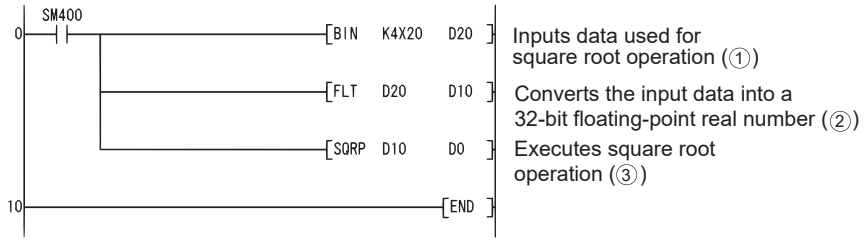
Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4100	The value specified in (S) is negative.	○	○	○	○	○	○
	The specified device value is -0.*2	○	○	○	○	—	—
4140	The specified device value is not within the following range: 0, $2^{-126} \leq$ | Specified device value | $< 2^{128}$ The specified device value is -0, unnormalized number, nonnumeric, and ±∞.	—	—	—	—	○	○
4141	The operation result exceeds the following range. (when an overflow occurs) |Operation result| $< 2^{128}$	—	—	—	—	○	○

*2 There are CPU modules that will not result in an operation error if -0 is specified. For details, refer to Page 88 Using single/double-precision real number data.

Program example

- The following program seeks the square root of the value set by the 4 BCD digits from X20 to X2F, and stores the result as a 32-bit floating decimal point type real number at D0 and D1.

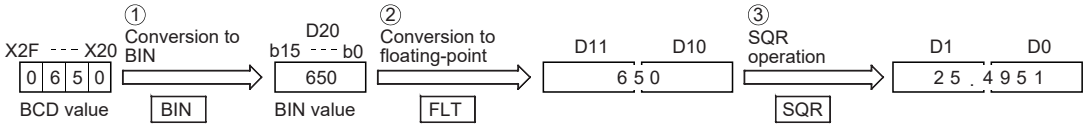
[Ladder Mode]



[List Mode]

Step	Instruction	Device	
0	LD	SM400	
1	BIN	K4X20	D20
4	FLT	D20	D10
7	SQRP	D10	D0
10	END		

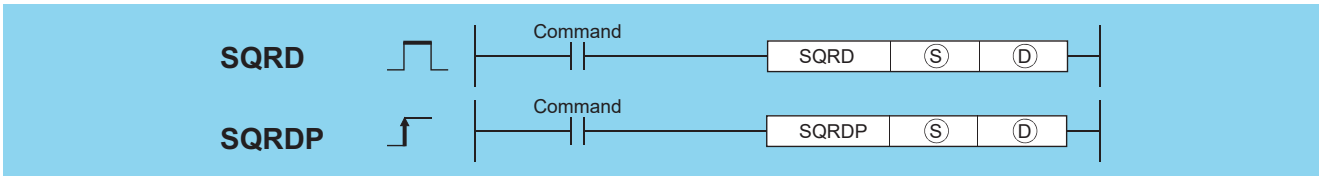
[Operations involved when value designated by X20 to X2F is 650]



Square root operation for floating-point data (double precision)

SQRD(P)

Basic | High performance | Process | Redundant | Universal | LCPU

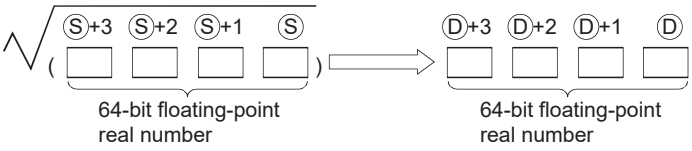


(S): Data of which the square root is obtained or head number of the devices where the data is stored (real number)
(D): Head number of the devices where the operation result will be stored (real number)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant E	Others
	Bit	Word		Bit	Word				
(S)	—	○		—				○	—
(D)	—	○		—				—	—

Processing details

- Returns the square root of the value designated at (S), and stores the operation result in the device number designated at (D).



- Only positive values can be designated by (S). (Operation cannot be performed on negative numbers.)
- When the operation results in -0 or an underflow, the result is processed as 0.
- When an input value is set using a programming tool, a rounding error may occur. For precautions, refer to Page 90 Precautions.

Operation error

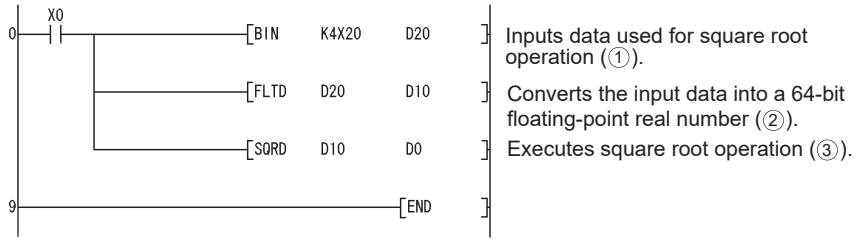
- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4100	The value specified in (S) is negative.	—	—	—	—	○	○
4140	The specified device value is not within the following range: 0, $2^{-1022} \leq$ | Specified device value | $< 2^{1024}$ The value of the specified device is -0, a subnormal number, NaN (not a number), or ±∞.	—	—	—	—	○	○
4141	The operation result exceeds the following range. (when an overflow occurs) |Operation result| $< 2^{1024}$	—	—	—	—	○	○

Program example

- The following program seeks the square root of the value set by the 4 BCD digits from X20 to X2F, and stores the result as a 64-bit floating decimal point type real number at D0 to D3.

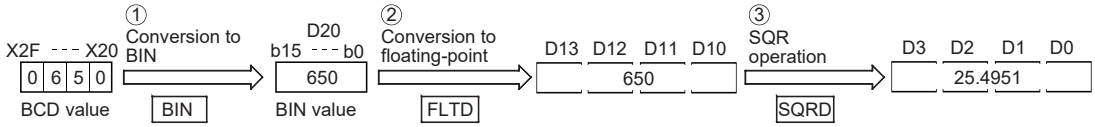
[Ladder Mode]



[List Mode]

Step	Instruction	Device	
0	LD	X0	
1	BIN	K4X20	D20
3	FLTD	D20	D10
6	SQRD	D10	D0
9	END		

[Operations involved when value designated by X20 to X2F is 650]

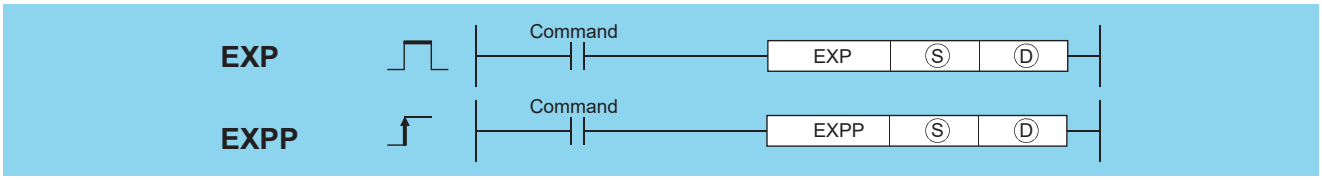


Exponent operation on floating-point data (single precision)

EXP(P)

Ver. Basic | High performance | Process | Redundant | Universal | LCPU

• Basic model QCPU: The serial number (first five digits) is "04122" or later.



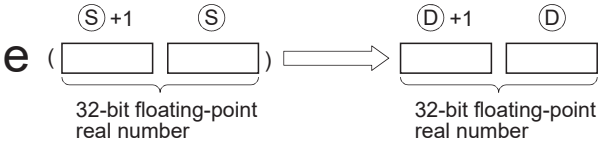
(S): Data of which the exponential value is obtained or head number of the devices where the data is stored (real number)
(D): Head number of the devices where the operation result will be stored (real number)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant E	Others
	Bit	Word		Bit	Word				
(S)	—	○		—	○		○*1	○	—
(D)	—	○		—	○		○*1	—	—

*1 Applicable for the Universal model QCPU, LCPU.

Processing details

• Returns the exponent of the value designated by (S), and stores the results of the operation at the device designated by (D).



• Exponent operations are calculated taking the base (e) to be "2.71828".
• When an input value is set using a programming tool, a rounding error may occur. For precautions, refer to Page 90 Precautions.

Operation error

• In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

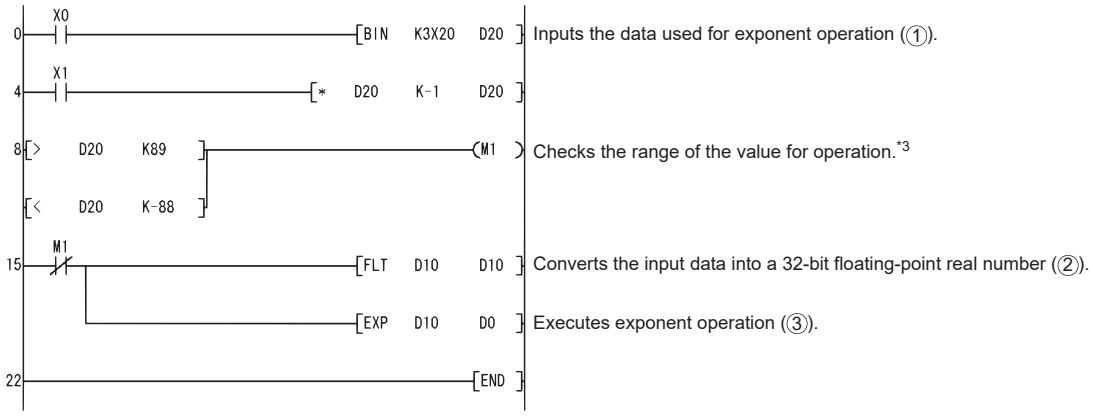
Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4100	The operation result is not within the following range: $2^{-126} \leq \lvert$ Operation result $\rvert \leq 2^{128}$	—	○	—	—	—	—
	The operation result is not within the following range: $2^{-126} \leq \lvert$ Operation result $\rvert < 2^{128}$	○	—	○	○	—	—
	The specified device value is -0.*2	○	○	○	○	—	—
4140	The specified device value is -0, unnormalized number, nonnumeric, or ±∞.	—	—	—	—	○	○
4141	The operation result exceeds the following range. (when an overflow occurs) |Operation result| $< 2^{128}$	—	—	—	—	○	○

*2 There are CPU modules that will not result in an operation error if -0 is specified. For details, refer to Page 88 Using single/double-precision real number data.

Program example

- The following program performs an exponent operation on the value set by the 2 BCD digits at X20 to X27, and stores the results as a 32-bit floating decimal point real number at D0 and D1.

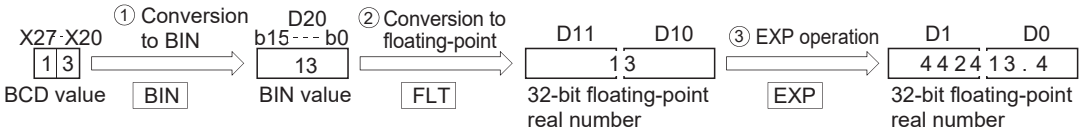
[Ladder Mode]



[List Mode]

Step	Instruction	Device		
0	LD	X0		
1	BIN	K3X20	D20	
4	LD	X1		
5	*	D20	K-1	D20
8	LD>	D20	K89	
11	OR<	D20	K-88	
14	OUT	M1		
15	LDI	M1		
16	FLT	D10	D10	
19	EXP	D10	D0	
22	END			

[Operations involved when value designated by X20 to X27 is 13]



*3 The operation result will be under 2^{129} if the BCD value of X20 to X27 is less than 89, from the calculation $\log_e 2^{129} = 89.4$. Because setting a value of over 90 will return an operation error, turn M1 ON if a value of over 90 has been set to avoid the error.

Point

Conversion from natural logarithm to common logarithm

In the CPU module, calculation is made using a natural logarithm.

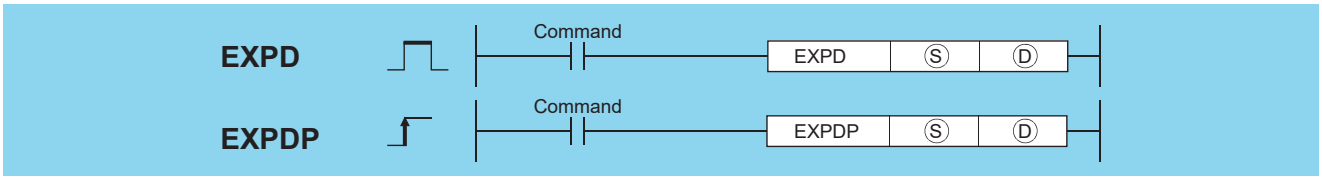
To obtain a common logarithm value, enter in, (S) a common logarithm value divided by 0.43429.

$$10^X = e^{\frac{X}{0.43429}}$$

Exponent operation on floating-point data (double precision)

EXPD(P)

Basic | High performance | Process | Redundant | Universal | LCPU

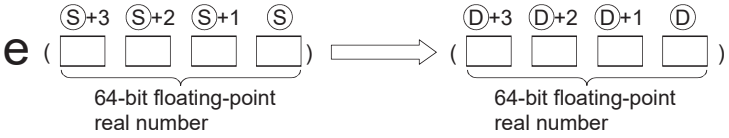


(S): Data of which the exponential value is obtained or head number of the devices where the data is stored (real number)
(D): Head number of the devices where the operation result will be stored (real number)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant E	Others
	Bit	Word		Bit	Word				
(S)	—	○		—				○	—
(D)	—	○		—				—	—

Processing details

- Returns the exponent of the value designated by (S), and stores the results of the operation at the device designated by (D).



- Exponent operations are calculated taking the base (e) to be "2.71828".
- When the operation results in -0 or an underflow, the result is processed as 0.
- When an input value is set using a programming tool, a rounding error may occur. For precautions, refer to Page 90 Precautions.

Operation error

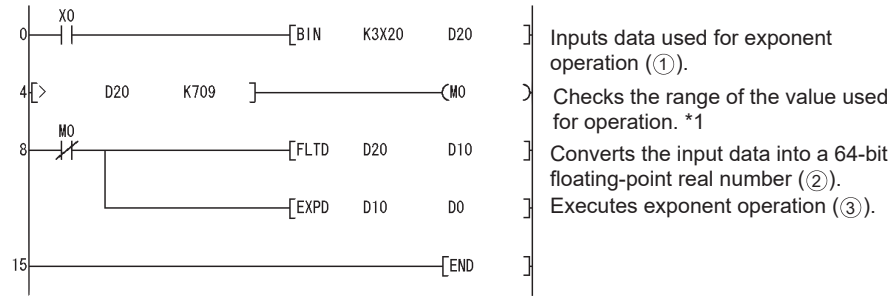
- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4140	The specified device value is not within the following range: 0, $2^{-1022} \leq$ | Specified device value | $< 2^{1024}$ The value of the specified device is -0, a subnormal number, NaN (not a number), or ±∞.	—	—	—	—	○	○
4141	The operation result exceeds the following range. (when an overflow occurs) |Operation result| $< 2^{1024}$	—	—	—	—	○	○

Program example

- The following program performs an exponent operation on the value set by the 2 BCD digits at X20 to X31, and stores the results as a 64-bit floating decimal point real number at D0 to D3.

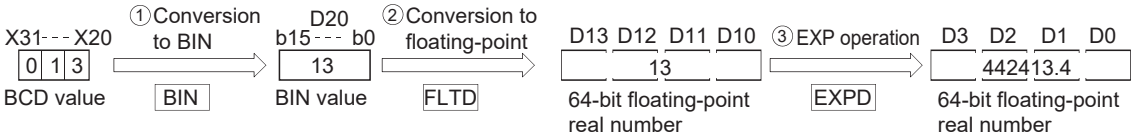
[Ladder Mode]



[List Mode]

Step	Instruction	Device	
0	LD	X0	
1	BIN	K3X20	D20
4	LD>	D20	K709
7	OUT	M0	
8	LDI	M0	
9	FLTD	D20	D10
12	EXPD	D10	D0
15	END		

[Operations involved when value designated by X20 to X31 is 13]



*1 The operation result will be under 2^{1024} if the BCD value of X20 to X31 is less than 709, from the calculation $\log_e 2^{1024} = 709.7832$. Because setting a value of over 710 will return an operation error, turn M0 ON if a value of over 710 has been set to avoid the error.

Point

Conversion from natural logarithm to common logarithm

In the CPU module, calculation is made using a natural logarithm.

To obtain a common logarithm value, enter in, (S) a common logarithm value divided by 0.43429.

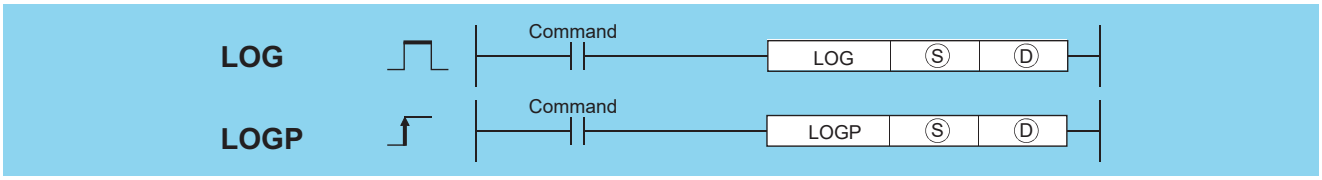
$$10^X = e^{\frac{X}{0.43429}}$$

Natural logarithm operation on floating-point data (single precision)

LOG(P)

Ver. Basic | High performance | Process | Redundant | Universal | LCPU

• Basic model QCPU: The serial number (first five digits) is "04122" or later.



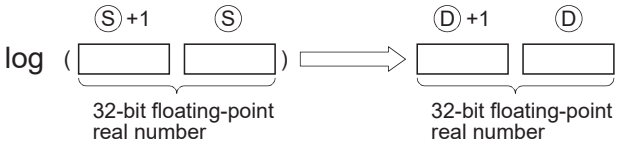
(S): Data of which the natural logarithm is obtained or head number of the devices where the data is stored (real number)
(D): Head number of the devices where the operation result will be stored (real number)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant E	Others
	Bit	Word		Bit	Word				
(S)	—	○		—	○		○*1	○	—
(D)	—	○		—	○		○*1	—	—

*1 Applicable for the Universal model QCPU, LCPU.

Processing details

• Returns the natural logarithm of the value designated by (S) taking (e) as base, and stores operation results at device designated by (D).



• Only positive values can be designated by (S). (Operation cannot be performed on negative numbers.)
• When an input value is set using a programming tool, a rounding error may occur. For precautions, refer to Page 90 Precautions.

Operation error

• In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

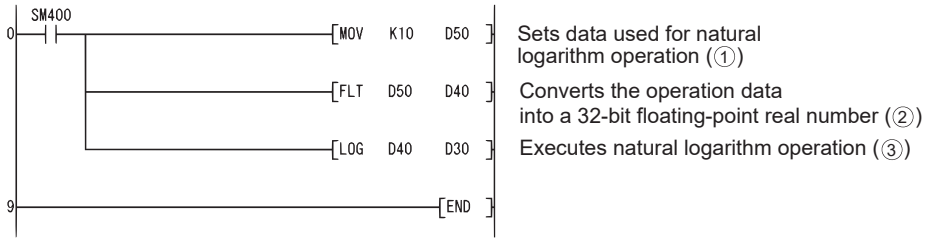
Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4100	The value specified in (S) is negative. The value specified in (S) is 0.	○	○	○	○	○	○
	The specified device value is -0.*2	○	○	○	○	—	—
4140	The specified device value is not within the following range: 0, $2^{-126} \leq$ | Specified device value | $< 2^{128}$ The specified device value is -0, unnormalized number, nonnumeric, and ±∞.	—	—	—	—	○	○
4141	The operation result exceeds the following range. (when an overflow occurs) |Operation result| $< 2^{128}$	—	—	—	—	○	○

*2 There are CPU modules that will not result in an operation error if -0 is specified. For details, refer to Page 88 Using single/double-precision real number data.

Program example

• The following program seeks the natural logarithm of the value "10" set by D50, and stores the result at D30 and D31.

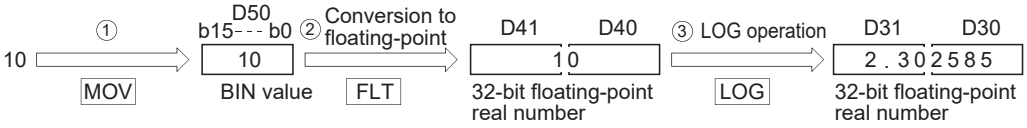
[Ladder Mode]



[List Mode]

Step	Instruction	Device	
0	LD	SM400	
1	MOV	K10	D50
3	FLT	D50	D40
6	LOG	D40	D30
9	END		

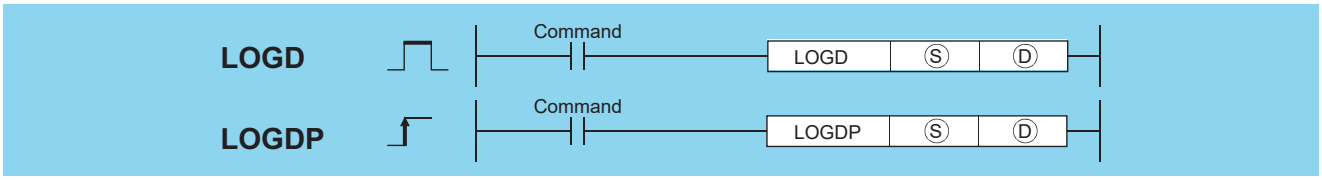
[Operation]



Natural logarithm operation on floating-point data (double precision)

LOGD(P)

Basic (×) | High performance (×) | Process (×) | Redundant (×) | Universal | LCPU

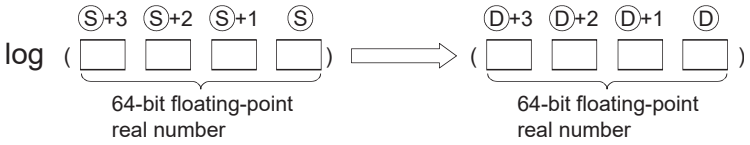


(S): Data of which the natural logarithm is obtained or head number of the devices where the data is stored (real number)
(D): Head number of the devices where the operation result will be stored (real number)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant E	Others
	Bit	Word		Bit	Word				
(S)	—	○		—				○	—
(D)	—	○		—				—	—

Processing details

- Returns the natural logarithm of the value designated by (S) taking (e) as base, and stores operation results at device designated by (D).



- Only positive values can be designated by (S). (Operation cannot be performed on negative numbers.)
- When the operation results in -0 or an underflow, the result is processed as 0.
- When an input value is set using a programming tool, a rounding error may occur. For precautions, refer to Page 90 Precautions.

Operation error

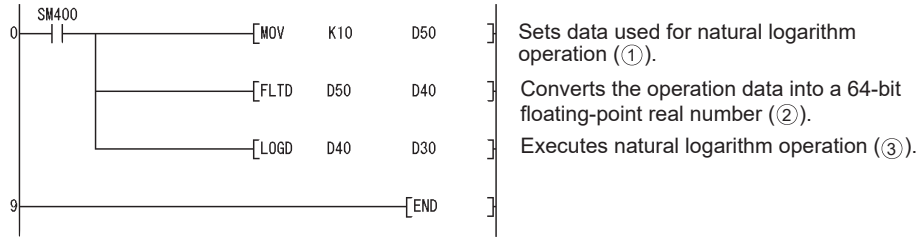
- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4100	The value specified in (S) is negative. The value specified in (S) is 0.	—	—	—	—	○	○
4140	The specified device value is not within the following range: 0, $2^{-1022} \leq$ | Specified device value | $< 2^{1024}$ The value of the specified device is -0, a subnormal number, NaN (not a number), or ±∞.	—	—	—	—	○	○
4141	The operation result exceeds the following range. (when an overflow occurs) |Operation result| $< 2^{1024}$	—	—	—	—	○	○

Program example

• The following program seeks the natural logarithm of the value "10" set by D50, and stores the result at D30 to D33.

[Ladder Mode]



Sets data used for natural logarithm operation (①).

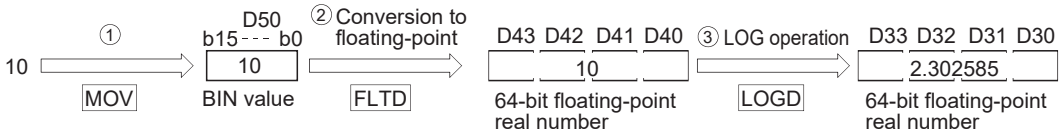
Converts the operation data into a 64-bit floating-point real number (②).

Executes natural logarithm operation (③).

[List Mode]

Step	Instruction	Device	
0	LD	SM400	
1	MOV	K10	D50
3	FLTD	D50	D40
6	LOGD	D40	D30
9	END		

[Operation]



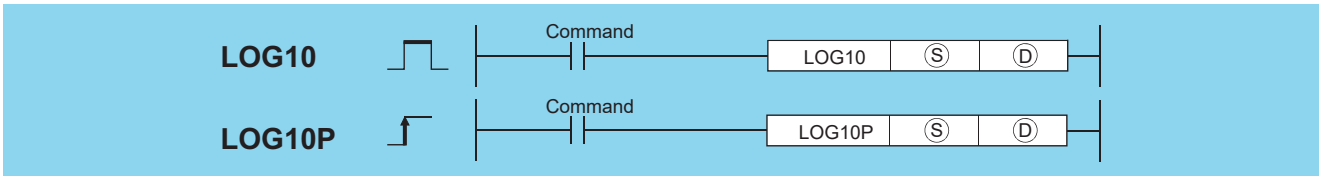
7

Common logarithm operation on floating-point data (single precision)

LOG10(P)

Basic (×) | High performance (×) | Process (×) | Redundant (×) | Universal (Ver.) | LCPU

- QnU(D)(H)CPU, QnUDE(H)CPU: the serial number (first five digits) is "10102" or later
- Q00UJCPU, Q00UCPU, Q01UCPU, QnUDVCPU, QnUDPVCPU: Supported



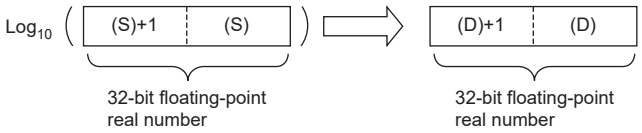
(S): Data of which the common logarithm is obtained or head number of the devices where the data are stored (real number)
(D): Head number of the devices where the operation result will be stored (real number)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant E	Others
	Bit	Word		Bit	Word				
(S)	—	○		—	○		○	△*1	—
(D)	—	○		—	○		○	—	—

*1 Available only for real number.

Processing details

- This instruction obtains the value specified by (S) for common logarithm (logarithm with base 10), and then stores the operation result into the device specified by (D).



- Only positive values can be specified by (S). (Operation cannot be performed on negative numbers.)
- If the value resulted from the operation is -0 or an underflow occurs, the result will be processed as 0.
- When an input value is set using a programming tool, a rounding error may occur. For precautions, refer to Page 90 Precautions.

Operation error

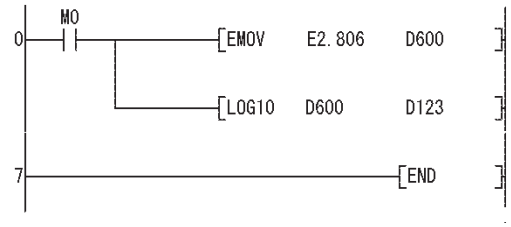
- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4100	The value specified in (S) is negative. The value specified in (S) is 0.	—	—	—	—	○	○
4140	The specified device value is not within the following range: 0, $2^{-126} \leq$ | Specified device value | $< 2^{128}$ The value of the specified device is -0, a subnormal number, NaN (not a number), or $\pm\infty$.	—	—	—	—	○	○
4141	The operation result exceeds the following range. (when an overflow occurs) |Operation result| $< 2^{128}$	—	—	—	—	○	○

Program example

- The following program obtains the value for common logarithm of the 32-bit floating-point data type real number specified by D600 or D601, when X10 is turned on. Then the program stores the operation result into D123 or D124.

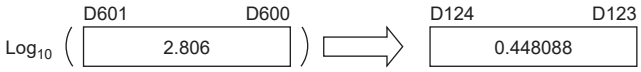
[Ladder Mode]



[List Mode]

Step	Instruction	Device	
0	LD	M0	
1	EMOV	E2.806	D600
4	LOG10	D600	D123
7	END		

[Operation]



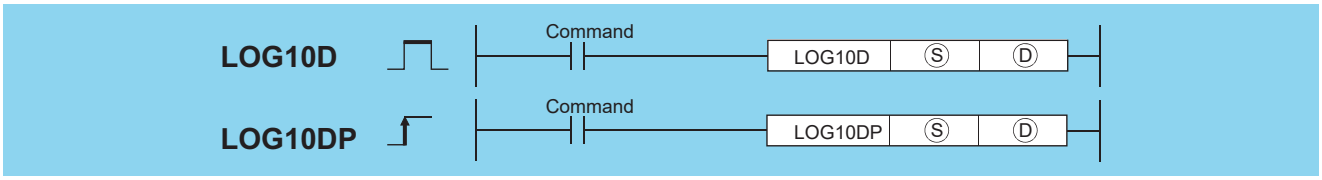
7

Common logarithm operation on floating-point data (double precision)

LOG10D(P)

Basic | High performance | Process | Redundant | Universal (Ver.) | LCPU

- QnU(D)(H)CPU, QnUDE(H)CPU: the serial number (first five digits) is "10102" or later
- Q00UJCPU, Q00UCPU, Q01UCPU, QnUDVCPU, QnUDPVCPU: Supported



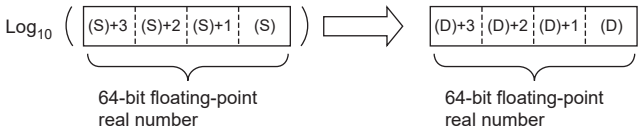
(S): Data of which the common logarithm is obtained or head number of the devices where the data are stored (real number)
(D): Head number of the devices where the operation result will be stored (real number)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(S)	—	○		—				△*1	—
(D)	—	○		—				—	—

*1 Available only for real number

Processing details

- This instruction obtains the value specified by (S) for common logarithm (logarithm with base 10), and then stores the operation result into the device specified by (D).



- Only positive values can be specified by (S). (Operation cannot be performed on negative numbers.)
- If the value resulted from the operation is -0 or an underflow occurs, the result will be processed as 0.
- When an input value is set using a programming tool, a rounding error may occur. For precautions, refer to Page 90 Precautions.

Operation error

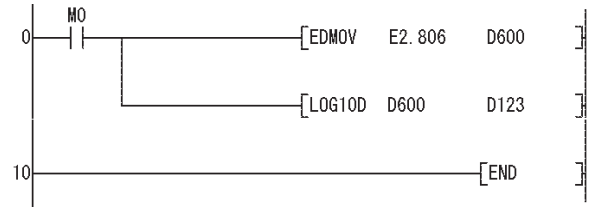
- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4100	The value specified in (S) is negative. The value specified in (S) is 0.	—	—	—	—	○	○
4140	The specified device value is not within the following range: 0, $2^{-1022} \leq$ | Specified device value | $< 2^{1024}$ The value of the specified device is -0, a subnormal number, NaN (not a number), or $\pm\infty$.	—	—	—	—	○	○
4141	The operation result exceeds the following range. (when an overflow occurs) |Operation result| $< 2^{1024}$	—	—	—	—	○	○

Program example

- This following program obtains the value for common logarithm of the 64-bit floating-point data type real number specified by D600 to D603 when M0 is turned on. Then the program stores the operation result into D123 to D126.

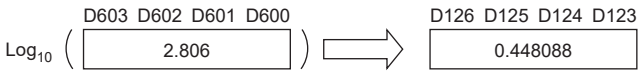
[Ladder Mode]



[List Mode]

Step	Instruction	Device	
0	LD	M0	
1	EDMOV	E2.806	D600
7	LOG10D	D600	D123
10	END		

[Operation]

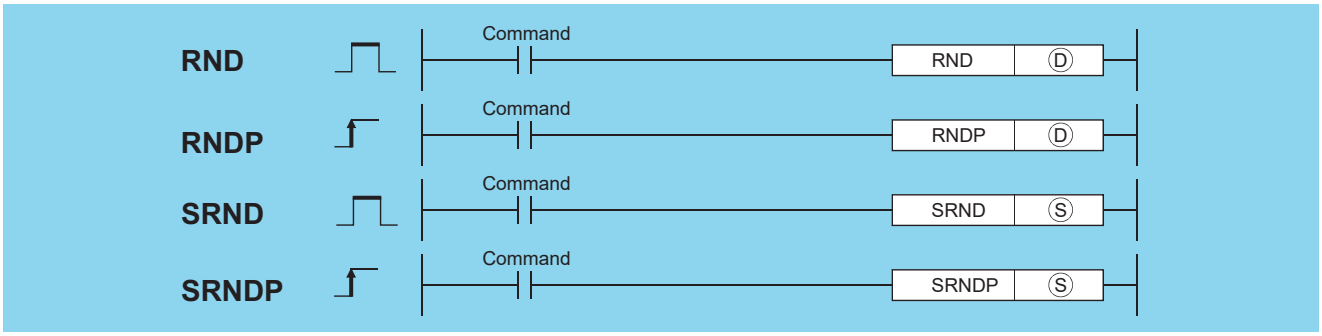


Random number generation, series updates

RND(P), SRND(P)

Ver. Basic | High performance | Process | Redundant | Universal | LCPU

• Basic model QCPU: The serial number (first five digits) is "04122" or later.



(D): Head number of the devices where random numbers will be stored (BIN 16 bits)
(S): Random number serial data or the first number of the devices where the random number serial data is stored (BIN 16 bits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(D)	○							—	—
(S)	○							○	—

Processing details

• The random number generation instruction generates random numbers conforming to a certain calculation formula. In the calculation using the formula, the result of previous calculation is used as a coefficient. The random series change instruction can change the random number generation pattern.

■RND

• Generates random number of from 0 to 32767, and stores at device designated by (D).

■SRND

• Updates random number series according to the 16-bit BIN data being stored in device designated by (S).

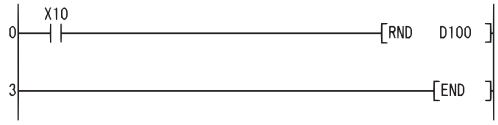
Operation error

• There is no operation error in the RND(P) or SRND(P) instruction.

Program example

• The following program stores random number at D100 when X10 is turned ON.

[Ladder Mode]

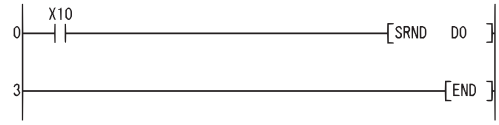


[List Mode]

Step	Instruction	Device
0	LD	X10
1	RND	D100
3	END	

• The following program updates a random number series according to the contents of D0 when X10 is turned ON.

[Ladder Mode]



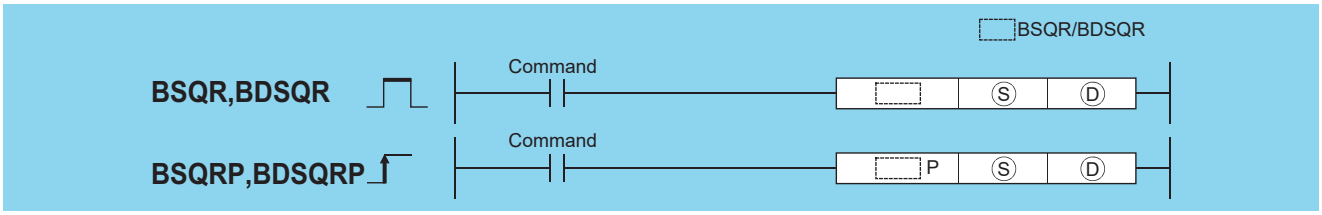
[List Mode]

Step	Instruction	Device
0	LD	X10
1	SRND	D0
3	END	

BCD 4-digit square roots, BCD 8-digit square roots

BSQR(P), BDSQR(P)

Basic | High performance | Process | Redundant | Universal | LCPU



(S): Data of which the square root is obtained or the number of the device where the data is stored (BSQR(P): BCD 4 digits, BDSQR(P): BCD 8 digits)
(D): Head number of the devices where the operation result will be stored (BCD 4 digits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(S)	○							○	—
(D)	○							—	—

Processing details

■BSQR

- Returns the square root of the value designated at (S), and stores the operation result in the device number designated at (D).

$$\sqrt{(S)} = (D).(D)+1$$

(D): Integer part, (D)+1: Decimal fraction part

- Values that can be designated at (S) are BCD values with a maximum of 4 digits (from 0 to 9999).
- The operation results of (D) and (D)+1 are stored as their respective BCD values of between 0 and 9999.
- Operation results are rounded off from the fifth decimal place. For this reason, the fourth decimal place has an error of ±1.

■BDSQR

- Calculates the square root of the values designated by (S) and (S)+1 and stores the results at the device designated by (D).

$$\sqrt{((S)+1)(S)} = (D).(D)+1$$

(S)+1, (S): 2-word data; (D): Integer part, (D)+1: Decimal fraction part

- BCD value of a maximum of 8 digits (0 to 99999999) can be designated by (S) and (S)+1.
- The operation results of (D) and (D)+1 are stored as their respective BCD values of between 0 and 9999.
- Operation results are rounded off from the fifth decimal place. For this reason, the fourth decimal place has an error of ±1.

Operation error

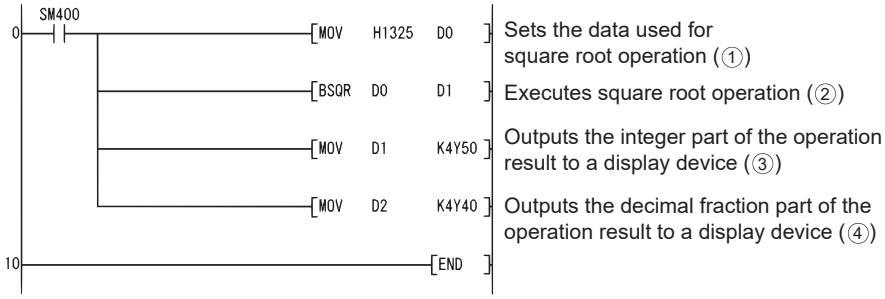
- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4100	The data specified in (S) is not a BCD value.	—	○	○	○	○	○

Program example

- The following program calculates the square root of BCD value 1325 and outputs the integer part to the 4 BCD digits from Y50 to Y5F, and the decimal fraction part to the 4 BCD digits from Y40 to Y4F.

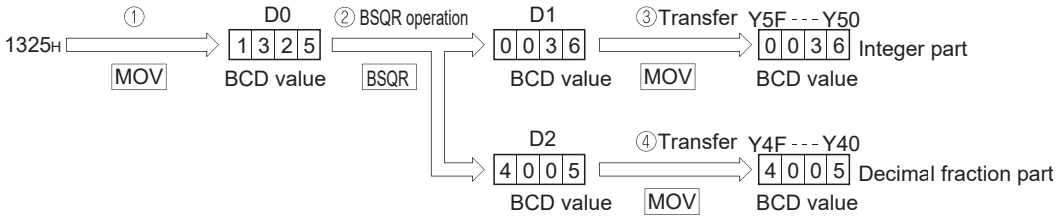
[Ladder Mode]



[List Mode]

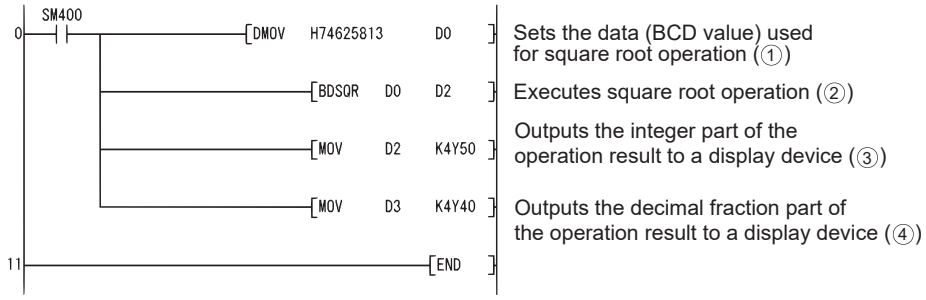
Step	Instruction	Device	
0	LD	SM400	
1	MOV	H1325	D0
3	BSQR	D0	D1
6	MOV	D1	K4Y50
8	MOV	D2	K4Y40
10	END		

[Operation]



• The following program calculates the square root of BCD value 74625813 and outputs the integer part of the result to the 4 BCD digits at Y50 to Y5F, and the decimal fraction part to the 4 BCD digits from Y40 to Y4F.

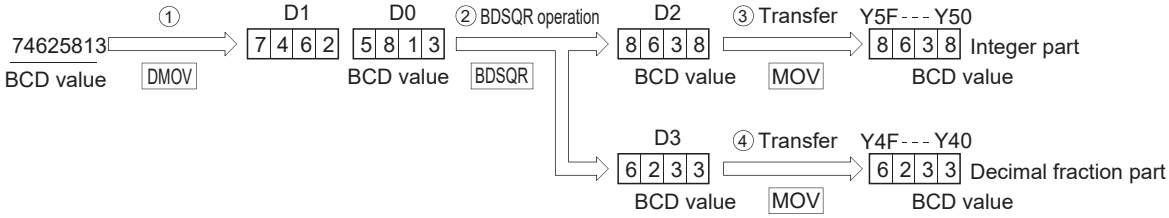
[Ladder Mode]



[List Mode]

Step	Instruction	Device		
0	LD	SM400		
1	DMOV	H74625813		D0
4	BDSQR	D0	D2	
7	MOV	D2	K4Y50	
9	MOV	D3	K4Y40	
11	END			

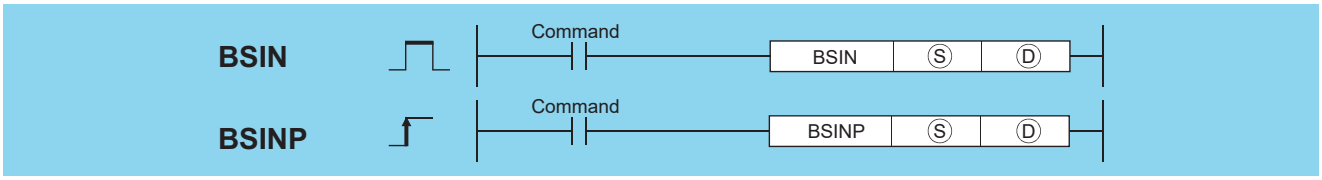
[Operation]



BCD type SIN operation

BSIN(P)

Basic | High performance | Process | Redundant | Universal | LCPU



(S): Data of which the SIN (sine) value is obtained or the number of the device where the data is stored (BCD 4 digits)
(D): Head number of the devices where the operation result will be stored (BCD 4 digits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(S)	○	○		○					—
(D)	—	○		—					—

Processing details

- Calculates the SIN (sine) value of value (angle) designated by (S), and stores the sign of the operation result in the device designated at (D), and the operation result in the devices designated at (D)+1 and (D)+2.

SIN (S) = (D) Sign | (D)+1 Integer part . (D)+2 Decimal fraction part

- The value designated at (S) is a BCD value which can be between 0 and 360 degrees (in units of degrees).
- The sign for the operation result stored in (D) will be "0" if the result is a positive value, and "1" if the result is a negative value.
- The operation results stored in (D)+1 and (D)+2 are BCD values within the range of -1.0000 and 1.0000.
- Operation results are rounded off to four decimal places.

Operation error

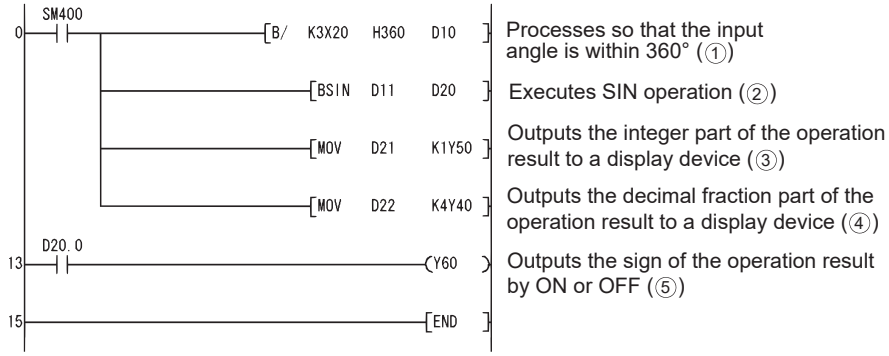
- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4100	The data specified in (S) is not a BCD value. The data specified in (S) is not within the range from 0 to 360.	—	○	○	○	○	○
4101	The points of the device specified in (D) exceeds those of the corresponding device.	—	—	—	—	○	○

Program example

- The program example below calculates the SIN of 3-digit BCD data designated by X20 to X2B, and outputs a 1-digit BCD part to the integer part from Y50 to Y53, and a 4-digit BCD fraction part from Y40 to Y4F. Y60 is turned ON if the results of the operation are negative. (If a value has been set at X20 to X2F that is greater than 360, it will be adjusted to be in the range from 0 to 360.)

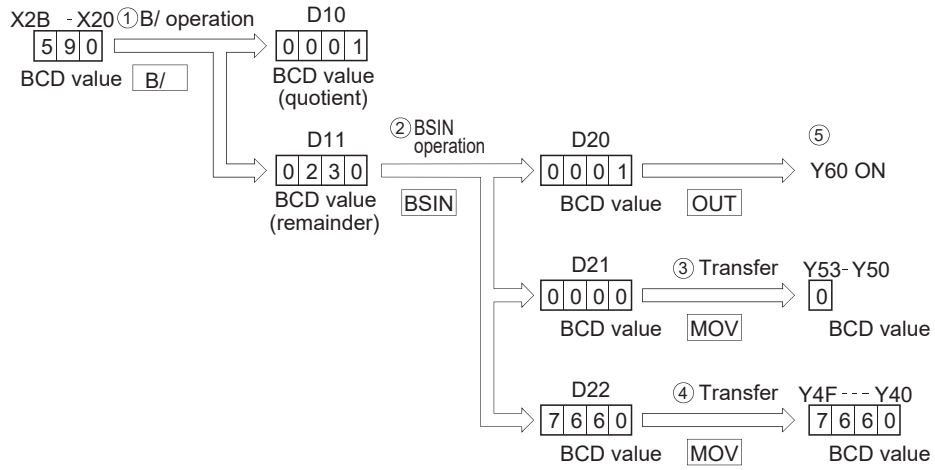
[Ladder Mode]



[List Mode]

Step	Instruction	Device		
0	LD	SM400		
1	B/	K3X20	H360	D10
5	BSIN	D11	D20	
8	MOV	D21	K1Y50	
11	MOV	D22	K4Y40	
13	LD	D20.0		
14	OUT	Y60		
15	END			

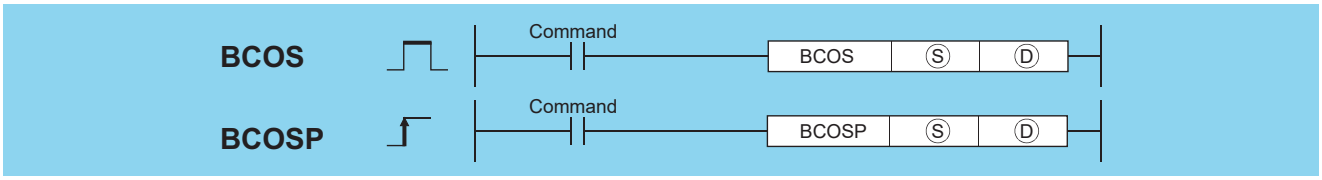
[Operations involved when value designated by X20 to X2B is 590]



BCD type COS operations

BCOS(P)

Basic | High performance | Process | Redundant | Universal | LCPU

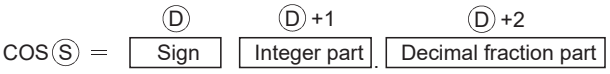


(S): Data of which the COS (cosine) value is obtained or head number of the devices where the data is stored (BCD 4 digits)
(D): Head number of the devices where the operation result will be stored (BCD 4 digits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(S)	○	○		○					—
(D)	—	○		—					—

Processing details

- Calculates COS (cosine) value of value (angle) designated by (S), then stores the sign for the operation result in the word device designated by (D), and the operation result in the word device designated by (D)+1 and (D)+2.



- The value designated at (S) is a BCD value which can be between 0 and 360 degrees (in units of degrees).
- The sign for the operation result stored in (D) will be "0" if the result is a positive value, and "1" if the result is a negative value.
- The operation results stored in (D)+1 and (D)+2 are BCD values within the range of -1.0000 and 1.0000.
- Operation results are rounded off to four decimal places.

Operation error

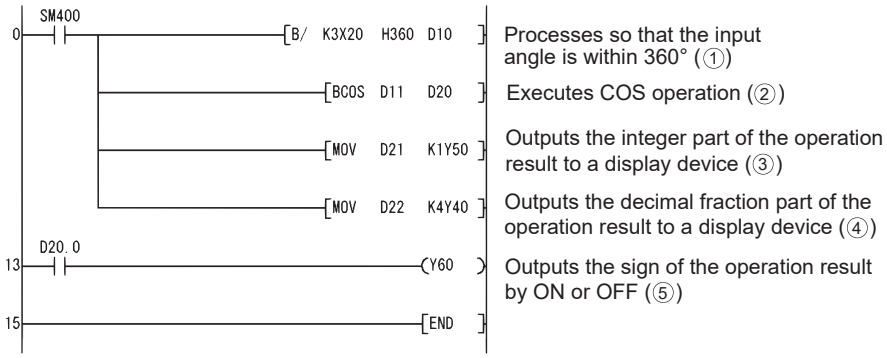
- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4100	The data specified in (S) is not a BCD value. The data specified in (S) is not in the range from 0 to 360.	—	○	○	○	○	○
4101	The points of the device specified in (D) exceeds those of the corresponding device.	—	—	—	—	○	○

Program example

- The following program calculates the cosine of the data designated by the 3 BCD digits from X20 to X2B and outputs the integer part of the result to 1 BCD digit from Y50 to Y53, and the decimal fraction part of the result to the 4 BCD digits from Y40 to Y4F. Y60 is turned ON if the results of the operation are negative.

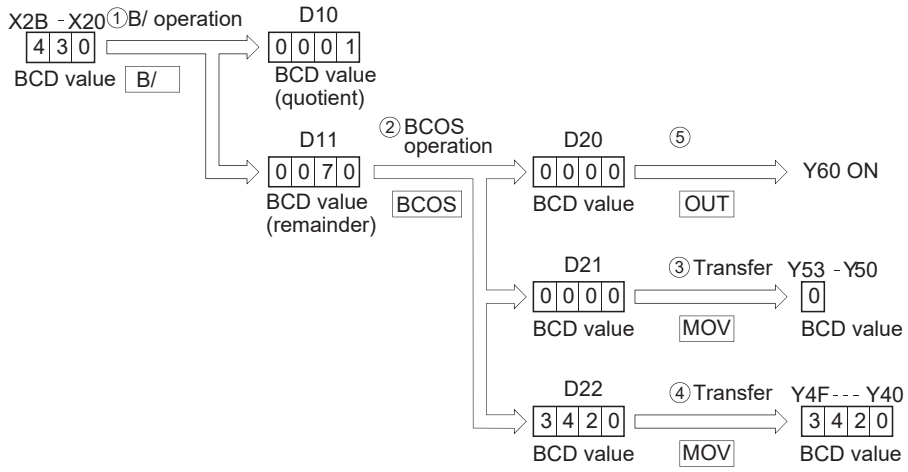
[Ladder Mode]



[List Mode]

Step	Instruction	Device		
0	LD	SM400		
1	B/	K3X20	H360	D10
5	BCOS	D11	D20	
8	MOV	D21	K1Y50	
11	MOV	D22	K4Y40	
13	LD	D20.0		
14	OUT	Y60		
15	END			

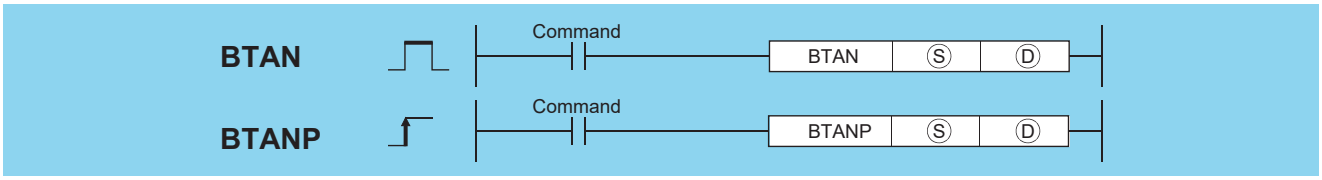
[Operations involved when value designated by X20 to X2B is 430]



BCD type TAN operation

BTAN(P)

Basic | High performance | Process | Redundant | Universal | LCPU



(S): Data of which the TAN (tangent) value is obtained or head number of the devices where the data is stored (BCD 4 digits)
(D): Head number of the devices where the operation result will be stored (BCD 4 digits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(S)	○	○		○					—
(D)	—	○		—					—

Processing details

- Calculates TAN (tangent) value for value (angle) designated by (S), and stores the sign for the operation result in the word device designated by (D), and the operation result in the word device designated by (D)+1 and (D)+2.

TAN (S) = (D) Sign | (D)+1 Integer part . (D)+2 Decimal fraction part

- The value designated at (S) is a BCD value which can be between 0 and 360 degrees (in units of degrees).
- The sign for the operation result stored in (D) will be "0" if the result is a positive value, and "1" if the result is a negative value.
- The operation results stored in (D)+1 and (D)+2 are BCD values within the range of from -57.2901 and 57.2902.
- Operation results are rounded off from the fifth decimal place.

Operation error

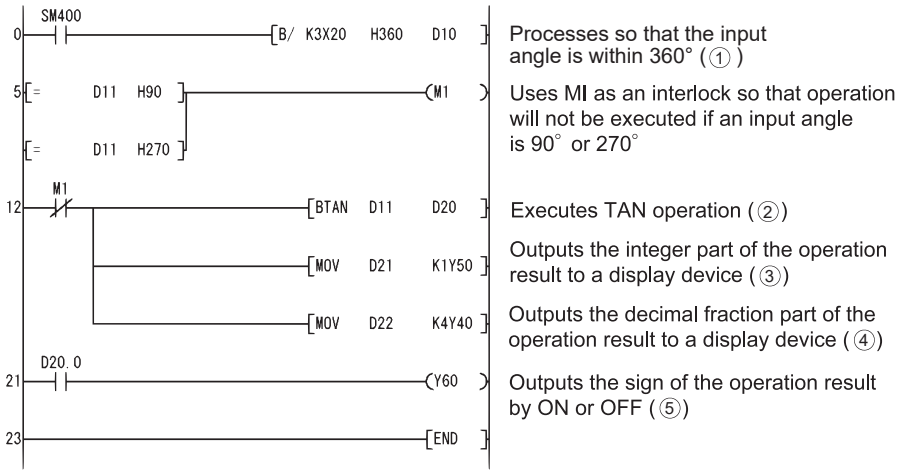
- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4100	The data specified in (S) is not a BCD value. The data specified in (S) is not in the range from 0 to 360. The data specified in (S) is 90° or 270°.	—	○	○	○	○	○
4101	The points of the device specified in (D) exceed those of the corresponding device.	—	—	—	—	○	○

Program example

- The following program calculates the tangent of the data stored in the 3 BCD digits from X20 to X2B, and stores the integer part of the results in the 4 BCD digits from Y50 to Y53, and the decimal fraction part in the 4 BCD digits from Y40 to Y4F. Y60 is turned ON if the results of the operation are negative.

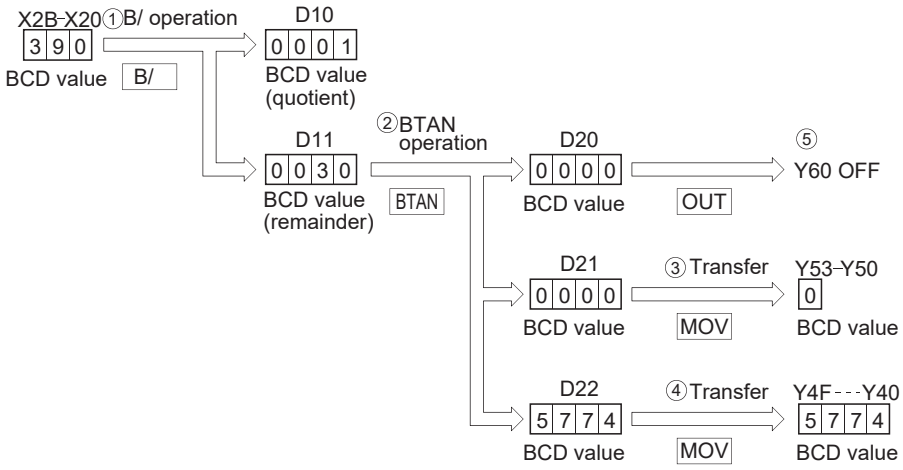
[Ladder Mode]



[List Mode]

Step	Instruction	Device		
0	LD	SM400		
1	B/	K3X20	H360	D10
5	LD=	D11	H90	
8	OR=	D11	H270	
11	OUT	M1		
12	LD I	M1		
13	BTAN	D11	D20	
16	MOV	D21	K1Y50	
19	MOV	D22	K4Y40	
21	LD	D20.0		
22	OUT	Y60		
23	END			

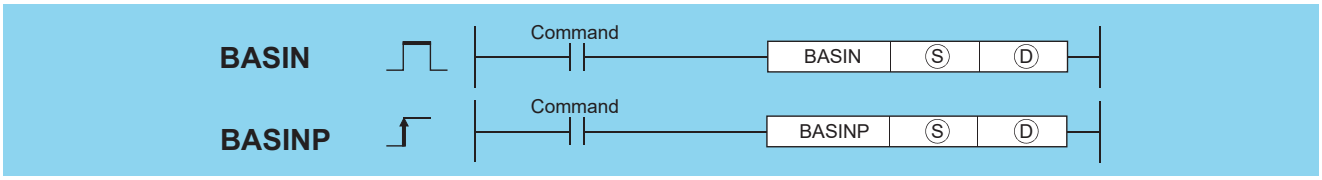
[Operations involved when X20 to X2B designate a value of 390]



BCD type arc sine operations

BASIN(P)

Basic | High performance | Process | Redundant | Universal | LCPU



(S): Number of the device where data of which the SIN^{-1} (inverse sine) value is obtained is stored (BCD 4 digits)
(D): Head number of the devices where the operation result will be stored (BCD 4 digits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant	Others
	Bit	Word		Bit	Word				
(S)	—	○		—				—	
(D)	○	○		○				—	

Processing details

- Returns the SIN^{-1} (inverse sine) value of the value designated by (S) and stores operation results (angles) at device designated by (D).

SIN^{-1} (Ⓢ Sign Ⓢ+1 Integer part . Ⓢ+2 Decimal fraction part) = Ⓓ

- A sign for the operation data is set at (S). If the operation data is a positive value, this is set at "0", and if it is a negative value, it is set at "1".
- The part before the decimal point and fraction part are stored at (S)+1 and (S)+2 respectively, as BCD values. (Settings can be between 0 and 1.0000.)
- Operation results stored at (D) are BCD values between 0 and 90 degrees, and 270 and 360 degrees (degree units).
- Calculation results are a value from which the decimal fraction part has been rounded.

Operation error

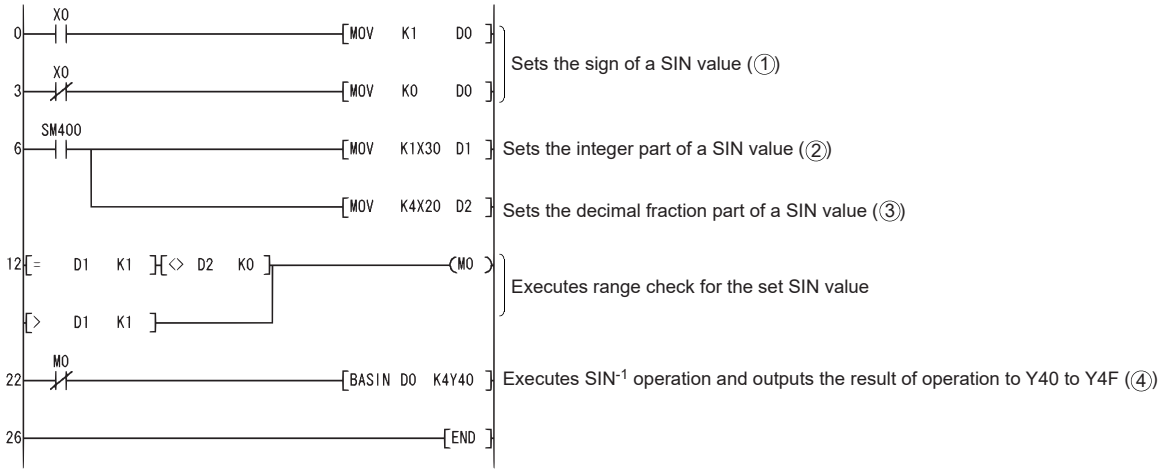
- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4100	The data specified in (S) is not a BCD value. The data specified in (S) is not in the range from -1.0000 to 1.0000.	—	○	○	○	○	○
4101	The points of the device specified in (S) exceed those of the corresponding device.	—	—	—	—	○	○

Program example

- The following program performs a SIN^{-1} operation on the sign (positive when X0 is OFF, and negative when X0 is ON), the BCD 1-digit integer part from X30 to X33 and the BCD 4-digit decimal fraction part from X20 to X2F, and outputs the calculated angle in 4 BCD digits from Y40 to Y4F.

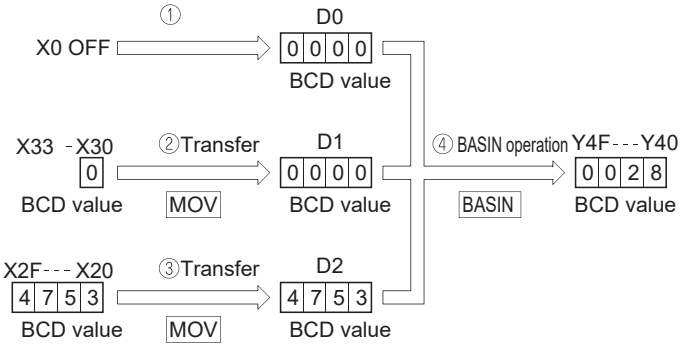
[Ladder Mode]



[List Mode]

Step	Instruction	Device	
0	LD	X0	
1	MOV	K1	D0
3	LDI	X0	
4	MOV	K0	D0
6	LD	SM400	
7	MOV	K1X30	D1
10	MOV	K4X20	D2
12	LD=	D1	K1
15	AND<>	D2	K0
18	OR>	D1	K1
21	OUT	M0	
22	LDI	M0	
23	BASIN	D0	K4Y40
26	END		

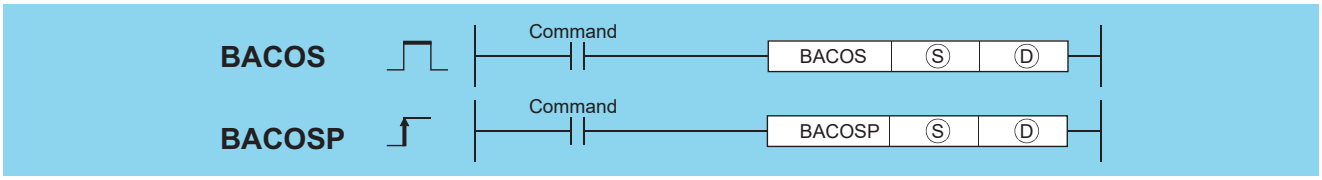
[Operations involved when X20 to X33 designates value of 0.4753]



BCD type arc cosine operation

BACOS(P)

Basic | High performance | Process | Redundant | Universal | LCPU



(S): Number of the device where data of which the COS^{-1} (inverse cosine) value is obtained is stored (BCD 4 digits)
(D): Head number of the devices where the operation result will be stored (BCD 4 digits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant	Others
	Bit	Word		Bit	Word				
(S)	—	○		—				—	
(D)	○	○		○				—	

Processing details

- Returns the COS^{-1} (inverse cosine) value of the value designated by (S), and stores operation results at device designated by (D).

$$COS^{-1}\left(\underset{\text{Sign}}{\boxed{\text{S}}}\quad\underset{\text{Integer part}}{\boxed{\text{S}+1}}.\underset{\text{Decimal fraction part}}{\boxed{\text{S}+2}}\right)=\text{D}$$

- A sign for the operation data is set at (S). If the operation data is a positive value, this is set at "0", and if it is a negative value, it is set at "1".
- The part before the decimal point and fraction part are stored at (S)+1 and (S)+2 respectively, as BCD values. (Settings can be between 0 and 1.0000.)
- The operation results stored at (D) will be a BCD value in the range of between 0 and 180° (degree units).
- Calculation results are a value from which the decimal fraction part has been rounded.

Operation error

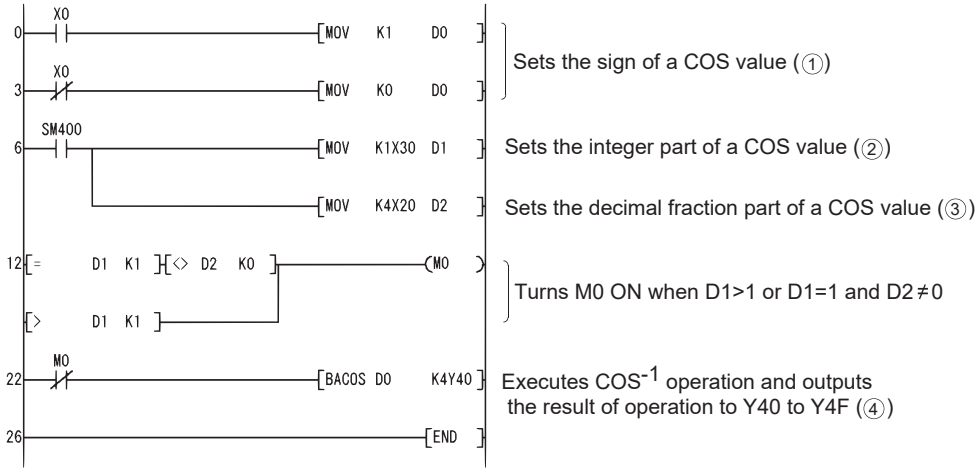
- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4100	The operation data specified in (S) is not a BCD value. The operation data specified in (S) is not in the range from -1.0000 to 1.0000.	—	○	○	○	○	○
4101	The points of the device specified in (S) exceed those of the corresponding device.	—	—	—	—	○	○

Program example

- The following program performs a COS^{-1} operation on the sign (positive when X0 is OFF, and negative when X0 is ON), the BCD 1-digit integer part from X30 to X33 and the BCD 4-digit decimal fraction part from X20 to X2F, and outputs the calculated angle in 4 BCD digits from Y40 to Y4F.

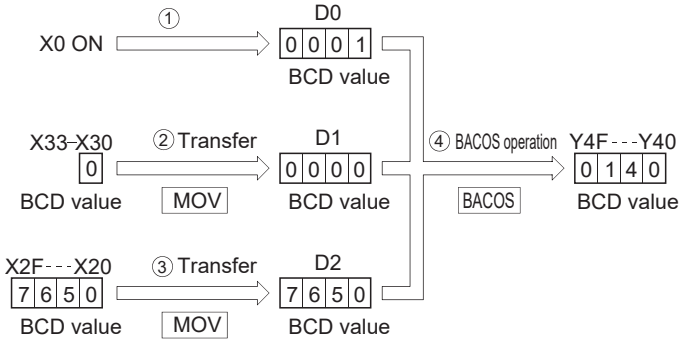
[Ladder Mode]



[List Mode]

Step	Instruction	Device	
0	LD	X0	
1	MOV	K1	D0
3	LD I	X0	
4	MOV	K0	D0
6	LD	SM400	
7	MOV	K1X30	D1
10	MOV	K4X20	D2
12	LD=	D1	K1
15	AND<>	D2	K0
18	OR>	D1	K1
21	OUT	M0	
22	LD I	M0	
23	BACOS	D0	K4Y40
26	END		

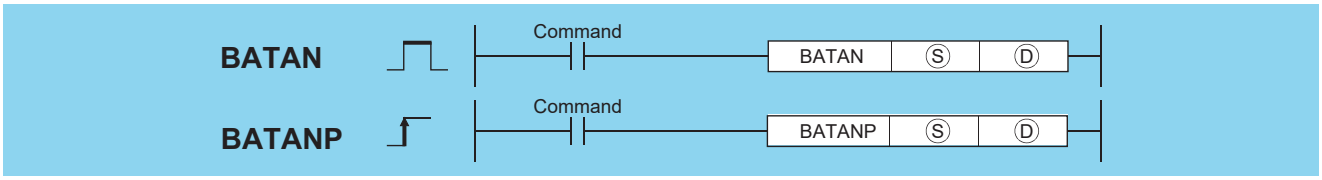
[Operations involved if X0 and X20 to X33 designate a value of -0.7650]



BCD type arc tangent operations

BATAN(P)

Basic | High performance | Process | Redundant | Universal | LCPU



(S): Number of the device where data of which the TAN^{-1} (inverse tangent) value is obtained is stored (BCD 4 digits)
(D): Head number of the devices where the operation result will be stored (BCD 4 digits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant	Others
	Bit	Word		Bit	Word				
(S)	—	○		—				—	
(D)	○	○		○				—	

Processing details

- Performs TAN^{-1} (inverse tangent) on value designated by (S) and stores operation results (angles) at device designated by (D).

TAN^{-1} ((S) Sign | (S)+1 Integer part . (S)+2 Decimal fraction part) = (D)

- A sign for the operation data is set at (S). If the operation data is a positive value, this is set at "0", and if it is a negative value, it is set at "1".
- The part before the decimal point and fraction part are stored at (S)+1 and (S)+2 respectively, as BCD values. (Values from 0 to 9999.9999 can be set.)
- Operation results stored at (D) are BCD values between 0 and 90 degrees, and 270 and 360 degrees (degree units).
- Calculation results are a value from which the decimal fraction part has been rounded.

Operation error

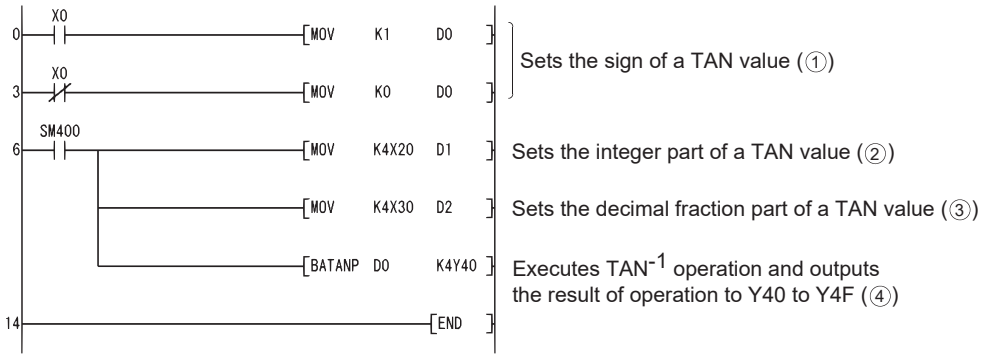
- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4100	The operation data specified in (S) is not a BCD value.	—	○	○	○	○	○
4101	The points of the device specified in (S) exceed those of the corresponding device.	—	—	—	—	○	○

Program example

- The following program performs a TAN^{-1} operation on the sign (positive when X0 is OFF, and negative when X0 is ON), the BCD 4-digit integer part from X20 to X2F and the BCD 4-digit decimal fraction part from X30 to X3F, and outputs the calculated angle in 4 BCD digits from Y40 to Y4F.

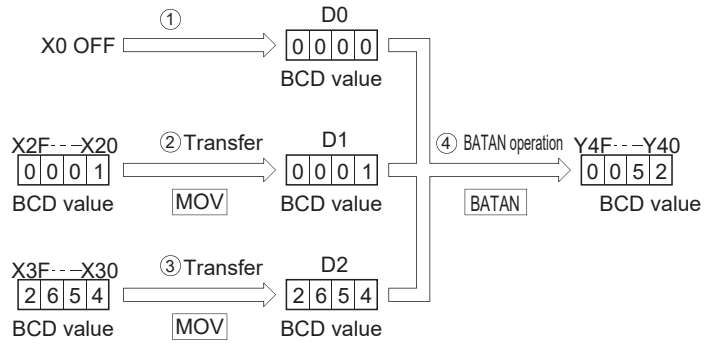
[Ladder Mode]



[List Mode]

Step	Instruction	Device	
0	LD	X0	
1	MOV	K1	D0
3	LDI	X0	
4	MOV	K0	D0
6	LD	SM400	
7	MOV	K4X20	D1
9	MOV	K4X30	D2
11	BATANP	D0	K4Y40
14	END		

[Operations involved when X0 and X20 to X2F designate a value of 1.2654]

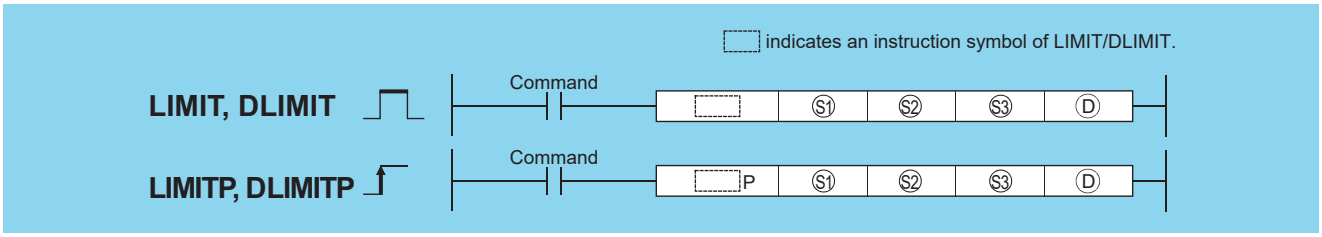


7.13 Data Control Instructions

Upper and lower limit controls for BIN 16-bit data, upper and lower limit controls for BIN 32-bit data

LIMIT(P), DLIMIT(P)

Basic | High performance | Process | Redundant | Universal | LCPU



(S1): Lower limit value (minimum output threshold value) (BIN 16/32 bits)
(S2): Upper limit value (maximum output threshold value) (BIN 16/32 bits)
(S3): Input value to be controlled by the upper and lower limit control (BIN 16/32 bits)
(D): Head number of the devices where the output value controlled by the upper and lower limit control will be stored (BIN 16/32 bits)

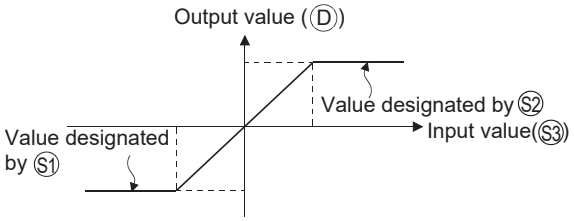
Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(S1)	○							○	—
(S2)	○							○	—
(S3)	○							○	—
(D)	○							—	—

Processing details

■LIMIT

- Controls the output value to be stored at the device designated by (D) by checking whether the input value (BIN 16 bits) designated by (S3) is within the range of upper and lower limit values specified by (S1) and (S2) or not. Output value is controlled in the way shown below:

Condition	Result
Lower limit value (S1) > Input value (S3)	Lower limit value (S1) → Output value (D)
Upper limit value (S2) < Input value (S3)	Upper limit value (S2) → Output value (D)
Lower limit value (S1) ≤ Input value (S3) ≤ Upper limit value (S2)	Input value (S3) → Output value (D)

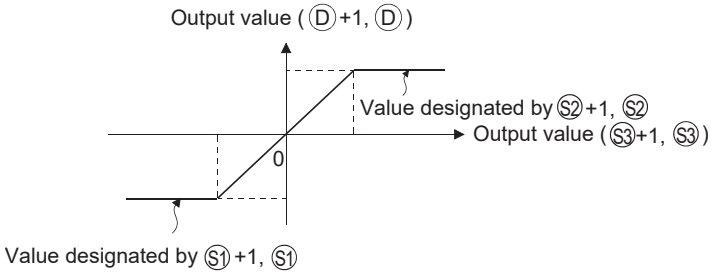


- Values in the range from -32768 and 32767 can be designated at (S1), (S2), and (S3).
- When control based only on upper limit values is performed, the lower limit value designated at (S1) is set at "-32678".
- When control based only on lower limit values is performed, the upper limit value designated at (S2) is set at "32767".

■DLIMIT

- The function controls the output value to be stored at the device designated by ((D), (D)+1) by checking whether the input value (BIN 32 bits) designated by ((S3), (S3)+1) is within the range of upper and lower limit values specified by ((S1), (S1)+1) and ((S2), (S2)+1) or not.

Condition	Result
Lower limit value ((S1)+1, (S1)) > Input value ((S3)+1, (S3))	Lower limit value ((S1)+1, (S1)) → Output value ((D)+1, (D))
Upper limit value ((S2)+1, (S2)) < Input value ((S3)+1, (S3))	Upper limit value ((S2)+1, (S2)) → Output value ((D)+1, (D))
Lower limit value ((S1)+1, (S1)) ≤ Input value ((S3)+1, (S3)) ≤ Upper limit value ((S2)+1, (S2))	Input value ((S3)+1, (S3)) → Output value ((D)+1, (D))



- The values designated by ((S1), (S1)+1), ((S2), (S2)+1), or ((S3), (S3)+1) are within the range of -2147483648 to 2147483647.
- To perform controls based only on the upper limit value, set the lower limit value designated by ((S1), (S1)+1) to "-2147483648".
- To perform controls based only on the lower limit value, set the upper limit value designated by ((S2), (S2)+1) to "2147483647".

Operation error

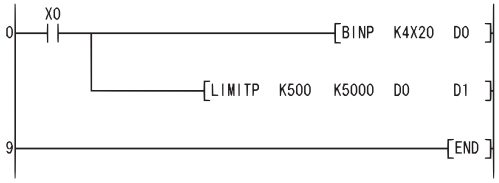
- In the following case, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4100	The lower limit value specified in (S1) is greater than the upper limit value specified in (S2).	○	○	○	○	○	○

Program example

- The following program conducts limit controls from 500 to 5000 on the data set as BCD values from X20 to X2F, and stores the result at D1 when X0 is turned ON.

[Ladder Mode]



[List Mode]

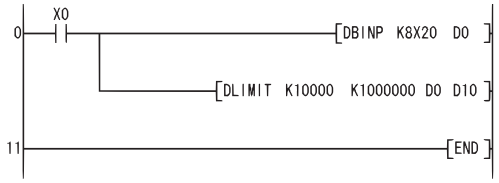
Step	Instruction	Device			
0	LD	X0			
1	BINP	K4X20	D0		
4	LIMITP	K500	K5000	D0	D1
9	END				

[Operation]

- D1 becomes 500 if D0 < 500. (example: D0 = 400 → D1 = 500)
- D1 becomes the value of D0 when 500 ≤ D0 ≤ 5000. (example: D0 = 1300 → D1 = 1300)
- D1 becomes 5000 when 5000 < D0. (example: D0 = 9600 → D1 = 5000)

- The following program conducts limit value controls from 10000 to 1000000 on the data set as BCD values from X20 to X3F when X0 is turned ON.

[Ladder Mode]



[List Mode]

Step	Instruction	Device			
0	LD	X0			
1	DBINP	K8X20	D0		
4	DLIMIT	K10000	K1000000	D0	D10
11	END				

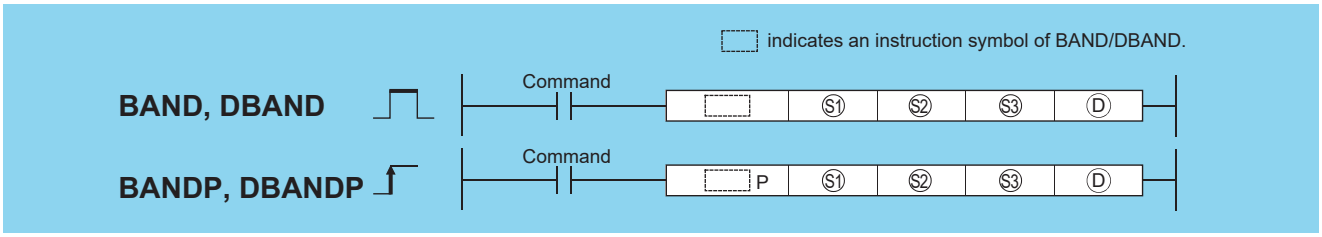
[Operation]

- (D11, D10) become 10000 if (D1, D0) are less than 10000. (example: (D1, D0) = 400 → (D11, D10) = 10000)
- (D11, D10) become the value of (D1, D0) if 10000 ≤ (D1, D0) ≤ 1000000. (example: (D1, D0) = 345678 → (D11, D10) = 345678)
- (D11, D10) become 1000000 if 1000000 < (D1, D0). (example: (D1, D0) = 9876543 → (D11, D10) = 1000000)

BIN 16-bit dead band controls, BIN 32-bit dead band controls

BAND(P), DBAND(P)

Basic | High performance | Process | Redundant | Universal | LCPU



(S1): Lower limit value of dead band (no output band) (BIN 16/32 bits)
(S2): Upper limit value of dead band (no output band) (BIN 16/32 bits)
(S3): Input value to be controlled by a dead band control (BIN 16/32 bits)
(D): Head number of the devices where the output value controlled by the dead band control will be stored (BIN 16/32 bits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(S1)	○							○	—
(S2)	○							○	—
(S3)	○							○	—
(D)	○							—	—

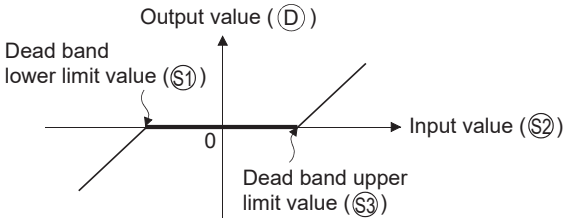
7

Processing details

■BAND

- Controls the output value to be stored at the device designated by (D) by checking whether the input value (BIN 16 bits) designated by (S3) is within the range of dead band upper and lower limit values specified by (S1) and (S2) or not. Output value is controlled in the way shown below:

Condition	Result
Lower limit value (S1) > Input value (S3)	Input value (S3) - Lower limit value (S1) → Output value (D)
Upper limit value (S2) < Input value (S3)	Input value (S3) - Upper limit value (S2) → Output value (D)
Lower limit value (S1) ≤ Input value (S3) ≤ Upper limit value (S2)	0 → Output value (D)



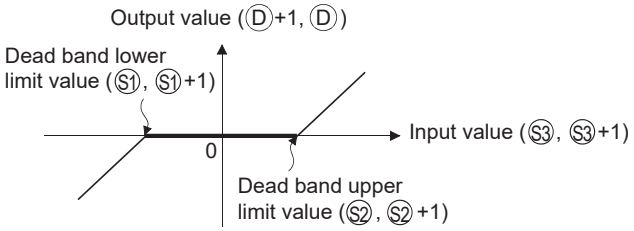
- The values that can be designated by (S1), (S2), and (S3) are in the range of from -32768 to 32767.
- The output value stored at (D) is a signed 16-bit BIN value. Therefore, if the operation results exceed the range of from -32768 to 32767, the following will take place:

Ex.
When the dead band lower limit value (S1) is 10 and the input value (S3) is -32768:
-32768 - 10 = 8000H - AH = 7FF6H = 32758

■DBAND

- Controls the output value to be stored at the device designated by (D) by checking whether the input value (BIN 32 bits) designated by ((S3), (S3)+1) is within the range of dead band upper and lower limit values specified by ((S1), (S1)+1) and ((S2), (S2)+1) or not. Output value is controlled in the way shown below:

Condition	Result
Lower limit value ((S1)+1, (S1)) > Input value ((S3)+1, (S3))	Input value ((S3)+1, (S3)) - Lower limit value ((S1)+1, (S1)) → Output value ((D)+1, (D))
Upper limit value ((S2)+1, (S2)) < Input value ((S3)+1, (S3))	Input value ((S3)+1, (S3)) - Upper limit value ((S2)+1, (S2)) → Output value ((D)+1, (D))
Lower limit value ((S1)+1, (S1)) ≤ Input value ((S3)+1, (S3)) ≤ Upper limit value ((S2)+1, (S2))	0 → Output value ((D)+1, (D))



- The values designated by ((S1), (S1)+1), ((S2), (S2)+1), or ((S3), (S3)+1) are within the range of from -2147483648 to 2147483647.
- The output value stored at (D), (D)+1 is a signed 32-bit BIN value. Therefore, if the operation results exceed the range of from -2147483648 to 2147483647, the following will take place:

Ex.

When the dead band lower limit value ((S1), (S1)+1) is 1000 and the input value ((S3), (S3)+1) is -2147483648

-2147483648-1000 = 80000000H-000003E8H = 7FFFFC18H = 2147482648

Operation error

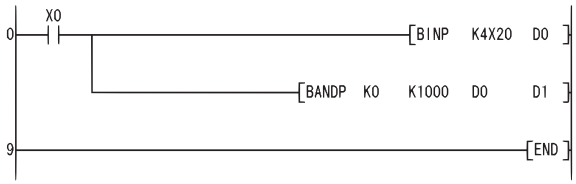
- In the following case, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4100	The lower limit value specified in (S1) is greater than the upper limit value specified in (S2).	○	○	○	○	○	○

Program example

- The following program performs the dead band control by applying the lower and upper limits of 0 and 1000 for the data set in BCD at X20 to X2F and stores the result of control at D1 when X0 is turned ON.

[Ladder Mode]



[List Mode]

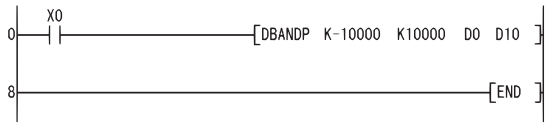
Step	Instruction	Device			
0	LD	X0			
1	BINP	K4X20	D0		
4	BANDP	K0	K1000	D0	D1
9	END				

[Operation]

- "0" is stored at D1 if 0 ≤ D0 ≤ 1000. (example: D0 = 500 → D1 = 0)
- The value of (D0) - 1000 is stored at D1 if 1000 < D0. (example: D0 = 7000 → D1 = 6000)

- The following program performs the dead band control by applying the lower and upper limits of -10000 and 10000 for the data set at D0 and D1 and stores the result of control at D10 and D11 when X0 is turned ON

[Ladder Mode]



[List Mode]

Step	Instruction	Device			
0	LD	X0			
1	DBANDP	K-10000	K10000	D0	D10
8	END				

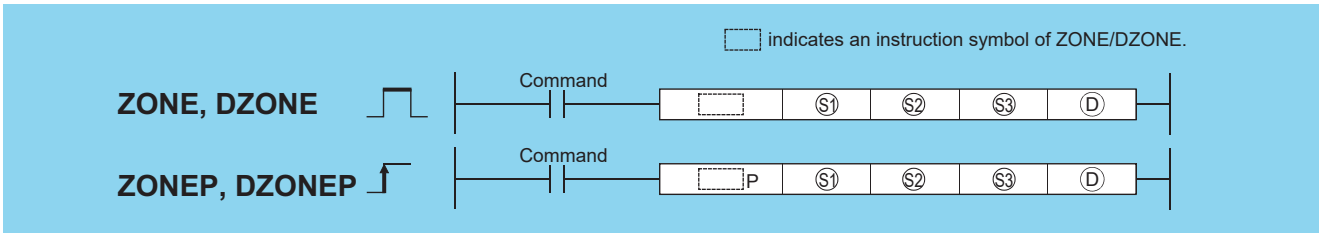
[Operation]

- The value (D1, D0) - (-10000) is stored at (D11, D10) if (D1, D0) < (-10000). (example: (D1, D0) = -12345 → (D11, D10) = -2345)
- The value 0 is stored at (D11, D10) if -10000 ≤ (D1, D0) ≤ 10000. (example: (D1, D0) = 6789 → (D11, D10) = 0)
- The value (D1, D0) -10000 is stored at (D11, D10) if 10000 < (D1, D0). (example: (D1, D0) = 50000 → (D11, D10) = 40000)

Zone control for BIN 16-bit data, zone control for BIN 32-bit data

ZONE(P), DZONE(P)

Basic | High performance | Process | Redundant | Universal | LCPU



(S1): Negative bias value to be added to an input value (BIN 16/32 bits)
(S2): Positive bias value to be added to an input value (BIN 16/32 bits)
(S3): Input value used for a zone control (BIN 16/32 bits)
(D): Head number of the devices where the output value controlled by the zone control will be stored (BIN 16/32 bits).

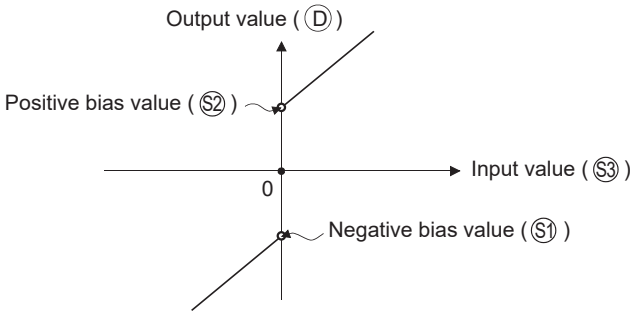
Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(S1)	○							○	—
(S2)	○							○	—
(S3)	○							○	—
(D)	○							—	—

Processing details

■ZONE

- Adds bias value designated by (S1) or (S2) to input value designated by (S3), and stores at device number designated by (D). Bias values are calculated in the following manner:

Condition	Result
When input value (S3) < 0	Input value (S3) + Negative bias value (S1) → Output value (D)
When input value (S3) = 0	0 → Output value (D)
When input value (S3) > 0	Input value (S3) + Positive bias value (S2) → Output value (D)



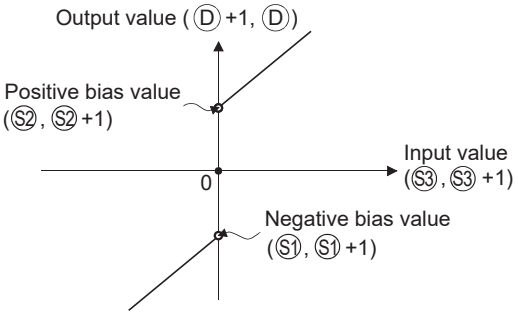
- The values that can be designated by (S1), (S2), and (S3) are in the range of from -32768 to 32767.
- The output value stored at (D) is a signed 16-bit BIN value. Therefore, if the operation results exceed the range of -32768 to 32767, the following will take place:

Ex.
When negative bias value is -100 and input value is -32768
-32768 + (-100) = 8000H + FF9CH = 7F9CH = 32668

■DZONE

- Adds bias value designated by ((S1), (S1)+1) or ((S2), (S2)+1) to input value designated by ((S3), (S3)+1), and stores the result at device number designated by ((D), (D)+1). Addition of the bias value is performed as follows:

Condition	Result
When input value ((S3)+1, (S3)) < 0	Input value ((S3)+1, (S3)) + Negative bias value ((S1)+1, (S1)) → Output value ((D)+1, (D))
When input value ((S3)+1, (S3)) = 0	0 → Output value ((D)+1, (D))
When input value ((S3)+1, (S3)) > 0	Input value ((S3)+1, (S3)) + Positive bias value ((S2)+1, (S2)) → Output value ((D)+1, (D))



- The values designated by ((S1), (S1)+1), ((S2), (S2)+1), or ((S3), (S3)+1) are within the range of from -2147483648 to 2147483647.
- The value stored at ((D), (D)+1) is a signed 32-bit BIN value. Therefore, if the operation results exceed the range of from -2147483648 to 2147483647, the following takes place:

Ex.

When negative bias value ((s1), (s1)+1) is -1000 and input value ((s3), (s3)+1) is -2147483648

-2147483648 + (-1000) = 80000000H + FFFFFC18H = 7FFFFC18H = 2147482648

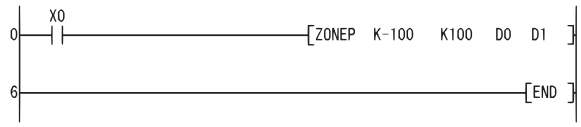
Operation error

- There is no operation error in the ZONE(P) or DZONE(P) instruction.

Program example

- The following program performs zone control by applying negative and positive bias values of -100 to 100 for the data set at D0 and stores the result of control at D1 when X0 is turned ON.

[Ladder Mode]



[List Mode]

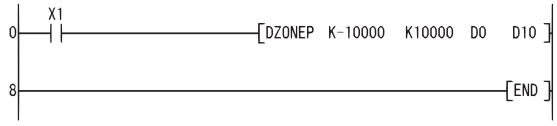
Step	Instruction	Device			
0	LD	X0			
1	ZONEP	K-100	K100	D0	D1
6	END				

[Operation]

- The value (D0) + (-100) is stored at D1 if D0 < 0. (example: D0 = -200 → D1 = -300)
- The value 0 is stored at D1 if D0 = 0.
- The value of (D0) + 100 is stored at D1 if 0 < D0. (example: D0 = 700 → D1 = 800)

- The following program performs zone control by applying negative and positive bias values of -10000 to 10000 for the data set at D0 and D1 and stores the result of control at D10 and D11 when X1 is turned ON.

[Ladder Mode]



[List Mode]

Step	Instruction	Device			
0	LD	X1			
1	DZONEP	K-10000	K10000	D0	D10
8	END				

[Operation]

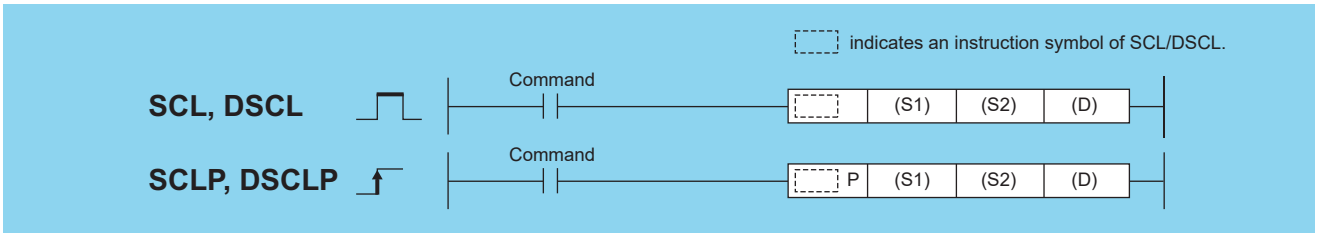
- The value (D1, D0) + (-10000) is stored at (D11, D10) if (D1, D0) < 0. (example: (D1, D0) = -12345 → (D11, D10) = -22345)
- The value 0 is stored at (D11, D10) if (D1, D0) = 0.
- The value (D1, D0) +10000 is stored at (D11, D10) if 0 < (D1, D0). (example: (D1, D0) = 50000 → (D11, D10) = 60000)

Scaling (coordinate data by point)

SCL(P), DSCL(P)

Basic | High performance | Process | Redundant | Ver. Universal | LCPU

- QnU(D)(H)CPU, QnUDE(H)CPU: the serial number (first five digits) is "10102" or later
- Q00UJCPU, Q00UCPU, Q01UCPU, QnUDVCPU, QnUDPVCPU: Supported



(S1): Input values for scaling or head number of the device where input values are stored(BIN 16/32 bits)
(S2): Head number of the devices where scaling conversion data are stored(BIN 16/32 bits)
(D): Head number of the devices where output values depending on scaling are stored(BIN 16/32 bits).

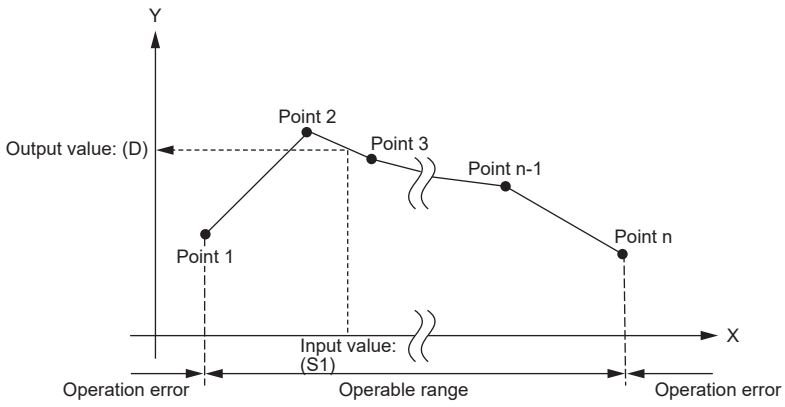
Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(S1)	○	○	○	○				○	—
(S2)	—	○	○	—				—	—
(D)	○	○	○	○				—	—

Processing details

■SCL(P)

- This instruction executes scaling for the scaling conversion data (16-bit data units) specified by (S2) with the input value specified by (S1), and then stores the operation result into the devices specified by (D). The scaling conversion is executed based on the scaling conversion data stored in the device specified by (S2) and up.

Setting item (n indicates the number of coordinates specified by (S2).)		Device assignment
Number of coordinate points		(S2)
Point 1	X coordinate	(S2)+1
	Y coordinate	(S2)+2
Point 2	X coordinate	(S2)+3
	Y coordinate	(S2)+4
⋮		
Point n	X coordinate	(S2)+2n-1
	Y coordinate	(S2)+2n

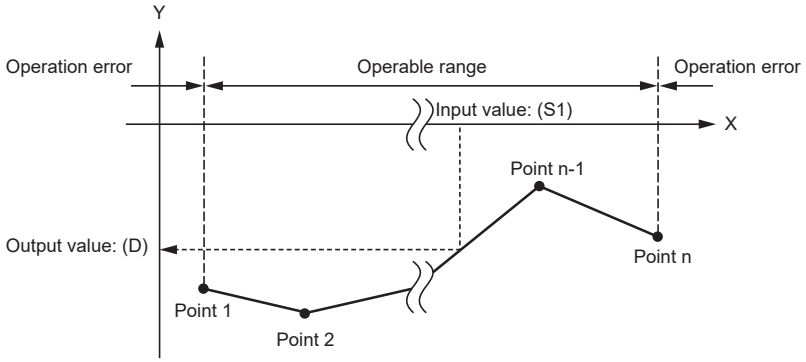


- If the value does not result in an integer, this instruction rounds the value to the whole number.
- Set the X coordinate of the scaling conversion data in ascending order.
- Set the input value (S1) within the range of the scaling conversion data (within the range of (S2) devices).
- If some specified points have same X coordinates, the Y coordinate data of the highest point number will be output.
- Specify the number of coordinate points of scaling conversion data from 1 to 32767.

■DSCL(P)

This instruction executes scaling for the scaling conversion data (32-bit data units) specified by (S2) with the input value specified (S1), and then stores the operation result into the devices specified by (D). The scaling conversion is executed based on the scaling conversion data stored in the device specified by (S2) and up.

Setting item (n indicates the number of coordinates specified by (S2).)		Device assignment
Number of coordinate points		(S2)+1, (S2)
Point 1	X coordinate	(S2)+3, (S2)+2
	Y coordinate	(S2)+5, (S2)+4
Point 2	X coordinate	(S2)+7, (S2)+6
	Y coordinate	(S2)+9, (S2)+8
⋮		
Point n	X coordinate	(S2)+4n-1, (S2)+4n-2
	Y coordinate	(S2)+4n+1, (S2)+4n



- If the value does not result in an integer, this instruction rounds the value to the whole number.
- Set the X coordinate of the scaling conversion data in ascending order.
- Set the input value (S1) within the range of the scaling conversion data (within the range of (S2) and (S2)+1 devices).
- If some specified points have same X coordinates, the Y coordinate data of the highest point number will be output.
- Specify the number of coordinate points of scaling conversion data from 1 to 32767.

Precautions

- There are two searching methods that depend on whether SM750 is on or off.

SM750	Searching method	Range of number of searches
OFF	Sequential search	1 ≤ Number of times ≤ 32767
ON	Binary search	1 ≤ Number of times ≤ 15

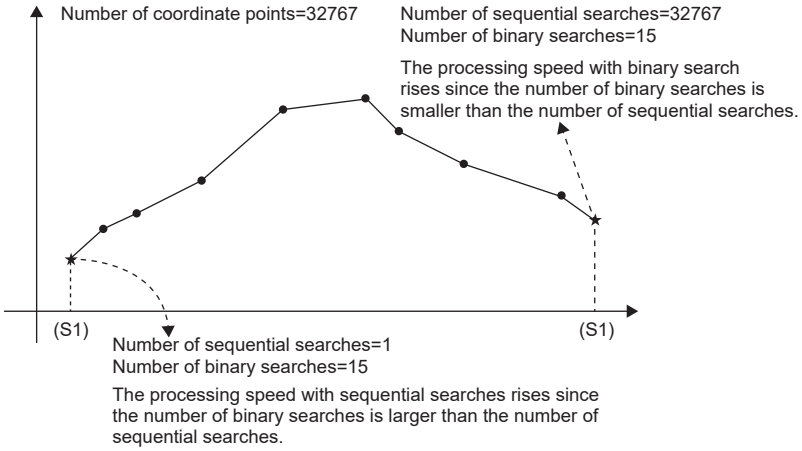
- When the scaling conversion data are set in ascending order, the searching methods change from one to the other depending on the SM750 status. Therefore, the processing speed also changes. The number of searches determines the processing speed. Fewer number of searches make the processing run faster.

• If the data processing speed with the sequential search rises:

If the number of coordinates is highest and the input value (S1) is within the coordinate range from 1 to 15 point, the number of sequential searches will be 15 or smaller. Therefore, the data processing speed with the sequential search will rise.

• If the data processing speed with the binary search rises:

If the maximum number of searches is 15 and the input value (S1) is out of the coordinate range, 16 or over, the number of binary searches will be equal to the number of sequential numbers or smaller. Therefore, the data processing speed with the binary search will rise.



Operation error

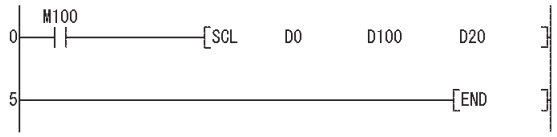
- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4100	The X coordinates of the scaling conversion data positioned before the point specified in (S1) are not set in ascending order. (However, this error is not detected when SM750 is on.) The input value specified in (S1) is not within the range of the scaling conversion data set. The number of X and Y coordinates of the device specified in (S2) is not within the range from 1 to 32767.	—	—	—	—	○	○
4101	The number of X and Y coordinates of the device specified in (S2) is not within the specified range.	—	—	—	—	○	○

Program example

- The following program executes scaling for the scaling conversion data of which the devices specified at D100 and up are set with the input value specified at D0, and then outputs the data at D20.

[Ladder Mode]



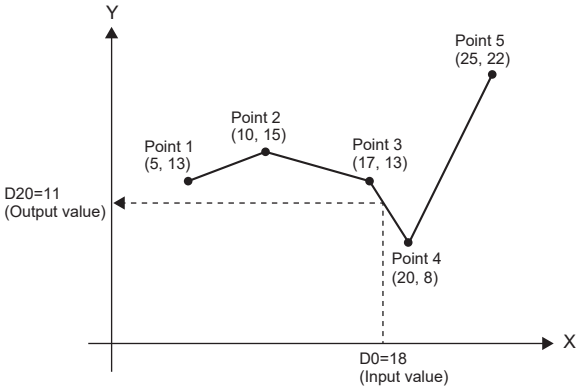
[List Mode]

Step	Instruction	Device		
0	LD	M100		
1	SCL	D0	D100	D20
5	END			

[Operation]

Scaling conversion data component

Setting item		Device	Setting contents
Number of coordinate points		D100	K5
Point 1	X coordinate	D101	K5
	Y coordinate	D102	K13
Point 2	X coordinate	D103	K10
	Y coordinate	D104	K15
Point 3	X coordinate	D105	K17
	Y coordinate	D106	K13
Point 4	X coordinate	D107	K20
	Y coordinate	D108	K8
Point 5	X coordinate	D109	K25
	Y coordinate	D110	K22

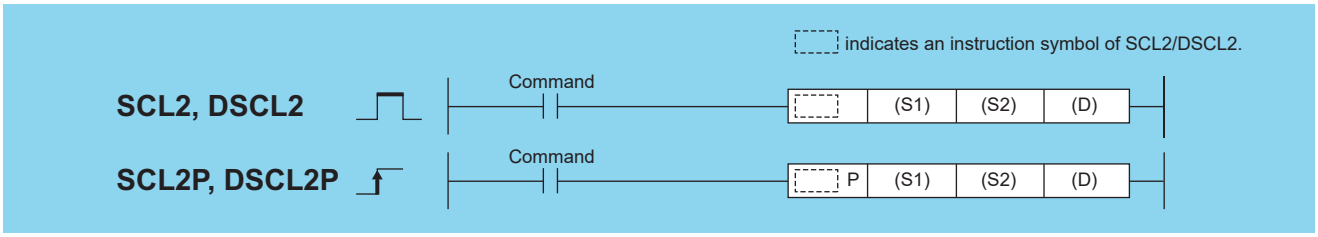


Scaling (coordinate data by X and Y)

SCL2(P), DSCL2(P)

Basic | High performance | Process | Redundant | Ver. Universal | LCPU

- QnU(D)(H)CPU, QnUDE(H)CPU: the serial number (first five digits) is "10102" or later
- Q00UJCPU, Q00UCPU, Q01UCPU, QnUDVCPU, QnUDPVCPU: Supported



(S1): Input values for scaling or head number of the device where input values are stored(BIN 16/32 bits)
(S2): Head number of the devices where scaling conversion data are stored(BIN 16/32 bits)
(D): Head number of the devices where output values depending on scaling are stored(BIN 16/32 bits).

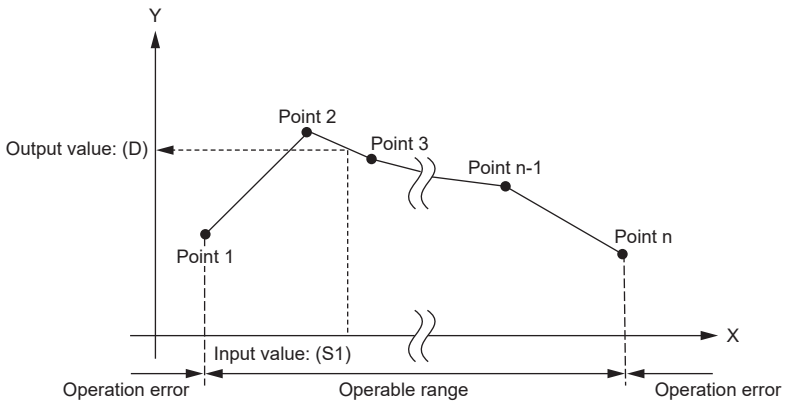
Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(S1)	—	○	○	○				○	—
(S2)	—	○	○	—				—	—
(D)	—	○	○	○				—	—

Processing details

■SCL2(P)

- This instruction executes scaling for the scaling conversion data (16-bit data units) specified by (S2) with the input value specified by (S1), and then stores the operation result into the devices specified by (D). The scaling conversion is executed based on the scaling conversion data stored in the device specified by (S2) and up.

Setting item (n indicates the number of coordinates specified by (S2).)		Device assignment
Number of coordinate points		(S2)
X coordinate	Point 1	(S2)+1
	Point 2	(S2)+2
	⋮	⋮
	Point n	(S2)+n
Y coordinate	Point 1	(S2)+n+1
	Point 2	(S2)+n+2
	⋮	⋮
	Point n	(S2)+2n

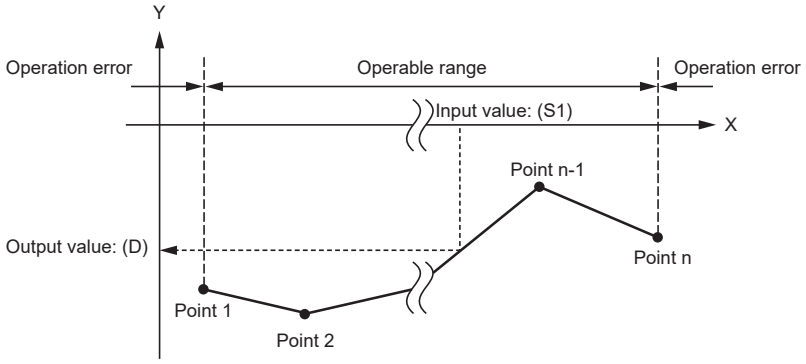


- If the value does not result in an integer, this instruction rounds the value to the whole number.
- Set the X coordinate of the scaling conversion data in ascending order.
- Set the input value (S1) within the range of the scaling conversion data (within the range of (S2) devices).
- If some specified points have same X coordinates, the Y coordinate data of the highest point number will be output.

■DSCL2(P)

- This instruction executes scaling for the scaling conversion data (32-bit data units) specified by (S2) with the input value specified by (S1), and then stores the operation result into the devices specified by (D). The scaling conversion is executed based on the scaling conversion data stored in the device specified by (S2) and up.

Setting item (n indicates the number of coordinates specified by (S2).)		Device assignment
Number of coordinate points		(S2)+1, (S2)
X coordinate	Point 1	(S2)+3, (S2)+2
	Point 2	(S2)+5, (S2)+4
	⋮	⋮
	Point n	(S2)+2n+1, (S2)+2n
Y coordinate	Point 1	(S2)+2n+3, (S2)+2n+2
	Point 2	(S2)+2n+5, (S2)+2n+4
	⋮	⋮
	Point n	(S2)+4n+1, (S2)+4n



- If the value does not result in an integer, this instruction rounds the value to the whole number.
- Set the X coordinate of the scaling conversion data in ascending order.
- Set the input value (S1) within the range of the scaling conversion data (within the range of (S2) and (S2)+1 devices).
- If some specified points have same X coordinates, the Y coordinate data of the highest point number will be output.
- Specify the number of coordinate points of scaling conversion data from 1 to 32767.

Point

When the coordinates of the scaling conversion data are set in ascending order, the searching methods change from one to the other depending on the SM750 status. Therefore, the processing speed also change. For details, refer to Page 672 Scaling (coordinate data by point).

Operation error

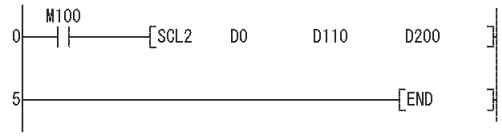
- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4100	The X coordinates are not set in ascending order. The input value specified in (S1) is not within the range of the scaling conversion data set. The number of X and Y coordinates of the device specified in (S2) is not within the range from 1 to 32767.	—	—	—	—	○	○
4101	The number of X and Y coordinates of the device specified in (S2) exceeds the specified range.	—	—	—	—	○	○

Program example

- The following program executes scaling for the scaling conversion data of which the devices specified at D110 and up are set with the input value specified at D0, and then outputs the data at D200.

[Ladder Mode]



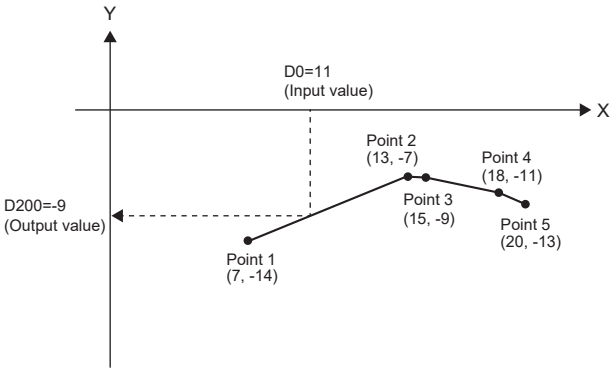
[List Mode]

Step	Instruction	Device		
0	LD	M100		
1	SCL2	D0	D110	D200
5	END			

[Operation]

Scaling conversion data component

Setting item		Device	Setting contents
Number of coordinate points		D110	K5
X coordinate	Point 1	D111	K7
	Point 2	D112	K13
	Point 3	D113	K15
	Point 4	D114	K18
	Point 5	D115	K20
Y coordinate	Point 1	D116	K-14
	Point 2	D117	K-7
	Point 3	D118	K-15
	Point 4	D119	K-11
	Point 5	D120	K-18



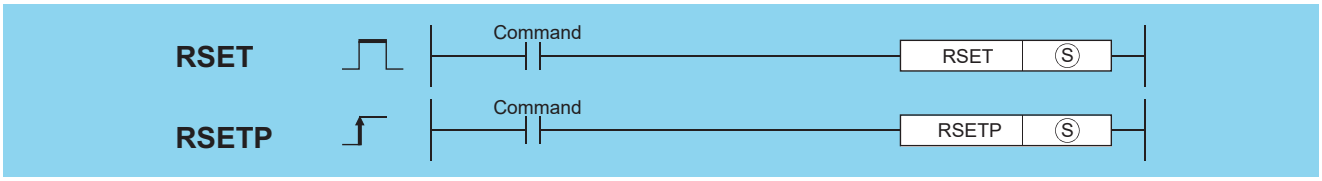
7.14 File Register Switching Instructions

Switching file register block numbers

RSET(P)

Ver. Basic | High performance | Process | Redundant | Ver. Universal | LCPU

- Q00JCPU cannot be used.
- Universal model QCPU: Models other than Q00UJCPU



(S): Block number data used to change the block number or the number of the device where the block number data is stored (BIN 16 bits)

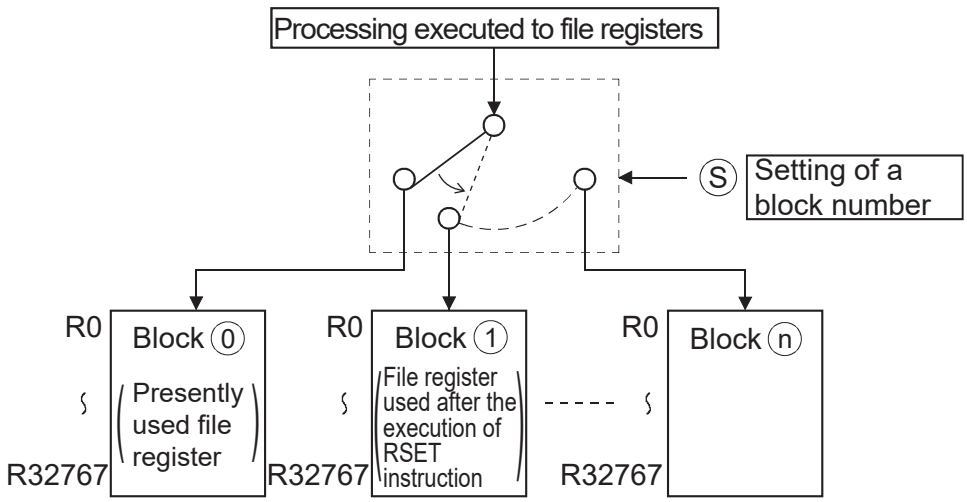
Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(S)	○								—

Processing details

- Changes the file register block number used in the program to the block number stored in the device designated at (S). Following the block number change, all file registers used in the sequence program are processed to the file register of the block number after the change.

Ex.

When switching block number from block No. 0 to block No. 1



Point

When a file register (R) is refreshed and the block No. of the file register is switched with the RSET instruction, follow restrictions.

For the restrictions on file registers, refer to Page 127 Precautions for Use of File Registers.

Operation error

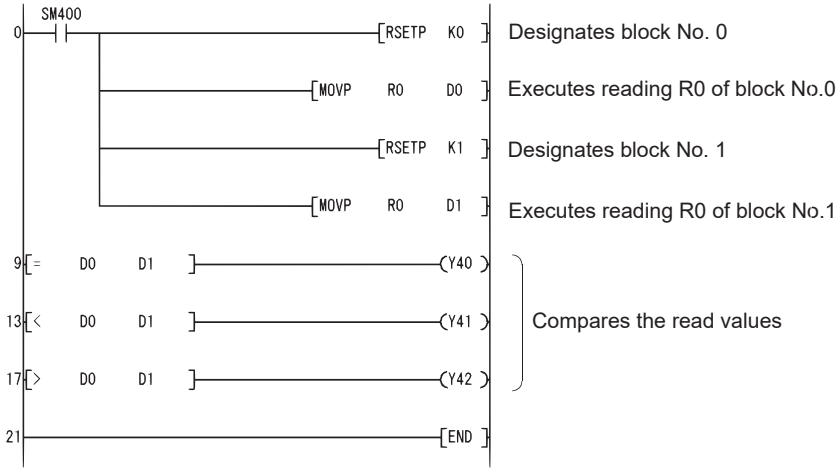
• In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4100	The block number specified in (S) does not exist.	○	○	○	○	○	○
4101	There is no file register for the specified block No.	○	○	○	○	○	○

Program example

• The following program compares R0 of block No. 0 and R0 of block No. 1.

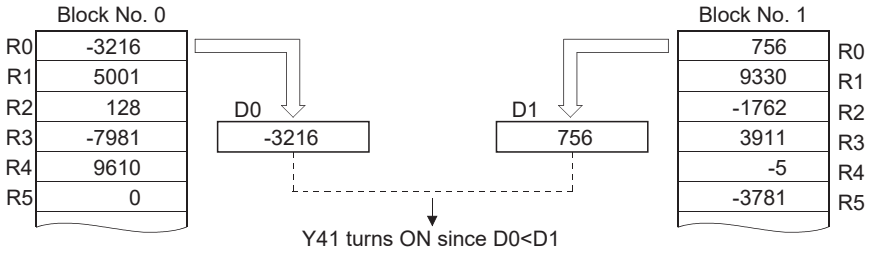
[Ladder Mode]



[List Mode]

Step	Instruction	Device	
0	LD	SM400	
1	RSETP	K0	
3	MOVP	R0	D0
5	RSETP	K1	
7	MOVP	R0	D1
9	LD=	D0	D1
12	OUT	Y40	
13	LD<	D0	D1
16	OUT	Y41	
17	LD>	D0	D1
20	OUT	Y42	
21	END		

[Operation]

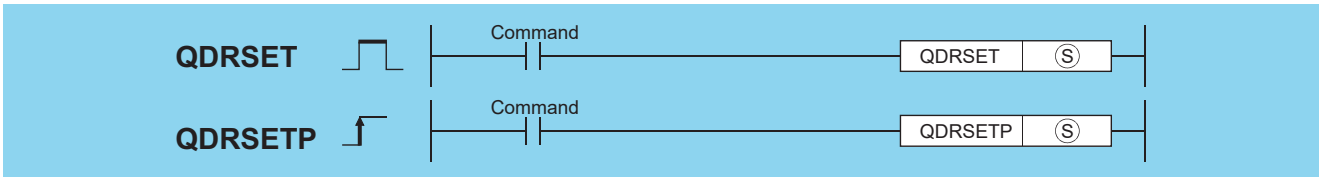


File setting for file register

QDRSET(P)

Basic | High performance | Process | Redundant | Universal (Ver.) | LCPU

• Universal model QCPU: Models other than Q00UJCPU



(S): Character string data of the drive No./file name in which the file register is set, or head number of the devices where the character string data is stored (character string)

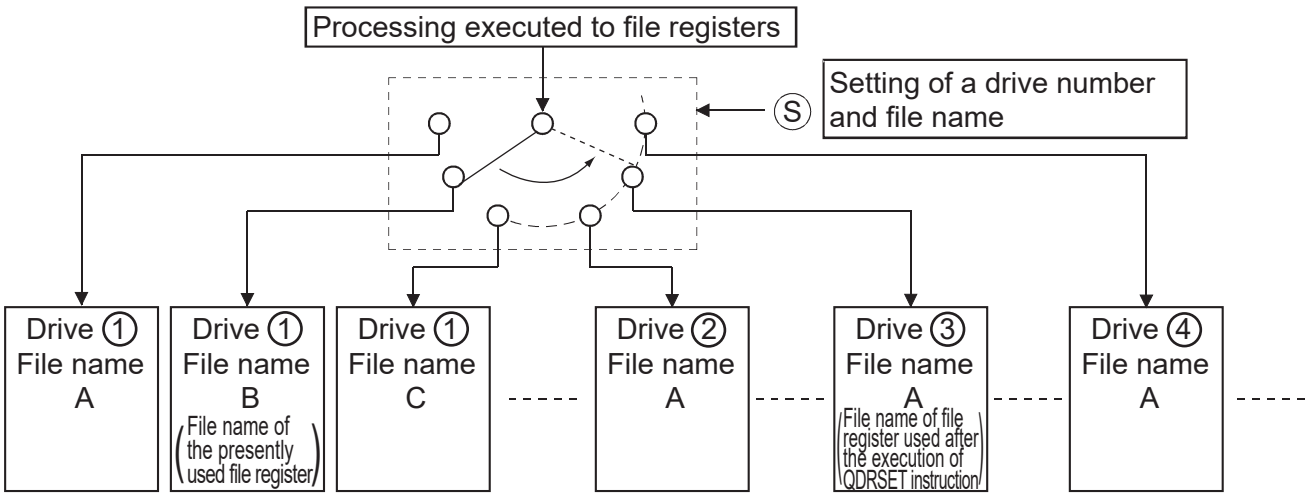
Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant $	Others
	Bit	Word		Bit	Word				
(S)	—	○		—				○	—

Processing details

• Changes the file register file name used in the program to the file name being stored at the device designated by (S). After the file names have been changed, all the file registers being used by the sequence program process the file register of the renamed file. The block No. of the file register of the renamed file is 0. Block number switches are performed by the RSET instruction.

Ex.

When switching from Drive No. 1/File name B to Drive No. 3/File name A



• Drive number can be designated from 1 to 4. (The drive number cannot be designated as drive 0 (program memory).) Note that available drives vary depending on the CPU module used. Refer to the manual of the CPU module and check the drives that can be specified.

• It is not necessary to designate the extension (.QDR) with the file name.

• A file name setting can be deleted by specifying the NULL code (00H) for the file name.

• File names specified by this instruction will be given priority even if a drive number and file name have been specified in the parameters.

Point

- If the file name is changed with the QDRSET instruction, the file name returns to the name specified by the parameter when the CPU module is switched from STOP to RUN. To maintain the file name even after the CPU mode is changed from STOP to RUN, execute the QDRSET instruction with the SM402 special relay, which turns ON during one scan when the CPU enters from STOP to RUN mode.
- For refreshing a file register, do not change the file name of the file register with the QDRSET instruction. For restrictions on file registers, refer to Page 127 Precautions for Use of File Registers.

Operation error

- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

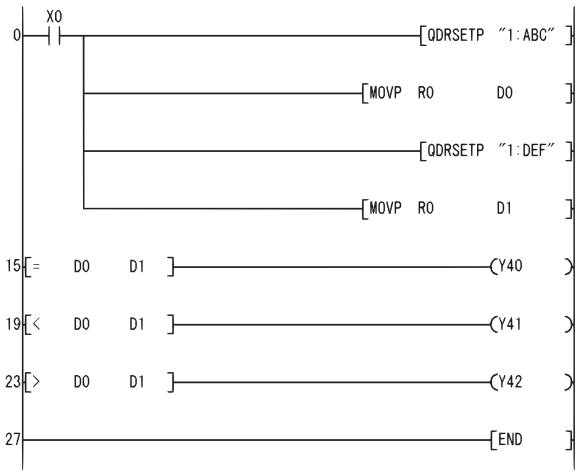
Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
2410	The file name does not exist at the drive number specified in (S).	—	○	○	○	○	—
4100	The number other than 1 or 3 is specified into the drive No.	—	—	—	—	○*1	—
	The character string designated by (S) is only the drive No.	—	—	—	—	○*1	—
4101	The character string designated by (S) exceeds 16383 characters. There is no NULL code "00H" within the range of the corresponding devices starting from the device number specified by (S).	—	—	—	—	○*1	—

*1 QnUDVCPU and QnUDPVCPU only

Program example

- The following program compares R0 of ABC in block No. 1 and R0 of DEF in block No. 1.

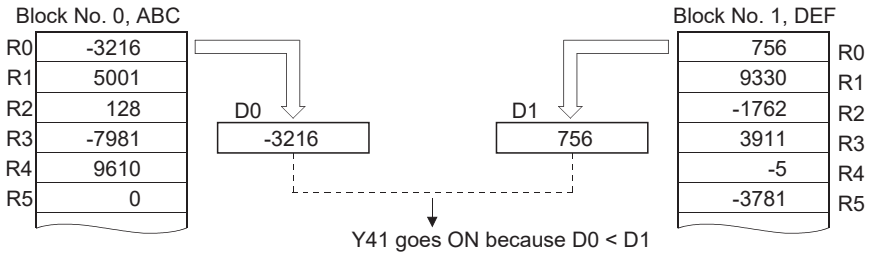
[Ladder Mode]



[List Mode]

Step	Instruction	Device	
0	LD	X0	
1	QDRSETP	"1:ABC"	
6	MOVP	R0	D0
8	QDRSETP	"1:DEF"	
13	MOVP	R0	D1
15	LD=	D0	D1
18	OUT	Y40	
19	LD<	D0	D1
22	OUT	Y41	
23	LD>	D0	D1
26	OUT	Y42	
27	END		

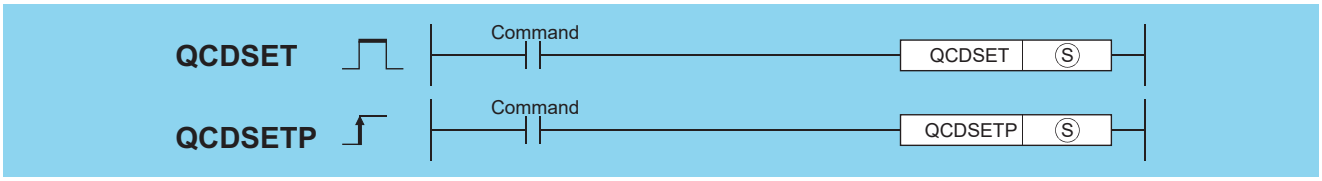
[Operation]



File setting for comments

QCDSET(P)

Basic | High performance | Process | Redundant | Universal | LCPU



(S): Character string data of the drive No./file name in which the comment file is set, or head number of the devices where the character string data is stored (character string)

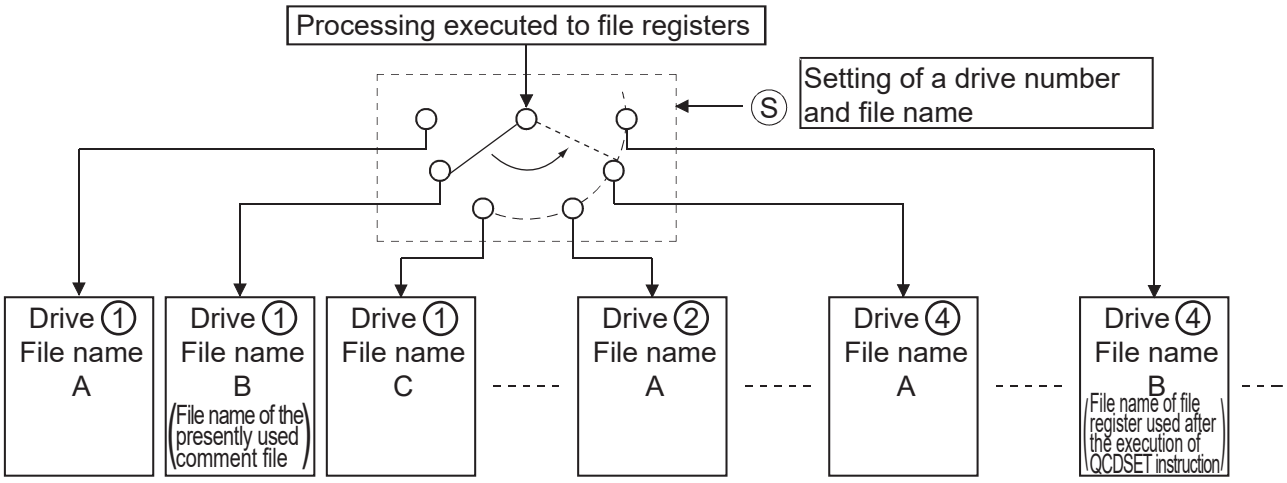
Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant $	Others
	Bit	Word		Bit	Word				
(S)	—	○		—				○	—

Processing details

- Changes the file register file name used in the program to the file name being stored at the device designated by (S). After the file name change, comment data being used by the sequence program perform processing in relation to the comment data of the file name after the change.

Ex.

When switching from Drive No. 1/File name B to Drive No. 4/File name B



- Drive number can be designated from 1 to 4. (The drive number cannot be designated as drive 0 (program memory).) Note that available drives vary depending on the CPU module used. Refer to the manual of the CPU module and check the drives that can be specified.
- It is not necessary to designate the extension (.QCD) with the file name.
- A file name setting can be deleted by specifying the NULL code (00H) for the file name.
- File names specified by this instruction will be given priority even if a drive number and file name have been specified in the parameters.

Point

If the file name is changed with the QCDSET instruction, the file name returns to the name specified by the parameter when the CPU module is switched from STOP to RUN.
To maintain the file name even after the CPU mode is changed from STOP to RUN, execute the QCDSET instruction with the SM402 special relay, which turns ON during one scan when the CPU enters from STOP to RUN mode.

Operation error

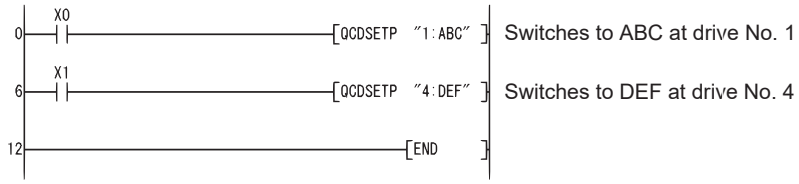
- In the following case, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
2410	The file name does not exist at the drive number specified in (S).	—	○	○	○	○	○
4101	There is no NULL code "00H" within the range of the corresponding devices starting from the device number specified by (S).	—	○	○	○	○	○

Program example

- The following program switches object file to file name ABC.QCD at drive No. 1 when X0 is ON, and to DEF.QCD at drive No. 3 when X1 is ON.

[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	X0
1	QCDSETP	"1:ABC"
6	LD	X1
7	QCDSETP	"4:DEF"
12	END	

Precautions

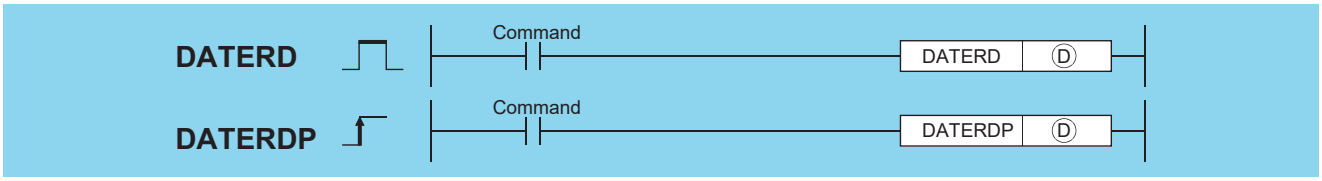
- This instruction will not be executed even when the execution command of this instruction is ON while SM721 (file access in execution) is ON for the Universal model QCPU and LCPU. Execute this instruction when SM721 is OFF.
- For the High-speed Universal model QCPU, Universal model Process CPU and LCPU, when the drive 2 (SD memory card) is specified as the drive number, this instruction cannot be executed while SM606 (SD memory card forced disable instruction) is ON. If the instruction is executed, the command will be ignored.

7.15 Clock Instructions

Reading clock data

DATERD(P)

Basic | High performance | Process | Redundant | Universal | LCPU



(D): Head number of the devices where the read clock data will be stored (BIN 16 bits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant	Others
	Bit	Word		Bit	Word				
(D)	—	○		—					

Processing details

- Reads "year, month, day, hour, minute, second, and day of week" from the clock element of the CPU module and stores it as BIN value to the device designated by (D) or later device.

Clock element ⇨

(D)	Year	(1980 to 2079)
(D)+1	Month	(1 to 12)
(D)+2	Day	(1 to 31)
(D)+3	Hour (24-hour clock)	(0 to 23)
(D)+4	Minute	(0 to 59)
(D)+5	Second	(0 to 59)
(D)+6	Day of week	(0 to 6)

- The "year" at (D) is stored as 4-digit year indication.
- The "day of week" at (D)+6 is stored as 0 to 6 to represent the days Sunday to Saturday.

Day of week	Sun	Mon	Tue	Wed	Thu	Fri	Sat
Stored data	0	1	2	3	4	5	6

- Compensation is made automatically for leap years.

Operation error

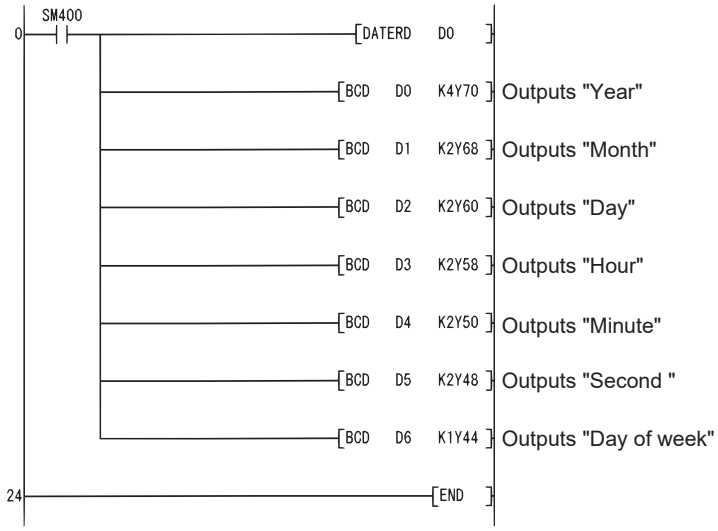
- In the following case, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4101	The range of the device specified by (D) exceeds the range of the corresponding device.	—	—	—	—	○	○

Program example

- The following program outputs the following clock data as BCD values:
  - Year: Y70 to Y7F
  - Month: Y68 to Y6F
  - Day: Y60 to Y67
  - Hour: Y58 to Y5F
  - Minute: Y50 to Y57
  - Second: Y48 to Y4F
  - Week: Y44 to Y47

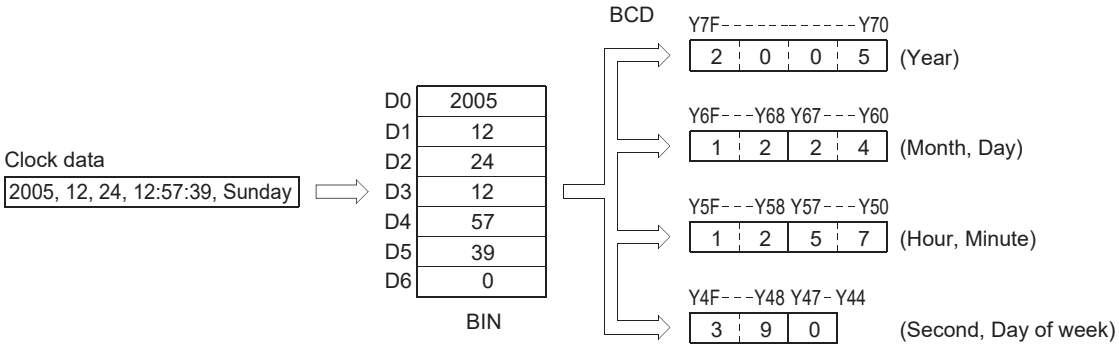
[Ladder Mode]



[List Mode]

Step	Instruction	Device	
0	LD	SM400	
1	DATERD	D0	
3	BCD	D0	K4Y70
6	BCD	D1	K2Y68
9	BCD	D2	K2Y60
12	BCD	D3	K2Y58
15	BCD	D4	K2Y50
18	BCD	D5	K2Y48
21	BCD	D6	K1Y44
24	END		

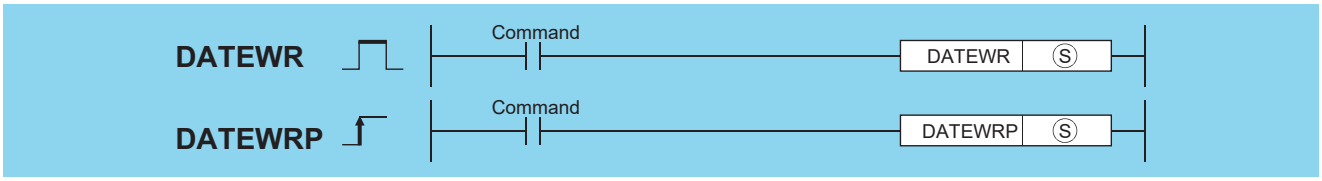
[Operation]



Writing clock data

DATEWR(P)

Basic | High performance | Process | Redundant | Universal | LCPU



(S): Head number of the devices where clock data to be written into the clock device is stored (BIN 16 bits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant	Others
	Bit	Word		Bit	Word				
(S)	—	○		—					

Processing details

- Writes clock data stored in the device number designated by (S) or later device number to the clock element of the CPU module.

(S)	Year
(S)+1	Month
(S)+2	Day
(S)+3	Hour
(S)+4	Minute
(S)+5	Second
(S)+6	Day of week

→ Clock element

- Each item is set as a BIN value.
- The "year" at (S) is designated by using four-digit year indication between 1980 to 2079.
- (S)+1 designates the "month" in values of from 1 to 12 (January to December).
- (S)+2 designates the "day" in values of from 1 to 31.
- (S)+3 designates the "hour" in values of from 0 to 23 (using 24-hour clock, from 0 hours to 23 hundred hours). (Uses the 24-hour clock.)
- (S)+4 designates the "minute" in values of from 0 to 59.
- (S)+5 designates the "second" in values of from 0 to 59.
- (S)+6 designates the "day of week" in values of from 0 to 6 (Sunday to Saturday).

Day of week	Sun	Mon	Tue	Wed	Thu	Fri	Sat
Stored data	0	1	2	3	4	5	6

Operation error

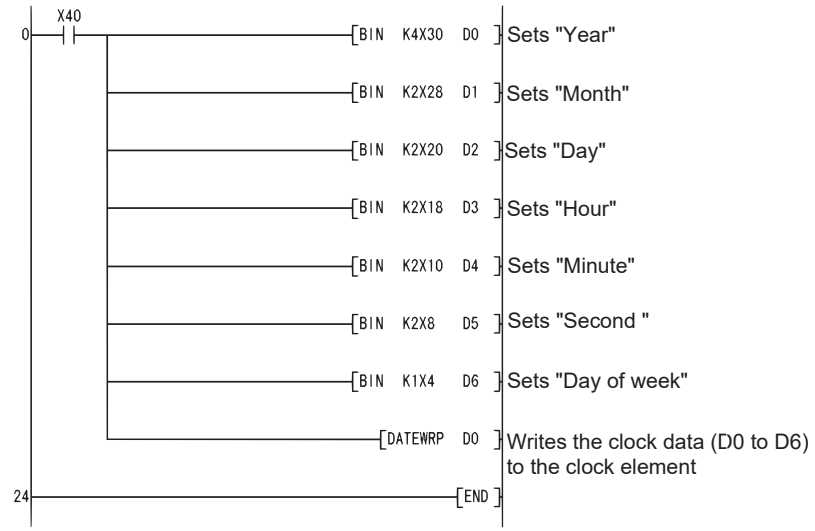
- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4100	The value outside the setting range has been set for each item.	○	○	○	○	○	○
4101	The range of the device specified by (S) exceeds the range of the corresponding device.	—	—	—	—	○	○

Program example

- The following program writes the following clock data to the clock element as BCD values when X40 is turned ON.
  - Year: X30 to X3F
  - Month: X28 to X2F
  - Day: X20 to X27
  - Hour: X18 to X1F
  - Minute: X10 to X17
  - Second: X8 to XF
  - Week: X4 to X7

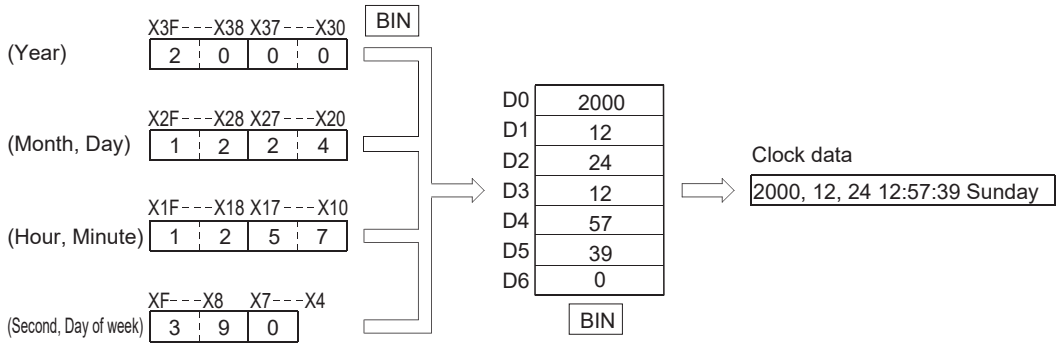
[Ladder Mode]



[List Mode]

Step	Instruction	Device	
0	LD	X40	
1	BIN	K4X30	D0
4	BIN	K2X28	D1
7	BIN	K2X20	D2
10	BIN	K2X18	D3
13	BIN	K2X10	D4
16	BIN	K2X8	D5
19	BIN	K1X4	D6
22	DATEWRP	D0	
24	END		

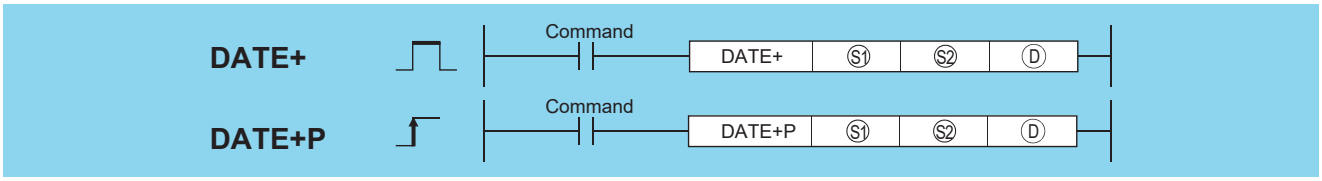
[Operation]



Clock data addition operation

DATE+(P)

Basic | High performance | Process | Redundant | Universal | LCPU



(S1): Head number of the devices where the clock data to be adjusted by addition is stored (BIN 16 bits)
(S2): Head number of the devices where the time data to be added for adjustment is stored (BIN 16 bits)
(D): Head number of the devices where the result of addition of clock (time) data will be stored (BIN 16 bits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant	Others
	Bit	Word		Bit	Word				
(S1)	—	○		—					
(S2)	—	○		—					
(D)	—	○		—					

Processing details

- Adds the time data designated by (S2) to the clock data designated by (S1), and stores the result into the area starting from the device designated by (D).

		Data range			Data range			Data range
(S1)	Hour	(0 to 23)	(S2)	Hour	(0 to 23)	(D)	Hour	(0 to 23)
(S1)+1	Minute	(0 to 59) +	(S2)+1	Minute	(0 to 59) ⇨	(D)+1	Minute	(0 to 59)
(S1)+2	Second	(0 to 59)	(S2)+2	Second	(0 to 59)	(D)+2	Second	(0 to 59)

For example, adding the time 7:48:10 to 6:32:40 would result in the following operation:

(S1)	Hour: 6		(S2)	Hour: 7		(D)	Hour: 14
(S1)+1	Minute: 32	+	(S2)+1	Minute: 48	⇨	(D)+1	Minute: 20
(S1)+2	Second: 40		(S2)+2	Second: 10		(D)+2	Second: 50

- If the results of the addition of time exceed 24 hours, 24 hours will be subtracted from the sum to make the final operation result. For example, if the time 20:20:20 were added to 14:20:30, the result would not be 34:40:50, but would instead be 10:40:50.

(S1)	Hour: 14		(S2)	Hour: 20		(D)	Hour: 10
(S1)+1	Minute: 20	+	(S2)+1	Minute: 20	⇨	(D)+1	Minute: 40
(S1)+2	Second: 30		(S2)+2	Second: 20		(D)+2	Second: 50

Point

See Page 687 Writing clock data for further information regarding the data that can be set for hours, minutes, and seconds.

Operation error

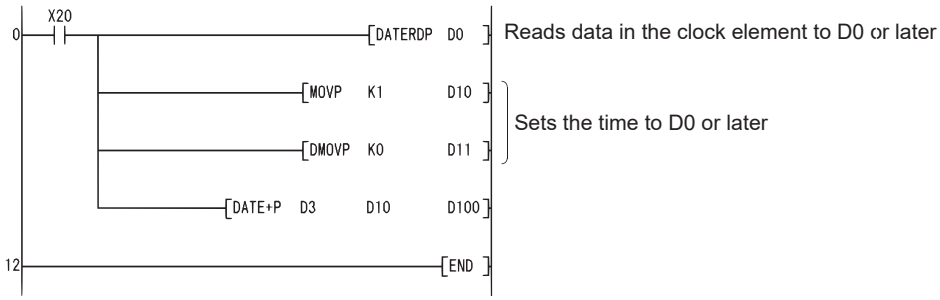
• In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4100	The value set for (S1) and (S2) is not within the setting range.	○	○	○	○	○	○
4101	The range of the device specified by (S1), (S2) or (D) exceeds the range of the corresponding device.	—	—	—	—	○	○

Program example

• The following program adds 1 hour to the clock data read from the clock element, and stores the results into the area starting from D100 when X20 is turned ON.

[Ladder Mode]

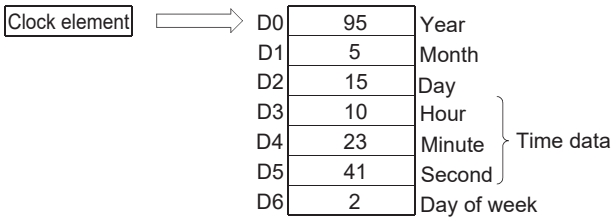


[List Mode]

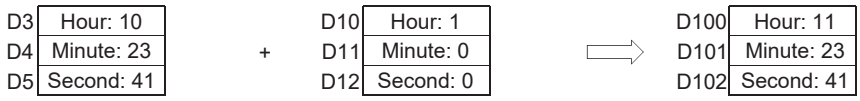
Step	Instruction	Device		
0	LD	X20		
1	DATERDP	D0		
3	MOVP	K1	D10	
5	DMOVP	K0	D11	
8	DATE+P	D3	D10	D100
12	END			

[Operation]

• Time data read operation triggered by DATERDP instruction.



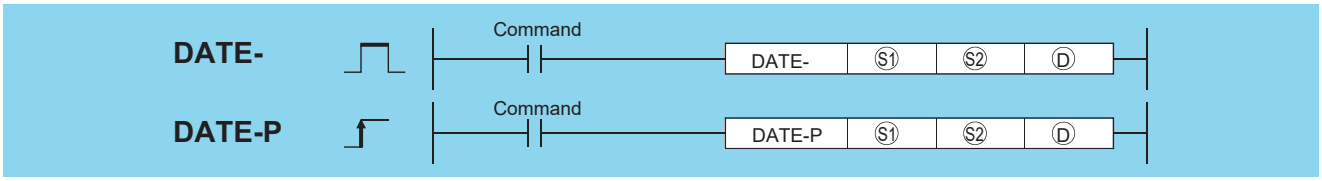
• Addition triggered by DATE+P instruction.



Clock data subtraction operation

DATE-(P)

Basic | High performance | Process | Redundant | Universal | LCPU

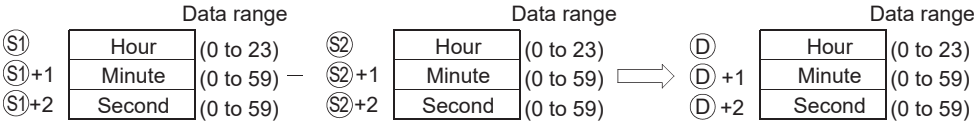


(S1): Head number of the devices where the clock time data to be adjusted by subtraction is stored (BIN 16 bits)
(S2): Head number of the devices where time data to be subtracted for adjustment is stored (BIN 16 bits)
(D): Head number of the devices where the result of subtraction of clock (time) data will be stored (BIN 16 bits)

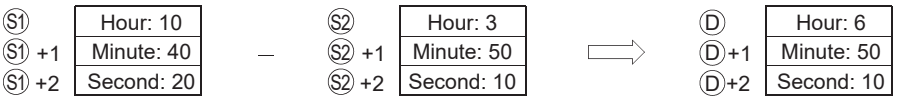
Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant	Others
	Bit	Word		Bit	Word				
(S1)	—	○		—					
(S2)	—	○		—					
(D)	—	○		—					

Processing details

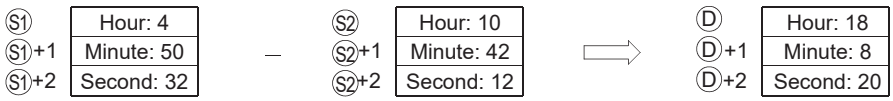
- Subtracts the time data designated by (S2) from the clock data designated by (S1), and stores the result into the area starting from the device designated by (D).



For example, if the clock time 3:50:10 were subtracted from the clock time 10:40:20, the operation would be performed as follows:



- If the subtraction results in a negative number, 24 will be added to the result to make a final operation result. For example, if the clock time 10:42:12 were subtracted from 4:50:32, the result would not be -6:8:20, but rather would be 18:8:20.



Point

See Page 687 Writing clock data for further information regarding the data that can be set for hours, minutes, and seconds.

Operation error

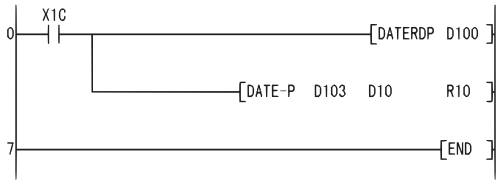
• In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4100	The value set for (S1) and (S2) is not within the setting range.	○	○	○	○	○	○
4101	The range of the device specified by (S1), (S2) or (D) exceeds the range of the corresponding device.	—	—	—	—	○	○

Program example

• The following program subtracts the time data stored in devices starting from D10 from the clock data read from the clock element when X1C is turned ON, and stores the result at devices starting from R10.

[Ladder Mode]

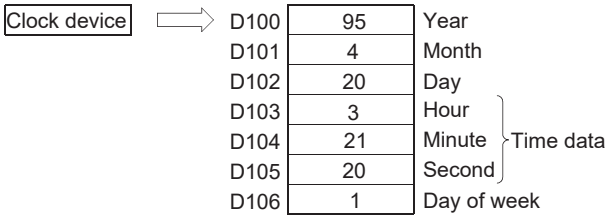


[List Mode]

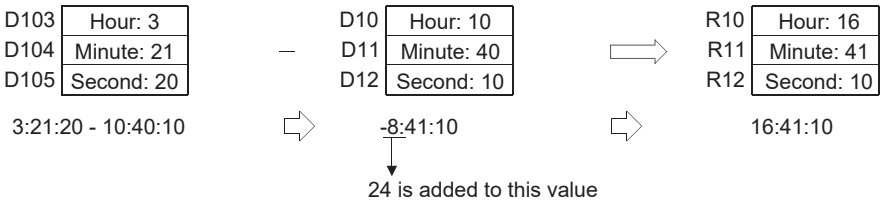
Step	Instruction	Device		
0	LD	X1C		
1	DATERDP	D100		
3	DATE-P	D103	D10	R10
7	END			

[Operation]

• Time data read operation triggered by DATERDP instruction.



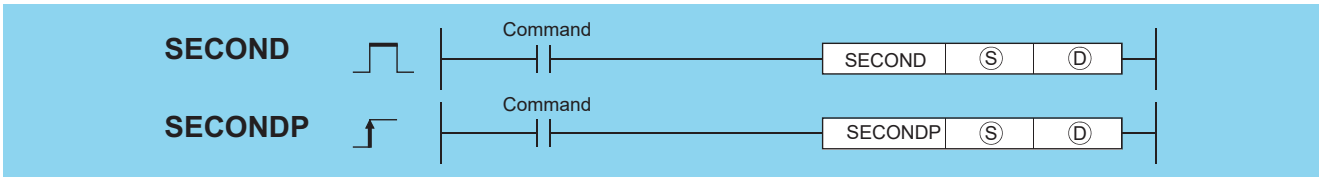
• Subtraction as triggered by DATE-P instruction (when 10 hours, 40 minutes, and 10 seconds have been designated by D10 to D12).



Time data conversion (from hour/minute/second to second)

SECOND(P)

Basic | High performance | Process | Redundant | Universal | LCPU

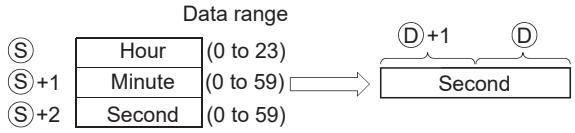


(S): Head number of the devices where the clock data before conversion is stored (BIN 16 bits)
(D): Head number of the devices where the clock data after conversion will be stored (BIN 32 bits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant	Others
	Bit	Word		Bit	Word				
(S)	—	○		—				—	
(D)	○	○		○				—	

Processing details

- Converts the time data stored in the area starting from the device designated by (S) to seconds and stores the conversion result into the device designated by (D).



For example, if the value were 4 hours, 29 minutes, and 31 seconds, the conversion would be made as follows:

Operation error

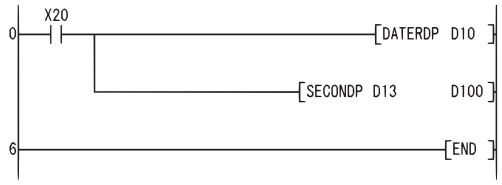
- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4100	The value set for (S) is not within the setting range.	○	○	○	○	○	○
4101	The range of the device specified by (S) exceeds the range of the corresponding device.	—	—	—	—	○	○

Program example

- The following program converts the clock time data read from the clock element into second when X20 is turned ON, and stores the result at D100 and D101.

[Ladder Mode]



[List Mode]

Step	Instruction	Device	
0	LD	X20	
1	DATERDP	D10	
3	SECONDP	D13	D100
6	END		

[Operation]

- Time data read operation triggered by DATERDP instruction.

Clock device ⇒	D10	95	Year
	D11	4	Month
	D12	20	Day
	D13	20	Hour
	D14	21	Minute (Time data)
	D15	23	Second
	D16	5	Day of week

- Conversion to seconds as triggered by the SECONDP instruction.

D13	20	⇒ D101,D100	73283
D14	21		
D15	23		

Time data conversion (from second to hour/minute/second)

HOUR(P)

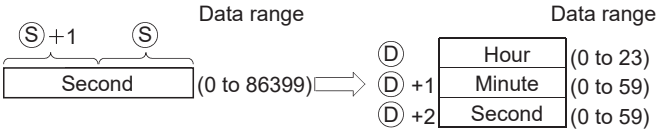
Basic | High performance | Process | Redundant | Universal | LCPU

(S): Head number of the devices where clock data before conversion is stored (BIN 32 bits)
(D): Head number of the devices where the clock data after conversion will be stored (BIN 16 bits)

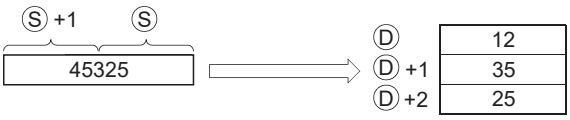
Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(S)	○	○		○				○	—
(D)	—	○		—				—	—

Processing details

- Converts the data in seconds stored in the device number designated by (S) to an hour/minute/second format, and stores the conversion result into the area starting from the device designated by (D).



For example, if 45325 seconds were the value designated, the conversion operation would be conducted as follows:



Operation error

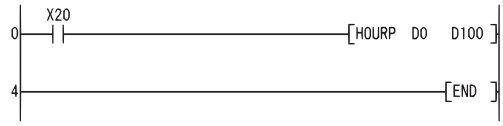
- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4100	The value set for (S) is not within the setting range.	○	○	○	○	○	○
4101	The range of the devices specified by (S) or (D) has exceeded the range of the corresponding devices.	—	—	—	—	○	○

Program example

- The following program converts the seconds stored at D0 and D1 into an hour, minute, second format, and stores the result at devices starting from D100 when X20 is turned ON.

[Ladder Mode]

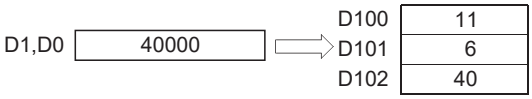


[List Mode]

Step	Instruction	Device	
0	LD	X20	
1	HOURP	D0	D100
4	END		

[Operation]

• Conversion to hour minute, and second format by the HOURP instruction (when the value 40000 seconds has been designated by D1 and D0).



Date and time data conversion (from year/month/day/time/minute/second to second)

DATE2SEC(P)

Basic | High performance | Process | Redundant | Universal (Ver.) | LCPU

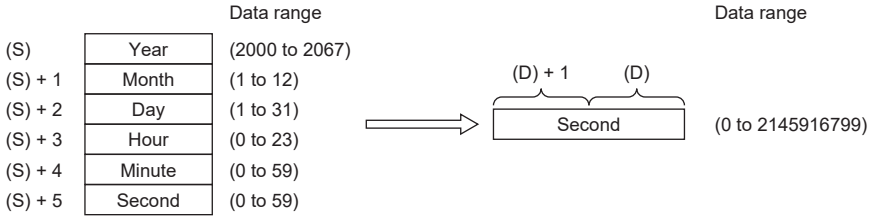
• QnUDVCPU, QnUDPVCPU: the serial number (first five digits) is "19042" or later.

(S): Head number of the devices where the date and time data before conversion is stored (Device name)
(D): Head number of the devices where the second data after conversion will be stored (BIN 32 bits)

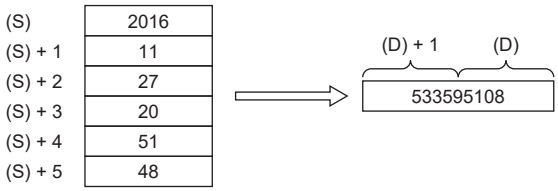
Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant	Others
	Bit	Word		Bit	Word				
(S)	—	○		—	—			—	
(D)	—	○		—	○			—	

Processing details

• Converts the date and time data stored in the area starting from the device specified by (S) into seconds and stores the conversion result into the area starting from the device specified by (D). The starting point (0 second) is January 1, 2000, 00:00:00.



For example, when November 27, 2016, 20:51:48 is specified, the data is converted as follows.



Operation error

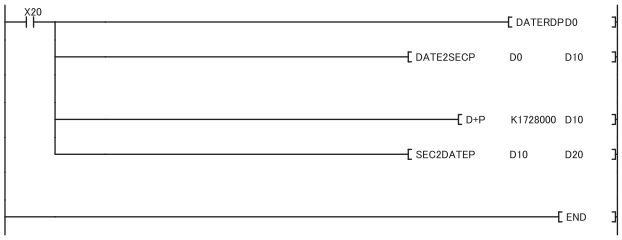
• In any of the following cases, an operation error occurs, the error flag (SM0) turns on, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4100	The date and time data in the device specified by (S) is out of range. A nonexistent date and time is set for (S). (Example: February 30, 2016)	—	—	—	—	○	—
4101	The range of the devices specified by (S) or (D) has exceeded the range of the corresponding devices.	—	—	—	—	○	—

Program example

- The following program adds 20 days (1728000 seconds) to the date and time data read from the clock element, and stores the results into the area starting from D20 when X20 is turned ON.

[Ladder Mode]

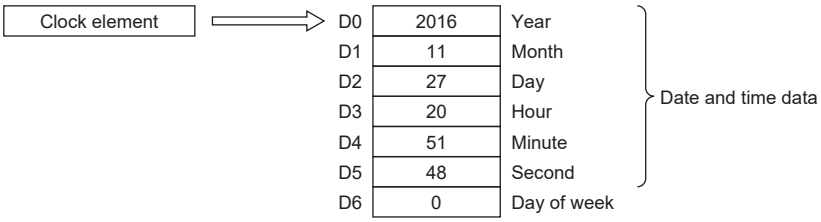


[List Mode]

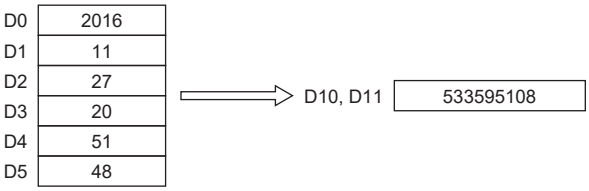
Step	Instruction	Device	
0	LD	X20	
1	DATERDP	D0	
4	DATE2SECP	D0	D10
8	D + P	K1728000	D10
13	SEC2DATEP	D10	D20
17	END		

[Operation]

- Reading date and time data triggered by the DATERDP instruction

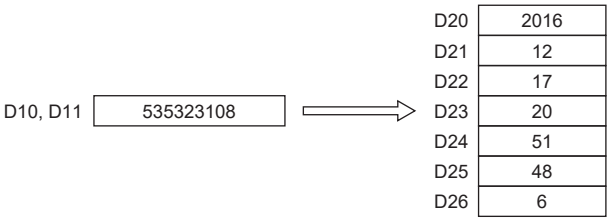


- Converting the data into seconds triggered by the DATE2SECP instruction



- Adding 20 days (1728000 seconds) to the converted second data

- Converting the second data into date and time data triggered by the SEC2DATEP instruction



Date and time data conversion (from second to year/month/day/time/minute/second/day of the week)

SEC2DATE(P)

Basic | High performance | Process | Redundant | Universal (Ver.) | LCPU

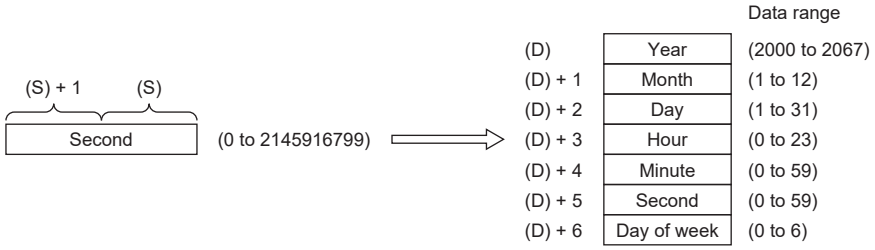
• QnUDVCPU, QnUDPVCPU: the serial number (first five digits) is "19042" or later.

(S): Head number of the devices where the second data before conversion is stored (BIN 32 bits)
(D): Head number of the devices where the date and time data after conversion will be stored (Device name)

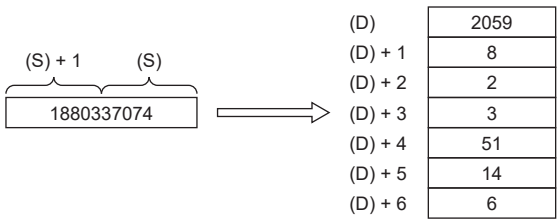
Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(S)	—	○		—	○			○	—
(D)	—	○		—	—			—	—

Processing details

• Converts the second data stored in the area starting from the device specified by (S) into date and time, and stores the conversion result into the area starting from the device specified by (D). The starting point (0 second) is January 1, 2000, 00:00:00.



For example, when 1880337074 seconds is specified, the data is converted as follows.



Operation error

• In any of the following cases, an operation error occurs, the error flag (SM0) turns on, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4100	The second data in the device specified by (S) is out of range.	—	—	—	—	○	—
4101	The range of the devices specified by (S) or (D) has exceeded the range of the corresponding devices.	—	—	—	—	○	—

Program example

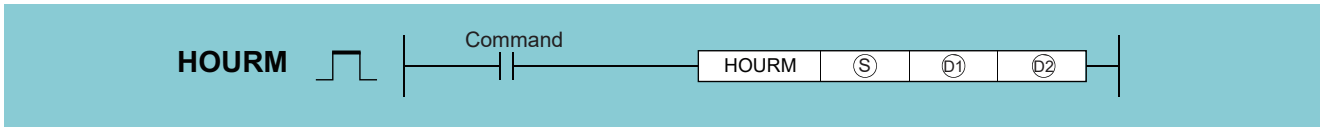
- For the program examples of the SEC2DATE(P) instruction, refer to the program example of Page 697 DATE2SEC(P).

Hour meter

HOURM

Basic | High performance | Process | Redundant | Universal | LCPU

• QnUDVCPU, QnUDPVCPU, LCPU: The serial number (first five digits) is "16112" or later.



(S): Start number of the device where the data of the time (to be set in increments of hour) during which (D2) is turned ON (BIN 16 bits)
(D1): Start number of the device where the data of the current value in increments of hour is stored (BIN 16 bits)
(D2): Start bit device number to which an alarm is to be output (Bit)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(S)	—	○	○	—	○	○	○	○	—
(D1)	—	○	○	—	—	—	—	—	—
(D2)	○	—	○	—	—	—	—	—	—

Processing details

• (D2) turns ON when the cumulative ON time of command input exceeds the time (in increments of hour) specified by (S). The current value is stored in increments of hour in the device specified by (D1) and the current value less than a hour is stored in increments of second in the device designated by (D1)+1.

• Measurement is continued even after the alarm output (D2) turns ON.
• Measurement is stopped when the current value (D1) per hour reaches the maximum value (32767) and the current value (D1)+1 per second reaches the maximum value (3599). To continue measurement, clear the current value in the devices specified by (D1) to (D1)+1.

• The change of clock data does not affect the operation of the HOURM instruction.

Point

The following programs cannot be used by the HOURM instruction.
• Interrupt programs
• Fixed scan execution type programs

Operation error

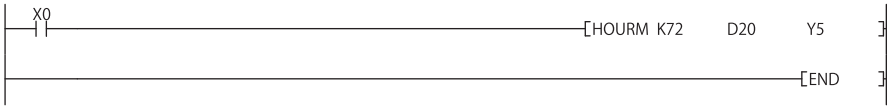
• In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4101	The data specified by (S) is smaller than 0. The device specified by (D1) exceeds the range of the number of device points.	—	—	—	—	○	○

Program example

• The sample program turns on Y5 when the cumulative ON time of X0 exceeds 72 hours.

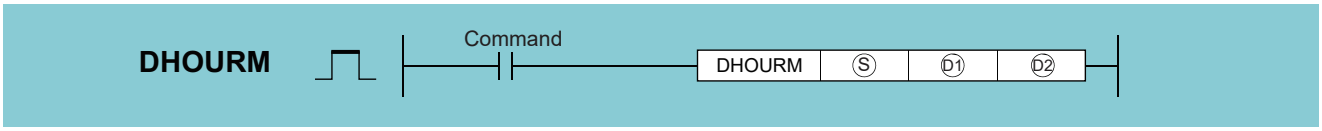
[Ladder Mode]



DHOURM

Basic | High performance | Process | Redundant | Universal Ver. | LCPU Ver.

• QnUDVCPU, QnUDPVCPU, LCPU: The serial number (first five digits) is "16112" or later.



(S): Start number of the device where the data of the time (to be set in increments of hour) during which (D2) is turned ON (BIN 32 bits)
(D1): Start number of the device where the data of the current value in increments of hour is stored (BIN 32 bits)
(D2): Start bit device number to which an alarm is to be output (Bit)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(S)	—	○	○	—	○	○	○	○	—
(D1)	—	○	○	—	—	—	—	—	—
(D2)	○	—	○	—	—	—	—	—	—

Processing details

• (D2) turns ON when the cumulative ON time of command input exceeds the time (in increments of hour) specified by (S). The current value is stored in increments of hour in the device specified by (D1)+1, (D1) and the current value less than a hour is stored in increments of second in the device designated by (D1)+2.

• Measurement is continued even after the alarm output (D2) turns ON.
• Measurement is stopped when the current values (D1)+1 and (D1) per hour reach the maximum value (2147483647) and the current value (D1)+2 per second reaches the maximum value (3599). To continue measurement, clear the current value in the devices specified by (D1) to (D1)+2.

• The change of clock data does not affect the operation of the DHOURM instruction.

Point

The following programs cannot be used by the DHOURM instruction.
• Interrupt programs
• Fixed scan execution type programs

Operation error

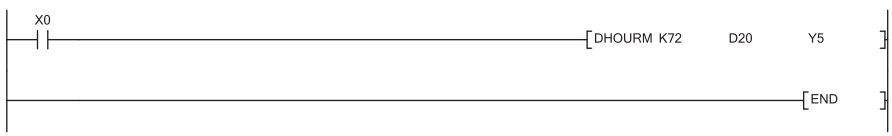
• In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4101	The data specified by (S) is smaller than 0. The device specified by (D1) exceeds the range of the number of device points.	—	—	—	—	○	○

Program example

• The sample program turns on Y5 when the cumulative ON time of X0 exceeds 72 hours.

[Ladder Mode]

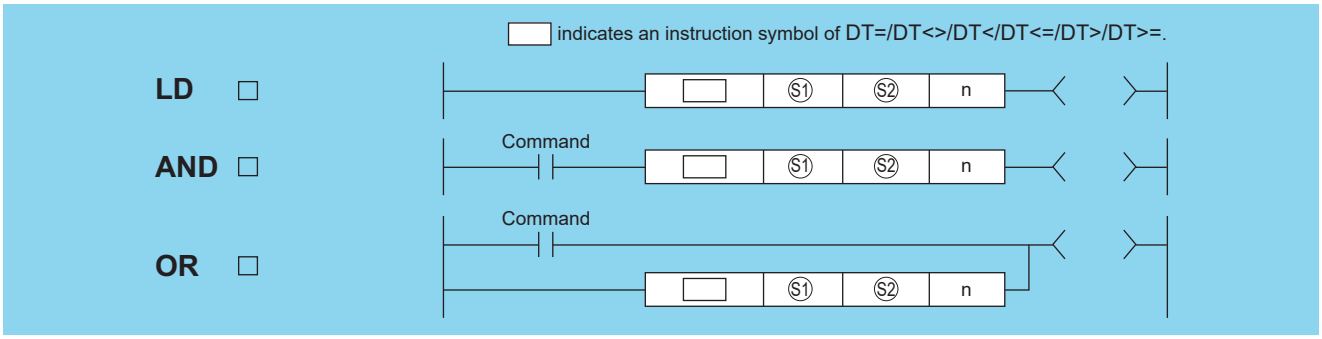


Date comparison

LDDT□, ANDDT□, ORDT□

Basic | High performance | Process | Redundant | Ver. Universal | LCPU

- QnU(D)(H)CPU, QnUDE(H)CPU: the serial number (first five digits) is "10102" or later
- Q00UJCPU, Q00UCPU, Q01UCPU, QnUDVCPU, QnUDPVCPU: Supported



(S1): Head number of the devices where the data to be compared are stored (BIN 16 bits)
(S2): Head number of the devices where the data to be compared are stored (BIN 16 bits)
n: Value of the data to be compared or the number of the stored data to be compared (BIN 16 bits)

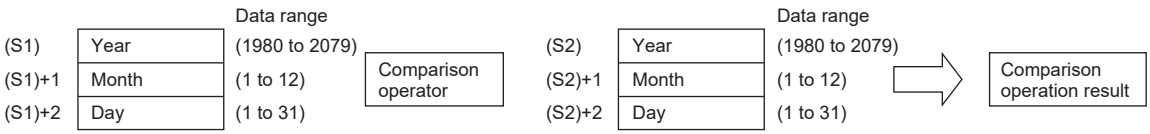
Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(S1)	—	○		—				—	—
(S2)	—	○		—				—	—
n	—	○		○				○	—

Processing details

- This instruction compares the date data specified by (S1) with those specified by (S2), or the date data specified by (S1) with the current date data. Setting n can determine the data to be compared.

• Comparison of given date data

This instruction treats the date data specified by (S1) and (S2) as a normally open contact, and then compares the data in accordance with the value of n.



• Comparison of current date data

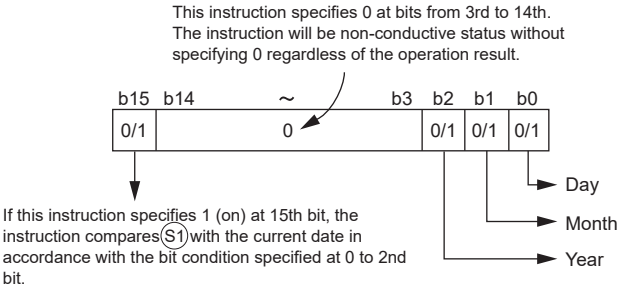
This instruction treats the date data specified by (S1) and the current date data as a normally open contact, and then compares the data in accordance with the value of n.
Time data specified by (S2) is treated as dummy data, and is ignored.

Point

When either (S1) or (S2) corresponds to any of the following in comparing given or current date data with given date data, the operation error (error code: 4101) or a malfunction may occurs.

- The range of the devices to be used for the index modification is specified over the range of the device specified by (S1) or (S2).
- File registers are specified by (S1) or (S2) without a register set.

- This instruction sets BIN values for each item.
- This instruction sets the year of four digits selected from 1980 to 2079 with the BIN value specified by (S1) or (S2).
- This instruction sets the month selected from 1 to 12 (January to December) with the BIN value specified by (S1)+1 or (S2)+1.
- This instruction sets the day selected from 1 to 31 (1st to 31st) for with the BIN value specified by (S1)+2 or (S2)+2.
- This instruction specifies the following values at n so that the data to be compared can be specified. The bit configuration specified at n is as follows.



Date data to be compared (from 0 to 2nd bit)

- 0: Does not compare specified date data (year/month/day).
- 1: Compares specified date data (year/month/day).

Operation data to be compared (15th bit)

- 0: Compares the date data specified by (S1) with the date data specified by (S2).
- 1: Compares the date data specified by (S1) with the current date data. Ignores the date data specified by (S2).

The following table shows processing details of bits to be compared.

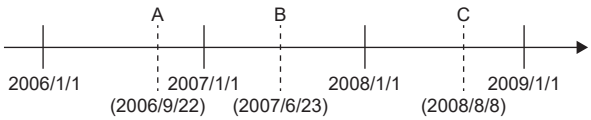
n value for comparison of specified date data with given date data	n value for comparison of specified date data with current date data	Date to be compared	Processing details
0001H	8001H	Day	Comparison of days ((S1)+2)
0002H	8002H	Month	Comparison of months ((S1)+1)
0003H	8003H	Month, day	Comparison of months ((S1)+1) and days ((S1)+2)
0004H	8004H	Year	Comparison of years ((S1))
0005H	8005H	Year, day	Comparison of years ((S1)) and days ((S1)+2)
0006H	8006H	Year, month	Comparison of years ((S1)) and months ((S1)+1)
0007H	8007H	Year, month, day	Comparison of years ((S1)), months ((S1)+1), and days ((S1)+2)
Other than 0001H to 0007H, 8001H to 8007H		No objects	No comparison of years ((S1)), months ((S1)+1), and days ((S1)+2) (Non-conductive)

- If the data stored in the devices to be compared are not recognized as date data, SM709 will be turned on after the instruction execution and no-conductive status will be made. Even if they are not recognized as date data but the range of the devices is within the setting range, SM709 will not be turned on. Moreover, if the range of devices specified by (S1) to (S1)+2 or (S2) to (S2)+2 exceeds the range of specified devices, SM709 will be turned on after the instruction execution and no-conductive status will be made. Once SM709 is turned on, on-status will be retained till when the CPU modules are reset or powered off. Therefore, turn off SM709 if necessary.

• The following table shows the comparison operation results for each instruction.

Instruction symbol	Condition	Comparison operation result	Instruction symbol	Condition	Comparison operation result
DT=	(S1)=(S2)	Conductive status	DT=	(S1)≠(S2)	No-conductive status
DT<>	(S1)≠(S2)		DT<>	(S1)=(S2)	
DT>	(S1)>(S2)		DT>	(S1)≤(S2)	
DT<=	(S1)≤(S2)		DT<=	(S1)>(S2)	
DT<	(S1)<(S2)		DT<	(S1)≥(S2)	
DT>=	(S1)≥(S2)		DT>=	(S1)<(S2)	

• The following figure shows the comparison example of dates.



The following table shows the conductive states resulting from performing the comparison operation of the dates A, B, and C shown above.
Even if the objects to be compared are under the same condition, the comparison operation results vary depending on the objects selected.

Comparison objects	Comparison condition (○: Conductive, ×: No-conductive)		
	A<B	B<C	A<C
Day	○	×	×
Month	×	○	×
Month, day	×	○	×
Year	○	○	○
Year, day	○	○	○
Year, month	○	○	○
Year, month, day	○	○	○
No objects	×	×	×

• Even if the dates to be compared do not exist practically, this instruction executes the comparison operation for the objects with the settable dates in accordance with the following condition.

Date A: 2006/02/30 (This date is settable, though it does not exist.)
Date B: 2007/03/29
Date C: 2008/02/31 (This date is settable, though it does not exist.)

Comparison objects	Comparison condition (○: Conductive, ×: No-conductive)		
	A<B	B<C	A<C
Day	×	×	○
Month	×	×	×
Month, day	○	×	○
Year	○	○	○
Year, day	○	○	○
Year, month	○	○	○
Year, month, day	○	○	○
No objects	×	×	×

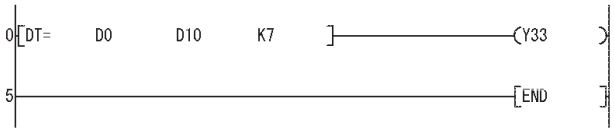
Operation error

• There is no operation error in the LDDT□, ANDDT□, or ORDT□ instruction.

Program example

- The following program compares the data stored in D0 with the data (year, month, and day) stored in D10, and makes Y33 be conductive status when the data stored in D0 meet the data stored in D10.

[Ladder Mode]

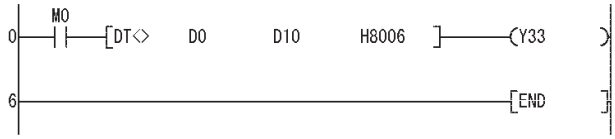


[List Mode]

Step	Instruction	Device		
0	LDDT=	D0	D10	K7
4	OUT	Y33		
5	END			

- The following program compares the data stored in D0 with the current date data (year and month), and makes Y33 be conductive status when the data stored in D0 do not meet the current date data, when M0 is turned on.

[Ladder Mode]



[List Mode]

Step	Instruction	Device		
0	LD	M0		
1	ANDDT<>	D0	D10	H8006
5	OUT	Y33		
6	END			

- The following program compares the data stored in D0 with the data (year and day) stored in D10, and makes Y33 be conductive status when the data value stored in D10 is smaller than the data value stored in D0, when M0 is turned on.

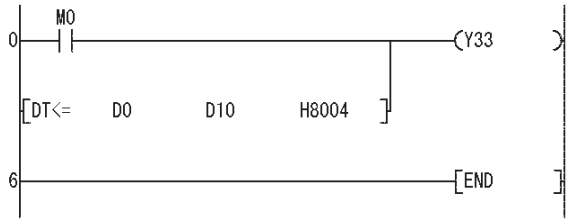
[Ladder Mode]

[List Mode]

Step	Instruction	Device		
0	LD	M0		
1	ANDDT>	D0	D10	K5
5	OUT	Y33		
6	END			

- The following program compares the data stored in D0 with the current date data (year), and makes Y33 be conductive status when the value of the current date data is the data value stored in D0 or larger.

[Ladder Mode]



[List Mode]

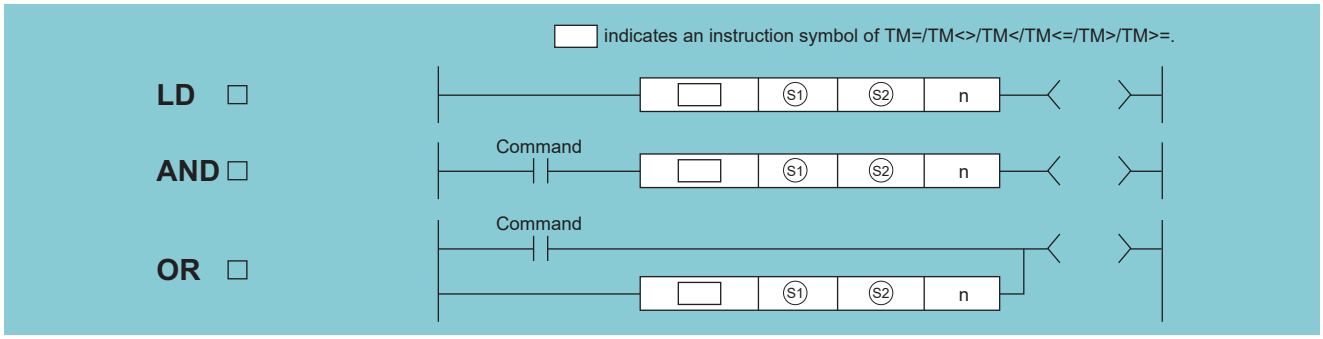
Step	Instruction	Device		
0	LD	M0		
1	ORDT<=	D0	D10	H8004
5	OUT	Y33		
6	END			

Time comparison

LDTM□, ANDTM□, ORTM□

Basic | High performance | Process | Redundant | Ver. Universal | LCPU

- QnU(D)(H)CPU, QnUDE(H)CPU: the serial number (first five digits) is "10102" or later
- Q00UJCPU, Q00UCPU, Q01UCPU, QnUDVCPU, QnUDPVCPU: Supported



(S1): Head number of the devices where the data to be compared are stored (BIN 16 bits)
(S2): Head number of the devices where the data to be compared are stored (BIN 16 bits)
n: Value of the data to be compared or the number of the stored data to be compared (BIN 16 bits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(S1)	—	○		—				—	—
(S2)	—	○		—				—	—
n	—	○		○				○	—

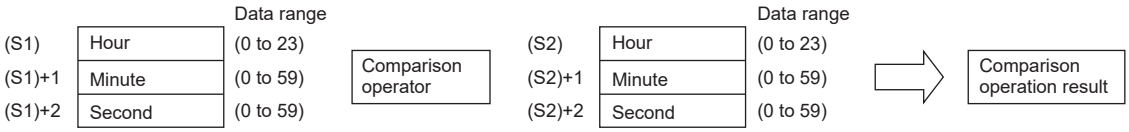
7

Processing details

- This instruction compares the clock data specified by (S1) with those specified by (S2), or the clock data specified by (S1) with the current time data. Setting n determines the data to be compared.

• Comparison of given clock data

This instruction treats the clock data specified by (S1) and the clock data specified by (S2) as a normally open contact, and compares the data in accordance with the value of n.



• Comparison of current time data

This instruction treats the clock data specified by (S1) and the current time data as a normally open contact, and compares the data in accordance with the value of n.

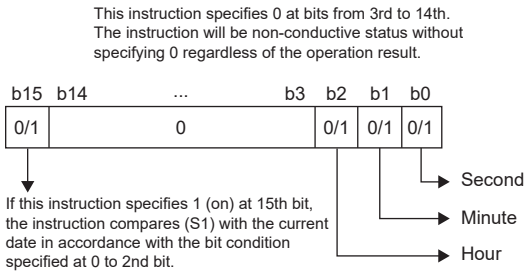
This instruction treats the clock data specified by (S2) as dummy data and ignores the data.

Point

When either (S1) or (S2) corresponds to any of the following conditions in comparing given or current time data with specified clock data, the operation error (error code: 4101) or a malfunction may occurs.

- The range of the devices to be used for the index modification is specified over the range of the device specified by (S1) or (S2).
- File registers are specified by (S1) or (S2) without a register set.

- This instruction sets BIN values for each item.
- This instructions sets the time selected from 0 to 23 (midnight to 23 o'clock) with the BIN value specified by (S1) or (S2). (Uses the 24-hour clock.)
- This instructions sets the minute selected from 0 to 59 (0 to 59 minutes) with BIN value specified by (S1)+1 or (S2)+1.
- This instructions sets the second selected from 0 to 59 (0 to 59 seconds) with BIN value specified by (S1)+2 or (S2)+2.
- This instruction specifies the following values at n so that the data to be compared can be specified. The bit configuration specified at n is as follows.



Clock data to be compared (from 0 to 2nd bit)

- 0: Does not compare specified clock data (hour/minute/second).
- 1: Compares specified clock data (hour/minute/second).

Operation data to be compared (15th bit)

- 0: Compares the clock data specified by (S1) with the clock data specified by (S2).
- 1: Compares the clock data specified by (S1) with the current time data. Ignores the clock data specified by (S2).

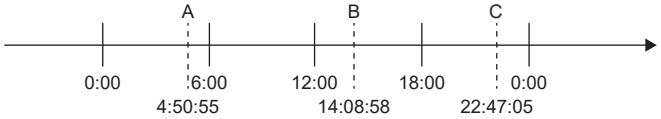
The following table shows processing details of bits to be compared.

n value for comparison of specified clock data with given clock data	n value for comparison of specified clock data with current time data	Time to be compared	Processing details
0001H	8001H	Second	Comparison of seconds ((S1)+2)
0002H	8002H	Minute	Comparison of minutes ((S1)+1)
0003H	8003H	Minute, second	Comparison of minutes ((S1)+1) and seconds ((S1)+2)
0004H	8004H	Hour	Comparison of hours ((S1))
0005H	8005H	Hour, second	Comparison of hours ((S1)) and seconds ((S1)+2)
0006H	8006H	Hour, minute	Comparison of hours ((S1)) and minutes ((S1)+1)
0007H	8007H	Hour, minute, second	Comparison of hours ((S1)), minutes ((S1)+1), and seconds ((S1)+2)
Other than 0001H to 0007H, 8001H to 8007H		None	No comparison of hours ((S1)), minutes ((S1)+1), and seconds ((S1)+2) (Non-conductive)

- If the data stored in the devices to be compared are not recognized as date data, SM709 will be turned on after the instruction execution and no-conductive status will be made. Once SM709 is turned on, on-status will be retained till when the CPU modules are reset or powered off. Therefore, turn off SM709 if necessary. Moreover, if the range of devices specified by (S1) to (S1)+2 or (S2) to (S2)+2 exceeds the range of specified devices, SM709 will be turned on and no-conductive status will be made.
- The following table shows the comparison operation results for each instruction.

Instruction symbol	Condition	Comparison operation result	Instruction symbol	Condition	Comparison operation result
TM=	(S1)=(S2)	Conductive status	TM=	(S1)≠(S2)	No-conductive status
TM<>	(S1)≠(S2)		TM<>	(S1)=(S2)	
TM>	(S1)>(S2)		TM>	(S1)≤(S2)	
TM<=	(S1)≤(S2)		TM<=	(S1)>(S2)	
TM<	(S1)<(S2)		TM<	(S1)≥(S2)	
TM>=	(S1)≥(S2)		TM>=	(S1)<(S2)	

- The following figure shows the comparison example of time.



The following table shows the conductive states resulting from performing the comparison operation of the time A, B, and C shown above.
Even if the objects to be compared are under the same condition, the comparison operation results vary depending on the objects selected.

Comparison objects	Comparison condition (○: Conductive, ×: No-conductive)		
	A<B	B<C	A<C
Second	○	×	×
Minute	×	○	×
Minute, second	×	○	×
Hour	○	○	○
Hour, second	○	○	○
Hour, minute	○	○	○
Hour, minute, second	○	○	○
None	×	×	×

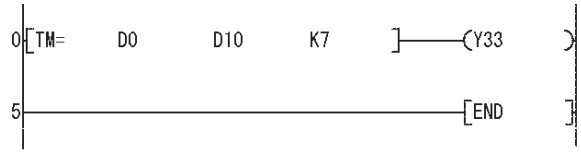
Operation error

- There is no operation error in the LDTM☐ instruction, ANDTM☐ instruction, and ORTM☐ instruction.

Program example

- The following program compares the data stored in D0 with the data (hour, minute, and second) stored in D10, and makes Y33 be conductive status when the data stored in D0 meet the data stored in D10.

[Ladder Mode]

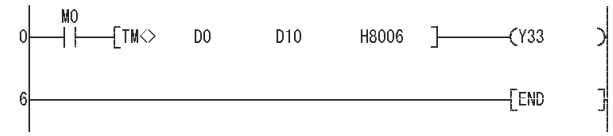


[List Mode]

Step	Instruction	Device		
0	LDTM=	D0	D10	K7
4	OUT	Y33		
5	END			

- The following program compares the data stored in D0 with the current time data (hour and minute), and makes Y33 be conductive status when the data stored in D0 do not meet the current date data, when M0 is turned on.

[Ladder Mode]



[List Mode]

Step	Instruction	Device		
0	LD	M0		
1	ANDTM<>	D0	D10	H8006
5	OUT	Y33		
6	END			

- The following program compares the data stored in D0 with the data (hour and second) stored in D10, and makes Y33 be conductive status when the data value stored in D10 is smaller than the data value stored in D0, when M0 is turned on.

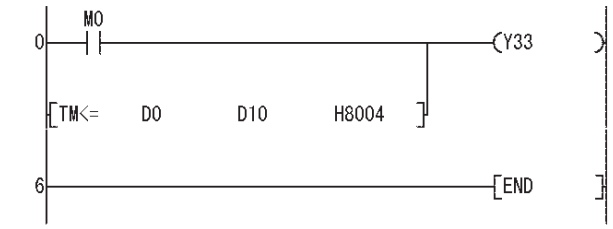
[Ladder Mode]

[List Mode]

Step	Instruction	Device		
0	LD	M0		
1	ANDTM>	D0	D10	K5
5	OUT	Y33		
6	END			

- The following program compares the data stored in D0 with the current time data (hour), and makes Y33 be conductive status when the value of the current time data is the data value stored in D0 or larger.

[Ladder Mode]



[List Mode]

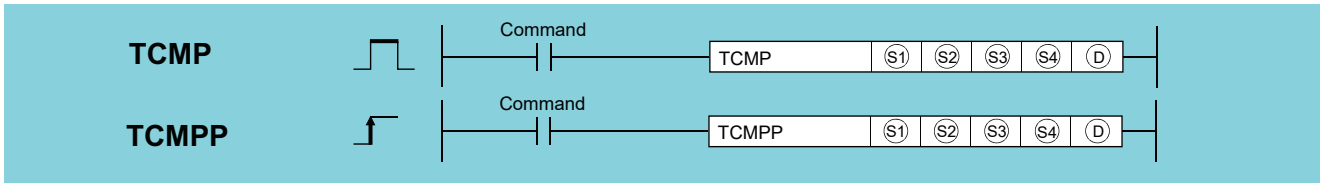
Step	Instruction	Device		
0	LD	M0		
1	ORTM<=	D0	D10	H8004
5	OUT	Y33		
6	END			

Clock data comparison

TCMP(P)

Basic | High performance | Process | Redundant | Universal (Ver.) | LCPU (Ver.)

• QnUDVCPU, QnUDPVCPU, LCPU: The serial number (first five digits) is "16112" or later.

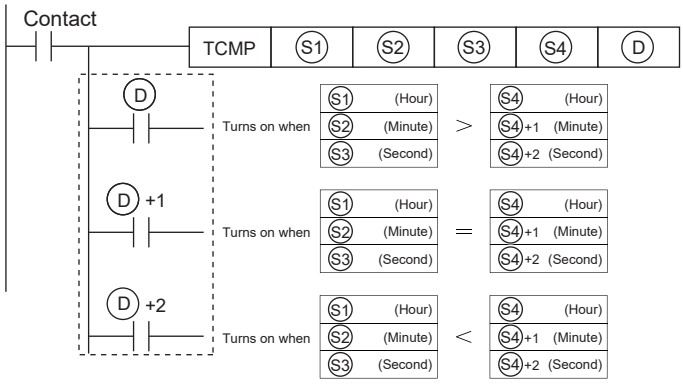


(S1): Start number of the devices where the comparison target time data "hour" is stored (0 to 23) (BIN 16 bits)
(S2): Start number of the devices where the comparison target time data "minute" is stored (0 to 59) (BIN 16 bits)
(S3): Start number of the devices where the comparison target time data "second" is stored (0 to 59) (BIN 16 bits)
(S4): Start number of the device where clock data (hour, minute, second) is stored (3 points occupied) (device name)
(D): Start bit device number to which the comparison result is output (3 points occupied) (Bit)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(S1)	—	○	○	—	○	○	—	○	—
(S2)	—	○	○	—	○	○	—	○	—
(S3)	—	○	○	—	○	○	—	○	—
(S4)	—	○	○	—	—	—	—	—	—
(D)	○	—	○	—	—	—	—	—	—

Processing details

• The comparison target time data (hour, minute, second) "(S1), (S2), (S3)" is compared with the time data [(S4), (S4)+1, (S4)+2], and one of (D), (D)+1, and (D)+2 turns on depending on the result (small, match, large).



Operation error

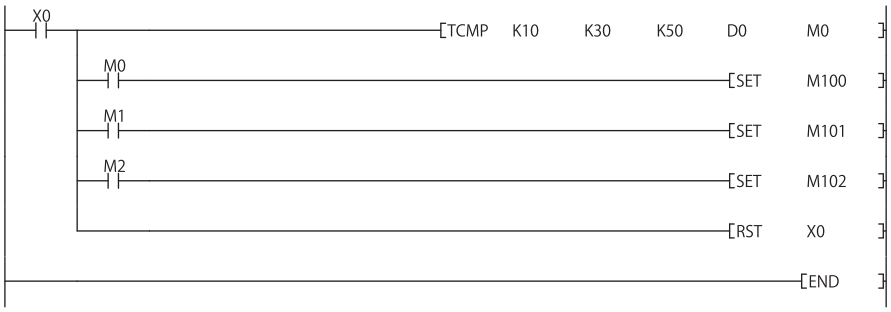
• In any of the following cases, an operation error occurs, the error flag (SM0) turns on, and an error code is stored in SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4101	The value in the device specified by (S1) exceeds the setting range (0 to 23). The value in the device specified by (S2) exceeds the setting range (0 to 59). The value in the device specified by (S3) exceeds the setting range (0 to 59).	—	—	—	—	○	○

Program example

• Program that compares specified time data with D0 (hour), D1 (minute), and D2 (second)

[Ladder Mode]



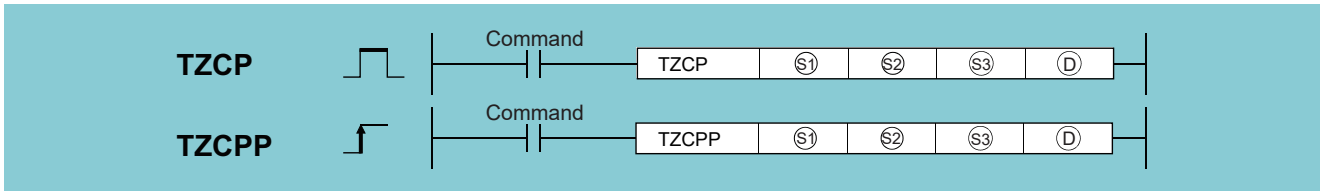
Compare 10:30:50 with D0 (hour), D1 (minute), and D2 (second).
M0 turns on when 10:30:50 > D0, D1, and D2.
M1 turns on when 10:30:50 = D0, D1, and D2.
M2 turns on when 10:30:50 < D0, D1, and D2.

Clock data band comparison

TZCP(P)

Basic | High performance | Process | Redundant | Universal Ver. | LCPU Ver.

• QnUDVCPU, QnUDPVCPU, LCPU: The serial number (first five digits) is "16112" or later.



(S1): Start number of the devices where the lower limit values (hour, minute, second) of the comparison target time data are stored (three points occupied) (device name)

(S2): Start number of the devices where the upper limit values (hour, minute, second) of the comparison target time data are stored (three points occupied) (device name)

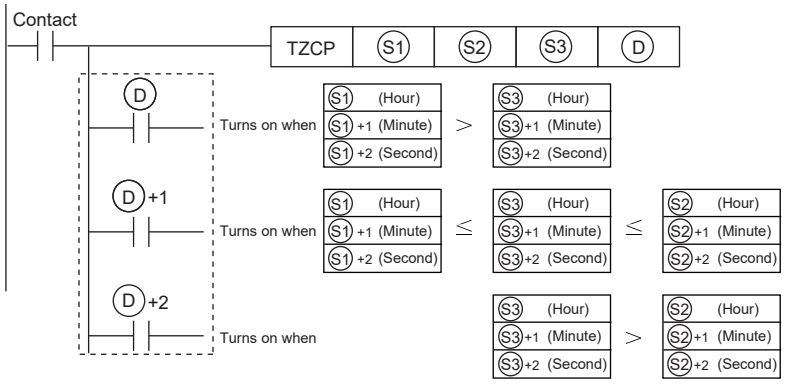
(S3): Start number of the device where the time data to be compared is stored (3 points occupied) (device name)

(D): Start bit device number to which the comparison result is output (3 points occupied) (Bit)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant	Others
	Bit	Word		Bit	Word				
(S1)	—	○	○	—	—	—	—	—	—
(S2)	—	○	○	—	—	—	—	—	—
(S3)	—	○	○	—	—	—	—	—	—
(D)	○	—	○	—	—	—	—	—	—

Processing details

• The comparison target time of two upper and lower points (hour, minute, second) is compared with the three points of time data "hour, minute, second" starting from (S3), and depending on the result (small, within the band, large), (D), (D)+1, or (D)+2 turns ON.



Operation error

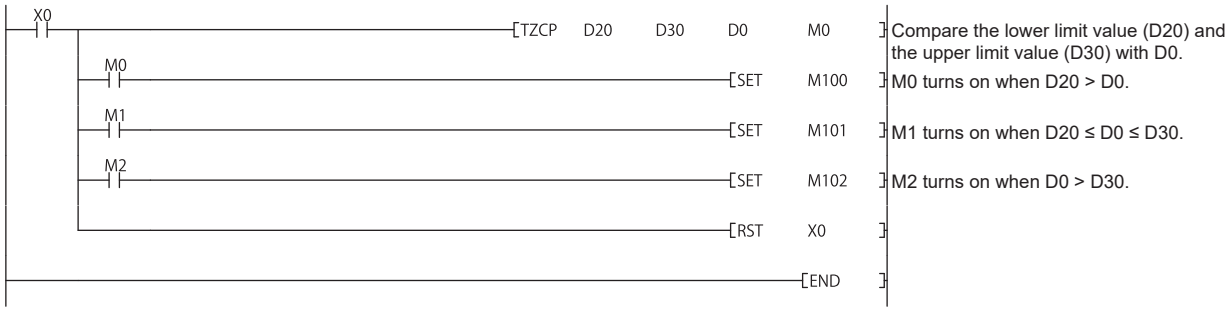
• In any of the following cases, an operation error occurs, the error flag (SM0) turns on, and an error code is stored in SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4101	The value in the device specified by (S1)+0, (S2)+0, (S3)+0 exceeds the setting range (0 to 23). The value in the device specified by (S1)+1, (S2)+1, (S3)+1 exceeds the setting range (0 to 59). The value in the device specified by (S1)+2, (S2)+2, (S3)+2 exceeds the setting range (0 to 59).	—	—	—	—	○	○

Program example

• The sample program compares any specified clock data with D0 (hour), D1 (minute), and D2 (second).

[Ladder Mode]



7.16 Expansion Clock Instructions

Reading expansion clock data

S(P).DATERD

Basic (×) | High performance (Ver.) | Process (Ver.) | Redundant (Ver.) | Universal | LCPU

- High Performance model QCPU: The serial number (first five digits) is "07032" or later.
- Process CPU: The serial number (first five digits) is "07032" or later.
- Redundant CPU: The serial number (first five digits) is "07032" or later.

(D): Head number of the devices where the read clock data will be stored (BIN 16 bits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant	Others
	Bit	Word		Bit	Word				
(D)	—	○		—					

Processing details

- Reads "year, month, day, hour, minute, second, day of the week, and millisecond" from the clock element of the CPU module, and stores it as BIN value into the device specified by (D) or later device.

Clock element	(D)	Year	(1980 to 2079)
	(D)+1	Month	(1 to 12)
	(D)+2	Day	(1 to 31)
	(D)+3	Hour (24-hour clock)	(0 to 23)
	(D)+4	Minute	(0 to 59)
	(D)+5	Second	(0 to 59)
	(D)+6	Day of week	(0 to 6)
	(D)+7	Millisecond	(0 to 999)

- The "year" at (D) is stored as 4-digit year indication.
- The "day of week" at (D)+6 is stored as 0 to 6 to represent the days Sunday to Saturday.

Day of week	Sunday	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday
Stored data	0	1	2	3	4	5	6

- Compensation is made automatically for leap years.

Operation error

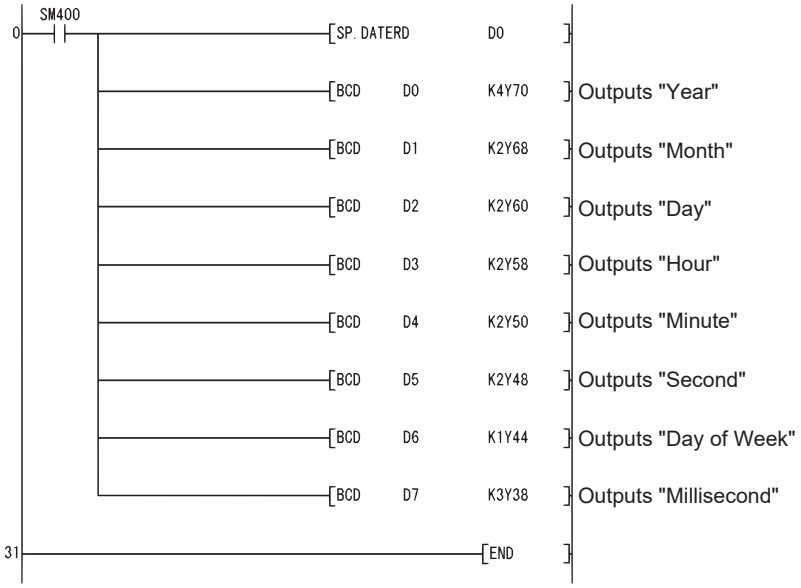
- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4101	The range of the device specified by (D) exceeds the range of the corresponding device.	—	—	—	—	○	○

Program example

- The following program outputs the following clock data as BCD values:
  - Year: Y70 to Y7F
  - Month: Y68 to Y6F
  - Day: Y60 to Y67
  - Hour: Y58 to Y5F
  - Minute: Y50 to Y57
  - Second: Y48 to Y4F
  - Week: Y44 to Y47
  - Millisecond: Y38 to Y43

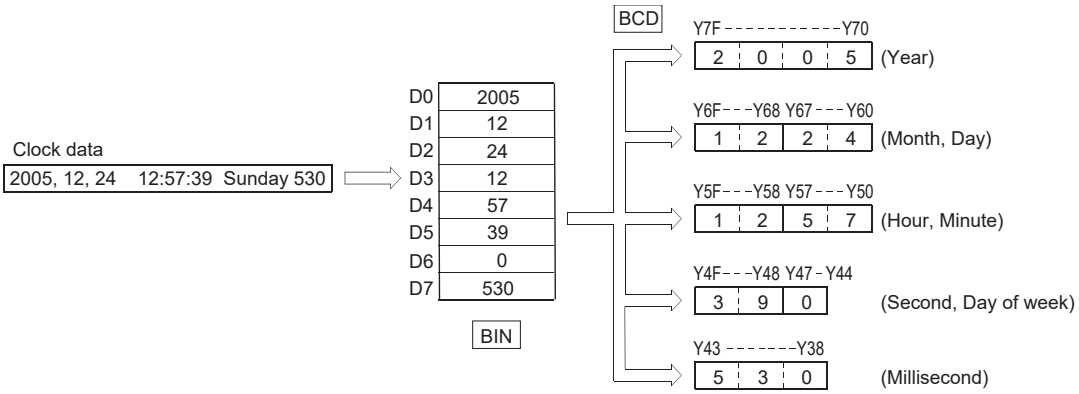
[Ladder Mode]



[List Mode]

Step	Instruction	Device	
0	LD	SM400	
1	SP. DATERD		D0
7	BCD	D0	K4Y70
10	BCD	D1	K2Y68
13	BCD	D2	K2Y60
16	BCD	D3	K2Y58
19	BCD	D4	K2Y50
22	BCD	D5	K2Y48
25	BCD	D6	K1Y44
28	BCD	D7	K3Y38
31	END		

[Operation]



Precautions

- This instruction reads clock data and stores those to a specified device even if a wrong clock data is set to the CPU module (example: Feb. 30th). When setting clock data with the DATEWR instruction or GX Developer, make sure to set a correct data.
- Time error of reading a clock data of millisecond is a maximum of 2ms. (Difference between the data memorized by clock element inside of the CPU module and the data read by this function.)
- Specifying digit for the bit device can be used only when the following conditions are met.
  - Digit specification: K4
  - Head of device: multiple of 16

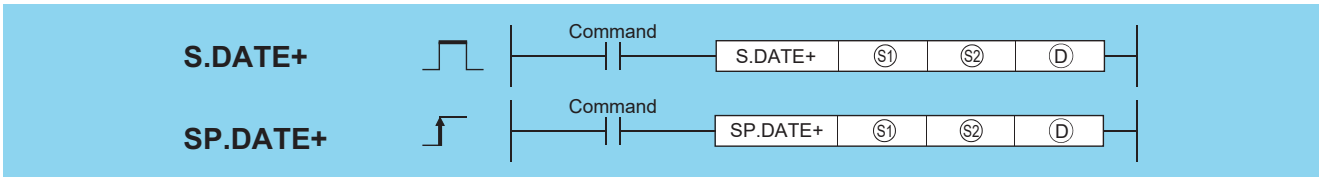
When the above conditions are not met, INSTRCT CODE ERR. (error code: 4004) will occur.

Expansion clock data addition operation

S(P).DATE+

Basic | Ver. High performance | Ver. Process | Ver. Redundant | Universal | LCPU

- High Performance model QCPU: The serial number (first five digits) is "07032" or later.
- Process CPU: The serial number (first five digits) is "07032" or later.
- Redundant CPU: The serial number (first five digits) is "07032" or later.



(S1): Head number of the devices where the clock data to be adjusted by addition is stored (BIN 16 bits)
(S2): Head number of the devices where the time data to be added for adjustment is stored (BIN 16 bits)
(D): Head number of the devices where the result of addition of clock (time) data will be stored (BIN 16 bits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant	Others
	Bit	Word		Bit	Word				
(S1)	—	○		—					
(S2)	—	○		—					
(D)	—	○		—					

Processing details

- Adds the time data designated by (S2) to the clock data designated by (S1), and stores the result into the area starting from the device designated by (D).

		Setting data			Setting data			Setting data
(S1)	Hour	(0 to 23)		(S2)	Hour	(0 to 23)	(D)	Hour	(0 to 23)
(S1)+1	Minute	(0 to 59)		(S2)+1	Minute	(0 to 59)	(D)+1	Minute	(0 to 59)
(S1)+2	Second	(0 to 59)	+	(S2)+2	Second	(0 to 59)	(D)+2	Second	(0 to 59)
(S1)+3	–			(S2)+3	–		(D)+3	–	
(S1)+4	Millisecond	(0 to 999)		(S2)+4	Millisecond	(0 to 999)	(D)+4	Millisecond	(0 to 999)

For example, adding the time 7:48:10:500 to 6:32:40:875 would result in the following operation:

(S1)	Hour: 6		(S2)	Hour: 7		(D)	Hour: 14
(S1)+1	Minute: 32		(S2)+1	Minute: 48		(D)+1	Minute: 20
(S1)+2	Second: 40	+	(S2)+2	Second: 10	⇨	(D)+2	Second: 51
(S1)+3	–		(S2)+3	–		(D)+3	–
(S1)+4	Millisecond: 875		(S2)+4	Millisecond: 500		(D)+4	Millisecond: 375

- If the results of the addition of time exceed 24 hours, 24 hours will be subtracted from the sum to make the final operation result. For example, when the time 20:20:20:500 is added to 14:20:30:875, the result is not 34:40:51:375, but 10:40:51:375.

(S1)	Hour: 14		(S2)	Hour: 20		(D)	Hour: 10
(S1)+1	Minute: 20		(S2)+1	Minute: 20		(D)+1	Minute: 40
(S1)+2	Second: 30	+	(S2)+2	Second: 20	⇨	(D)+2	Second: 51
(S1)+3	–		(S2)+3	–		(D)+3	–
(S1)+4	Millisecond: 875		(S2)+4	Millisecond: 500		(D)+4	Millisecond: 375

Point

Devices, (S1)+3, (S2)+3, and (D)+3 are not used for operation.

A clock data read by the S(P).DATERD instruction can be directly added.

(D)	Hour
(D)+1	Minute
(D)+2	Second
(D)+3	Day of week
(D)+4	Millisecond

When the clock data is read by the S(P).DATERD instruction, day of week is inserted between "second" and "millisecond".
If the S(P).DATE+ instruction is used to read the clock data, the data can be directly used for addition since it does not perform the calculation for the day of a week.

Operation error

- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4100	The value set for (S1) and (S2) is not within the setting range. (☞ Processing details)	—	○	○	○	○	○
4101	The range of the device specified by (S1), (S2) or (D) exceeds the range of the corresponding device.	—	—	—	—	○	○

Precautions

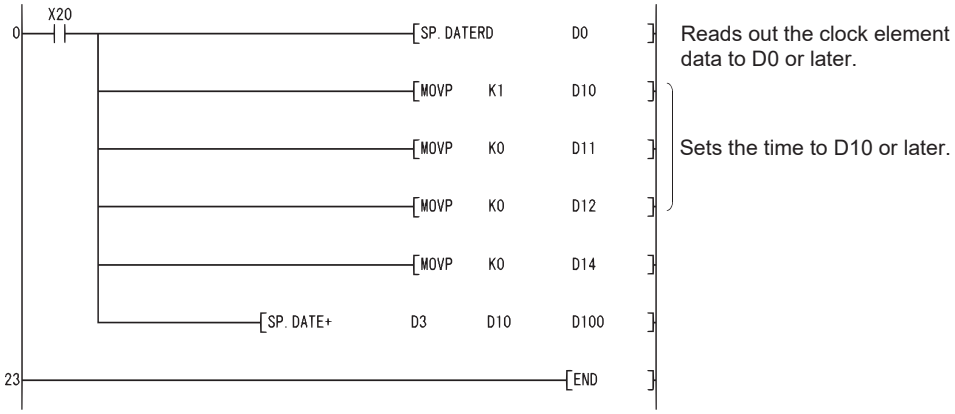
- Specifying digit for the bit device can be used only when the following conditions are met.
  - Digit specification: K4
  - Head of device: multiple of 16

When the above conditions are not met, INSTRCT CODE ERR. (error code: 4004) will occur.

Program example

- The following program adds 1 hour to the clock data read from the clock element, and stores the results into the area starting from D100 when X20 is turned ON.

[Ladder Mode]

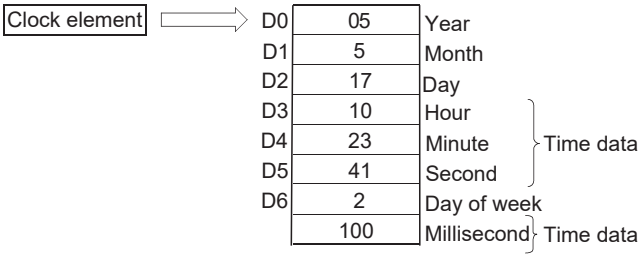


[List Mode]

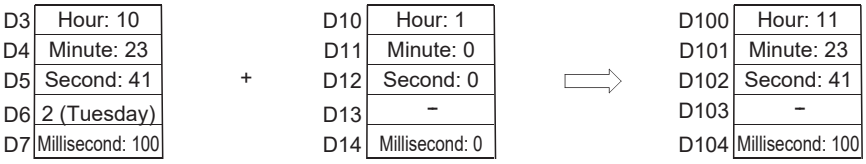
Step	Instruction	Device		
0	LD	X20		
1	SP. DATERD		D0	
7	MOVP	K1	D10	
9	MOVP	K0	D11	
11	MOVP	K0	D12	
13	MOVP	K0	D14	
15	SP. DATE+	D3	D10	D100
23	END			

[Operation]

- Time data read operation by the SP.DATERD instruction



- Addition by the SP.DATE+ instruction

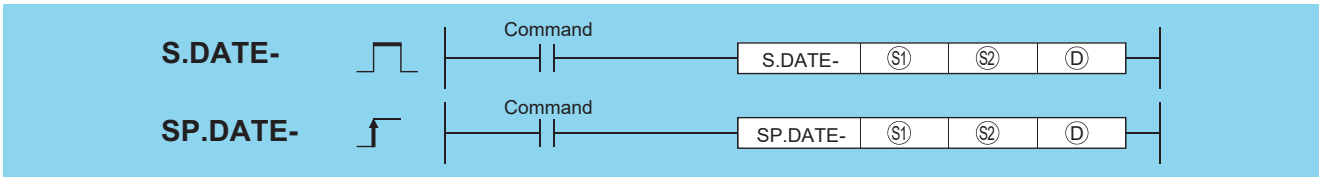


Expansion clock data subtraction operation

S(P).DATE-

Basic | High performance (Ver.) | Process (Ver.) | Redundant (Ver.) | Universal | LCPU

- High Performance model QCPU: The serial number (first five digits) is "07032" or later.
- Process CPU: The serial number (first five digits) is "07032" or later.
- Redundant CPU: The serial number (first five digits) is "07032" or later.



(S1): Head number of the devices where the clock time data to be adjusted by subtraction is stored (BIN 16 bits)
(S2): Head number of the devices where time data to be subtracted for adjustment is stored (BIN 16 bits)
(D): Head number of the devices where the result of subtraction of clock (time) data will be stored (BIN 16 bits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant	Others
	Bit	Word		Bit	Word				
(S1)	—	○		—					
(S2)	—	○		—					
(D)	—	○		—					

Processing details

- Subtracts the time data designated by (S2) from the clock data designated by (S1), and stores the result into the area starting from the device designated by (D).

		Setting data			Setting data			Setting data
(S1)	Hour	(0 to 23)		(S2)	Hour	(0 to 23)	(D)	Hour	(0 to 23)
(S1)+1	Minute	(0 to 59)		(S2)+1	Minute	(0 to 59)	(D)+1	Minute	(0 to 59)
(S1)+2	Second	(0 to 59)	–	(S2)+2	Second	(0 to 59)	(D)+2	Second	(0 to 59)
(S1)+3	–			(S2)+3	–		(D)+3	–	
(S1)+4	Millisecond	(0 to 999)		(S2)+4	Millisecond	(0 to 999)	(D)+4	Millisecond	(0 to 999)

For example, when the clock time 3:50:10:500 is subtracted from the clock time 10:40:20:875, the operation is performed as follows:

(S1)	Hour: 10		(S2)	Hour: 3		(D)	Hour: 6
(S1)+1	Minute: 40		(S2)+1	Minute: 50		(D)+1	Minute: 50
(S1)+2	Second: 20	–	(S2)+2	Second: 10	⇨	(D)+2	Second: 10
(S1)+3	–		(S2)+3	–		(D)+3	–
(S1)+4	Millisecond: 875		(S2)+4	Millisecond: 500		(D)+4	Millisecond: 375

- If the subtraction results in a negative number, 24 will be added to the result to make a final operation result. For example, when the clock time 10:42:12:500 is subtracted from 4:50:32:875, the result is not 6:8:20:375, but 18:8:20:375.

(S1)	Hour: 4		(S2)	Hour: 10		(D)	Hour: 18
(S1)+1	Minute: 50		(S2)+1	Minute: 42		(D)+1	Minute: 8
(S1)+2	Second: 32	–	(S2)+2	Second: 12	⇨	(D)+2	Second: 20
(S1)+3	–		(S2)+3	–		(D)+3	–
(S1)+4	Millisecond: 875		(S2)+4	Millisecond: 500		(D)+4	Millisecond: 375

Point

Devices, (S1)+3, (S2)+3, and (D)+3 are not used for operation.

A clock data read by S(P).DATERD instruction can be directly subtracted.

(D)	Hour
(D)+1	Minute
(D)+2	Second
(D)+3	Day of week
(D)+4	Millisecond

When the clock data is read by the S(P).DATERD instruction, day of week is inserted between "second" and "millisecond".
If the S(P).DATE- instruction is used to read the clock data, the data can be directly used for subtraction since it does not perform the calculation for the day of the week.

Operation error

- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4100	The value set for (S1) and (S2) is not within the setting range. (☞ Processing details)	—	○	○	○	○	○
4101	The range of the device specified by (S1), (S2) or (D) exceeds the range of the corresponding device.	—	—	—	—	○	○

Precautions

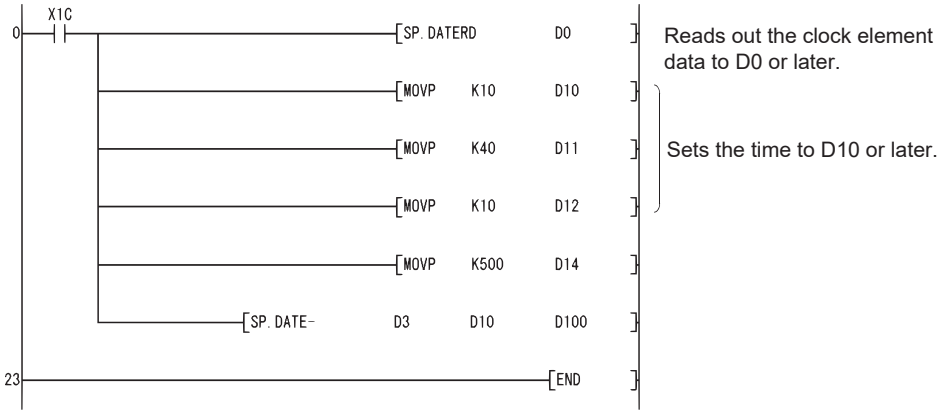
- Specifying digit for the bit device can be used only when the following conditions are met.
  - Digit specification: K4
  - Head of device: multiple of 16

When the above conditions are not met, INSTRCT CODE ERR. (error code: 4004) will occur.

Program example

- The following program subtracts the time data stored in the area starting from D10 from the clock data read from the clock element when X1C is turned ON, and stores the result into the area starting from D100.

[Ladder Mode]

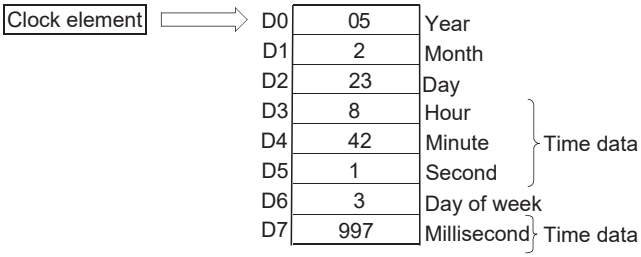


[List Mode]

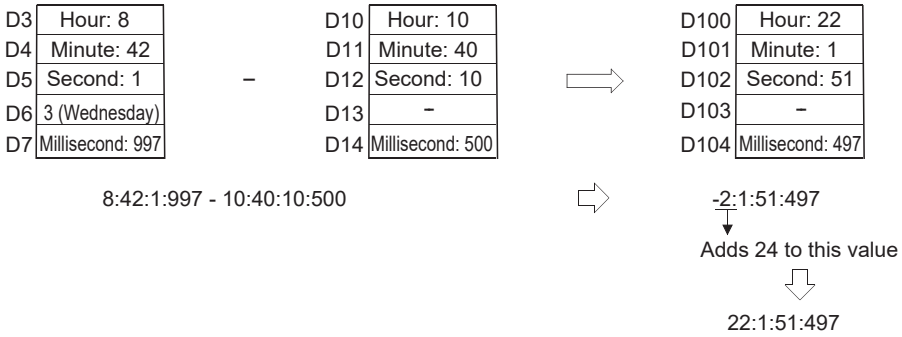
Step	Instruction	Device		
0	LD	X1C		
1	SP. DATERD		D0	
7	MOVP	K10	D10	
9	MOVP	K40	D11	
11	MOVP	K10	D12	
13	MOVP	K500	D14	
15	SP. DATE-	D3	D10	D100
23	END			

[Operation]

- Time data read operation by the SP.DATERD instruction



- Subtraction by the SP.DATE- instruction



8:42:1:997 - 10:40:10:500

-2:1:51:497

Adds 24 to this value

22:1:51:497

7.17 Program Control Instructions

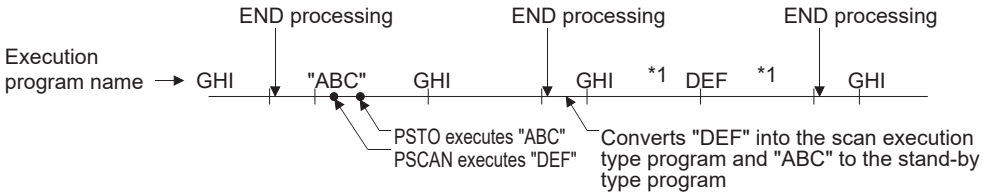
Processing when the execution type is converted with the program control instruction is as follows.

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

Point

Once the fixed scan execution type program is changed to another execution type, it cannot be returned to the fixed scan execution type.

As program execution type conversions by PSCAN and PSTOP instructions occur at the END processing, such conversions are impossible during program execution. When different execution types have been set for the same program in the same scan, the execution type will be that specified by the execution switching command that was executed last.



*1 The order of "GHI" and "DEF" program execution is determined by the program settings parameters.

Switching from the fixed scan execution type program to the execution type program is performed in the following timing.

• For the Universal model QCPU, LCPU

The execution type is changed when the execution of the fixed scan execution type is stopped at the END processing after the program control instruction execution.

• Basic model QCPU, High Performance model QCPU, Process CPU, and Redundant CPU

The execution of the fixed scan execution type is stopped at the execution of the program control instruction, and the execution type is changed at the END processing.

When the POFF instruction is executed, the output is turned OFF at the next scan, and the execution type will be the stand-by type at the second next scan and later. If executed prior to the output OFF processing, the program control instruction is ignored.

Program standby

PSTOP(P)

Basic | High performance | Process | Redundant | Universal | LCPU

(S): Character string for the name of the program file to be set in the stand-by status or head number of the devices where the character string data is stored (character string)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant $	Others
	Bit	Word		Bit	Word				
(S)	—	○		—				○	—

Processing details

- Places the file name program stored in the device designated by (S) in the stand-by status.
- Only the programs stored in the drive No. 0 (program memory/internal RAM) can be set as the stand-by type.
- The specified program is placed in the stand-by status when END processing is performed.
- This instruction will be given priority even in cases when a program execution type has been designated in the parameters.
- It is not necessary to designate the extension (.QPG) with the file name. (Only .QPG files will be acted on.)

Operation error

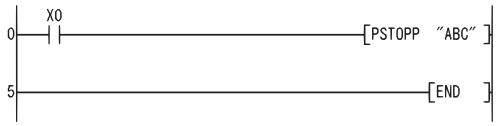
- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
2410	The program with the file name specified by (S) does not exist.	—	○	○	○	○	○
2412	The program type of the file name specified by (S) is the SFC program.	—	○	○	○	○	○
4101	The range of the device specified by (S) exceeds the range of the corresponding device.	—	○	○	○	○	○

Program example

- The following program places the program with the file name ABC in the stand-by status when X0 goes ON.

[Ladder Mode]



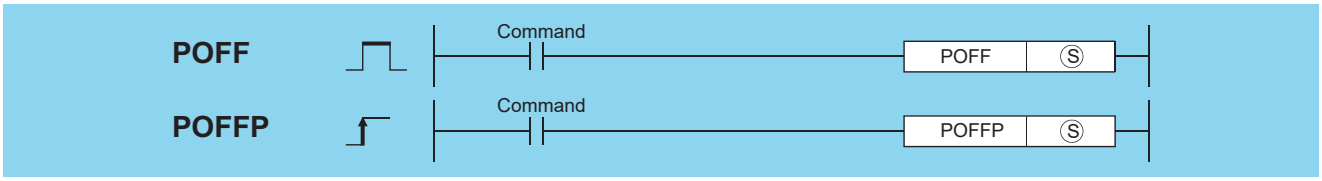
[List Mode]

Step	Instruction	Device
0	LD	X0
1	PSTOPP	"ABC"
5	END	

Program output OFF standby

POFF(P)

Basic | High performance | Process | Redundant | Universal | LCPU



(S): File name of the program to be set in the standby status by turning OFF the output, or the device where the file name is stored (character string)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant $	Others
	Bit	Word		Bit	Word				
(S)	—	○		—				○	—

Processing details

- Changes the execution type of the program with the file name stored in the device designated by (S).
  - Scan execution type: Turns OFF outputs at the next scan (Non-execution processing). Programs are set as the stand-by type after the subsequent scan.
  - Low speed execution type: Stops the execution of the low speed execution type program and turns OFF outputs at the next scan. Programs are set as the stand-by type after the subsequent scan.
  - Fixed scan execution type: The type of the program changes to the scan execution type and the program turns off outputs at the next scan (Non-execution processing). The type of the program changes to the standby type at the scan after that.
- Only the programs stored in the drive No. 0 (program memory) can be set as the stand-by type.
- This instruction will be given priority even in cases when a program execution type has been designated in the parameters.
- It is not necessary to designate the extension (.QPG) with the file name. (Only .QPG files will be acted on.)

Operation error

- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
2410	The program with the file name specified by (S) does not exist.	—	○	○	○	○	○
2411	The program with the file name designated by (S) is not registered in the parameters.	—	○	○	○	○	○
4101	The range of the device specified by (S) exceeds the range of the corresponding device.	—	○	○	○	○	○

Point

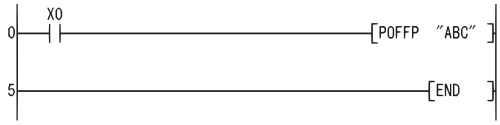
Non-execution processing is identical to the processing that is conducted when the condition contacts for the individual coil instructions are in the OFF state.
The operation results for the individual coil instructions following non-execution processing will be as follows, regardless of the ON/OFF status of the individual contacts:

- OUT instruction: Forced OFF
- SET instruction, RST instruction, SFT instruction, Basic instruction, Application instruction: Maintains status
- PLS instruction, Pulse generation instruction (□P): Processing identical to when condition contacts are OFF
- Current value of low speed/high speed timer: 0
- Current value of retentive timer, Current value of counter: Preserves

Program example

- The following program makes the program with the file name ABC non-executionable and places it in the standby status when X0 is turned ON.

[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	X0
1	POFFP	"ABC"
5	END	

Program scan execution registration

PSCAN(P)

Basic | High performance | Process | Redundant | Universal | LCPU

(S): File name of the program to be set as a scan execution type, or head number of the devices where the file name is stored (character string)

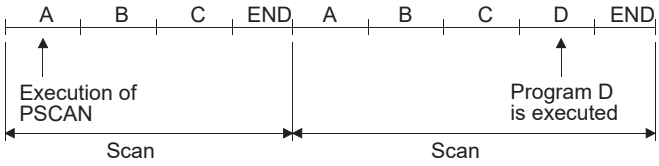
Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant $	Others
	Bit	Word		Bit	Word				
(S)	—	○		—				○	—

Processing details

- Sets the program whose file name is being stored at the device designated by (S) in the scan execution type.
- Only the programs stored in the drive No. 0 (program memory/internal RAM) can be set as the scan execution type.
- Designated programs assume the scan execution type with END processing.

Ex.

When programs A, B, and C exist and program A performs "PSCAN" of program D.



- This instruction will be given priority even in cases when a program execution type has been designated in the parameters.
- It is not necessary to designate the extension (.QPG) with the file name. (Only .QPG files will be acted on.)

Operation error

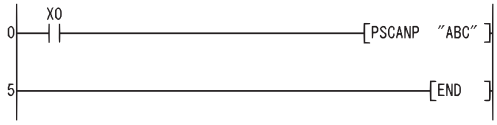
- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
2410	The program with the file name specified by (S) does not exist.	—	○	○	○	○	○
2411	The program with the file name designated by (S) is not registered in the parameters.	—	○	○	○	○	○
2504	The specified file name is the SFC program, and the SFC program for the other file name has been already started. (Dual activation error of the SFC program)	—	○	○	○	—	—
4101	The range of the device specified by (S) exceeds the range of the corresponding device.	—	○	○	○	○	○
4131	The specified file name is the SFC program, and the SFC program for the other file name has been already started. (Dual activation error of the SFC program)	—	—	—	—	○	○

Program example

• The following program sets the program with file name ABC as scan execution type when X0 is turned ON.

[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	X0
1	PSCANP	"ABC"
5	END	

Program low speed execution registration

PLOW(P)

Basic | High performance | Process | Redundant | Universal | LCPU

(S): File name of the program to be set as a low speed execution type, or head number of the devices where the file name is stored (character string)

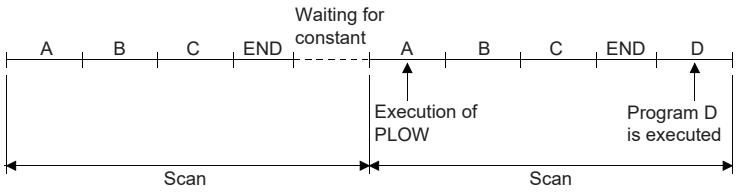
Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant $	Others
	Bit	Word		Bit	Word				
(S)	—	○		—				○	—

Processing details

- Sets the program whose file name is being stored at the device designated by (S) in low-speed execution type.
- Only the programs stored in the drive No. 0 (program memory/internal RAM) can be set as the low speed execution type.
- Designated programs assume the low speed execution type with END processing.

Ex.

When programs A, B, and C exist and program A performs "PLOW" of program D. (Assume that the constant scan has been set.)



- This instruction will be given priority even in cases when a program execution type has been designated in the parameters.
- It is not necessary to designate the extension (.QPG) with the file name. (Only .QPG files will be acted on.)

Operation error

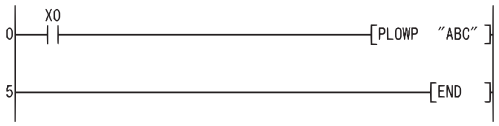
- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
2410	The program with the specified file name does not exist.	—	○	○	—	—	—
4235	There is a CHK instruction in the program with the specified file name.	—	○	○	—	—	—

Program example

- The following program sets the program with file name ABC as low-speed execution type when X0 is turned ON.

[Ladder Mode]



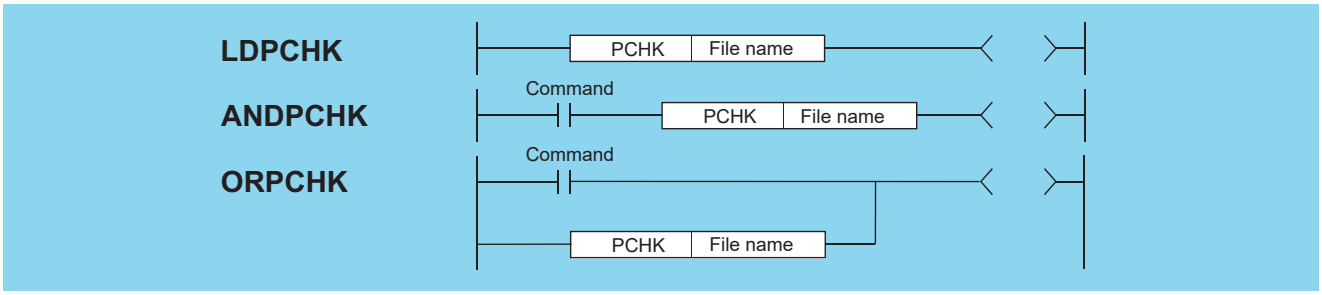
[List Mode]

Step	Instruction	Device
0	LD	X0
1	PLOWP	"ABC"
5	END	

Program execution status check

LDPCHK, ANDPCHK, ORPCHK

Basic | High performance | Process | Redundant | Universal | LCPU



(S): File name of the program whose execution status will be checked (character string)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant $	Others
	Bit	Word		Bit	Word				
(S)	—							○	—

Processing details

- Checks whether the program of the specified file name is in execution or not (non-execution).
- The instruction is in conduction when the program of the specified file name is in execution, and the instruction is in non-conduction when the program is in non-execution.
- Specify the file name without an extension (.QPG). For example, specify "ABC" when the file name is ABC.QPG.
- Non-execution indicates that the program execution type is a stand-by type. Execution indicates that the program execution type is a scan execution type (including during output OFF (during non-execution processing)), low speed execution type or fixed scan execution type.

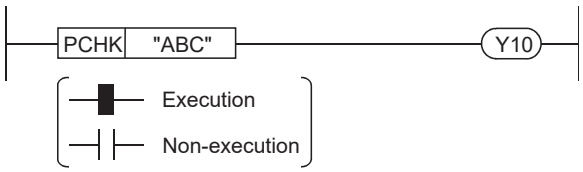
Operation error

- In any of the following cases, an operation error occurs, the error flag (SM0) turns on, and an error code is stored in SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
2410	The program with the specified file name does not exist.	—	○	○	○	—	—

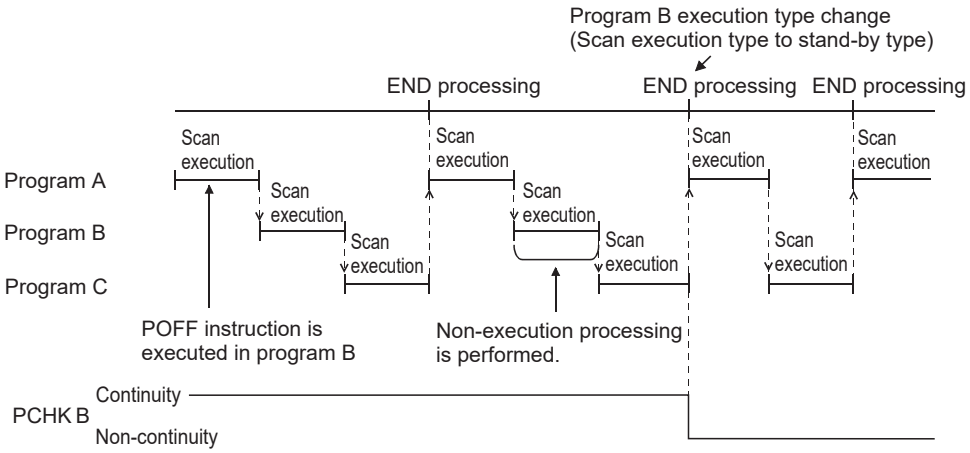
Program example

- Program that keeps Y10 ON when the program file "ABC.QPG" is being executed.



Point

The PCHK instruction is in conduction when the program of the specified file name (target program) is in execution, and the instruction is in non-conduction when the program is in non-execution.
When the target program is set to non-execution (stand-by type) with the POFF instruction, the PCHK instruction is in conduction while the non-execution processing of the target program is being performed.
At the END processing of the scan where the non-execution processing is completed, the target program is put into non-execution (stand-by type), and the PCHK instruction is brought into non-conduction.
Therefore, note that if the PCHK instruction is executed for the program where the non-execution processing has been completed by the POFF instruction, the PCHK instruction may be brought into conduction.
The following chart shows the operation performed when program A executes the POFF instruction of program B and program C executes the PCHK instruction of program B with the programs being executed in order of program A, program B, and program C.



7.18 PID Instruction

Overview

Types of PID instructions

The following types of PID instructions are available.

Type	Compatible model	Reference
Process control instruction	• QnPHCPU • QnPRHCPU • QnUDPVCPU	MELSEC-Q Programming/Structured Programming Manual (Process Control Instructions)
PID control instruction	• QnACPU • QnCPU • QnHCPU • QnPRHCPU • QnUCPU • QnUDVCPU • QnUDPVCPU • LCPU	MELSEC-Q/L/QnA Programming Manual (PID Control Instructions)
PID operation instruction	• QnUDVCPU • QnUDPVCPU • LCPU	Page 735 PID control

PID operation instruction

To come close to the set value (SV), the PID operation instruction calculates the manipulated value (MV) from the process value (PV) by combining the P operation (proportional operation), I operation (integral operation), and D operation (derivative operation).

■Alarm output function

This function can turn ON the alarm output according to the rates of change of input (process value) and output (value).

■Settings of upper and lower limits of output value

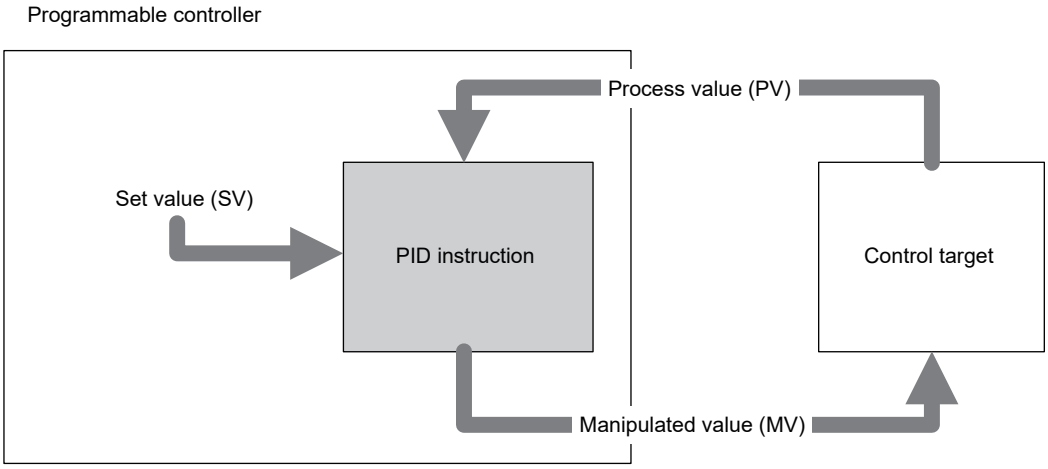
This function can set the upper and lower limit values of output.

■Auto tuning function

Proportional gain (K_P), integral time (T_I), and derivative time (T_D) can be set automatically. The limit cycle method or step response method can be selected.

■Operation method of PID operation instruction

PID velocity type/process value derivative type operations are performed.



Operation method of PID operation instruction (reference)

The instruction performs the PID operation using the speed type or process value differential type operational expression.
The operational expression of direct action or reverse action is performed according to the data in bit 0 of (S3)+1 (operation setting (ACT))
Each value required in the operation calculates using parameters specified (S3) or later.

Bit 0 of (S3)+1	PID operation method
Direct action (OFF)	$\Delta MV = K_P \left\{ (EV_n - EV_{n-1}) + \frac{T_S}{T_I} EV_n + D_n \right\}$ $EV_n = PV_{nf} - SV$ $D_n = \frac{T_D}{T_S + K_D \cdot T_D} (-2PV_{nf-1} + PV_{nf} + PV_{nf-2}) + \frac{K_D \cdot T_D}{T_S + K_D \cdot T_D} \cdot D_{n-1}$ $MV_n = \sum \Delta MV$
Reverse action (ON)	$\Delta MV = K_P \left\{ (EV_n - EV_{n-1}) + \frac{T_S}{T_I} EV_n + D_n \right\}$ $EV_n = SV - PV_{nf}$ $D_n = \frac{T_D}{T_S + K_D \cdot T_D} (2PV_{nf-1} - PV_{nf} - PV_{nf-2}) + \frac{K_D \cdot T_D}{T_S + K_D \cdot T_D} \cdot D_{n-1}$ $MV_n = \sum \Delta MV$

■Symbol

- EV_n: Deviation at current sample
- EV_{n-1}: Deviation before one cycle
- SV: Set value
- PV_{nf}: Process value at current sample (after filter)
- PV_{nf-1}: Process value before one cycle (after filter)
- PV_{nf-2}: Process value before two cycles (after filter)
- ΔMV: Rate of output change
- MV_n: Current manipulated value
- D_n: Current derivative term
- D_{n-1}: Derivative term before one cycle
- K_P: Proportional gain
- T_S: Sampling period
- T_I: Integral constant
- T_D: Derivative constant
- K_D: Derivative gain

■Calculation formula of PV_{nf}

Process value after the filter (PV_{nf}) = PV_n + L(PV_{nf-1} - PVn)

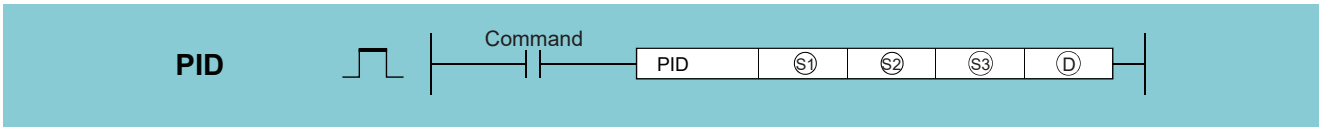
- PV_{nf}: Process value at current sample
- L: Filter coefficient
- PV_{nf-1}: Process value before one cycle (after filter)

PID control

PID

Basic | High performance | Process | Redundant | Ver. Universal | Ver. LCPU

• QnUDVCPU, QnUDPVCPU, LCPU: The serial number (first five digits) is "16112" or later.

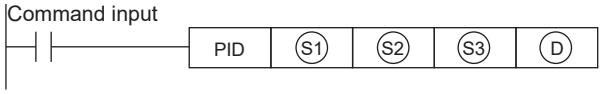


(S1): Device number where the set value (SV) is stored (device name)
(S2): Device number where the process value (PV) is stored (device name)
(S3): Start number of devices where parameters are stored (device name)
(D): Device number where the manipulated value (MV) is stored (device name)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant	Others
	Bit	Word		Bit	Word				
(S1)	—	○	○	—	○	○	—	—	—
(S2)	—	○	○	—	○	○	—	—	—
(S3)	—	○	○	—	—	—	—	—	—
(D)	—	○	○	—	○	○	—	—	—

Processing details

• When the program is run with the set value (S1), process value (S2), and parameters ((S3) to (S3)+6) set, the manipulated value (MV) is stored in the output value (D) every sampling time (S3).



• The setting items of arguments are listed below.

Setting item		Description	Setting range
(S1)	Set value (SV)	Set the set value (SV). However, when the limit cycle method is used, the set values for auto tuning and PID control may be different. In this case, a value obtained by adding a bias value is set for the time being and, when the auto tuning flag turns OFF, the actual set value needs to be stored.	1 point
(S2)	Process value (PV)	Set the input value for PID operation.	1 point
(S3)	Parameters*1	• When the limit cycle method is used Devices of 29 points from the start device specified in (S3) are occupied. • When the step response method is used When none of bits 1, 2, and 5 of the action setting (ACT) is "0", devices of 25 points from the start device specified in (S3) are occupied. When all of bits 1, 2, and 5 of the action setting (ACT) are "0", devices of 20 points from the start device specified in (S3) are occupied.	See the left.
(D)	Manipulated value (MV)	• When normal processing is performed The user sets the initial output value before driving the instruction. After the instruction is driven, the operation result is stored. • When the limit cycle method is used The ULV or LLV value is automatically output during auto tuning, and the given MV value is set after the end of auto tuning. • When the step response method is used The user sets the step output value before driving the instruction. The MV output cannot be changed by the PID instruction during auto tuning.	1 point

*1 When auto tuning is not used, the same number of points as that used when the step response method is used is occupied.

• The setting items of the parameter (S3) are listed below.

Setting item			Description	Remark
(S3)	Sampling time (T_S)		1 to 32767 ms	The instruction cannot be used with a value smaller than the operation cycle.
(S3)+1	Operation setting (ACT)	Bit 0	0: Direct action 1: Reverse action	Specify the operation direction.
		Bit 1	0: Disable alert for rate of change in input. 1: Enable alert for rate of change in input.	—
		Bit 2	0: Disable alert for rate of change in output. 1: Enable alert for rate of change in output.	Do not set bits 2 and 5 to ON simultaneously.
		Bit 3	Unusable	—
		Bit 4	0: Disable auto tuning. 1: Enable auto tuning.	—
		Bit 5	0: Disable upper and lower limit setting of output value. 1: Enable upper and lower limit setting of output value.	Do not set bits 2 and 5 to ON simultaneously.
		Bit 6	0: Step response method 1: Limit cycle method	Select the auto tuning mode.
		Bits 7 to 15	Unusable	—
(S3)+2	Input filter constant (α)		0 to 99%	If 0 is specified, no input filter is used.
(S3)+3	Proportional gain (K_P)		1 to 32767%	—
(S3)+4	Integral time (T_I)		0 to 32767 × 100ms	Specifying 0 assumes ∞. (No integration)
(S3)+5	Derivative gain (K_D)		0 to 100%	Specifying 0 means no derivative gain.
(S3)+6	Derivative time (T_D)		0 to 32767% × 10ms	Specifying 0 means no derivation.
(S3)+7 to (S3)+19	Occupied for internal processing of PID operation, and therefore it is prohibited to change data.			
(S3)+20*2	Rate of change in input (increase side) Alert setting value		0 to 32767 × 100ms	Action direction (ACT): (S3)+1 Enabled when bit 1 = 1.
(S3)+21*2	Rate of change in input (decrease side) Alert setting value		0 to 32767	Action direction (ACT): (S3)+1 Enabled when bit 1 = 1.
(S3)+22*2	Rate of change in output (increase side) Alert setting value		0 to 32767	Action direction (ACT): (S3)+1 Enabled when bit 2 = 1 and bit 5 = 0.
	Output upper limit setting value		-32768 to 32767	Action direction (ACT): (S3)+1 Enabled when bit 2 = 0 and bit 5 = 1.
(S3)+23*2	Rate of change in output (decrease side) Alert setting value		0 to 32767	Action direction (ACT): (S3)+1 Enabled when bit 2 = 1 and bit 5 = 0.
	Output lower limit setting value		-32768 to 32767	Action direction (ACT): (S3)+1 Enabled when bit 2 = 0 and bit 5 = 1.
(S3)+24*2	Alarm output	Bit 0	0: Rate of change in input (increase side) not exceeded 1: Rate of change in input (increase side) exceeded	Action direction (ACT): (S3)+1 Enabled when bit 1 = 1 or bit 2 = 1.
		Bit 1	0: Rate of change in input (decrease side) not exceeded 1: Rate of change in input (decrease side) exceeded	—
		Bit 2	0: Rate of change in output (increase side) not exceeded 1: Rate of change in output (increase side) exceeded	—
		Bit 3	0: Rate of change in output (decrease side) not exceeded 1: Rate of change in output (decrease side) exceeded	—
(S3)+25	PV value threshold (Hysteresis)width (SHpv)		Set depending to the fluctuation of process value (PV).	Action direction (ACT) Bit 6: Occupied when the limit cycle method (ON) is selected.
(S3)+26	Upper limit value of output (ULV)		Set the maximum output value (ULV) of manipulated value (MV).	
(S3)+27	Lower limit value of output value (LLV)		Set the minimum output value (LLV) of manipulated value (MV).	
(S3)+28	Wait setting parameter (K_W) from the end of tuning cycle to the start of PID control		-50 to 32717%	

*2 (S3)+20 to (S3)+24 are occupied when bit 1 of the action setting (ACT) in (S3)+1 = 1 and bit 2 = 1 or bit 5 = 1.

Precautions

- The instruction can be executed multiple times (no limit in the number of loops). However, be careful not to duplicate (S3) used for operation and the device number specified by (D).

Operation error

- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4101	The device specified by (S3) exceeds the range of the number of device points.	—	—	—	—	○	○
9100	A data error occurred in the control parameter setting values or during PID operation because the sampling time (T_s) is out of the target range ($T_s \leq 0$).	—	—	—	—	○	○
9101	A data error occurred in the control parameter setting values or during PID operation because the input filter constant (α) is out of the target range ($\alpha<0$ or $100 \leq \alpha$).	—	—	—	—	○	○
9102	A data error occurred in the control parameter setting values or during PID operation because the proportional gain (K_P) is out of the target range ($K_P<0$).	—	—	—	—	○	○
9103	A data error occurred in the control parameter setting values or during PID operation because the integral time (T_I) is out of the target range ($T_I<0$).	—	—	—	—	○	○
9104	A data error occurred in the control parameter setting values or during PID operation because the derivative gain (K_D) is out of the target range ($K_D<0$ or $201 \leq K_D$).	—	—	—	—	○	○
9105	A data error occurred in the control parameter setting values or during PID operation because the derivative time (T_D) is out of the target range ($T_D<0$).	—	—	—	—	○	○
9106	Sampling time (T_S) ≤ operation cycle	—	—	—	—	○	○
9107	Rate of change in the process value went out of the range ($\Delta PV<-32768$ or $32767<\Delta PV$).	—	—	—	—	○	○
9108	Error value went out of the range (EV<-32768 or 32767<EV).	—	—	—	—	○	○
9109	Integrally calculated value went out of the range (from -32768 to 32767).	—	—	—	—	○	○
9110	Derivative value went out of the range because derivative gain (K_D) went out of the range.	—	—	—	—	○	○
9111	Derivatively calculated value wend output of the range (from -32768 to 32767).	—	—	—	—	○	○
9112	PID operation result went out of the range (from -32768 to 32767).	—	—	—	—	○	○
9113	PID output upper limit setting value<output lower limit setting value	—	—	—	—	○	○
9114	Error in the setting value of alert for rate of change in PID input or in the setting value of alert for rate of change in output(setting value<0)	—	—	—	—	○	○
9115	The result of auto tuning by the step response method is improper as follows. • The error at the start of auto tuning is 150 or less. • The error at the end of auto tuning is 1/3 of that at the start of auto tuning.	—	—	—	—	○	○
9116	A mismatch occurred in the action direction of auto tuning by the step response method. • A mismatch occurred between the action direction assumed from the process value at the time of starting auto tuning and the direction actual action by auto tuning output.	—	—	—	—	○	○
9117	Auto tuning by the step response method is faulty as follows. • Auto tuning failed to operate correctly because the set value fluctuated during auto tuning.	—	—	—	—	○	○
9118	Auto tuning by the limit cycle method resulted in an error in the values set for auto tuning output as follows. • ULV (upper limit) ≤ LLV (lower limit)	—	—	—	—	○	○
9119	Auto tuning by the limit cycle method resulted in an error in the values set for the auto tuning PV threshold (hysteresis) as follows. • $SH_{PV}<0$	—	—	—	—	○	○
9120	Auto tuning by the limit cycle method resulted in an error in the auto tuning transition status as follows. • The data in the device that controls the transition status has been rewritten abnormally.	—	—	—	—	○	○

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
9121	Auto tuning by the limit cycle method resulted in an error by exceeding the auto tuning measurement time as follows. • $\tau_{on}>\tau$, $\tau_{on}<0$, $\tau<0$	—	—	—	—	○	○
9122	Auto tuning by the limit cycle method went out of the range of proportional gain (K_P=0 to 32767) as the result of auto tuning as follows. • The change in the process values (PV) is small with regard to the output values.	—	—	—	—	○	○
9123	Auto tuning by the limit cycle method went out of the range of the integral time (T_I=0 to 32767) as the result of auto tuning as follows. • More time than necessary is taken for auto tuning.	—	—	—	—	○	○
9124	Auto tuning by the limit cycle method went out of the range of the derivative time (T_D=0 to 32767) as the result of auto tuning as follows. • More time than necessary is taken for auto tuning.	—	—	—	—	○	○

Program example

- For the program examples of the PID instruction, see Page 753 Example of practical program (step response method).

Parameters

This section explains the parameters of the PID operation instruction (PID).

Sampling time (T_S): (S3)

Setting range: 1 to 32767ms

Set the cycle (ms) for PID operation.

• PID control or auto tuning (limit cycle method)

"Operation cycle of programmable controller"<"sampling time"

• Auto tuning (step response method)

1000ms (1s) or more

■Maximum error

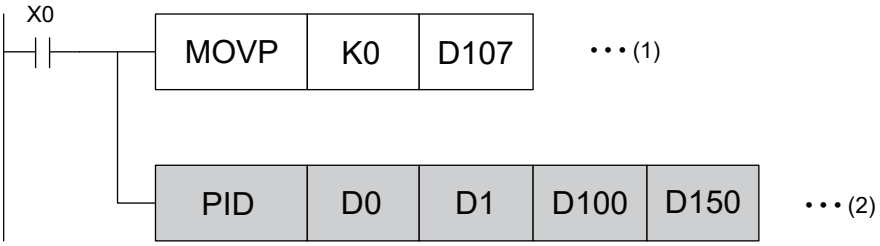
The maximum error of the sampling time (T_S) ranges from -(1 operation cycle + 1ms) to + (1 operation cycle).

• When the sampling time (T_S) is a small value

The variations of the above maximum error may become a problem. Execute the instruction in constant scan mode or program it in the timer interrupt routine.

• When the sampling time is shorter than one operation cycle of programmable controller

A PID operation error (9106H) occurs but PID operation is executed assuming sampling time (T_S) = operation cycle. In this case, use the PID operation instruction in the timer interrupt, and clear (S3)+7 immediately before execution of the PID operation instruction.



(1) Execute (S3)+7 (clear the internal processing register with the pulse conversion command during initial execution of the interrupt routine).
(2) Execute the PID operation.

Action setting (ACT): (S3)+1

■Direct action/reverse action: (S3)+1 Bit 0

Setting range: OFF = direct action/ON = reverse action

Select the direct action or reverse action for the PID control direction.

• In the case of auto tuning (limit cycle method)

For auto tuning, the user needs to set the relevant PID control direction, direct action or reverse action.

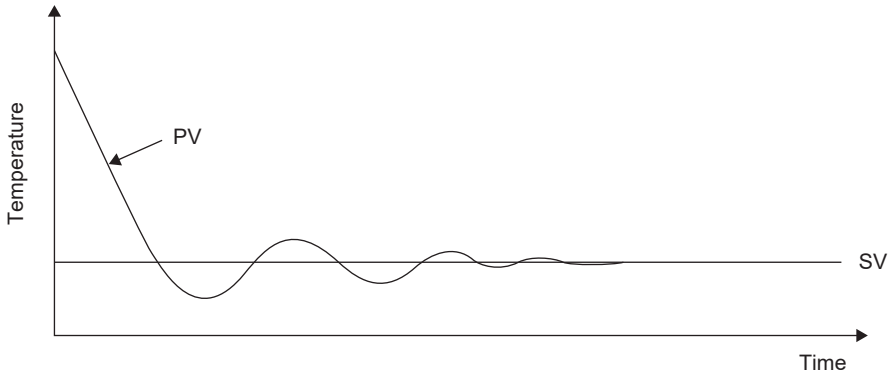
• In the case of auto tuning (step response method)

At completion of auto tuning executed in whichever mode, direct action or reverse action mode, setting is made automatically.

[Direct action ((S3)+1 Bit 0 = OFF)]

The manipulated value (MV) increases as the process value (PV) increases more from the set value (SV).

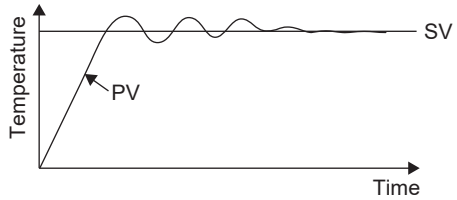
For example, cooling is the direct action.



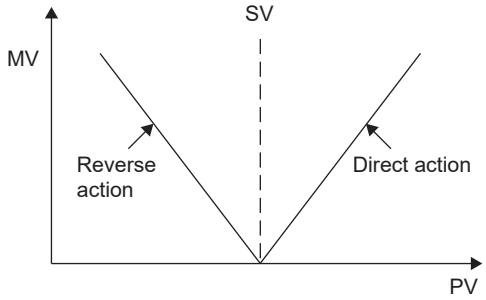
[Reverse action ((S3)+1 Bit 0 = ON)]

The manipulated value (MV) increases as the process value (PV) decreases more from the set value (SV).

For example, heating is the reverse action.



The following figure shows the relationships between the direct/reverse actions and the manipulated value (MV), process value (PV), and set value (SV).



■Alert setting (rate of change in input, rate of change in output): (S3)+1 Bits 1, 2

Setting range: OFF = Disable alert for change in input/Enable alert for change in input

The rates of changes in input and output can be checked. The result of checking can be confirmed by (S3)+24. (For the operation of input/output value upper/lower limit alarm output, see (S3)+24 (warning output flag operation).

[Rate of change in input ((S3)+1 Bit 1)]

To use the alert for the rate of change in input, set the following bits to ON and set the values to be checked.

Setting item			Description	Setting range
Action setting (ACT)	(S3)+1	Bit 1	Alert for rate of change in input	ON: Enable OFF: Disable
Setting value for alert for rate of change in input	(S3)+20		Setting value for alert for rate of change in input (increase side)	0 to 32767
	(S3)+21		Setting value for alert for rate of change in input (decrease side)	

[Rate of change in output ((S3)+1 Bit 2)]

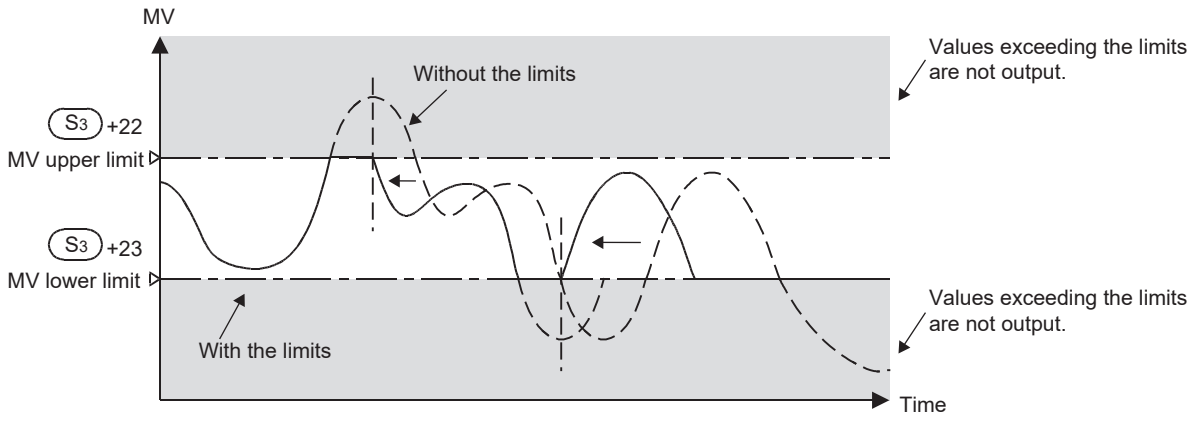
To use the alert for the rate of change in output, set the following bits to ON and set the values to be checked.

Setting item			Description	Setting range
Action setting (ACT)	(S3)+1	Bit 2	Alert for rate of change in output	ON: Enable OFF: Disable
		Bit 5	Settings of upper and lower limits of output value	Always set to OFF
Setting value for alert for rate of change in input	(S3)+22		Setting value for alert for rate of change in output (increase side)	0 to 32767
	(S3)+23		Setting value for alert for rate of change in output (decrease side)	

The rate of change is determined by "previous value - this value".

■Settings of upper and lower limits of output value: (S3)+1 Bit 5

The settings of upper and lower limits of output value are as follows.



Setting the upper and lower limits of output value has the effect of suppressing the increase of the integral term of PID control. When using this function, be sure to set bit 2 of (S3)+1 to OFF.

Setting item			Description	Setting range
Action setting (ACT)	(S3)+1	Bit 2	Alert for rate of change in output	Always set to OFF
		Bit 5	Settings of upper and lower limits of output value	ON: Enable OFF: Disable

Input filter (α): (S3)+2

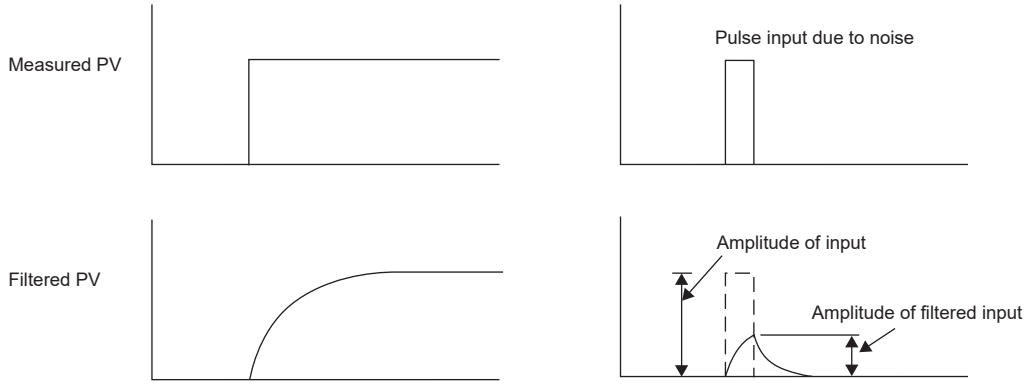
Setting range: 0 to 99[%]

PID control: proportional operation, integral operation, derivative operation

The input filter (α) is a software filter that reduces the variations caused by process value (PV) noise. Effects of noise can be suppressed by setting the time constant of the filter according to the characteristics and noise levels of the control target.

- Too small a time constant reduces the effects of the filter.
- Too large a time constant deteriorates the response of input.

The input filter (α) acts on the set value (SV) and accordingly affects the proportional operation, integral operation, and derivative operation.



Proportional gain (K_P): (S3)+3

Setting range: 1 to 32767[%]

PID control: proportional operation

The manipulated value (MV) increases in proportion to the error (difference between the set value (SV) and process value (PV)) in the proportional operation. This proportion is called the proportional gain (K_P) and is expressed in the following relational expression.

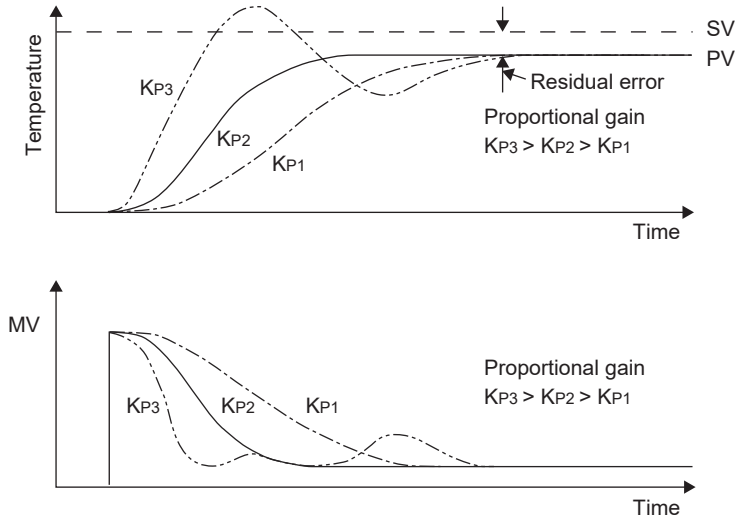
Manipulated value (MV) = Proportional gain (K_P) × error (EV)

The reciprocal of the proportional gain (K_P) is called the proportional band.

As the proportional gain (K_P) increases, the movement of getting the process value (PV) closer to the set value (SV) will be stronger.

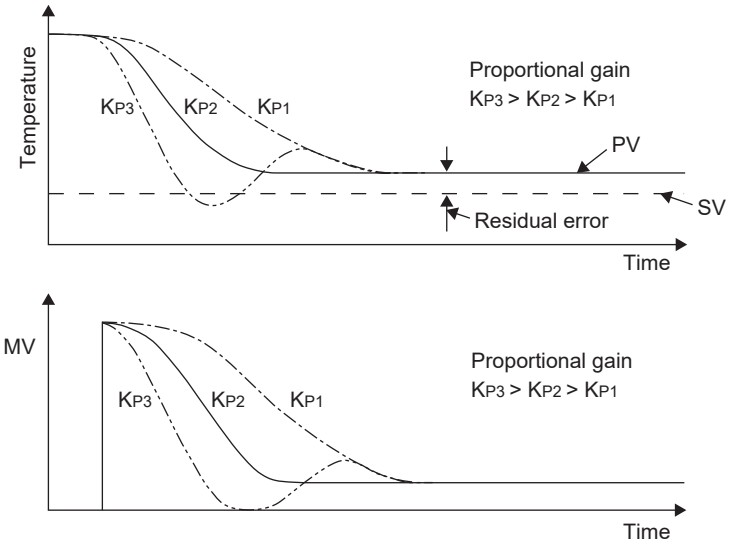
Ex.

Proportional operation (P operation) in heating (reverse action)



Ex.

Proportional operation (P operation) in cooling (direct action)



Integral time (T_I): (S3)+4

Setting range: 0 to 32767 [×100ms] (0 is treated as ∞. (No integration))

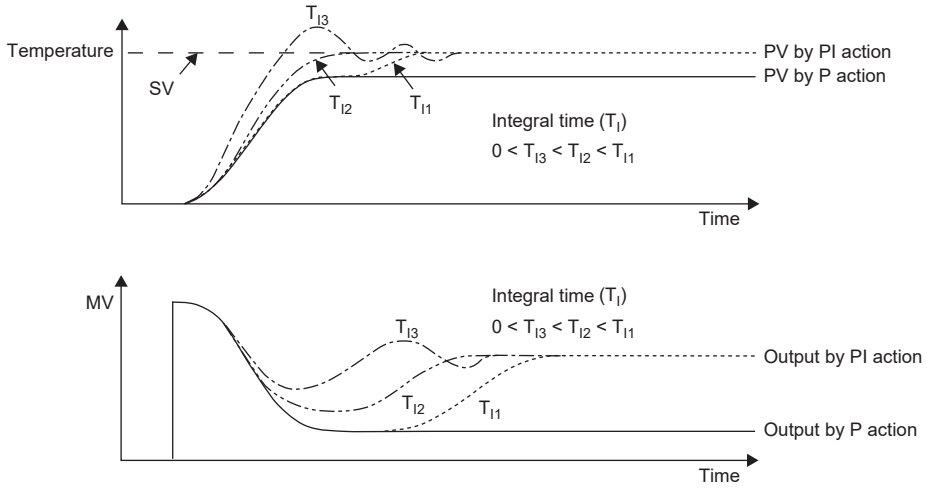
PID control: integral operation

The time taken from when an error occurs in the integral operation to when the output of the integral operation becomes the output of the proportional operation is called the integral time expressed in T_I.

When T_I is reduced, the integral operation becomes stronger.

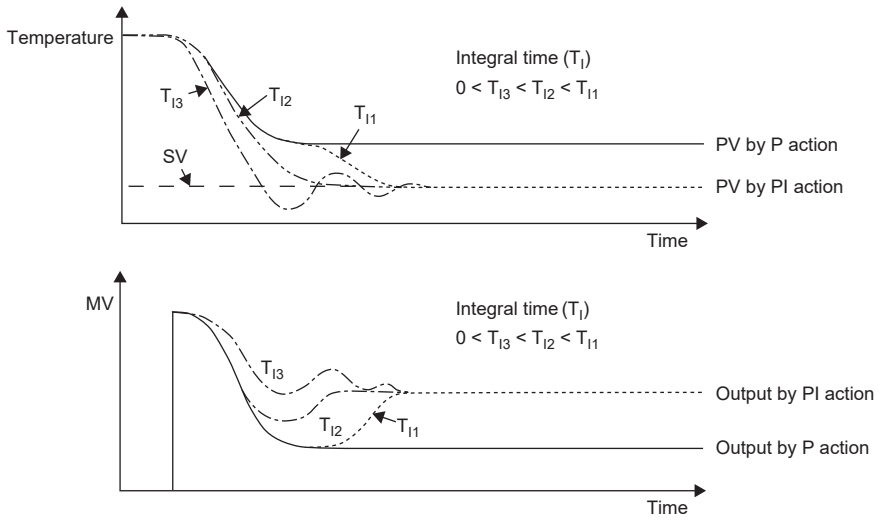
Ex.

PI operation in heating (reverse action)



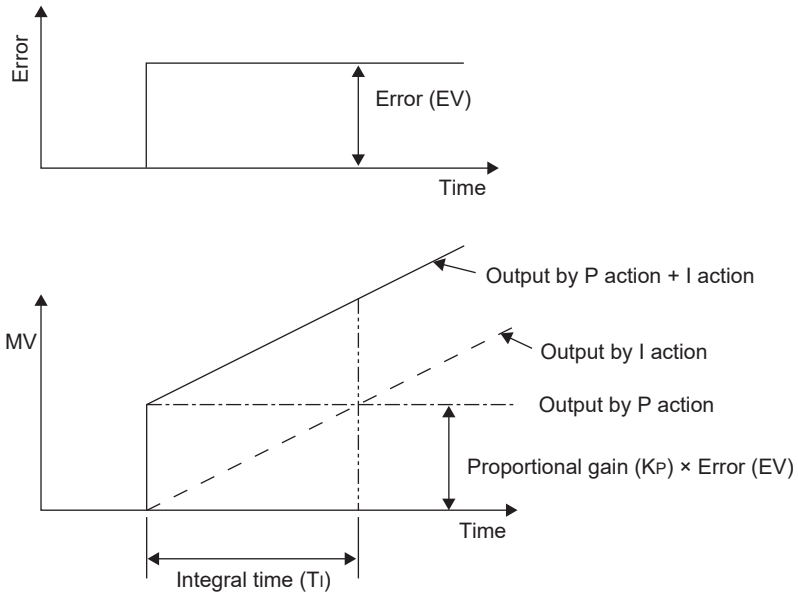
Ex.

PI operation in cooling (direct action)



Point

The integral operation is an operation to change the output to remove the error generated in succession. This can eliminate the residual error occurring in the proportional operation.



Derivative gain (K_D): (S3)+5

Setting range: 0 to 100[%]

PID control: derivative operation

A filter is applied to the output by the derivative operation. The derivative gain (K_D) affects only the derivative operation.

- When the derivative gain (K_D) is small, output responds instantaneously, specifically to a change caused in the process value by disturbance.
- When the derivative gain (K_D) is large, output takes time to respond to a change caused in the process value by disturbance.

Point

Set the derivative gain (K_D) to 0, and make adjustment with the input filter (α). If the change in output responds too sensitive to disturbance, increase the value.

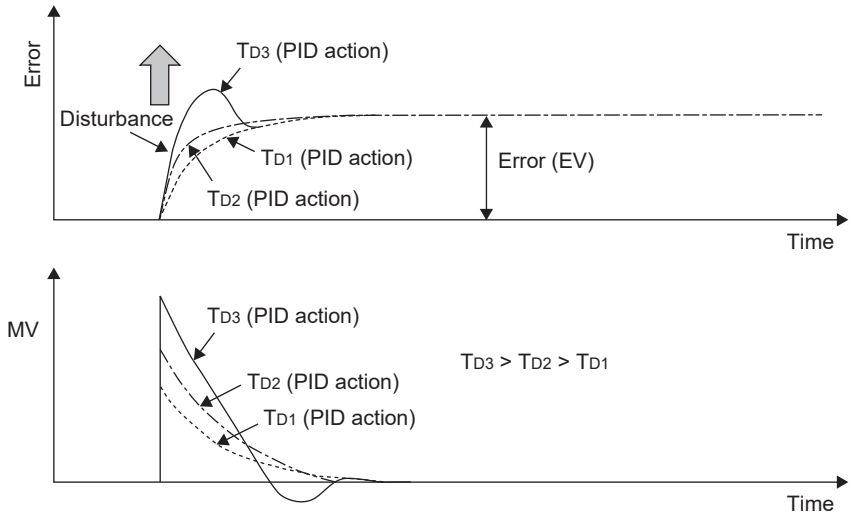
Derivative time (T_D): (S3)+6

Setting range: 0 to 32767 [×10ms]

PID control: derivative operation

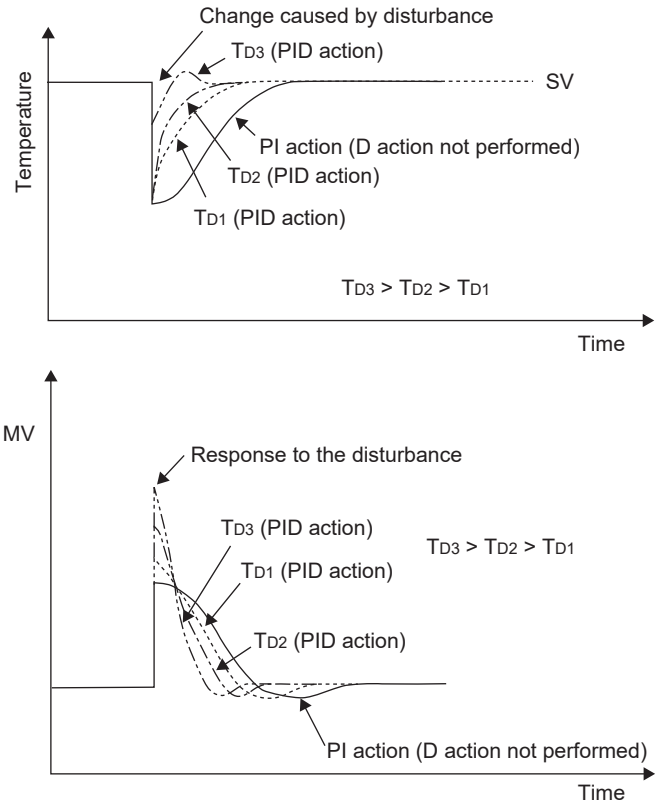
PID control is used to sensitively react to changes caused in the process value (PV) by disturbance and minimize the changes.

- Increasing the derivative time (T_D) enhances the movement to prevent the controlled system from varying greatly due to disturbance.



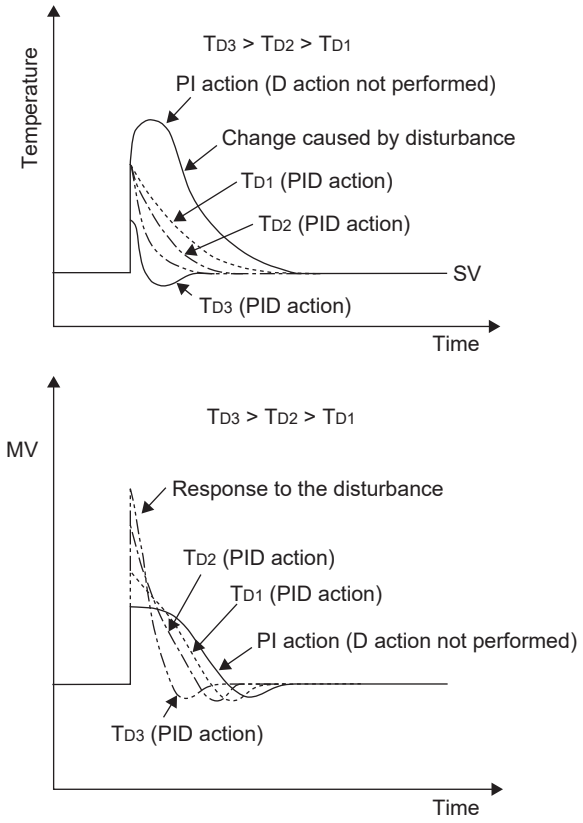
Ex.

PID operation in heating (reverse action)



Ex.

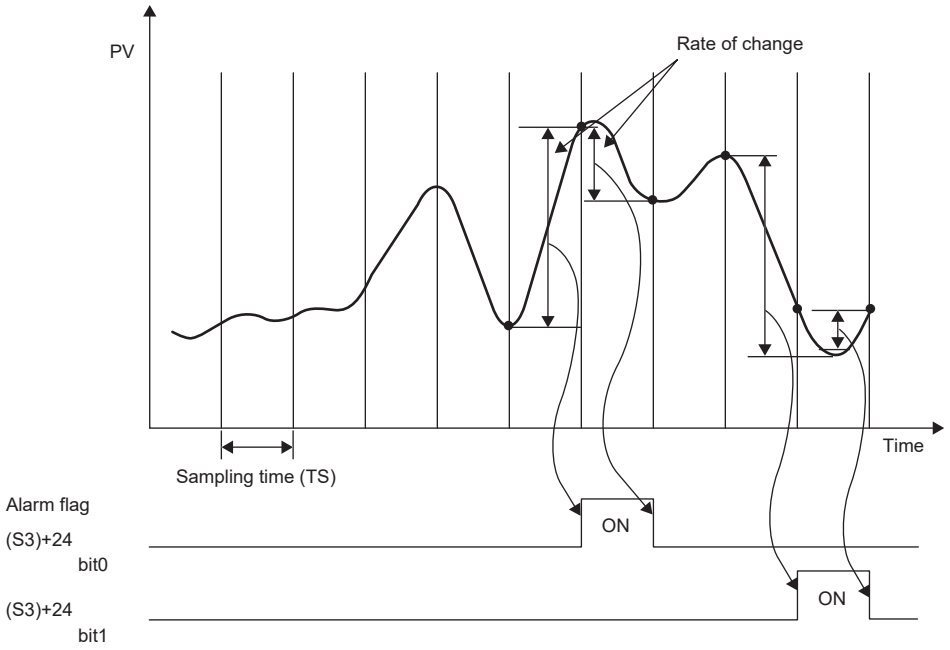
PID operation in cooling (direct action)



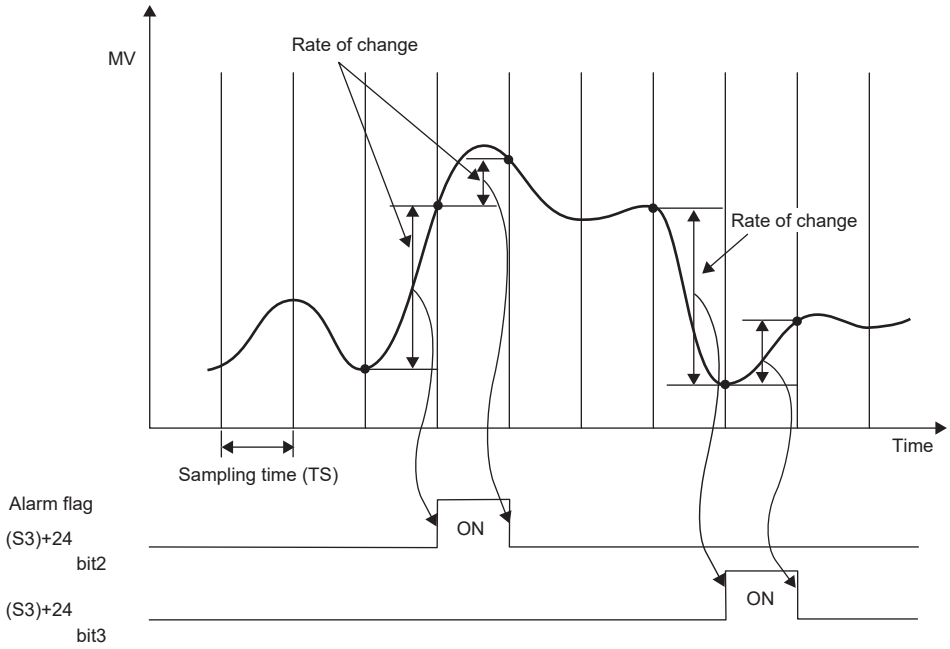
Warning output flag operation: (S3)+24

When the specified rate of input/output changes is exceeded, each bit of (S3)+24 turns on as an alarm flag immediately after the execution of the PID operation instruction.

- Rate of input change (bit 1 of (S3)+1 is ON)



- Rate of output change (bit 2 of (S3)+1 is ON)



Auto tuning

This section explains the auto tuning function of the PID operation instruction.

The auto tuning function automatically sets the proportional gain and integral time that are important constants in order to optimize PID control. The auto tuning function can be implemented in two methods: limit cycle method and step response method.

Limit cycle method

■Parameters set by auto tuning (limit cycle method)

- (S3)+3 (proportional gain (K_P))
- (S3)+4 (integral time (T_I))
- (S3)+6 (derivative time (T_D))

■Auto tuning procedure

No.	Procedure	Description
1	Set the direct action or reverse action.	Set the direct action or reverse action flag (bit 0) of (S3)+1 (action setting (ACT)).
2	Select the auto tuning method. (Limit cycle method)	Set the auto tuning method (bit 6) of (S3)+1 (action setting (ACT)) to ON. (If it is set to OFF, auto tuning is performed in Page 751 Step response method.)
3	Set the auto tuning flag to ON.	Set bit 4 of (S3)+1 (action setting (ACT)) to ON.
4	Set the input filter.	Set (S3)+2 (input filter constant (α)).
5	Set the sampling time.	Set (S3) (sampling time (T_S)).
6	Set the maximum output value (ULV).	Set the maximum output value (ULV) in (S3)+26 (upper limit of output value (ULV)).
7	Set the minimum output value (LLV).	Set the minimum output value (LLV) in (S3)+27 (lower limit of output value (LLV)).
8	Set the threshold value (hysteresis) (SH_{PV}).	Set (S3)+25 (PV value threshold (hysteresis) width (SH_{PV}).
9	Set the set value (SV).	Set the set value (SV) in (S1) of the PID operation instruction.
10	Set the command input of the PID operation instruction to ON to start auto tuning.	Auto tuning is performed according to the process value (PV).

When tuning is completed, the the auto tuning flags (bits 4 and 6) of (S3)+1 (action setting (ACT)) are set to OFF.

■Determining the three PID constants (limit cycle method) [Reference]

To obtain good control result of PID control, optimal values of individual constants (parameters) appropriate to the controlled system need to be determined. Here, the limit cycle method is explained as a method of determining the amplitude (a) and vibration cycle (τ, τ_{on}) of the input values and calculating the proportional gain (K_P), integral time (T_I), and derivative time (T_D) using the expressions provided in "operation characteristics and three constants" below.

• Limit cycle method

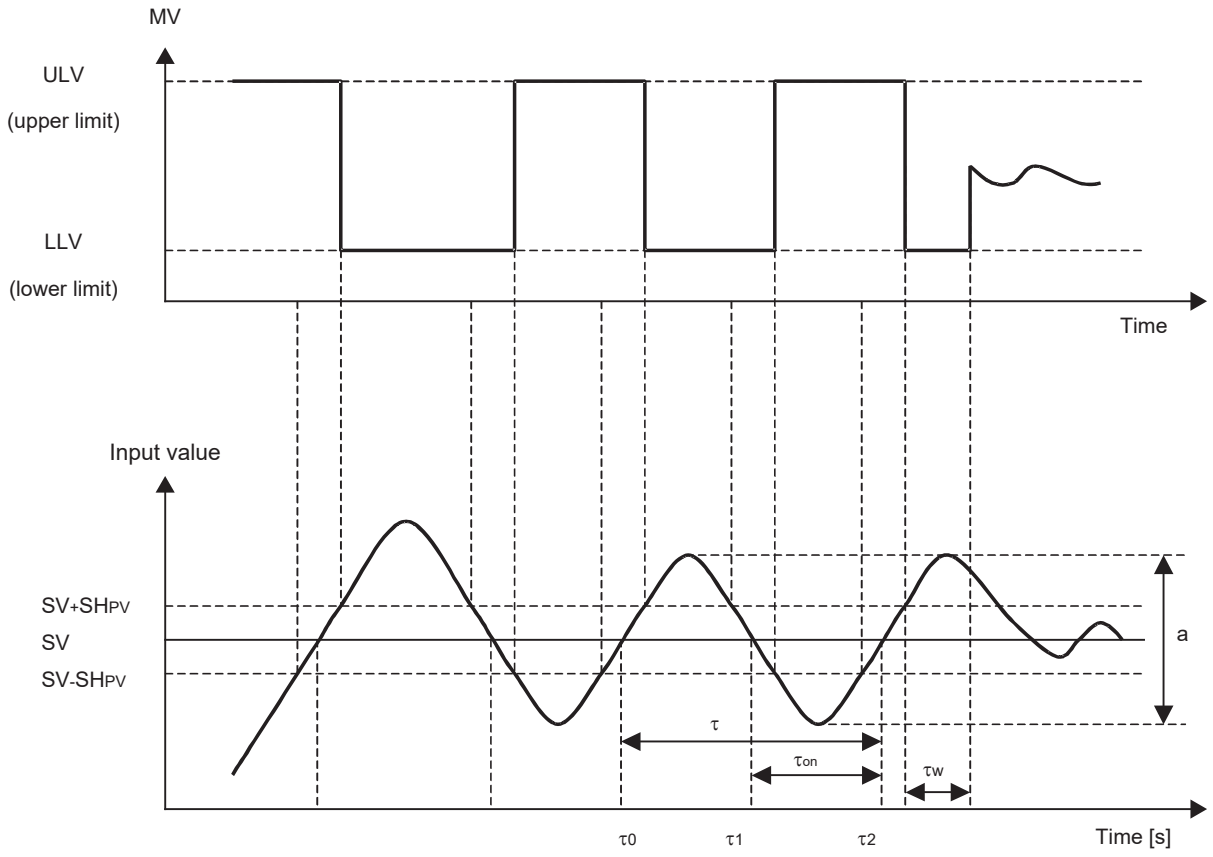
This method determines three PID constants by measuring a change generated in the input value when two-position control (outputting the output upper limit value (ULV) and output lower limit value (LLV) by switching them according to the error) is performed.

• Operation characteristics (example of reverse action)

For the duration of τ_W after the end of tuning cycle, the manipulated value (MV) is held at the output lower limit value (LLV) and the operation transitions to normal PID control.

τ_W can be determined by $(50+K_W)/100 \times (\tau-\tau_{on})$, and the wait setting parameter (K_W) can be set in parameter (S3)+28.

(Setting range K_W = -50 to 32717[%], Operating as τ_W=0 when an abnormal range is specified)



(SH_{PV}): PV input threshold value (hysteresis)

• Operation characteristics and three constants

Control mode	Proportional gain (K_P) [%]	Integral time (T_I) [×100ms]	Derivative time (T_D) [×10ms]
Proportional control (P operation) only	$\frac{1}{a}$ (ULV-LLV) × 100	—	—
PI control (PI operation)	$\frac{0.9}{a}$ (ULV-LLV) × 100	$33 \times \tau_{on}\left(1 - \frac{\tau_{on}}{\tau}\right)$	—
PID control (PID operation)	$\frac{1.2}{a}$ (ULV-LLV) × 100	$20 \times \tau_{on}\left(1 - \frac{\tau_{on}}{\tau}\right)$	$50 \times \tau_{on}\left(1 - \frac{\tau_{on}}{\tau}\right)$

Step response method

■Parameters set by auto tuning (limit cycle method)

- (S3)+1 (action setting (ACT)) bit 0 (action direction)
- (S3)+3 (proportional gain (K_P))
- (S3)+4 (integral time (T_I))
- (S3)+6 (derivative time (T_D))

■Auto tuning procedure

1. Transfer the output values for auto tuning to (D) (manipulated value (MV)).

Set the output value for auto tuning to the "maximum output allowable for the output equipment × 0.5 to 1".

2. Set the parameters (S3), (S1) (set value (SV)), which are not set by auto tuning, according to the system.

If the following precautions are not observed, auto tuning may not be performed correctly.

[Setting items]

Setting item		Remark
(S1)	Set value (SV)	The difference from the process value (PV) is 150 or greater(☞ [Precautions on setting]).
(S3)	Sampling time (T_S)	1000 (ms) or more (☞ [Precautions on setting])
(S3)+2	Input filter (α)	—
(S3)+5	Derivative gain (K_D)	When the input filter is set, the derivative gain should normally be set to 0.
Others		To be set as needed

[Precautions on setting]

- Difference between the set value (SV) and process value (PV)

To perform correct auto tuning, the difference between the process value and set value must be 150 or more at the beginning of auto tuning. If the difference is less than 150, set the SV for auto tuning.

After completion of auto tuning, set the SV back to the original value.

Setting item		Remark
(S1)	Set value (SV)	Set the value so that the difference from the set value at the beginning of auto tuning is 150 or more.

- Setting time of sampling time (T_S)

Set the sampling time for auto tuning to 1s (100ms) or more.

Set it also as is sufficiently larger than the output change period.

3. Setting bit 4 of (S3)+1 (action setting (ACT)) to ON starts auto tuning.

When the rate of change from the process value at the start of auto tuning to the set value is 1/3 or more, auto tuning is complete and bit 4 of (S3)+1 (action setting (ACT)) is set to OFF automatically.

Point

Start auto tuning when the system is stable. If not, auto tuning may not be performed correctly.

■Determining the three PID constants (step response method) [Reference]

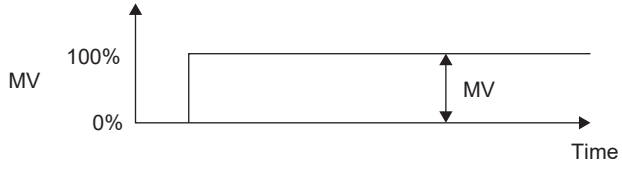
To obtain good control result of PID control, optimal values of individual constants (parameters) appropriate to the controlled system need to be determined. Here, the step response method is explained as a method of determining the optimal values of three PID constants (proportional gain (K_P), integral time (T_I), and derivative time (T_D)).

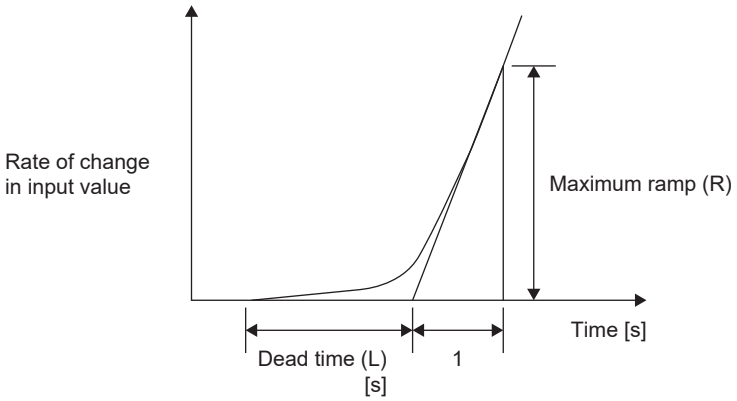
• Step response method

The step response method is a method of determining three PID constants from the operation characteristics (maximum ramp (R), dead time (L)) that are determined from changes in input caused by giving stepwise output from 0 to 100%*1 to the control system.

*1 Stepwise output can also be obtained in 0 → 75% or 0 → 50%.

• Operation characteristics





• Operation characteristics and three constants

Control mode	Proportional gain (K_P) [%]	Integral time (T_I) [×100ms]	Derivative time (T_D) [×10ms]
Proportional control (P operation) only	$\frac{1}{RL} \times MV \times 100$	—	—
PI control (PI operation)	$\frac{0.9}{RL} \times MV \times 100$	33L	—
PID control (PID operation)	$\frac{1.2}{RL} \times MV \times 100$	20L	50L

Precautions on auto tuning

• Programming measures in case of no change in the input value (PV)

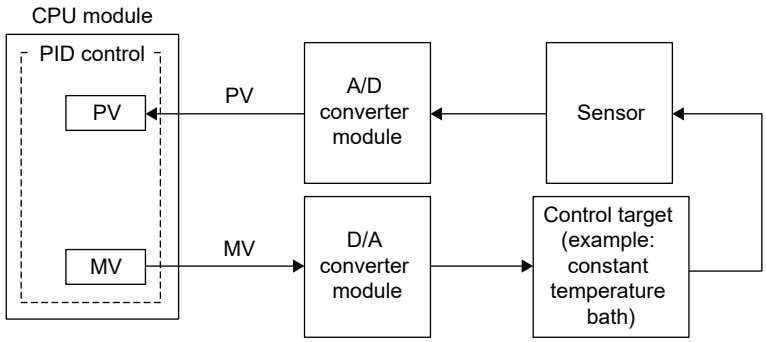
If the input value (PV) does not change normally due to a factor such as analog input disconnection, auto tuning does not end. Detect and avoid such problem by using a sequence to monitor the input value and the elapsed time from the start of tuning.

Example of practical program (step response method)

This section provides a sample program to implement auto tuning using the step response method in the following system.

System and operation example

■System configuration



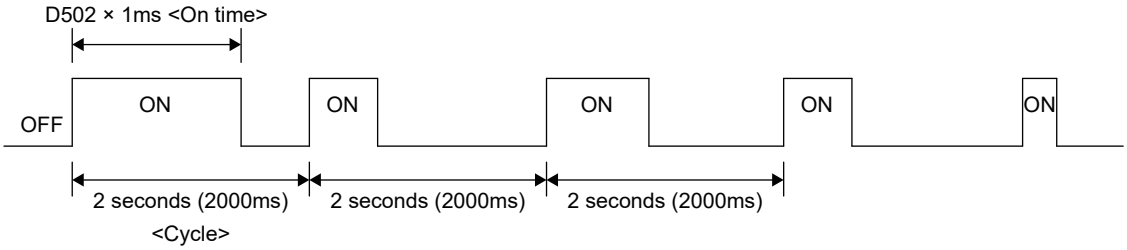
X2: Auto tuning command
X3: PID control command
Y20: Error display
Y21: Heater output

■Description

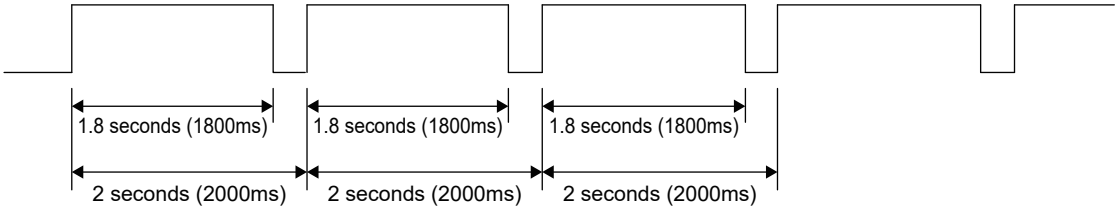
Setting item				During auto tuning	During PID control
Set value (SV)			(S1)	500(+50℃)	500(+50℃)
Parameters	Sampling time (T_S)		(S3)	3000 ms	500 ms
	Input filter (α)		(S3)+2	70%	70%
	Derivative gain (K_D)		(S3)+5	0%	0%
	Upper limit of output value		(S3)+22	2000 (2 seconds)	2000
	Lower limit of output value (LLV)		(S3)+23	0	0
	Action setting (ACT)	Alert for rate of change in input	(S3)+1(bit 1)	None	None
		Alert for rate of change in output	(S3)+1(bit 2)	None	None
		Settings of upper and lower limits of output value	(S3)+1(bit 5)	Yes	Yes
Manipulated value (MV)			(D)	1800	According to arithmetic operation

■Operation of output value

• PID control



• Auto tuning with maximum output 90%

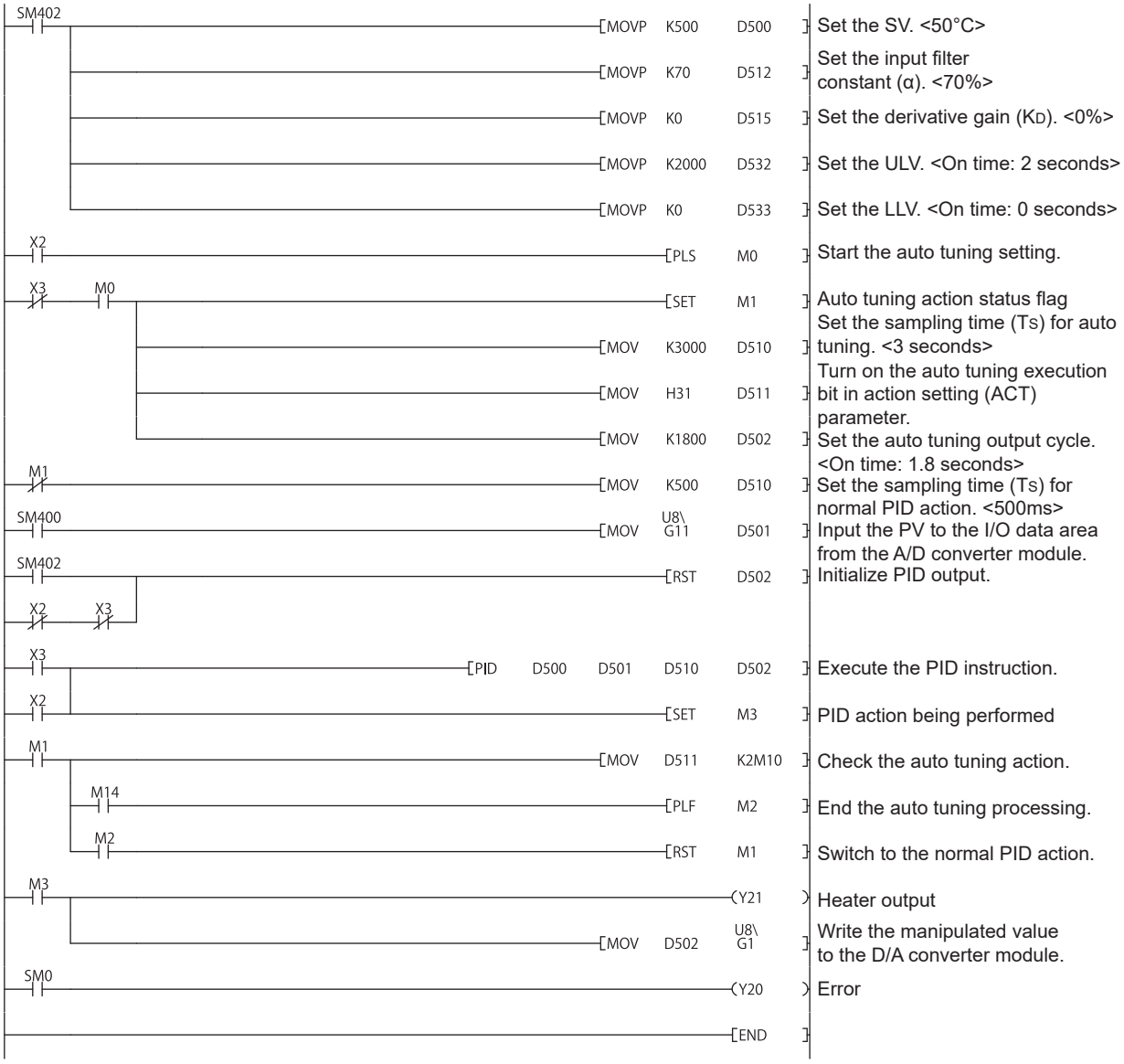


Program example in auto tuning (step response method) + PID control

• Device

Device	Description
X2	Start PID control after auto tuning.
X3	Start PID control (without auto tuning).
M0	Auto tuning setting flag
M1	Auto tuning operation flag
M2	Stop auto tuning.
M3	PID running
M14	Auto tuning operation flag

• Program

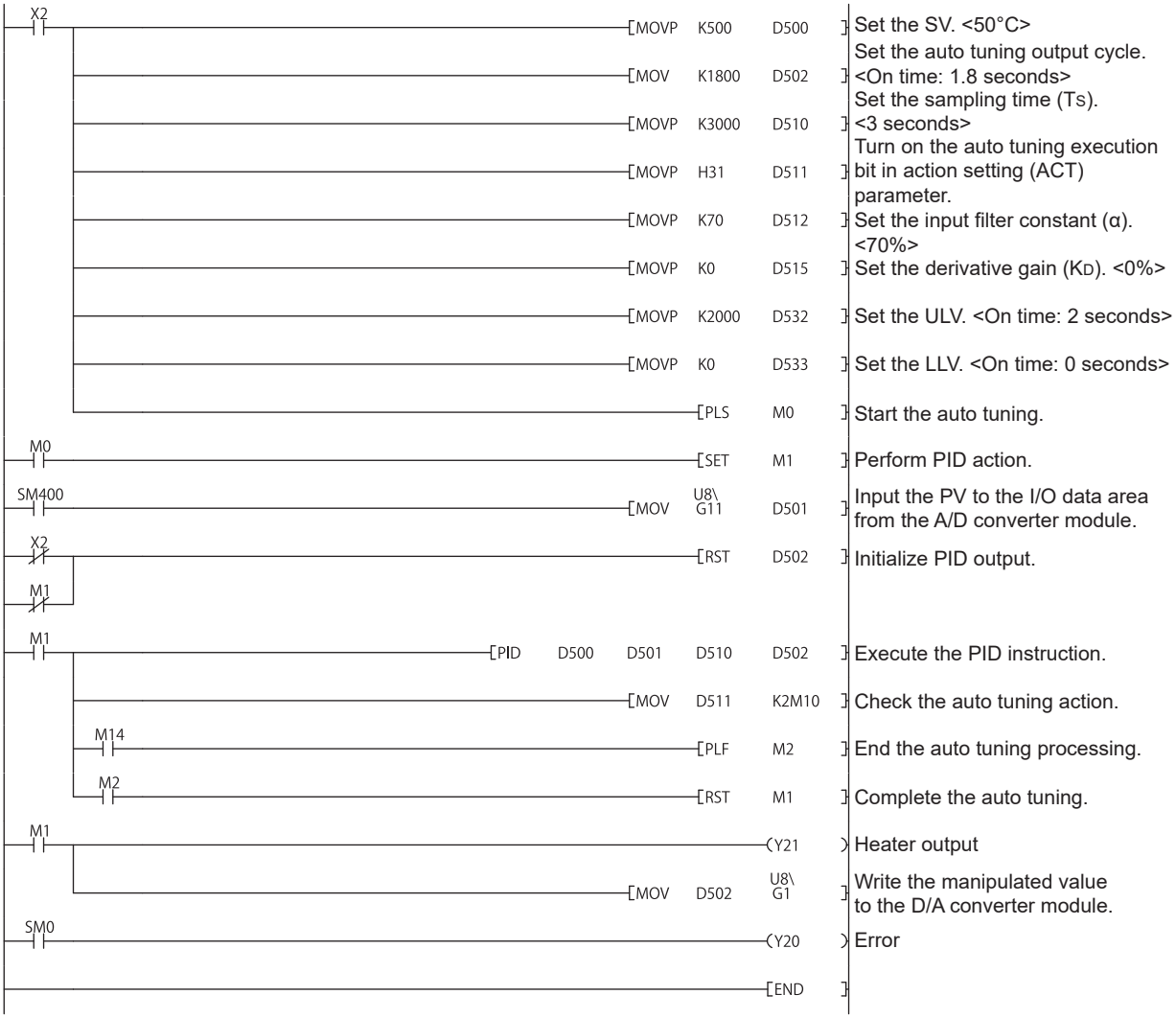


Program example in auto tuning (step response method) only

• Device

Device	Description
X2	Start auto tuning.
M0	Start auto tuning.
M1	PID operation
M2	Stop auto tuning.
M14	Auto tuning operation flag

• Program



Troubleshooting

Error code

If an error occurs in the set value of a control parameter or the data used for PID operation, the operation error flag (SM0) turns on and an error code is stored in SD0 according to the error content.
For the error code, refer to the User's Manual (Hardware Design, Maintenance and Inspection) of the CPU module used.

Precautions

For the PID process value (PV), normal measurement data needs to have been read before execution of PID operation. Especially when performing PID operation over the input value of analog input blocks, note the conversion time.

7.19 Other Instructions

Watchdog timer reset

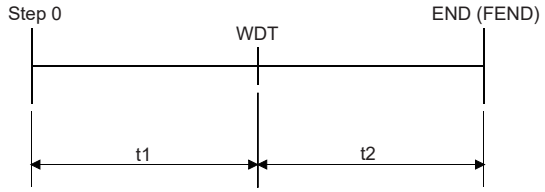
WDT(P)

Basic | High performance | Process | Redundant | Universal | LCPU

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant	Others
	Bit	Word		Bit	Word				
—	—								

Processing details

- Resets watchdog timer during the execution of a sequence program.
- Used in cases where the scan time exceeds the value set for the watchdog timer due to prevailing conditions. If the scan time exceeds the watchdog timer setting value on every scan, change the watchdog timer settings at the parameter settings of a programming tool.
- Make sure that the setting for t1 from step 0 to the WDT instruction and the setting for t2 from the WDT instruction to the END (FEND) instruction do not exceed the setting value of the watchdog timer.



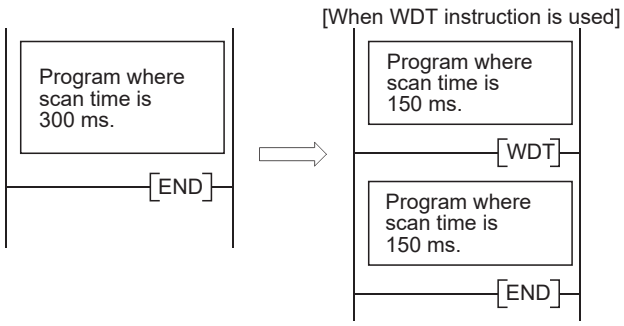
- The WDT instruction can be used two or more times during a single scan, but care should be taken in such cases, because of the time required until the output goes OFF during the generation of an error.
- Scan time values stored at the special register will not be cleared even if the WDT or WDTP instruction is executed. Accordingly, there are times when the value for the scan time for the special register is greater than the value of the watchdog timer set at the parameters.

Operation error

- There is no operation error in the WDT(P) instruction.

Program example

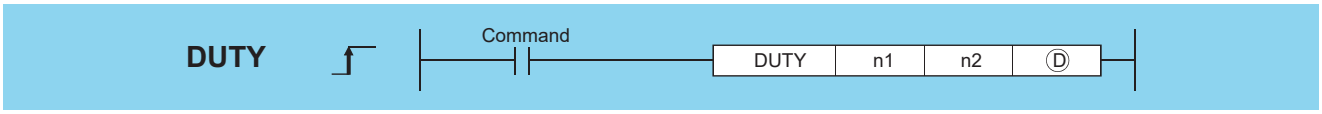
- The following program has a watchdog timer setting of 200ms, when due to the execution conditions program execution requires 300ms from step 0 to the END (FEND) instruction.



Timing pulse generation

DUTY

Basic | High performance | Process | Redundant | Universal | LCPU



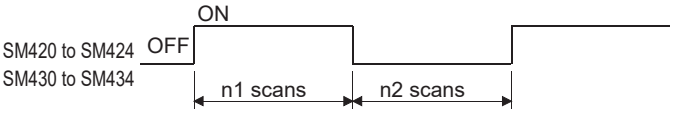
n1: Number of scans for ON (BIN 16 bits)
n2: Number of scans for OFF (BIN 16 bits)
(D): User timing clock (SM420 to SM424, SM430 to SM434) (bits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
n1	○	○							—
n2	○	○							—
(D)	○*1	—							—

*1 Only SM420 to SM424, SM430 to SM434 can be used.

Processing details

- Turns the user timing clock (SM420 to SM424, SM430 to M434), designated by (D), ON for the duration equivalent to the number of scans specified by n1, and OFF for the duration equivalent to the number of scans specified by n2.



- Scan execution type programs use SM420 to SM424, and low speed execution type programs use SM430 to SM434.
- The following will take place if both n1 and n2 have been set for 0:
  - n1=0, n2≥0: SM420 to SM424 and SM430 to SM434 will stay OFF.
  - n1>0, n2=0: SM420 to SM424 and SM430 to SM434 will stay ON.
- The data designated by n1, n2, and (D) is registered with the system when the DUTY instruction is executed, and the timing pulse is turned ON and OFF by END processing.

Operation error

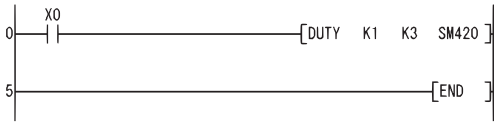
- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4100	The values of n1 and n2 are less than 0.	○	○	○	○	○	○
4101	The device specified in (D) is not from SM420 to SM424 or SM430 to SM434.	○	○	○	○	○	○

Program example

• The following program turns SM420 ON for 1 scan, and OFF for 3 scans if X0 is ON.

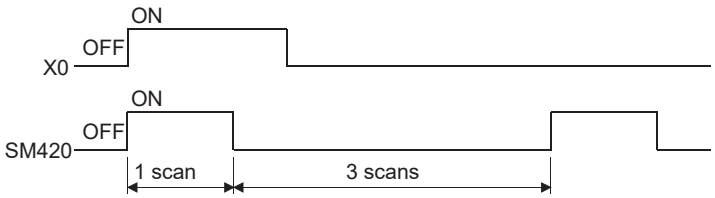
[Ladder Mode]



[List Mode]

Step	Instruction	Device		
0	LD	X0		
1	DUTY	K1	K3	SM420
5	END			

[Operation]



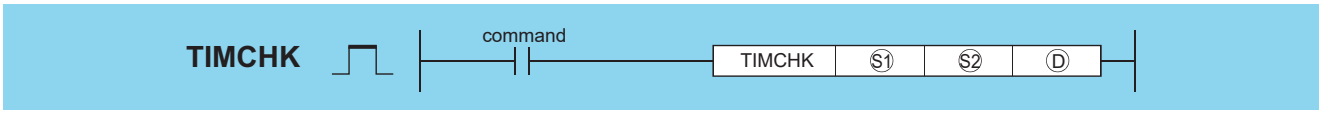
7

Time check

TIMCHK

Ver. Basic | High performance | Process | Redundant | Universal | LCPU

• Basic model QCPU: The serial number (first five digits) is "04122" or later.



(S1): Device where the measured current value will be stored (unit: 100ms) (BIN 16 bits)
(S2): The set value for measurement or the device where the set value is stored (unit: 100ms) (Setting range: 0 ≤ s2 ≤ 32767) (BIN 16 bits)
(D): Device to be turned ON at time-out (bits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(S1)	—	○		—					—
(S2)	○	○		○					—
(D)	○	○ (Other than T, ST, C)		—					—

Processing details

- Measures the ON time of the device used as a condition, and turns ON the device specified by (S2) if the condition device remains ON for longer than the time set to the device specified by (D).
- The current value of the device specified by (S1) is cleared to 0 and the device specified by (D) is turned OFF at the rising edge of the execution command. The current value of the device designated by (S1) and the ON status of the device designated by (D) are retained after the execution command turns OFF.
- The measured current value is stored in units of 100ms. Set the value for measurement in units of 100ms.
- When a value other than 0 to 32767 is specified in (S2), (D) turns on at the scan after the scan in which the execution command turns on.

Operation error

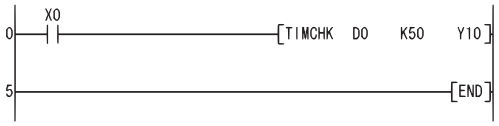
- In any of the following cases, an operation error occurs, the error flag (SM0) turns on, and an error code is stored in SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4100	The device that cannot be specified has been specified.	—	—	—	—	○	○

Program example

- Program where the ON time of X0 is set to 5s, the current value storage device to D0, and the device that will turn ON at time-out to Y10.

[Ladder Mode]



[List Mode]

Step	Instruction	Device		
0	LD	X0		
1	TIMCHK	D0	K50	Y10
5	END			

Direct 1-byte read from file register

ZRRDB(P)

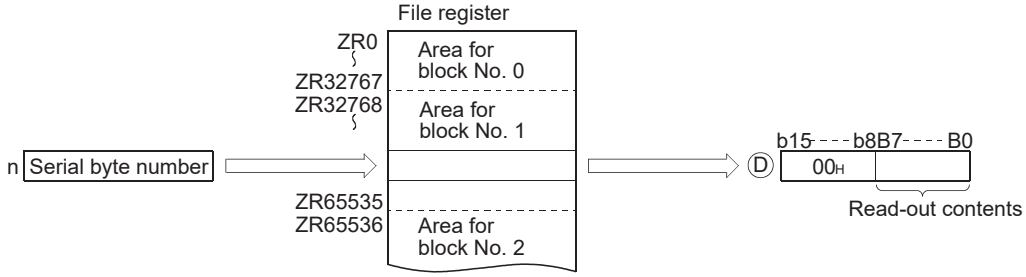
Basic | High performance | Process | Redundant | Universal | LCPU

n: Serial byte number for the file register to be read (BIN 32 bits)
(D): Number of the device where the read data will be stored (BIN 16 bits)

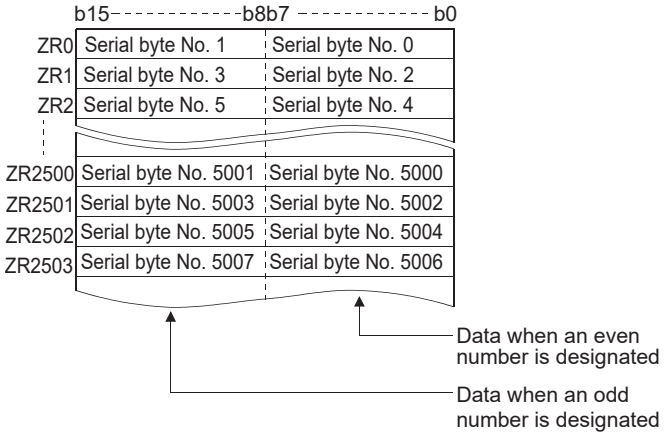
Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
n	○							○	—
(D)	○							—	—

Processing details

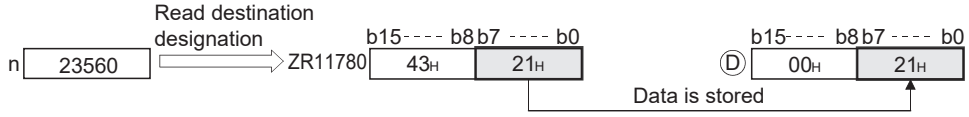
- Reads the serial byte number designated by n that does not signify a block number, and stores at the lower 8 bits of the device designated by (D). The upper 8 bits designated by (D) will become 00H.



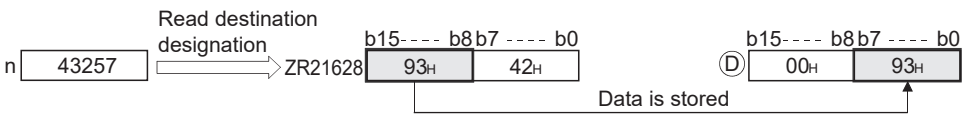
- The correspondence between file register numbers and serial byte numbers is as indicated below:



• If the value of n has been designated as 23560, the data at the lower 8 bits of ZR11780 will be read.



• If the value of n has been designated as 43257, the data at the upper 8 bits of ZR21628 will be read.



Operation error

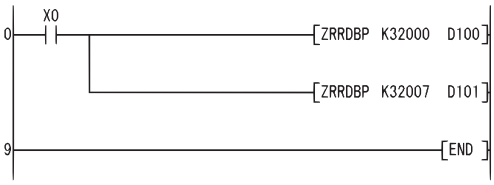
- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4101	The specified device number (serial byte number) exceeds the available range.	—	—	—	—	○	○

Program example

- The following program reads the lower 8 bits of ZR16000 and the upper 8 bits of ZR16003, and stores them at D100 and D101 when X0 is ON.

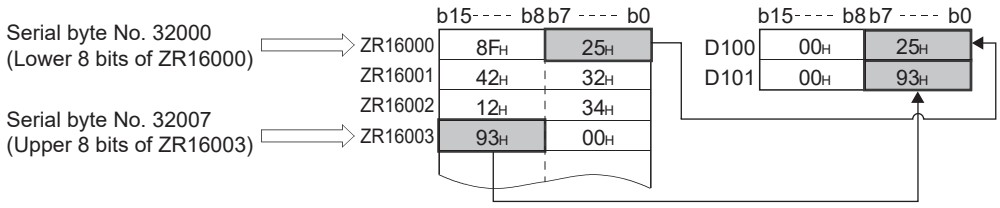
[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	X0
1	ZRRDBP	K32000 D100
5	ZRRDBP	K32007 D101
9	END	

[Operation]



File register direct 1-byte write

ZRWRB(P)

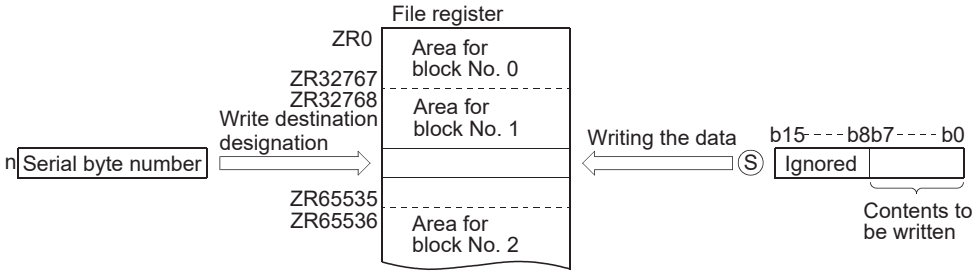
Basic | High performance | Process | Redundant | Universal | LCPU

n: Serial byte number for the file register to be written (BIN 32 bits)
(S): Number of the device where the data to be written is stored (BIN 16 bits)

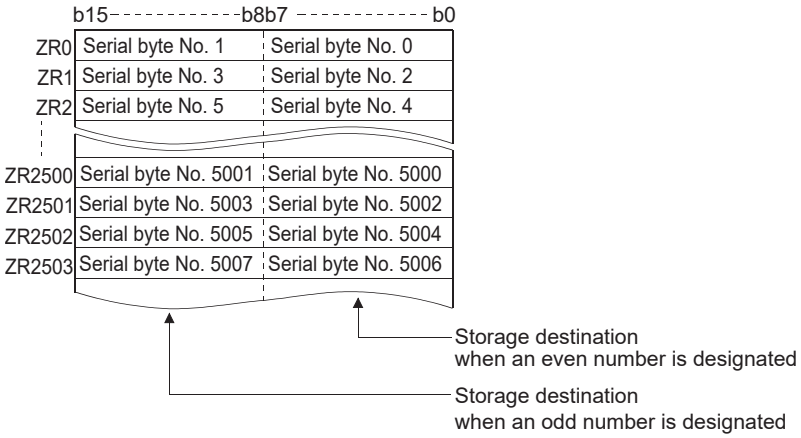
Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
n	○								—
(S)	○								—

Processing details

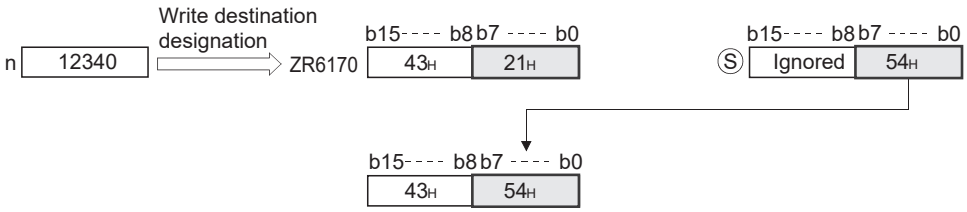
- Writes the lower 8 bits of data stored in the device designated by (S) that does not signify a block number to the file register of the serial byte number designated by n. The upper 8 bits of data in the device designated by are ignored (S).



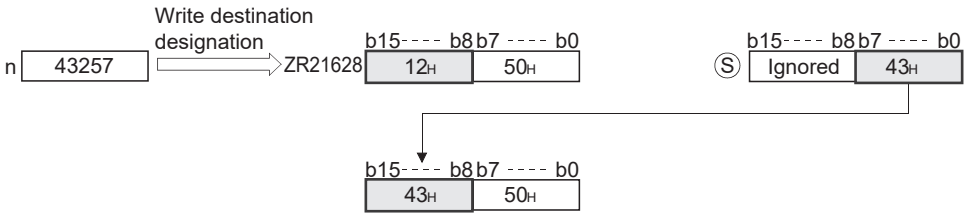
- The correspondence between file register numbers and serial byte numbers is as indicated below:



• If n=12340 is specified, the data will be written to the lower 8 bits of ZR6170.



• If n=43257 is specified, the data will be written to the upper 8 bits of ZR21628.



Operation error

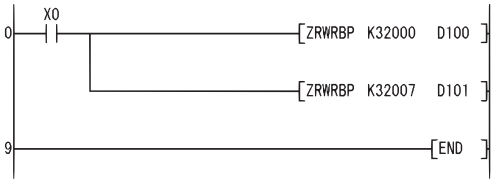
• In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4101	The specified device number (serial byte number) exceeds the available range.	—	—	—	—	○	○

Program example

• The following program writes the data at the lower bits of D100 and D101 to the lower 8 bits of ZR16000 and the upper 8 bits of ZR16003 when X0 is turned ON.

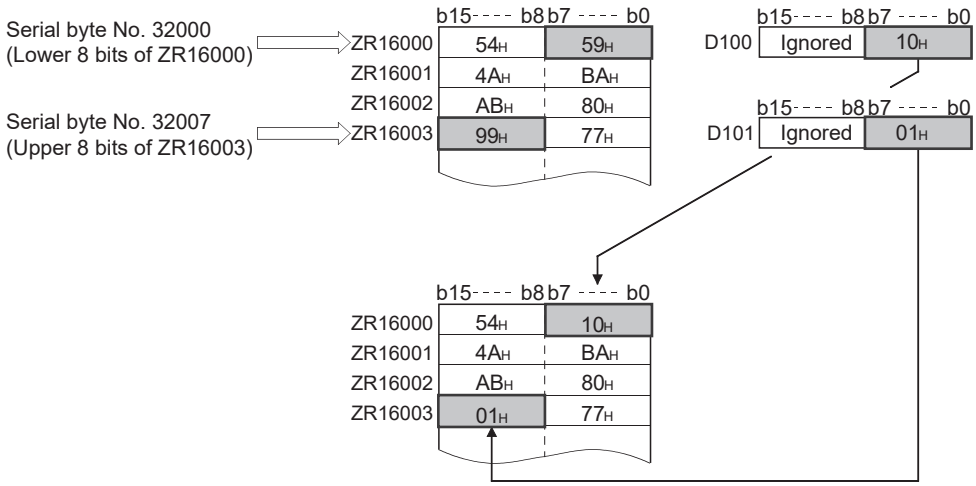
[Ladder Mode]



[List Mode]

Step	Instruction	Device	
0	LD	X0	
1	ZRWRBP	K32000	D100
5	ZRWRBP	K32007	D101
9	END		

[Operation]



Indirect address read operations

ADRSET(P)

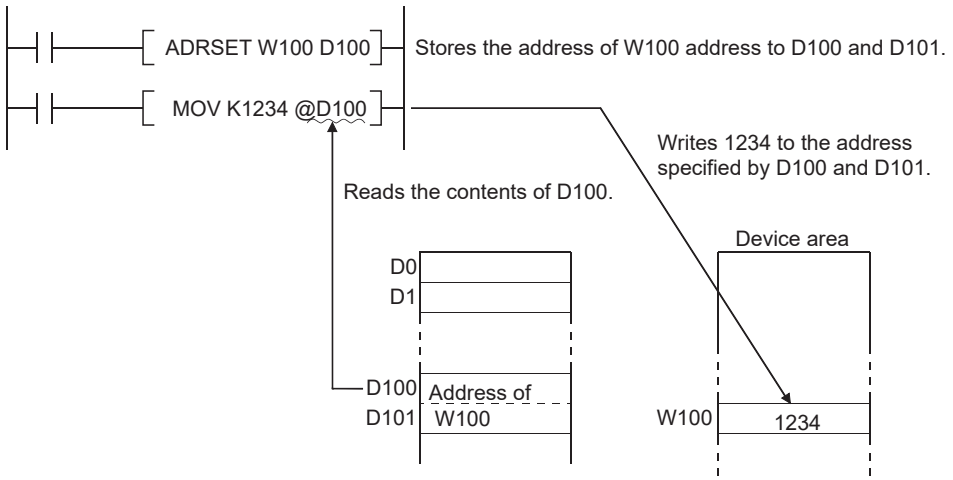
Basic | High performance | Process | Redundant | Universal | LCPU

(S): Number of the device whose indirect address is read out (Device name)
(D): Head number of the device where the indirect address of the device designated by (S) will be stored (BIN 32 bits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant	Others
	Bit	Word		Bit	Word				
(S)	○			—					
(D)	○			—					

Processing details

- Stores the indirect address of the device designated by (S) at (D) and (D)+1. The address stored at the device designated by (D) is used when an indirect device address is performed by the sequence program.



- A bit device designation cannot be made at (S).

Point

See Page 106 Indirect Specification for further information on indirect designations.

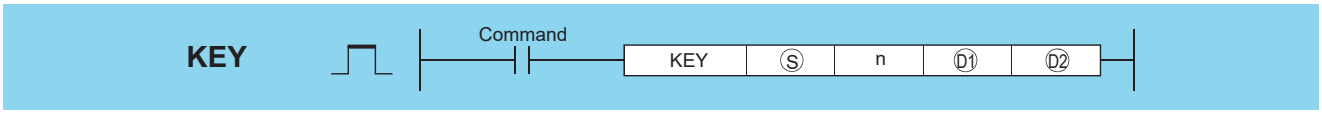
Operation error

- There is no operation error in the ADRSET(P) instruction.

Numerical key input using keyboard

KEY

Basic | High performance | Process | Redundant | Universal | LCPU

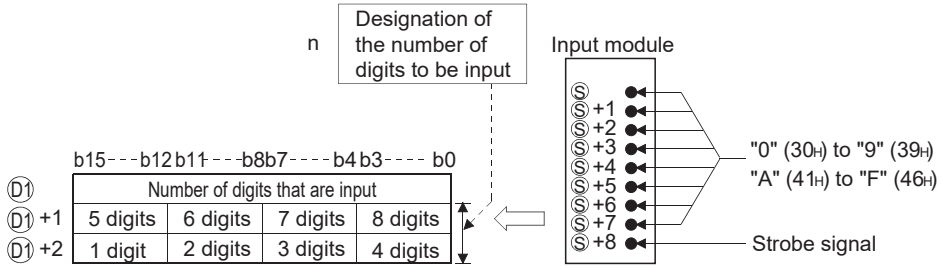


(S): Head number of the devices (X) to which a numeral will be input (bits)
n: Number of digits of the numeral to be input (BIN 16 bits)
(D1): Head number of the devices where the input numeral will be stored (BIN 16 bits)
(D2): Number of the bit device to turn ON at the completion of input (bits)

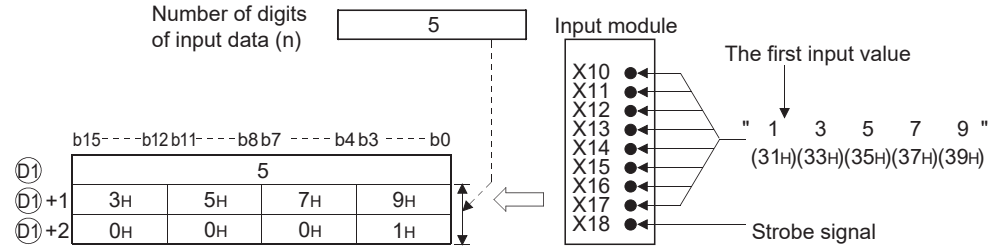
Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(S)	○ (Only X)	—		—			—		—
n	○	○		○			○		—
(D1)	—	○		—			—		—
(D2)	○	○		○			—		—

Processing details

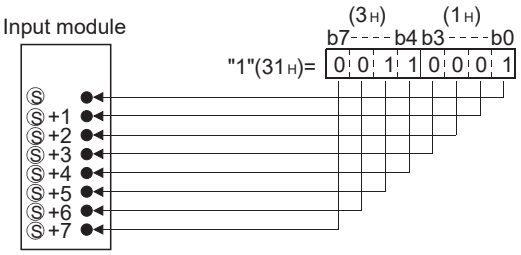
- Fetches ASCII data from the 8 points of input (X) designated by (S), converts it to hexadecimal values and stores the result in the area starting from the device designated by (D1).



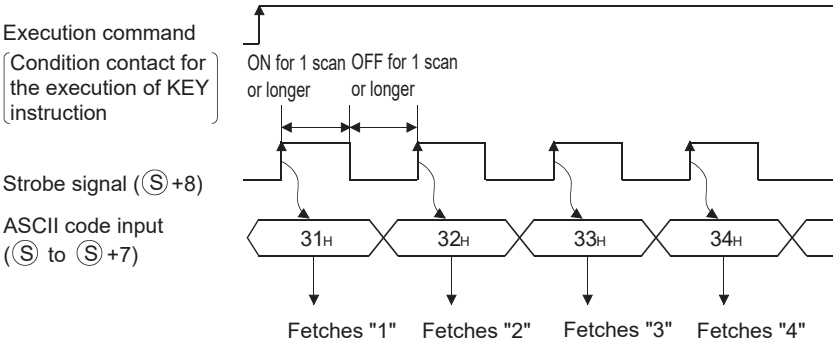
For example, in a case where the number of digits (n) of input data has been set at 5, and the values "31H", "33H", "35H", "37H" and "39H" have been input through X10 to X18 of the input module, the following will take place:



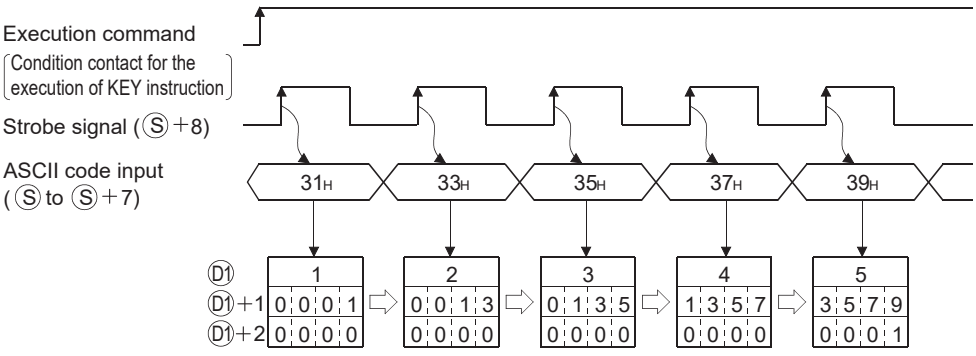
- Numerical input to input (X) designated by (S) undergoes bit development at (S) through (S)+7 and is input as the ASCII code corresponding to the numbers. ASCII code which can be input is from 30H (0) to 39H (9), and from 41H (A) to 46H (F).



- After ASCII code is input to (S) to (S)+7, the strobe signal at (S)+8 goes ON to incorporate the designated numbers internally. The strobe signal should be held at its ON or OFF status for more than one scan of the sequence program. If this time is less than 1 scan, there will be cases when the data is correctly incorporated.



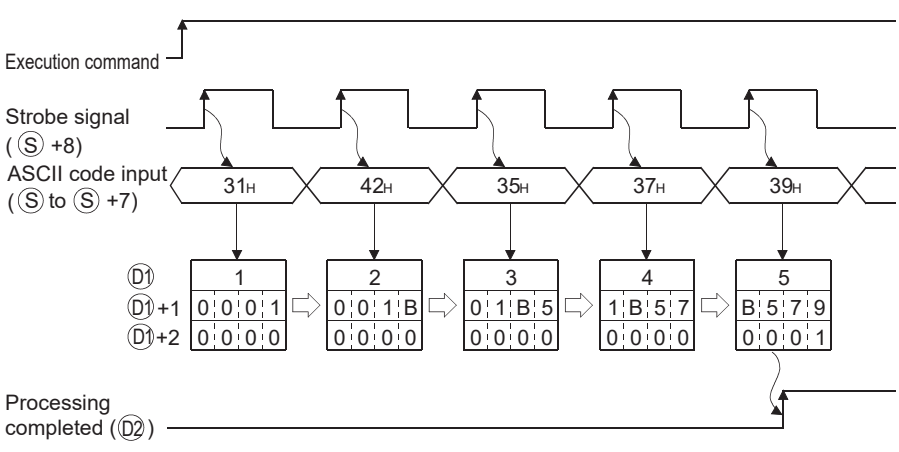
- Be sure to keep the execution command (condition contact for the KEY instruction) ON until the specified number of digits has been input. The KEY instruction cannot be executed if the execution command turns OFF.
- The digits for the numbers actually fetched to (D1) will be stored at the device designated by (D1), and these will be converted to the ASCII codes input at (D1)+1 and (D1)+2, converted to hexadecimal BIN values, and stored.



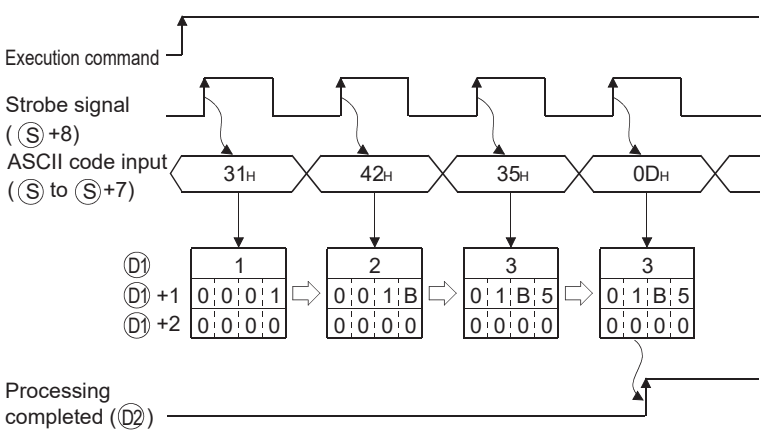
- The number of digits that can be designated by n is from 1 to 8.
- Fetching of the input data is completed when any of the inputs shown below has been made. At the completion, the bit device designated by (D2) is turned ON.
  - When the number of digits specified by n has been input
  - When the "0DH" code has been input

For example, the operations at the location designated if n = 5 will be as indicated below:

When the designated number of digits are input



When 0DH code is input



If input processing is to be performed a second time, it is necessary to clear the number of digits input and the input data stored at (D1), and turn OFF the designated device at the user program.

If (D1) is not cleared and (D2) not turned OFF, the next input processing cannot be performed.

Operation error

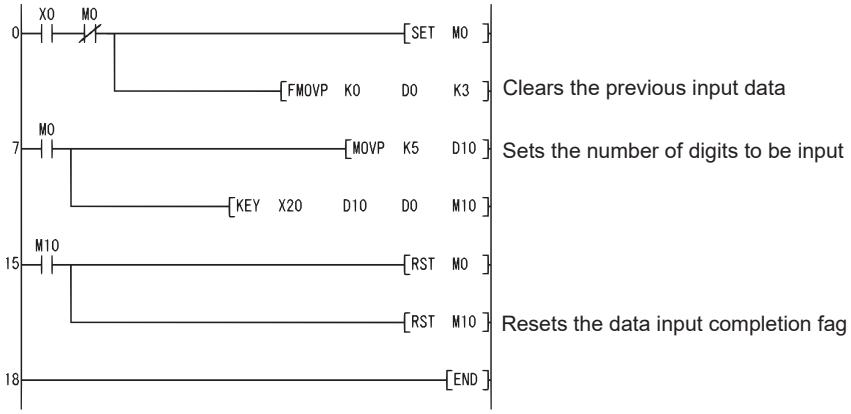
- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4100	The device specified in (S) is not an input (X) device. The number of digits specified in n is outside the range from 1 to 8.	—	○	○	—	—	—

Program example

- The following program fetches data of the 5 or fewer digits from the numerical keypad connected to X20 to X28, and stores it to the area from D0 to D2 when X0 is turned ON.

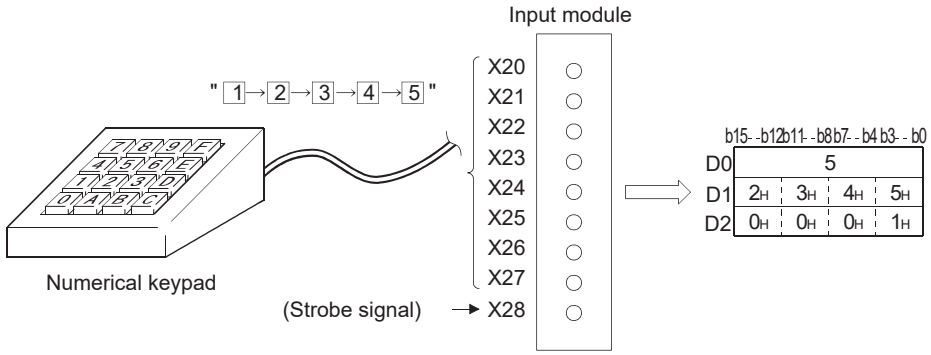
[Ladder Mode]



[List Mode]

Step	Instruction	Device			
0	LD	X0			
1	ANI	M0			
2	SET	M0			
3	FMOVP	K0	D0	K3	
7	LD	M0			
8	MOVP	K5	D10		
10	KEY	X20	D10	D0	M10
15	LD	M10			
16	RST	M0			
17	RST	M10			
18	END				

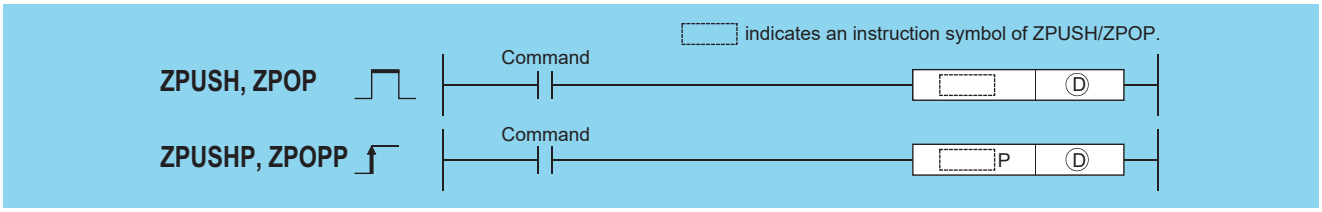
[Operation]



Batch save of index register, batch recovery of index register

ZPUSH(P), ZPOP(P)

Basic | High performance | Process | Redundant | Universal | LCPU



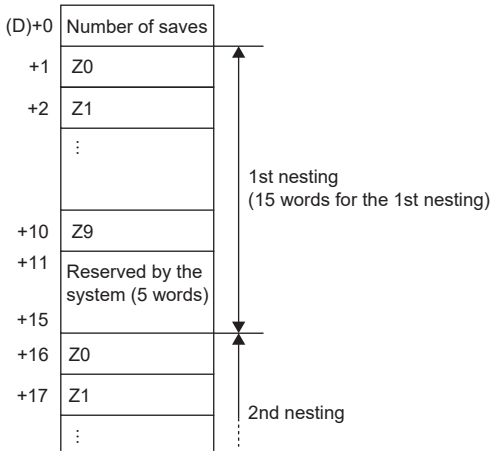
(D): Head number of the devices to/from which contents of an index register are saved/recovered (BIN 16 bits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant	Others
	Bit	Word		Bit	Word				
(D)	—	○		—					

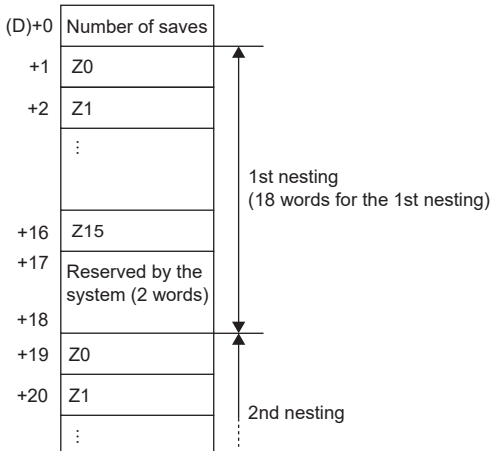
Processing details

■ZPUSH

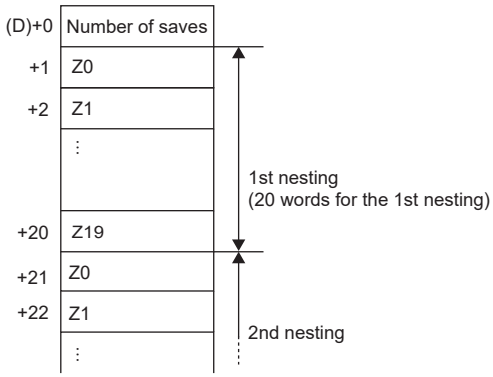
- Saves the contents of the following index registers to after the device specified by (D). (When contents of an index register are saved, (D)+0 (the number of saves made) is increased by 1.)
  - Basic model QCPU: Z0 to Z9
  - High Performance model QCPU/Process CPU/Redundant CPU: Z0 to Z15
  - Universal model QCPU/LCPU: Z0 to Z19
- The ZPOP instruction is used for data recovery. Nesting is possible within the ZPUSH to ZPOP cycle.
- If nesting has been done, each time the ZPUSH instruction is executed, the field used following (D) will be added to, so a field large enough to accommodate the number of times the instruction will be used should be maintained from the beginning.
- The composition of the field used following (D) is as shown below:
  - When Basic model QCPU is used



• When using a High Performance model QCPU/Process CPU/Redundant CPU



• When using Universal model QCPU/LCPU



■ZPOP

• Recovers the contents saved in the area starting from the device designated by (D) to the index register. (When the saved content is read out to the index register, (D)+0 (the number of saves made) is decreased by 1.)

Operation error

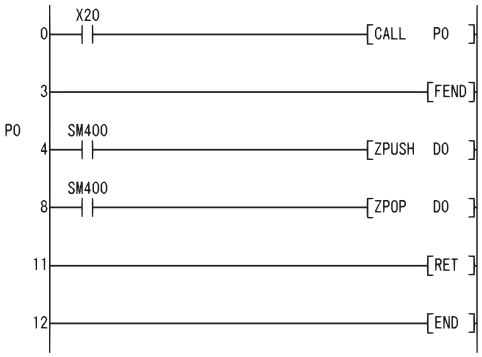
• In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4100	The operation result of (D)+0 (the number of saves made) is 0 in the ZPOP(P) instruction.	○	○	○	○	○	○
4101	For the ZPUSH(P) instruction, the range of the device specified by (D), exceeds the range of the corresponding device.	○	○	○	○	○	○

Program example

- The following program saves the contents of the index register to the fields following D0 before calling the subroutine following P0 that uses the index register.

[Ladder Mode]



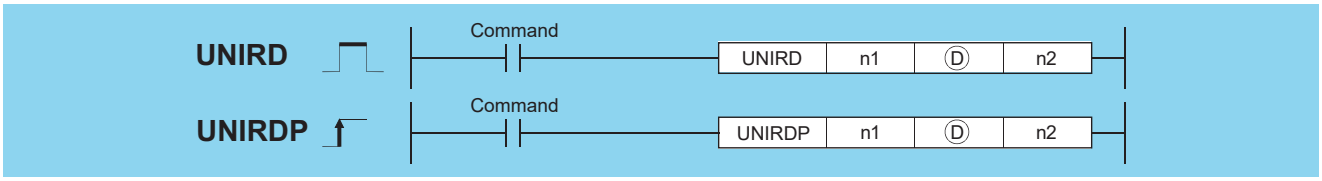
[List Mode]

Step	Instruction	Device
0	LD	X20
1	CALL	P0
3	FEND	
4		P0
5	LD	SM400
6	ZPUSH	D0
8	LD	SM400
9	ZPOP	D0
11	RET	
12	END	

Reading module information

UNIRD(P)

Basic | High performance | Process | Redundant | Universal | LCPU



n1: Value (0 to FFn) which the start I/O number of the module information read source is divided by 16 (BIN 16 bits)
(D): Head number of the devices where the module information will be stored (device name)
n2: The number of points of read data (0 to 256) (BIN 16 bits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
n1	○	○		—				○	—
(D)	—	○		—				—	—
n2	○	○		—				○	—

Processing details

- Reads the module information as much as designated by n2 from the module designated by n1, and stores that information into the area starting from the device designated by (D). (Reads the status of the actually installed modules even if the module type and the number of points are changed by I/O assignment.)
- The details of the module information are described as follows:

Bit: b15 b14 b13 b12 b11 b10 b9 b8 b7 b6 b5 b4 b3 b2 b1 b0
Individual module information

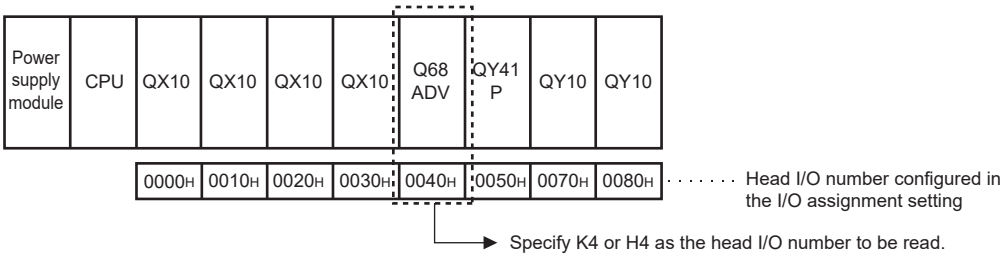
Bit	Item	Description	
		QCPU	LCPU
b0 b1 b2	Number of I/O points	000: 16 points 001: 32 points 010: 48 points 011: 64 points 100: 128 points 101: 256 points 110: 512 points 111: 1024 points	
b3 b4 b5	Module type	000: Input module 001: Output module 010: I/O combined module 011: Intelligent function module	000: Input module 001: Output module 010: I/O combined module 011: Intelligent function module 111: CPU Built-in I/O
b6	External supply power status (For future expansion)	1: External supply power is connected. 0: External supply power is not connected.	Fixed to 0
b7	Presence/absence of fuse blown	1: Some modules have fuse blown. 0: Normal	Fixed to 0
b8	Online module replacement status/execution from the standby system	1: Module information on the extension base unit is tried to be read during online module change or from the CPU module of standby system in the redundant system.*1 0: Other than above	Fixed to 0
b9	Minor/medium error status	1: Minor/medium error occurred 0: Normal	
b10 b11	Module error status	00: No module error 01: Minor error 10: Medium error 11: Serious error	

Bit	Item	Description	
		QCPU	LCPU
b12	Module ready status	1: Normal 0: Module error occurred	
b13	Empty	Fixed to 0	
b14	Module type	1: A series module 0: Q series module/L series module	Fixed to 0
b15	Module installation status	1: Modules are installed. 0: No modules are installed.	

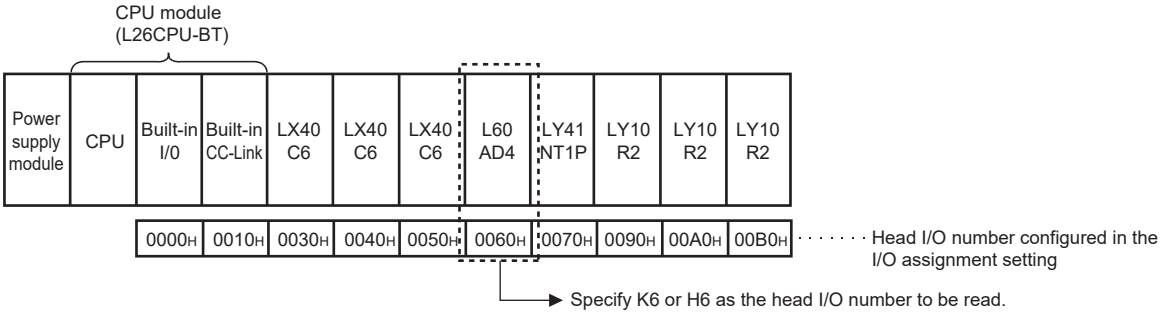
*1 The Universal model QCPU used in the multiple CPU system is turned ON during the online module change of the module controlled by the other CPU.

- The value of n1 is specified by the first 3 digits of the hexadecimal 4 digits that represent the head I/O number of the module from which the module information is read.

[QCPU]



[LCPU]



Operation error

- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

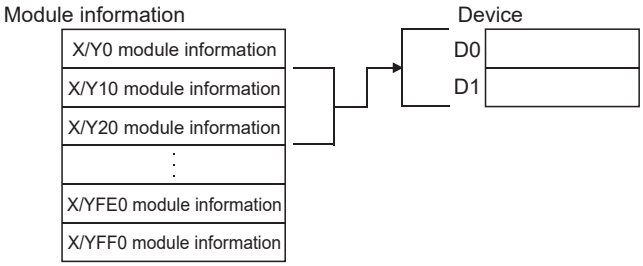
Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4100	n1 is a value other than 0 to FFH. n2 is a value other than 0 to 256. The total of n1 and n2 is equal to or greater than 257.	—	○	○	○	○	○*2
	n1 is a value other than 0 to 3FH. n2 is a value other than 0 to 64. The total of n1 and n2 is equal to or greater than 65.	Q00/ Q01	—	—	—	—	○*3
	n1 is a value other than 0 to FH. n2 is a value other than 0 to 16. The total of n1 and n2 is equal to or greater than 17.	Q00J	—	—	—	—	—
4101	The range of the device specified by (D) exceeds the range from (D) to (D)+n2 (including (D)).	○	○	○	○	○	○

*2 For only L06CPU, L06CPU-P, L26CPU, L26CPU-P, L26CPU-BT, and L26CPU-PBT

*3 For only L02SCPU, L02SCPU-P, L02CPU, and L02CPU-P

Program example

- The following program stores the module information at I/O numbers 10H and 20H into the devices starting from D0 when X0 is turned ON.



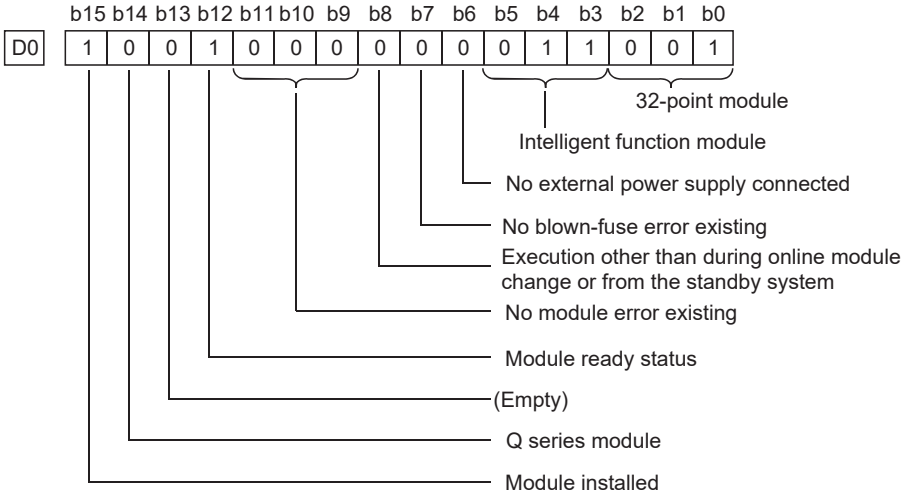
[Ladder Mode]

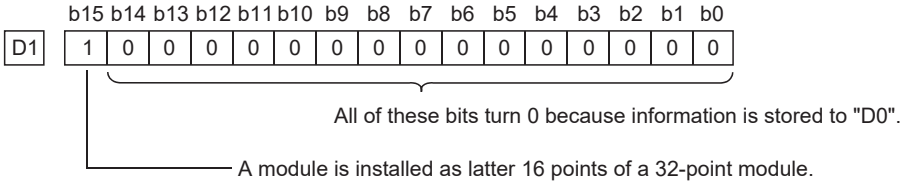
[List Mode]

Step	Instruction	Device		
0	LD	X0		
1	UNIRDP	H1	D0	K2
5	END			

Readout result (When read to D0)

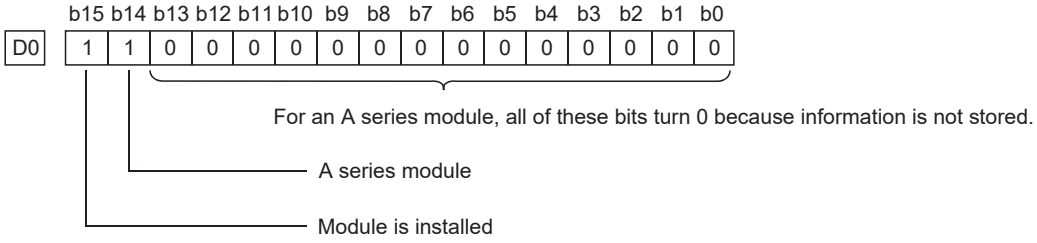
- 32-point intelligent function module for Q series

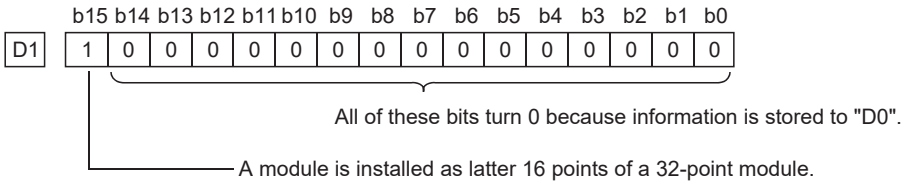




With a 48- or 64-point module, the same contents as those of D1 are stored in D2 or D2 and D3 respectively.

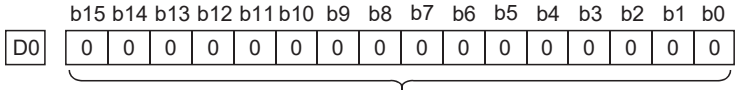
• 32-point module for A series



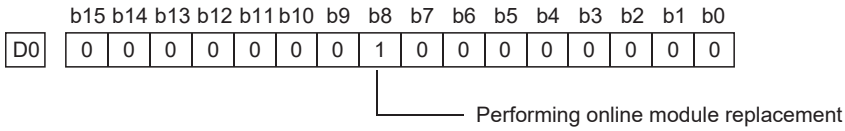


With a 48- or 64-point module, the same contents as those of D1 are stored in D2 or D2 and D3 respectively.

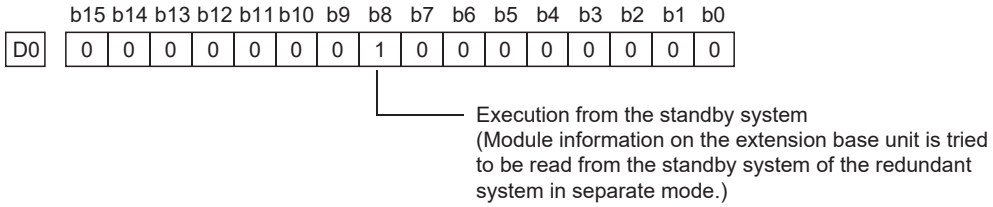
• Empty slot



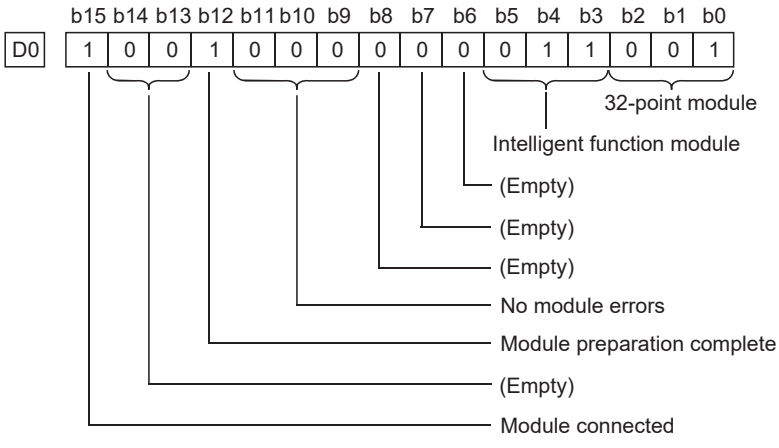
• Performing online module replacement

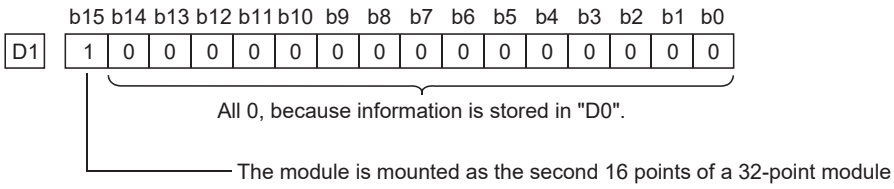


• Module information on the extension base unit is tried to be read from the standby system of the redundant system in separate mode.



• L series 32-point intelligent function module





Reading module model name

TYPERD(P)

Basic | High performance | Process | Redundant | Ver. Universal | LCPU

• Universal model QCPU: The serial number (first five digits) is "11043" or later.

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
n	○	○		—				○	—
(D)	—	○		—				—	—

Setting data

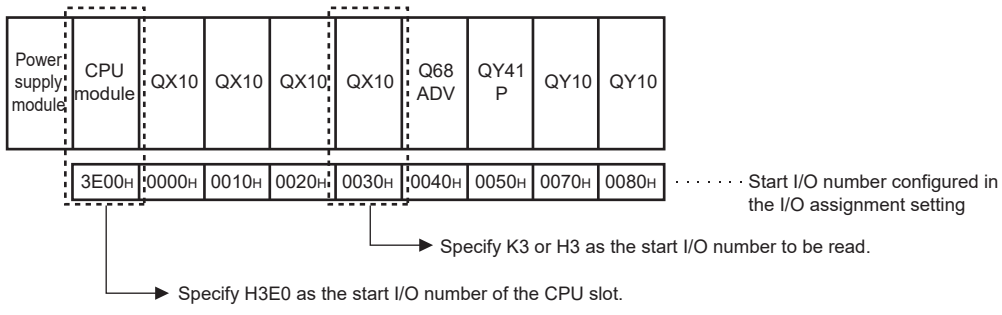
Setting data	Description		Setting range	Set by	Data type
n	Value obtained by dividing the start I/O number of a module whose model name is to be read by 16		0 to FFH, 3E0 to 3E3H (Universal model QCPU) 0 to FFH, 3E0(LCPU)	User	BIN 16 bits
(D)	(D)+0	Execution result of the instruction	Within each device range	System	BIN 16 bits
	(D)+1 to (D)+9	Module model name			Character string

Processing details

- This instruction reads the module information stored in the area starting from the I/O number specified by "n", and stores it in the area starting from the device specified by (D).
- For the Universal model QCPU, the following six modules (only Q series) are supported.
  - CPU module
  - Input module
  - Output module
  - I/O combined module
  - Intelligent function module
  - GOT (bus connection)
- For the LCPU, the following four models are supported.
  - CPU module
  - Input module
  - Output module
  - Intelligent function module
- The value of n is specified by the first 3 digits of the hexadecimal 4 digits that represent the start I/O number of a module whose model name is to be read.

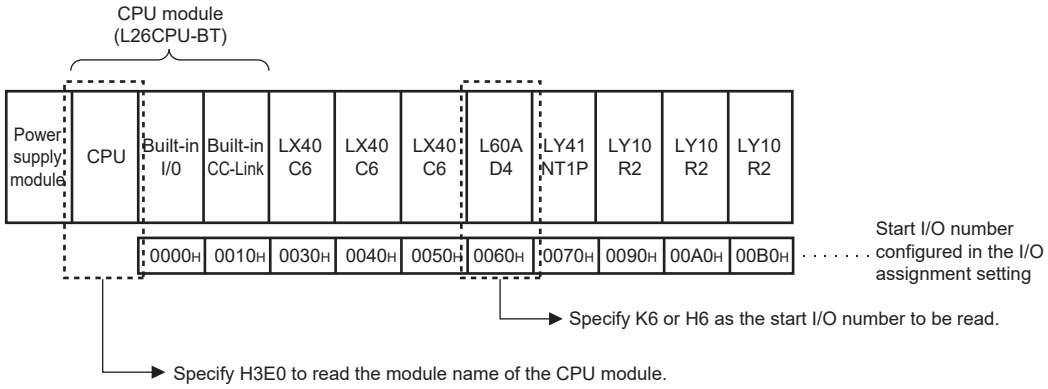
When the target module occupies one slot

Universal model QCPU



When the target module occupies one slot

LCPU (L26CPU-BT)



Point

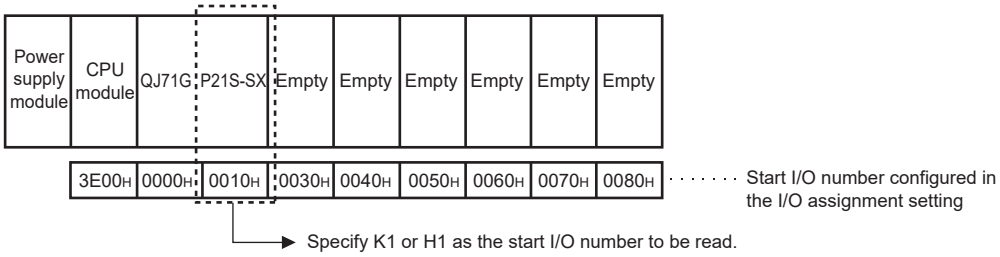
On the LCPU, if the built-in I/O or first I/O on the built-in CC-Link is specified, then the model name of the CPU module is read.

When the target module occupies two slots

The start I/O number to be specified may differ from that of the mounted module.
For the start I/O number, refer to the manual of each module.
The value of n is specified by the first 3 digits of the hexadecimal 4 digits that represent the start I/O number of a module whose model name is to be read.

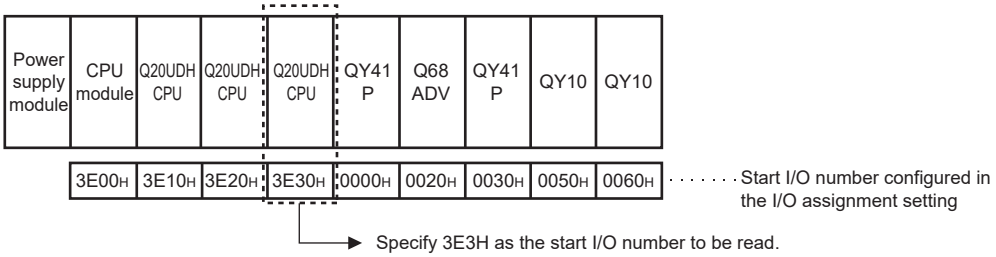
QJ71GP21S-SX

Specify a value that is the sum of the start I/O number of the mounted module and 0010H.



When the target module is a CPU module in multiple CPU systems

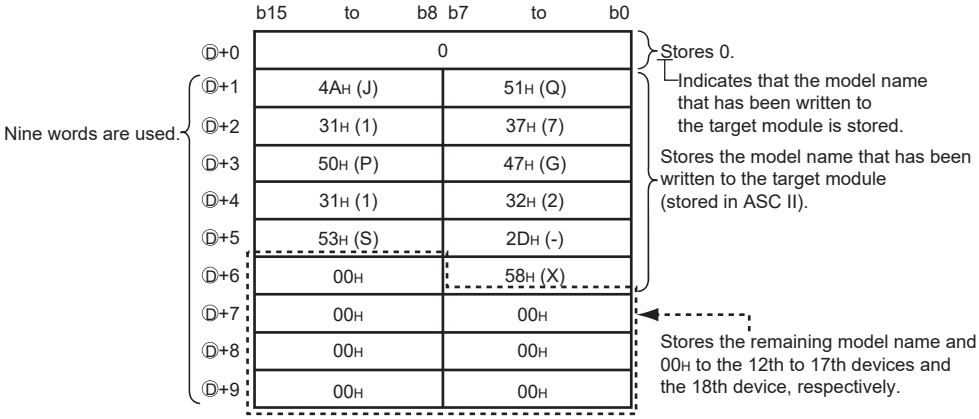
Specify the first 3 digits of the hexadecimal 4 digits which represent the start I/O number of each CPU module.



Or, the model name can be read by specifying the start I/O number of a module controlled by another CPU.

• (D)+0 and (D)+1 to (D)+9 store the execution result of the instruction and module model name, respectively. A value stored in (D) is as follows:

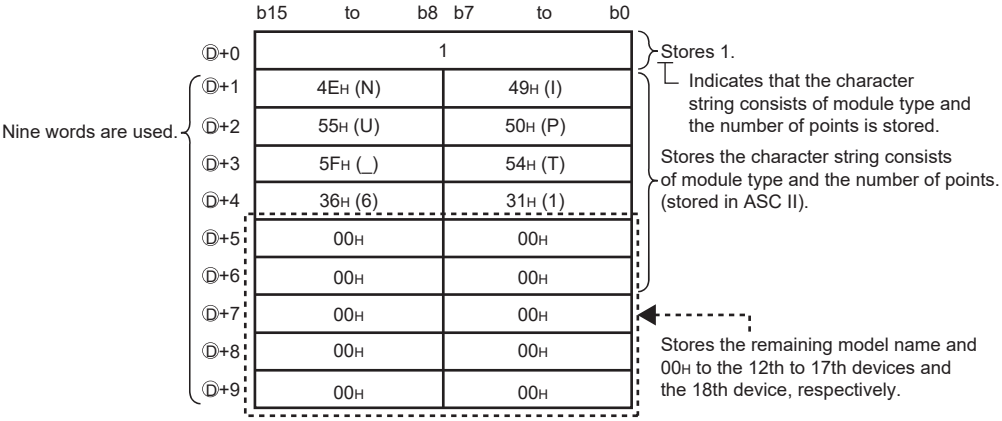
■When the model name has been written to the target module (example: QJ71GP21-SX)



The following table shows the examples of model names stored in (D)+1 to (D)+9.

Target module	Stored model name
CPU module	Q06UDEHCPU
Intelligent function module	QJ71GP21-SX
GOT	GOT1000

■When the model name has not been written to the target module (example: QX40)



The following table shows the examples of character strings stored in (D)+1 to (D)+9.

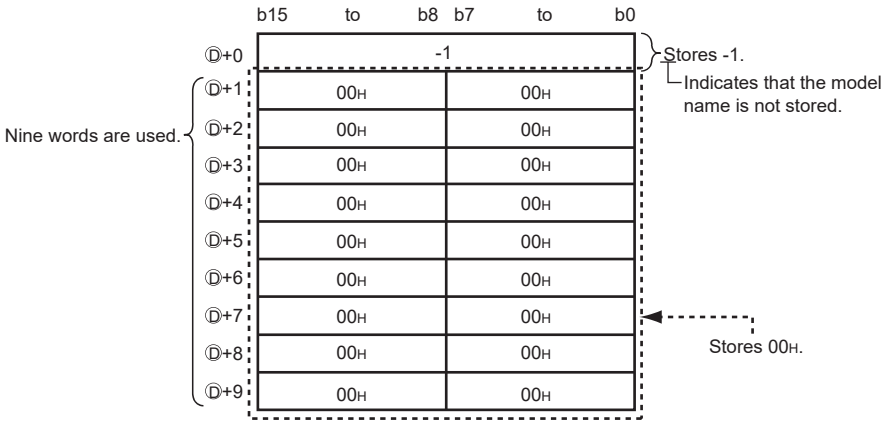
Target module	Stored character string
Input module (16 points)	INPUT_16
Output module (32 points)	OUTPUT_32
I/O combined module (64 points)	MIXED_64
Intelligent function module (16 points)	INTELLIGENT_16

[Character string indicating module type]
- Input module: INPUT
- Output module: OUTPUT
- I/O combined module: MIXED
- Intelligent function module (includes the QI60 and GOT): INTELLIGENT

[Character string indicating the number of points]
- 16 points: _16
- 32 points: _32
- 48 points: _48
- 64 points: _64
- 128 points: _128
- 256 points: _256
- 512 points: _512
- 1024 points: _1024

■Others

- The specified slot is empty or the target module is during online module change.
- The specified value (n) is not the start I/O number.
- The specified value (n) is within the allowable setting range, but cannot be set in the I/O assignment setting screen of the PLC parameter dialog box.



Operation error

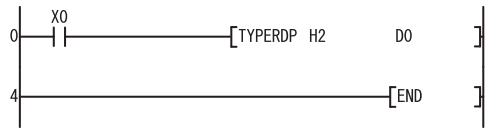
- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
2110	The target module cannot be communicated due to a failure.	—	—	—	—	○	○
4101	The range of the device specified by (D) exceeds that of the device that can be used.	—	—	—	—	○	○
	The value specified in n is not within the range from 0 to FFH or 3E0H to 3E3H.	—	—	—	—	○	—
	The value specified in n is not within the range from 0 to FFH or 3E0H.	—	—	—	—	—	○

Program example

- The following program stores the model name of a module having the start I/O number 0020H in the area starting from D0 when X0 is turned on.

[Ladder Mode]



[List Mode]

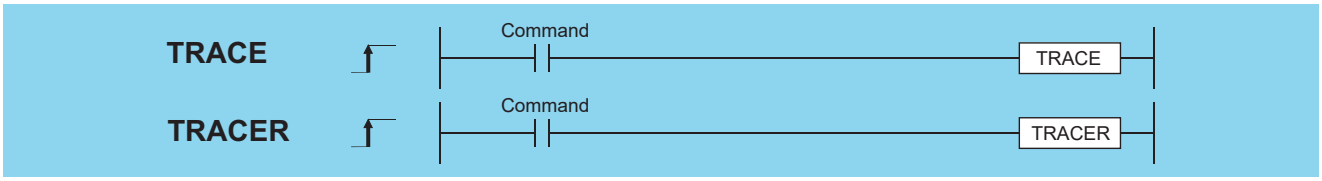
Step	Instruction	Device
0	LD	X0
1	TYPERDP	H2 D0
4	END	

Trace set, trace reset

TRACE, TRACER

Basic | High performance | Process | Redundant | Universal (Ver.) | LCPU

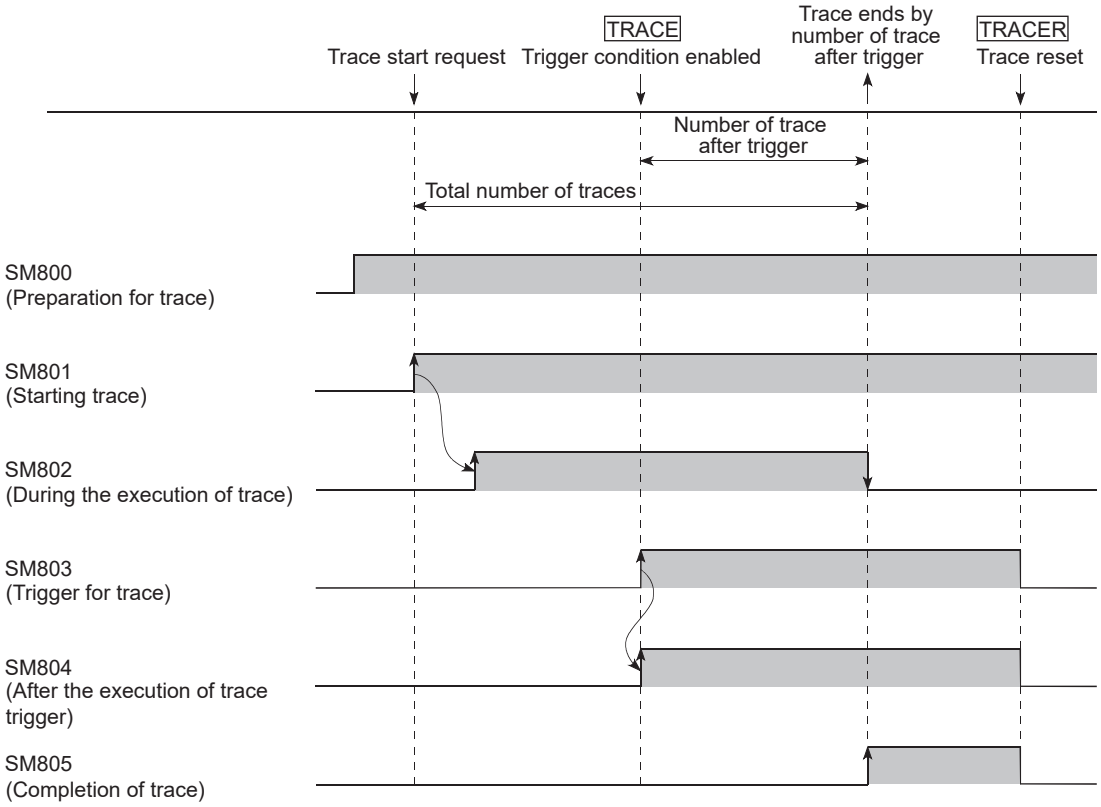
- Universal model QCPU: Models other than Q00UJCPU



Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant	Others
	Bit	Word		Bit	Word				
—	—								

Processing details

- The sampling trace is the function that collects the device data of a CPU module consecutively.
- To execute the sampling trace, turn ON SM801 when SM800 is ON.



■TRACE

- The TRACE instruction is an instruction which performs the following: turn ON SM803, perform the sampling for the number of the sampling trace after executing the TRACE instruction, latch the sampling traces result, and stop the sampling trace.
- The sampling is stopped if SM801 is turned OFF during the trace execution.
- After the TRACE instruction is executed and the sampling trace is stopped, SM805 is turned ON.
- Once the TRACE instruction is executed, the second and the subsequent TRACE instructions are ignored. When the TRACER instruction is executed, the TRACE instruction is enabled again.

■TRACER

- The TRACER instruction resets the TRACE instruction. When the TRACER instruction is executed, the TRACE instruction is enabled again.
- When the TRACER instruction is executed, SM803 to SM805 are turned OFF.

Point

- The target devices for the sampling trace and its timing can be set with a programming tool. For details of the sampling trace, refer to the user's manual (Function Explanation, Program Fundamentals) for the CPU module used.
- The execution of sampling traces is also possible with a programming tool. For the execution of sampling traces by the programming tool, refer to the operating manual for the programming tool used.

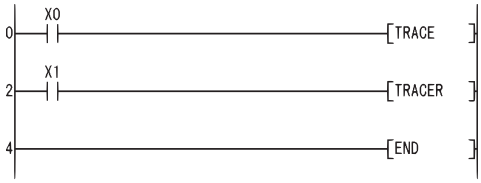
Operation error

- There is no operation error in the TRACE or TRACER instruction.

Program example

- The following program executes the TRACE instruction when X0 is turned ON, and resets the TRACE instruction with the TRACER instruction when X1 is turned ON.

[Ladder Mode]



[List Mode]

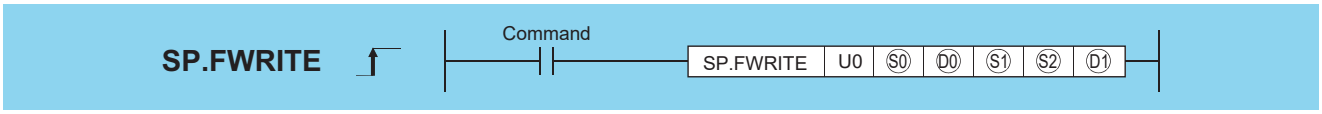
Step	Instruction	Device
0	LD	X0
1	TRACE	
2	LD	X1
3	TRACER	
4	END	

Writing data to designated file

SP.FWRITE

Basic | High performance | Process | Redundant | Universal (Ver.) | LCPU (Ver.)

- Universal model QCPU: Models other than Q00UJCPU, Q00UCPU, and Q01UCPU
- Built-in Ethernet port LCPU: Supported
- L02SCPU and L02SCPU-P cannot be used.



Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant		Others
	Bit	Word		Bit	Word			K, H	$	
(S0)	○	○		—				○	—	—
(D0)	—	△*1		—				—	—	—
(S1)	—	○		—				—	—	—
(S2)	—	△*2		—				—	○	—
(D1)	△*1	△ (Other than T, ST, C)*1		—				—	—	—

*1 Local devices and the file registers set for individual programs cannot be used.

*2 For the Universal model QCPU and LCPU, the file registers that have been specified separately for each local device and program cannot be used only if the number of requested write data exceeds 1024.

Setting data

Setting data	Meaning	Setting range	Set by	Data type
U0	Dummy	—	—	BIN 16 bits
(S0)	Drive designation	2	User	BIN 16 bits

<table>
<tr><th>Setting data</th><th colspan="3">Meaning</th><th>Setting range</th><th>Set by</th><th>Data type</th></tr>
<tr><td>(D0)</td><td colspan="5">Head number of the devices storing the control data. The following control data is required.</td><td>BIN 16 bits</td></tr>
<tr><td></td><td>Device</td><td>Item</td><td>Contents/setting data</td><td>Setting range</td><td>Set by</td><td></td></tr>
<tr><td></td><td>(D0)</td><td>Execution/ completion type</td><td>Designate the execution type.
0000H: Write binary data
0100H: Write data after CSV format conversion</td><td>0000H
0100H</td><td>User</td><td></td></tr>
<tr><td></td><td>(D0)+1</td><td>(Not used)</td><td>Used by system</td><td>—</td><td>System</td><td></td></tr>
<tr><td></td><td>(D0)+2</td><td>Writing result (No. of written data)</td><td>Contains the number of actually written data against the data designated by (S2). The unit of the value depends on data type specified at (D0)+7.</td><td>—</td><td>System</td><td></td></tr>
<tr><td></td><td>(D0)+3</td><td>(Not used)</td><td>—</td><td>—</td><td>—</td><td></td></tr>
<tr><td></td><td>(D0)+4
(D0)+5</td><td>File position</td><td>Set the file position when binary data writing is specified by (D0).
00000000H: Starting at the beginning of the file
00000001H to FFFFFFFEH: From the designated position (The unit for the value is determined by word/ byte unit designation.)
FFFFFFFFH: Addition starts from the end of the file.
When write data after CSV format conversion is specified at (D0)
• For the High Performance model QCPU of which the first five digits of the serial number are "01111" or lower, always set the beginning (0H) of the file.
• For the High Performance model QCPU/Process CPU/Redundant CPU/Universal model QCPU of which the first five digits of the serial number are "01112" or higher, set the file position.
00000000H to FFFFFFFEH: Starting at the beginning of the file
FFFFFFFFH: Adding at the end of the file</td><td>00000000H to FFFFFFFFH</td><td>User</td><td></td></tr>
<tr><td></td><td>(D0)+6</td><td>No. of columns designation</td><td>When binary write is specified at (D0), always set 0.
When write data after CSV format conversion is specified at (D0), set the number of columns where data will be written.
0: No columns. Regarded as one row.
Other than 0: Set to the specified number of columns.</td><td>0H to FFFFH (0 to 65535)</td><td>User</td><td></td></tr>
<tr><td></td><td>(D0)+7</td><td>Data type specification</td><td>0: Word
1: Byte</td><td>0, 1</td><td>User</td><td></td></tr>
<tr><td>(S1)</td><td colspan="5">Head number of the devices storing a file name. A file name is expressed as follows:</td><td>BIN 16 bits</td></tr>
<tr><td></td><td>Device</td><td>Item</td><td>Contents/setting data</td><td>Setting range</td><td>Set by</td><td></td></tr>
<tr><td></td><td>(S1) to (S1)+□</td><td>File name character string</td><td>Designate the character string of a file name.
• When omitting an extension, also omit the "." (Period).
• Limit the file name within 8 characters + period + 3 characters.
• When 9 or more characters are used, the extension is ignored regardless of its presence, and "BIN" or "CSV" is automatically assigned as an extension.</td><td>Character string</td><td>User</td><td></td></tr>
<tr><td>(S2)</td><td colspan="5">Head number of the devices storing the data. Written data is expressed as follows:</td><td>BIN 16 bits</td></tr>
<tr><td></td><td>Device</td><td>Item</td><td>Contents/setting data</td><td>Setting range</td><td>Set by</td><td></td></tr>
<tr><td></td><td>(S2)</td><td>No. of request write data</td><td>Designate the number of data to request writing (word units).
This data should be designated in units of words even when byte is designated by (D0)+7.</td><td>1 to 480
1 to 32767*3</td><td>User</td><td></td></tr>
<tr><td></td><td>(S2)+1 to (S2)+□</td><td>Write data</td><td>Data to request writing.</td><td>0000H to FFFFH</td><td></td><td></td></tr>
</table>

Setting data	Meaning			Setting range	Set by	Data type
(D1)	Bit device that turned ON at the completion of the processing. ((D1)+1 is also turned ON at error completion.)					Bit
	Device	Item	Contents/setting data	Setting range	Set by	
	(D1)	Completion signal	Indicates the completion of the processing. ON: Completed OFF: Not completed	—	System	
	(D1)+1	Error completion signal	Indicates whether the processing is normally completed or abnormally completed. ON: Error completion OFF: Normal completion	—		

*3 Indicates the range applicable for the Universal model QCPU, LCPU.

Precautions

- For only QCPU, only the ATA card drive (2) can be set as (S0) (drive designation). Note that when the Flash card is loaded, the SP.FWRITE instruction cannot be used to perform writing. The SRAM card, standard RAM or standard ROM drive cannot be set. For the High-speed Universal model QCPU, Universal model Process CPU and LCPU, only the SD memory card drive (2) can be set for (S0) (drive designation).
- For CSV setting, the data written are decimal values.

Ex.

Character "A" (41H) → "65" is written.

Handling range: -32768 to 32767

- For binary write, the word-specified file position setting range is 00000000H to 7FFFFFFFH and FFFFFFFFH.
- For the High-speed Universal model QCPU, Universal model Process CPU and LCPU, this instruction cannot be executed while SM606 (SD memory card forced disable instruction) is ON. If the instruction is executed, the command will be ignored.

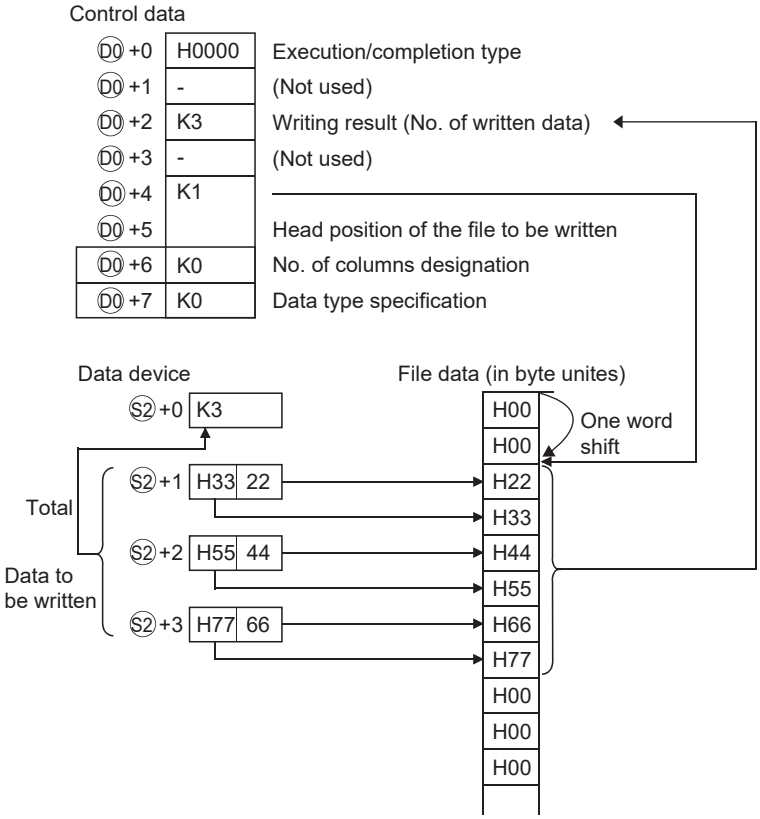
Processing details

- The designated number of data is written to the designated file. Set the execution/completion type in the control data to designate whether to write binary data without any conversion or to convert binary data into CSV format data before writing it. (For QCPU, writing is only supported for ATA cards. For the High-speed Universal model QCPU, Universal model Process CPU and LCPU, it is only supported for SD memory cards.)
- The execution completion bit device ((D1)) is automatically turned ON at the END processing after the completion of the instruction is detected. The bit device is turned OFF at the execution of the END instruction in the next scan. Use this bit device as the execution completion flag for the SP.FWRITE instruction.
  - When this instruction is completed abnormally, the error completion device ((D1)+1) is turned ON/OFF in synchronization with the processing complete ((D1)) device. Use this device as the error completion flag for this instruction.
  - SM721 is turned ON during the execution of the instruction. This instruction cannot be executed while SM721 is ON. (If an attempt is made, no processing is performed.)
  - When an error is detected at the execution of the instruction (before SM721 is turned ON), the processing complete device (D1), the error completion device ((D1)+1), and SM721 are not turned ON.
- Be sure to use in units of words to designate the No. of request write data (S2) and the file position ((D0)+4 and (D0)+5).

■When writing binary data

- If the extension of the target file is omitted, ".BIN" is used as an extension.
- When the designated file does not exist, a new file is created and the data is saved from the beginning of the file. The attributes of this new file are set using the archive attributes.
- When the designated file exists, the data is saved from the beginning of the file. When the size of the data exceeds that of the existing area in the file during the writing, the excess data is added/saved.
- If the file position specified is greater than the existing file size:
  - The High Performance model QCPU of which the first five digits of the serial number are "01111" or lower results in an error.
  - The High Performance model QCPU/Process CPU/Redundant CPU/Universal model QCPU/LCPU of which the first five digits of the serial number are "01112" or higher performs writing at point 0 and is completed normally.
- An error occurs when the saving space becomes full while data is added and saved. In such a case, the data that is successfully added/saved remains in the medium. The error completion is indicated after as much data as possible is added/saved.

The following shows the method for writing binary data when No. of request write data and file position are specified.



■When writing data after CSV format conversion

- If the extension is omitted, ".CSV" is used as an extension.
- When the existing file is specified:

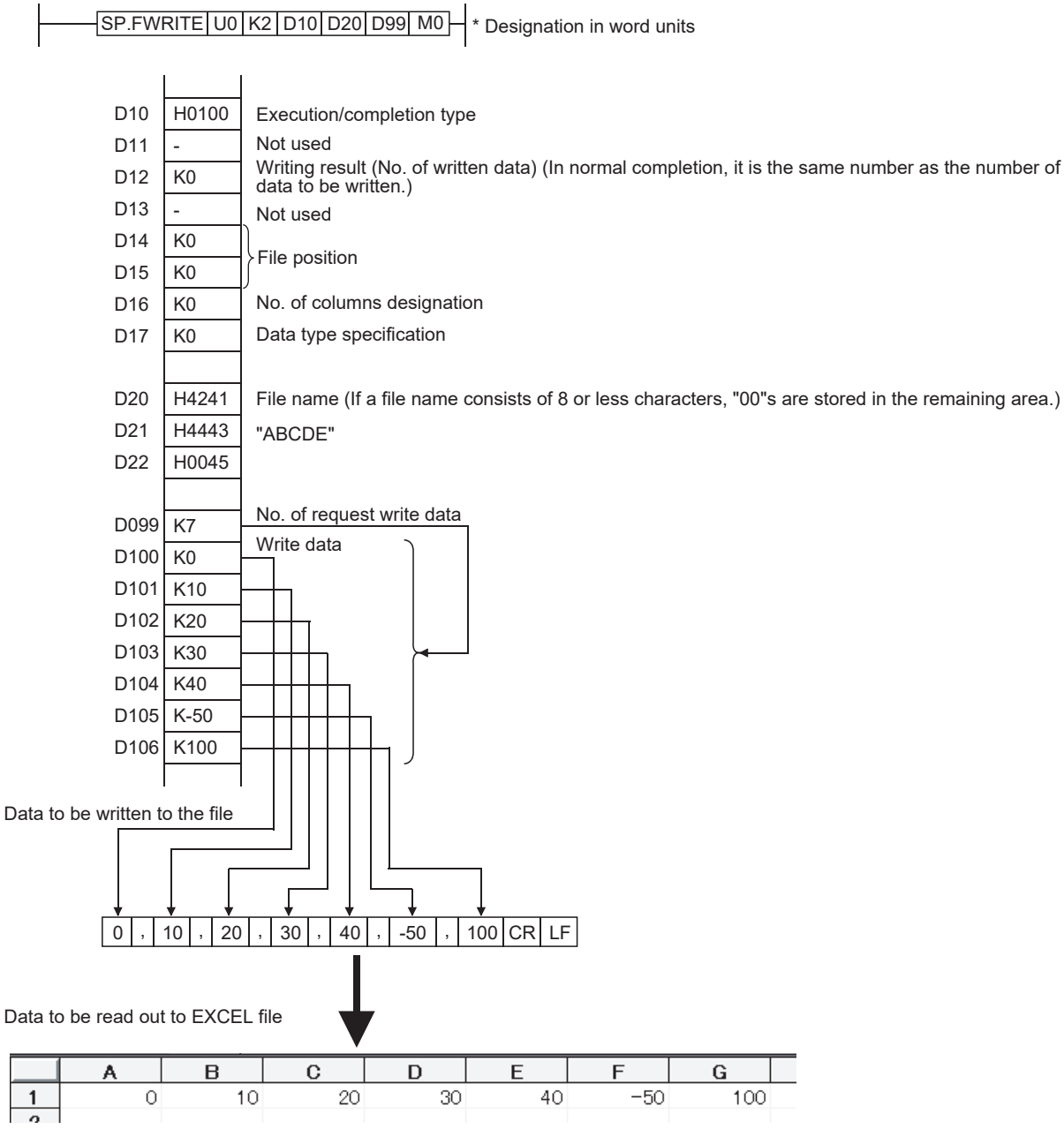
[High Performance model QCPU of which the first five digits of the serial number are "01111" or lower]
File contents are all deleted and data are saved, starting at the beginning.
[High Performance model QCPU/Process CPU/Redundant CPU/Universal model QCPU/LCPU of which the first five digits of the serial number are "01112" or higher]

- When other than FFFFFFFFH is set at ((D0)+4, (D0)+5), file contents are all deleted and data are saved, starting at the beginning.
- When FFFFFFFFH is set at ((D0)+4, (D0)+5), data are saved, starting at the end of the file.

- When the designated file does not exist, a new file is created and the data is saved from the beginning of the file. The attributes of this new file are set using the archive attributes.
- An error occurs when the saving space becomes full while data is added and saved. In such a case, the data that is successfully added/saved remains in the medium. The error completion is indicated after as much data as possible is added/saved.
- When the designated number of columns is "0", the data is stored as single-row data in CSV format file.

Ex.

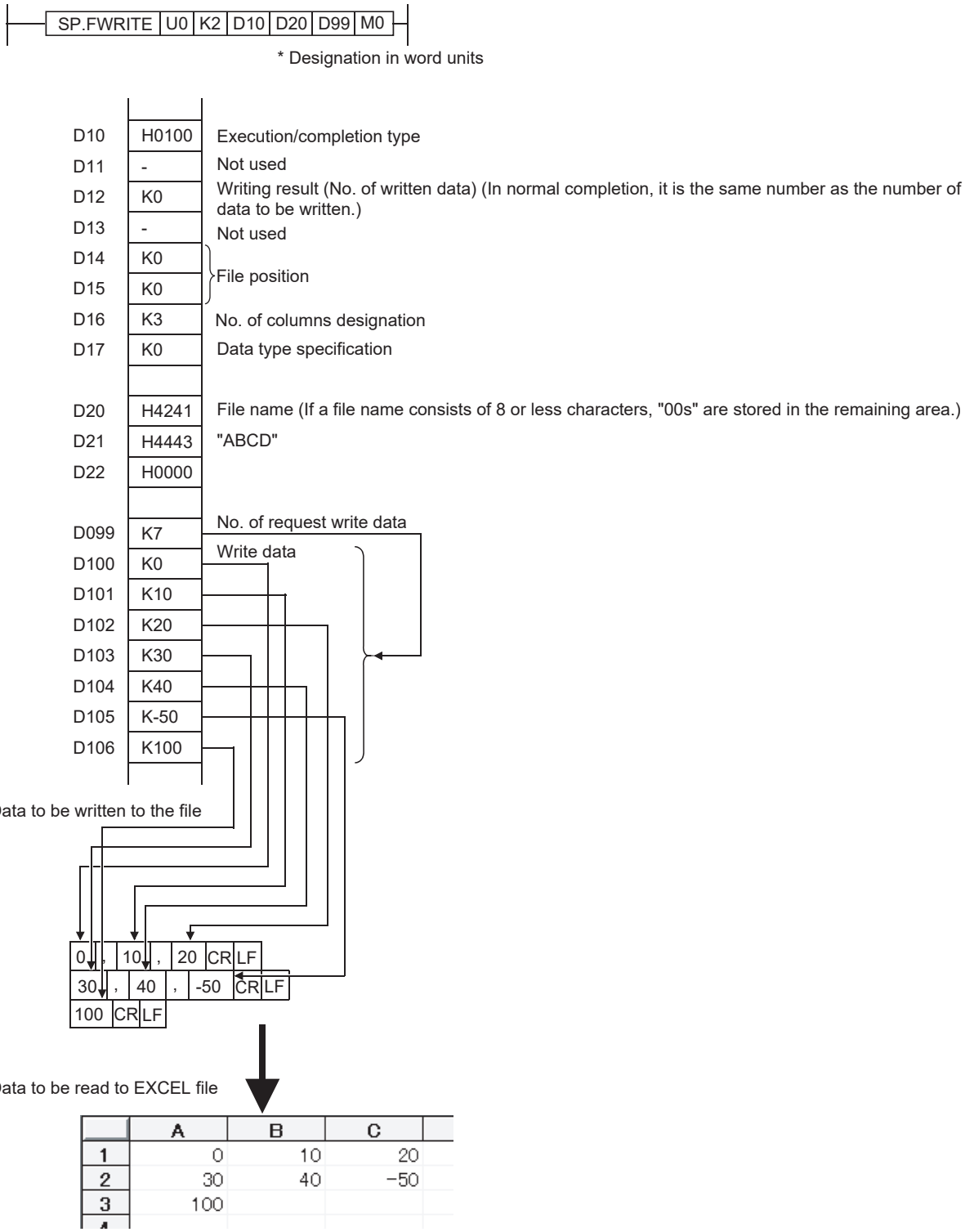
When data is written after CSV format conversion and the designated No. of columns is "0":



- When data is written after CSV format conversion and the designated number of columns is other than "0", the data is stored as table data with designated number of columns in a CSV format file.

Ex.

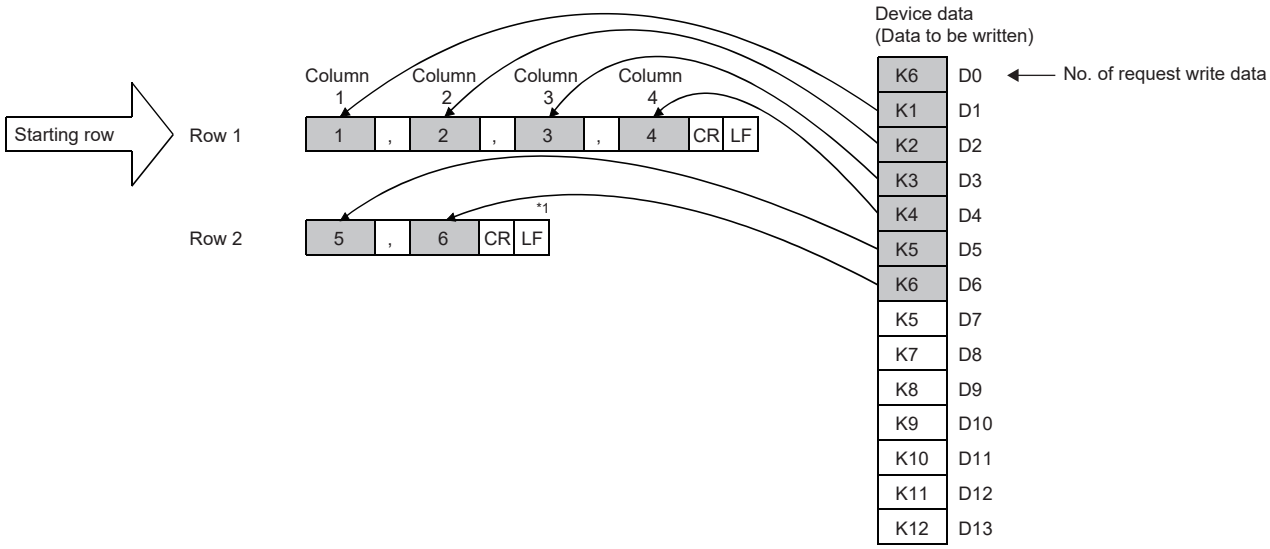
When data is written after CSV format conversion and the designated No. of columns is other than "0":



• When data is added by the High Performance model QCPU/Process CPU/Redundant CPU/Universal model QCPU/LCPU of which the first five digits of the serial number are 01112 or higher:

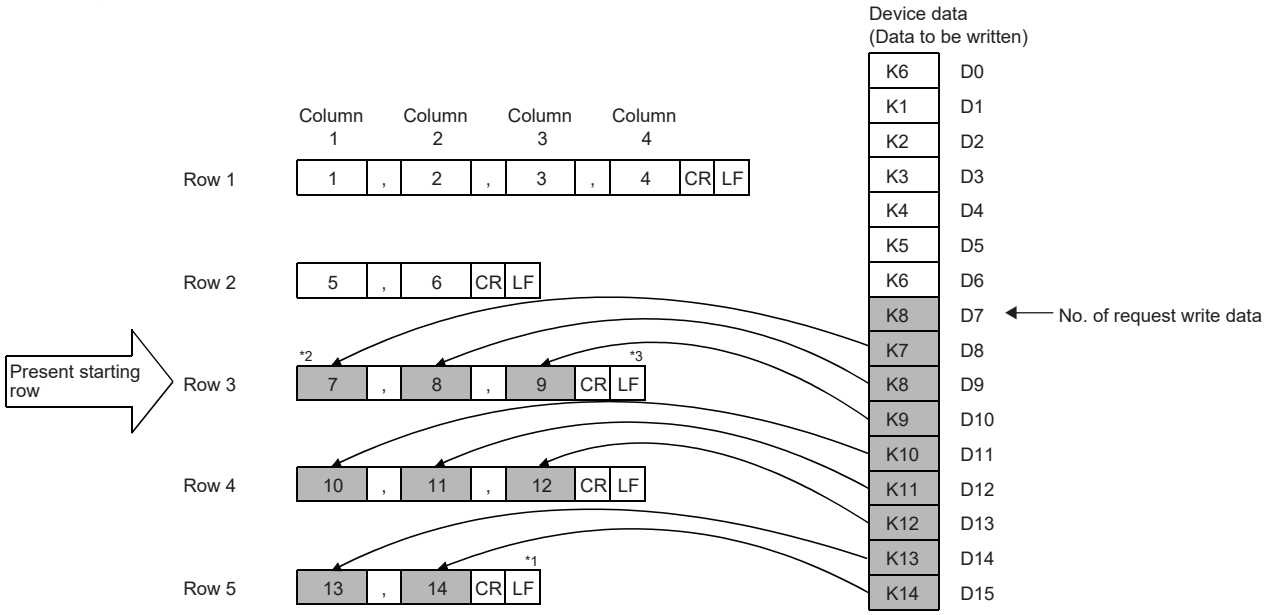
[Specify the file to which data will be written.] (If a file exists, delete it and create a new file again.)

Execution type	= CSV format	File position	= 0H (New file is created)
No. of columns designation	= 4H*1*3	Write head device	= D0
Data type specification	= Word	No. of request write data	= 6H*1



[In the addition mode, make addition from the end of the file.]

Execution type	= CSV format	File position	= FFFFFFFFH (Addition mode)
No. of columns designation	= 3H*1*3	Write head device	= D7
Data type specification	= Word	No. of request write data	= 8H*1



*1 Unless the "No. of request write data" is set to an integral multiple of "No. of columns designation", the column numbers will be random.

*2 Since the last data is always followed by the line feed code, addition normally starts at the beginning of the new row in the addition mode.

*3 If, in the addition mode, "column designation" is changed from that in the previous writing, the column numbers are shifted.

• Do not execute the SP.FWRITE instruction in an interrupt program. (If execute it, the operation is not guaranteed.)

• Below is the method for calculating the file size (total number of bytes) when a CSV format file is written to the ATA card.

Total number of bytes = Total bytes excluding final line + bytes of final line
(Number of bytes on a line = number of columns[*4] + 1 + total bytes of all data values on line[*5])

*4 For all lines but the final line, this is the specified number of columns. The number of columns on the final line depends on the number of columns specified via the amount of data written. It is calculated as follows.
- The number of lines excluding the final line is calculated. (Number of lines excluding final line = Amount of data in write request ÷ number of columns (remainders discarded))
- The number of columns in the final line is calculated. Number of columns in final line = Amount of data in write request - number of lines excluding final line × number of columns)

*5 The number of bytes for each data value is calculated as shown below.

Sign of data value	Bytes per data value	Byte count range	Examples
Positive	Num. digits	1 to 5 (word specified) 1 to 3 (byte specified)	12345: 5 bytes 67: 2 bytes
Negative	Num. digits + 1	2 to 6 (word specified) 2 to 4 (byte specified)	-12345: 6 bytes -67: 3 bytes

Operation error

• In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

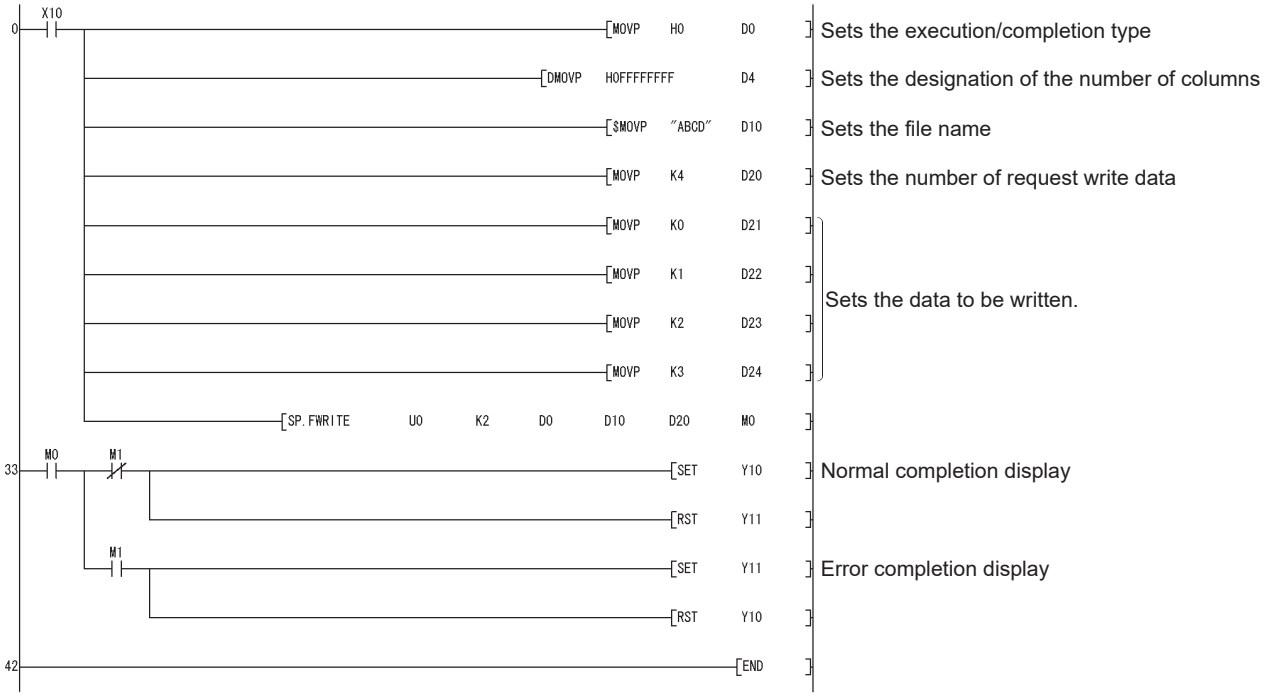
Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4004	The device that cannot be specified has been specified.	—	○	○	○	○	○
4100	Values specified in control data (D0) and the subsequent devices are out of the setting range. No space is found when a new file is created. A value that cannot be used has been set for the file name (S1). The attribute of the file name (S1) is "read only".	—	○	○	○	○	○
	The drive specified by drive designation device (S0) contains the medium other than the ATA card. Space in the ATA card is insufficient. An access error occurred in the ATA card.	—	○	○	○	○	—
	The drive specified by drive designation device (S0) contains the medium other than the SD Memory card. Space in the SD Memory card is insufficient. An access error occurred in the SD Memory card.	—	—	—	—	○	○
	If the instruction accesses a file which has been accessed by another function	—	—	—	—	○	○
	If the instruction writes data into an SD memory card with the write protect switch enabled (write inhibited)	—	—	—	—	○	○
4101	The value specified in "No. of request write data" (S2) is out of the setting range, or exceeds the device range specified in ((S2)+1) or the subsequent devices.	—	○	○	○	○	○
	The range of the device specified in (D0) or (D1) exceeds that of the corresponding device.	—	—	—	—	○	○

Program example

- When X10 is turned ON, the following program adds four bytes of binary data (00H, 01H, 02H, and 03H) to file "ABCD.BIN" in the memory card inserted to drive 2.

  • Assume that 8 points from (D0) are reserved for the control data devices.

[Ladder Mode]



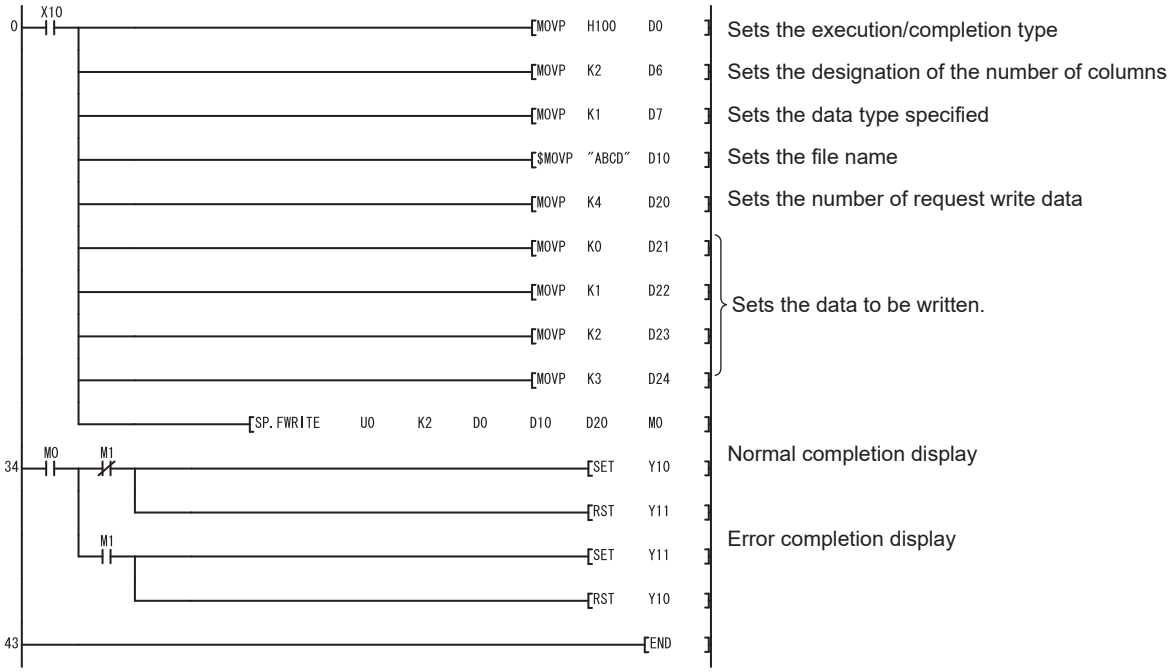
[List Mode]

Step	Instruction	Device					
0	LD	X10					
1	MOVP	H0	D0				
3	DMOVP	H0FFFFFFFF	D4				
6	$MOVP	"ABCD"	D10				
11	MOVP	K4	D20				
13	MOVP	K0	D21				
15	MOVP	K1	D22				
17	MOVP	K2	D23				
19	MOVP	K3	D24				
21	SP.FWRITE	U0	K2	D0	D10	D20	M0
33	LD	M0					
34	MPS						
35	ANI	M1					
36	SET	Y10					
37	RST	Y11					
38	MPP						
39	AND	M1					
40	SET	Y11					
41	RST	Y10					
42	END						

• When X10 is turned ON, the following program creates a file named "ABCD.CSV" in the memory card inserted to drive 2, and writes four bytes of data (00H, 01H, 02H, and 03H) as two-column table data in CSV format.

• Assume that 8 points from (D0) are reserved for the control data devices.

[Ladder Mode]



[List Mode]

Step	Instruction	Device					
0	LD	X10					
1	MOVP	H100	D0				
3	MOVP	K2	D6				
5	MOVP	K1	D7				
7	$MOVP	"ABCD"	D10				
12	MOVP	K4	D20				
14	MOVP	K0	D21				
16	MOVP	K1	D22				
18	MOVP	K2	D23				
20	MOVP	K3	D24				
22	SP.FWRITE		U0	K2	D0	D10	D20	M0
34	LD	M0					
35	MPS						
36	ANI	M1					
37	SET	Y10					
38	RST	Y11					
39	MPP						
40	AND	M1					
41	SET	Y11					
42	RST	Y10					
43	END						

• The written file is displayed as follows:

0	,	0	,	CR	LF
1	,	0	,	CR	LF
2	,	0	,	CR	LF
3	,	0	,	CR	LF

Contents of the file to be written

Data to be read to the EXCEL file

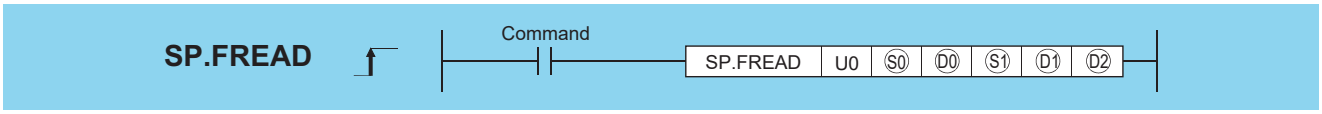
	A	B
1	0	0
2	1	0
3	2	0
4	3	0

Reading data from designated file

SP.FREAD

Basic | High performance | Process | Redundant | Universal (Ver.) | LCPU (Ver.)

- Universal model QCPU: Models other than Q00UJCPU, Q00UCPU, and Q01UCPU
- Built-in Ethernet port LCPU: Supported
- L02SCPU and L02SCPU-P cannot be used.



Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant		Others
	Bit	Word		Bit	Word			K, H	$	
(S0)	○	○		—				○	—	—
(D0)	—	○		—				—	—	—
(S1)	—	○		—				—	○	—
(D1)	—	△*1		—				—	—	—
(D2)	△*1	△ (Other than T, ST, C)*1		—				—	—	—

*1 Local devices and the file registers set for individual programs cannot be used.

Setting data

Setting data	Meaning			Setting range	Set by	Data type
U0	Dummy			—	—	BIN 16 bits
(S0)	Drive designation			2	User	BIN 16 bits
(D0)	Head number of the devices storing the control data. The following control data is required.					BIN 16 bits
	Device	Item	Contents/setting data	Setting range	Set by	
	(D0)	Execution/ completion type	Designate the execution type. 0000H: Read binary data 0100H: Read data after CSV format conversion	0000H 0100H	User	
	(D0)+1	(Not used)	Used by system	—	System	
	(D0)+2	No. of request read data	Designate the number of data to request reading. (Unit: Word) Even when byte is specified at (D0)+7 by data type specification, specify the value in units of words (16 bits), not in units of bit devices.	1 to 480 1 to 32767*2	User	
	(D0)+3	(Not used)	—	—	—	

Setting data	Meaning			Setting range	Set by	Data type
(D0)	(D0)+4 (D0)+5	File position	Designate the file position to start reading when binary data reading is designated by (D0). 00000000H: Starting at the beginning of the file 00000001H to FFFFFFFEH: From the designated position (The unit for the value is determined by word/byte unit designation.) FFFFFFFFH: Setting disabled When CSV format read is specified at (D0) • For the High Performance model QCPU of which the first five digits of the serial number are "01111" or lower, always set the beginning (0H) of the file. • For the High Performance model QCPU/Process CPU/Redundant CPU/Universal model QCPU/LCPU of which the first five digits of the serial number are "01112" or higher, set the file position (Row). 00000000H: Read starts at the beginning of the file. 00000001H to FFFFFFFEH: Read starts at the specified row. FFFFFFFFH: Read continues, starting at the previous read position.	00000000H to FFFFFFFFH	User	BIN 16 bits
	(D0)+6	No. of columns designation	When binary read is specified at (D0), always set 0. When read data after CSV format conversion is specified at (D0), set the number of columns from where data will be read. 0: No columns. Regarded as one row. Other than 0: Regarded as the specified number of columns.	0H to FFFFH (0 to 65535)	User	
	(D0)+7	Data type specification	0: Word 1: Byte	0, 1	User	
(S1)	Head number of the devices storing a file name. A file name is expressed as follows:					BIN 16 bits
	Device	Item	Contents/setting data	Setting range	Set by	
	(S1) to (S1)+□	File name character string	Designate the character string of a file name. • When omitting an extension, also omit the "." (Period). • Limit the file name within 8 characters + period + 3 characters. • When 9 or more characters are used, the extension is ignored regardless of its presence, and "BIN" or "CSV" is regarded as an extension.	Character string	User	
(D1)	Head number of the devices for storing the read data.					BIN 16 bits
	Device	Item	Contents/setting data	Setting range	Set by	
	(D1)	Reading result (No. of read data)	Contains the number of actually read data against the data designated by (D0)+2. The unit on the value depends on data type specification.	—	System	
	(D1)+1 to (D1)+□	Reading data	Read data	—	System	
(D2)	Bit device that turned ON at the completion of the processing. ((D2)+1 is also turned ON at error completion.)					Bit
	Device	Item	Contents/setting data	Setting range	Set by	
	(D2)	Completion signal	Indicates the completion of the processing. ON: Completed OFF: Not completed	—	System	
	(D2)+1	Error completion signal	Indicates whether the processing is normally completed or abnormally completed. ON: Error completion OFF: Normal completion	—		

*2 Indicates the range applicable for the Universal model QCPU, LCPU.

Precautions

- At (S0) (drive designation), only the ATA card drive (2) can be set.(For QCPU) Note that when the Flash card is loaded, the SP.FREAD instruction cannot be used to perform read. The SRAM card, standard RAM or standard ROM drive cannot be set. For the High-speed Universal model QCPU, Universal model Process CPU and LCPU, only the SD memory card drive (2) can be set for (S0) (drive designation).
- For CSV setting, the data read are decimal values.

Ex.

Character "A" (41H) → "65" is read.

Handling range: -32768 to 32767

- For binary read, the word-specified file position setting range is 00000000H to 7FFFFFFFH.
- For the High-speed Universal model QCPU, Universal model Process CPU and LCPU, this instruction cannot be executed while SM606 (SD memory card forced disable instruction) is ON. If the instruction is executed, the command will be ignored.

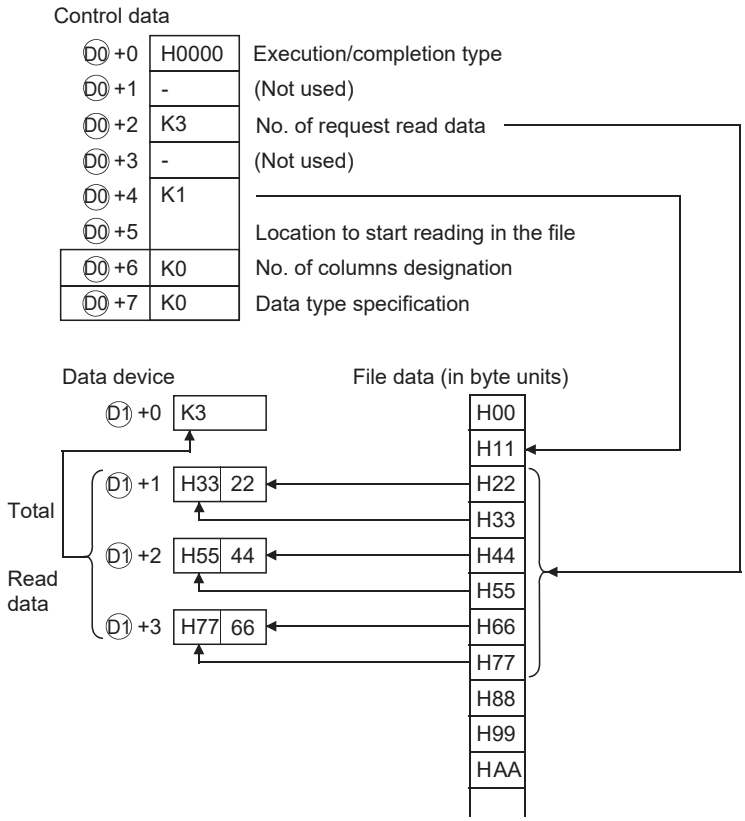
Processing details

- Data is read from the designated file. Set the execution/completion type in the control data to designate whether to read binary data without any conversion or to convert binary data into CSV format data before reading it. (The QCPU reads data only from ATA cards; on the other hand, the High-speed Universal model QCPU, Universal model Process CPU and LCPU read data only from SD memory cards.)
- The execution completion bit device (D2) is automatically turned ON at the END processing after the completion of the instruction is detected. The bit device is turned OFF at the execution of the END instruction in the next scan. Use this bit device as the execution completion flag for the SP.FWRITE instruction.
  - When this instruction is completed abnormally, the error completion device ((D2)+1) is turned ON/OFF in synchronization with the execution completion (D2) device. Use this device as the error completion flag for this instruction.
  - SM721 is turned ON during the execution of the instruction. This instruction cannot be executed while SM721 is ON. (If an attempt is made, no processing is performed.)
  - When an error is detected at the execution of the instruction (before SM721 is turned ON), the processing complete device (D1), the error completion device ((D1)+1), and SM721 are not turned ON.
- Be sure to use word units to designate the number of request read data ((D0)+2), file position ((D0)+4 and (D0)+5), and reading result (No. of read data) (D1).

■When reading binary data

- If the extension of the target file is omitted, ".BIN" is used as an extension.
- When the designated file does not exist, an error occurs.
- If the position specified is greater than the existing file size:
  - The High Performance model QCPU of which the first five digits of the serial number are "01111" or lower results in an error.
  - The High Performance model QCPU/Process CPU/Redundant CPU/Universal model QCPU/LCPU of which the first five digits of the serial number are '01112' or higher will perform reading at point 0 and will be completed normally.

The following shows how the data is read in binary data reading operation.



■When reading data after CSV format conversion

- The elements in CSV format file (cells for EXCEL) are read by each row. The numerical value and character strings are converted into binary data and stored in the device.
- If the extension is omitted, ".CSV" is used as an extension.
- When the designated file does not exist, an error occurs.
- The data designated by the number of request read data ((D0)+2) are read from the beginning of the file. When the last data of the file is reached before the specified number of data are read:
  - The High Performance model QCPU of which the first five digits of the serial number are "01111" or lower results in an error.
  - The High Performance model QCPU/Process CPU/Redundant CPU/Universal model QCPU/LCPU whose the first five digits of the serial number are '01112' or higher reads the data up to the point where the reading is possible.

• When the designated number of columns is 0, the data is read by ignoring the rows in CSV format file.

Ex.

When data is read after CSV format conversion and the designated No. of columns is 0:

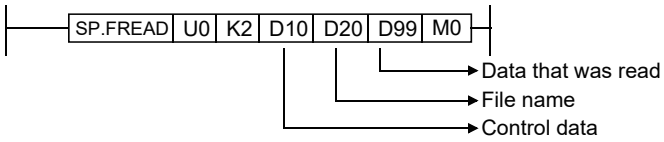
Data created by EXCEL

	A	B	C
1	Main / sub item		Measured value
2	Length	1	3
3	Temperature	−21	

Data saved in the CSV format

Main / sub item	,	,	Measured value	CR	LF			
Length	,	1	,	3	CR	LF		
Temperature	,	-21	,	CR	LF			

Data to be read into devices



Control data

D10	H0100	Execution/completion type
D11	-	Not used
D12	K9	No. of request read data
D13	-	Not used
D14	K0	File position
D15	K0	
D16	K0	No. of columns designation
D17	K0	Data type specification
D20	H4241	File name
D21	H4443	"ABCDE"
D22	H0045	

Loaded data

Stores the number of read data	D099	K9	··· Reading result (No. of read data)
Main/sub item	D100	K0	··· Conversion data (0) is stored since "Main/sub item" is nonnumeric data.
Data between , and ,	D101	K0	··· Conversion data (0) is stored since " " is nonnumeric data.
Measured value	D102	K0	··· Conversion data (0) is stored since "Measured value" is nonnumeric data.
Length	D103	K0	··· Conversion data (0) is stored since "Length" is nonnumeric data.
Read data: 1	D104	K1	··· Since " 1 " is a numeric value, it is converted to a binary value.
Read data: 3	D105	K3	··· Since " 3 " is a numeric value, it is converted to a binary value.
Temperature	D106	K0	··· Conversion data (0) is stored since "Temperature" is nonnumeric data.
-21	D107	K-21	··· Since " -21 " is a numeric value, it is converted to a binary value.
Data between , and CR	D108	K0	··· Conversion data (0) is stored since " " is nonnumeric data.

• If the number of columns varies in each row, the data is also read by ignoring the rows.

Point

Such file cannot be created using EXCEL. This happens when CSV file is modified by a user.

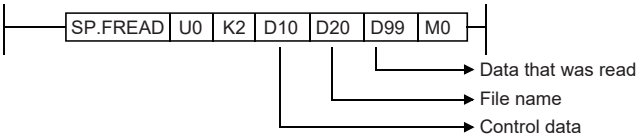
Ex.

If the number of columns varies in each row when the data is read:

Data saved as a CSV file

Main / sub item	,	,	Measured value	,	Excess	CR	LF	
Length	CR	LF						
Temperature	,	-21	,	CR	LF			

Data to be read into devices



Control data

D10	H0100	Execution/completion type
D11	-	Not used
D12	K7	Number of request read data
D13	-	Not used
D14	K0	File position
D15	K0	
D16	K0	Number of columns designation
D17	K0	Data type specification
D20	H4241	File name
D21	H4443	"ABCD"
D22	H0000	

Loaded data

Stores the number of read data

	D099	K7	··· Reading result (Number of read data)
Main/sub item	D100	K0	··· Conversion data (0) is stored since "Main/sub item" is nonnumeric data.
Data between , and ,	D101	K0	··· Conversion data (0) is stored since " " is nonnumeric data.
Measured value	D102	K0	··· Conversion data (0) is stored since "Measured value" is nonnumeric data.
Excess	D103	K0	··· Conversion data (0) is stored since "Excess" is nonnumeric data.
Length	D104	K0	··· Conversion data (0) is stored since "Length" is nonnumeric data.
Temperature	D105	K0	··· Conversion data (0) is stored since "Temperature" is nonnumeric data.
-21	D106	K-21	··· Since " -21 " is a numeric value, it is converted to a binary value.

Read data

- When data is read after CSV format conversion and the designated number of columns is other than 0, the data is read as the table with designated number of columns in CSV format file. The elements outside of the designated columns are ignored.

Ex.

When data is read after CSV format conversion and the designated No. of columns is other than "0":

Data created by EXCEL

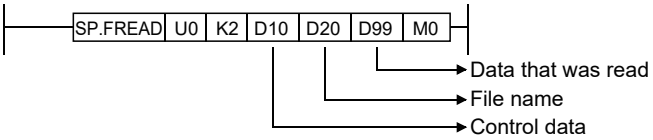
	A	B	C
1	Main / sub item		Measured value
2	Length	1	3
3	Temperature	−21	

Data saved in the CSV format

Main / sub item	,	,	Measured value	CR	LF			
Length	,	1	,	3	CR	LF		
Temperature	,	-21	,	CR	LF			

Elements outside the designated number of columns are ignored.

Data to be read into devices



Control data

D10	H0100	Execution/completion type
D11	-	Not used
D12	K6	No. of request read data
D13	-	Not used
D14	K0	File position
D15	K0	
D16	K2	No. of columns designation
D17	K0	Data type specification
D20	H4241	File name
D21	H4443	"ABCD"
D22	H0000	

Loaded data

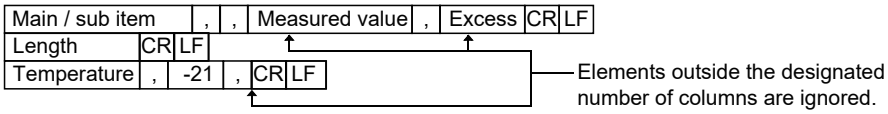
Stores the number of read data

Read data				
		D099	K6	··· Reading result (No. of read data)
	Main/sub item	D100	K0	··· Conversion data (0) is stored since "Main/sub item" is nonnumeric data.
	Data between , and ,	D101	K0	··· Conversion data (0) is stored since " " is nonnumeric data.
	Length	D102	K0	··· Conversion data (0) is stored since "Length" is nonnumeric data.
	1	D103	K1	··· Since " 1 " is a numeric value, it is converted to a binary value.
	Temperature	D104	K0	··· Conversion data (0) is stored since "Temperature" is nonnumeric data.
	-21	D105	K-21	··· Since " -21 " is a numeric value, it is converted to a binary value.

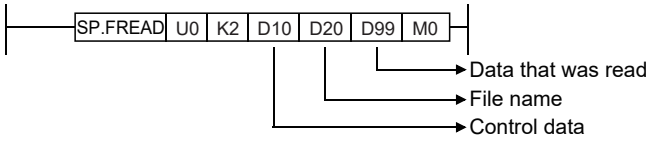
- If the number of columns varies in each row, the elements outside the designated number of columns are ignored and, for a row where elements are insufficient to satisfy the designate number of columns, "0" is added as a replacement for element.

Ex.

If the number of columns varies in each row when the data is read:



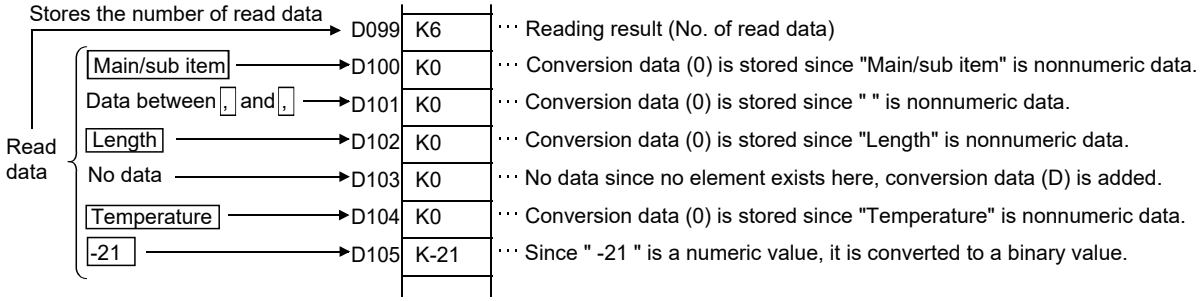
Data to be read into devices



Control data

D10	H0100	Execution/completion type
D11	-	Not used
D12	K6	No. of request read data
D13	-	Not used
D14	K0	File position
D15	K0	
D16	K2	No. of columns designation
D17	K0	Data type specification
D20	H4241	File name
D21	H4443	"ABCD"
D22	H0000	

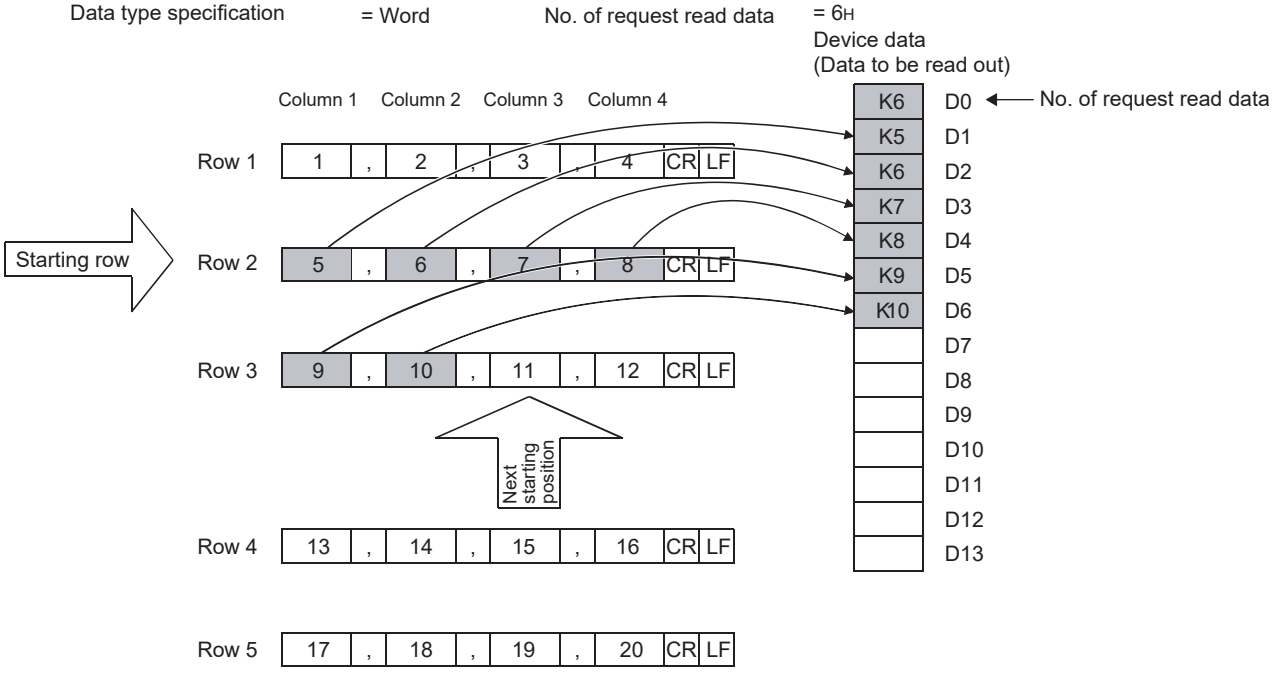
Loaded data



• With the High Performance model QCPU/Process CPU/Redundant CPU/Universal model QCPU/LCPU whose first five digits of the serial number are "01112" or later, it is possible to divide read operation into multiple times.

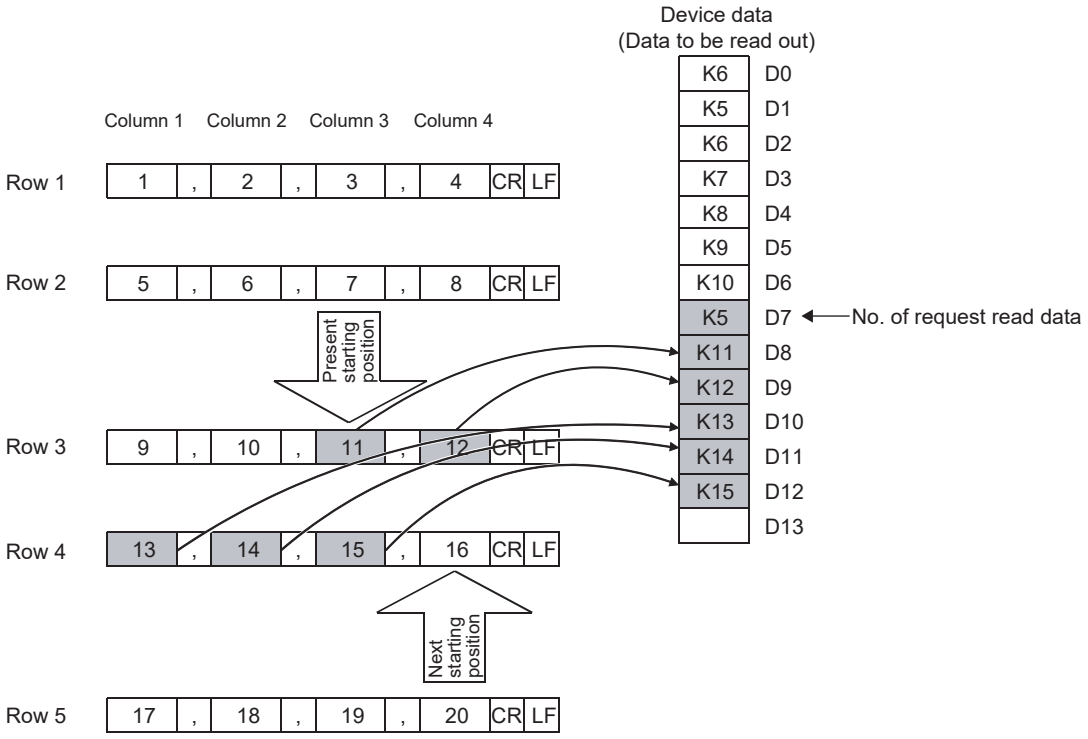
[Specify the row desired to start read.]

Execution type	= CSV format	Starting row number	= 2H
No. of columns designation	= 4H	Read head device	= D0
Data type specification	= Word	No. of request read data	= 6H



[In the continuation mode, read continues from the end of the previous read position.]

Execution type	= CSV format	Starting row number	= FFFFFFFH (Continuation mode)
No. of columns designation	= 4H	Read head device	= D7
Data type specification	= Word	No. of request read data	= 5H



Point

- When read is performed in the continuation mode, the previous addition cannot be made normally if the "execution type", "No. of columns designation" and "data type specification" settings differ from those at the previous time.
- The previous addition cannot be made normally if the SP.FREAD instruction or SP.FWRITE instruction with another setting is executed while data is being read continuously in the continuation mode.

- When data is read after CSV format conversion, the numerical values that are out of range or the elements other than numerical values in the object CSV format file are converted into 0H.
- When data is read after CSV format conversion, numerical values are read and converted as follows:

Numerical values in CSV format		-32768 to -1	0 to 32767	32768 to 65535
Word device	Without a sign	32768 to 65535	0 to 32767	32768 to 65535
	With a sign	-32768 to -1	0 to 32767	-32768 to -1

- Do not execute this instruction in an interrupt program. (Otherwise, a malfunction may result.)

Operation error

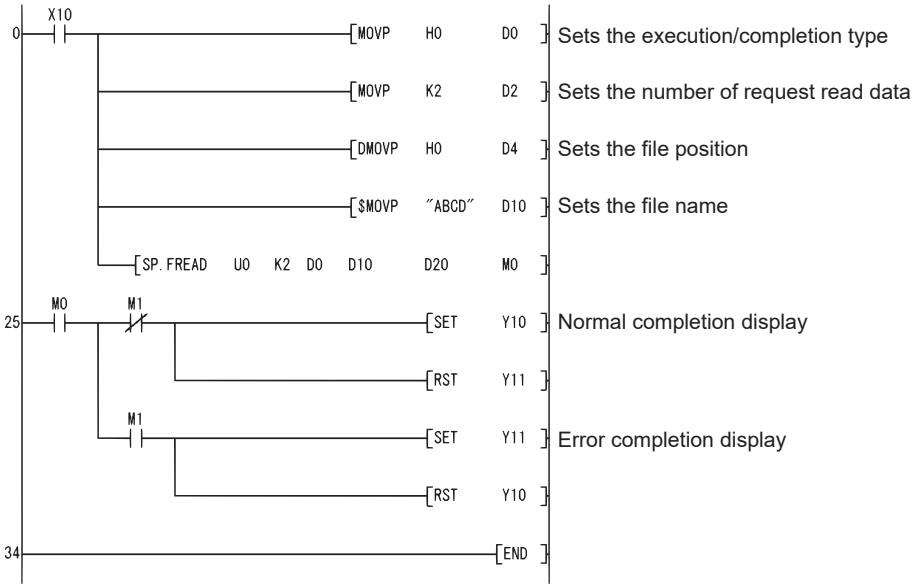
- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
2410	The file name specified in file name character string (S1) or the subsequent devices does not exist in the specified drive.	—	○	○	○	○	○
4004	The device that cannot be specified has been specified.	—	○	○	○	○	○
4100	Values designated in control data (D0) and the subsequent devices are out of the setting range. (Excluding ((D0)+2)	—	○	○	○	○	○
	The drive specified by drive designation device (S0) contains the medium other than the ATA card. An access error occurred in the ATA card.	—	○	○	○	○	—
	When binary data is read, the number of data in the file is less than the size designated by the number of request read data ((D0)+2).	—	○	—	—	—	—
	The drive specified by drive designation device (S0) contains the medium other than the SD Memory card. An access error occurred in the SD Memory card.	—	—	—	—	○	○
4101	The value specified in number of data blocks to be read ((D0)+2) is out of the setting range. The size of read data exceeds that of the reading device.	—	○	○	○	○	○
	The range of the device specified by (D0) or (D2) exceeds the range of the corresponding device.	—	—	—	—	○	○

Program example

- The following program reads 4 bytes of binary data from the beginning of file "ABCD.BIN" in the memory card inserted to drive 2 when X10 is turned ON.
  - Assume that 8 points from (D0) are reserved for the control data devices.
  - Assume that 100 bytes from D20 are reserved for the reading devices.

[Ladder Mode]

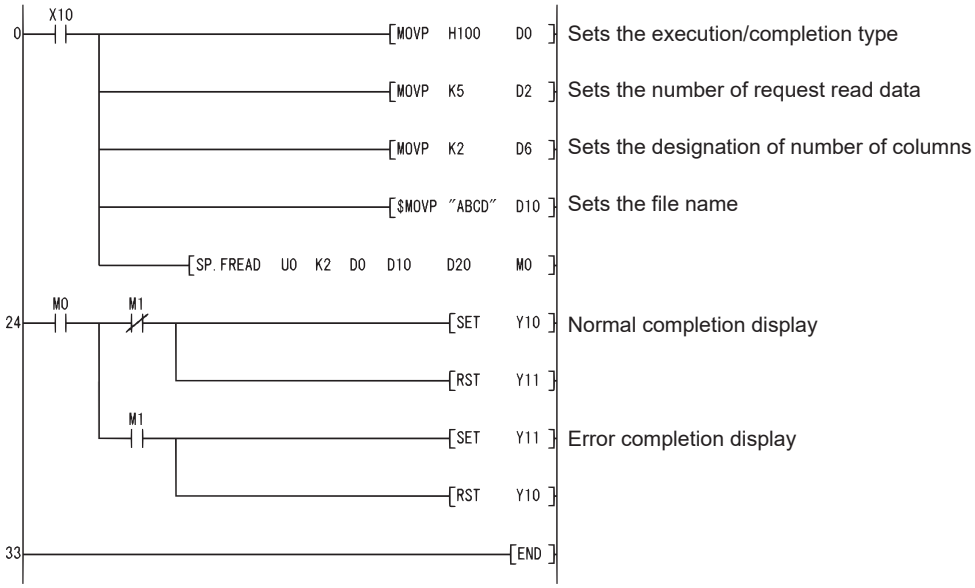


[List Mode]

Step	Instruction	Device				
0	LD	X10				
1	MOVP	H0	D0			
3	MOVP	K2	D2			
5	DMOVP	H0	D4			
8	$MOVP	"ABCD"	D10			
13	SP. FREAD	U0	K2	D0	D10	D20	M0
25	LD	M0				
26	MPS					
27	AN I	M1				
28	SET	Y10				
29	RST	Y11				
30	MPP					
31	AND	M1				
32	SET	Y11				
33	RST	Y10				
34	END					

- The following program reads file "ABCD.CSV" in the memory card inserted to drive 2 as two-column table data in CSV format when X10 is turned ON.
  - Assume that 8 points from (D0) are reserved for the control data devices.
  - Assume that 100 bytes from D20 are reserved for the reading devices.
  - Assume that the target CSV format file contains numerical values only.

[Ladder Mode]



[List Mode]

Step	Instruction	Device				
0	LD	X10				
1	MOVP	H100	D0			
3	MOVP	K5	D2			
5	MOVP	K2	D6			
7	$MOVP	"ABCD"	D10			
12	SP. FREAD	U0	K2	D0	D10	D20	M0
24	LD	M0				
25	MPS					
26	AND	M1				
27	SET	Y10				
28	RST	Y11				
29	MPP					
30	AND	M1				
31	SET	Y11				
32	RST	Y10				
33	END					

Writing data to standard ROM

SP.DEVST

Basic | High performance | Process | Redundant | Universal | LCPU

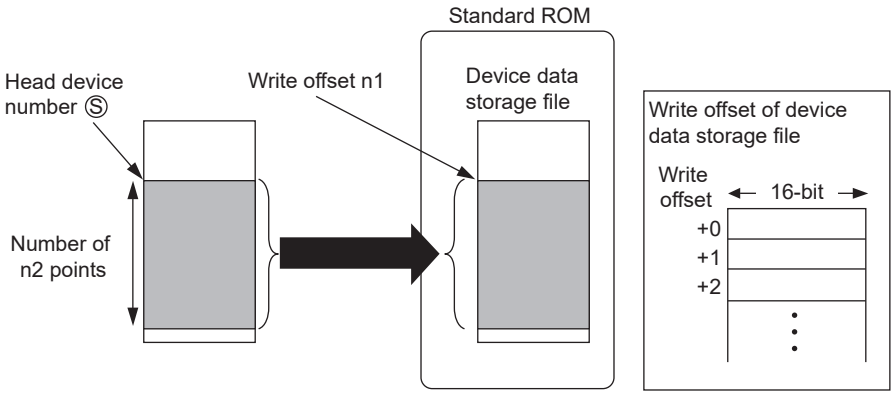
n1: Write offset of the device data storage file (specified in units of 16-bit words) (BIN 32-bit)
(S): Head device number written to the standard ROM (device name)
n2: The number of write points (BIN 16-bit)
(D): (D)+0: Completion device (bit)
(D)+1: Error completion device (bit)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
n1	—	○	○	—				○	—
(S)	—	○	○	—				—	—
n2	○	○	○	—				○	—
(D)	○ (Other than T, ST, C)*1	—	○*2	—				—	—

*1 Local devices cannot be used.
*2 The file register specified for each program cannot be used.

Processing details

- Writes device data for the number of points specified at n2 of the device (S) to the write offset, which is specified for n1, of the device data storage file in the standard ROM. n1 is the offset from the head of device data storage file and specified by word offset (in units of 16-bit words).



- When the SP.DEVST instruction is used, the device data storage file needs to be set. Refer to the User's Manual (Function Description/Program Fundamentals) of the CPU module used.
- Since the completion device ((D)+0) in the standard ROM automatically turns ON at execution of the END instruction, which detects the completion of this instruction, and turns OFF with the END instruction of next scan, it is used as an execution completion flag of this instruction.
- When this instruction is completed in error, the error completion device ((D)+1) turns ON/OFF at the same timing with the completion device ((D)+0).This device is used as an error completion flag of this instruction.
- SM721 turns ON during execution of this instruction. When SM721 has already turned ON, this instruction cannot be executed. (If an attempt is made, no processing is performed.)
- When an error is detected at execution of this instruction, the completion device ((D)+0), error completion device ((D)+1) and SM721 do not turn ON.

Operation error

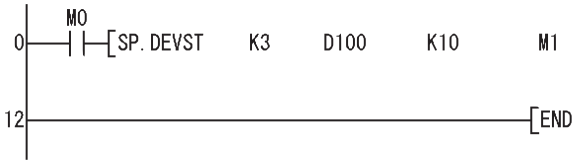
- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
1610	The write frequency into standard ROM exceeds 100,000 times during execution of SP.DEVST instruction.	—	—	—	—	○	○
2410	The device data storage file is not set at "PLC file" of PLC parameter on.	—	—	—	—	○	○
4100	The range of the write offset specified in n1 is out of the device data storage file range. The number of n2 points from the write offset specified at n1 is out of the device data storage file range. The value specified in n2 is 0 or a negative value.	—	—	—	—	○	○
4101	The range of the device specified by (S) exceeds the range from (S) to (S)+n2 (including (S)). The device specified by (D) exceeds the range of the corresponding device.	—	—	—	—	○	○
4113	The value out of the given range is specified in the SD695 during execution of SP.DEVST instruction. The write frequency into standard ROM at the current day exceeds the value specified by the SD695 during execution of SP.DEVST instruction.	—	—	—	—	○	○

Program example

- The program which writes the ten points of data from D100 to the device data storage file in the standard ROM when M0 turns ON.

[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	M0
1	SP. DEVST	K3 D100 K10 M1
12	END	

Precautions

- The value written to the standard ROM is the value at execution of this instruction.
- The standard ROM write count index (SD687 and SD688) is increased by the execution of the SP.DEVST instruction. If the standard ROM write count index exceeds one hundred thousand times, FLASH ROM ERROR (error code: 1610) occurs.
- To prevent the increase in the number of writes to ROM due to the unconsidered execution of instructions, set SD695 (Specification of the number of writes to standard ROM) to limit the number of writes in a day. The limitation on the number of writes is 36 times (default value). Set the number of writes in SD695 as needed. When the number of writes exceeded the set number, OPERATION ERROR (error code: 4113) will occur. The number of executions of write instructions to the data memory in a day is initialized to 0 at the following timing.
  - When the CPU module is powered off and on, or reset
  - When the date (year, month, day) of clock data is changed due to temporal progress
  - When clock data (year, month, day) in the CPU module is changed due to a function that changes clock data
- Data is written to the device data storage file when the END instruction is executed. Data is written to the device data storage file when the END instruction is executed right after the execution of the SP.DEVST instruction. Therefore, writing to the device data storage file may take multiple scans depending on the number of write points. Determine if data has been written by checking a completion device.

Reading data from standard ROM

S(P).DEVLD

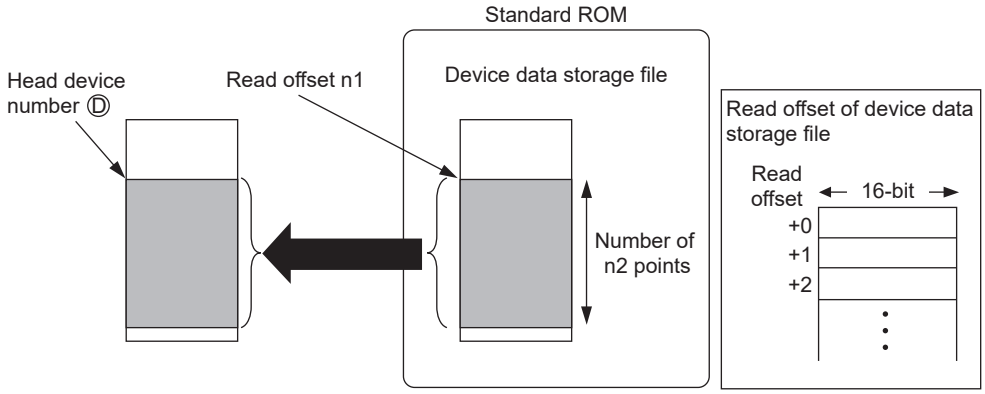
Basic | High performance | Process | Redundant | Universal | LCPU

n1: Read offset of the device data storage file (specified in units of 16-bit words) (BIN 32-bit)
(D): Head device number read from the standard ROM (device name)
n2: The number of reading points (BIN 16-bit)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
n1	—	○		—				○	—
(D)	—	○		—				—	—
n2	○	○		—				○	—

Processing details

- Reads device data for the number of points specified at n2 from the read offset, which is specified for n1, of the device data storage file in the standard ROM, and stores the data to the device specified for (D). n1 is the offset from the head of device data storage file and specified by word offset (in units of 16-bit words).



- When the SP.DEVST instruction is used, the device data storage file needs to be set. Refer to the User's Manual (Function Description/Program Fundamentals) of the CPU module used.

Operation error

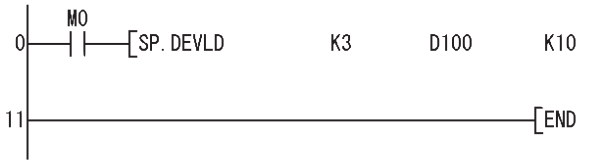
- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
2410	The device data storage file is not set at "PLC file" of PLC parameter.	—	—	—	—	○	○
4100	The address specified in n1 is out of the standard ROM range. The address of n2, specified in n1, is out of the standard ROM range. The value specified in n2 is 0 or a negative value.	—	—	—	—	○	○
4101	The range of n2 exceeds that of the device specified in (D).	—	—	—	—	○	○

Program example

- The program which reads the ten points of data from D100 to the device data storage file in the standard ROM when M0 turns ON.

[Ladder Mode]



[List Mode]

Step	Instruction	Device		
0	LD	M0		
1	SP. DEVLD	K3	D100	K10
11	END			

Loading program from memory card

PLOADP

Basic | High performance | Process | Redundant | Universal | LCPU

(S): Drive No. storing the program to be loaded, character string data of the file name, or head number of the devices storing the character string data (BIN 16 bits)*1

(D): Device that turns ON for 1 scan by the instruction completion (bits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant $	Others
	Bit	Word		Bit	Word				
(S)	—	○		—				○	—
(D)	△*2	—		—				—	—

*1 Designated as "<Drive No.>:<File Name>". Example 1: MAIN

*2 Local devices cannot be used.

Processing details

- The program stored in the memory card or standard ROM is transferred to the program memory (drive 0). If the transferred program is not registered to the program setting of the PLC parameter window, its program setting in the CPU module is set to the standby type. At this time, the program setting of the PLC parameter dialog box does not change. (To transfer a program with the PLOADP instruction, a continuous free space is required in the program memory.)
- The program added using the PLOADP instruction is assigned the lowest number among the unused program Nos. (To assign a program number manually, store the program number to be assigned in SD720.) The following example assumes that "MAIN6" is added by the PLOADP instruction.
  - When the program Nos. have been set consecutively, the new program is added at the end of the preset program Nos. When programs No. 1 to 5 have been set, the new program is added as program No. 6.

Program No.	Program name
1	MAIN1
2	MAIN2
3	MAIN3
4	MAIN4
5	MAIN5

Adds "MAIN6" by the PLOADP instruction.

Program No.	Program name
1	MAIN1
2	MAIN2
3	MAIN3
4	MAIN4
5	MAIN5
6	MAIN6

←Added at the end.

  - When there are multiple open program Nos., the program designated by the PLOADP instruction is added to the lowest number among them to be added. (The open program Nos. are made when programs are deleted by the PUNLOADP instruction.) When programs No. 2 and 4 are open, the new program is added as program No. 2.

Program No.	Program name
1	MAIN1
2	Empty
3	MAIN3
4	Empty
5	MAIN5

Adds "MAIN6" by the PLOADP instruction.

Program No.	Program name
1	MAIN1
2	MAIN6
3	MAIN3
4	Empty
5	MAIN5

←Added to the smallest program number which is empty.

- Drive Nos.1, 2, and 4 can be specified. (Drive 3 cannot be specified.)
  - Drive 1: Memory card (RAM)
  - Drive 2: Memory card (ROM)
  - Drive 4: Standard ROM
- It is not necessary to designate the extension (.QPG) with the file name.
- The bit device specified by (D) is turned ON during the END processing of the scan where this instruction is completed. The bit device is turned OFF at the next END processing.

• The PLC file settings of the loaded program are set as follows:

■File usage for each program

All the usage of file register, device initial value, comment, and local device of the program transferred by this instruction are set as "Use PLC file setting". However, an error will be returned if both of the conditions below are met when the program is transferred using this instruction.

• Setting is made so that local devices are used in the PLC file setting.
• The number of programs in the program memory exceeds the number of programs set at the parameters.

To use local devices in the program transferred by this instruction, register a dummy program file in the parameter, delete the dummy file with the PUNLOADP instruction, and then load the program with the PLOADP instruction.

■I/O refresh setting

Nothing is set for both input and output for the I/O refresh setting of the program transferred by this instruction.

Operation error

• In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
2401	The file size of the local devices cannot be reserved.	—	○	○	—	—	—
2410	The file name does not exist at the drive number specified in (S). The program file which has the same name as the program file to be loaded already exists.	—	○	○	—	—	—
2413	There is not enough memory to load the specified program in drive 0.	—	○	○	—	—	—
4100	The drive No. specified in (S) is invalid.	—	○	○	—	—	—
4101	The same number of files as that indicated in the table below has been already registered in the program memory. The program No. stored in SD720 is already used, or is larger than the largest program No. shown in the table below.	—	○	○	—	—	—

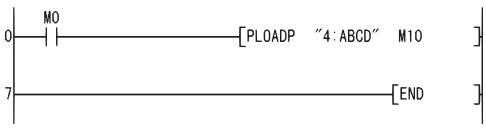
The following table lists the number of files and the maximum program No. of each CPU module.

CPU model name	Program memory (number of files)	Largest program No.
Q02(H)CPU, Q02PHCPU	28	28
Q06HCPU, Q06PHCPU	60	60
Q12HCPU	124	124
Q25HCPU	124	124
Q12PHCPU	124	124
Q25PHCPU	124	124

Program example

• The following program transfers "ABCD.QPG" stored in drive 4 to drive 0 and places the program in standby status when M0 is turned ON.

[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	M0
1	PLOADP	"4:ABCD" M10
7	END	

Precautions

- The PLOADP, PUNLOADP and PSWAPP instructions cannot be executed simultaneously. If two or more of the above instructions are executed simultaneously, the instruction executed later will not be executed. When using the above instructions, provide interlocks manually to avoid simultaneous execution.
- Do not execute this instruction in an interrupt program. (Otherwise, a malfunction may result.)
- To execute the program that was transferred to the program memory with the PLOADP instruction, execute the scan execution type with the PSCAN instruction (☞ Page 728 Program scan execution registration)
- The "PLOADP instruction" and "Write during RUN" processing cannot be executed simultaneously.
  - When a write during RUN request is given during processing of the PLOADP instruction, write during RUN is delayed. Write during RUN is started after the processing of the PLOADP instruction is completed.
  - When the PLOADP instruction is executed during write during RUN, the processing of the PLOADP instruction is delayed. The processing of the PLOADP instruction is started after completion of write during RUN.
- Do not execute "Read from PLC" or "Verify with PLC" and the instruction simultaneously. If the instruction is executed, "Read from PLC" or "Verify with PLC" is not complete normally because the program file status stored in the program memory is changed. If executed, execute "Read from PLC" or "Verify with PLC" again after the instruction completion.

Unloading program from program memory

PUNLOADP

Basic | High performance | Process | Redundant | Universal | LCPU

(S): Character string data of the program file name to be unloaded, or head number of the devices storing the character string data (BIN 16 bits)
(D): Device turned ON for 1 scan on completion of the instruction (bits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant $	Others
	Bit	Word		Bit	Word				
(S)	—	○		—				○	—
(D)	△*1	—		—				—	—

*1 Local devices cannot be used.

Processing details

- The standby program stored in the program memory (drive 0) is deleted from the program memory. (The program set as the "scan execution type" with the PSCAN instruction or the program set as the "low speed execution type" with the PLOW instruction cannot be deleted.)
- The program No. deleted by the PUNLOADP instruction is made "Empty". When programs No. 1 to 5 have been set in the program setting of the PLC parameter dialog box, deleting program No. 2 with this instruction makes program No. 2 open.

Program No.	Program name
1	MAIN1
2	MAIN2
3	MAIN3
4	MAIN4
5	MAIN5

Deletes "MAIN2" by the PUNLOADP instruction.

Program No.	Program name
1	MAIN1
2	Empty
3	MAIN3
4	MAIN4
5	MAIN5

←Program No. 2 is deleted.

- It is not necessary to designate the extension (.QPG) with the file name.
- The bit device specified by (D) is turned ON during the END processing of the scan where this instruction is completed. The bit device is turned OFF at the next END processing.
- When the programmable controller is powered OFF, then ON or the CPU module is reset after execution of the PUNLOADP instruction, the following operation is performed.
  - When boot setting has been made in the PLC parameter dialog box, the program where the boot setting has been made is transferred to the program memory. When the program deleted by the PUNLOADP instruction is not to be executed, delete the corresponding program name from the boot setting and program setting of the PLC parameter dialog box.
  - When boot setting has not been made in the PLC parameter dialog box, "FILE SET ERROR (error code: 2400)" occurs. When the program deleted by the PUNLOADP instruction is not to be executed, delete the corresponding program name from the program setting of the PLC parameter dialog box. When the program deleted by the PUNLOADP instruction is to be executed again, write the corresponding program to the CPU module.

Operation error

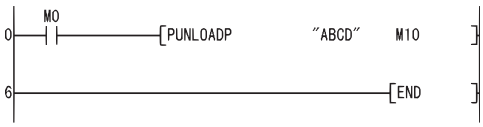
- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
2410	The file name specified in (S) does not exist.	—	○	○	—	—	—
4101	The program specified in (S) is not in standby status or is being executed.	—	○	○	—	—	—

Program example

- The following program deletes "ABCD.QPG" stored in drive 0 from the memory when M0 turns from OFF to ON.

[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	M0
1	PUNLOADP	"ABCD" M10
6	END	

Precautions

- The PLOADP, PUNLOADP and PSWAPP instructions cannot be executed simultaneously. If two or more of the above instructions are executed simultaneously, the instruction executed later will not be executed. When using the above instructions, provide interlocks manually to avoid simultaneous execution.
- Do not execute this instruction in an interrupt program. (Otherwise, a malfunction may result.)
- The program to be deleted from the program memory by this instruction should be set to the "standby execution type" with the PSTOP instruction beforehand. (☞ Page 725 Program standby)
- The "PUNLOADP instruction" and "write during RUN" processing cannot be executed simultaneously.
  - When a write during RUN request is given during processing of the PUNLOADP instruction, write during RUN is delayed. Write during RUN is started after the processing of the PUNLOADP instruction is completed.
  - When the PUNLOADP instruction is executed during write during RUN, the processing of the PUNLOADP instruction is delayed. The processing of the PUNLOADP instruction is started after completion of write during RUN.
- Do not execute "Read from PLC" or "Verify with PLC" and the instruction simultaneously. If the instruction is executed, "Read from PLC" or "Verify with PLC" is not complete normally because the program file status stored in the program memory is changed. If executed, execute "Read from PLC" or "Verify with PLC" again after the instruction completion.

PSWAPP

Basic | High performance | Process | Redundant | Universal | LCPU

(S1): Character string data of the file name of the program to be unloaded, or head number of the devices storing the character string data (BIN 16 bits)
(S2): Drive No. storing the program to be loaded, character string data of the file name, or head number of the devices storing the character string data (BIN 16 bits)*1
(D): Device turned ON for 1 scan on completion of the instruction (bits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant $	Others
	Bit	Word		Bit	Word				
(S1)	—	○		—				○	—
(S2)	—	○		—				○	—
(D)	△*2	—		—				—	—

*1 Designated as "<Drive No.>:<File Name>". Example 1: MAIN
*2 Local devices cannot be used.

Processing details

- The standby type program stored in the program memory (drive 0) designated by (S1) is deleted from the program memory, and at the same time, the program stored in the memory card or standard ROM designated by (S2) is transferred to the program memory and placed in standby status. (When the program is transferred to the program memory, the program must have a continuous free space.)
- The program set as the "scan execution type" with the PSCAN instruction or the program set as the "low speed execution type" with the PLOW instruction cannot be deleted.
- The program to be transferred to the program memory by the PSWAPP instruction will have the program No. of the program to be deleted from the program memory. (If there is an open program No. before the program to be deleted from the program memory, the program to be transferred to the program memory will not have the open program No.)
- When program No. 2 is "Empty", the program transferred to the program memory is registered as program No. 3 by the program swapping of program No. 3 with this instruction.

Program No.	Program name
1	MAIN1
2	Empty
3	MAIN3
4	MAIN4
5	MAIN5

Swaps "MAIN3" with "MAIN6" by the PSWAPP instruction.

Program No.	Program name	
1	MAIN1	
2	Empty	
3	MAIN6	←MAIN6 enters
4	MAIN4	
5	MAIN5	

- Drive Nos.1, 2, and 4 can be specified. (Drive 3 cannot be specified.)
  - Drive 1: Memory card (RAM)
  - Drive 2: Memory card (ROM)
  - Drive 4: Standard ROM
- It is not necessary to designate the extension (.QPG) with the file name.
- The bit device specified by (D) is turned ON during the END processing of the scan where this instruction is completed. The bit device is turned OFF at the next END processing.
- When the programmable controller is powered OFF, then ON or the CPU module is reset after execution of the PSWAPP instruction, the following operation is performed.
  - When boot setting has been made in the PLC parameter dialog box, the program where the boot setting has been made is transferred to the program memory. When the program replaced by the PSWAPP instruction is to be executed, change the boot setting and program setting of the PLC parameter dialog box for the corresponding program name.
  - When boot setting has not been made in the PLC parameter dialog box, "FILE SET ERROR (error code: 2400)" occurs. When the program replaced by the PSWAPP instruction is to be executed, change the program setting of the PLC parameter dialog box for the corresponding program name. To execute the program set in the program setting of the PLC parameter dialog box, write the corresponding program to the CPU module again.

- The PLC file settings of the program on which the PSWAPP instruction has been conducted are set as follows:
  - When the boot setting has been made in the PLC parameter, the program where the boot setting has been made is transferred to the program memory. To execute the program replaced by the PSWAPP instruction, change the boot setting and the program setting in the PLC parameter for the corresponding program name.
  - When the boot setting has not been made in the PLC parameter, "FILE SET ERROR (error code: 2400)" occurs. To execute the program replaced by the PSWAPP instruction, change the program setting of the PLC parameter for the corresponding program name. To execute the program set in the program setting of the PLC parameter, write the corresponding program to the CPU module again.

Operation error

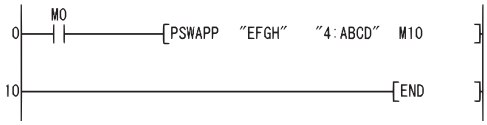
- In any of the following cases, an operation error occurs, the error flag (SM0) turns on, and an error code is stored in SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
2410	The drive No. or the file name specified in (S1) or (S2) does not exist.	—	○	○	—	—	—
2413	There is not enough memory to load the specified program in drive 0.	—	○	○	—	—	—
4100	The drive No. specified in (S1) is invalid.	—	○	○	—	—	—
4101	The program specified by (S1) is not a standby program or is being executed.	—	○	○	—	—	—

Program example

- The following program deletes "EFGH.QPG" stored in drive 0 from the memory, transfers "ABCD.QPG" stored in drive 4 to drive 0, and places the program in standby status when M0 is turned from OFF to ON.

[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	M0
1	PSWAPP	"EFGH" "4:ABCD" M10
10	END	

Precautions

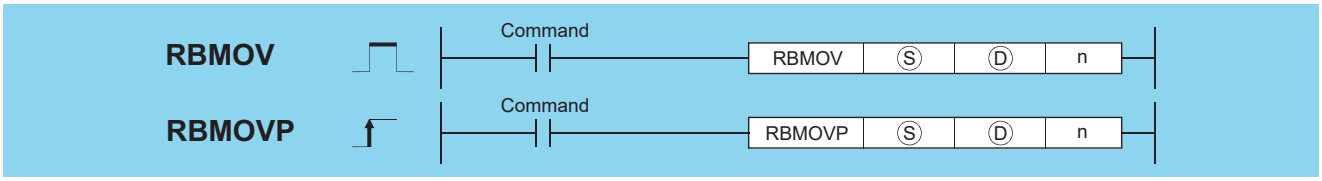
- The PLOADP, PUNLOADP and PSWAPP instructions cannot be executed simultaneously. If two or more of the above instructions are executed simultaneously, the instruction executed later will not be executed. When using the above instructions, provide interlocks manually to avoid simultaneous execution.
- Do not execute this instruction in an interrupt program. (Execution of this instruction in an interrupt program can cause a malfunction.)
- The "PSWAPP instruction" and "write during RUN" processing cannot be executed simultaneously.
  - When a write during RUN request is given during processing of the PSWAPP instruction, write during RUN is delayed. Write during RUN is started after the processing of the PSWAPP instruction is completed.
  - When the PSWAPP instruction is executed during write during RUN, the processing of the PSWAPP instruction is delayed. The processing of the PSWAPP instruction is started after completion of write during RUN.
- Do not execute "Read from PLC" or "Verify with PLC" and the instruction simultaneously. If the instruction is executed, "Read from PLC" or "Verify with PLC" is not complete normally because the program file status stored in the program memory is changed. If executed, execute "Read from PLC" or "Verify with PLC" again after the instruction completion.

High-speed block transfer of file register

RBMOV(P)

Basic | High performance | Process | Redundant | Universal (Ver.) | LCPU

• Universal model QCPU: Models other than Q00UJCPU

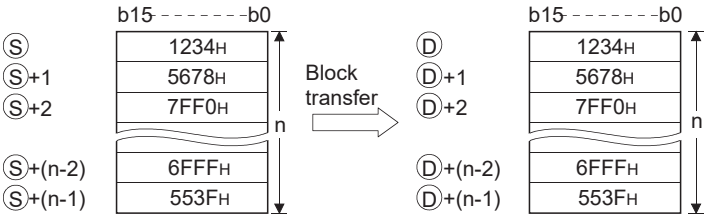


(S): Head number of the devices where the data to be transferred is stored (BIN 16 bits)
(D): Head number of the devices of transfer destination (BIN 16 bits)
n: Number of data to be transferred (BIN 16 bits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(S)	○						—		—
(D)	○						—		—
n	○						○		—

Processing details

• Transfers in batch 16-bit data of n points from the device designated by (S) to location n points from the device designated by (D).



• The transfer is available even if there is an overlap between the source and destination devices. For the transmission to the smaller number of device, the data is transferred from (S). For the transmission to the larger number of device, the data is transferred from (S)+(n-1). However, as shown in the example below, when transferring data from R to ZR, or from ZR to R, the range to be transferred (source) and the range of destination must not overlap. (QnUDVCPU and QnUDPVCPU are excluded.)

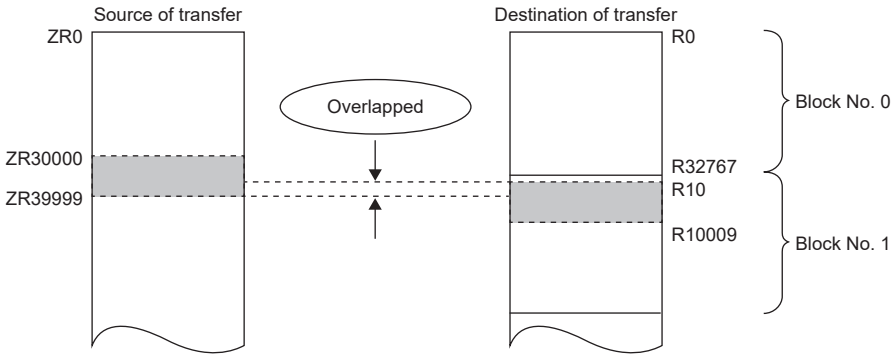
- ZR transfer range ((specified head No. of ZR) to (specified head No. of ZR + the number of transfers -1))
- R transfer range ((specified head No. of R + file register block No. × 32768) to (specified head No. of R + file register block No. × 32768 + the number of transfers -1))

Ex.

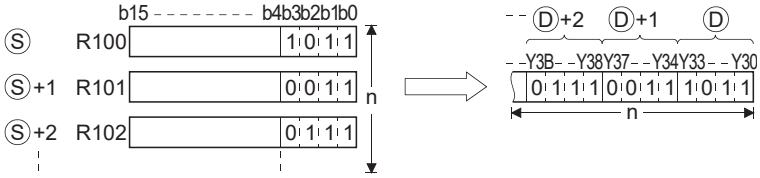
Transfer ranges of ZR and R overlap when transferring 10000 points of data from ZR30000 (source) to R10 (block No.1 of the destination).

- ZR transfer range → (30000) to (30000+10000-1) → (30000) to (39999)
- R transfer range → (10+(1 × 32768)) to (10+(1 × 32768)+10000-1) → (32778) to (42777)

Therefore, the range 32778 to 39999 overlaps.



- When (S) is a word device and (D) is a bit device, the number of bits designated by the bit device digit specification will be transferred. If K1Y30 has been designated by (D), the lower four bits of the word device designated by (S) will become the object.



- The RBMOV (P) instruction is useful to batch transfer a large quantity of file register data with the QnHCPU/QnPHCPU/QnPRHCPU. For the QnUCPU, the processing speed of the RBMOV instruction is equivalent to that of the BMOV instruction. The comparison of processing speed between the RBMOV and BMOV instructions is as follows:

■Transfer from file registers to internal devices/internal devices to file registers

CPU	Instruction	Target memory where file register is stored	1 word		1000 words		10000 words	
			Min.	Max.	Min.	Max.	Min.	Max.
QnHCPU QnPHCPU QnPRHCPU	RBMOV	Standard RAM	20.0μs		91.0μs		775.0μs	
		SRAM card	22.0μs		305.0μs		2900.0μs	
		Flash card*1	22.5μs		405.0μs		3950.0μs	
	BMOV	Standard RAM	7.5μs		76.2μs		720.0μs	
		SRAM card	8.0μs		384.0μs		3900.0μs	
		Flash card*1			418.0μs		4250.0μs	
QnCPU	RBMOV	Standard RAM	45.5μs		215.0μs		1850.0μs	
		SRAM card	49.5μs		540.0μs		5150.0μs	
		Flash card*1						
	BMOV	Standard RAM	17.5μs		177.0μs		1700.0μs	
		SRAM card	18.0μs		500.0μs		5050.0μs	
		Flash card*1			572.0μs		5800.0μs	
Q00UCPU Q01UCPU	RBMOV	Standard RAM	12.2μs	34.9μs	121.5μs	145.1μs	1111.5μs	1135.1μs
		SRAM card*2	—	—	—	—	—	—
		Flash card*2	—	—	—	—	—	—
	BMOV	Standard RAM	7.3μs	13.8μs	116.5μs	124.2μs	1106.5μs	1114.2μs
		SRAM card*2	—	—	—	—	—	—
		Flash card*2	—	—	—	—	—	—
Q02UCPU	RBMOV	Standard RAM	9.4μs	31.3μs	118.5μs	141.3μs	1108.5μs	1131.3μs
		SRAM card	9.4μs	31.4μs	178.5μs	201.3μs	1708.5μs	1731.3μs
		Flash card*1	9.4μs	32.1μs	278.5μs	301.3μs	2708.5μs	2731.3μs
	BMOV	Standard RAM	5μs	11.6μs	114.5μs	122.3μs	1104.5μs	1112.3μs
		SRAM card	5.1μs	11.7μs	174.5μs	182.3μs	1704.5μs	1712.3μs
		Flash card*1	5μs	11.6μs	274.5μs	282.3μs	2704.5μs	2712.3μs
Q03UD(E)CPU	RBMOV	Standard RAM	11.3μs	16.8μs	120.7μs	127.1μs	1110.7μs	1117.1μs
		SRAM card	11.2μs	16.7μs	180.7μs	187.1μs	1710.7μs	1717.1μs
		Flash card*1	11.3μs	16.8μs	280.7μs	287.1μs	2710.7μs	2717.1μs
	BMOV	Standard RAM	4.8μs	6.6μs	114.7μs	117.1μs	1104.7μs	1107.1μs
		SRAM card	4.8μs	6.6μs	174.7μs	177.1μs	1704.7μs	1707.1μs
		Flash card*1	4.8μs	6.5μs	274.7μs	277.1μs	2704.7μs	2707.1μs
Q04UD(E)HCPU Q06UD(E)HCPU Q10UD(E)HCPU Q13UD(E)HCPU Q20UD(E)HCPU Q26UD(E)HCPU Q50UDEHCPU Q100UDEHCPU	RBMOV	Standard RAM	9.2μs	15.1μs	61.0μs	68.6μs	531.0μs	538.6μs
		SRAM card	9.4μs	15.6μs	165.0μs	172.6μs	1576.0μs	1583.6μs
		Flash card*1	9.4μs	15.7μs	260.0μs	267.6μs	2526.0μs	2533.6μs
	BMOV	Standard RAM	4.1μs	5.6μs	56.0μs	58.6μs	526.0μs	528.6μs
		SRAM card	4.5μs	6.1μs	160.0μs	162.6μs	1571.0μs	1573.6μs
		Flash card*1	4.3μs	6.2μs	255.0μs	257.6μs	2521.0μs	2523.6μs
Q03UDVCPU	RBMOV	Standard RAM	3.7μs	21.0μs	80.6μs	89.3μs	822.2μs	831.4μs
		Extended SRAM cassette	3.7μs	21.0μs	102.6μs	118.1μs	1056.4μs	1072.0μs
	BMOV	Standard RAM	1.9μs	7.9μs	79.5μs	82.0μs	820.6μs	823.1μs
		Extended SRAM cassette	1.9μs	7.9μs	102.6μs	107.9μs	1055.5μs	1057.5μs
Q04UDVCPU, Q04UDPVCPU, Q06UDVCPU, Q06UDPVCPU, Q13UDVCPU, Q13UDPVCPU, Q26UDVCPU, Q26UDPVCPU	RBMOV	Standard RAM	3.7μs	21.0μs	42.1μs	57.7μs	413.0μs	428.6μs
		Extended SRAM cassette	3.7μs	21.0μs	102.6μs	118.1μs	1056.4μs	1072.0μs
	BMOV	Standard RAM	1.9μs	7.9μs	41.0μs	47.1μs	411.6μs	417.7μs
		Extended SRAM cassette	1.9μs	7.9μs	102.6μs	107.9μs	1055.4μs	1060.9μs

*1 When file registers are stored in the Flash card, no processing is performed for transfer from internal devices to file registers.
*2 Unusable for the Q00UCPU and Q01UCPU.

■Transfer from file registers to file registers

CPU	Instruction	Target memory where file register is stored	1 word		1000 words		10000 words	
			Min.	Max.	Min.	Max.	Min.	Max.
QnHCPU QnPHCPU QnPRHCPU	RBMOV	Standard RAM	20.0μs		91.0μs		775.0μs	
		SRAM card	22.5μs		545.0μs		5300.0μs	
	BMOV	Standard RAM	7.5μs		77.0μs		720.0μs	
		SRAM card	8.5μs		692.0μs		7050.0μs	
QnCPU	RBMOV	Standard RAM	45.5μs		215.0μs		1850.0μs	
		SRAM card	50.0μs		870.0μs		8350.0μs	
	BMOV	Standard RAM	17.5μs		179.0μs		1700.0μs	
		SRAM card	18.5μs		839.0μs		8600.0μs	
Q00UCPU Q01UCPU	RBMOV	Standard RAM	12.6μs	35.3μs	232.5μs	256.1μs	2211.5μs	2235.1μs
		SRAM card*1	—	—	—	—	—	—
	BMOV	Standard RAM	7.7μs	14.2μs	227.5μs	234.2μs	2206.5μs	2214.2μs
		SRAM card*1	—	—	—	—	—	—
Q02UCPU	RBMOV	Standard RAM	9.6μs	31.5μs	228.5μs	252.3μs	2208.5μs	2231.3μs
		SRAM card	9.6μs	31.5μs	378.5μs	401.3μs	3708.5μs	3731.3μs
	BMOV	Standard RAM	5.2μs	11.8μs	224.5μs	232.3μs	2204.5μs	2212.3μs
		SRAM card	5.2μs	11.8μs	374.5μs	382.3μs	3704.5μs	3712.3μs
Q03UD(E)CPU	RBMOV	Standard RAM	11.2μs	16.7μs	230.7μs	237.1μs	2210.7μs	2217.1μs
		SRAM card	11.6μs	16.7μs	380.7μs	387.1μs	3710.7μs	3717.1μs
	BMOV	Standard RAM	4.9μs	6.7μs	224.7μs	227.1μs	2204.7μs	2207.1μs
		SRAM card	5.2μs	6.7μs	374.7μs	377.1μs	3704.7μs	3707.1μs
Q04UD(E)HCPU Q06UD(E)HCPU Q10UD(E)HCPU Q13UD(E)HCPU Q20UD(E)HCPU Q26UD(E)HCPU Q50UDEHCPU Q100UDEHCPU	RBMOV	Standard RAM	9.3μs	15.5μs	118.0μs	124.6μs	1102.0μs	1107.6μs
		SRAM card	9.7μs	15.5μs	365.0μs	371.6μs	3571.0μs	3578.6μs
	BMOV	Standard RAM	4.3μs	6.2μs	113.0μs	115.6μs	1096.0μs	1098.6μs
		SRAM card	4.5μs	6.1μs	360.0μs	362.6μs	3566.0μs	3568.6μs
Q03UDVCPU	RBMOV	Standard RAM	3.7μs	20.7μs	162.0μs	171.2μs	1637.7μs	1646.4μs
		Extended SRAM cassette	3.7μs	20.7μs	216.7μs	232.1μs	2197.4μs	2212.5μs
	BMOV	Standard RAM	1.9μs	8.0μs	161.1μs	163.7μs	1636.2μs	1638.8μs
		Extended SRAM cassette	1.9μs	8.3μs	216.4μs	221.7μs	2197.4μs	2201.7μs
Q04UDVCPU, Q04UDPVCPU, Q06UDVCPU, Q06UDPVCPU, Q13UDVCPU, Q13UDPVCPU, Q26UDVCPU, Q26UDPVCPU	RBMOV	Standard RAM	3.5μs	20.7μs	84.6μs	99.5μs	836.3μs	851.7μs
		Extended SRAM cassette	3.6μs	20.7μs	216.7μs	232.1μs	2197.4μs	2212.5μs
	BMOV	Standard RAM	1.8μs	8.0μs	83.1μs	89.0μs	835.0μs	840.9μs
		Extended SRAM cassette	1.8μs	8.3μs	216.4μs	221.7μs	2197.4μs	2201.7μs

*1 Unusable for the Q00UCPU and Q01UCPU.

Operation error

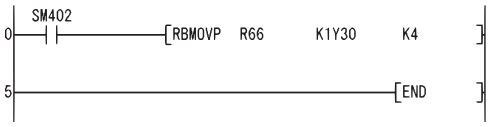
• In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4101	The points specified in n exceed those of the corresponding device specified in (S) or (D). The file register is not specified for either (S) or (D).	—	○	○	○	○	—

Program example

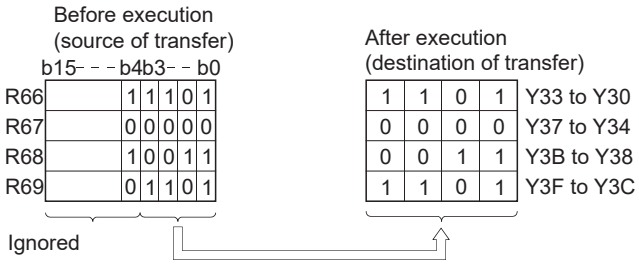
• The following program outputs the lower four bits of data in R66 to R69 to Y30 through Y3F in units of 4 points.

[Ladder Mode]



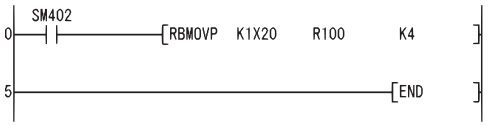
[List Mode]

Step	Instruction	Device		
0	LD	SM402		
1	RBMOVP	R66	K1Y30	K4
5	END			



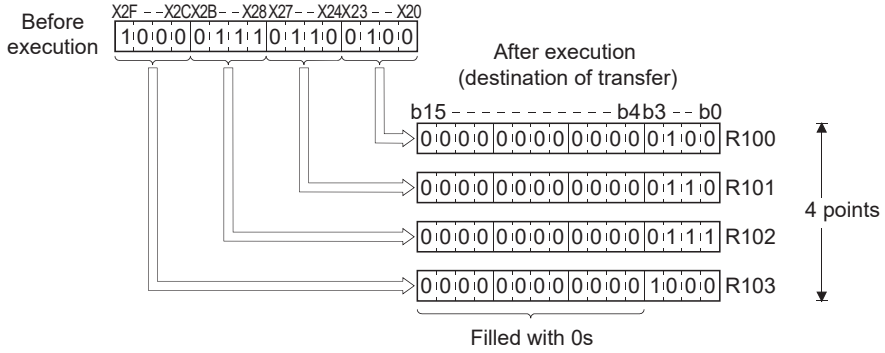
• The following program outputs the data in X20 to X2F to R100 to R103 in units of 4 points.

[Ladder Mode]



[List Mode]

Step	Instruction	Device		
0	LD	SM402		
1	RBMOVP	K1X20	R100	K4
5	END			



User message

UMSG

Basic | High performance | Process | Redundant | Universal | Ver. LCPU

- Built-in Ethernet port LCPU: Supported
- L02SCPU and L02SCPU-P cannot be used.

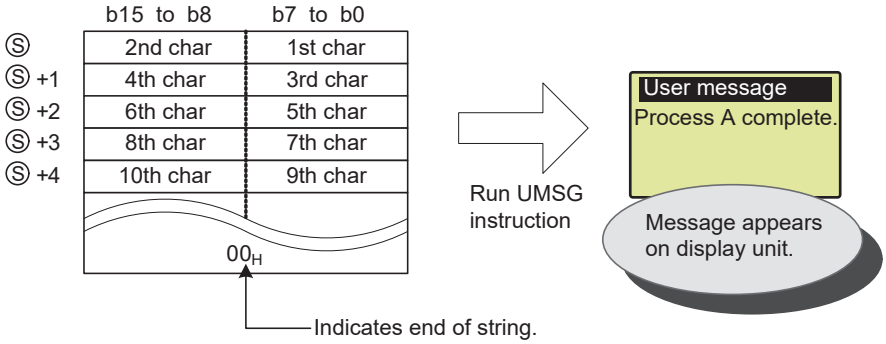
(S): String to display on display unit, or lead number (string) of device storing string to display

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant		Others
	Bit	Word		Bit	Word			K, H	Real string	
(S)	—	○		—					△*1	—

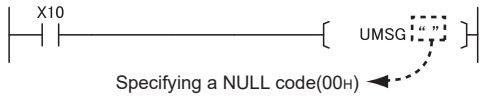
*1 Only strings can be used

Processing details

- The string data specified by (S) is displayed as a user message in the display unit. The character string specified directly by (S) (enclosed in double quotes (" ")) or the character string from the number for the device specified by (S) to the number for the device storing the NULL code "00H" is displayed.



- Strings of up to 128 single-byte characters can be displayed in the display unit.
- The user message is displayed when the UMSG instruction command is rising. If the string is changed while the command is on, then the modified user message will appear in the display unit.
- The string specified by the UMSG instruction is displayed upon END processing. If two or more UMSG instructions are executed, then the last UMSG instruction executed before the END is valid. If two or more programs are running, then the last UMSG instruction to be executed is valid.
- This instruction is not processed if it is run when no display unit is mounted.
- If the "ESC" key on the display unit is pressed while a user message is being displayed, the displayed message will disappear. To display the message again, execute "User Message" from the menu screen on the display unit.
- If a NULL code (00H) is specified as the argument to this instruction, then any message currently being displayed will disappear. The procedure for specifying a NULL code (00H) in the instruction parameter is as follows.



- See the MELSEC-L CPU Module User's Manual (Function Explanation, Program Fundamentals) for details about the display unit.

Operation error

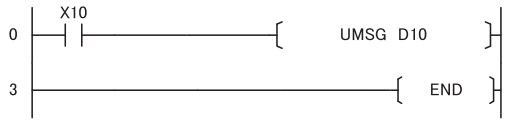
• In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4100	More than 128 characters are specified in the (S) string.	—	—	—	—	—	○
4101	There is no NULL code (00H) within the range of the target device following the device number specified by (S)	—	—	—	—	—	○

Program example

• This program displays the string stored after D10 on the display unit, when X10 is set to "on".

[Ladder Mode]

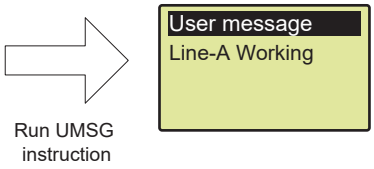


[List Mode]

Step	Instruction	Device
0	LD	X10
1	UMSG	D10
3	END	

[Operation]

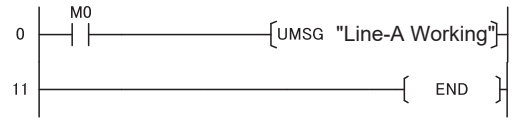
	b15 to b8	b7 to b0
D10	4CH (i)	69H (L)
D11	6EH (e)	65H (n)
D12	2DH (A)	41H (-)
D13	20H (w)	77H ( )
D14	6FH (r)	72H (o)
D15	6BH (i)	69H (k)
D16	6EH (g)	67H (n)
D17	00H	



Run UMSG instruction

• This program displays "Line-A Working" on the display unit when M0 is set to "on".

[Ladder Mode]



[List Mode]

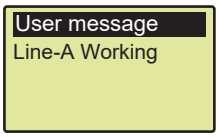
Step	Instruction	Device
0	LD	M0
1	UMSG	"Line-A Working"
11	END	

[Operation]

b15 to b8	b7 to b0
60_H	82_H
89_H	83_H
43_H	83_H
93_H	83_H
40_H	81_H
$5E_H$	89_H
$5D_H$	93_H
86_H	92_H
0000_H	

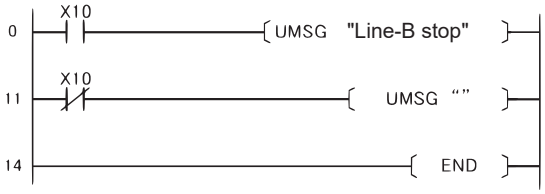
"Line-A Working"

Run UMSG instruction



• This program displays "Line-B stop" on the display unit when X10 is set to "on", and clears the message when X10 is set to "off".

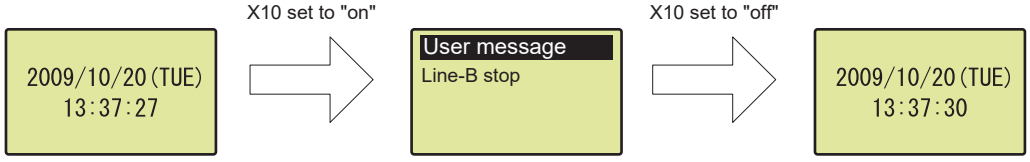
[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	X10
1	UMSG	"Line-B stop"
11	LDI	X10
12	UMSG	""
14	END	

[Operation]



MEMO

8 INSTRUCTIONS FOR DATA LINK

Abbreviation of instruction symbols

In this chapter, instruction names are abbreviated as follows if not specified particularly.

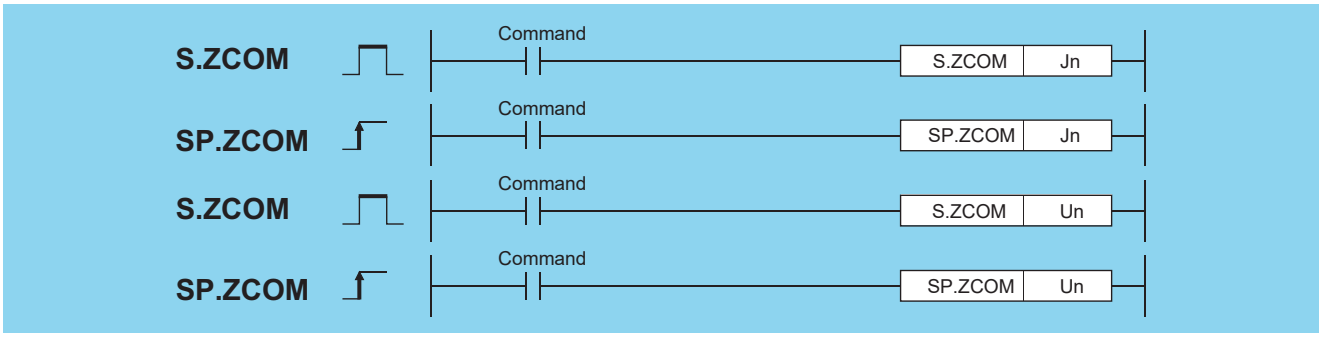
- S(P).ZCOM → ZCOM
- S(P).RTREAD → RTREAD
- S(P).RTWRITE → RTWRITE

8.1 Network Refresh Instructions

Refresh for the designated module

S(P).ZCOM

Basic | High performance | Process | Redundant | Universal | LCPU



Jn: Network No. of host station (BIN 16 bits)
Un: Start I/O number of the host station network module*1 (BIN 16 bits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant	Others J/U
	Bit	Word		Bit	Word				
J/U	—								○

*1 Specified with the upper three digits of the four hexadecimal digits representing the start I/O number.

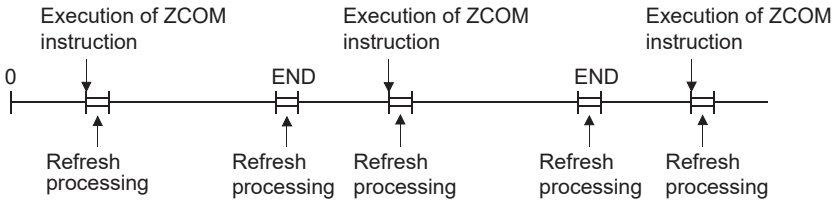
The ZCOM instruction is used to perform refresh at any timing during execution of a sequence program.

The targets of refresh performed by the ZCOM instruction are indicated below.

- Refresh of CC-Link IE Controller Network (when refresh parameters are set) (QCPU only)
- Refresh of CC-Link IE Field Network (when refresh parameters are set) (Universal model QCPU whose serial number (first five digits) is "12012" or later and LCPU whose serial number (first five digits) is "13012" or later only)
- Refresh of MELSECNET/H (when refresh parameters are set) (QCPU only)
- Auto refresh of CC-Link (when refresh device is set)
- Auto refresh of intelligent function module (when auto refresh is set)

Processing details

- When the ZCOM instruction is executed, the CPU module temporarily suspends processing of the sequence program and conducts refresh processing of the network modules designated by Jn/Un. (For LCPU whose serial number (first five digits) is "13011" or earlier, the designation by Jn cannot be made.)

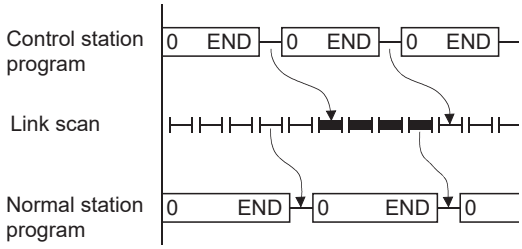


- The ZCOM instruction does not perform the following processing.
  - Communication processing between CPU module and programming tool
  - Monitor processing of other station
  - Read processing of buffer memory of other intelligent function module by serial communication module.
  - Low-speed cyclic data transmission of MELSECNET/H
- The ZCOM instruction can be used as many times as desired in sequence programs. However, note that each execution of a refresh operation will lengthen the sequence program scan time by the amount of time required for the refresh operation.
- Designating "Un" in the argument enables the target designation of the intelligent function as well as the network modules. In this case, the auto refresh is performed for the buffer memory of the intelligent function modules. (It replaces the FROM/TO instructions.)
- Only with the Universal model QCPU and LCPU, interruption of processing is enabled during the execution of the ZCOM instruction. However, when refresh data are used in an interrupted program, the data can split.

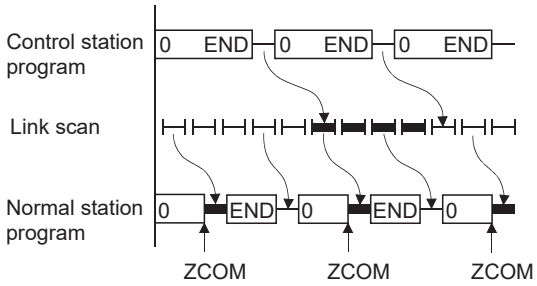
■CC-Link IE Controller Network and MELSECNET/H (PLC to PLC network)

When the scan time for the sequence program of host station is longer than the scan time for the other station, the ZCOM instruction is used to ensure the data reception from the other station.

- Example of data communications when the ZCOM instruction is not used



- Example of data communications when the ZCOM instruction is used

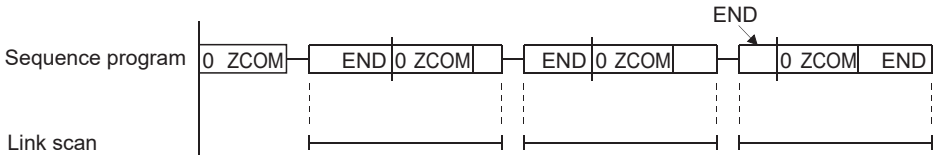


For details on the transmission delay time on CC-Link IE Controller Network and MELSECNET/H (PLC to PLC network), refer to the manuals below:

📖 MELSEC-Q CC-Link IE Controller Network Reference Manual

📖 Q Corresponding MELSECNET/H Network System Reference Manual (PLC to PLC network)

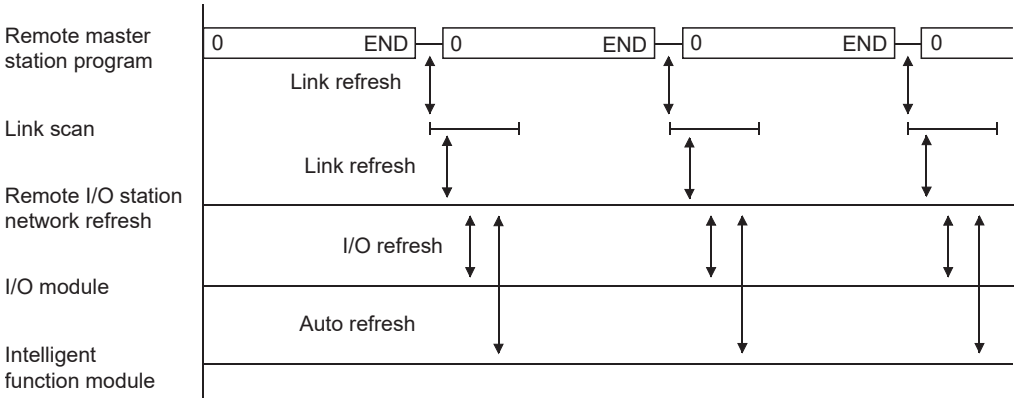
When the link scan time is longer than the sequence program scan time, data communications will not be faster even if the ZCOM instruction is used.



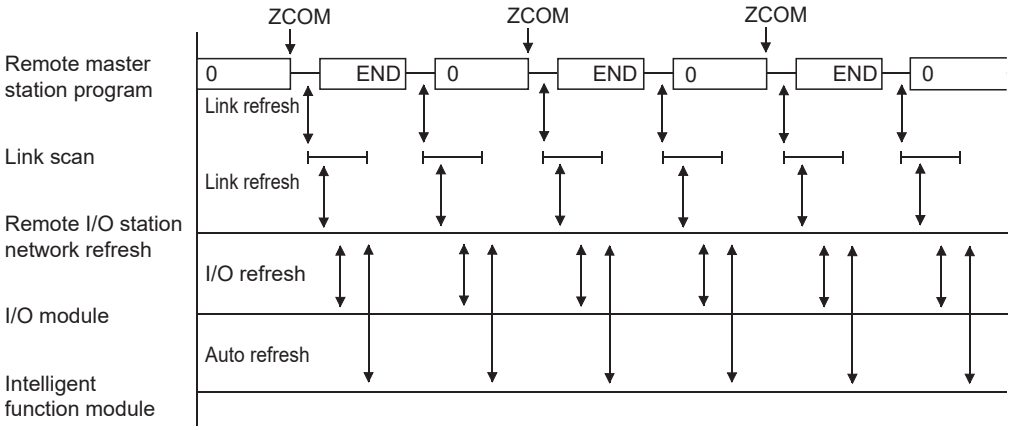
■MELSECNET/H (remote I/O network)

The link refresh of the remote master station is performed by the "END processing" of the CPU module.
Since link scan is performed at completion of link refresh, link scan 'synchronizes' with the program of the CPU module.
When the ZCOM instruction is used at the remote master station, link refresh is performed at the point of ZCOM instruction execution, and link scan is performed at completion of link refresh.
Hence, use of the ZCOM instruction at the remote master station speeds up send/receive processing to/from the remote I/O station.

• When the ZCOM instruction is not used



• When the ZCOM instruction is used



For details on the transmission delay time on MELSECNET/H (remote I/O network), refer to the manual below:
📖 Q Corresponding MELSECNET/H Network System Reference Manual (Remote I/O network)

Point

• The ZCOM instruction cannot be used in a fixed cycle execution type program or interrupt program.
• The Redundant CPU has restrictions on use of the ZCOM instruction.
Refer to the manual below for details.
📖 QnPRHCPU User's Manual (Redundant System)

Operation error

• In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
2111	The module specified with the head I/O number is not a network module or intelligent function module.	○	○	○	○	—	—
4102	The specified network number is not connected to the host station.	○	○	○	○	○	○*1
	The module specified with the head I/O number is not a network module or intelligent function module.	—	—	—	—	○	○

*1 This error applies to modules whose first five digits of the serial number is "13012" or later.

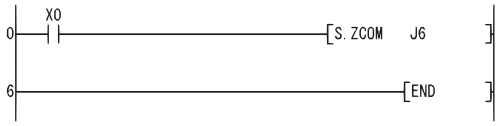
Point

To perform only communication with external devices, use the COM instruction (☞ Page 489 Refresh, Page 491 Select refresh (COM)).

Program example

• The following program conducts a link refresh for the network module of network No. 6 while X0 is ON.

[Ladder Mode]

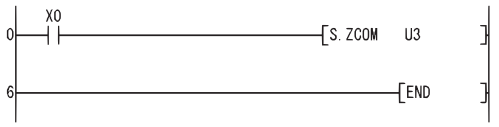


[List Mode]

Step	Instruction	Device
0	LD	X0
1	S. ZCOM	J6
6	END	

• The following program conducts a link refresh for the network module mounted to the position whose head I/O number is a X/Y30 to X/Y4F while X0 is ON.

[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	X0
1	S. ZCOM	U3
6	END	

8.2 Reading/Writing Routing Information

Reading routing information

S(P).RTREAD

Basic (×) | High performance | Process | Redundant | Universal | LCPU (Ver.)

• LCPU: The serial number (first five digits) is "13012" or later.

n: Transfer destination network No. (1 to 239) (BIN 16 bits)
(D): Head number of the devices that stores the read data (Device name)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
n	○	○		—				○	—
(D)	—	○		—				—	—

Processing details

- Reads data from transfer destination network number specified by n, using routing information set by the routing parameters, and stores it into the area starting from (D).
- If no data for the transfer destination network number specified by n is set at the routing parameters, stores 0 into the area starting from (D).
- The contents of the data stored in the area starting from (D) is as indicated below.

(Individual data ranges)

(D)+0	Relay network number	(1 to 239)
+1	Relay station number	See the table below.
+2	Dummy	

[Specification range of relay station number]

Network type	Specification range
MELSECNET/H	1 to 64
CC-Link IE Controller Network	1 to 120
CC-Link IE Field Network	Master station: Fixed at 125. (The fixed value is stored.) Local station: 1 to 120 (A station number is stored.)

Operation error

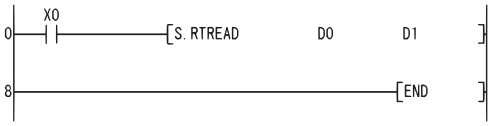
- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4004	When the device which cannot be used the argument is specified	—	○	○	○	○	○
4100	The value in n is the value other than 1 to 239.	—	○	○	○	○	○
4101	The device specified for n and (D) exceeds the range of the corresponding device.	—	—	—	—	○	○

Program example

• The following program reads the routing information for the network number specified by D0 when X0 is turned ON.

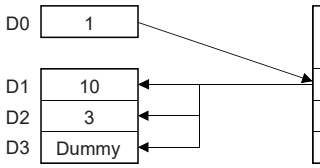
[Ladder Mode]



[List Mode]

Step	Instruction	Device	
0	LD	X0	
1	S. RTREAD	D0	D1
8	END		

[Operation]



[Routing parameter setting]

Transfer destination network number	Relay network number	Relay station number
1	10	3
2	10	2
3	10	1

Registering routing information

S(P).RTWRITE

Basic (×) | High performance | Process | Redundant | Universal | LCPU (Ver.)

• LCPU: The serial number (first five digits) is "13012" or later.

n: Transfer destination network No. (1 to 239) (BIN 16 bits)
(S): Head number of the devices where the data to be written is stored (Device name)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
n	○	○		—				○	—
(S)	—	○		—				—	—

Processing details

• Registers routing information of (S) or later in the area for the transfer destination network number specified by n in the routing parameters.

CPU module	Registration number
• High Performance model QCPU • Process CPU • Redundant CPU • Universal model QCPU whose serial number (first five digits) is "14111" or earlier • High-speed Universal model QCPU whose serial number (first five digits) is "15042" or earlier • Universal model Process CPU whose serial number (first five digits) is "15071" or earlier • LCPU	Up to 64 modules
• The Universal model QCPU whose serial number (first five digits) is "14112" or later, (except the High-speed Universal model QCPU and Universal model Process CPU) • High-speed Universal model QCPU whose serial number (first five digits) is "15043" or later • Universal model Process CPU whose serial number (first five digits) is "15072" or later	Up to 238 modules

• The following shows the contents of data to be set at (S) or later.

(Individual data ranges)

(S) +0	Relay network number
+1	Relay station number
+2	Dummy

See the table below.

[Specification range of relay station number]

Network type	Specification range
MELSECNET/H	1 to 64
CC-Link IE Controller Network	1 to 120
CC-Link IE Field Network	Master station: Fixed at 125. Local station: 1 to 120

• If data for the transfer destination network number specified by n is set in the routing parameters, it is used to update the data in the area starting from (S).
• If data in both (S)+0 and (S)+1 are 0, data for the transfer destination network number specified by n are deleted from the routing parameters.

Operation error

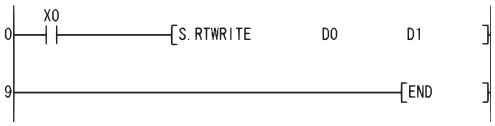
• In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4004	When the device which cannot be used the argument is specified	—	○	○	○	○	○
4100	The value in n is the value other than 1 to 239. The data of (S) or later exceeds each setting range. If the number of routing data registered in the routing parameter of the network parameters plus the number of routing data registered using the RTWRITE instruction exceeds the maximum registration number. If deleting a transfer destination network number, which is not registered in the routing parameter, is specified.	—	○	○	○	○	○
4101	The device specified for n and (D) exceeds the range of the corresponding device.	—	—	—	—	○	○

Program example

• The following program writes the routing information specified by D1 to D3 to the network module of the network number specified by D0 when X0 is turned ON.

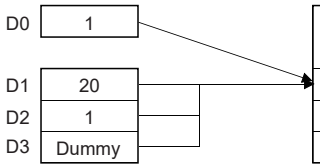
[Ladder Mode]



[List Mode]

Step	Instruction	Device	
0	LD	X0	
1	S.RTWRITE	D0	D1
9	END		

[Operation]



[Routing parameter setting]

Transfer destination network number	Relay network number	Relay station number
1	20	1
2	10	2
3	10	1

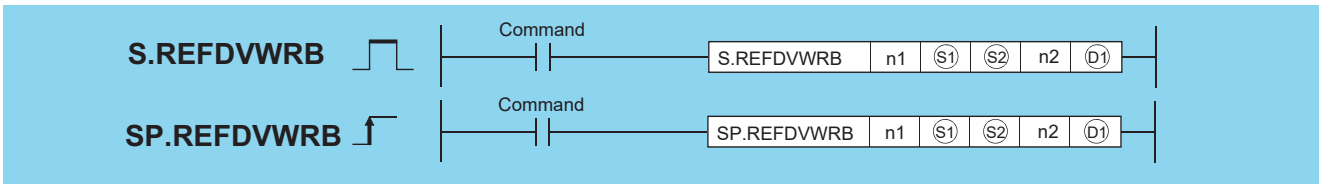
8.3 Refresh Device Write/Read Instructions

Refresh device write (in 1-bit units)

S(P).REFDVWRB

Basic (×) | High performance (×) | Process (×) | Redundant (×) | Universal (Ver.) | LCPU (Ver.)

- QnUD(H)CPU, QnUDE(H)CPU: the serial number (first five digits) is "14072" or later
- QnUDVCPU, QnUDPVCPU: the serial number (first five digits) is "16043" or later.
- Built-in Ethernet port LCPU: Supported
- Q00UJCPU, Q00UCPU, Q01UCPU, Q02UCPU, L02SCPU, and L02SCPU-P cannot be used.



n1: Start I/O number (0H to FEH) (BIN 16-bit)*1 of the master station controlling the station assigned the refresh device which writes data
(S1): Start number of the device stored control data (device name)
(S2): Start number of the device stored write data to the refresh device assigned the device specified in (S1)+0 and (S1)+1 (device name)
n2: Number of write points (1 to 2147483647) (BIN 32-bit)
(D1): Start number of the bit device which turns on for 1 scan by the instruction completion. (D1)+1 also turns on at the error completion (bit).

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
n1	—	○	○	—				○	—
(S1)	—	○	○	—				—	—
(S2)	△*2	—	—	—				—	—
n2	—	○	○	—				○	—
(D1)	△*2	—	—	—				—	—

*1 The first 3 digits of the hexadecimal 4 digits which represent the start I/O number.
*2 Local devices and the devices designated for individual programs cannot be used.

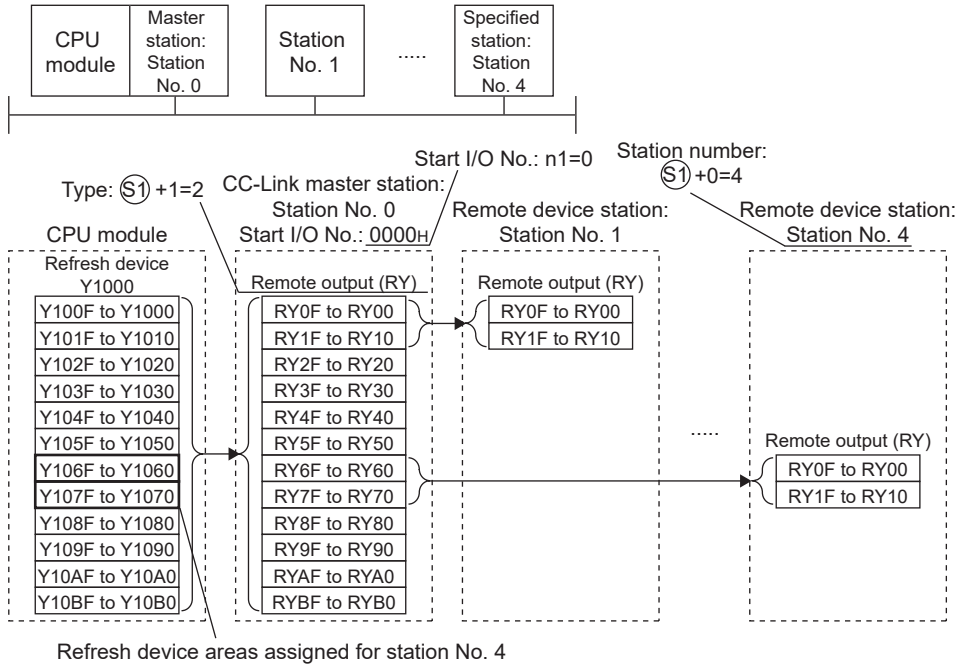
Processing details

- The contents of the data stored in the area starting from (S1) are as indicated below.

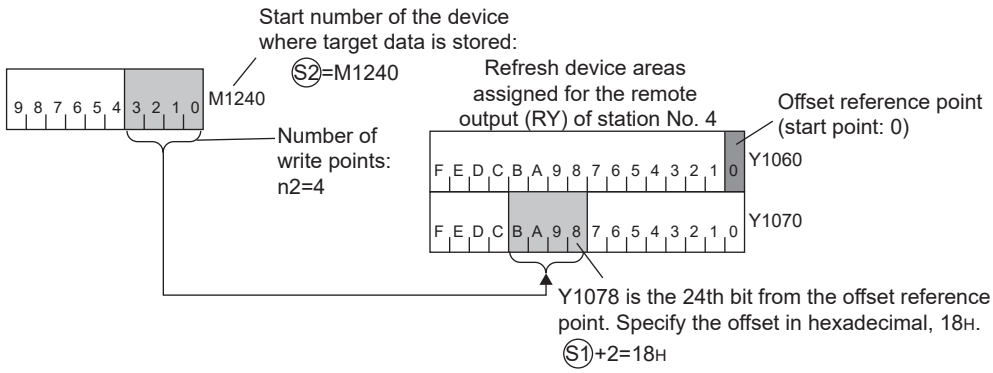
Device	Item	Setting contents	Setting range
(S1)+0	Station number	Station number for the station assigned the refresh device which writes data. When the link special relay (SB) is specified as type of (S1)+1, the setting is disabled.	1 to 120
(S1)+1	Type	Type of the refresh device which writes data • 1: Remote input (RX) • 2: Remote output (RY) • 3: Link special relay (SB)	1 to 3
(S1)+2 (S1)+3	Offset	Offset from the head of the refresh device assigned the device specified in (S1)+0 and (S1)+1	0 to 2147483647

- Instruction execution possibility of an execution type for each program.
  - Enabled: Initial program and scan execution type program
  - Disabled: Fixed scan execution type program and interrupt program
- To write data to a refresh device, the data reflection to the station number specified by the instruction is executed at the timing for the auto refresh.
- Number of points specified in n2 is written from the device specified in (S2) to the offset specified in (S1)+2 of the refresh device assigned for the device specified in (S1)+1 of the target station specified in n1 and (S1)+0.

[Structure]



- At the above configuration, number of points specified in n2 is written from the device specified in (S2) to the offset (Y1078) specified in (S1)+2 of the device assigned for the station number 4.



Point

When a refresh range per station is assigned in transfer settings, specify number of write points so that the range written data from the specified offset is within the range assigned in the same transfer setting. An error occurs if the number of write points over the range assigned in each transfer setting is specified.

- The station type which can and cannot specify with the start I/O number is as follows.

Specification possibility	Station type
Enabled	CC-Link master station, CC-Link master station (compatible with redundant function), CC-Link IE Field Network master station
Disabled	CC-Link local station, CC-Link standby master station, CC-Link IE Field Network local station, CC-Link IE Field Network submaster station

- Because the available range of the station number is 1 to 120, the station number for the master station in n1 cannot be specified to (S1)+0. If the station number is specified, the "OPERATION ERROR" (error code: 4102) occurs.
- SM739 (Refresh device write/read instruction in execution flag) turns on during the instruction execution. When SM739 is on, the following instructions cannot be executed.
  - S(P).REFDVWRB
  - S(P).REFDVWRW
  - S(P).REFDVRDB
  - S(P).REFDVRDW

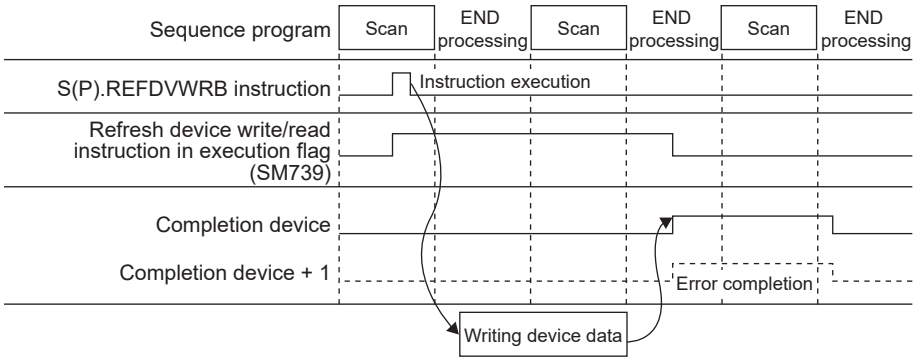
If these instructions are executed, no processing is performed. When an error is detected at the instruction execution (before SM739 turns ON), the completion device ((D1)+0), the completion device ((D1)+1), and SM739 do not turn on.

- The instruction completion can be checked in the completion device ((D1)+0 and (D1)+1).
  - Completion device ((D1)+0)

The device turns on at the END processing in a scan where the instruction is completed and turns off at the next END processing.

  - Completion device ((D1)+1)

The device turns on or off by the status when the instruction is completed.
Normal completion: No change from off
Error completion: The device turns on at the END processing in a scan where the instruction is completed and turns off at the next END processing.



- A module set parameters by the dedicated instruction and a CC-Link module operating by the automatic CC-Link startup cannot be specified with the instruction.
- The write source (points in n2 from (S2)) and write destination (points in n2 from a device specified in control data) are overlapped, data can be written. Write data starting from (S2) when data are written to the smaller device number. Write data starting from (S2)+((n2)-1) when data are written to the larger device number.

Operation error

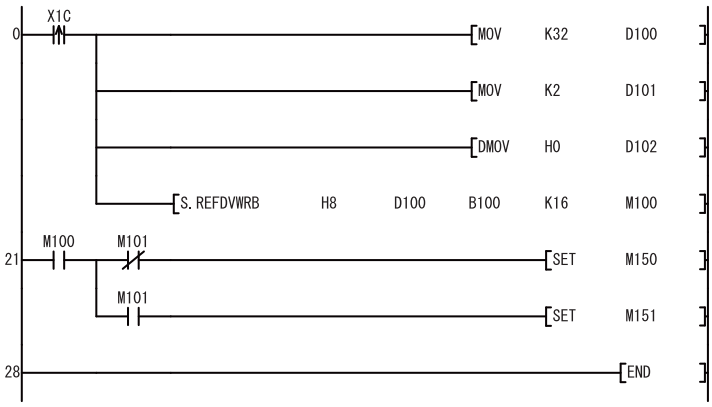
- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4002	The instruction is used for the CPU module that cannot support.	—	—	—	—	○	○
	When the module cannot be specified the start I/O number in n1						
4004	When the device which cannot be specified is specified	—	—	—	—	○	○
4101	When the specified device exceeds the range of the number of device points	—	—	—	—	○	○
	When the start I/O number in n1 is out of the specified range						
	When the device type number in (S1)+1 is out of the specified range						
	When the write offset in (S1)+2 is out of the specified range						
	When the number of write points in n2 is out of the specified range						
	When the number of write points in n2 exceeds the range of the number of device points						
4102	When the station number in (S1)+0 is out of the specified range	—	—	—	—	○	○
	When the station number in (S1)+0 does not exist						
	When the station number in (S1)+0 is the master station specified in n1						
4150	When the start I/O number in n1 is the station type which cannot be specified	—	—	—	—	○	○
	When the start I/O number in n1 does not exist in the network parameter						
4151	When the device in (S1)+1 of the station number specified in (S1)+0 is not assigned the refresh device	—	—	—	—	○	○
	When the write offset in (S1)+2 exceeds the refresh device range assigned for the device in (S1)+1 of the station number specified in (S1)+0						
	When the number of write points in n2 exceeds the assignment range of the setting for one transfer from the write offset in (S1)+2						

Program example

- The following program writes 16 device values in B100 to the head of the refresh device (offset: 0) assigned for the remote output (RY) in the remote I/O station on the station number 32 controlled by the CC-Link master station of the start I/O number 0080H when X1C is turned on.

[Ladder Mode]



[List Mode]

Step	Instruction	Device					
0	LDP	X1C					
1	MOV	K32	D100				
3	MOV	K2	D101				
5	DMOV	H0	D102				
8	S. REFDVWRB		H8	D100	B100	K16	M100
21	LD	M100					
22	MPS						
23	ANI	M101					
24	SET	M150					
25	MPP						
26	AND	M101					
27	SET	M151					
28	END						

Precautions

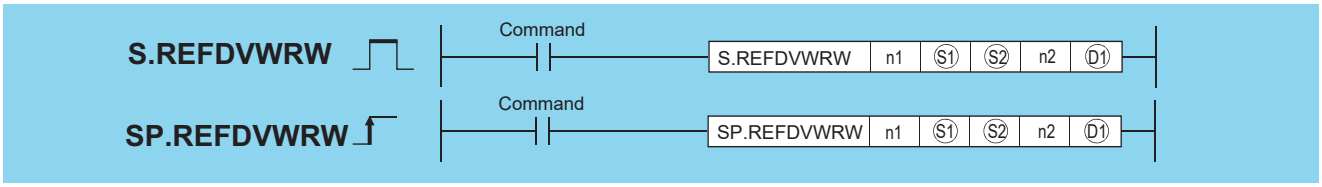
- Do not execute the instruction in an interrupt program. If the instruction is executed, no processing is performed. In addition, the completion device ((D1)+0), the completion device ((D1)+1), and SM739 do not turn on. If the instruction is executed in a fixed scan execution type program, they also do not turn on.
- When the instruction is executed, do not rewrite the device data in (S2) until the completion device turns on.

Refresh device write (in 16-bit units)

S(P).REFDVWRW

Basic | High performance | Process | Redundant | Ver. Universal | Ver. LCPU

- QnUD(H)CPU, QnUDE(H)CPU: the serial number (first five digits) is "14072" or later
- QnUDVCPU, QnUDPVCPU: the serial number (first five digits) is "16043" or later.
- Built-in Ethernet port LCPU: Supported
- Q00UJCPU, Q00UCPU, Q01UCPU, Q02UCPU, L02SCPU, and L02SCPU-P cannot be used.



n1: Start I/O number (0H to FEH) (BIN 16-bit)*1 of the master station controlling the station assigned the refresh device which writes data
(S1): Start number of the device stored control data (device name)
(S2): Start number of the device stored write data to the refresh device assigned the device specified in (S1)+0 and (S1)+1 (device name)
n2: Number of write points (1 to 2147483647) (BIN 32-bit)
(D1): Start number of the bit device which turns on for 1 scan by the instruction completion. (D1)+1 also turns on at the error completion (bit).

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
n1	—	○	○	—				○	—
(S1)	—	○	○	—				—	—
(S2)	—	△*2	△*2	—				—	—
n2	—	○	○	—				○	—
(D1)	△*2	—	—	—				—	—

*1 The first 3 digits of the hexadecimal 4 digits which represent the start I/O number.
*2 Local devices and the devices designated for individual programs cannot be used.

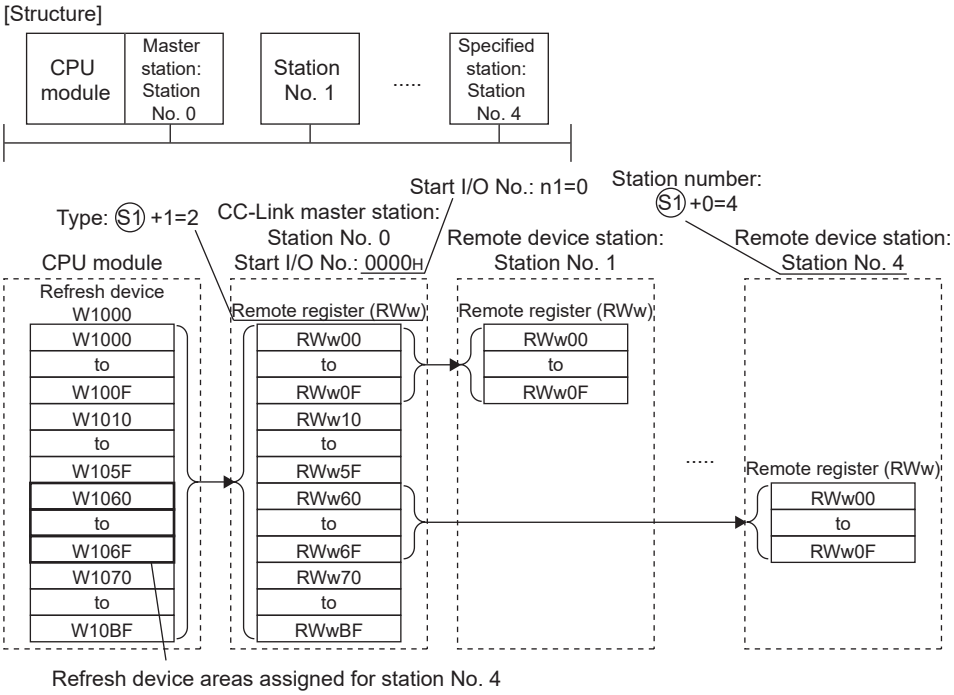
Processing details

- The contents of the data stored in the area starting from (S1) are as indicated below.

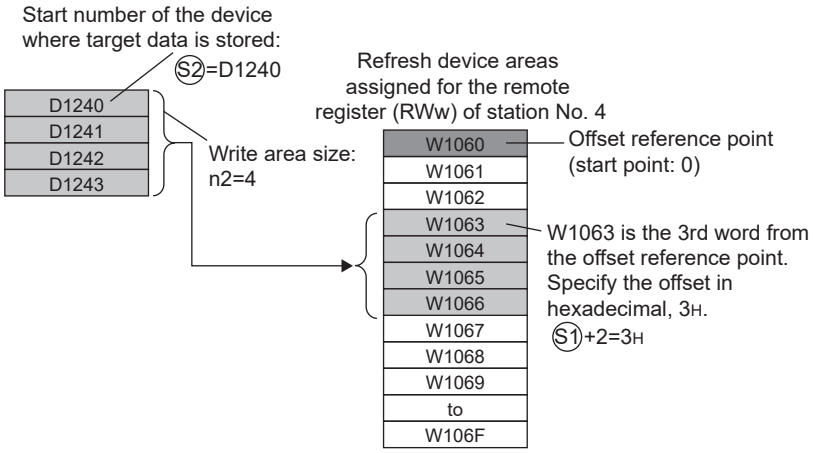
Device	Item	Setting contents	Setting range
(S1)+0	Station number	Station number for the station assigned the refresh device which writes data. When the link special register (SW) is specified as type of (S1)+1, the setting is disabled.	1 to 120
(S1)+1	Type	Type of the refresh device which writes data • 1: Remote register (RWr) • 2: Remote register (RWw) • 3: Link special register (SW)	1 to 3
(S1)+2 (S1)+3	Offset	Offset from the head of the refresh device assigned the device specified in (S1)+0 and (S1)+1	0 to 2147483647

- Instruction execution possibility of an execution type for each program.
  - Enabled: Initial program and scan execution type program
  - Disabled: Fixed scan execution type program and interrupt program
- To write data to a refresh device, the data reflection to the station number specified by the instruction is executed at the timing for the auto refresh.

• Number of points specified in n2 is written from the device specified in (S2) to the offset specified in (S1)+2 of the refresh device assigned for the device specified in (S1)+1 of the target station specified in n1 and (S1)+0.



• At the above configuration, number of points specified in n2 is written from the device specified in (S2) to the offset (W1063) specified in (S1)+2 of the device assigned for the station number 4.



Point

When a refresh range per station is assigned in transfer settings, specify number of write points so that the range written data from the specified offset is within the range assigned in the same transfer setting. An error occurs if the number of write points over the range assigned in each transfer setting is specified.

- The station type which can and cannot specify with the start I/O number is as follows.

Specification possibility	Station type
Enabled	CC-Link master station, CC-Link master station (compatible with redundant function), CC-Link IE Field Network master station
Disabled	CC-Link local station, CC-Link standby master station, CC-Link IE Field Network local station, CC-Link IE Field Network submaster station

- Because the available range of the station number is 1 to 120, the station number for the master station in n1 cannot be specified to (S1)+0. If the station number is specified, the "OPERATION ERROR" (error code: 4102) occurs.
- SM739 (Refresh device write/read instruction in execution flag) turns on during the instruction execution. When SM739 is on, the following instructions cannot be executed.
  - S(P).REFDVWRB
  - S(P).REFDVWRW
  - S(P).REFDVRDB
  - S(P).REFDVRDW

If these instructions are executed, no processing is performed. When an error is detected at the instruction execution (before SM739 turns ON), the completion device ((D1)+0), the completion device ((D1)+1), and SM739 do not turn on.

- The instruction completion can be checked in the completion device ((D1)+0 and (D1)+1).
  - Completion device ((D1)+0)

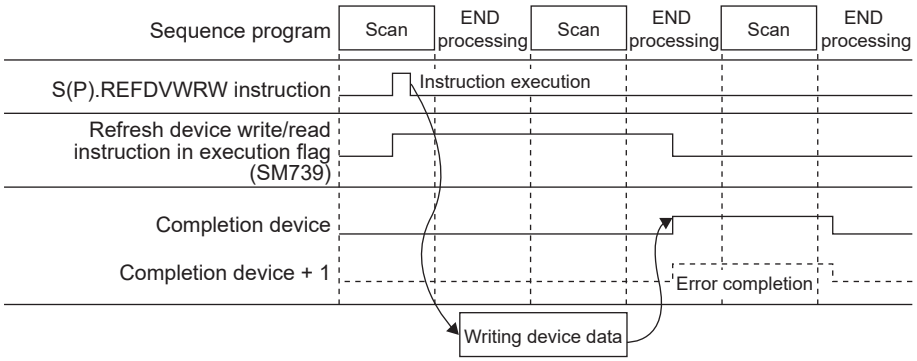
The device turns on at the END processing in a scan where the instruction is completed and turns off at the next END processing.

  - Completion device ((D1)+1)

The device turns on or off by the status when the instruction is completed.

Normal completion: No change from off

Error completion: The device turns on at the END processing in a scan where the instruction is completed and turns off at the next END processing.



- A module set parameters by the dedicated instruction and a CC-Link module operating by the automatic CC-Link startup cannot be specified with the instruction.
- The write source (points in n2 from (S2)) and write destination (points in n2 from a device specified in control data) are overlapped, data can be written. Write data starting from (S2) when data are written to the smaller device number. Write data starting from (S2)+((n2)-1) when data are written to the larger device number.

Operation error

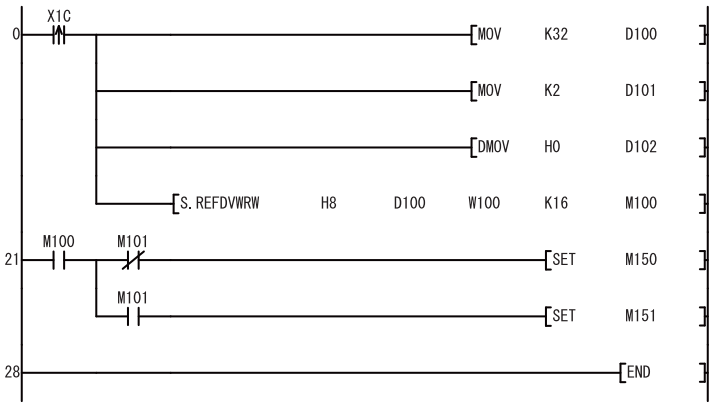
- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4002	The instruction is used for the CPU module that cannot support.	—	—	—	—	○	○
	When the module cannot be specified the start I/O number in n1						
4004	When the device which cannot be specified is specified	—	—	—	—	○	○
4101	When the specified device exceeds the range of the number of device points	—	—	—	—	○	○
	When the start I/O number in n1 is out of the specified range						
	When the device type number in (S1)+1 is out of the specified range						
	When the write offset in (S1)+2 is out of the specified range						
	When the number of write points in n2 is out of the specified range						
	When the number of write points in n2 exceeds the range of the number of device points						
4102	When the station number in (S1)+0 is out of the specified range	—	—	—	—	○	○
	When the station number in (S1)+0 does not exist						
	When the station number in (S1)+0 is the master station specified in n1						
4150	When the start I/O number in n1 is the station type which cannot be specified	—	—	—	—	○	○
	When the start I/O number in n1 does not exist in the network parameter						
4151	When the device in (S1)+1 of the station number specified in (S1)+0 is not assigned the refresh device	—	—	—	—	○	○
	When the write offset in (S1)+2 exceeds the refresh device range assigned for the device in (S1)+1 of the station number specified in (S1)+0						
	When the number of write points in n2 exceeds the assignment range of the setting for one transfer from the write offset in (S1)+2						

Program example

- The following program writes 16 device values in W100 to the head of the refresh device (offset: 0) assigned for the remote register (RWw) in the remote device station on the station number 32 controlled by the CC-Link master station of the start I/O number 0080H when X1C is turned on.

[Ladder Mode]



[List Mode]

Step	Instruction	Device					
0	LDP	X1C					
1	MOV	K32	D100				
3	MOV	K2	D101				
5	DMOV	H0	D102				
8	S. REFDVWRW		H8	D100	W100	K16	M100
21	LD	M100					
22	MPS						
23	ANI	M101					
24	SET	M150					
25	MPP						
26	AND	M101					
27	SET	M151					
28	END						

Precautions

- Do not execute the instruction in an interrupt program. If the instruction is executed, no processing is performed. In addition, the completion device ((D1)+0), the completion device ((D1)+1), and SM739 do not turn on. If the instruction is executed in a fixed scan execution type program, they also do not turn on.
- When the instruction is executed, do not rewrite the device data in (S2) until the completion device turns on.
- Specifying digit for the bit device can be used only when the following conditions are met.
  - Digit specification: K4
  - Head of device: multiple of 16

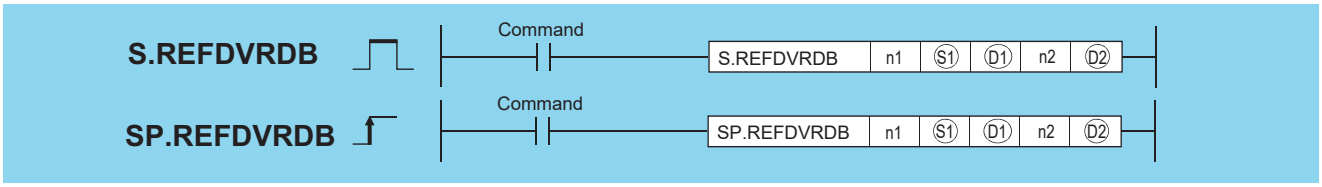
When the above conditions are not met, INSTRCT CODE ERR. (error code: 4004) will occur.

Refresh device read (in 1-bit units)

S(P).REFDVRDB

Basic | High performance | Process | Redundant | Universal (Ver.) | LCPU (Ver.)

- QnUD(H)CPU, QnUDE(H)CPU: the serial number (first five digits) is "14072" or later
- QnUDVCPU, QnUDPVCPU: the serial number (first five digits) is "16043" or later.
- Built-in Ethernet port LCPU: Supported
- Q00UJCPU, Q00UCPU, Q01UCPU, Q02UCPU, L02SCPU, and L02SCPU-P cannot be used.



n1: Start I/O number (0H to FEH) (BIN 16-bit)*1 of the master station controlling the station assigned the refresh device which reads data
(S1): Start number of the device stored control data (device name)
(D1): Start number of the device stored read data from the refresh device assigned the device specified in (S1)+0 and (S1)+1 (device name)
n2: Number of read points (1 to 2147483647) (BIN 32-bit)
(D2): Start number of the bit device which turns on for 1 scan by the instruction completion. (D1)+1 also turns on at the error completion (bit).

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
n1	—	○	○	—				○	—
(S1)	—	○	○	—				—	—
(D1)	△*2	—	—	—				—	—
n2	—	○	○	—				○	—
(D2)	△*2	—	—	—				—	—

*1 The first 3 digits of the hexadecimal 4 digits which represent the start I/O number.
*2 Local devices and the devices designated for individual programs cannot be used.

Processing details

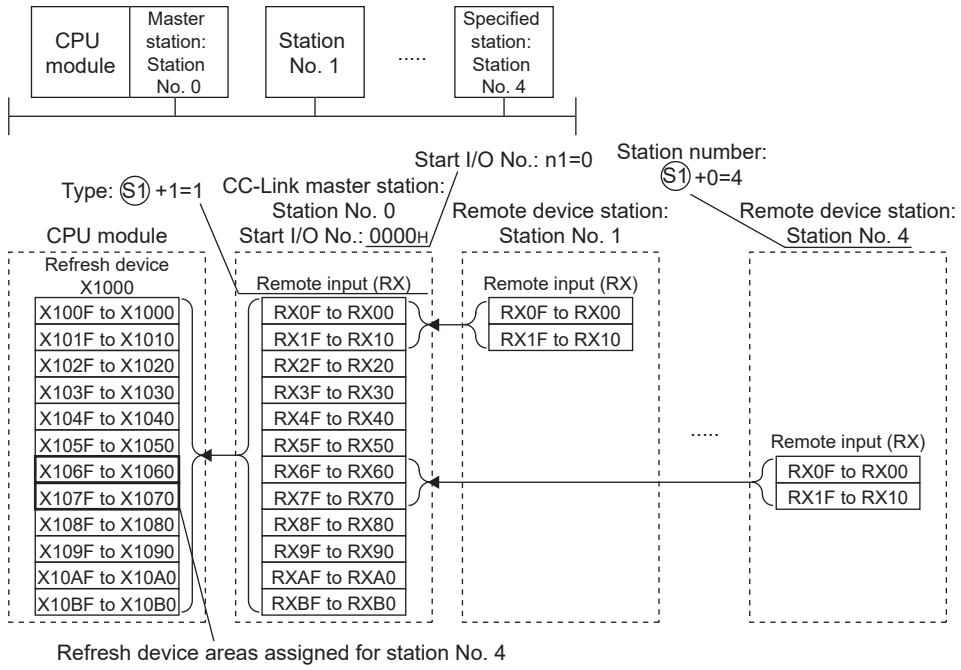
- The contents of the data stored in the area starting from (S1) are as indicated below.

Device	Item	Setting contents	Setting range
(S1)+0	Station number	Station number for the station assigned the refresh device which reads data. When the link special relay (SB) is specified as type of (S1)+1, the setting is disabled.	1 to 120
(S1)+1	Type	Type of the refresh device which reads data • 1: Remote input (RX) • 2: Remote output (RY) • 3: Link special relay (SB)	1 to 3
(S1)+2 (S1)+3	Offset	Offset from the head of the refresh device assigned the device specified in (S1)+0 and (S1)+1	0 to 2147483647

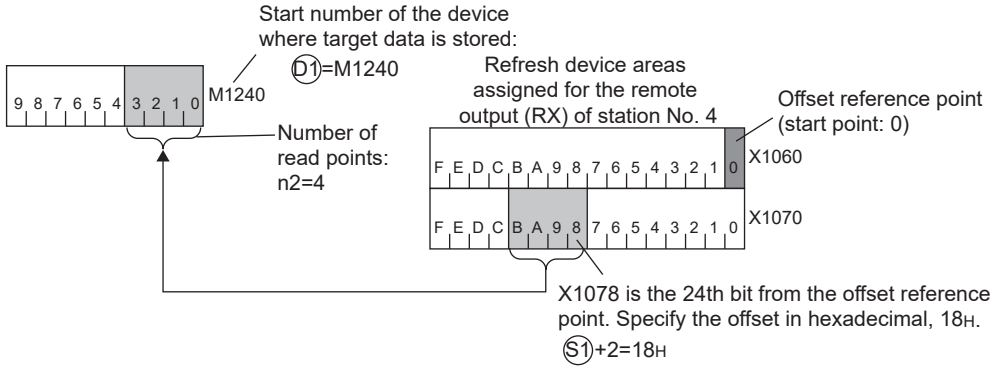
- Instruction execution possibility of an execution type for each program.
  - Enabled: Initial program and scan execution type program
  - Disabled: Fixed scan execution type program and interrupt program
- To read data from a refresh device, the data reflection to the station number specified by the instruction is executed at the timing for the auto refresh.

• Number of points specified in n2 is read from the device specified in (D1) to the offset specified in (S1)+2 of the refresh device assigned for the device specified in (S1)+1 of the target station specified in n1 and (S1)+0.

[Structure]



• At the above configuration, number of points specified in n2 is read starting from the offset (X1078) specified in (S1)+2 of the device assigned for the station number 4, then the number of points is read to devices starting from the device specified in (D1).



Point

When a refresh range per station is assigned in transfer settings, specify number of read points so that the range read data from the specified offset is within the range assigned in the same transfer setting. An error occurs if the number of read points over the range assigned in each transfer setting is specified.

- The station type which can and cannot specify with the start I/O number is as follows.

Specification possibility	Station type
Enabled	CC-Link master station, CC-Link master station (compatible with redundant function), CC-Link IE Field Network master station
Disabled	CC-Link local station, CC-Link standby master station, CC-Link IE Field Network local station, CC-Link IE Field Network submaster station

- Because the available range of the station number is 1 to 120, the station number for the master station in n1 cannot be specified to (S1)+0. If the station number is specified, the "OPERATION ERROR" (error code: 4102) occurs.
- SM739 (Refresh device write/read instruction in execution flag) turns on during the instruction execution. When SM739 is on, the following instructions cannot be executed.
  - S(P).REFDVWRB
  - S(P).REFDVWRW
  - S(P).REFDVRDB
  - S(P).REFDVRDW

If these instructions are executed, no processing is performed. When an error is detected at the instruction execution (before SM739 turns ON), the completion device ((D2)+0), the completion device ((D2)+1), and SM739 do not turn on.

- The instruction completion can be checked in the completion device ((D2)+0 and (D2)+1).
  - Completion device ((D2)+0)

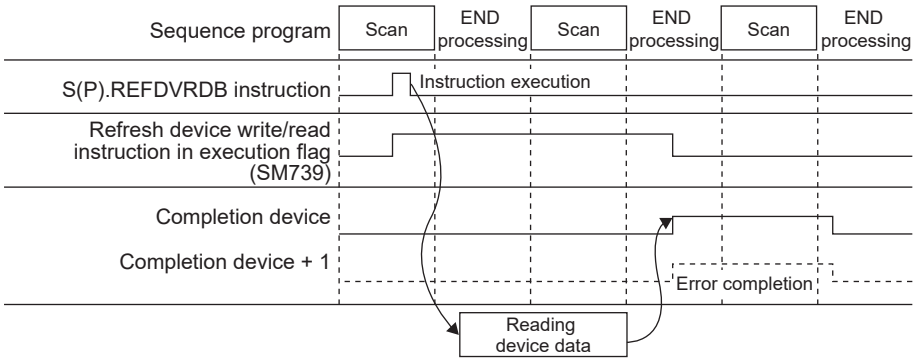
The device turns on at the END processing in a scan where the instruction is completed and turns off at the next END processing.

  - Completion device ((D2)+1)

The device turns on or off by the status when the instruction is completed.

Normal completion: No change from off

Error completion: The device turns on at the END processing in a scan where the instruction is completed and turns off at the next END processing.



- A module set parameters by the dedicated instruction and a CC-Link module operating by the automatic CC-Link startup cannot be specified with the instruction.
- The read source (points in n2 from a device specified in control data) and read destination (points in n2 from (D1)) are overlapped, data can be read. Read data starting from a device specified in control data when data are read to the smaller device number. Read data starting from a device specified in control data + ((n2)-1) when data are read to the larger device number.

Operation error

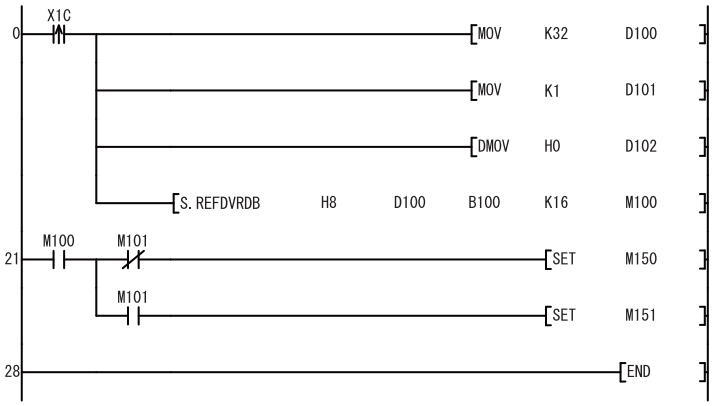
- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4002	The instruction is used for the CPU module that cannot support.	—	—	—	—	○	○
	When the module cannot be specified the start I/O number in n1						
4004	When the device which cannot be specified is specified	—	—	—	—	○	○
4101	When the specified device exceeds the range of the number of device points	—	—	—	—	○	○
	When the start I/O number in n1 is out of the specified range						
	When the device type number in (S1)+1 is out of the specified range						
	When the read offset in (S1)+2 is out of the specified range						
	When the number of read points in n2 is out of the specified range						
	When the number of read points in n2 exceeds the range of the number of device points						
4102	When the station number in (S1)+0 is out of the specified range	—	—	—	—	○	○
	When the station number in (S1)+0 does not exist						
	When the station number in (S1)+0 is the master station specified in n1						
4150	When the start I/O number in n1 is the station type which cannot be specified	—	—	—	—	○	○
	When the start I/O number in n1 does not exist in the network parameter						
4151	When the device in (S1)+1 of the station number specified in (S1)+0 is not assigned the refresh device	—	—	—	—	○	○
	When the read offset in (S1)+2 exceeds the refresh device range assigned for the device in (S1)+1 of the station number specified in (S1)+0						
	When the number of read points in n2 exceeds the assignment range of the setting for one transfer from the read offset in (S1)+2						

Program example

- The following program reads 16 device values from the head of the refresh device (offset: 0) assigned for the remote input (RX) in the remote I/O station on the station number 32 controlled by the CC-Link master station of the start I/O number 0080H to B100 when X1C is turned on.

[Ladder Mode]



[List Mode]

Step	Instruction	Device					
0	LDP	X1C					
1	MOV	K32	D100				
3	MOV	K1	D101				
5	DMOV	H0	D102				
8	S. REFDVRDB		H8	D100	B100	K16	M100
21	LD	M100					
22	MPS						
23	ANI	M101					
24	SET	M150					
25	MPP						
26	AND	M101					
27	SET	M151					
28	END						

Precautions

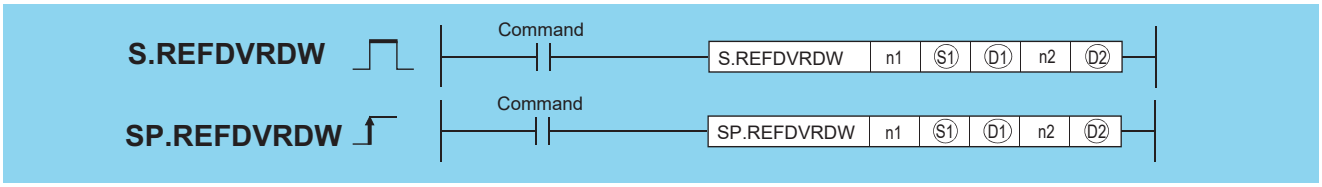
- Do not execute the instruction in an interrupt program. If the instruction is executed, no processing is performed. In addition, the completion device ((D2)+0), the completion device ((D2)+1), and SM739 do not turn on. If the instruction is executed in a fixed scan execution type program, they also do not turn on.

Refresh device read (in 16-bit units)

S(P).REFDVRDW

Basic | High performance | Process | Redundant | Universal | LCPU

- QnUD(H)CPU, QnUDE(H)CPU: the serial number (first five digits) is "14072" or later
- QnUDVCPU, QnUDPVCPU: the serial number (first five digits) is "16043" or later.
- Built-in Ethernet port LCPU: Supported
- Q00UJCPU, Q00UCPU, Q01UCPU, Q02UCPU, L02SCPU, and L02SCPU-P cannot be used.



n1: Start I/O number (0H to FEH) (BIN 16-bit)*1 of the master station controlling the station assigned the refresh device which reads data
(S1): Start number of the device stored control data (device name)
(D1): Start number of the device stored read data from the refresh device assigned the device specified in (S1)+0 and (S1)+1 (device name)
n2: Number of read points (1 to 2147483647) (BIN 32-bit)
(D2): Start number of the bit device which turns on for 1 scan by the instruction completion. (D2)+1 also turns on at the error completion (bit).

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
n1	—	○	○	—				○	—
(S1)	—	○	○	—				—	—
(D1)	—	△*2	△*2	—				—	—
n2	—	○	○	—				○	—
(D2)	△*2	—	—	—				—	—

*1 The first 3 digits of the hexadecimal 4 digits which represent the start I/O number.
*2 Local devices and the devices designated for individual programs cannot be used.

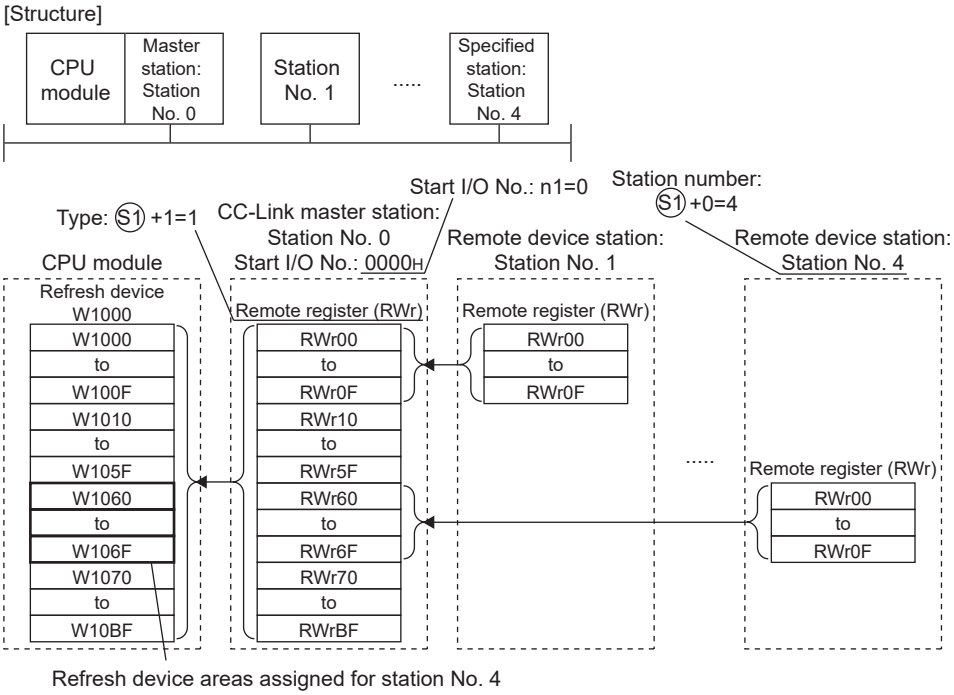
Processing details

- The contents of the data stored in the area starting from (S1) are as indicated below.

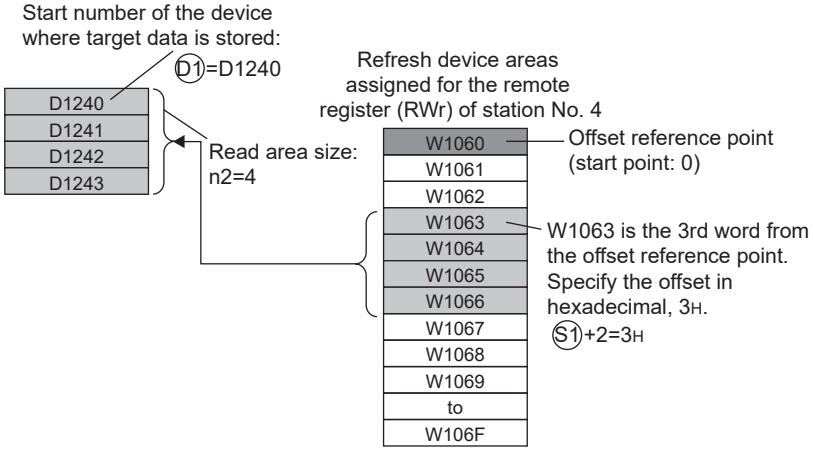
Device	Item	Setting contents	Setting range
(S1)+0	Station number	Station number for the station assigned the refresh device which reads data. When the link special register (SW) is specified as type of (S1)+1, the setting is disabled.	1 to 120
(S1)+1	Type	Type of the refresh device which reads data • 1: Remote register (RWr) • 2: Remote register (RWw) • 3: Link special register (SW)	1 to 3
(S1)+2 (S1)+3	Offset	Offset from the head of the refresh device assigned the device specified in (S1)+0 and (S1)+1	0 to 2147483647

- Instruction execution possibility of an execution type for each program.
  - Enabled: Initial program and scan execution type program
  - Disabled: Fixed scan execution type program and interrupt program
- To read data from a refresh device, the data reflection to the station number specified by the instruction is executed at the timing for the auto refresh.

• Number of points specified in n2 is read from the device specified in (D1) to the offset specified in (S1)+2 of the refresh device assigned for the device specified in (S1)+1 of the target station specified in n1 and (S1)+0.



• At the above configuration, number of points specified in n2 is read starting from the offset (W1063) specified in (S1)+2 of the device assigned for the station number 4, then the number of points is read to devices starting from the device specified in (D1).



Point

When a refresh range per station is assigned in transfer settings, specify number of read points so that the range read data from the specified offset is within the range assigned in the same transfer setting. An error occurs if the number of read points over the range assigned in each transfer setting is specified.

- The station type which can and cannot specify with the start I/O number is as follows.

Specification possibility	Station type
Enabled	CC-Link master station, CC-Link master station (compatible with redundant function), CC-Link IE Field Network master station
Disabled	CC-Link local station, CC-Link standby master station, CC-Link IE Field Network local station, CC-Link IE Field Network submaster station

- Because the available range of the station number is 1 to 120, the station number for the master station in n1 cannot be specified to (S1)+0. If the station number is specified, the "OPERATION ERROR" (error code: 4102) occurs.
- SM739 (Refresh device write/read instruction in execution flag) turns on during the instruction execution. When SM739 is on, the following instructions cannot be executed.
  - S(P).REFDVWRB
  - S(P).REFDVWRW
  - S(P).REFDVRDB
  - S(P).REFDVRDW

If these instructions are executed, no processing is performed. When an error is detected at the instruction execution (before SM739 turns ON), the completion device ((D2)+0), the completion device ((D2)+1), and SM739 do not turn on.

- The instruction completion can be checked in the completion device ((D2)+0 and (D2)+1).
  - Completion device ((D2)+0)

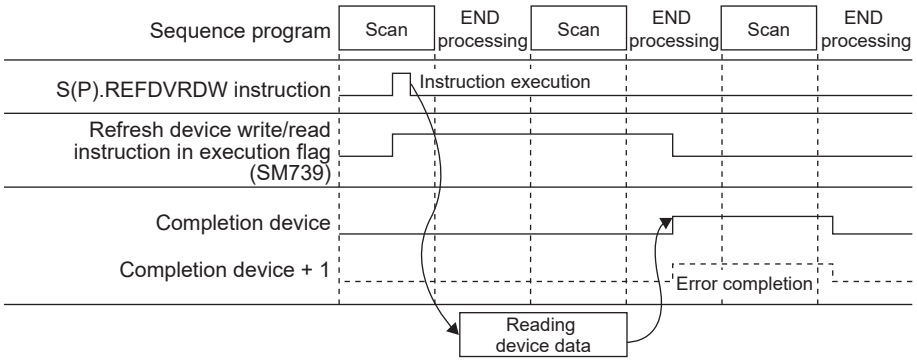
The device turns on at the END processing in a scan where the instruction is completed and turns off at the next END processing.

  - Completion device ((D2)+1)

The device turns on or off by the status when the instruction is completed.

Normal completion: No change from off

Error completion: The device turns on at the END processing in a scan where the instruction is completed and turns off at the next END processing.



- A module set parameters by the dedicated instruction and a CC-Link module operating by the automatic CC-Link startup cannot be specified with the instruction.
- The read source (points in n2 from a device specified in control data) and read destination (points in n2 from (D1)) are overlapped, data can be read. Read data starting from a device specified in control data when data are read to the smaller device number. Read data starting from a device specified in control data + ((n2)-1) when data are read to the larger device number.

Operation error

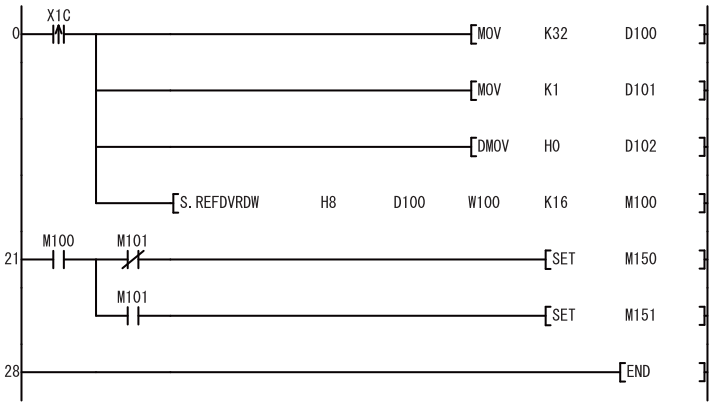
- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4002	The instruction is used for the CPU module that cannot support.	—	—	—	—	○	○
	When the module cannot be specified the start I/O number in n1						
4004	When the device which cannot be specified is specified	—	—	—	—	○	○
4101	When the specified device exceeds the range of the number of device points	—	—	—	—	○	○
	When the start I/O number in n1 is out of the specified range						
	When the device type number in (S1)+1 is out of the specified range						
	When the read offset in (S1)+2 is out of the specified range						
	When the number of read points in n2 is out of the specified range						
	When the number of read points in n2 exceeds the range of the number of device points						
4102	When the station number in (S1)+0 is out of the specified range	—	—	—	—	○	○
	When the station number in (S1)+0 does not exist						
	When the station number in (S1)+0 is the master station specified in n1						
4150	When the start I/O number in n1 is the station type which cannot be specified	—	—	—	—	○	○
	When the start I/O number in n1 does not exist in the network parameter						
4151	When the device in (S1)+1 of the station number specified in (S1)+0 is not assigned the refresh device	—	—	—	—	○	○
	When the read offset in (S1)+2 exceeds the refresh device range assigned for the device in (S1)+1 of the station number specified in (S1)+0						
	When the number of read points in n2 exceeds the assignment range of the setting for one transfer from the read offset in (S1)+2						

Program example

- The following program reads 16 device values from the head of the refresh device (offset: 0) assigned for the remote register (RWr) in the remote device station on the station number 32 controlled by the CC-Link master station of the start I/O number 0080H to W100 when X1C is turned on.

[Ladder Mode]



[List Mode]

Step	Instruction	Device					
0	LDP	X1C					
1	MOV	K32	D100				
3	MOV	K1	D101				
5	DMOV	H0	D102				
8	S. REFDVRDW		H8	D100	W100	K16	M100
21	LD	M100					
22	MPS						
23	ANI	M101					
24	SET	M150					
25	MPP						
26	AND	M101					
27	SET	M151					
28	END						

Precautions

- Do not execute the instruction in an interrupt program. If the instruction is executed, no processing is performed. In addition, the completion device ((D2)+0), the completion device ((D2)+1), and SM739 do not turn on. If the instruction is executed in a fixed scan execution type program, they also do not turn on.
- Specifying digit for the bit device can be used only when the following conditions are met.
  - Digit specification: K4
  - Head of device: multiple of 16

When the above conditions are not met, INSTRCT CODE ERR. (error code: 4004) will occur.

9 MULTIPLE CPU DEDICATED INSTRUCTIONS

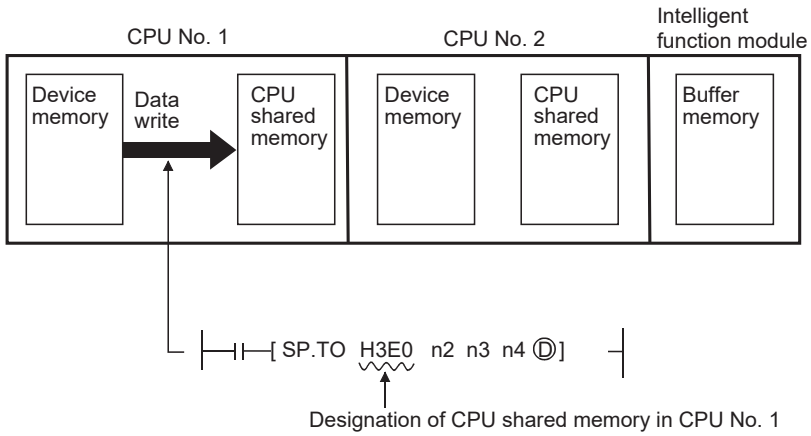
9.1 Writing to the CPU Shared Memory of Host CPU

The S.TO or TO instruction is used to write to the CPU shared memory of the host station in the multiple CPU system.
The following table indicates the usability of the S.TO and TO instructions.

CPU module		S.TO instruction	TO instruction
Basic model QCPU	Q00JCPU	Unusable	Unusable
	Q00CPU, Q01CPU	Usable	Usable
High Performance model QCPU	Q02CPU, Q02HCPU, Q06HCPU, Q12HCPU, Q25HCPU	Usable	Unusable
Process CPU	Q02PHCPU, Q06PHCPU, Q12PHCPU, Q25PHCPU	Usable	Unusable
Redundant CPU	Q12PRHCPU, Q25PRHCPU	Unusable	Unusable
Universal model QCPU	Q00UJCPU	Unusable	Unusable
	Q00UCPU, Q01UCPU, Q02UCPU, Q03UDCPU, Q03UDVCPU, Q03UDECPU, Q04UDHCPU, Q04UDVCPU, Q04UDPVCPU, Q04UDEHCPU, Q06UDHCPU, Q06UDVCPU, Q06UDPVCPU, Q06UDEHCPU, Q10UDHCPU, Q10UDEHCPU, Q13UDHCPU, Q13UDVCPU, Q13UDPVCPU, Q13UDEHCPU, Q20UDHCPU, Q20UDEHCPU, Q26UDHCPU, Q26UDVCPU, Q26UDPVCPU, Q26UDEHCPU, Q50UDEHCPU, Q100UDEHCPU	Usable	Usable
LCPU	L02SCPU, L02SCPU-P, L02CPU, L02CPU-P, L06CPU, L06CPU-P, L26CPU, L26CPU-P, L26CPU-BT, L26CPU-PBT	Unusable	Unusable

Operation of S.TO instruction

The S.TO instruction can write data to the CPU shared memory of the host CPU module.
The following figure shows the processing performed when the S.TO instruction is executed in CPU No. 1.

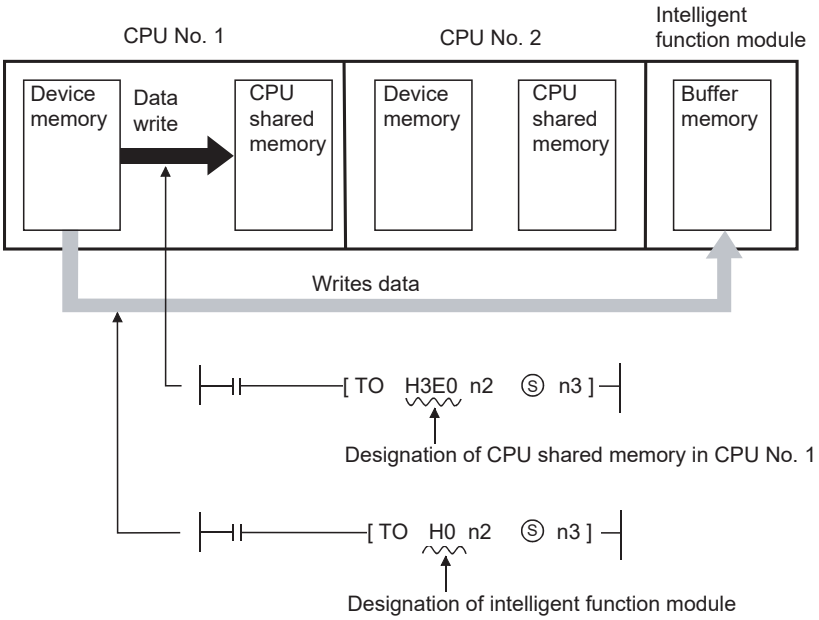


Operation of the TO instruction

The TO instruction can write device memory data to the following memories.

- CPU shared memory of host CPU module
- Buffer memory of intelligent function module

The following figure shows the processing performed when the TO instruction is executed in CPU No. 1.



Point

Both of the S.TO and TO instructions can be used for the Basic model QCPU and Universal model QCPU to write data to the CPU shared memory. However, use of the TO instruction is recommended to write data to the CPU shared memory of the host CPU module, since use of S.TO instruction reduces the number of steps and processing time.

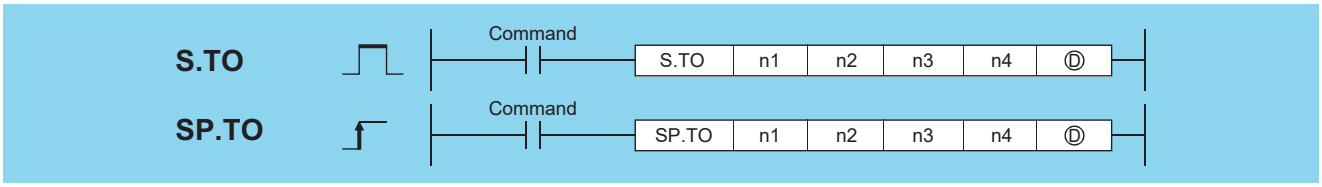
Refer to Page 511 Writing 1-word data to the intelligent function module, writing 2-word data to the intelligent function module when writing to the buffer memory of the intelligent function module by the TO instruction.

Writing to host CPU shared memory

S(P).TO

Basic (Ver.) | High performance (Ver.) | Process | Redundant (×) | Universal | LCPU (×)

- Q00CPU, Q01CPU: The serial number (first five digits) is "04122" or later.
- High Performance model QCPU: Function version B or later



n1: Start I/O number of the host CPU module*1 (BIN 16 bits)
n2: CPU shared memory address of the write destination host CPU (BIN 16 bits)
· Basic model QCPU: 0 to 511
· High Performance model QCPU, Process CPU, Universal model QCPU: 0 to 4095
n3: Head number of the devices where data to be written is stored (BIN 16 bits)
n4: Number of data blocks to be written (BIN 16 bits)
· Basic model QCPU: 1 to 320
· High Performance model QCPU, Process CPU: 1 to 256
· Universal model QCPU: 1 to 2048
(D): Device of the host CPU which is turned ON for one scan by the completion of writing (bits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
n1	—	○		—				○	—
n2	—	○		—				○	—
n3	—	○		—				—	—
n4	—	○		—				○	—
(D)	○	○		—				—	—

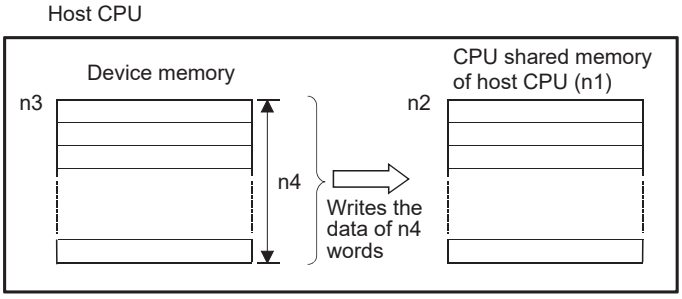
*1 Specified with the upper three digits of the four hexadecimal digits representing the start I/O number.

The value of n1 is specified by the upper three digits of the four hexadecimal digits representing the start I/O number of the slot where the CPU module has been mounted.

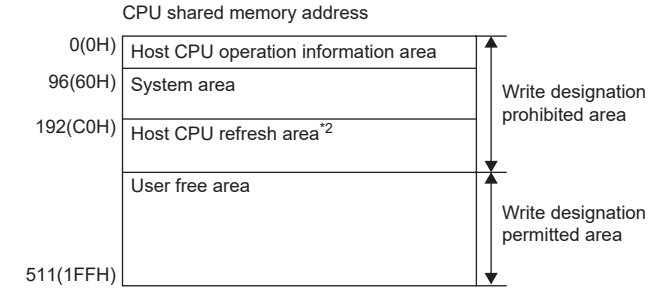
Slot number	Head I/O number	n1
CPU slot	3E00	3E0
Slot 0	3E10	3E1
Slot 1	3E20	3E2
Slot 2	3E30	3E3

Processing details

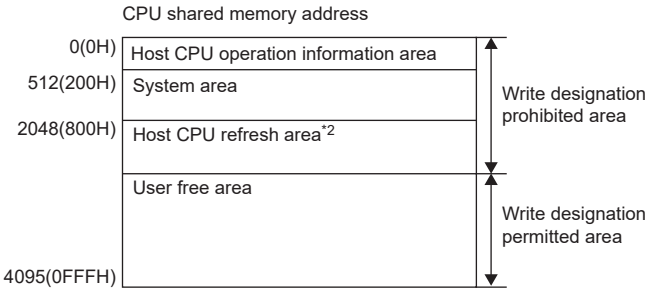
- Writes device data of words n3 to n4 to the CPU shared memory address specified by n2 of the host CPU module or later address. When writing is completed, the completion bit specified by (D) turns ON.



- CPU shared memory address of the Basic model QCPU



- CPU shared memory address of the High Performance model QCPU, Process CPU and Universal model QCPU*3



*2 Usable as a user free area when auto refresh setting is not made.
In addition, even when auto refresh setting is made, the auto refresh send range or later is usable as a user free area.

*3 Data cannot be written to the multiple CPU high speed transmission area of the Universal model QCPU with the S(P).TO instruction.

- When the number of write points is 0, no processing is performed and the completion device does not turn ON, either.
- The S.TO instruction can be executed once to one scan for each CPU. When execution condition is established at two or more places at the same time, the S.TO instruction executed later is not processed since handshake is established automatically.
- The number of data that can be written varies depending on the target CPU module.

CPU module	Number of write points
Basic model QCPU	1 to 320
High Performance model QCPU, Process CPU,	1 to 256
Universal model QCPU	1 to 2048

Point

Writing data to CPU shared memory can be performed using the intelligent function module device. For intelligent function module device, refer to the User's Manual (Function Explanation, Program Fundamentals) for the CPU module used.

Operation error

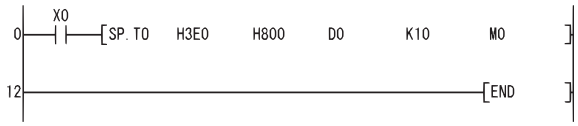
- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
2107	When the head I/O number (n1) of the host CPU is other than that of the host CPU.	—	○	○	—	—	—
2110	No CPU module is installed at the position specified for the head I/O number of the CPU module.	○	○	○	—	○	—
4002	When the specified instruction is improper.	○	○	○	—	○	—
4003	When the number of devices specified is incorrect.	○	○	○	—	○	—
4004	When an Unavailable device is specified.	○	○	○	—	○	—
4100	When the head I/O number (n1) of the host CPU is other than 3E0H/3E1H/ 3E2H/3E3H.	○	○	○	—	○	—
4101	When the host CPU operation information area, system area, or host CPU refresh area is specified to the CPU shared memory address (n2) of the write destination.	—	○	○	—	—	—
	When the number of write points (n4) is outside the specified range of the setting data. When the head of the CPU shared memory address (n2) of the write destination host CPU exceeds the CPU shared memory address range. When the CPU shared memory address (n2) + the number of write points (n4) of the write destination host CPU exceeds the CPU shared memory address range. When the head number of the devices (n3) where the data to be written is stored + the number of write points (n4) exceeds the device range.	○	○	○	—	○	—
4111	When the host CPU operation information area, system area, or host CPU refresh area is specified to the CPU shared memory address (n2) of the write destination.	○	—	—	—	○	—
4112	When the head I/O number (n1) of the host CPU is other than that of the host CPU.	○	—	—	—	○	—

Program example

- The following program stores 10 points of data from D0 into address 800H of the CPU shared memory of CPU No. 1 when X0 is turned ON.

[Ladder Mode]



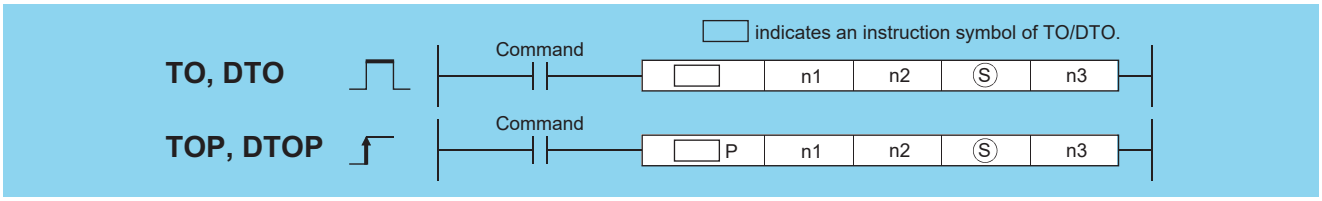
[List Mode]

Step	Instruction	Device				
0	LD	X0				
1	SP. TO	H3E0	H800	D0	K10	M0
12	END					

TO(P), DTO(P)

Ver. Basic | High performance | Process | Redundant | Universal | LCPU

• Q00CPU, Q01CPU: The serial number (first five digits) is "04122" or later.



n1: Start I/O number of the host CPU module*1 (BIN 16 bits)
· Basic model QCPU: 3E0H
· Universal model QCPU: 3E0H to 3E3H

n2: CPU shared memory address of the write destination host CPU (BIN 16 bits)
· Basic model QCPU: 192 to 511
· Universal model QCPU: 2048 to 4095, 10000 to 24335*2

(S): Data to be written or head number of the devices where the data to be written is stored (BIN 16 bits)

n3: Number of data blocks to be written (BIN 16 bits)
· Basic model QCPU: TO(P): 1 to 320, DTO(P): 1 to 160
· Universal model QCPU: TO(P): 1 to 14336, DTO(P): 1 to 7168*2

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others U
	Bit	Word		Bit	Word				
n1	○			○				○	○
n2	○			○				○	—
(S)	○			—				○	—
n3	○			○				○	—

*1 Specified with the upper three digits of the four hexadecimal digits representing the start I/O number.

*2 The setting range varies depending on the auto refresh setting range of the multiple CPU high speed transmission function.

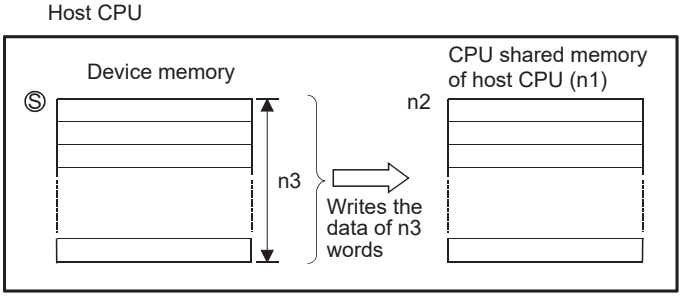
The n1 is specified by the first 3 digits of the hexadecimal 4 digits which represent the head I/O number of the slot mounted to the CPU module.

Slot number	Head I/O number	n1
CPU slot	3E00	3E0
Slot 0	3E10	3E1
Slot 1	3E20	3E2
Slot 2	3E30	3E3

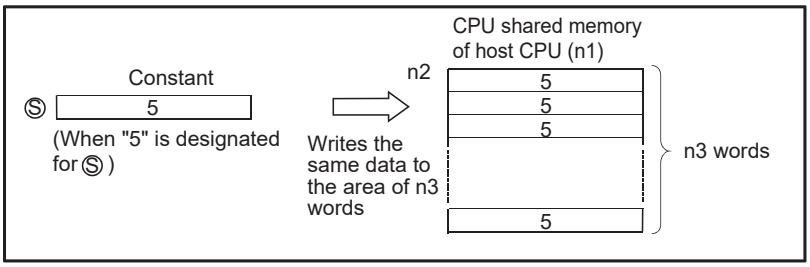
Processing details

■TO

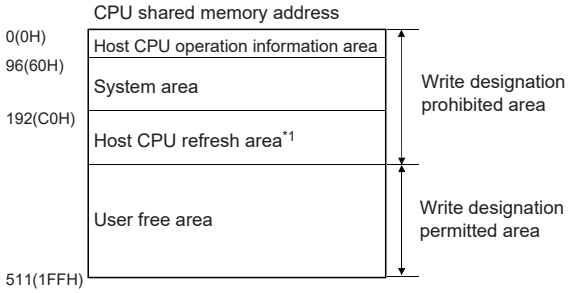
• Writes device data of words (S) to n3 to the CPU shared memory address specified by n2 of the host CPU module or later address.



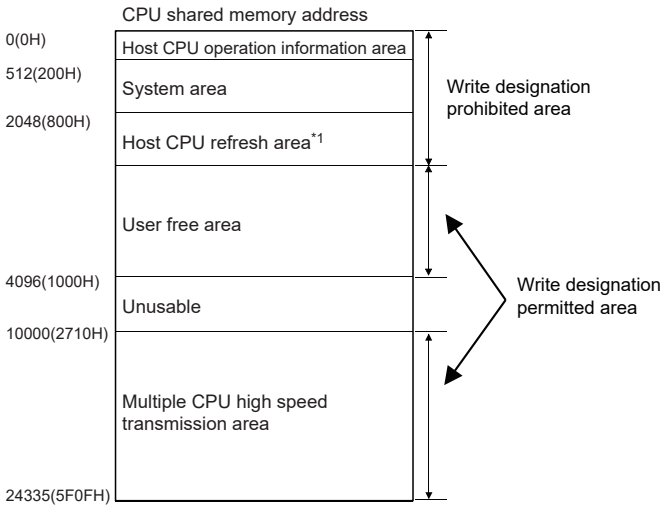
• When a constant is specified to (S), writes the same data (value specified to (S)) to the area of n3 words from the specified CPU shared memory.



• CPU shared memory address of the Basic model QCPU



• CPU shared memory address of the Universal model QCPU*2



*1 Usable as a user free area when auto refresh setting is not made.
In addition, even when auto refresh setting is made, the auto refresh send range or later is usable as a user free area.

*2 With the following CPU modules, data cannot be written to the multiple CPU high speed transmission area.
Q00UCPU, Q01UCPU, Q02UCPU

• No processing is performed when the number of write points is 0.

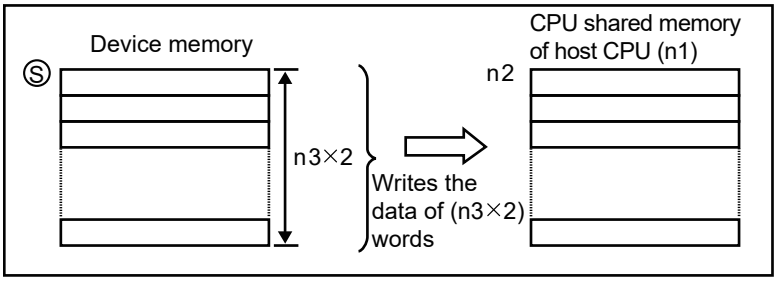
• The number of write data varies depending on the target CPU module.

CPU module	Number of write points
Basic model QCPU	1 to 320
Universal model QCPU	1 to 14336

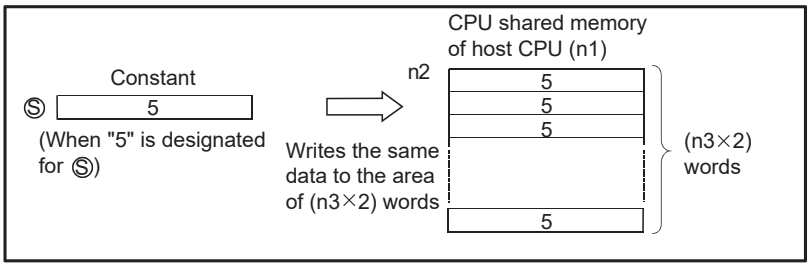
■DTO

- Writes device data of words (S) to (n3×2) to the CPU shared memory address specified by n2 of the host CPU module or later address.

Host CPU



- When a constant is specified to (S), writes the same data (value specified to (S)) to the area of (n3×2) words from the specified CPU shared memory.



- No processing is performed when the number of write points is 0.
- The number of data that can be written varies depending on the target CPU module.

CPU module	Number of write points
Basic model QCPU	1 to 160
Universal model QCPU	1 to 7168

Point

Writing data to CPU shared memory can be performed using the intelligent function module device. For intelligent function module device, refer to the User's Manual (Function Explanation, Program Fundamentals) for the CPU module used.

Operation error

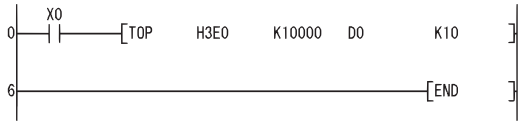
- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
2110	No CPU module is installed at the position specified for the head I/O number of the CPU module.	○	—	—	—	○	—
4101	When the number of write points (n3) is outside the specified range of the setting data. When the CPU shared memory address (n2) of the write destination host CPU + the number of write points (n3) exceeds the CPU shared memory range. When the head number of the devices (S) where the data to be written is stored + the number of write points (n3) exceeds the device range. When the head of CPU shared memory address (n2) of the write destination host CPU is outside the allowable range.	○	—	—	—	○	—
4111	When the head of CPU shared memory address (n2) of the write destination host CPU is an invalid value.	○	—	—	—	○	—
4112	When the I/O number specified in (n1) is other than that of the host CPU (Exclude the case of when the multiple CPU high speed transmission area of other CPU is used.)	○	—	—	—	○	—

Program example

- The following program stores 10 points of data from D0 into address 10000 of the CPU shared memory of CPU No. 1 when X0 is turned ON.

[Ladder Mode]

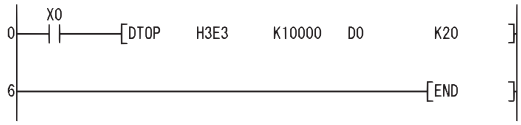


[List Mode]

Step	Instruction	Device			
0	LD	X0			
1	TOP	H3E0	K10000	D0	K10
6	END				

- The following program stores 20 points of data from D0 into address 10000 of the CPU shared memory of CPU No. 4 when X0 is turned ON.

[Ladder Mode]



[List Mode]

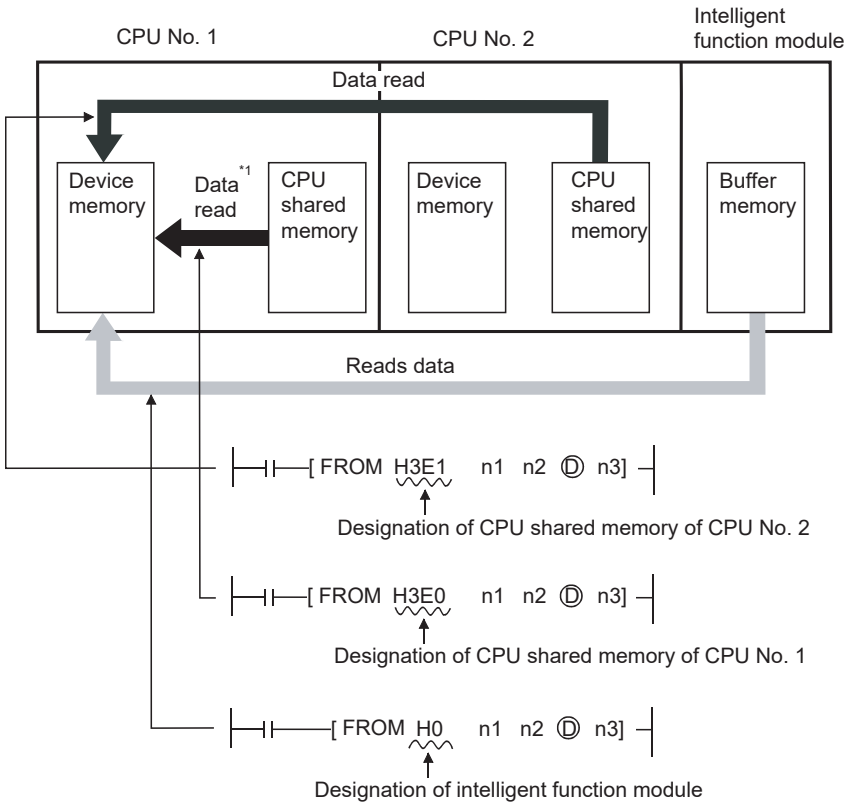
Step	Instruction	Device			
0	LD	X0			
1	DTOP	H3E3	K10000	D0	K20
6	END				

9.2 Reading from the CPU Shared Memory of Another CPU

The FROM(P)/DFRO(P) instruction of Multiple CPU system can be read from the following memories.

- Buffer memory of intelligent function module
- CPU shared memory of other CPU module
- CPU shared memory of host CPU module (applicable for the Basic model QCPU and Universal model QCPU)

The following figure shows the processing performed when the FROM(P) instruction is executed in CPU No. 1.



*1 Applicable for the Basic model QCPU and Universal model QCPU

Point

Refer to Page 508 Reading 1-word data from the intelligent function module, reading 2-word data from the intelligent function module for reading the buffer memory of the intelligent function module with the FROM/DFRO instruction.

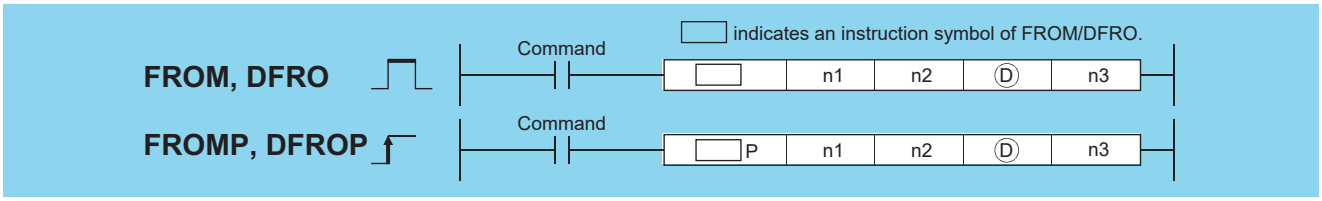
Reading from other CPU shared memory

FROM(P), DFRO(P)

Ver. Basic | Ver. High performance | Process | Redundant (×) | Universal | LCPU (×)

- Q00CPU, Q01CPU: The serial number (first five digits) is "04122" or later.
- High Performance model QCPU: Function version B or later

■When Basic model QCPU, Universal model QCPU is used



n1: Start I/O number of the reading target CPU module*1 (BIN 16 bits)
· Basic model QCPU: 3E0H to 3E2H
· Universal model QCPU: 3E0H to 3E3H

n2: Head address of data to be read (BIN 16 bits)
· Basic model QCPU: 0 to 512
· Universal model QCPU: 0 to 4095, 10000 to 24335*2

(D): Head number of the devices where the read data is stored (BIN 16 bits)

n3: Number of read data (BIN 16 bits)
· Basic model QCPU: FROM(P): 1 to 512, DFRO(P): 1 to 256
· Universal model QCPU: FROM(P): 1 to 14336*2, DFRO(P): 1 to 7168*2

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others U
	Bit	Word		Bit	Word				
n1	○	○		○				○	○
n2	○	○		○				○	—
(D)	—	○		—				—	—
n3	○	○		○				○	—

*1 Specified with the upper three digits of the four hexadecimal digits representing the start I/O number.

*2 The setting range varies depending on the auto refresh setting range of the multiple CPU high speed transmission function.

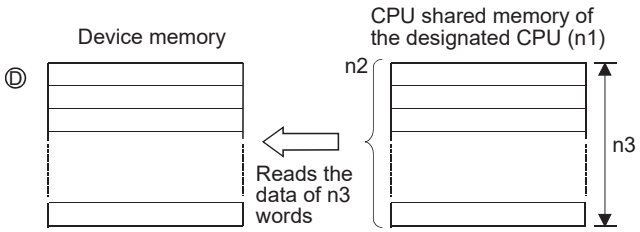
The n1 is specified by the first 3 digits of the hexadecimal 4 digits which represent the head I/O number of the slot mounted to the CPU module.

Slot number	Head I/O number	n1
CPU slot	3E00	3E0
Slot 0	3E10	3E1
Slot 1	3E20	3E2
Slot 2	3E30	3E3

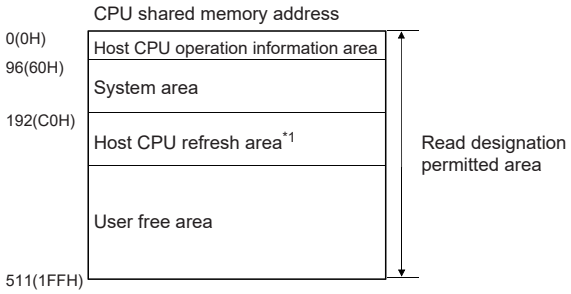
Processing details

■FROM

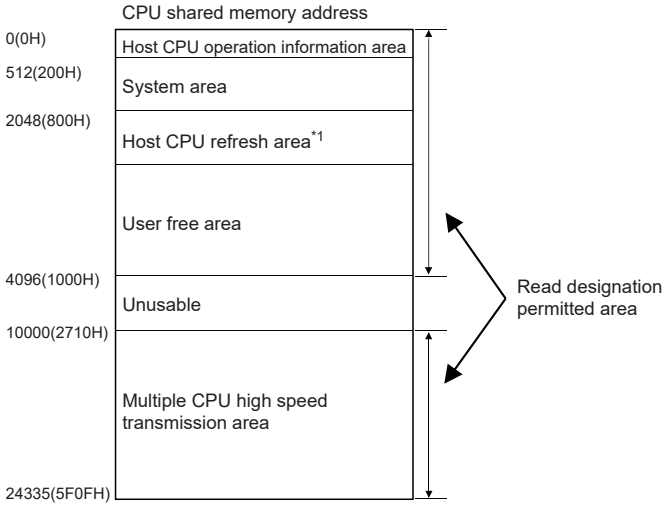
- Reads the data of n3 words from the CPU shared memory address designated by n2 of the CPU module designated by n1, and stores that data into the area starting from the device designated by (D).



- CPU shared memory address of the Basic model QCPU



- CPU shared memory address of the Universal model QCPU*2



*1 Usable as a user free area when auto refresh setting is not made.
When auto refresh setting is made, the auto refresh send range and later are usable as a user free area.

*2 With the following CPU modules, data cannot be read from the multiple CPU high speed transmission area.
Q00UCPU, Q01UCPU, Q02UCPU

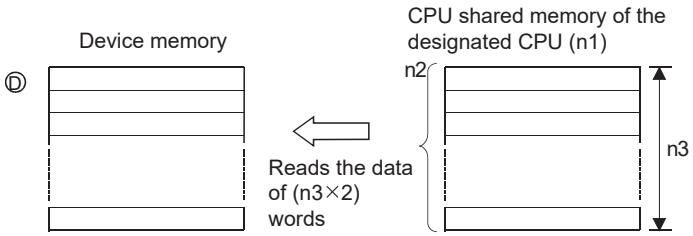
- When 0 is specified in n3 as the number of data to be read, no processing is performed.
- The number of data to be read changes depending on the target CPU module.

CPU module	Number of read points
Basic model QCPU	1 to 512
Universal model QCPU	1 to 14336

- An instruction which has been executed will result in non-processing if it fails to access the target module because the module is faulty or busy in processing.

■DFRO

- Reads the data of (n3×2) words from the CPU shared memory address designated by n2 of the CPU module designated by n1, and stores that data into the area starting from the device designated by (D).



- When 0 is specified in n3 as the number of data to be read, no processing is performed.
- The number of data to be read changes depending on the target CPU module.

CPU module	Number of read points
Basic model QCPU	1 to 256
Universal model QCPU	1 to 7168

- An instruction which has been executed will result in non-processing if it fails to access the target module because the module is faulty or busy in processing.

Point

Read of data from the CPU shared memory can also be performed using the intelligent function module devices.
For intelligent function module device, refer to the User's Manual (Function Explanation, Program Fundamentals) for the CPU module used.

Operation error

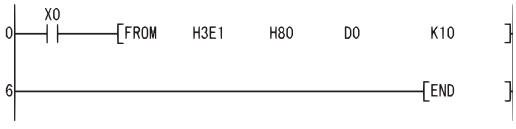
- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
2110	No CPU module is installed at the position specified for the head I/O number of the CPU module.	○	—	—	—	○	—
4101	The head of the CPU shared memory address (n2) which performs reading is outside the CPU shared memory range. The address of the CPU shared memory (n2) which performs reading + the number of read points (n3) is outside the CPU shared memory range. The read data storage device number (D) plus the number of read points (n3) is outside the specified device range. When the head of the CPU shared memory address (n2) which performs reading is an invalid value. (4097 to 9999)	○	—	—	—	○	—

Program example

- The following program stores 10 points of data from address C0H of the CPU shared memory of CPU No. 2 into the area starting from D0 when X0 is turned ON.

[Ladder Mode]

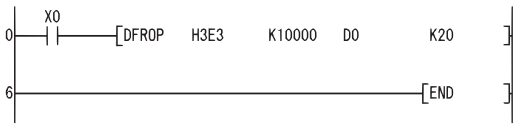


[List Mode]

Step	Instruction	Device
0	LD	X0
1	FROM	H3E1 H80 D0 K10
6	END	

- The following program stores 20 points of data from address 10000 of the CPU shared memory of CPU No. 4 into the area starting from D0 when X0 is turned ON.

[Ladder Mode]

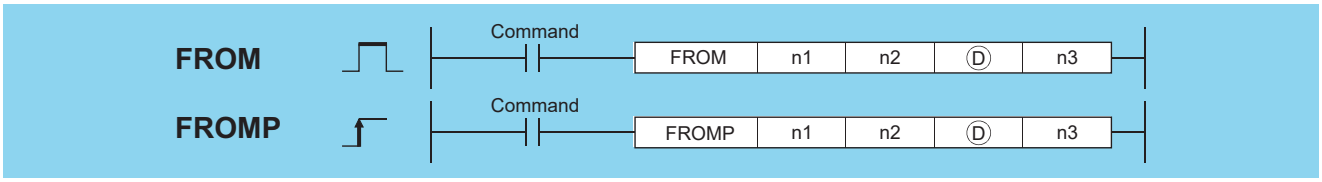


[List Mode]

Step	Instruction	Device
0	LD	X0
1	DFROP	H3E3 K10000 D0 K20
6	END	

FROM(P)

■When High Performance model QCPU, Process CPU is used



n1: Start I/O number of the reading target CPU module*1 (3E0H to 3E3H) (BIN 16 bits)
n2: Head address of data to be read (BIN 16 bits)
(D): Head number of the devices where the read data is stored (BIN 16 bits)
n3: Number of read data (BIN 16 bits)

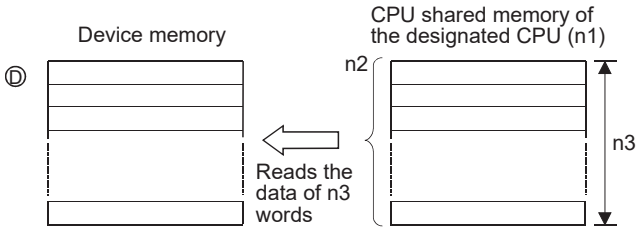
Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others U
	Bit	Word		Bit	Word				
n1	—	○		○				○	○
n2	—	○		○				○	—
(D)	—	○		—				—	—
n3	—	○		○				○	—

*1 Specified with the upper three digits of the four hexadecimal digits representing the start I/O number.
The value of n1 is specified by the upper three digits of the four hexadecimal digits representing the start I/O number of the slot where the CPU module has been mounted.

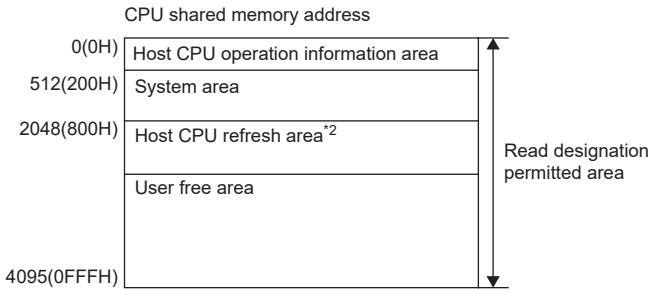
Slot number	Head I/O number	n1
CPU slot	3E00	3E0
Slot 0	3E10	3E1
Slot 1	3E20	3E2
Slot 2	3E30	3E3

Processing details

- Reads the data of n3 words from the CPU shared memory address designated by n2 of the CPU module designated by n1, and stores that data into the area starting from the device designated by (D).



- CPU shared memory address of the High Performance model QCPU and Process CPU



*2 Usable as a user free area when auto refresh setting is not made.
When auto refresh setting is made, the auto refresh send range and later are usable as a user free area.

- When 0 is specified in n3 as the number of data to be read, no processing is performed.
- The number of data to be read changes depending on the target CPU module.

CPU module	Number of read points
High Performance model QCPU Process CPU	1 to 4096

- An instruction which has been executed will result in non-processing if it fails to access the target module because the module is faulty or busy in processing.

Point

Read of data from the CPU shared memory can also be performed using the intelligent function module devices.
For intelligent function module device, refer to the User's Manual (Function Explanation, Program Fundamentals) for the CPU module used.

Operation error

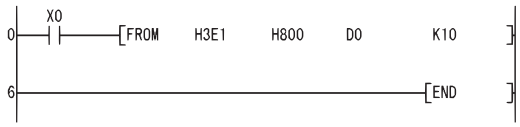
- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
2110	No CPU module is installed at the position specified for the head I/O number of the CPU module.	—	○	○	—	—	—
4101	The head of the CPU shared memory address (n2) which performs reading is outside the CPU shared memory range. The address of the CPU shared memory (n2) which performs reading + the number of read points (n3) is outside the CPU shared memory range. The read data storage device number (D) plus the number of read points (n3) is outside the specified device range. When the head of the CPU shared memory address (n2) which performs reading is an invalid value. (4097 to 9999)	—	○	○	—	—	—

Program example

- The following program stores data of 10 points from address 800H of the CPU shared memory of CPU No. 2. into the area starting from D0 when X0 is turned ON.

[Ladder Mode]



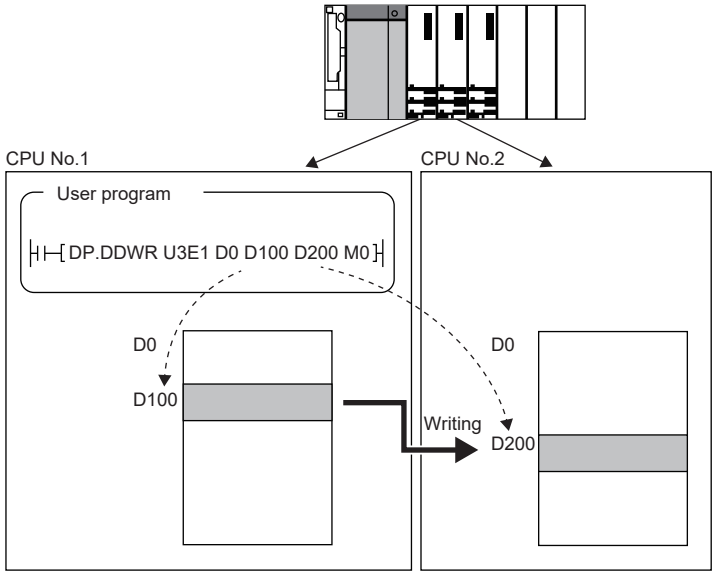
[List Mode]

Step	Instruction	Device
0	LD	X0
1	FROM	H3E1 H800 D0 K10
6	END	

10 MULTIPLE CPU HIGH-SPEED TRANSMISSION DEDICATED INSTRUCTIONS

10.1 Overview

The multiple CPU high-speed transmission dedicated instruction directs the Universal model QCPU to write/read device data to/from the Universal model QCPU in another CPU.
The following shows an operation when CPU No.1 writes device data to CPU No.2 with the multiple CPU high-speed transmission dedicated instruction.



Point

The multiple CPU high-speed transmission dedicated instruction in either host CPU or another CPU (target CPU module of instruction) is available only for the following CPU modules.

- Q03UDCPU, Q04UDHCPU, Q06UDHCPU: The serial number (first five digits) is "10012" or later.
- Q10UDHCPU, Q13UDHCPU, Q20UDHCPU, Q26UDHCPU
- Built-in Ethernet port QCPU

Parameter setting and system configuration to execute the instruction

The multiple CPU high-speed transmission dedicated instruction can be executed in the following parameter setting and system configuration.

- For CPU No. 1, QnUD(H)CPU or, Built-in Ethernet port QCPU is used.
- The multiple CPU high speed main base unit (Q3□DB) is used.
- "Use multiple CPU high speed transmission" is selected in the Multiple CPU settings screen of PLC parameter.

Writable/readable devices

■Writable/readable device names

The following table shows the devices that can be written to/read from the Universal model QCPU in another CPU with the multiple CPU high-speed transmission dedicated instruction.

Category	Type	Device name	Setting of target device (○:Settable, △:Settable with conditions)	Remark
Internal user device	Bit device	X, Y, M, L, B, F, SB	△	Requirements for the setting • Digits are specified by 16 bits (4 digits). • The start bit device is multiples of 16 (10H).
	Word device	T, ST, C, D, W, SW	○	—
Internal system device	Bit device	SM	△	Requirements for the setting • Digits are specified by 16 bits (4 digits). • The start bit device is multiples of 16 (10H).
	Word device	SD	○	—
File register	Word device	R, ZR	○	—

Point

SB, SW, SM, and SD include system information area.
Take care not to destroy the system information when writing data to the devices above with the D(P).DDWR instruction of the multiple CPU high-speed transmission dedicated instruction.

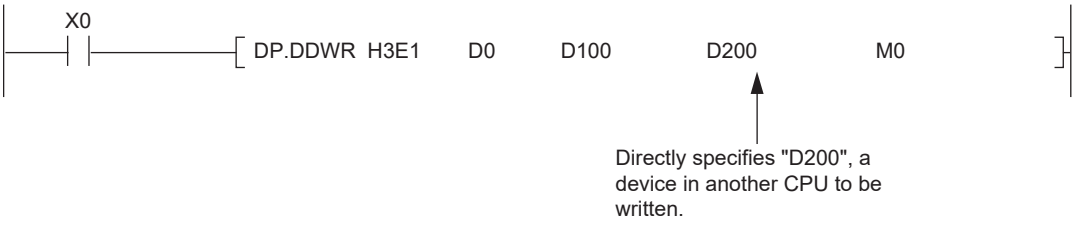
Specification method of a device and writable/readable device range

There are two methods for specifying a device in another CPU: device specification and string specification.
They differ in writable/readable device range to another CPU.

■Device specification

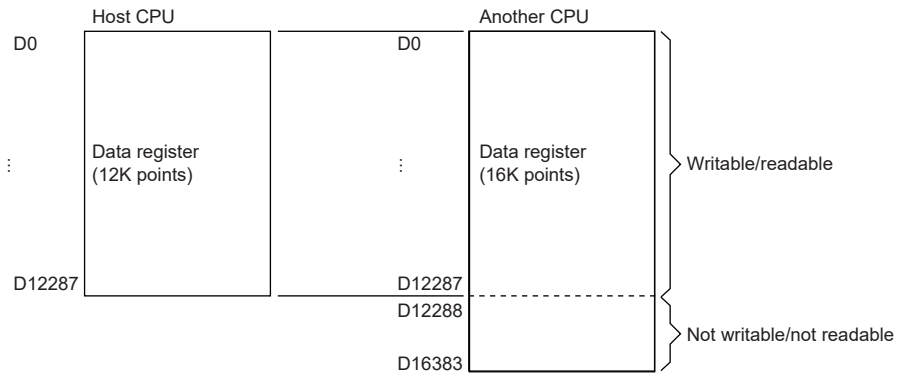
The device specification is a method to directly specify a device in another CPU to be written/read.

[Program for device specification with the DP.DDWR instruction]



In the device specification, data can be written/read within the device range of host CPU. For example, when data register in host CPU is 12K points and data register in another CPU is 16K points, data can be written/read by 12K points from the start of the data register in another CPU.

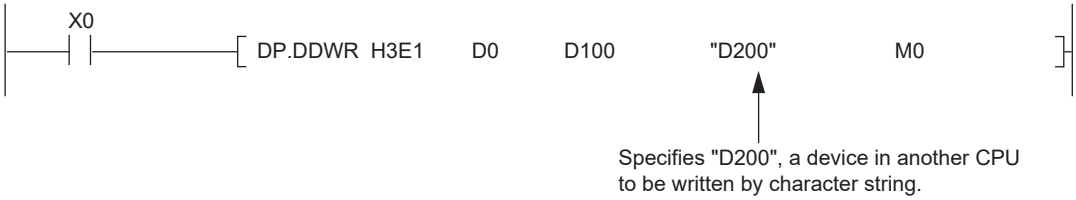
[Writable/readable device range in device specification]



■String specification

The string specification is a method to specify a device in another CPU to be written/read by character string.

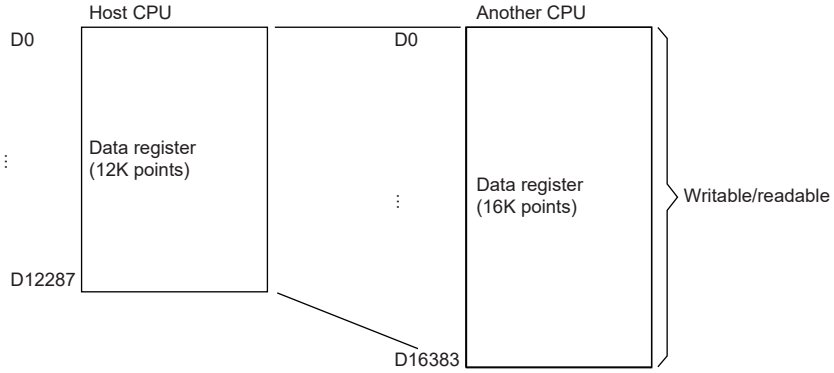
[Program for string specification with the DP.DDWR instruction]



In the string specification, data can be written to/read from all device ranges of another CPU.

For example, when data register in host CPU is 12K points and data register in another CPU is 16K points, data can be written/read by 16K points from the start of the data register in another CPU.

[Writable/readable device range in string specification]



Point

The following explains precautions for string specification.

- The number of characters that can be specified is 32.
- Whether "0" is appended at the start of the device number or not, the devices are processed as the same. For example, both "D1" and "D0001" are processed as "D1".
- Whether a device is specified by upper case character or lower-case character, they are processed as the same. For example, both "D1" and "d1" are processed as "D1".
- If a device not existing in another CPU is specified by a character string, the instruction will be completed abnormally.

Managing the multiple CPU high speed transmission area

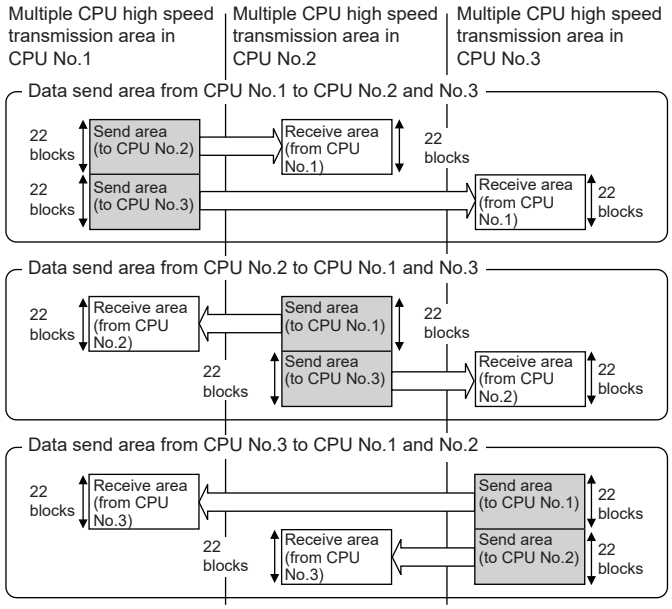
The multiple CPU high speed transmission area is managed by blocks in units of 16 words.

The following table shows the number of blocks that can be used in each CPU and the number of blocks used in the instruction.

Number of CPU modules	System area*1	
	1K points	2K points
2	46	110
3	22	54
4	14	35

*1 For setting of the system area, refer to the QCPU User's Manual (Multiple CPU System).

The following shows configuration of the multiple CPU high speed transmission area when the multiple CPU system is configured with three CPU modules and the system area size is 1K word.



Number of blocks used for the instruction

The number of blocks used for the instruction depends on the number of write points.

The following table shows the number of blocks used for the instruction.

Number of write/read points specified by the instruction	D(P).DDWR instruction	D(P).DDRD instruction
1 to 4	1	1
5 to 20	2	
21 to 36	3	
37 to 52	4	
53 to 68	5	
69 to 84	6	
85 to 100	7	

The instructions that can be executed concurrently

For the Universal model QCPU, the multiple CPU high-speed transmission dedicated instructions can be concurrently executed within the range satisfying the following formula.

[The number of blocks that can be used in each CPU] ≧ [Total number of blocks used for the instructions concurrently executed]

When the number of blocks used for the multiple CPU high-speed transmission dedicated instructions exceeds the total number of blocks in the multiple CPU high speed transmission area, the instruction will not be executed in the scan (no processing) but executed at the next scan.

Note that the instruction will be completed abnormally when the number of empty blocks in the multiple CPU high speed transmission area is less than the setting values of SD796 to SD799 (maximum number of used blocks for multiple CPU high-speed transmission dedicated instruction setting) at the execution of the instruction.

The following table shows execution possibility of the multiple CPU high-speed transmission dedicated instructions when the number of empty blocks in the multiple CPU high speed transmission area is less than the number of blocks used for the multiple CPU high-speed transmission dedicated instructions or the setting values of SD796 to SD799.

Magnitude relation between SD setting value and the number of empty blocks	**Magnitude relation between the number of blocks used for the instructions[*1] and the number of empty blocks**	
	Number of blocks used for the instruction[*1] ≤ Number of empty blocks[*2]	**Number of blocks used for the instruction[*1] > Number of empty blocks[*2]**
SD setting value[*3] ≤ Number of empty blocks[*2]	Executed	Not executed (no processing)
SD setting value[*3] > Number of empty blocks[*2]	Completed abnormally	

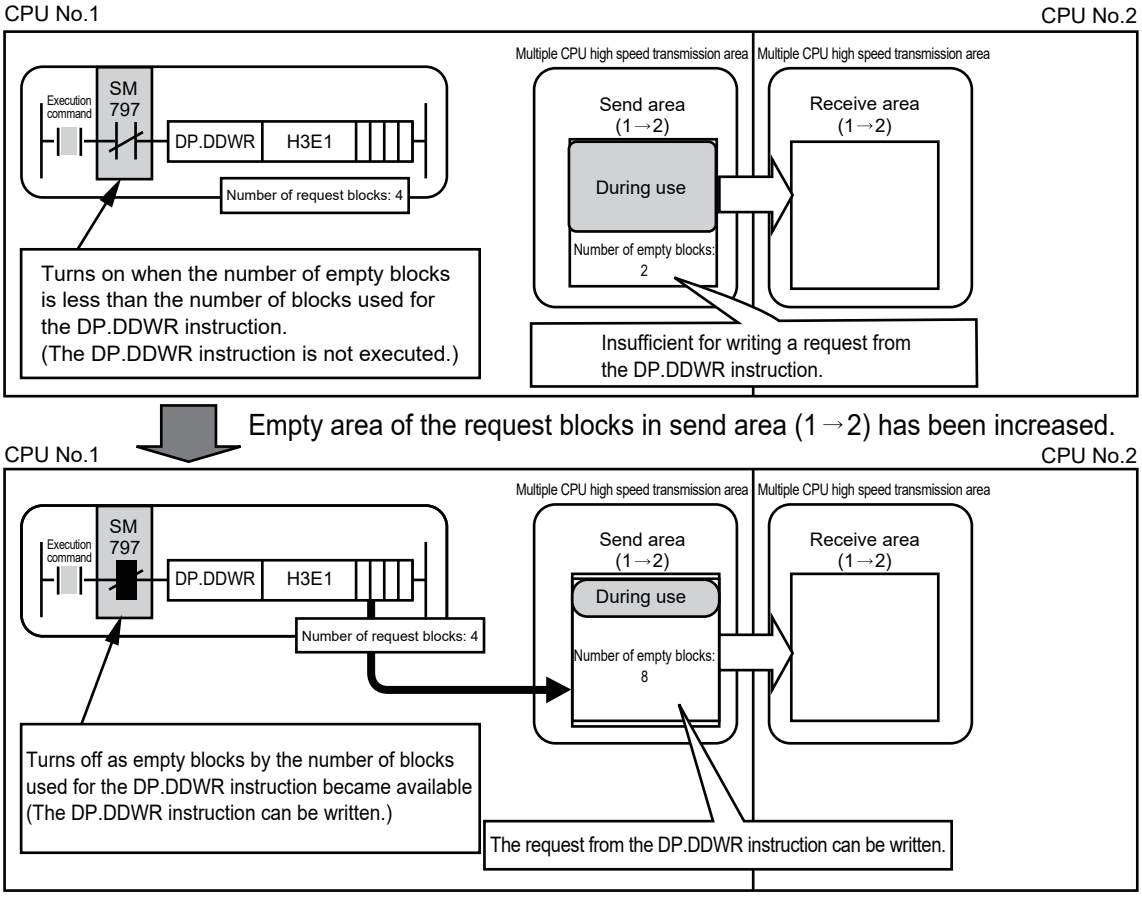
*1 The number of blocks used for the multiple CPU high-speed transmission dedicated instruction.
*2 The number of empty blocks in the multiple CPU high-speed transmission area.
*3 Setting values from SD796 of SD799.

Interlock when using the multiple CPU high-speed transmission dedicated instruction

Special relays SM796 to SM799 (maximum number of used blocks for multiple CPU high-speed transmission dedicated instruction setting) can be used as an interlock for the multiple CPU high-speed transmission dedicated instruction.
When executing the multiple CPU high-speed transmission dedicated instructions concurrently, use SM796 to SM799 as an interlock for the instructions.

Point

When using special relays SM796 to SM799, set the maximum number of blocks for the instruction used for each CPU to special registers SD796 to SD799. (For example, when the maximum number of blocks for the multiple CPU high-speed transmission dedicated instruction to be executed to CPU No.3 is 5, set 5 to SD798.)
When the multiple CPU high speed transmission area becomes equal to or less than the number of blocks set at SD796 to SD799, the corresponding special relay (SM796 to SM799) turns on.

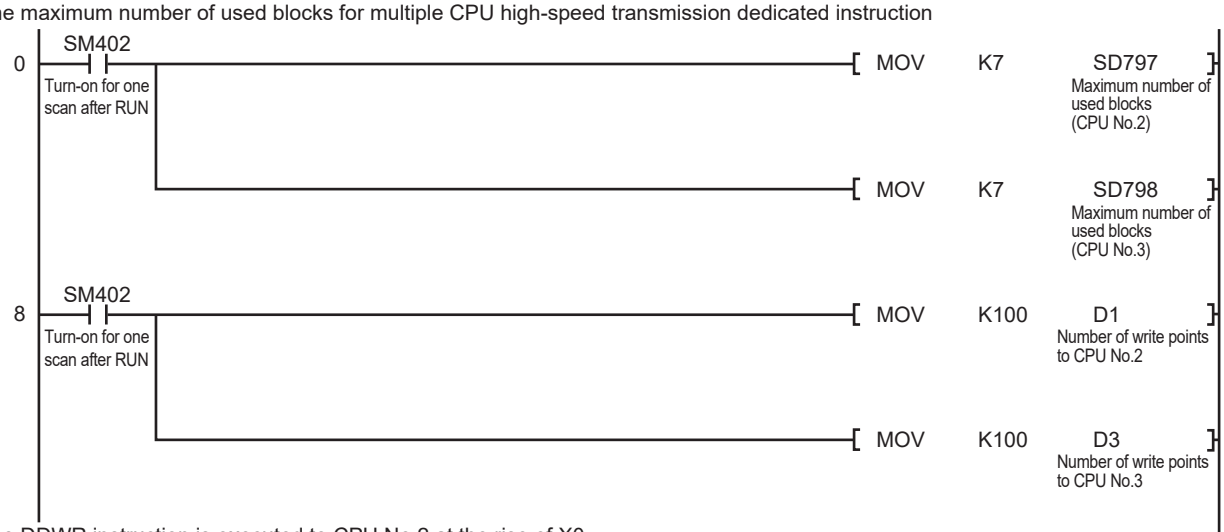


Program example

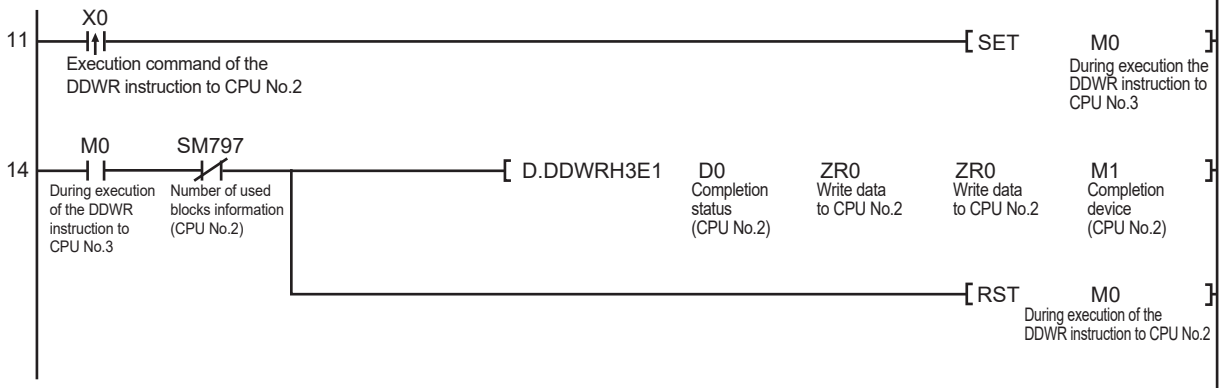
■Program example when SM796 to SM799 are used as an interlock

The following shows a program that executes the D.DDWR instruction to CPU No.2 at the rise of X0, and executes the D.DDWR instruction to CPU No.3 at the rise of X1.

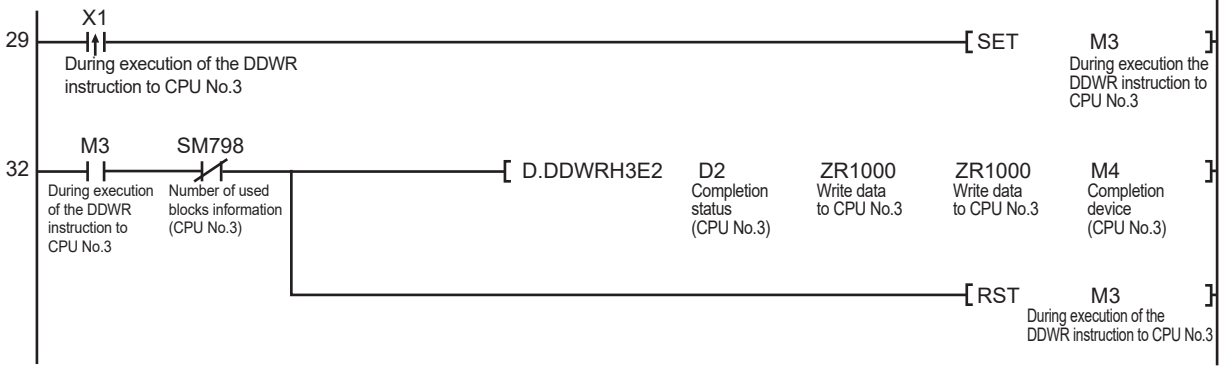
The maximum number of used blocks for multiple CPU high-speed transmission dedicated instruction



The DDWR instruction is executed to CPU No.2 at the rise of X0



The DDWR instruction is executed to CPU No.3 at the rise of X1



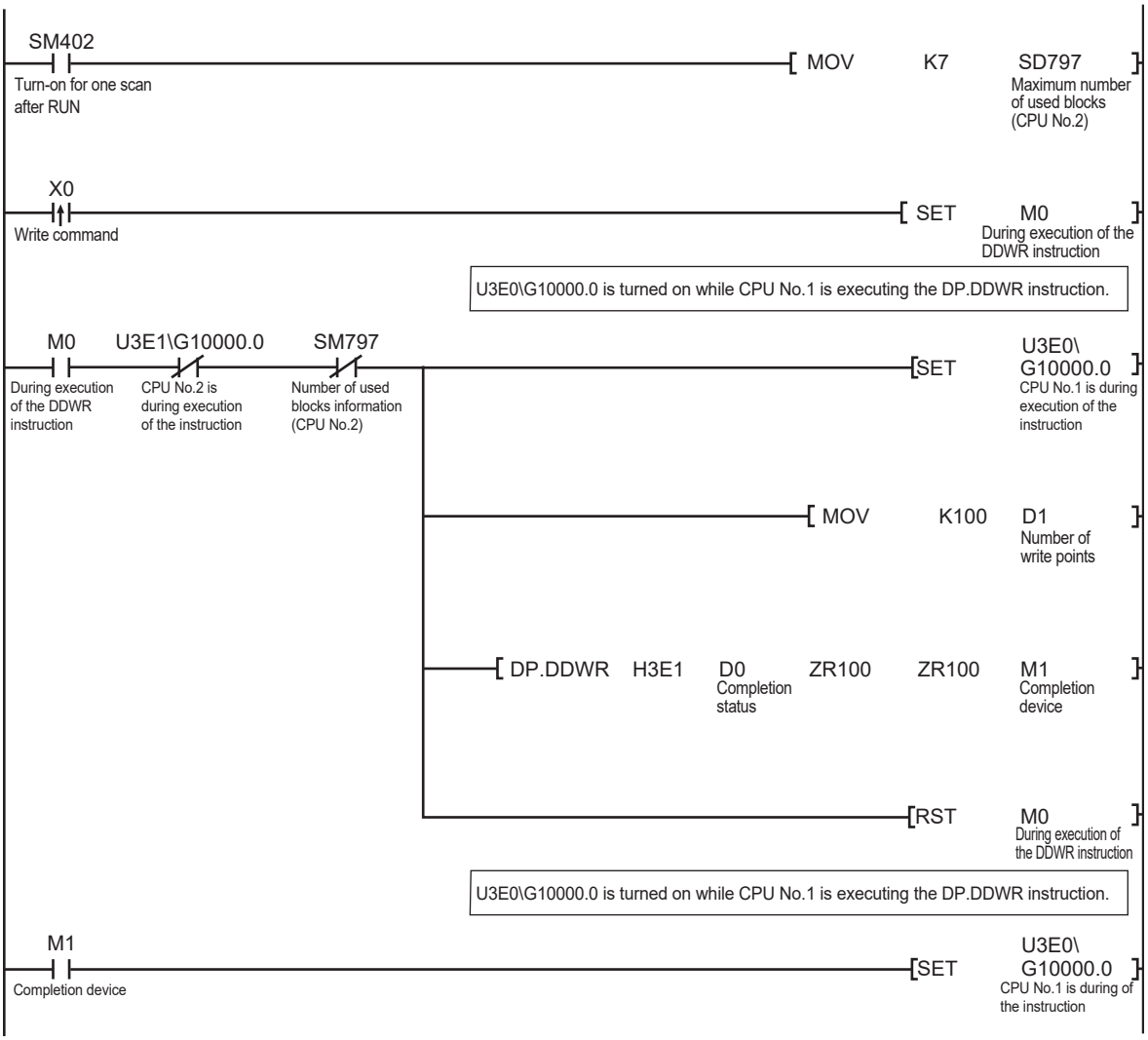
■Program example when the multiple CPU high-speed transmission dedicated instructions are executed to CPU modules by turns

When the multiple CPU high-speed transmission dedicated instructions are executed to Universal model QCPUs by turns, release an interlock to prevent the concurrent execution.

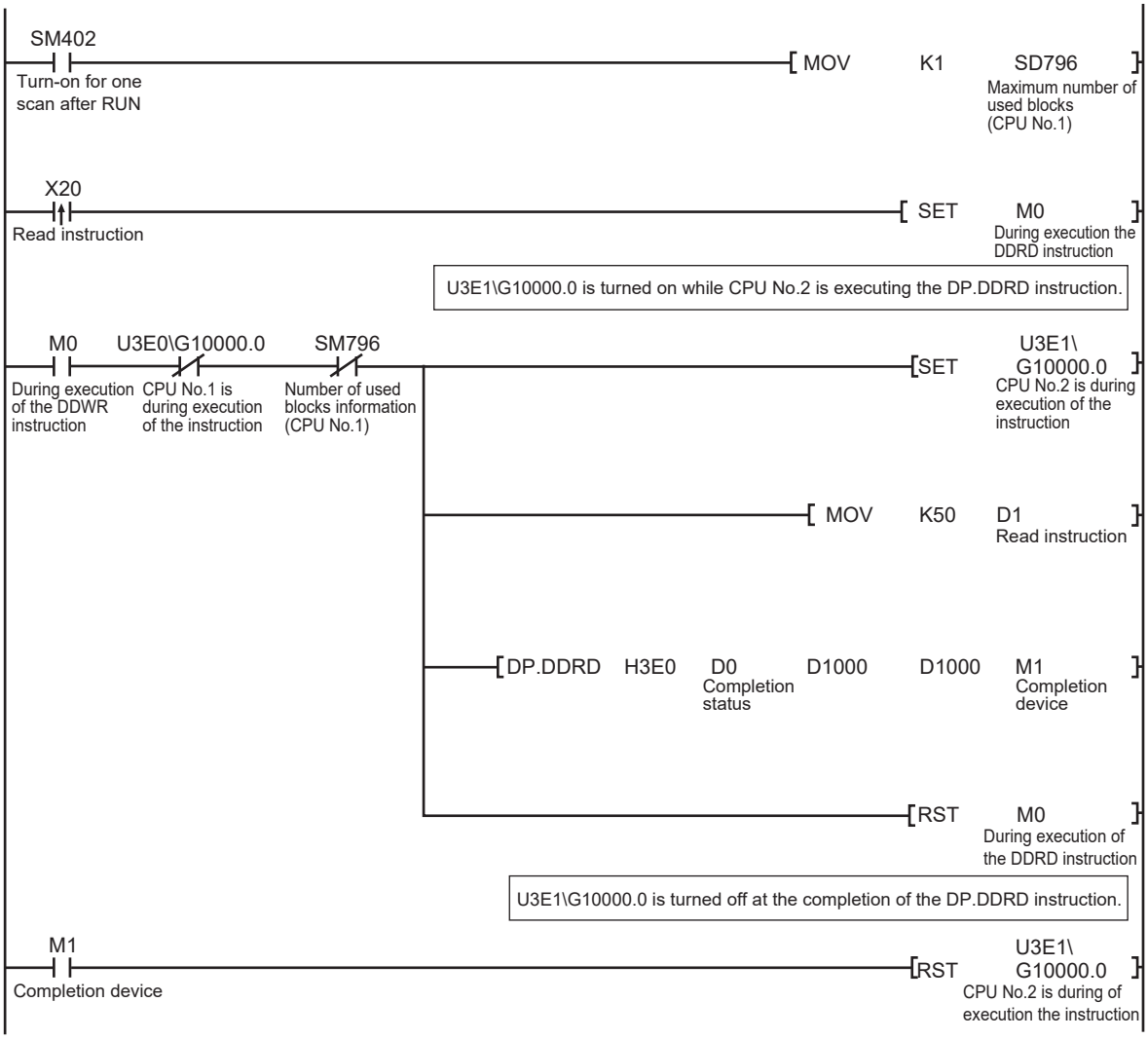
Use the cyclic transmission area device (from U3En\G10000) as an interlock.

The following shows a program example when the multiple CPU high-speed transmission dedicated instructions are executed at CPU No. 1 and 2 by turns.

- Program example when the multiple CPU high-speed transmission dedicated instruction is executed at CPU No.1



• Program example when the multiple CPU high-speed transmission dedicated instruction is executed at CPU No.2



■Program example when data exceeding 100 words are written/read with the multiple CPU high-speed transmission dedicated instruction

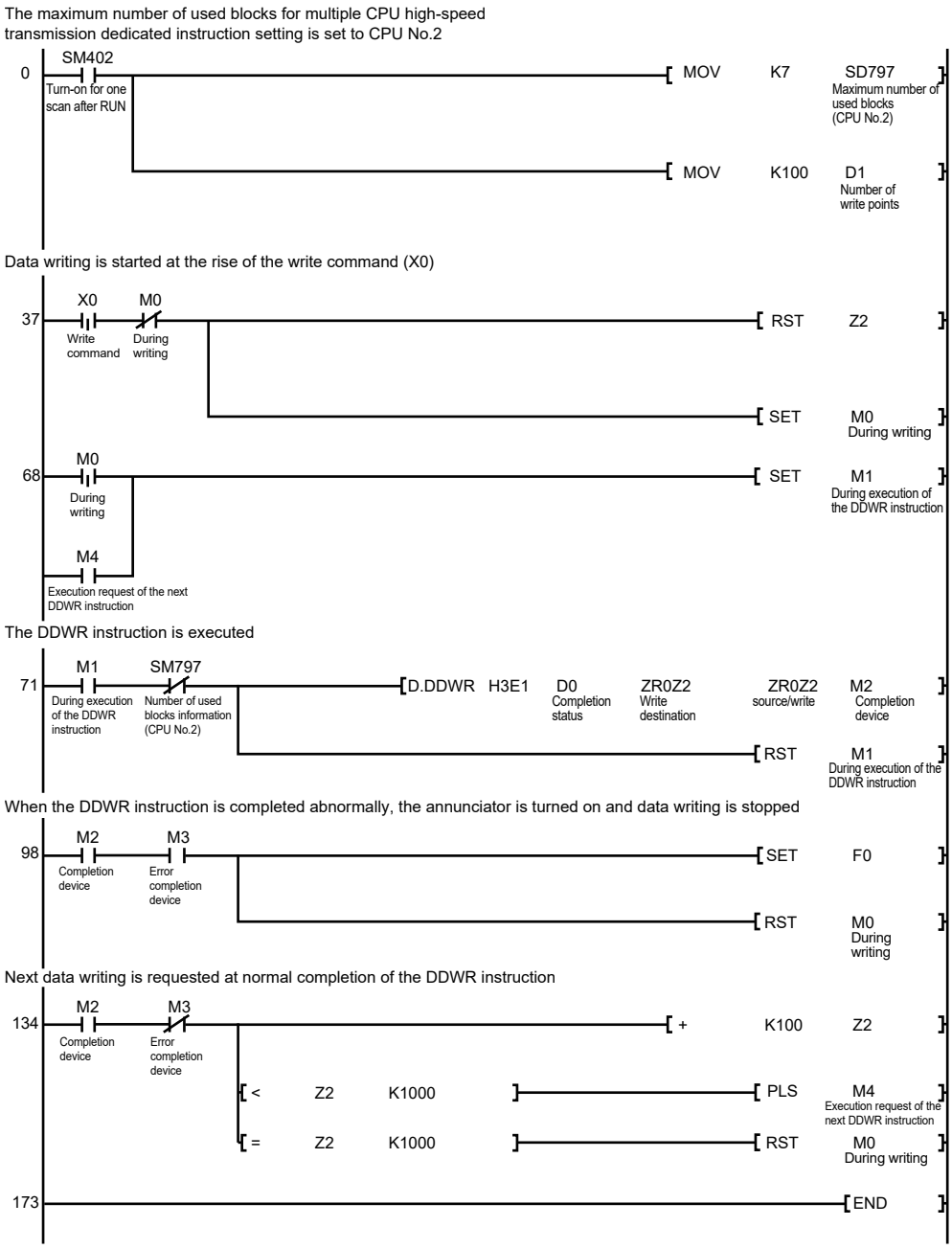
The maximum number of write/read points that can be processed with the multiple CPU high-speed transmission dedicated instruction is 100 words. Data exceeding 100 words can be written/read by executing the multiple CPU high-speed transmission dedicated instruction at several times.

The following shows a program example using the D(P).DDWR instruction of the multiple CPU high-speed transmission dedicated instruction. The similar program can be used when using the D(P).DDRD instruction of the multiple CPU high-speed transmission dedicated instruction.

- Program example when one D(P).DDWR instruction is executed

The following shows a program example that writes ZR0 to ZR999 (1000 points) in CPU No.1 to ZR0 to ZR999 in CPU No.2 with the D.DDWR instruction.

In the following program example, the next D.DDWR instruction is executed after the completion device of the D.DDWR instruction (M2) turns on so that only one D.DDWR instruction may be executed.

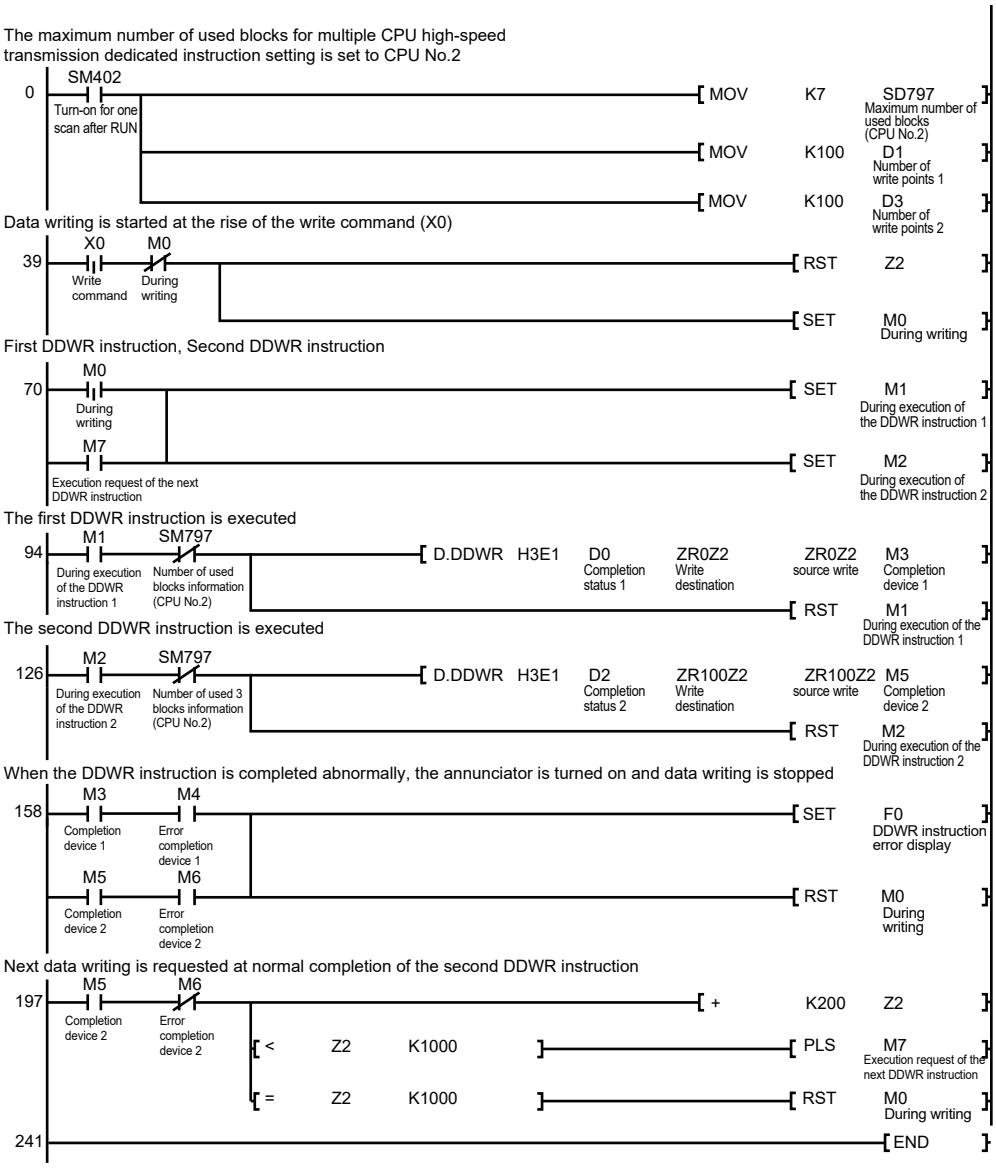


• Program example when the D(P).DDWR instructions are executed concurrently

The following shows a program example that writes ZR0 to ZR999 (1000 points) in CPU No.1 to ZR0 to ZR999 in CPU No.2 with the D.DDWR instruction.

As shown on the program example, multiple CPU device write/read instructions can be executed concurrently.

When reading/writing devices with the multiple CPU high-speed transmission dedicated instructions concurrently, the more the total number of blocks in the multiple CPU high speed transmission area (send area), the more the time taken to complete reading/writing with the multiple CPU high-speed transmission dedicated instruction can be shortened.

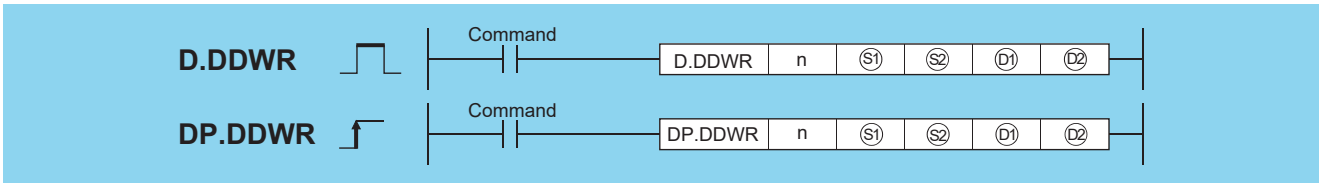


10.2 Writing Devices to Another CPU

D(P).DDWR

Basic | High performance | Process | Redundant | Universal (Ver.) | LCPU

- Universal model QCPU: The serial number (first five digits) is "10012" or later.
- Q00UJCPU, Q00UCPU, Q01UCPU, and Q02UCPU cannot be used.



Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word*5		Bit	Word				
n*1	—	○	○	—				○	—
(S1)*2	—	△*3	△*4	—				—	—
(S2)*2	—	○	○	—				—	—
(D1)*2	—	○	○	—				—	—
(D2)*2	△*6	—	△*4	—				—	—

*1 Index modification cannot be made to setting data n.
*2 Index modification cannot be made to setting data from (S1) to (D2).
*3 Local devices cannot be used.
*4 File registers cannot be used per program.
*5 FD and @□(indirect specification) cannot be used.
*6 FX and FY cannot be used.

Setting data

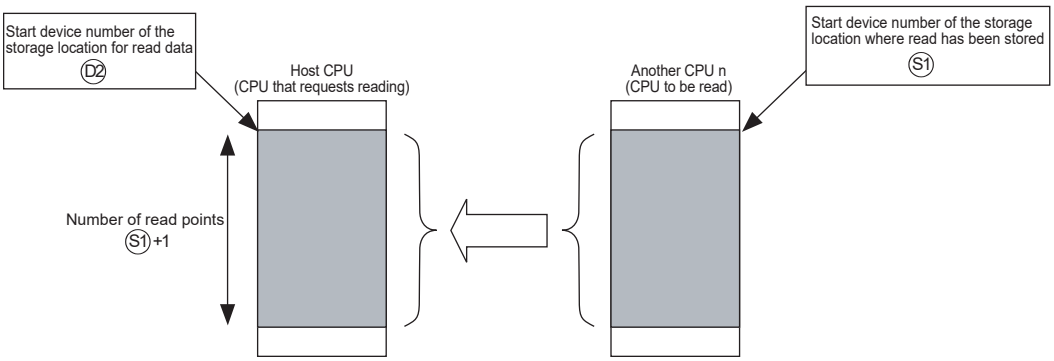
Setting data	Description	Data type
n	The result of dividing the start I/O number of another CPU by 16 CPU No.1: 3E0H, CPU No.2: 3E1H, CPU No.3: 3E2H, CPU No.4: 3E3H	BIN 16 bits
(S1)	Start device of the host CPU that stores control data	Device name
(S2)	Start device of the host CPU that stores data to be written	
(D1)	Start device of another CPU where data to be written will be stored	Device*7 Character string*8*9
(D2)	Completion device	Bit

*7 By specifying a file register (R, ZR), data can be written to devices in another CPU, outside the range of host CPU.
*8 By specifying the start device by " ", data can be written to devices in another CPU, outside the range of host CPU.
*9 Indexed devices cannot be specified (e.g. D0Z0).

Device	Item	Setting data	Setting range	Set by
(S1)+0	Completion status	An execution result upon completion of the instruction is stored. 0000(H): No errors (normal completion) Other than 0000(H): Error code (error completion)	—	System
(S1)+1	Number of write points	Set the number of write points in units of words.	1 to 100	User

Processing details

- In multiple CPU system, data stored in a device specified by host CPU (S2) or later is stored by the number of write points specified by ((D2)+1) into a device specified by another CPU (n) (D1) or later.



- Whether to complete the D(P).DDWR instruction normally can be checked by the completion device ((D2)+0) and completion status display device ((D2)+1).

• Completion device ((D2)+0)

Turns on at END processing in the scan where the instruction has been completed, and turns off at the next END processing.

• Completion status display device ((D2)+1)

This device turns on/off depending on the status upon completion of the instruction.

Normal completion: Off

Error completion: Turns on at END processing in the scan where the instruction has been completed, and turns off at the next END processing. At error completion, an error code is stored at control data ((S1)+0: Completion status).

- The number of blocks used for the instruction depends on the number of write points (☞ Page 868 Overview)

Number of blocks used for the instruction

Number of write points specified by the instruction	D(P).DDWR instruction
1 to 4	1
5 to 20	2
21 to 36	3
37 to 52	4
53 to 68	5
69 to 84	6
85 to 100	7

- The instruction will be completed abnormally when there are no empty blocks in the multiple CPU high speed transmission area. Set the number of blocks used for the instruction at special registers (SD796 to SD799), and use the special relays (SM796 to SM799) as an interlock prevent error completion (☞ Page 868 Overview)

Operation error

• In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4350	Specified another CPU is incorrect. Or the multiple CPU high-speed transmission dedicated instruction is disabled. • The reserved CPU has been specified. • A CPU that is not mounted has been specified. • Another CPU start I/O number divided by 16n is not within the range from 3E0H to 3E3H. • The instruction was executed when the module is set to "Do not use multiple CPU high speed transmission". • The instruction was executed with the CPU module that cannot use this instruction. • The host CPU has been specified. • The CPU where the instruction cannot be executed has been specified.	—	—	—	—	○	—
4351	Another CPU does not support this instruction.	—	—	—	—	○	—
4352	The number of devices is incorrect.	—	—	—	—	○	—
4353	The device that cannot be used for the instruction has been specified.	—	—	—	—	○	—
4354	A device has been specified by the character string that cannot be used.	—	—	—	—	○	—
4355	The number of write points, ((S1)+1), is other than 0 to 100.	—	—	—	—	○	—

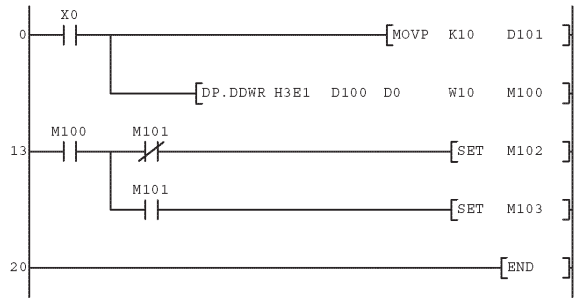
• In any of the following cases, the instruction is completed abnormally, and an error code is stored into a device specified at completion status storage device ((S1)+0).

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
0010H	The request of the instruction to the target CPU is more than the acceptable value. (No empty block exists in the multiple CPU high speed transmission area.)	—	—	—	—	○	—
1001H	A device of another CPU specified in (D1) cannot be used for the CPU, or is outside the device range.	—	—	—	—	○	—
1003H	The response of the instruction from another CPU cannot be returned. (No empty block exists in the multiple CPU high speed transmission area.)	—	—	—	—	○	—
1080H	The number of write points set with the D(P).DDWR instruction is 0.	—	—	—	—	○	—
1090H	The device set as the write destination of the D(P).DDWR instruction is write-protected from outside the CPU module.	—	—	—	—	○	—

Program example

• This program stores data by 10 words starting from D0 in host CPU into W10 or later in CPU No.2 when X0 turns on.

[Ladder Mode]



[List Mode]

Step	Instruction	Device				
0	LD	X0				
1	MOVP	K10	D101			
3	DP.DDWR	H3E1	D100	D0	W10	M100
13	LD	M100				
14	MPS					
15	ANI	M101				
16	SET	M102				
17	MPP					
18	AND	M101				
19	SET	M103				
20	END					

Precautions

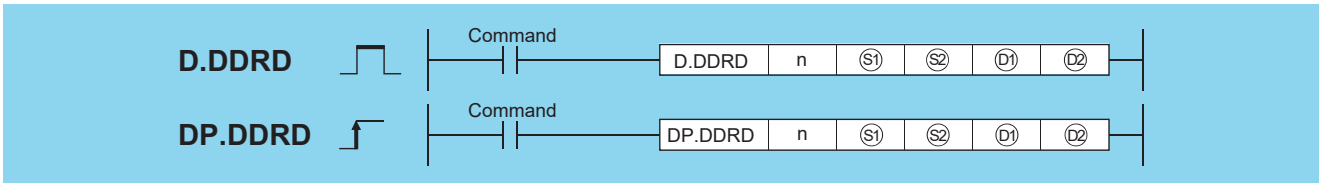
- Digit specification of bit device is possible for n, (S2), and (D1). Note that when the digit specification of bit device is made to (S2) or (D1), the following conditions must be met.
  - Digits are specified by 16 bits (4 digits).
  - The start bit device is multiples of 16 (10H).
- Execute this instruction after checking that the write target CPU is powered on. Not doing so may end up no processing.
- If changing a range of the device specified at setting data between after execution of the instruction and turn-on of the completion device, data to be stored by system (completion status, completion device) cannot be stored normally.
- SB, SW, SM, and SD include system information area. Take care not to destroy the system information when writing data to the SB, SW, SM, and SD with the D(P).DDWR instruction of the multiple CPU high-speed transmission dedicated instruction.

10.3 Reading Devices from Another CPU

D(P).DDRD

Basic | High performance | Process | Redundant | Ver. Universal | LCPU

- Universal model QCPU: The serial number (first five digits) is "10012" or later.
- Q00UJCPU, Q00UCPU, Q01UCPU, and Q02UCPU cannot be used.



Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word*5		Bit	Word				
n*1	—	○	○	—				○	—
(S1)*2	—	△*3	△*4	—				—	—
(S2)*2	—	○	○	—				—	—
(D1)*2	—	○	○	—				—	—
(D2)*2	△*6	—	△*4	—				—	—

*1 Index modification cannot be made to setting data n.
*2 Index modification cannot be made to setting data from (S1) to (D2).
*3 Local devices cannot be used.
*4 File registers cannot be used per program.
*5 FD and @□(indirect specification) cannot be used.
*6 FX and FY cannot be used.

Setting data

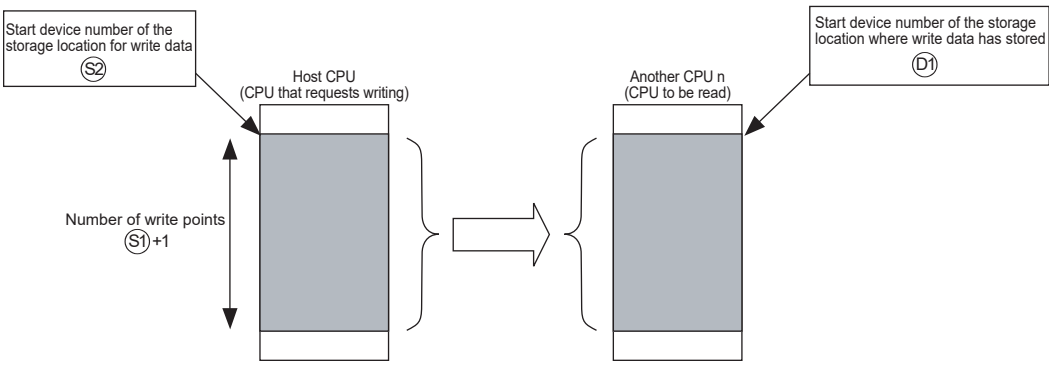
Setting data	Description	Data type
n	The result of dividing the start I/O number of another CPU by 16 CPU No.1: 3E0H, CPU No.2: 3E1H, CPU No.3: 3E2H, CPU No.4: 3E3H	BIN 16 bits
(S1)	Start device of the host CPU that stores control data	Device name
(S2)	Start device of another CPU that stores data to be read	Device*7 Character string*8*9
(D1)	Start device of the host CPU where read data will be stored	Device name
(D2)	Completion device	Bit

*7 By specifying a file register (R, ZR), data can be read to devices in another CPU, outside the range of host CPU.
*8 By specifying the start device by " ", data can be read to devices in another CPU, outside the range of host CPU.
*9 Indexed devices cannot be specified (e.g. D0Z0).

Device	Item	Setting data	Setting range	Set by
(S1)+0	Completion status	An execution result upon completion of the instruction is stored. 0000(H): No errors (normal completion) Other than 0000(H): Error code (error completion)	—	System
(S1)+1	Number of read points	Set the number of read points in units of words.	1 to 100	User

Processing details

In multiple CPU system, data stored in a device specified by another CPU (n) (D1) or later is stored by the number of read points specified by (S1)+1 into a device specified by host CPU (S2) or later.



- Whether to complete the D(P).DDRD instruction normally can be checked by the completion device ((D2)+0) and completion status display device ((D2)+1).

• Completion device ((D2)+0)

Turns on at END processing in the scan where the instruction has been completed, and turns off at the next END processing.

• Completion status display device ((D2)+1)

This device turns on/off depending on the status upon completion of the instruction.

Normal completion: Off

Error completion: Turns on at END processing in the scan where the instruction has been completed, and turns off at the next END processing. At error completion, an error code is stored at control data ((S1)+0: Completion status).

- The number of blocks used for the instruction depends on the number of read points. (☞ Page 868 Overview)

Number of blocks used for the instruction

Number of read points specified by the instruction	D(P).DDRD instruction
1 to 100	1

- The instruction will be completed abnormally when there are no empty blocks in the multiple CPU high speed transmission area. Set the number of blocks used for the instruction at special registers (SD796 to SD799), and use the special relays (SM796 to SM799) as an interlock prevent error completion (☞ Page 868 Overview)

Operation error

• In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4350	Specified another CPU is incorrect. Or the multiple CPU high-speed transmission dedicated instruction is disabled. • The reserved CPU has been specified. • A CPU that is not mounted has been specified. • Another CPU start I/O number divided by 16n is not within the range from 3E0H to 3E3H. • The instruction was executed when the module is set to "Do not use multiple CPU high speed transmission". • The instruction was executed with the CPU module that cannot use this instruction. • The host CPU has been specified. • The CPU where the instruction cannot be executed has been specified.	—	—	—	—	○	—
4351	Another CPU does not support this instruction.	—	—	—	—	○	—
4352	The number of devices is wrong.	—	—	—	—	○	—
4353	The device that cannot be used for the instruction has been specified.	—	—	—	—	○	—
4354	A device has been specified by the character string that cannot be used.	—	—	—	—	○	—
4355	The number of read points ((S1)+1) is other than 0 to 100.	—	—	—	—	○	—

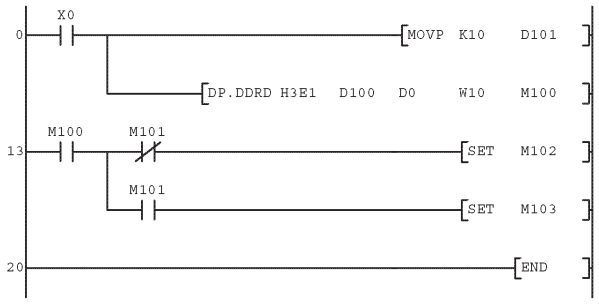
• In any of the following cases, the instruction is completed abnormally, and an error code is stored into a device specified at completion status storage device ((S1)+0).

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
0010H	The request of the instruction to the target CPU is more than the acceptable value. (No empty block exists in the multiple CPU high speed transmission area.)	—	—	—	—	○	—
1001H	The device for another CPU specified at (S2) cannot be used at another CPU, or is out of device range.	—	—	—	—	○	—
1003H	The response of the instruction from another CPU module cannot be returned. (No empty block exists in the multiple CPU high speed transmission area.)	—	—	—	—	○	—
1081H	The number of read points set with the D(P).DDRD instruction is other than 0.	—	—	—	—	○	—

Program example

• This program stores data by 10 words starting from D0 in CPU No.2 into W10 or later in host CPU when X0 turns on.

[Ladder Mode]



[List Mode]

Step	Instruction	Device				
0	LD	X0				
1	MOVP	K10	D101			
3	DP.DDRD	H3E1	D100	D0	W10	M100
13	LD	M100				
14	MPS					
15	ANI	M101				
16	SET	M102				
17	MPP					
18	AND	M101				
19	SET	M103				
20	END					

Precautions

- Digit specification of bit device is possible for n, (S2), and (D1). Note that when the digit specification of bit device is made to (S2) or (D1), the following conditions must be met.
  - Digits are specified by 16 bits (4 digits).
  - The start bit device is multiples of 16 (10H).
- Execute this instruction after checking that the read target CPU is powered on. Not doing so may end up no processing.
- If changing a range of the device specified at setting data between after execution of the instruction and turn-on of the completion device, data to be stored by system (completion status, completion device) cannot be stored normally.

11 REDUNDANT SYSTEM INSTRUCTIONS (FOR REDUNDANT CPU)

11.1 System Switching

SP.CONTSW

Basic | High performance | Process | **Redundant** | Universal | LCPU

(S): Value other than 0 and used to identify the processing that issued the system switching request (BIN 16 bits)
(D): Error completion device number (bits)

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Other
	Bit	Word		Bit	Word				
(S)	—	○		—				○	—
(D)	○	○*1		—				—	—

*1 The bit specification for the word device is available.

Processing details

- Switches between the control system and standby system at the END processing of the scan executed with the SP.CONTSW instruction.
- When using the SP.CONTSW instruction for system switching, the "manual switching enable flag (SM1592)" must have been turned ON (enabled) in advance.
- (S) is provided to identify the processing block of the program where system switching occurred when multiple SP.CONTSW instructions are used.
- At (S), specify a value within the ranges -32768 to -1 and 1 to 32767 (1H to FFFFH).
- The (S) value specified by the SP.CONTSW instruction is stored into the "system switching instruction argument (SD6)" of the error common information when the system switching is normally completed. *2 When multiple SP.CONTSW instructions are executed during the same scan, the argument of the SP.CONTSW instruction executed first is stored into the system switching instruction argument (SD6).

*2 The (S) value specified for the SP.CONTSW instruction can be confirmed in the error common information of the PLC diagnostics dialog box on GX Developer.

- The (S) value specified by the SP.CONTSW instruction is stored into the "system switching instruction argument (SD1602)" of the new control system CPU module when system switching is normally completed. *3 By reading the SD1602 value from the new control system CPU module, which the SP.CONTSW instruction was used for system switching can be confirmed.

*3 The new control system CPU module means the CPU module that was switched from the standby system to the control system by the SP.CONTSW instruction.

- The error completion device is turned ON by the control system CPU module when system switching by the SP.CONTSW instruction was unsuccessful. When OPERATION ERROR is detected due to any of the following reasons at the execution of the SP.CONTSW instruction, the error completion device is turned ON during the instruction execution.
  - 0 is specified at (S) of the executed SP.CONTSW instruction.
  - The "manual switching enable flag (SM1592)" is OFF.
  - The SP.CONTSW instruction was executed by the standby system in the separate mode.
  - The SP.CONTSW instruction was executed in the debug mode.

If systems could not be switched due to any of the reasons given in the following table, the error completion device turns ON when system switching is executed in the END processing.

Reason No.	Reasons for system switching failure
0	Normally completed
1	Tracking cable is disconnected or faulty.
2	Hardware fault, power-off, reset or watchdog timer error occurred in the standby system.
3	Watchdog timer error occurred in the control system.
4	Preparations being made for tracking transfer.
5	Communication time-out.
6	Stop error occurred in the standby system. (Excluding watchdog timer error)
7	Operating status different between the control system and standby system.
8	Memory copy being executed from the control system to the standby system.
9	Write during RUN being executed.
10	Network fault detected by the standby system.

When the error completion device was turned ON due to unsuccessful system switching, 16 is stored into the "reason(s) for system switching (SD1588)" and the reason No. of the above table into the "reason(s) for system switching failure (SD1589)".

- Use a user program or GX Developer to turn OFF the error completion bit that has turned ON. If normal system switching is performed by the execution of the SP.CONTSW instruction with the error completion device ON, the error completion device of the new standby system CPU module is also turned OFF. When system switching is performed due to a factor other than the SP.CONTSW instruction, however, the error completion device is not turned OFF.

Operation error

• In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
4110	The value specified at (S) is 0 at execution of the SP.CONTSW instruction.	—	—	—	○	—	—
4120	The manual switching enable flag (SM1592) is OFF (disabled) at the execution of the SP.CONTSW instruction.	—	—	—	○	—	—
4121	The SP.CONTSW instruction was executed by the standby system CPU module in the separate mode. The SP.CONTSW instruction was executed in the debug mode.	—	—	—	○	—	—

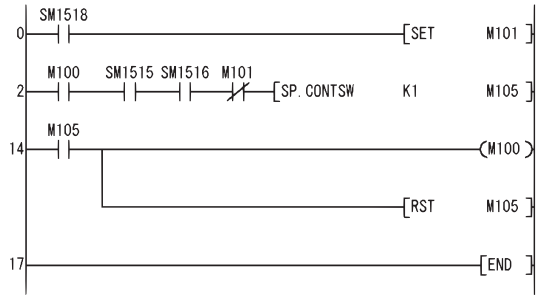
• If system switching was unsuccessful, the error flag (SM0) is turned ON and an error code is stored into SD0.

Error code	Error details	Q00J/ Q00/ Q01	QnH	QnPH	QnPRH	QnU	LCPU
6220	The tracking cable is disconnected or faulty. Hardware fault, power-off, reset or watchdog timer error occurred in the standby system. Watchdog timer error occurred in the control system. Preparations are being made for tracking transfer. Communication time-out occurred. A stop error, excluding watchdog timer error, occurred in the standby system. The operating status differs between the control system and standby system. Memory copy is being executed from the control system to the standby system. Writing during RUN Network fault was detected by the standby system.	—	—	—	○	—	—

Program example

• The following program executes system switching on the rising edge of the system switching command (M100). If the system switching command (M100) remains ON, the SP.CONTSW instruction is also executed by the new control system CPU module after system switching. Therefore, M101 is added to the execution conditions as a consecutive switching prevention flag.

[Ladder Mode]



[List Mode]

Step	Instruction	Device		
0	LD	SM1518		
1	SET	M101		
2	LD	M100		
3	AND	SM1515		
4	AND	SM1516		
5	ANI	M101		
6	SP. CONTSW		K1	M105
14	LD	M105		
15	OUT	M100		
16	RST	M105		
17	END			

APPENDICES

Appendix 1 Operation Processing Time

Definition

Processing time taken by the QCPU, LCPU is the total of the following processing times.

- Total of each instruction processing time
- END processing time (including I/O refresh time)
- Processing time for the function that increases the scan time

Instruction processing time

For the processing time of each instruction, refer to the following.

☞ Page 891 Operation processing time of Basic model QCPU
☞ Page 905 Operation processing time of High Performance model QCPU/Process CPU/Redundant CPU
☞ Page 928 Operation processing time of Universal model QCPU
☞ Page 1035 Operation processing time of LCPU

Other processing times

Refer to the following manual(s) for the END processing time, I/O refresh time, and processing time for the function that increases the scan time.

📖 QnUCPU User's Manual (Function Explanation, Program Fundamentals)
📖 Qn(H)/QnPH/QnPRHCPU User's Manual (Function Explanation, Program Fundamentals)
📖 MELSEC-L CPU Module User's Manual (Function Explanation, Program Fundamentals)

Operation processing time of Basic model QCPU

The processing time for the individual instructions are shown in the table on the following pages.
Operation processing times can vary substantially depending on the nature of the sources and destinations of the instructions, and the values contained in the following tables should therefore be taken as a set of general guidelines to processing time rather than as being strictly accurate.

Point

When using the file resister (ZR), module access device (Un\G□, U3En\G0 to G511), or link direct device (Jn\□), add the processing time shown in Page 904 Table of the time to be added when file register, module access device or link direct device is used to that of each instruction.

Processing time

■Sequence instructions

Instruction		Condition (device)		Processing time (μs)		
				Q00JCPU	Q00CPU	Q01CPU
LD LDI AND ANI OR ORI		X0		0.20	0.16	0.10
		D0.0		0.30	0.24	0.15
LDP LDF ANDP ANDF ORP ORF		X0		0.30	0.24	0.15
		D0.0				
ANB ORB MPS MRD MPP		—		0.20	0.16	0.10
INV		When not executed		0.20	0.16	0.10
		When executed				
MEP MEF		When not executed		0.30	0.24	0.15
		When executed				
EGP		When not executed	(OFF→OFF) (ON→ON)	0.20	0.16	0.10
		When executed	(OFF→ON) (ON→OFF)			
EGF		When not executed	(OFF→OFF) (ON→ON)	17	9.5	9.4
		When executed	(OFF→ON) (ON→OFF)	18	14	14
OUT	Y	When not changed	(OFF→OFF)	0.20	0.16	0.10
			(ON→ON)			
		When changed	(OFF→ON)	0.20	0.16	0.10
			(ON→OFF)			
	D0.0	When not changed	(OFF→OFF)	0.40	0.32	0.20
			(ON→ON)			
		When changed	(OFF→ON)	0.40	0.32	0.20
			(ON→OFF)			
	F	When not executed		24	20	19
		When executed		260	210	200

A

Instruction		Condition (device)			Processing time (μs)		
					Q00JCPU	Q00CPU	Q01CPU
OUT	T	When not executed			1.1	0.88	0.55
		When executed	After time up		1.1	0.88	0.55
			When added	K	1.1	0.88	0.55
				D	1.2	0.96	0.60
	C	When not executed			1.1	0.88	0.55
		When executed	After time up		1.1	0.88	0.55
			When added	K	1.1	0.88	0.55
				D	1.2	0.96	0.60
OUTH	T	When not executed			1.1	0.88	0.55
		When executed	After time up		1.1	0.88	0.55
			When added	K	1.1	0.88	0.55
				D	1.2	0.96	0.60
SET	Y	When not executed			0.20	0.16	0.10
		When executed	When not changed (ON→ON)		0.20	0.16	0.10
			When changed (OFF→ON)		0.20	0.16	0.10
	D0.0	When not executed			0.40	0.32	0.20
		When executed	When not changed (ON→ON)		0.40	0.32	0.20
			When changed (OFF→ON)		0.40	0.32	0.20
	F	When not executed			0.50	0.44	0.25
		When executed			255	205	195
RST	Y	When not executed			0.20	0.16	0.10
		When executed	When not changed (OFF→OFF)		0.20	0.16	0.10
			When changed (ON→OFF)		0.20	0.16	0.10
	D0.0	When not executed			0.40	0.32	0.20
		When executed	When not changed (ON→ON)		0.40	0.32	0.20
			When changed (OFF→ON)		0.40	0.32	0.20
	SM	When not executed			0.20	0.16	0.10
		When executed			0.20	0.16	0.10
	F	When not executed			0.48	0.44	0.25
		When executed			75	69	65
	T, C	When not executed			0.80	0.64	0.40
		When executed			1.0	0.80	0.50
	D	When not executed			0.40	0.32	0.20
		When executed			0.60	0.48	0.30
	Z	When not executed			0.50	0.40	0.25
		When executed			9.4	7.9	7.4
	R	When not executed			—	0.32	0.20
		When executed			—	0.48	0.30
PLS		—			12	9.5	9.2
PLF		—			11	9.5	8.9
FF	Y	When not executed			0.68	0.40	0.25
		When executed			7.5	6.2	5.7
DELTA	DY0	When not executed			0.50	0.40	0.25
		When executed			26	21	21
DELTAP	DY0	When not executed			0.48	0.40	0.25
		When executed			58	45	43
SFT SFTP		When not executed			0.50	0.34	0.25
		When executed			12	8.7	8.3
MC		M0			0.40	0.32	0.20
		D0.0			3.3	2.9	2.8
MCR		—			0.20	0.16	0.10

Instruction	Condition (device)	Processing time (μs)		
		Q00JCPU	Q00CPU	Q01CPU
FEND END	Error check performed	660	600	520
	No error check performed • Battery check • Fuse blown check • I/O module verification	660	600	520
NOP	—	0.20	0.16	0.10
NOPLF PAGE	—	0.20	0.16	0.10

■Basic instructions

The processing time when the instruction is not executed is calculated as follows:

- Q00JCPU: 0.20 × (No. of steps for each instruction + 1) μs
- Q00CPU: 0.16 × (No. of steps for each instruction + 1) μs
- Q01CPU: 0.10 × (No. of steps for each instruction + 1) μs

Instruction	Condition (device)		Processing time (μs)		
			Q00JCPU	Q00CPU	Q01CPU
LD=	In conductive status		0.80	0.64	0.40
	In non-conductive status		0.80	0.64	0.40
AND=	When not executed		0.70	0.56	0.35
	When executed	In conductive status	0.80	0.64	0.40
		In non-conductive status	0.80	0.64	0.40
OR=	When not executed		0.70	0.56	0.35
	When executed	In conductive status	0.80	0.64	0.40
		In non-conductive status	0.80	0.64	0.40
LD<>	In conductive status		0.80	0.64	0.40
	In non-conductive status		0.80	0.64	0.40
AND<>	When not executed		0.70	0.56	0.35
	When executed	In conductive status	0.80	0.64	0.40
		In non-conductive status	0.80	0.64	0.40
OR<>	When not executed		0.70	0.56	0.35
	When executed	In conductive status	0.80	0.64	0.40
		In non-conductive status	0.80	0.64	0.40
LD>	In conductive status		0.80	0.64	0.40
	In non-conductive status		0.80	0.64	0.40
AND>	When not executed		0.70	0.56	0.35
	When executed	In conductive status	0.80	0.64	0.40
		In non-conductive status	0.80	0.64	0.40
OR>	When not executed		0.70	0.56	0.35
	When executed	In conductive status	0.80	0.64	0.40
		In non-conductive status	0.80	0.64	0.40
LD<=	In conductive status		0.80	0.64	0.40
	In non-conductive status		0.80	0.64	0.40
AND<=	When not executed		0.70	0.56	0.35
	When executed	In conductive status	0.80	0.64	0.40
		In non-conductive status	0.80	0.64	0.40
OR<=	When not executed		0.70	0.56	0.35
	When executed	In conductive status	0.80	0.64	0.40
		In non-conductive status	0.80	0.64	0.40
LD<	In conductive status		0.80	0.64	0.40
	In non-conductive status		0.80	0.64	0.40
AND<	When not executed		0.70	0.56	0.35
	When executed	In conductive status	0.80	0.64	0.40
		In non-conductive status	0.80	0.64	0.40

Instruction	Condition (device)		Processing time (μs)		
			Q00JCPU	Q00CPU	Q01CPU
OR<	When not executed		0.70	0.56	0.35
	When executed	In conductive status	0.80	0.64	0.40
		In non-conductive status	0.80	0.64	0.40
LD>=	In conductive status		0.80	0.64	0.40
	In non-conductive status		0.80	0.64	0.40
AND>=	When not executed		0.70	0.56	0.35
	When executed	In conductive status	0.80	0.64	0.40
		In non-conductive status	0.80	0.64	0.40
OR>=	When not executed		0.70	0.56	0.35
	When executed	In conductive status	0.80	0.64	0.40
		In non-conductive status	0.80	0.64	0.40
LDD=	In conductive status		1.0	0.80	0.50
	In non-conductive status		1.0	0.80	0.50
ANDD=	When not executed		0.80	0.64	0.40
	When executed	In conductive status	1.0	0.80	0.50
		In non-conductive status	1.0	0.80	0.50
ORD=	When not executed		0.80	0.64	0.40
	When executed	In conductive status	1.0	0.80	0.50
		In non-conductive status	1.0	0.80	0.50
LDD<>	In conductive status		1.0	0.80	0.50
	In non-conductive status		1.0	0.80	0.50
ANDD<>	When not executed		0.80	0.64	0.40
	When executed	In conductive status	1.0	0.80	0.50
		In non-conductive status	1.0	0.80	0.50
ORD<>	When not executed		0.80	0.64	0.40
	When executed	In conductive status	1.0	0.80	0.50
		In non-conductive status	1.0	0.80	0.50
LDD>	In conductive status		1.0	0.80	0.50
	In non-conductive status		1.0	0.80	0.50
ANDD>	When not executed		0.80	0.64	0.40
	When executed	In conductive status	1.0	0.80	0.50
		In non-conductive status	1.0	0.80	0.50
ORD>	When not executed		0.80	0.64	0.40
	When executed	In conductive status	1.0	0.80	0.50
		In non-conductive status	1.0	0.80	0.50
LDD<=	In conductive status		1.0	0.80	0.50
	In non-conductive status		1.0	0.80	0.50
ANDD<=	When not executed		0.80	0.64	0.40
	When executed	In conductive status	1.0	0.80	0.50
		In non-conductive status	1.0	0.80	0.50
ORD<=	When not executed		0.80	0.64	0.40
	When executed	In conductive status	1.0	0.80	0.50
		In non-conductive status	1.0	0.80	0.50
LDD<	In conductive status		1.0	0.80	0.50
	In non-conductive status		1.0	0.80	0.50
ANDD<	When not executed		0.80	0.64	0.40
	When executed	In conductive status	1.0	0.80	0.50
		In non-conductive status	1.0	0.80	0.50
ORD<	When not executed		0.80	0.64	0.40
	When executed	In conductive status	1.0	0.80	0.50
		In non-conductive status	1.0	0.80	0.50

Instruction	Condition (device)		Processing time (μs)		
			Q00JCPU	Q00CPU	Q01CPU
LDD>=	In conductive status		1.0	0.80	0.50
	In non-conductive status		1.0	0.80	0.50
ANDD>=	When not executed		0.80	0.64	0.40
	When executed	In conductive status	1.0	0.80	0.50
		In non-conductive status	1.0	0.80	0.50
ORD>=	When not executed		0.80	0.64	0.40
	When executed	In conductive status	1.0	0.80	0.50
		In non-conductive status	1.0	0.80	0.50
BKCMP= (S1) (S2) (D) n BKCMP=P (S1) (S2) (D) n	n=1		130	105	97
	n=96		205	175	165
BKCMP<> (S1) (S2) (D) n BKCMP<>P (S1) (S2) (D) n	n=1		130	105	98
	n=96		210	180	165
BKCMP> (S1) (S2) (D) n BKCMP>P (S1) (S2) (D) n	n=1		130	105	97
	n=96		210	180	165
BKCMP>= (S1) (S2) (D) n BKCMP>=P (S1) (S2) (D) n	n=1		130	105	98
	n=96		205	175	165
BKCMP< (S1) (S2) (D) n BKCMP<P (S1) (S2) (D) n	n=1		130	105	98
	n=96		210	180	165
BKCMP<= (S1) (S2) (D) n BKCMP<=P (S1) (S2) (D) n	n=1		130	105	97
	n=96		205	175	165
+ (S) (D) +P (S) (D)	When executed		1.0	0.80	0.50
+ (S1) (S2) (D) +P (S1) (S2) (D)	When executed		1.2	0.96	0.60
- (S) (D) -P (S) (D)	When executed		1.0	0.80	0.50
- (S1) (S2) (D) -P (S1) (S2) (D)	When executed		1.2	0.96	0.60
D+ (S) (D) D+P (S) (D)	When executed		1.3	1.04	0.65
D+ (S1) (S2) (D) D+P (S1) (S2) (D)	When executed		1.5	1.2	0.75
D- (S) (D) D-P (S) (D)	When executed		1.3	1.04	0.65
D- (S1) (S2) (D) D-P (S1) (S2) (D)	When executed		1.5	1.2	0.75
* (S1) (S2) (D) *P (S1) (S2) (D)	When executed		1.1	0.88	0.55
/ (S1) (S2) (D) /P (S1) (S2) (D)	—		19	16	15
D* (S1) (S2) (D) D*P (S1) (S2) (D)	—		41	34	31
D/ (S1) (S2) (D) D/P (S1) (S2) (D)	—		28	23	21
B+ (S) (D) B+P (S) (D)	—		34	28	26
B+ (S1) (S2) (D) B+P (S1) (S2) (D)	—		47	39	37
B- (S) (D) B-P (S) (D)	—		34	28	26
B- (S1) (S2) (D) B-P (S1) (S2) (D)	—		48	40	38
DB+ (S) (D) DB+P (S) (D)	—		58	48	44
DB+ (S1) (S2) (D) DB+P (S1) (S2) (D)	—		60	49	46

Instruction	Condition (device)	Processing time (μs)		
		Q00JCPU	Q00CPU	Q01CPU
DB- (S) (D) DB-P (S) (D)	—	59	48	45
DB- (S1) (S2) (D) DB-P (S1) (S2) (D)	—	60	51	45
B* (S1) (S2) (D) B*P (S1) (S2) (D)	—	42	35	33
B/ (S1) (S2) (D) B/P (S1) (S2) (D)	—	48	40	37
DB* (S1) (S2) (D) DB*P (S1) (S2) (D)	—	140	120	110
DB/ (S1) (S2) (D) DB/P (S1) (S2) (D)	—	83	69	65
BK+ (S1) (S2) (D) n BK+P (S1) (S2) (D) n	n=1	105	86	80
	n=96	185	155	140
BK- (S1) (S2) (D) n BK-P (S1) (S2) (D) n	n=1	105	86	80
	n=96	185	155	140
INC INCP	—	0.70	0.56	0.35
DINC DINCP	—	0.90	0.72	0.45
DEC DECP	—	0.70	0.56	0.35
DDEC DDECP	—	0.90	0.72	0.45
BCD BCDP	—	20	16	15
DBCD DBCDP	—	26	21	20
BIN BINP	—	19	16	15
DBIN DBINP	—	22	18	17
DBL DBLP	—	19	16	15
WORD WORDP	—	23	19	17
GRY GRYP	—	19	16	15
DGRY DGRYP	—	23	19	17
GBIN GBINP	—	52	42	40
DGBIN DGBINP	—	110	88	84
NEG NEGP	—	16	13	12
DNEG DNEGP	—	19	17	15
BKBCD (S) (D) n BKBCDP (S) (D) n	n=1	78	63	57
	n=96	315	275	250
BKBIN (S) (D) n BKBINP (S) (D) n	n=1	74	61	57
	n=96	285	255	230
MOV MOVP	(S)=D0, (D)=D1	0.70	0.56	0.35
	(S)=D0, (D)=J1\W1	155	130	120
DMOV DMOVP	(S)=D0, (D)=D1	0.90	0.72	0.45
	(S)=D0, (D)=J1\W1	165	135	120

Instruction	Condition (device)		Processing time (μs)		
			Q00JCPU	Q00CPU	Q01CPU
$MOV $MOVP	0 characters		46	38	35
	32 characters		98	80	73
CML CMLP	—		0.70	0.56	0.35
DCML DCMLP	—		0.90	0.72	0.45
BMOV (S) (D) n BMOVP (S) (D) n	n=1		27	21	20
	n=96		72	62	53
FMOV (S) (D) n FMOVP (S) (D) n	n=1		23	19	17
	n=96		48	41	36
XCH XCHP	—		7.6	6.3	5.7
DXCH DXCHP	—		9.5	8.0	7.1
BXCH (D1) (D2) n BXCHP (D1) (D2) n	n=1		62	51	48
	n=96		165	140	125
SWAP SWAPP	—		17	14	13
CJ	—		10	8.5	8.1
SCJ	—		10	8.5	8.1
JMP	—		11	8.5	8.1
GOEND	—		3.3	2.9	2.8
DI	—		13	12	11
EI	—		14	11	11
IMASK	—		41	34	35
IRET	—		205	170	155
RFS RFSP	X	n=1	55	46	43
		n=96	79	64	59
	Y	n=1	54	45	41
		n=96	73	61	56

■Application instructions

The processing time when the instruction is not executed is calculated as follows:

- Q00JCPU: 0.20 × (No. of steps for each instruction + 1) μs
- Q00CPU: 0.16 × (No. of steps for each instruction + 1) μs
- Q01CPU: 0.10 × (No. of steps for each instruction + 1) μs

Instruction	Condition (device)	Processing time (μs)		
		Q00JCPU	Q00CPU	Q01CPU
WAND (S) (D) WANDP (S) (D)	When executed	1.0	0.80	0.50
WAND (S1) (S2) (D) WANDP (S1) (S2) (D)	When executed	1.2	0.96	0.60
DAND (S) (D) DANDP (S) (D)	When executed	1.3	1.04	0.65
DAND (S1) (S2) (D) DANDP (S1) (S2) (D)	When executed	1.5	1.2	0.75
BKAND (S1) (S2) (D) n BKANDP (S1) (S2) (D) n	n=1	110	87	79
	n=96	185	155	140
WOR (S) (D) WORP (S) (D)	When executed	1.0	0.80	0.50
WOR (S1) (S2) (D) WORP (S1) (S2) (D)	When executed	1.2	0.96	0.60
DOR (S) (D) DORP (S) (D)	When executed	1.3	1.04	0.65

Instruction	Condition (device)	Processing time (μs)		
		Q00JCPU	Q00CPU	Q01CPU
DOR (S1) (S2) (D) DORP (S1) (S2) (D)	When executed	1.5	1.2	0.75
BKOR (S1) (S2) (D) n BKORP (S1) (S2) (D) n	n=1	110	87	81
	n=96	185	155	140
WXOR (S) (D) WXORP (S) (D)	When executed	1.0	0.80	0.50
WXOR (S1) (S2) (D) WXORP (S1) (S2) (D)	When executed	1.2	0.96	0.60
DXOR (S) (D) DXORP (S) (D)	When executed	1.3	1.04	0.65
DXOR (S1) (S2) (D) DXORP (S1) (S2) (D)	When executed	1.5	1.2	0.75
BKXOR (S1) (S2) (D) n BKXORP (S1) (S2) (D) n	n=1	110	87	81
	n=96	185	155	140
WXNR (S) (D) WXNRP (S) (D)	When executed	1.0	0.80	0.50
WXNR (S1) (S2) (D) WXNRP (S1) (S2) (D)	When executed	1.2	0.96	0.60
DXNR (S) (D) DXNRP (S) (D)	When executed	1.3	1.04	0.65
DXNR (S1) (S2) (D) DXNRP (S1) (S2) (D)	When executed	1.5	1.2	0.75
BKXNR (S1) (S2) (D) n BKXNRP (S1) (S2) (D) n	n=1	110	87	82
	n=96	185	155	140
ROR (D) n RORP (D) n	n=1	13	11	9.7
	n=15	13	11	9.7
RCR (D) n RCRP (D) n	n=1	15	12	12
	n=15	15	13	12
ROL (D) n ROLP (D) n	n=1	13	11	10
	n=15	13	11	10
RCL (D) n RCLP (D) n	n=1	15	13	12
	n=15	16	13	12
DROR (D) n DRORP (D) n	n=1	15	12	12
	n=31	15	13	12
DRCR (D) n DRCRP (D) n	n=1	17	14	14
	n=31	18	16	15
DROL (D) n DROLP (D) n	n=1	14	13	12
	n=31	14	13	12
DRCL (D) n DRCLP (D) n	n=1	18	15	14
	n=31	20	17	16
SFR (D) n SFRP (D) n	n=1	13	10	9.7
	n=15	13	11	9.5
SFL (D) n SFLP (D) n	n=1	12	10	9.5
	n=15	12	9.8	9.5
BSFR (D) n BSFRP (D) n	n=1	42	35	33
	n=96	69	58	54
BSFL (D) n BSFLP (D) n	n=1	41	34	32
	n=96	63	53	50
DSFR (D) n DSFRP (D) n	n=1	19	16	15
	n=96	71	61	53
DSFL (D) n DSFLP (D) n	n=1	19	16	15
	n=96	70	60	52
BSET (D) n BSETP (D) n	n=1	27	22	20
	n=15	27	22	20

Instruction	Condition (device)		Processing time (μs)		
			Q00JCPU	Q00CPU	Q01CPU
BRST (D) n BRSTP (D) n	n=1		27	22	21
	n=15		27	22	21
TEST (S1) (S2) (D) TESTP (S1) (S2) (D)	—		35	30	27
DTEST (S1) (S2) (D) DTESTP (S1) (S2) (D)	—		37	31	28
BKRST (D) n BKRSTP (D) n	n=1		49	41	38
	n=96		64	54	50
SER (S1) (S2) (D) n SERP (S1) (S2) (D) n	n=1	All match	56	54	42
		None match	56	54	42
	n=96	All match	280	240	220
		None match	280	240	220
DSER (S1) (S2) (D) n DSERP (S1) (S2) (D) n	n=1	All match	71	67	53
		None match	71	67	54
	n=96	All match	495	415	375
		None match	500	415	375
SUM SUMP	(S)=0		32	26	25
	(S)=FFFFH		27	22	21
DSUM DSUMP	(S)=0		54	44	42
	(S)=FFFFFFFFH		54	44	42
DECO (S) (D) n DECOP (S) (D) n	n=2		60	50	46
	n=8		80	65	61
ENCO (S) (D) n ENCOP (S) (D) n	n=2	M1=ON	66	55	51
		M4=ON	66	54	51
	n=8	M1=ON	90	76	71
		M256=ON	76	74	71
SEG SEGP	—		8.0	6.8	6.1
DIS (S) (D) n DISP (S) (D) n	n=1		47	39	36
	n=4		53	43	40
UNI (S) (D) n UNIP (S) (D) n	n=1		54	44	41
	n=4		60	49	46
NDIS (S1) (D) (S2) NDISP (S1) (D) (S2)	—		92	76	38
NUNI (S1) (D) (S2) NUNIP (S1) (D) (S2)	—		47	39	36
WTOB (S) (D) n WTOBP (S) (D) n	n=1		56	46	42
	n=96		190	155	145
BTOW (S) (D) n BTOWP (S) (D) n	n=1		56	46	42
	n=96		190	155	145
MAX (S) (D) n MAXP (S) (D) n	n=1		48	40	36
	n=96		300	240	235
MIN (S) (D) n MINP (S) (D) n	n=1		48	40	36
	n=96		300	240	235
DMAX (S) (D) n DMAXP (S) (D) n	n=1		52	43	39
	n=96		600	490	460
DMIN (S) (D) n DMINP (S) (D) n	n=1		52	43	39
	n=96		585	475	445
SORT (S1) n (S2) (D1) (D2)	n=1, (S2)=1		66	55	50
	n=96, (S2)=16		329	270	252
DSORT (S1) n (S2) (D1) (D2)	n=1, (S2)=1		98	57	52
	n=96, (S2)=16		386	317	294

Instruction	Condition (device)	Processing time (μs)		
		Q00JCPU	Q00CPU	Q01CPU
WSUM (S) (D) n WSUMP (S) (D) n	n=1	52	43	40
	n=96	175	140	135
DWSUM (S) (D) n DWSUMP (S) (D) n	n=1	61	51	46
	n=96	515	420	395
FOR n	n=0	11	8.9	8.1
NEXT	—	8.8	7.3	6.8
BREAK BREAKP	—	37	30	28
CALL Pn CALLP Pn	—	17	14	13
CALL Pn (S1) to (S5) CALLP Pn (S1) to (S5)	—	245	200	190
RET	Return to original program	16	13	12
FCALL Pn FCALLP Pn	—	29	24	22
FCALL Pn (S1) to (S5) FCALLP Pn (S1) to (S5)	—	250	205	190
COM	—	110	77	72
IX	—	65	54	51
IXEND	—	30	26	25
IXDEV+IXSET	Number of contacts 1	145	120	110
	Number of contacts 14	770	630	585
FIFW FIFWP	Number of data points 0	36	32	28
	Number of data points 96	36	32	28
FIFR FIFRP	Number of data points 1	45	41	36
	Number of data points 96	93	82	70
FPOP FPOPP	Number of data points 1	40	37	32
	Number of data points 96	40	37	32
FINS FINSP	Number of data points 0	53	44	38
	Number of data points 96	100	89	76
FDEL FDELP	Number of data points 1	60	50	43
	Number of data points 96	110	95	82
FROM n1 n2 (D) n3 FROMP n1 n2 (D) n3*1	n3=1	125	105	93
	n3=1000	740	695	685
DFRO n1 n2 (D) n3 DFROP n1 n2 (D) n3*1	n3=1	130	110	100
	n3=500	745	695	675
TO n1 n2 (S) n3 TOP n1 n2 (S) n3*1	n3=1	120	105	92
	n3=1000	735	680	645
DTO n1 n2 (S) n3 DTOP n1 n2 (S) n3*1	n3=1	130	110	99
	n3=500	740	680	640
LIMIT LIMITP	—	34	28	26
DLIMIT DLIMITP	—	41	34	30
BAND BANDP	—	33	28	25
DBAND DBANDP	—	40	34	30
ZONE ZONEP	—	31	25	24
DZONE DZONEP	—	37	29	28
RSET RSETP	—	—	18	16

Instruction	Condition (device)	Processing time (μs)		
		Q00JCPU	Q00CPU	Q01CPU
DATERD DATERDP	—	30	25	23
DATEWR DATEWRP	—	69	57	54
DATE+ DATE+P	No digit increase	47	39	36
	Digit increase	50	42	38
DATE- DATE - P	No digit increase	47	40	36
	Digit increase	50	42	38
SECOND SECONDP	—	28	24	22
HOUR HOURP	—	38	32	29
WDT WDTP	—	18	15	14
DUTY	—	41	36	32
ZRRDB ZRRDBP	—	—	24	22
ZRWRB ZRWRBP	—	—	27	24
ADRSET ADRSETP	—	23	19	18
ZPUSH ZPUSHP	—	38	33	30
ZPOP ZPOPP	—	37	31	29

*1 The FROM/TO instruction differs in processing time according to the number of slots and the loaded modules. (The CPU also differs in processing time according to the extension base type.)

■Data link instruction

The processing time when the instruction is not executed is calculated as follows:

- Q00JCPU: 0.20 × (No. of steps for each instruction + 1) μs
- Q00CPU: 0.16 × (No. of steps for each instruction + 1) μs
- Q01CPU: 0.10 × (No. of steps for each instruction + 1) μs

Instruction	Condition (device)	Processing time (μs)		
		Q00JCPU	Q00CPU	Q01CPU
ZCOM	—	105	82	80

■Processing time for QCPU instructions (QCPU instructions only)

Instruction	Condition (device)	Processing time (μs)		
		Q00JCPU	Q00CPU	Q01CPU
UNIRD UNIRDP	n=1	96	80	74
	n=16	440	370	340

■Instructions executable by the product with the first five digits of the serial No. "04122" or higher

Instruction	Condition (device)			Processing time (μs)		
				Q00JCPU	Q00CPU	Q01CPU
LDE=	Single precision	In conductive status		43.0	35.5	33.0
		In non-conductive status		46.0	38.0	35.5
ANDE=	Single precision	When not executed		1.5	1.2	1.0
		When executed	In conductive status	35.5	29.5	26.5
			In non-conductive status	42.0	35.0	32.5

Instruction	Condition (device)			Processing time (μs)		
				Q00JCPU	Q00CPU	Q01CPU
ORE=	Single precision	When not executed		1.5	1.2	1.0
		When executed	In conductive status	42.0	35.0	32.5
			In non-conductive status	37.0	31.0	28.5
LDE<>	Single precision	In conductive status		46.0	38.0	35.5
		In non-conductive status		43.5	36.0	33.0
ANDE<>	Single precision	When not executed		1.5	1.2	1.0
		When executed	In conductive status	38.5	31.5	29.0
			In non-conductive status	39.5	33.0	30.5
ORE<>	Single precision	When not executed		1.5	1.2	1.0
		When executed	In conductive status	45.0	37.5	35.0
			In non-conductive status	34.5	29.0	26.5
LDE>	Single precision	In conductive status		46.0	37.5	35.5
		In non-conductive status		46.0	38.5	35.0
ANDE>	Single precision	When not executed		1.5	1.2	1.0
		When executed	In conductive status	38.5	32.0	29.0
			In non-conductive status	42.0	35.0	32.5
ORE>	Single precision	When not executed		1.5	1.2	1.0
		When executed	In conductive status	45.0	37.5	34.5
			In non-conductive status	37.0	31.0	29.0
LDE<=	Single precision	In conductive status		45.5	37.5	35.0
		In non-conductive status		46.5	38.5	35.5
ANDE<=	Single precision	When not executed		1.5	1.2	1.0
		When executed	In conductive status	38.5	31.5	29.0
			In non-conductive status	42.5	35.5	32.5
ORE<=	Single precision	When not executed		1.5	1.2	1.0
		When executed	In conductive status	45.0	37.5	34.5
			In non-conductive status	37.5	31.5	28.5
LDE<	Single precision	In conductive status		45.5	37.5	35.0
		In non-conductive status		46.5	38.5	35.5
ANDE<	Single precision	When not executed		1.5	1.2	1.0
		When executed	In conductive status	38.0	31.5	29.0
			In non-conductive status	42.5	35.5	32.5
ORE<	Single precision	When not executed		1.5	1.2	1.0
		When executed	In conductive status	45.0	37.5	34.5
			In non-conductive status	37.5	31.5	29.0
LDE>=	Single precision	In conductive status		45.5	38.0	35.5
		In non-conductive status		46.5	38.0	35.0
ANDE>=	Single precision	When not executed		1.5	1.2	1.0
		When executed	In conductive status	38.5	32.0	29.0
			In non-conductive status	42.5	35.5	32.5
ORE>=	Single precision	When not executed		1.5	1.2	1.0
		When executed	In conductive status	45.0	38.5	34.5
			In non-conductive status	37.5	31.0	28.5

Instruction	Condition (device)		Processing time (μs)		
			Q00JCPU	Q00CPU	Q01CPU
E+ (S) (D) E+P (S) (D)	Single precision	(S)=0, (D)=0	29.5	25.0	23.0
		(S)=2^{127}, (D)=2^{127}	65.5	60.5	49.5
E+ (S1) (S2) (D) E+P (S1) (S2) (D)	Single precision	(S1)=0, (S2)=0	31.0	27.0	24.0
		(S1)=2^{127}, (S2)=2^{127}	66.5	56.0	51.0
E- (S) (D) E-P (S) (D)	Single precision	(S)=0, (D)=0	29.5	25.0	23.0
		(S)=2^{127}, (D)=2^{127}	48.5	41.0	37.5
E- (S1) (S2) (D) E-P (S1) (S2) (D)	Single precision	(S1)=0, (S2)=0	31.0	27.0	24.0
		(S1)=2^{127}, (S2)=2^{127}	50.5	42.5	38.5
E* (S1) (S2) (D) E*P (S1) (S2) (D)	Single precision	(S1)=0, (S2)=0	30.0	25.5	23.0
		(S1)=2^{127}, (S2)=2^{127}	65.5	55.0	49.5
E/ (S1) (S2) (D) E/P (S1) (S2) (D)	Single precision	(S1)=0, (S2)=1	30.0	26.0	23.0
		(S1)=2^{127}, (S2)=-2^{126}	69.5	57.5	53.0
INT INTP	Single precision	(S)=0	21.5	18.5	16.0
		(S)=32766.5	38.0	32.0	29.5
DINT DINTP	Single precision	(S)=0	23.0	19.5	17.5
		(S)=1234567890.3	42.0	35.5	32.0
FLT FLTP	Single precision	(S)=0	22.5	19.5	17.0
		(S)=7FFFH	26.5	23.0	20.0
DFLT DFLTP	Single precision	(S)=0	23.0	20.0	17.5
		(S)=7FFFFFFFH	26.0	23.5	19.5
ENEG ENEGP	(S)=0		20.5	17.0	15.5
	(S)=E-1.0		31.5	26.0	24.0
EMOV EMOVP	—		1.5	1.2	1.0
ESTR ESTRP	—		604.0	686.0	831.0
EVAL EVALP	Decimal point format all 2-digit specification		138.0	148.0	196.0
	Exponent format all 6-digit specification		164.0	177.0	214.0
SIN SINP	Single precision		204.0	173.0	157.0
COS COSP	Single precision		187.0	158.0	144.0
TAN TANP	Single precision		224.0	190.0	173.0
RAD RADP	Single precision		51.0	43.0	39.0
DEG DEGP	Single precision		51.0	43.0	39.0
SQR SQRP	Single precision		60.0	51.0	46.5
EXP EXPP	Single precision	(S)=-10	306.0	259.0	235.0
		(S)=1	306.0	259.0	235.0
LOG LOGP	Single precision	(S)=1	73.0	61.5	56.0
		(S)=10	301.0	255.0	232.0
RND RNDP	—		12.5	11.0	10.0
SRND SRNDP	—		13.5	12.0	11.0
COM[*1]	With auto refresh of CPU shared memory	Refresh range: 2K words (0.5K words assigned equally to all CPUs)	—	920	880
	Without auto refresh of CPU shared memory	—	—	150	135

Instruction	Condition (device)		Processing time (μs)		
			Q00JCPU	Q00CPU	Q01CPU
FROM	Reading from host CPU shared memory	n3=1	—	100	90
		n3=320	—	440	420
	Reading from other CPU shared memory	n3=1	—	110	105
		n3=320	—	305	290
TO	Writing to host CPU shared memory	n3=1	—	100	95
		n3=320	—	440	425
S.TO	Writing to host CPU shared memory	n4=1	—	205	195
		n4=320	—	545	525

*1 If the processing overlaps those of the other CPUs in a multiple CPU system, the processing time increases by a maximum of the following time.

• For a system having only the main base unit

(Instruction processing time increase) = 4 × 0.54 × (number of points processed) × (number of other CPUs) (μs)

• For a system including extension base units

(Instruction processing time increase) = 4 × 1.30 × (number of points processed) × (number of other CPUs) (μs)

■Table of the time to be added when file register, module access device or link direct device is used

Device name	Data	Device specification location	Processing time (μs)		
			Q00JCPU	Q00CPU	Q01CPU
File register (ZR)	Bit	Source	—	34	32
		Destination	—	23	22
	Word	Source	—	13	12
		Destination	—	9	8
	Double word	Source	—	14	13
		Destination	—	10	9
Module access device (Un\G□, U3En\G0 to G511)	Bit	Source	99	82	77
		Destination	167	137	129
	Word	Source	74	61	58
		Destination	72	60	56
	Double word	Source	76	63	59
		Destination	92	75	71
Link direct device (Jn\□)	Bit	Source	178	147	137
		Destination	303	248	233
	Word	Source	154	126	118
		Destination	153	125	117
	Double word	Source	155	127	119
		Destination	163	133	125

Operation processing time of High Performance model QCPU/ Process CPU/Redundant CPU

The processing time for the individual instructions are shown in the table on the following pages.
Operation processing times can vary substantially depending on the nature of the sources and destinations of the instructions, and the values contained in the following tables should therefore be taken as a set of general guidelines to processing time rather than as being strictly accurate.

Point

When using the file resister (ZR), module access device (Un\G□, U3En\G0 to G4095), or link direct device (Jn\□), add the processing time shown in Page 927 Table of the time to be added when file register, module access device or link direct device is used to that of each instruction.

Processing time

■Sequence instructions

Instruction	Condition (device)				Processing time (μs)			
					Qn	QnH	QnPH	QnPRH
LD LDI AND ANI OR ORI	—				0.079	0.034	0.034	0.034
LDP LDF ANDP ANDF ORP ORF	—				0.158	0.068	0.068	0.068
ANB ORB MPS MRD MPP	—				0.079	0.034	0.034	0.034
INV	When not executed				0.079	0.034	0.034	0.034
	When executed							
MEP MEF	When not executed				0.173	0.073	0.073	0.073
	When executed							
EGP EGF	When not executed	(OFF→OFF) (ON→ON)			0.158	0.068	0.068	0.068
	When executed	(OFF→ON) (ON→OFF)						
OUT	When not changed	(OFF→OFF)			0.158	0.068	0.068	0.068
		(ON→ON)						
	When changed	(OFF→ON)			0.158	0.068	0.068	0.068
		(ON→OFF)						
	F	When not executed			2.8	1.2	1.2	1.2
		When executed			162	69.7	69.7	69.7
	T	When not executed			0.63	0.27	0.27	0.27
		When executed	After time up		0.63	0.27	0.27	0.27
			When added	K	0.63	0.27	0.27	0.27
				D	0.63	0.27	0.27	0.27
	C	When not executed			0.63	0.27	0.27	0.27
		When executed	After time up		0.63	0.27	0.27	0.27
			When added	K	0.63	0.27	0.27	0.27
				D	0.63	0.27	0.27	0.27

Instruction	Condition (device)				Processing time (μs)			
					Qn	QnH	QnPH	QnPRH
OUTH	T	When not executed			0.63	0.27	0.27	0.27
		When executed	After time up		0.63	0.27	0.27	0.27
			When added	K	0.63	0.27	0.27	0.27
				D	0.63	0.27	0.27	0.27
SET	When not executed				0.158	0.068	0.068	0.068
	When executed	When not changed (ON→ON)			0.158	0.068	0.068	0.068
		When changed (OFF→ON)			0.158	0.068	0.068	0.068
	F	When not executed			0.47	0.20	0.20	0.20
		When executed			161	69	69	69
RST	When not executed				0.158	0.068	0.068	0.068
	When executed	When not changed (OFF→OFF)			0.158	0.068	0.068	0.068
		When changed (ON→OFF)			0.158	0.068	0.068	0.068
	SM	When not executed			0.158	0.068	0.068	0.068
		When executed			0.158	0.068	0.068	0.068
	F	When not executed			0.47	0.20	0.20	0.20
		When executed			90	38	38	38
	T, C	When not executed			0.63	0.27	0.27	0.27
		When executed			0.63	0.27	0.27	0.27
	D	When not executed			0.24	0.10	0.10	0.10
		When executed			0.24	0.10	0.10	0.10
	Z	When not executed			0.47	0.20	0.20	0.20
		When executed			4.3	1.9	1.9	1.9
	R	When not executed			0.40	0.17	0.17	0.17
		When executed			0.40	0.17	0.17	0.17
PLS PLF	—				1.0	0.44	0.44	0.44
FF	Y	When not executed			0.47	0.20	0.20	0.20
		When executed			0.47	0.20	0.20	0.20
DELTA DELTAP	DY0	When not executed			0.47	0.20	0.20	0.20
		When executed			5.9	2.6	2.6	2.6
SFT SFTP	When not executed				0.47	0.20	0.20	0.20
	When executed				1.66	0.71	0.71	0.71
MC	—				0.24	0.10	0.10	0.10
MCR	—				0.079	0.034	0.034	0.034
FEND END	Error check performed				380	150	150	500
	No error check performed • Battery check • Fuse blown check • I/O module verification				380	150	150	500
NOP	—				0.079	0.034	0.034	0.034
NOPLF PAGE	—				0.079	0.034	0.034	0.034

■Basic instructions

The processing time when the instruction is not executed is calculated as follows:

- Q02CPU: 0.079 × (No. of steps for each instruction + 1) μs
- Q02HCPU, Q06HCPU, Q12HCPU, Q25HCPU, Q02PHCPU, Q06PHCPU, Q12PHCPU, Q25PHCPU, Q12PRHCPU, Q25PRHCPU: 0.034 × (No. of steps for each instruction + 1) μs

Instruction	Condition (device)		Processing time (μs)			
			Qn	QnH	QnPH	QnPRH
LD=	In conductive status		0.24	0.10	0.10	0.10
	In non-conductive status		0.24	0.10	0.10	0.10
AND=	When not executed		0.24	0.10	0.10	0.10
	When executed	In conductive status	0.24	0.10	0.10	0.10
		In non-conductive status	0.24	0.10	0.10	0.10
OR=	When not executed		0.24	0.10	0.10	0.10
	When executed	In conductive status	0.24	0.10	0.10	0.10
		In non-conductive status	0.24	0.10	0.10	0.10
LD<>	In conductive status		0.24	0.10	0.10	0.10
	In non-conductive status		0.24	0.10	0.10	0.10
AND<>	When not executed		0.24	0.10	0.10	0.10
	When executed	In conductive status	0.24	0.10	0.10	0.10
		In non-conductive status	0.24	0.10	0.10	0.10
OR<>	When not executed		0.24	0.10	0.10	0.10
	When executed	In conductive status	0.24	0.10	0.10	0.10
		In non-conductive status	0.24	0.10	0.10	0.10
LD>	In conductive status		0.24	0.10	0.10	0.10
	In non-conductive status		0.24	0.10	0.10	0.10
AND>	When not executed		0.24	0.10	0.10	0.10
	When executed	In conductive status	0.24	0.10	0.10	0.10
		In non-conductive status	0.24	0.10	0.10	0.10
OR>	When not executed		0.24	0.10	0.10	0.10
	When executed	In conductive status	0.24	0.10	0.10	0.10
		In non-conductive status	0.24	0.10	0.10	0.10
LD<=	In conductive status		0.24	0.10	0.10	0.10
	In non-conductive status		0.24	0.10	0.10	0.10
AND<=	When not executed		0.24	0.10	0.10	0.10
	When executed	In conductive status	0.24	0.10	0.10	0.10
		In non-conductive status	0.24	0.10	0.10	0.10
OR<=	When not executed		0.24	0.10	0.10	0.10
	When executed	In conductive status	0.24	0.10	0.10	0.10
		In non-conductive status	0.24	0.10	0.10	0.10
LD<	In conductive status		0.24	0.10	0.10	0.10
	In non-conductive status		0.24	0.10	0.10	0.10

Instruction	Condition (device)		Processing time (μs)			
			Qn	QnH	QnPH	QnPRH
AND<	When not executed		0.24	0.10	0.10	0.10
	When executed	In conductive status	0.24	0.10	0.10	0.10
		In non-conductive status	0.24	0.10	0.10	0.10
OR<	When not executed		0.24	0.10	0.10	0.10
	When executed	In conductive status	0.24	0.10	0.10	0.10
		In non-conductive status	0.24	0.10	0.10	0.10
LD>=	In conductive status		0.24	0.10	0.10	0.10
	In non-conductive status		0.24	0.10	0.10	0.10
AND>=	When not executed		0.24	0.10	0.10	0.10
	When executed	In conductive status	0.24	0.10	0.10	0.10
		In non-conductive status	0.24	0.10	0.10	0.10
OR>=	When not executed		0.24	0.10	0.10	0.10
	When executed	In conductive status	0.24	0.10	0.10	0.10
		In non-conductive status	0.24	0.10	0.10	0.10
LDD=	In conductive status		0.55	0.24	0.24	0.24
	In non-conductive status		0.39	0.17	0.17	0.17
ANDD=	When not executed		0.39	0.17	0.17	0.17
	When executed	In conductive status	0.55	0.24	0.24	0.24
		In non-conductive status	0.39	0.17	0.17	0.17
ORD=	When not executed		0.39	0.17	0.17	0.17
	When executed	In conductive status	0.55	0.24	0.24	0.24
		In non-conductive status	0.55	0.24	0.24	0.24
LDD<>	In conductive status		0.55	0.24	0.24	0.24
	In non-conductive status		0.55	0.24	0.24	0.24
ANDD<>	When not executed		0.39	0.17	0.17	0.17
	When executed	In conductive status	0.55	0.24	0.24	0.24
		In non-conductive status	0.55	0.24	0.24	0.24
ORD<>	When not executed		0.39	0.17	0.17	0.17
	When executed	In conductive status	0.55	0.24	0.24	0.24
		In non-conductive status	0.55	0.24	0.24	0.24
LDD>	In conductive status		0.55	0.24	0.24	0.24
	In non-conductive status		0.55	0.24	0.24	0.24
ANDD>	When not executed		0.39	0.17	0.17	0.17
	When executed	In conductive status	0.55	0.24	0.24	0.24
		In non-conductive status	0.55	0.24	0.24	0.24

Instruction	Condition (device)		Processing time (μs)			
			Qn	QnH	QnPH	QnPRH
ORD>	When not executed		0.39	0.17	0.17	0.17
	When executed	In conductive status	0.55	0.24	0.24	0.24
		In non-conductive status	0.55	0.24	0.24	0.24
LDD<=	In conductive status		0.55	0.24	0.24	0.24
	In non-conductive status		0.55	0.24	0.24	0.24
ANDD<=	When not executed		0.39	0.17	0.17	0.17
	When executed	In conductive status	0.55	0.24	0.24	0.24
		In non-conductive status	0.55	0.24	0.24	0.24
ORD<=	When not executed		0.39	0.17	0.17	0.17
	When executed	In conductive status	0.55	0.24	0.24	0.24
		In non-conductive status	0.55	0.24	0.24	0.24
LDD<	In conductive status		0.55	0.24	0.24	0.24
	In non-conductive status		0.55	0.24	0.24	0.24
ANDD<	When not executed		0.39	0.17	0.17	0.17
	When executed	In conductive status	0.55	0.24	0.24	0.24
		In non-conductive status	0.55	0.24	0.24	0.24
ORD<	When not executed		0.39	0.17	0.17	0.17
	When executed	In conductive status	0.55	0.24	0.24	0.24
		In non-conductive status	0.55	0.24	0.24	0.24
LDD>=	In conductive status		0.55	0.24	0.24	0.24
	In non-conductive status		0.55	0.24	0.24	0.24
ANDD>=	When not executed		0.39	0.17	0.17	0.17
	When executed	In conductive status	0.55	0.24	0.24	0.24
		In non-conductive status	0.55	0.24	0.24	0.24
ORD>=	When not executed		0.39	0.17	0.17	0.17
	When executed	In conductive status	0.55	0.24	0.24	0.24
		In non-conductive status	0.55	0.24	0.24	0.24
LDE=[*1]	Single precision	In conductive status	93	40	6.4	6.4
			14.9	6.4		
		In non-conductive status	92	40	6.4	6.4
			14.9	6.4		
	Double precision	In conductive status	93	40	—	—
			14.9	6.4		
		In non-conductive status	92	40	—	—
			14.9	6.4		

A

Instruction	Condition (device)			Processing time (μs)			
				Qn	QnH	QnPH	QnPRH
ANDE=[*1]	Single precision	When not executed		0.55	0.24	0.24	0.24
		When executed	In conductive status	93	40	6.4	6.4
				14.9	6.4		
			In non-conductive status	92	40	6.4	6.4
				14.9	6.4		
	Double precision	When not executed		—	—	—	—
		When executed	In conductive status	93	40	—	—
				14.9	6.4		
			In non-conductive status	92	40	—	—
				14.9	6.4		
ORE=[*1]	Single precision	When not executed		0.55	0.24	0.24	0.24
		When executed	In conductive status	93	40	6.4	6.4
				14.9	6.4		
			In non-conductive status	92	40	6.4	6.4
				14.9	6.4		
	Double precision	When not executed		0.55	0.24	—	—
		When executed	In conductive status	93	40	—	—
				14.9	6.4		
			In non-conductive status	92	40	—	—
				14.9	6.4		
LDE<>[*1]	Single precision	In conductive status		92	40	6.4	6.4
				14.9	6.4		
		In non-conductive status		92	40	6.4	6.4
				14.9	6.4		
	Double precision	In conductive status		92	40	—	—
				14.9	6.4		
		In non-conductive status		92	40	—	—
				14.9	6.4		
ANDE<>[*1]	Single precision	When not executed		0.55	0.24	0.24	0.24
		When executed	In conductive status	92	40	6.4	6.4
				14.9	6.4		
			In non-conductive status	93	40	6.4	6.4
				14.9	6.4		
	Double precision	When not executed		0.55	0.24	—	—
		When executed	In conductive status	92	40	—	—
				14.9	6.4		
			In non-conductive status	92	40	—	—
				14.9	6.4		
ORE<>[*1]	Single precision	When not executed		0.55	0.24	0.24	0.24
		When executed	In conductive status	93	40	6.4	6.4
				14.9	6.4		
			In non-conductive status	92	40	6.4	6.4
				14.9	6.4		
	Double precision	When not executed		0.55	0.24	—	—
		When executed	In conductive status	93	40	—	—
				14.9	6.4		
			In non-conductive status	92	40	—	—
				14.9	6.4		

Instruction	Condition (device)			Processing time (μs)			
				Qn	QnH	QnPH	QnPRH
LDE>[*1]	Single precision	In conductive status		92	40	6.4	6.4
				14.9	6.4		
		In non-conductive status		92	40	6.4	6.4
				14.9	6.4		
	Double precision	In conductive status		92	40	—	—
				14.9	6.4		
		In non-conductive status		92	40	—	—
				14.9	6.4		
ANDE>[*1]	Single precision	When not executed		0.55	0.24	0.24	0.24
		When executed	In conductive status	92	40	6.4	6.4
				14.9	6.4		
			In non-conductive status	93	40	6.4	6.4
				14.9	6.4		
	Double precision	When not executed		0.55	0.24	—	—
		When executed	In conductive status	92	40	—	—
				14.9	6.4		
			In non-conductive status	92	40	—	—
				14.9	6.4		
ORE>[*1]	Single precision	When not executed		0.55	0.24	0.24	0.24
		When executed	In conductive status	93	40	6.4	6.4
				14.9	6.4		
			In non-conductive status	92	40	6.4	6.4
				14.9	6.4		
	Double precision	When not executed		0.55	0.24	—	—
		When executed	In conductive status	93	40	—	—
				14.9	6.4		
			In non-conductive status	92	40	—	—
				14.9	6.4		
LDE<=[*1]	Single precision	In conductive status		93	40	6.4	6.4
				14.9	6.4		
		In non-conductive status		92	40	6.4	6.4
				14.9	6.4		
	Double precision	In conductive status		93	40	—	—
				14.9	6.4		
		In non-conductive status		92	40	—	—
				14.9	6.4		
ANDE<=[*1]	Single precision	When not executed		0.55	0.24	0.24	0.24
		When executed	In conductive status	92	40	6.4	6.4
				14.9	6.4		
			In non-conductive status	92	40	6.4	6.4
				14.9	6.4		
	Double precision	When not executed		0.55	0.24	—	—
		When executed	In conductive status	92	40	—	—
				14.9	6.4		
			In non-conductive status	92	40	—	—
				14.9	6.4		

Instruction	Condition (device)			Processing time (μs)			
				Qn	QnH	QnPH	QnPRH
ORE<=[*1]	Single precision	When not executed		0.55	0.24	0.24	0.24
		When executed	In conductive status	92	40	6.4	6.4
				14.9	6.4		
			In non-conductive status	92	40	6.4	6.4
				14.9	6.4		
	Double precision	When not executed		0.55	0.24	—	—
		When executed	In conductive status	92	40	—	—
				14.9	6.4		
			In non-conductive status	92	40	—	—
				14.9	6.4		
LDE<[*1]	Single precision	In conductive status		92	40	6.4	6.4
				14.9	6.4		
		In non-conductive status		92	40	6.4	6.4
				14.9	6.4		
	Double precision	In conductive status		92	40	—	—
				14.9	6.4		
		In non-conductive status		92	40	—	—
				14.9	6.4		
ANDE<[*1]	Single precision	When not executed		0.55	0.24	0.24	0.24
		When executed	In conductive status	92	40	6.4	6.4
				14.9	6.4		
			In non-conductive status	92	40	6.4	6.4
				14.9	6.4		
	Double precision	When not executed		0.55	0.24	—	—
		When executed	In conductive status	92	40	—	—
				14.9	6.4		
			In non-conductive status	92	40	—	—
				14.9	6.4		
ORE<[*1]	Single precision	When not executed		0.55	0.24	0.24	0.24
		When executed	In conductive status	93	40	6.4	6.4
				14.9	6.4		
			In non-conductive status	92	40	6.4	6.4
				14.9	6.4		
	Double precision	When not executed		0.55	0.24	—	—
		When executed	In conductive status	93	40	—	—
				14.9	6.4		
			In non-conductive status	92	40	—	—
				14.9	6.4		
LDE>=[*1]	Single precision	In conductive status		93	40	6.4	6.4
				14.9	6.4		
		In non-conductive status		92	40	6.4	6.4
				14.9	6.4		
	Double precision	In conductive status		93	40	—	—
				14.9	6.4		
		In non-conductive status		92	40	—	—
				14.9	6.4		

Instruction	Condition (device)			Processing time (μs)			
				Qn	QnH	QnPH	QnPRH
ANDE>=[*1]	Single precision	When not executed		0.55	0.24	0.24	0.24
		When executed	In conductive status	92	40	6.4	6.4
				14.9	6.4		
			In non-conductive status	92	40	6.4	6.4
				14.9	6.4		
	Double precision	When not executed		0.55	0.24	—	—
		When executed	In conductive status	92	40	—	—
				14.9	6.4		
			In non-conductive status	92	40	—	—
				14.9	6.4		
ORE>=[*1]	Single precision	When not executed		0.55	0.24	0.24	0.24
		When executed	In conductive status	92	40	6.4	6.4
				14.9	6.4		
			In non-conductive status	92	40	6.4	6.4
				14.9	6.4		
	Double precision	When not executed		0.55	0.24	—	—
		When executed	In conductive status	92	40	—	—
				14.9	6.4		
			In non-conductive status	92	40	—	—
				14.9	6.4		
LD$=	In conductive status			38	16	16	16
	In non-conductive status			34	15	15	15
AND$=	When not executed			0.56	0.23	0.23	0.23
	When executed		In conductive status	39	17	17	17
			In non-conductive status	32	14	14	14
OR$=	When not executed			0.56	0.24	0.24	0.24
	When executed		In conductive status	40	17	17	17
			In non-conductive status	33	14	14	14
LD$<>	In conductive status			32	14	14	14
	In non-conductive status			40	17	17	17
AND$<>	When not executed			0.56	0.23	0.23	0.23
	When executed		In conductive status	33	14	14	14
			In non-conductive status	39	17	17	17
OR$<>	When not executed			0.56	0.24	0.24	0.24
	When executed		In conductive status	32	14	14	14
			In non-conductive status	39	17	17	17
LD$>	In conductive status			32	14	14	14
	In non-conductive status			40	17	17	17
AND$>	When not executed			0.56	0.23	0.23	0.23
	When executed		In conductive status	33	14	14	14
			In non-conductive status	39	17	17	17

Instruction	Condition (device)		Processing time (μs)			
			Qn	QnH	QnPH	QnPRH
OR$>	When not executed		0.56	0.24	0.24	0.24
	When executed	In conductive status	32	14	14	14
		In non-conductive status	39	17	17	17
LD$<=	In conductive status		40	17	17	17
	In non-conductive status		32	14	14	14
AND$<=	When not executed		0.56	0.23	0.23	0.23
	When executed	In conductive status	39	17	17	17
		In non-conductive status	32	14	14	14
OR$<=	When not executed		0.56	0.24	0.24	0.24
	When executed	In conductive status	40	17	17	17
		In non-conductive status	33	14	14	14
LD$<	In conductive status		32	14	14	14
	In non-conductive status		40	17	17	17
AND$<	When not executed		0.56	0.23	0.23	0.23
	When executed	In conductive status	32	14	14	14
		In non-conductive status	39	16	16	16
OR$<	When not executed		0.56	0.24	0.24	0.24
	When executed	In conductive status	32	14	14	14
		In non-conductive status	39	16	16	16
LD$>=	In conductive status		40	17	17	17
	In non-conductive status		32	14	14	14
AND$>=	When not executed		0.56	0.23	0.23	0.23
	When executed	In conductive status	39	16	16	16
		In non-conductive status	32	14	14	14
OR$>=	When not executed		0.56	0.24	0.24	0.24
	When executed	In conductive status	39	17	17	17
		In non-conductive status	32	14	14	14
BKCMP= (S1) (S2) (D) n	n=1		48	21	21	21
BKCMP=P (S1) (S2) (D) n	n=96		142	61	61	61
BKCMP<> (S1) (S2) (D) n	n=1		48	21	21	21
BKCMP<>P (S1) (S2) (D) n	n=96		150	65	65	65
BKCMP> (S1) (S2) (D) n	n=1		48	21	21	21
BKCMP>P (S1) (S2) (D) n	n=96		142	61	61	61
BKCMP>= (S1) (S2) (D) n	n=1		48	21	21	21
BKCMP>=P (S1) (S2) (D) n	n=96		150	65	65	65
BKCMP< (S1) (S2) (D) n	n=1		48	21	21	21
BKCMP<P (S1) (S2) (D) n	n=96		158	68	68	68
BKCMP<= (S1) (S2) (D) n	n=1		48	21	21	21
BKCMP<=P (S1) (S2) (D) n	n=96		150	65	65	65
+ (S) (D) +P (S) (D)	When executed		0.39	0.17	0.17	0.17

Instruction	Condition (device)		Processing time (μs)			
			Qn	QnH	QnPH	QnPRH
+ (S1) (S2) (D) +P (S1) (S2) (D)	When executed		0.47	0.20	0.20	0.20
- (S) (D) -P (S) (D)	When executed		0.39	0.17	0.17	0.17
- (S1) (S2) (D) -P (S1) (S2) (D)	When executed		0.47	0.20	0.20	0.20
D+ (S) (D) D+P (S) (D)	When executed		0.71	0.31	0.31	0.31
D+ (S1) (S2) (D) D+P (S1) (S2) (D)	When executed		0.79	0.34	0.34	0.34
D- (S) (D) D-P (S) (D)	When executed		0.71	0.30	0.30	0.30
D- (S1) (S2) (D) D-P (S1) (S2) (D)	When executed		0.79	0.34	0.34	0.34
* (S1) (S2) (D) *P (S1) (S2) (D)	When executed		0.47	0.20	0.20	0.20
/ (S1) (S2) (D) /P (S1) (S2) (D)	—		2.7	1.2	1.2	1.2
D* (S1) (S2) (D) D*P (S1) (S2) (D)	—		7.9	3.4	3.4	3.4
D/ (S1) (S2) (D) D/P (S1) (S2) (D)	—		14	6.1	6.1	6.1
B+ (S) (D) B+P (S) (D)	—		2.2	1.0	1.0	1.0
B+ (S1) (S2) (D) B+P (S1) (S2) (D)	—		5.0	2.2	2.2	2.2
B- (S) (D) B-P (S) (D)	—		2.0	0.9	0.9	0.9
B- (S1) (S2) (D) B-P (S1) (S2) (D)	—		4.9	2.1	2.1	2.1
DB+ (S) (D) DB+P (S) (D)	—		12	5.0	5.0	5.0
DB+ (S1) (S2) (D) DB+P (S1) (S2) (D)	—		12	5.3	5.3	5.3
DB- (S) (D) DB-P (S) (D)	—		11	4.8	4.8	4.8
DB- (S1) (S2) (D) DB-P (S1) (S2) (D)	—		12	5.2	5.2	5.2
B* (S1) (S2) (D) B*P (S1) (S2) (D)	—		3.7	1.6	1.6	1.6
B/ (S1) (S2) (D) B/P (S1) (S2) (D)	—		3.8	1.6	1.6	1.6
DB* (S1) (S2) (D) DB*P (S1) (S2) (D)	—		24	10	10	10
DB/ (S1) (S2) (D) DB/P (S1) (S2) (D)	—		27	12	12	12
E+ (S) (D) E+P (S) (D)	Single precision	(S)=0, (D)=0	1.8	0.78	0.78	0.78
		(S)=2^{127}, (D)=2^{127}	1.8	0.78	0.78	0.78
	Double precision	(S)=0, (D)=0	203	87	—	—
		(S)=2^{127}, (D)=2^{127}	203	87	—	—
E+ (S1) (S2) (D) E+P (S1) (S2) (D)	Single precision	(S1)=0, (S2)=0	2.4	1.1	1.1	1.1
		(S1)=2^{127}, (S2)=2^{127}	2.4	1.1	1.1	1.1
	Double precision	(S1)=0, (S2)=0	209	90	—	—
		(S1)=2^{127}, (S2)=2^{127}	209	90	—	—
E- (S) (D) E-P (S) (D)	Single precision	(S)=0, (D)=0	1.8	0.78	0.78	0.78
		(S)=2^{127}, (D)=2^{127}	1.8	0.78	0.78	0.78
	Double precision	(S)=0, (D)=0	202	87	—	—
		(S)=2^{127}, (D)=2^{127}	202	87	—	—

Instruction	Condition (device)		Processing time (μs)			
			Qn	QnH	QnPH	QnPRH
E- (S1) (S2) (D) E-P (S1) (S2) (D)	Single precision	(S1)=0, (S2)=0	2.4	1.1	1.1	1.1
		(S1)=2^{127}, (S2)=2^{127}	2.4	1.1	1.1	1.1
	Double precision	(S1)=0, (S2)=0	210	90	—	—
		(S1)=2^{127}, (S2)=2^{127}	210	90	—	—
E* (S1) (S2) (D) E*P (S1) (S2) (D)	Single precision	(S1)=0, (S2)=0	2.4	1.1	1.1	1.1
		(S1)=2^{126}, (S2)=2^{127}	2.4	1.1	1.1	1.1
	Double precision	(S1)=0, (S2)=0	222	96	—	—
		(S1)=2^{126}, (S2)=2^{127}	222	96	—	—
E/ (S1) (S2) (D) E/P (S1) (S2) (D)	Single precision	(S1)=0, (S2)=1	12	5.2	5.2	5.2
		(S1)=2^{127}, (S2)=-2^{126}	12	5.2	5.2	5.2
	Double precision	(S1)=0, (S2)=1	369	159	—	—
		(S1)=2^{127}, (S2)=-2^{126}	369	159	—	—
$+ (S) (D) $+P (S) (D)	—		68	29	29	29
$+ (S1) (S2) (D) $+P (S1) (S2) (D)	—		81	35	35	35
INC INCP	—		0.32	0.14	0.14	0.14
DINC DINCP	—		0.47	0.20	0.20	0.20
DEC DECP	—		0.32	0.14	0.14	0.14
DDEC DDECP	—		0.47	0.20	0.20	0.20
BCD BCDP	—		1.1	0.48	0.48	0.48
DBCD DBCDP	—		3.2	1.4	1.4	1.4
BIN BINP	—		1.0	0.44	0.44	0.44
DBIN DBINP	—		1.9	0.82	0.82	0.82
INT INTP	Single precision	(S)=0	3.2	1.4	1.4	1.4
		(S)=32766.5	3.2	1.4	1.4	1.4
	Double precision	(S)=0	22	9.3	—	—
		(S)=32766.5	22	9.3	—	—
DINT DINTP	Single precision	(S)=0	2.5	1.1	1.1	1.1
		(S)=1234567890.3	2.5	1.1	1.1	1.1
	Double precision	(S)=0	24	10	—	—
		(S)=1234567890.3	24	10	—	—
FLT FLTP	Single precision	(S)=0	2.1	0.92	0.92	0.92
		(S)=7FFFH	2.1	0.92	0.92	0.92
	Double precision	(S)=0	22	9.6	—	—
		(S)=7FFFH	22	9.6	—	—
DFLT DFLTP	Single precision	(S)=0	2.1	0.88	0.88	0.88
		(S)=7FFFFFFFH	2.1	0.88	0.88	0.88
	Double precision	(S)=0	26	11	—	—
		(S)=7FFFFFFFH	26	11	—	—
DBL DBLP	—		4.5	1.9	1.9	1.9
WORD WORDP	—		4.7	2.0	2.0	2.0
GRY GRYP	—		4.7	2.0	2.0	2.0

Instruction	Condition (device)	Processing time (μs)			
		Qn	QnH	QnPH	QnPRH
DGRY DGRYP	—	5.3	2.3	2.3	2.3
GBIN GBINP	—	18	7.7	7.7	7.7
DGBIN DGBINP	—	32	14	14	14
NEG NEGP	—	3.6	1.6	1.6	1.6
DNEG DNEGP	—	4.3	1.8	1.8	1.8
ENEG ENEGP	—	3.9	1.7	1.7	1.7
BKBCD (S) (D) n	n=1	38	17	17	17
BKBCDP (S) (D) n	n=96	99	43	43	43
BKBIN (S) (D) n	n=1	38	17	17	17
BKBINP (S) (D) n	n=96	99	43	43	43
MOV MOVP	(S)=D0, (D)=D1	0.24	0.10	0.10	0.10
	(S)=D0, (D)=J1\W1	140*2	60*2	60*2	60*2
DMOV DMOVP	(S)=D0, (D)=D1	0.47	0.20	0.20	0.20
	(S)=D0, (D)=J1\W1	147*2	64*2	64*2	64*2
EMOV EMOVP	—	0.63	0.27	0.27	0.27
$MOV $MOVP	—	40	17	17	17
CML CMLP	—	0.40	0.17	0.17	0.17
DCML DCMLP	—	0.55	0.24	0.24	0.24
BMOV (S) (D) n	n=1	17	7.1	7.1	7.1
BMOVP (S) (D) n	n=96	32	14	14	14
FMOV (S) (D) n	n=1	6.7	2.9	2.9	2.9
FMOVP (S) (D) n	n=96	14	6.1	6.1	6.1
XCH XCHP DXCH DXCHP	—	1.3	0.54	0.54	0.54
BXCH (D1) (D2) n	n=1	31	13	13	13
BXCHP (D1) (D2) n	n=96	84	36	36	36
SWAP SWAPP	—	3.7	1.6	1.6	1.6
CJ	—	3.2	1.4	1.4	1.4
SCJ	—	3.2	1.4	1.4	1.4
JMP	—	3.2	1.4	1.4	1.4
GOEND	—	0.39	0.34	0.34	0.34
DI	—	0.95	0.41	0.41	0.41
EI	—	1.3	0.54	0.54	0.54
IMASK	—	11	4.6	4.6	4.6
IRET	—	1.6	0.68	0.68	0.68
RFS RFSP	n=1	6.7	4.7	4.7	4.7
	n=96	19	13	13	13
UDCNT1	—	15	6.5	6.5	—
UDCNT2	—	16	6.8	6.8	—
TTMR	—	10	4.4	4.4	—
STMR	—	20	7.1	7.1	—
ROTC	—	26	11	11	—

Instruction	Condition (device)	Processing time (μs)			
		Qn	QnH	QnPH	QnPRH
RAMP	—	18	7.7	7.7	—
SPD	—	19	8.3	8.3	—
PLSY	—	10	4.5	4.5	—
PWM	—	9.1	3.9	3.9	—
MTR	—	11	4.9	4.9	—

*1 The Qn/QnH changes in processing time depending on the serial No. of the CPU module.
Top: The first five digits of the serial No. are "05031" or lower
Bottom: The first five digits of the serial No. are "05032" or higher
For the condition to be satisfied when the instruction is not executed, there is no differentiation between the top and bottom.

*2 This value indicates the processing time when Q312B is used.

■Application instructions

The processing time when the instruction is not executed is calculated as follows:

- Q02CPU: 0.079 × (No. of steps for each instruction + 1) μs
- Q02HCPU, Q06HCPU, Q12HCPU, Q25HCPU, Q02PHCPU, Q06PHCPU, Q12PHCPU, Q25PHCPU, Q12PRHCPU, Q25PRHCPU: 0.034 × (No. of steps for each instruction + 1) μs

Instruction	Condition (device)	Processing time (μs)			
		Qn	QnH	QnPH	QnPRH
WAND (S) (D) WANDP (S) (D)	When executed	0.39	0.17	0.17	0.17
WAND (S1) (S2) (D) WANDP (S1) (S2) (D)	When executed	0.47	0.20	0.20	0.20
DAND (S) (D) DANDP (S) (D)	When executed	0.71	0.31	0.31	0.31
DAND (S1) (S2) (D) DANDP (S1) (S2) (D)	When executed	0.79	0.34	0.34	0.34
BKAND (S1) (S2) (D) n BKANDP (S1) (S2) (D) n	n=1	36	16	16	16
	n=96	74	32	32	32
WOR (S) (D) WORP (S) (D)	When executed	0.40	0.17	0.17	0.17
WOR (S1) (S2) (D) WORP (S1) (S2) (D)	When executed	0.47	0.20	0.20	0.20
DOR (S) (D) DORP (S) (D)	When executed	0.71	0.31	0.31	0.31
DOR (S1) (S2) (D) DORP (S1) (S2) (D)	When executed	0.79	0.34	0.34	0.34
BKOR (S1) (S2) (D) n BKORP (S1) (S2) (D) n	n=1	36	16	16	16
	n=96	74	32	32	32
WXOR (S) (D) WXORP (S) (D)	When executed	0.39	0.17	0.17	0.17
WXOR (S1) (S2) (D) WXORP (S1) (S2) (D)	When executed	0.47	0.20	0.20	0.20
DXOR (S) (D) DXORP (S) (D)	When executed	0.71	0.31	0.31	0.31
DXOR (S1) (S2) (D) DXORP (S1) (S2) (D)	When executed	0.79	0.34	0.34	0.34
BKXOR (S1) (S2) (D) n BKXORP (S1) (S2) (D) n	n=1	36	16	16	16
	n=96	74	32	32	32
WXNR (S) (D) WXNRP (S) (D)	When executed	0.40	0.17	0.17	0.17
WXNR (S1) (S2) (D) WXNRP (S1) (S2) (D)	When executed	0.47	0.20	0.20	0.20
DXNR (S) (D) DXNRP (S) (D)	When executed	0.71	0.31	0.31	0.31
DXNR (S1) (S2) (D) DXNRP (S1) (S2) (D)	When executed	0.79	0.34	0.34	0.34
BKXNR (S1) (S2) (D) n BKXNRP (S1) (S2) (D) n	n=1	36	16	16	16
	n=96	74	32	32	32
ROR (D) n RORP (D) n	n=1	2.0	0.85	0.85	0.85
	n=15	2.0	0.85	0.85	0.85
RCR (D) n RCRP (D) n	n=1	1.6	0.68	0.68	0.68
	n=15	1.6	0.68	0.68	0.68
ROL (D) n ROLP (D) n	n=1	2.0	0.85	0.85	0.85
	n=15	2.0	0.85	0.85	0.85
RCL (D) n RCLP (D) n	n=1	1.6	0.68	0.68	0.68
	n=15	1.6	0.68	0.68	0.68
DROR (D) n DRORP (D) n	n=1	3.9	1.7	1.7	1.7
	n=31	4.0	1.7	1.7	1.7

Instruction	Condition (device)		Processing time (μs)			
			Qn	QnH	QnPH	QnPRH
DRCR (D) n DRCRP (D) n	n=1		4.3	1.8	1.8	1.8
	n=31		4.3	1.9	1.9	1.9
DROL (D) n DROLP (D) n	n=1		3.9	1.7	1.7	1.7
	n=31		4.0	1.7	1.7	1.7
DRCL (D) n DRCLP (D) n	n=1		4.3	1.8	1.8	1.8
	n=31		4.3	1.9	1.9	1.9
SFR (D) n SFRP (D) n	n=1		1.7	0.75	0.75	0.75
	n=15		2.0	0.85	0.85	0.85
SFL (D) n SFLP (D) n	n=1		1.7	0.75	0.75	0.75
	n=15		2.0	0.85	0.85	0.85
BSFR (D) n BSFRP (D) n	n=1		20	8.6	8.6	8.6
	n=96		24	10	10	10
BSFL (D) n BSFLP (D) n	n=1		20	8.5	8.5	8.5
	n=96		23	10	10	10
DSFR (D) n DSFRP (D) n	n=1		1.3	0.58	0.58	0.58
	n=96		25	11	11	11
DSFL (D) n DSFLP (D) n	n=1		1.3	0.58	0.58	0.58
	n=96		26	11	11	11
BSET (D) n BSETP (D) n	n=1		7.6	3.3	3.3	3.3
	n=15		7.6	3.3	3.3	3.3
BRST (D) n BRSTP (D) n	n=1		7.6	3.3	3.3	3.3
	n=15		7.6	3.3	3.3	3.3
TEST (S1) (S2) (D) TESTP (S1) (S2) (D)	—		8.2	3.5	3.5	3.5
DTEST (S1) (S2) (D) DTESTP (S1) (S2) (D)	—		9.2	3.9	3.9	3.9
BKRST (D) n BKRSTP (D) n	n=1		18	7.8	7.8	7.8
	n=96		19	8.2	8.2	8.2
SER (S1) (S2) (D) n SERP (S1) (S2) (D) n	n=1	All match	22	9.6	9.6	9.6
		None match	21	8.9	8.9	8.9
	n=96	All match	115	49	49	49
		None match	133	57	57	57
DSER (S1) (S2) (D) n DSERP (S1) (S2) (D) n	n=1	All match	23	9.9	9.9	9.9
		None match	23	9.7	9.7	9.7
	n=96	All match	142	61	61	61
		None match	132	57	57	57
SUM SUMP	(S)=0 (S)=FFFF		3.9	1.7	1.7	1.7
DSUM DSUMP	(S)=0		4.7	2.0	2.0	2.0
	(S)=FFFFFFFFH		12	5.0	5.0	5.0
DECO (S) (D) n DECOP (S) (D) n	n=2		20	8.6	8.6	8.6
	n=8		27	12	12	12
ENCO (S) (D) n ENCOP (S) (D) n	n=2	M1=ON	21	9.1	9.1	9.1
		M4=ON	21	9.1	9.1	9.1
	n=8	M1=ON	28	12	12	12
		M256=ON	26	11	11	11
SEG SEGP	—		1.3	0.54	0.54	0.54
DIS (S) (D) n DISP (S) (D) n	n=1		18	7.7	7.7	7.7
	n=4		19	8.3	8.3	8.3
UNI (S) (D) n UNIP (S) (D) n	n=1		21	8.9	8.9	8.9
	n=4		23	9.7	9.7	9.7

Instruction	Condition (device)	Processing time (μs)			
		Qn	QnH	QnPH	QnPRH
NDIS (S1) (D) (S2) NDISP (S1) (D) (S2)	—	41	18	18	18
NUNI (S1) (D) (S2) NUNIP (S1) (D) (S2)	—	42	18	18	18
WTOB (S) (D) n WTOBP (S) (D) n	n=1	47	20	20	20
	n=96	99	43	43	43
BTOW (S) (D) n BTOWP (S) (D) n	n=1	45	19	19	19
	n=96	89	38	38	38
MAX (S) (D) n MAXP (S) (D) n	n=1	17	7.1	7.1	7.1
	n=96	136	59	59	59
MIN (S) (D) n MINP (S) (D) n	n=1	17	7.1	7.1	7.1
	n=96	159	69	69	69
DMAX (S) (D) n DMAXP (S) (D) n	n=1	27	12	12	12
	n=96	181	78	78	78
DMIN (S) (D) n DMINP (S) (D) n	n=1	27	12	12	12
	n=96	112	48	48	48
SORT (S1) n (S2) (D1) (D2)	n=1, (S2)=1	16	7.1	7.1	7.1
	n=96, (S2)=16	87.8	37.9	37.9	37.9
DSORT (S1) n (S2) (D1) (D2)	n=1, (S2)=1	17	7.1	7.1	7.1
	n=96, (S2)=16	96.1	41.6	41.6	41.6
WSUM (S) (D) n WSUMP (S) (D) n	n=1	16.4	7.1	7.1	7.1
	n=96	68.4	29.5	29.5	29.5
DWSUM (S) (D) n DWSUMP (S) (D) n	n=1	18.9	8.2	8.2	8.2
	n=96	130.4	56.1	56.1	56.1
FOR n	n=0	2.3	1.0	1.0	1.0
NEXT	—	3.3	1.4	1.4	1.4
BREAK BREAKP	—	11	4.6	4.6	4.6
CALL Pn CALLP Pn	Internal file pointer	2.1	0.88	0.88	0.88
	Common pointer	33	14	14	14
CALL Pn (S1) to (S5) CALLP Pn (S1) to (S5)	—	135	58	58	58
RET	Return to original program	2.9	1.3	1.3	1.3
	Return to other program	20	8.5	8.5	8.5
FCALL Pn FCALLP Pn	Internal file pointer	3.6	1.6	1.6	1.6
	Common pointer	20	8.7	8.7	8.7
FCALL Pn (S1) to (S5) FCALLP Pn (S1) to (S5)	—	134	57	57	57
ECALL * Pn ECALLP * Pn *: Program name	—	77	33	33	33
ECALL * Pn (S1) to (S5) ECALLP * Pn (S1) to (S5) *: Program name	—	162	70	70	70
EFCALL * Pn EFCALLP * Pn *: Program name	—	78	34	34	34
EFCALL * Pn (S1) to (S5) EFCALLP * Pn (S1) to (S5) *: Program name	—	200	86	86	86
COM	—	55	16	16	16
IX	—	12	5.2	5.2	5.2
IXEND	—	4.7	2.0	2.0	2.0

Instruction	Condition (device)		Processing time (μs)			
			Qn	QnH	QnPH	QnPRH
IXDEV+IXSET	Number of contacts 1		48	21	21	21
	Number of contacts 14		93	40	40	40
FIFW FIFWP	Number of data points 0		11	4.5	4.5	4.5
	Number of data points 96		11	4.5	4.5	4.5
FIFR FIFRP	Number of data points 1		13	5.6	5.6	5.6
	Number of data points 96		32	14	14	14
FPOP FPOPP	Number of data points 1		16	7.0	7.0	7.0
	Number of data points 96		16	7.0	7.0	7.0
FINS FINSP	Number of data points 0		20	8.4	8.4	8.4
	Number of data points 96		36	15	15	15
FDEL FDELP	Number of data points 1		19	7.5	7.5	7.5
	Number of data points 96		39	15	15	15
FROM n1 n2 (D) n3 FROMP n1 n2 (D) n3[*1]	n3=1		47	22	22	22
	n3=1000		476	437	437	437
DFRO n1 n2 (D) n3 DFROP n1 n2 (D) n3[*1]	n3=1		51	24	24	24
	n3=500		478	437	437	437
TO n1 n2 (S) n3 TOP n1 n2 (S) n3[*1]	n3=1		48	20	20	20
	n3=1000		479	412	412	412
DTO n1 n2 (S) n3 DTOP n1 n2 (S) n3[*1]	n3=1		50	23	23	23
	n3=500		457	416	416	416
PR	SM701ON	Variable 1 character	33	11	11	—
		Variable 32 character	48	18	18	—
	SM701OFF		21	7.8	7.8	—
PRC	—		181	16	16	—
LEDR	No display → no display		0.40	0.17	0.17	0.17
	LED instruction execution → no display		103	44	44	44
CHKST	—		5.8	2.5	2.5	2.5
CHK	1 contact no error		24	10	10	10
	150 contact no error		1676	721	721	721
	1 contact error		88	38	38	38
CHKCIR	10 steps		5.8	2.5	2.5	2.5
BINDA BINDAP	(S)=1		15	6.7	6.7	6.7
	(S)=-32768		24	10	10	10
DBINDA DBINDAP	(S)=1		43	18	18	18
	(S)=-2147483648		86	37	37	37
BINHA BINHAP	(S)=1		18	7.7	7.7	7.7
	(S)=FFFFH		19	8.2	8.2	8.2
DBINHA DBINHAP	(S)=1		23	10	10	10
	(S)=FFFFFFFFH		24	10	10	10
BCDDA BCDDAP	(S)=1		23	9.8	9.8	9.8
	(S)=9999		21	8.9	8.9	8.9
DBCDDA DBCDDAP	(S)=1		22	9.5	9.5	9.5
	(S)=99999999		29	13	13	13
DABIN DABINP	(S)=1		57	25	25	25
	(S)=-32768		58	25	25	25
DDABIN DDABINP	(S)=1		92	40	40	40
	(S)=-2147483648		106	46	46	46
HABIN HABINP	(S)=1		13	5.8	5.8	5.8
	(S)=FFFFH		15	6.4	6.4	6.4

Instruction	Condition (device)		Processing time (μs)			
			Qn	QnH	QnPH	QnPRH
DHABIN DHABINP	(S)=1		22	9.5	9.5	9.5
	(S)=FFFFFFFFH		25	11	11	11
DABCD DABCDP	(S)=1		16	6.9	6.9	6.9
	(S)=9999		17	7.2	7.2	7.2
DDABCD DDABCDP	(S)=1		25	11	11	11
	(S)=99999999		29	13	13	13
COMRD COMRDP	—		40	17	17	17
LEN LENP	1 character		18	8.0	8.0	8.0
	96 characters		86	37	37	37
STR STRP	—		53	23	23	23
DSTR DSTRP	—		123	53	53	53
VAL VALP	—		95	41	41	41
DVAL DVALP	—		166	72	72	72
ESTR ESTRP	—		564	243	243	243
EVAL EVALP	Decimal point format all 2-digit specification		100	43	43	43
	Exponent format all 6-digit specification		127	55	55	55
ASC (S) (D) n ASCP (S) (D) n	n=1		64	28	28	28
	n=96		289	125	125	125
HEX (S) (D) n HEXP (S) (D) n	n=1		60	26	26	26
	n=96		343	148	148	148
RIGHT (S) (D) n RIGHTP (S) (D) n	n=1		49	21	21	21
	n=96		131	56	56	56
LEFT (S) (D) n LEFTP (S) (D) n	n=1		50	21	21	21
	n=96		131	56	56	56
MIDR MIDRP	—		53	23	23	23
MIDW MIDWP	—		128	55	55	55
INSTR INSTRP	No match		58	25	25	25
	Match	Head	55	24	24	24
		End	58	25	25	25
EMOD EMODP	—		527	227	227	227
EREXP EREXPP	—		1656	713	713	713
SIN SINP	Single precision		115	50	50	50
	Double precision		1945	837	—	—
COS COSP	Single precision		122	53	53	53
	Double precision		2618	1127	—	—
TAN TANP	Single precision		123	53	53	53
	Double precision		2618	1127	—	—
ASIN ASINP	Single precision		111	48	48	48
	Double precision		2491	1072	—	—
ACOS ACOSP	Single precision		115	49	49	49
	Double precision		2367	1019	—	—
ATAN ATANP	Single precision		157	68	68	68
	Double precision		3140	1352	—	—

Instruction	Condition (device)		Processing time (μs)			
			Qn	QnH	QnPH	QnPRH
RAD RADP	Single precision		17	7.2	7.2	7.2
	Double precision		24	10	—	—
DEG DEGP	Single precision		17	7.2	7.2	7.2
	Double precision		23	9.9	—	—
SQR SQRP	Single precision		28	12	12	12
	Double precision		1812	780	—	—
EXP EXPP	Single precision	(S)=-10	129	56	56	56
		(S)=1				
	Double precision	(S)=-10	2386	1026	—	—
		(S)=1				
LOG LOGP	Single precision	(S)=1	113	49	49	49
		(S)=10				
	Double precision	(S)=1	2146	924	—	—
		(S)=10				
RND RNDP	—		3.9	1.7	1.7	1.7
SRND SRNDP	—		3.5	1.5	1.5	1.5
BSQR BSQRP	(S)=0		6.2	2.7	2.7	2.7
	(S)=9999		38	16	16	16
BDSQR BDSQRP	(S)=0		6.2	2.7	2.7	2.7
	(S)=99999999		38	16	16	16
BSIN BSINP	—		12	5.1	5.1	5.1
BCOS BCOSP	—		12	5.2	5.2	5.2
BTAN BTANP	—		12	5.2	5.2	5.2
BASIN BASINP	—		20	8.7	8.7	8.7
BACOS BACOSP	—		21	9.0	9.0	9.0
BATAN BATANP	—		22	9.6	9.6	9.6
LIMIT LIMITP	—		10	4.3	4.3	4.3
DLIMIT DLIMITP	—		11	4.7	4.7	4.7
BAND BANDP	—		9.8	4.2	4.2	4.2
DBAND DBANDP	—		11	4.9	4.9	4.9
ZONE ZONEP	—		9.1	3.9	3.9	3.9
DZONE DZONEP	—		11	4.6	4.6	4.6
RSET RSETP	—		6.8	2.9	2.9	2.9
QDRSET QDRSETP	—		205	88	88	88
QCDSET QCDSETP	—		147	63	63	63
DATERD DATERDP	—		13	5.5	5.5	5.5
DATEWR DATEWRP	—		15	6.4	6.4	6.4

Instruction	Condition (device)	Processing time (μs)			
		Qn	QnH	QnPH	QnPRH
DATE+ DATE+P	No digit increase	13	5.4	5.4	5.4
	Digit increase	13	5.4	5.4	5.4
DATE- DATE - P	No digit increase	12	5.2	5.2	5.2
	Digit increase	12	5.2	5.2	5.2
SECOND SECONDP	—	10	4.5	4.5	4.5
HOUR HOURP	—	12	5.2	5.2	5.2
MSG	1 character	3.0	1.3	1.3	1.3
	32 characters	3.0	1.3	1.3	1.3
PKEY	Initial time	20	8.6	8.6	8.6
	No reception	19	8.2	8.2	8.2
PSTOP PSTOPP	—	79	34	34	34
POFF POFFP	—	79	34	34	34
PSCAN PSCNAP	—	75	32	32	32
PLOW PLOWP	—	80	34	34	—
WDT WDTP	—	5.9	2.6	2.6	2.6
DUTY	—	9.3	4.0	4.0	4.0
ZRRDB ZRRDBP	—	7.9	3.4	3.4	3.4
ZRWRB ZRWRBP	—	9.4	4.0	4.0	4.0
ADRSET ADRSETP	—	4.9	2.1	2.1	2.1
KEY	—	17	7.3	7.3	—
ZPUSH ZPUSHP	—	11	4.7	4.7	4.7
ZPOP ZPOPP	—	5.1	2.2	2.2	2.2

*1 This value indicates the processing times taken when the Q312B is used to execute the instruction for the QJ71C24 in slot 0.
The FROM/TO instruction differs in processing time according to the number of slots and the loaded modules.
(The QnCPU/QnHCPU also differs in processing time according to the extension base type.)

• Instructions available from function version A

Instruction	Condition (device)		Processing time (μs)			
			Qn	QnH	QnPH	QnPRH
UNIRD	—		79	34	34	34
TRACE	Start		176	76	76	76
	STRA execution completion		6.3	2.7	2.7	2.7
TRACER	—		19	8.2	8.2	8.2
SP.FWRITE	—		84	36	36	36
SP.FREAD	—		82	35	35	35
PLOADP	—		58	25	25	—
PUNLOADP	—		272	117	117	—
PSWAPP	—		308	133	133	—
RBMOV	When standard RAM is used	1 point	45.5	20	20	20
		1000 points	215	91	91	91
	When SRAM card is used	1 point	49.5	22	22	22
		1000 points	540	305	305	305

• Instructions available from function version B

Instruction	Condition/number of points processed			Processing time (μs)			
				Qn	QnH	QnPH	QnPRH
COM[*2]	With auto refresh of CPU shared memory	Refresh range: 2K words (0.5K words assigned equally to all CPUs)		720	660	660	—
		Refresh range: 4K words (1K words assigned equally to all CPUs)		860	730	730	—
	Without auto refresh of CPU shared memory	—		43	20	20	20
FROM[*2]	Reading from other CPU shared memory	n3=1		59	29	29	—
		n3=1000		530	500	500	—
	Reading buffer memory of intelligent function module[*3]	n3=1	Main base unit	51	24	24	—
			Extension base unit	54	27	27	—
		n3=1000	Main base unit	540	480	480	—
			Extension base unit	1100	1050	1050	—
S.TO	Write to CPU shared memory of host CPU	n2=1		74	33	33	—
		n2=256		126	54	54	—
S (P). DATERD[*4]	Reading expansion clock data	—		25	11	11	11
S (P). DATE+[*4]	Expansion clock data addition operation	—		38	17	17	17
S (P). DATE-[*4]	Expansion clock data subtraction operation	—		38	17	17	17

*2 If the processing overlaps those of the other CPUs in a multiple CPU system, the processing time increases by a maximum of the following time.

• For a system having only the main base unit

(Instruction processing time increase) = 0.54 × (number of points processed) × (number of other CPUs) (μs)

• For a system including extension base units

(Instruction processing time increase) = 1.30 × (number of points processed) × (number of other CPUs) (μs)

*3 In a multiple CPU system, the instruction processing time for the intelligent function module under control of the host CPU is equal to that for the intelligent function module under control of another CPU.

*4 Products with the first five digits of the serial No. "07032" or higher are applicable.

■Data link instruction

The processing time when the instruction is not executed is calculated as follows:

• Q02CPU: 0.079 × (No. of steps for each instruction + 1) μs

• Q02HCPU, Q06HCPU, Q12HCPU, Q25HCPU, Q02PHCPU, Q06PHCPU, Q12PHCPU, Q25PHCPU, Q12PRHCPU, Q25PRHCPU: 0.034 × (No. of steps for each instruction + 1) μs

Instruction	Condition (device)	Processing time (μs)			
		Qn	QnH	QnPH	QnPRH
ZCOM	—	691	289	289	289

■Redundant system instructions (for redundant CPU)

Instruction	Condition (device)	Processing time (μs)			
		Qn	QnH	QnPH	QnPRH
SP.CONTSW	—	—	—	—	9.6

■Table of the time to be added when file register, module access device or link direct device is used

Device name		Data	Device specification location	Processing time (μs)			
				QnCPU	QnHCPU	QnPHCPU	QnPRHCPU
File register (ZR)	When standard RAM is used	Bit	Source	5.56	2.40	2.40	2.40
			Destination	4.44	1.91	1.91	1.91
		Word	Source	2.60	1.12	1.12	1.12
			Destination	3.76	1.62	1.62	1.62
		Double word	Source	2.83	1.22	1.22	1.22
			Destination	4.00	1.72	1.72	1.72
	When SRAM card is used (Q2MEM-1MBS, Q2MEM-2MBS)	Bit	Source	5.22	2.25	2.25	2.25
			Destination	4.09	1.76	1.76	1.76
		Word	Source	2.25	0.97	0.97	0.97
			Destination	3.42	1.47	1.47	1.47
		Double word	Source	2.49	1.07	1.07	1.07
			Destination	3.65	1.57	1.57	1.57
	When SRAM card is used (Q3MEM-4MBS)	Bit	Source	5.16	2.2	2.2	2.2
			Destination	4.05	1.73	1.73	1.73
		Word	Source	2.23	0.92	0.92	0.92
			Destination	3.34	1.44	1.44	1.44
		Double word	Source	2.37	0.98	0.98	0.98
			Destination	3.57	1.5	1.5	1.5
Module access device (Un\G□, U3En\G0 to G4095)		Bit	Source	35.56	15.31	15.31	15.31
			Destination	65.08	28.01	28.01	28.01
		Word	Source	32.76	14.10	14.10	14.10
			Destination	28.84	12.41	12.41	12.41
		Double word	Source	32.99	14.20	14.20	14.20
			Destination	29.07	12.51	12.51	12.51
Link direct device (Jn\□)		Bit	Source	75.67	32.57	32.57	32.57
			Destination	138.65	59.67	59.67	59.67
		Word	Source	72.73	31.30	31.30	31.30
			Destination	137.32	59.10	59.10	59.10
		Double word	Source	72.96	31.40	31.40	31.40
			Destination	137.55	59.20	59.20	59.20

A

Operation processing time of Universal model QCPU

The processing time for the individual instructions are shown in the table on the following pages.
Operation processing times can vary substantially depending on the nature of the sources and destinations of the instructions, and the values contained in the following tables should therefore be taken as a set of general guidelines to processing time rather than as being strictly accurate.

Subset instruction processing time

The following describes the subset instruction processing time.

Point

- The processing time shown in "Subset instruction processing time table" applies when the device used in an instruction meets the device condition for subset processing. (For device condition triggering subset processing, refer to Page 109 Subset processing.)
- When using the file resister (R, ZR), extended data register (D), extended link register (W), or module access device (U3En\G10000 and later), add the processing time shown in Page 950 Table of the time to be added when file register, extended data register, extended link register, and module access device are used to that of each instruction.
- When using the F, T(ST), or C device with the OUT, SET, or RST instruction, add the processing time shown in Page 953 Table of the time to be added when F/T(ST)/C device is used in OUT/SET/RST instruction to that of each instruction.
- Since the processing time of an instruction varies depending on that of the cash function, both the minimum and maximum values are described in the table.

■Subset instruction processing time table

- When using Q00UJCPU, Q00UCPU, Q01UCPU and Q02UCPU

Category	Instruction	Condition (device)		Processing time (μs)							
				Q00UJCPU		Q00UCPU		Q01UCPU		Q02UCPU	
				Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.
Sequence instruction	LD LDI AND ANI OR ORI LDP LDF ANDP ANDF ORP ORF	When executed		0.120		0.080		0.060		0.040	
	LDPI LDFI	When executed		0.360		0.240		0.180		0.120	
	ANDPI ANDFI ORPI ORFI	When executed		0.480		0.320		0.240		0.160	
	OUT	When not changed		0.120		0.080		0.060		0.040	
		When changed									
	SET RST	When not executed		0.120		0.080		0.060		0.040	
		When executed	When not changed								
			When changed								

Category	Instruction	Condition (device)		Processing time (μs)							
				Q00UJCPU		Q00UCPU		Q01UCPU		Q02UCPU	
				Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.
Basic instruction	LD=	In conductive status		0.360		0.240		0.180		0.120	
		In non-conductive status									
	AND=	When not executed		0.360		0.240		0.180		0.120	
		When executed	In conductive status								
			In non-conductive status								
	OR=	When not executed		0.360		0.240		0.180		0.120	
		When executed	In conductive status								
			In non-conductive status								
	LD<>	In conductive status		0.360		0.240		0.180		0.120	
		In non-conductive status									
	AND<>	When not executed		0.360		0.240		0.180		0.120	
		When executed	In conductive status								
			In non-conductive status								
	OR<>	When not executed		0.360		0.240		0.180		0.120	
		When executed	In conductive status								
			In non-conductive status								
	LD>	In conductive status		0.360		0.240		0.180		0.120	
		In non-conductive status									
	AND>	When not executed		0.360		0.240		0.180		0.120	
		When executed	In conductive status								
			In non-conductive status								
	OR>	When not executed		0.360		0.240		0.180		0.120	
		When executed	In conductive status								
			In non-conductive status								
	LD<=	In conductive status		0.360		0.240		0.180		0.120	
		In non-conductive status									
	AND<=	When not executed		0.360		0.240		0.180		0.120	
		When executed	In conductive status								
			In non-conductive status								
	OR<=	When not executed		0.360		0.240		0.180		0.120	
		When executed	In conductive status								
			In non-conductive status								

Category	Instruction	Condition (device)		Processing time (μs)							
				Q00UJCPU		Q00UCPU		Q01UCPU		Q02UCPU	
				Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.
Basic instruction	LD<	In conductive status		0.360		0.240		0.180		0.120	
		In non-conductive status									
	AND<	When not executed		0.360		0.240		0.180		0.120	
		When executed	In conductive status								
			In non-conductive status								
	OR<	When not executed		0.360		0.240		0.180		0.120	
		When executed	In conductive status								
			In non-conductive status								
	LD>=	In conductive status		0.360		0.240		0.180		0.120	
		In non-conductive status									
	AND>=	When not executed		0.360		0.240		0.180		0.120	
		When executed	In conductive status								
			In non-conductive status								
	OR>=	When not executed		0.360		0.240		0.180		0.120	
		When executed	In conductive status								
			In non-conductive status								
	LDD=	In conductive status		0.360		0.240		0.180		0.120	
		In non-conductive status									
	ANDD=	When not executed		0.360		0.240		0.180		0.120	
		When executed	In conductive status								
			In non-conductive status								
	ORD=	When not executed		0.360		0.240		0.180		0.120	
		When executed	In conductive status								
			In non-conductive status								
	LDD<>	In conductive status		0.360		0.240		0.180		0.120	
		In non-conductive status									
	ANDD<>	When not executed		0.360		0.240		0.180		0.120	
		When executed	In conductive status								
			In non-conductive status								
	ORD<>	When not executed		0.360		0.240		0.180		0.120	
		When executed	In conductive status								
			In non-conductive status								

Category	Instruction	Condition (device)		Processing time (μs)							
				Q00UJCPU		Q00UCPU		Q01UCPU		Q02UCPU	
				Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.
Basic instruction	LDD>	In conductive status		0.360		0.240		0.180		0.120	
		In non-conductive status									
	ANDD>	When not executed		0.360		0.240		0.180		0.120	
		When executed	In conductive status								
			In non-conductive status								
	ORD>	When not executed		0.360		0.240		0.180		0.120	
		When executed	In conductive status								
			In non-conductive status								
	LDD<=	In conductive status		0.360		0.240		0.180		0.120	
		In non-conductive status									
	ANDD<=	When not executed		0.360		0.240		0.180		0.120	
		When executed	In conductive status								
			In non-conductive status								
	ORD<=	When not executed		0.360		0.240		0.180		0.120	
		When executed	In conductive status								
			In non-conductive status								
	LDD<	In conductive status		0.360		0.240		0.180		0.120	
		In non-conductive status									
	ANDD<	When not executed		0.360		0.240		0.180		0.120	
		When executed	In conductive status								
			In non-conductive status								
	ORD<	When not executed		0.360		0.240		0.180		0.120	
		When executed	In conductive status								
			In non-conductive status								
	LDD>=	In conductive status		0.360		0.240		0.180		0.120	
		In non-conductive status									
	ANDD>=	When not executed		0.360		0.240		0.180		0.120	
		When executed	In conductive status								
			In non-conductive status								
	ORD>=	When not executed		0.360		0.240		0.180		0.120	
		When executed	In conductive status								
			In non-conductive status								

Category	Instruction	Condition (device)		Processing time (μs)							
				Q00UJCPU		Q00UCPU		Q01UCPU		Q02UCPU	
				Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.
Basic instruction	+ (S) (D)	When executed		0.360		0.240		0.180		0.120	
	+ (S1) (S2) (D)	When executed		0.480		0.320		0.240		0.160	
	- (S) (D)	When executed		0.360		0.240		0.180		0.120	
	- (S1) (S2) (D)	When executed		0.480		0.320		0.240		0.160	
	D+ (S) (D)	When executed		0.360		0.240		0.180		0.120	
	D+ (S1) (S2) (D)	When executed		0.480		0.320		0.240		0.160	
	D- (S) (D)	When executed		0.360		0.240		0.180		0.120	
	D- (S1) (S2) (D)	When executed		0.480		0.320		0.240		0.160	
	* (S1) (S2) (D)	When executed		0.420		0.300		0.240		0.180	
	/ (S1) (S2) (D)	When executed		0.520		0.400		0.340		0.280	
	D* (S1) (S2) (D)	When executed		0.500		0.380		0.320		0.260	
	D/ (S1) (S2) (D)	When executed		0.640		0.520		0.460		0.400	
	B+ (S) (D)	When executed		3.100	12.300	3.100	12.300	3.100	12.300	3.300	8.300
	B+ (S1) (S2) (D)	When executed		5.900	13.500	5.900	13.500	5.900	13.500	4.600	6.200
	B- (S) (D)	When executed		3.150	12.300	3.150	12.300	3.150	12.300	3.300	9.000
	B- (S1) (S2) (D)	When executed		5.950	13.600	5.950	13.600	5.950	13.600	4.600	8.200
	B* (S1) (S2) (D)	When executed		3.700	12.100	3.700	12.100	3.700	12.100	4.000	8.200
	B/ (S1) (S2) (D)	When executed		4.000	14.000	4.000	14.000	4.000	14.000	4.200	12.400
	E+ (S) (D)	Single precision	(S)=0, (D)=0	0.420		0.300		0.240		0.180	
			(S)=2^{127}, (D)=2^{127}	0.420		0.300		0.240		0.180	
	E+ (S1) (S2) (D)	Single precision	(S1)=0, (S2)=0	0.540		0.380		0.300		0.220	
			(S1)=2^{127}, (S2)=2^{127}	0.540		0.380		0.300		0.220	
	E- (S) (D)	Single precision	(S)=0, (D)=0	0.420		0.300		0.240		0.180	
			(S)=2^{127}, (D)=2^{127}	0.420		0.300		0.240		0.180	
	E- (S1) (S2) (D)	Single precision	(S1)=0, (S2)=0	0.540		0.380		0.300		0.220	
			(S1)=2^{127}, (S2)=2^{127}	0.540		0.380		0.300		0.220	
	E* (S1) (S2) (D)	Single precision	(S1)=0, (S2)=0	0.420		0.300		0.240		0.180	
			(S1)=2^{127}, (S2)=2^{127}	0.420		0.300		0.240		0.180	
	E/ (S1) (S2) (D)	Single precision	(S1)=2^{127}, (S2)=2^{127}	4.900	18.900	4.900	18.900	4.900	18.900	5.100	14.100
	INC	When executed		0.240		0.160		0.120		0.080	
	DINC	When executed		0.240		0.160		0.120		0.080	
	DEC	When executed		0.240		0.160		0.120		0.080	
	DDEC	When executed		0.240		0.160		0.120		0.080	
	BCD	When executed		0.320		0.240		0.200		0.160	
	DBCD	When executed		0.400		0.320		0.280		0.240	
	BIN	When executed		0.260		0.180		0.140		0.100	
	DBIN	When executed		0.260		0.180		0.140		0.100	
	FLT	Single precision	(S)=0	0.300		0.220		0.180		0.140	
			(S)=7FFFH	0.300		0.220		0.180		0.140	
	DFLT	Single precision	(S)=0	0.300		0.220		0.180		0.140	
			(S)=7FFFFFFFH	0.300		0.220		0.180		0.140	

Category	Instruction	Condition (device)		Processing time (μs)							
				Q00UJCPU		Q00UCPU		Q01UCPU		Q02UCPU	
				Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.
Basic instruction	INT	Single precision	(S)=0	0.300		0.220		0.180		0.140	
			(S)=32766.5	0.300		0.220		0.180		0.140	
	DINT	Single precision	(S)=0	0.300		0.220		0.180		0.140	
			(S)=1234567890.3	0.300		0.220		0.180		0.140	
	MOV	—		0.240		0.160		0.120		0.080	
	DMOV	—		0.240		0.160		0.120		0.080	
	EMOV	—		0.240		0.160		0.120		0.080	
	CML	—		0.240		0.160		0.120		0.080	
	DCML	—		0.240		0.160		0.120		0.080	
	BMOV	SM237=ON	n=1	4.200	4.600	4.200	4.600	4.200	4.600	4.100	4.500
			n=96	4.850	5.150	4.850	5.150	4.850	5.150	4.700	5.100
		SM237=OFF	n=1	6.800	11.300	6.800	11.300	6.800	11.300	6.300	8.900
			n=96	7.450	11.900	7.450	11.900	7.450	11.900	5.900	9.500
	FMOV	SM237=ON	n=1	4.100	4.600	4.100	4.600	4.100	4.600	4.100	4.600
			n=96	4.800	5.200	4.800	5.200	4.800	5.200	4.800	5.200
		SM237=OFF	n=1	4.600	8.250	4.600	8.250	4.600	8.250	4.600	7.900
			n=96	6.150	10.600	6.150	10.600	6.150	10.600	5.300	8.500
	XCH	—		2.250	8.100	2.250	8.100	2.250	8.100	2.500	6.000
	DXCH	—		2.400	8.200	2.400	8.200	2.400	8.200	2.800	7.900
	DFMOV	SM237=ON	n=1	2.700	2.800	2.700	2.800	2.700	2.800	2.350	2.450
			n=96	6.500	6.800	6.500	6.800	6.500	6.800	5.950	6.000
		SM237=OFF	n=1	4.000	8.150	4.000	8.150	4.000	8.150	3.000	6.950
			n=96	8.000	12.200	8.000	12.200	8.000	12.200	6.600	10.600
	CJ	—		3.500	10.100	3.500	10.100	3.500	10.100	1.900	10.100
	SCJ	—		3.500	10.100	3.500	10.100	3.500	10.100	1.900	10.100
	JMP	—		3.500	10.100	3.500	10.100	3.500	10.100	1.900	10.100
Application instruction	WAND (S) (D)	When executed		0.360		0.240		0.180		0.120	
	WAND (S1) (S2) (D)	When executed		0.480		0.320		0.240		0.160	
	DAND (S) (D)	When executed		0.360		0.240		0.180		0.120	
	DAND (S1) (S2) (D)	When executed		0.480		0.320		0.240		0.160	
	WOR (S) (D)	When executed		0.360		0.240		0.180		0.120	
	WOR (S1) (S2) (D)	When executed		0.480		0.320		0.240		0.160	
	DOR (S) (D)	When executed		0.360		0.240		0.180		0.120	
	DOR (S1) (S2) (D)	When executed		0.480		0.320		0.240		0.160	
	WXOR (S) (D)	When executed		0.360		0.240		0.180		0.120	
	WXOR (S1) (S2) (D)	When executed		0.480		0.320		0.240		0.160	
	DXOR (S) (D)	When executed		0.360		0.240		0.180		0.120	
	DXOR (S1) (S2) (D)	When executed		0.480		0.320		0.240		0.160	
	WXNR (S) (D)	When executed		0.360		0.240		0.180		0.120	
	WXNR (S1) (S2) (D)	When executed		0.480		0.320		0.240		0.160	
	DXNR (S) (D)	When executed		0.360		0.240		0.180		0.120	
	DXNR (S1) (S2) (D)	When executed		0.480		0.320		0.240		0.160	

Category	Instruction	Condition (device)	Processing time (μs)							
			Q00UJCPU		Q00UCPU		Q01UCPU		Q02UCPU	
			Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.
Application instruction	ROR (D) n	n=1	2.250	10.800	2.250	10.800	2.250	10.800	2.300	7.800
		n=15	2.250	10.800	2.350	10.800	2.350	10.800	2.400	7.800
	RCR (D) n	n=1	2.250	10.800	2.250	10.800	2.250	10.800	2.300	3.900
		n=15	2.250	10.800	2.250	10.800	2.250	10.800	2.400	4.100
	ROL (D) n	n=1	2.250	10.800	2.350	10.800	2.350	10.800	2.500	4.600
		n=15	2.250	10.800	2.350	10.800	2.350	10.800	2.400	4.600
	RCL (D) n	n=1	2.250	11.500	2.300	11.500	2.300	11.500	2.400	7.500
		n=15	2.250	11.500	2.300	11.500	2.300	11.500	2.500	7.500
	DROR (D) n	n=1	2.350	11.500	2.350	11.500	2.350	11.500	2.400	10.300
		n=31	2.350	11.500	2.350	11.500	2.350	11.500	2.500	10.300
	DRCR (D) n	n=1	2.350	13.300	2.350	13.300	2.350	13.300	2.500	12.700
		n=31	2.350	14.900	2.350	14.900	2.350	14.900	2.500	12.700
	DROL (D) n	n=1	2.350	10.800	2.350	10.800	2.350	10.800	2.500	11.800
		n=31	2.350	10.800	2.350	10.800	2.350	10.800	2.500	11.800
	DRCL (D) n	n=1	2.350	13.300	2.350	13.300	2.350	13.300	2.500	5.100
		n=31	2.350	13.300	2.350	13.300	2.350	13.300	2.500	5.100
	SFR (D) n	n=1	2.350	9.900	2.350	9.900	2.350	9.900	2.400	6.100
		n=15	2.350	9.900	2.350	9.900	2.350	9.900	2.300	5.700
	SFL (D) n	n=1	2.350	9.850	2.350	9.850	2.350	9.850	2.400	4.300
		n=15	2.350	9.850	2.350	9.850	2.350	9.850	2.400	4.300
	DSFR (D) n	n=1	3.250	15.500	3.250	15.500	3.250	15.500	3.300	12.000
		n=96	32.600	45.000	32.600	45.000	32.600	45.000	32.600	42.200
	DSFL (D) n	n=1	3.200	15.500	3.200	15.500	3.200	15.500	3.300	8.200
		n=96	32.600	45.100	32.600	45.100	32.600	45.100	32.600	37.700
	SUM	(S)=0	3.100	8.950	3.100	8.950	3.100	8.950	3.400	6.700
		(S)=FFFFH	3.000	8.850	3.000	8.850	3.000	8.850	3.500	6.700
	SEG	When executed	2.100	7.700	2.100	7.700	2.100	7.700	2.100	5.900
	FOR	—	1.500	7.500	1.500	7.500	1.500	7.500	1.200	6.300
	CALL Pn	Internal file pointer	4.800	5.400	4.800	5.400	4.800	5.400	2.700	4.800
		Common pointer	7.100	30.500	7.100	30.500	7.100	30.500	4.400	5.700
	CALL Pn (S1) to (S5)	—	50.200	62.000	50.200	62.000	50.200	62.000	28.700	42.600

Point

For the instructions for which a rising edge instruction (□P) is not described, the processing time is the same as an ON execution instruction.
(Example) MOVP instruction, WANDP instruction etc.

• When using Q03UD(E)CPU, Q04UD(E)HCPU, Q06UD(E)HCPU, Q10UD(E)HCPU, Q13UD(E)HCPU, Q20UD(E)HCPU, Q26UD(E)HCPU, Q50UDEHCPU, and Q100UDEHCPU

Category	Instruction	Condition (device)		Processing time (μs)							
				Q03UD(E)CPU		Q04UD(E)HCPU, Q06UD(E)HCPU		Q10UD(E)HCPU, Q13UD(E)HCPU, Q20UD(E)HCPU, Q26UD(E)HCPU		Q50UDEHCPU, Q100UDEHCPU	
				Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.
Sequence instruction	LD LDI AND ANI OR ORI LDP LDF ANDP ANDF ORP ORF	When executed		0.020		0.0095		0.0095		0.0095	
	LDPI LDFI	When executed		0.060		0.0285		0.0285		0.0285	
	ANDPI ANDFI ORPI ORFI	When executed		0.080		0.038		0.038		0.038	
	OUT	When not changed		0.020		0.0095		0.0095		0.0095	
		When changed									
	SET RST	When not executed		0.020		0.0095		0.0095		0.0095	
		When executed	When not changed								
			When changed								
Basic instruction	LD=	In conductive status		0.060		0.0285		0.0285		0.0285	
		In non-conductive status									
	AND=	When not executed		0.060		0.0285		0.0285		0.0285	
		When executed	In conductive status								
			In non-conductive status								
	OR=	When not executed		0.060		0.0285		0.0285		0.0285	
		When executed	In conductive status								
			In non-conductive status								
	LD<>	In conductive status		0.060		0.0285		0.0285		0.0285	
		In non-conductive status									

A

Category	Instruction	Condition (device)		Processing time (μs)							
				Q03UD(E)CPU		Q04UD(E)HCPU, Q06UD(E)HCPU		Q10UD(E)HCPU, Q13UD(E)HCPU, Q20UD(E)HCPU, Q26UD(E)HCPU		Q50UDEHCPU, Q100UDEHCPU	
				Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.
Basic instruction	AND<>	When not executed		0.060		0.0285		0.0285		0.0285	
		When executed	In conductive status								
			In non-conductive status								
	OR<>	When not executed		0.060		0.0285		0.0285		0.0285	
		When executed	In conductive status								
			In non-conductive status								
	LD>	In conductive status		0.060		0.0285		0.0285		0.0285	
		In non-conductive status									
	AND>	When not executed		0.060		0.0285		0.0285		0.0285	
		When executed	In conductive status								
			In non-conductive status								
	OR>	When not executed		0.060		0.0285		0.0285		0.0285	
		When executed	In conductive status								
			In non-conductive status								
	LD<=	In conductive status		0.060		0.0285		0.0285		0.0285	
		In non-conductive status									
	AND<=	When not executed		0.060		0.0285		0.0285		0.0285	
		When executed	In conductive status								
			In non-conductive status								
	OR<=	When not executed		0.060		0.0285		0.0285		0.0285	
		When executed	In conductive status								
			In non-conductive status								
	LD<	In conductive status		0.060		0.0285		0.0285		0.0285	
		In non-conductive status									
	AND<	When not executed		0.060		0.0285		0.0285		0.0285	
		When executed	In conductive status								
			In non-conductive status								

Category	Instruction	Condition (device)		Processing time (μs)							
				Q03UD(E)CPU		Q04UD(E)HCPU, Q06UD(E)HCPU		Q10UD(E)HCPU, Q13UD(E)HCPU, Q20UD(E)HCPU, Q26UD(E)HCPU		Q50UDEHCPU, Q100UDEHCPU	
				Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.
Basic instruction	OR<	When not executed		0.060		0.0285		0.0285		0.0285	
		When executed	In conductive status								
			In non-conductive status								
	LD>=	In conductive status		0.060		0.0285		0.0285		0.0285	
		In non-conductive status									
	AND>=	When not executed		0.060		0.0285		0.0285		0.0285	
		When executed	In conductive status								
			In non-conductive status								
	OR>=	When not executed		0.060		0.0285		0.0285		0.0285	
		When executed	In conductive status								
			In non-conductive status								
	LDD=	In conductive status		0.060		0.0285		0.0285		0.0285	
		In non-conductive status									
	ANDD=	When not executed		0.060		0.0285		0.0285		0.0285	
		When executed	In conductive status								
			In non-conductive status								
	ORD=	When not executed		0.060		0.0285		0.0285		0.0285	
		When executed	In conductive status								
			In non-conductive status								
	LDD<>	In conductive status		0.060		0.0285		0.0285		0.0285	
		In non-conductive status									
	ANDD<>	When not executed		0.060		0.0285		0.0285		0.0285	
		When executed	In conductive status								
			In non-conductive status								
	ORD<>	When not executed		0.060		0.0285		0.0285		0.0285	
		When executed	In conductive status								
			In non-conductive status								
	LDD>	In conductive status		0.060		0.0285		0.0285		0.0285	
		In non-conductive status									

A

Category	Instruction	Condition (device)		Processing time (μs)							
				Q03UD(E)CPU		Q04UD(E)HCPU, Q06UD(E)HCPU		Q10UD(E)HCPU, Q13UD(E)HCPU, Q20UD(E)HCPU, Q26UD(E)HCPU		Q50UDEHCPU, Q100UDEHCPU	
				Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.
Basic instruction	ANDD>	When not executed		0.060		0.0285		0.0285		0.0285	
		When executed	In conductive status								
			In non-conductive status								
	ORD>	When not executed		0.060		0.0285		0.0285		0.0285	
		When executed	In conductive status								
			In non-conductive status								
	LDD<=	In conductive status		0.060		0.0285		0.0285		0.0285	
		In non-conductive status									
	ANDD<=	When not executed		0.060		0.0285		0.0285		0.0285	
		When executed	In conductive status								
			In non-conductive status								
	ORD<=	When not executed		0.060		0.0285		0.0285		0.0285	
		When executed	In conductive status								
			In non-conductive status								
	LDD<	In conductive status		0.060		0.0285		0.0285		0.0285	
		In non-conductive status									
	ANDD<	When not executed		0.060		0.0285		0.0285		0.0285	
		When executed	In conductive status								
			In non-conductive status								
	ORD<	When not executed		0.060		0.0285		0.0285		0.0285	
		When executed	In conductive status								
			In non-conductive status								
	LDD>=	In conductive status		0.060		0.0285		0.0285		0.0285	
		In non-conductive status									
	ANDD>=	When not executed		0.060		0.0285		0.0285		0.0285	
		When executed	In conductive status								
			In non-conductive status								

Category	Instruction	Condition (device)		Processing time (μs)							
				Q03UD(E)CPU		Q04UD(E)HCPU, Q06UD(E)HCPU		Q10UD(E)HCPU, Q13UD(E)HCPU, Q20UD(E)HCPU, Q26UD(E)HCPU		Q50UDEHCPU, Q100UDEHCPU	
				Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.
Basic instruction	ORD>=	When not executed		0.060		0.0285		0.0285		0.0285	
		When executed	In conductive status								
			In non-conductive status								
	+ (S) (D)	When executed		0.060		0.0285		0.0285		0.0285	
	+ (S1) (S2) (D)	When executed		0.080		0.038		0.038		0.038	
	- (S) (D)	When executed		0.060		0.0285		0.0285		0.0285	
	- (S1) (S2) (D)	When executed		0.080		0.038		0.038		0.038	
	D+ (S) (D)	When executed		0.060		0.0285		0.0285		0.0285	
	D+ (S1) (S2) (D)	When executed		0.080		0.038		0.038		0.038	
	D- (S) (D)	When executed		0.060		0.0285		0.0285		0.0285	
	D- (S1) (S2) (D)	When executed		0.080		0.038		0.038		0.038	
	* (S1) (S2) (D)	When executed		0.120		0.057		0.057		0.057	
	/ (S1) (S2) (D)	When executed		0.220		0.110		0.110		0.110	
	D* (S1) (S2) (D)	When executed		0.200		0.095		0.095		0.095	
	D/ (S1) (S2) (D)	When executed		0.340		0.170		0.170		0.170	
	B+ (S) (D)	When executed		3.300	5.500	3.000	4.100	3.000	4.100	3.000	4.100
	B+ (S1) (S2) (D)	When executed		4.600	6.200	4.200	5.900	4.200	5.900	4.200	5.900
	B- (S) (D)	When executed		3.300	4.400	2.900	3.800	2.900	3.800	2.900	3.800
	B- (S1) (S2) (D)	When executed		4.600	6.300	4.200	4.600	4.200	4.600	4.200	4.600
	B* (S1) (S2) (D)	When executed		4.000	4.800	3.400	4.800	3.400	4.800	3.400	4.800
	B/ (S1) (S2) (D)	When executed		4.200	5.700	3.700	5.200	3.700	5.200	3.700	5.200
	E+ (S) (D)	Single precision	(S)=0, (D)=0	0.120		0.057		0.057		0.057	
			(S)=2^{127}, (D)=2^{127}	0.120		0.057		0.057		0.057	
	E+ (S1) (S2) (D)	Single precision	(S1)=0, (S2)=0	0.140		0.0665		0.0665		0.0665	
			(S1)=2^{127}, (S2)=2^{127}	0.140		0.0665		0.0665		0.0665	
	E- (S) (D)	Single precision	(S)=0, (D)=0	0.120		0.057		0.057		0.057	
			(S)=2^{127}, (D)=2^{127}	0.120		0.057		0.057		0.057	
	E- (S1) (S2) (D)	Single precision	(S1)=0, (S2)=0	0.140		0.0665		0.0665		0.0665	
			(S1)=2^{127}, (S2)=2^{127}	0.140		0.0665		0.0665		0.0665	
	E* (S1) (S2) (D)	Single precision	(S1)=0, (S2)=0	0.120		0.057		0.057		0.057	
			(S1)=2^{127}, (S2)=2^{127}	0.120		0.057		0.057		0.057	
	E/ (S1) (S2) (D)	Single precision	(S1)=2^{127}, (S2)=2^{127}	4.500	5.600	3.900	4.900	0.285		0.285	
	INC	When executed		0.040		0.019		0.019		0.019	
	DINC	When executed		0.040		0.019		0.019		0.019	
	DEC	When executed		0.040		0.019		0.019		0.019	
	DDEC	When executed		0.040		0.019		0.019		0.019	

A

Category	Instruction	Condition (device)		Processing time (μs)							
				Q03UD(E)CPU		Q04UD(E)HCPU, Q06UD(E)HCPU		Q10UD(E)HCPU, Q13UD(E)HCPU, Q20UD(E)HCPU, Q26UD(E)HCPU		Q50UDEHCPU, Q100UDEHCPU	
				Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.
Basic instruction	BCD	When executed		0.120		0.057		0.057		0.057	
	DBCD	When executed		0.200		0.095		0.095		0.095	
	BIN	When executed		0.060		0.0285		0.0285		0.0285	
	DBIN	When executed		0.060		0.0285		0.0285		0.0285	
	FLT	Single precision	(S)=0	0.100		0.0475		0.0475		0.0475	
			(S)=7FFFH	0.100		0.0475		0.0475		0.0475	
	DFLT	Single precision	(S)=0	0.100		0.0475		0.0475		0.0475	
			(S)=7FFFFFFFH	0.100		0.0475		0.0475		0.0475	
	INT	Single precision	(S)=0	0.100		0.0475		0.0475		0.0475	
			(S)=32766.5	0.100		0.0475		0.0475		0.0475	
	DINT	Single precision	(S)=0	0.100		0.0475		0.0475		0.0475	
			(S)=1234567890.3	0.100		0.0475		0.0475		0.0475	
	MOV	—		0.040		0.019		0.019		0.019	
	DMOV	—		0.040		0.019		0.019		0.019	
	EMOV	—		0.040		0.019		0.019		0.019	
	CML	—		0.040		0.019		0.019		0.019	
	DCML	—		0.040		0.019		0.019		0.019	
	BMOV	n=1	—	6.300	8.200	5.400	7.000	5.400	7.000	5.400	7.000
			SM237=OFF*1	8.200	10.600	3.900	5.100	3.900	5.100	3.900	5.100
			SM237=ON*1	6.000	7.800	2.900	3.700	2.900	3.700	2.900	3.700
		n=96	—	7.100	8.800	5.900	7.600	5.900	7.600	5.900	7.600
			SM237=OFF*1	9.300	11.900	4.400	5.700	4.400	5.700	4.400	5.700
			SM237=ON*1	7.100	9.100	3.400	4.300	3.400	4.300	3.400	4.300
	FMOV	n=1	—	5.300	5.900	4.200	4.800	4.200	4.800	4.200	4.800
			SM237=OFF*1	7.000	8.000	3.400	3.800	3.400	3.800	3.400	3.800
			SM237=ON*1	5.900	6.800	2.800	3.200	2.800	3.200	2.800	3.200
		n=96	—	5.300	7.600	4.400	6.800	4.400	6.800	4.400	6.800
			SM237=OFF*1	7.400	12.200	3.600	5.800	3.600	5.800	3.600	5.800
			SM237=ON*1	6.300	11.000	3.000	5.200	3.000	5.200	3.000	5.200
	XCH	—		2.500	2.900	1.800	2.300	1.800	2.300	1.800	2.300
	DXCH	—		2.800	3.700	2.100	2.900	2.100	2.900	2.100	2.900
	DFMOV*2	n=1	SM237=OFF	2.600	3.750	2.250	3.150	2.250	3.150	2.250	3.150
			SM237=ON	2.050	2.250	1.750	1.750	1.750	1.750	1.750	1.750
		n=96	SM237=OFF	5.850	7.350	4.200	5.500	4.200	5.500	5.380	7.440
			SM237=ON	5.300	6.000	3.650	4.150	3.650	4.150	4.700	5.500
	CJ	—		1.800	2.800	1.400	2.400	1.400	2.400	1.400	2.400
	SCJ	—		1.800	2.800	1.400	2.400	1.400	2.400	1.400	2.400
	JMP	—		1.800	2.800	1.100	2.400	1.100	2.400	1.100	2.400
Application instruction	WAND (S) (D)	When executed		0.060		0.0285		0.0285		0.0285	
	WAND (S1) (S2) (D)	When executed		0.080		0.038		0.038		0.038	
	DAND (S) (D)	When executed		0.060		0.0285		0.0285		0.0285	
	DAND (S1) (S2) (D)	When executed		0.080		0.038		0.038		0.038	
	WOR (S) (D)	When executed		0.060		0.0285		0.0285		0.0285	

Category	Instruction	Condition (device)	Processing time (μs)							
			Q03UD(E)CPU		Q04UD(E)HCPU, Q06UD(E)HCPU		Q10UD(E)HCPU, Q13UD(E)HCPU, Q20UD(E)HCPU, Q26UD(E)HCPU		Q50UDEHCPU, Q100UDEHCPU	
			Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.
Application instruction	WOR (S1) (S2) (D)	When executed	0.080		0.038		0.038		0.038	
	DOR (S) (D)	When executed	0.060		0.0285		0.0285		0.0285	
	DOR (S1) (S2) (D)	When executed	0.080		0.038		0.038		0.038	
	WXOR (S) (D)	When executed	0.060		0.0285		0.0285		0.0285	
	WXOR (S1) (S2) (D)	When executed	0.080		0.038		0.038		0.038	
	DXOR (S) (D)	When executed	0.060		0.0285		0.0285		0.0285	
	DXOR (S1) (S2) (D)	When executed	0.080		0.038		0.038		0.038	
	WXNR (S) (D)	When executed	0.060		0.0285		0.0285		0.0285	
	WXNR (S1) (S2) (D)	When executed	0.080		0.038		0.038		0.038	
	DXNR (S) (D)	When executed	0.060		0.0285		0.0285		0.0285	
	DXNR (S1) (S2) (D)	When executed	0.080		0.038		0.038		0.038	
	ROR (D) n	n=1	2.300	3.100	1.700	2.500	1.700	2.500	1.700	2.500
		n=15	2.400	3.100	1.800	2.500	1.800	2.500	1.800	2.500
	RCR (D) n	n=1	2.300	3.900	1.700	3.200	1.700	3.200	1.700	3.200
		n=15	2.400	4.100	1.700	3.200	1.700	3.200	1.700	3.200
	ROL (D) n	n=1	2.400	3.300	1.800	3.200	1.800	3.200	1.800	3.200
		n=15	2.400	3.300	1.800	3.200	1.800	3.200	1.800	3.200
	RCL (D) n	n=1	2.400	2.700	1.800	2.100	1.800	2.100	1.800	2.100
		n=15	2.400	2.800	1.800	2.200	1.800	2.200	1.800	2.200
	DROR (D) n	n=1	2.400	3.400	1.900	2.700	1.900	2.700	1.900	2.700
		n=31	2.500	3.400	1.900	2.700	1.900	2.700	1.900	2.700
	DRCR (D) n	n=1	2.500	4.800	1.900	4.200	1.900	4.200	1.900	4.200
		n=31	2.500	4.900	1.900	4.200	1.900	4.200	1.900	4.200
	DROL (D) n	n=1	2.500	3.900	1.800	3.200	1.800	3.200	1.800	3.200
		n=31	2.500	3.900	1.800	3.300	1.800	3.300	1.800	3.300
	DRCL (D) n	n=1	2.500	4.800	1.900	3.800	1.900	3.800	1.900	3.800
		n=31	2.500	4.600	1.900	3.800	1.900	3.800	1.900	3.800
	SFR (D) n	n=1	2.400	3.900	1.700	2.600	1.700	2.600	1.700	2.600
		n=15	2.300	3.900	1.800	2.600	1.800	2.600	1.800	2.600
	SFL (D) n	n=1	2.400	4.300	1.800	2.700	1.800	2.700	1.800	2.700
		n=15	2.400	4.300	1.800	2.700	1.800	2.700	1.800	2.700
	DSFR (D) n	n=1	2.700	4.800	2.200	4.300	2.200	4.300	2.200	4.300
		n=96	32.600	35.900	23.900	26.100	23.900	26.100	23.900	26.100
	DSFL (D) n	n=1	2.700	4.600	2.100	4.000	2.100	4.000	2.100	4.000
		n=96	32.600	35.300	23.700	25.800	23.700	25.800	23.700	25.800
	SUM	(S)=0	3.400	4.300	2.900	3.600	2.900	3.600	2.900	3.600
		(S)=FFFFH	3.500	4.200	2.900	3.600	2.900	3.600	2.900	3.600
	SEG	When executed	2.100	2.800	1.500	2.100	1.500	2.100	1.500	2.100
	FOR	—	1.200	2.400	0.870	2.100	0.870	2.100	0.870	2.100
	CALL Pn	Internal file pointer	2.600	4.000	2.300	3.600	2.300	3.600	2.300	3.600
		Common pointer	4.000	5.300	3.200	4.900	3.200	4.900	3.200	4.900
	CALL Pn (S1) to (S5)	—	28.700	33.400	26.100	29.300	26.100	29.300	26.100	29.300

*1 Can be used only for the Q03UDCPU, Q04UDHCPU, and Q06UDHCPU whose first five digits of serial number is "10012" or later.
*2 Can be used only for the Q03UD(E)CPU, Q04UD(E)HCPU, Q06UD(E)HCPU, Q13UD(E)HCPU, and Q26UD(E)HCPU whose first five digits of serial number is "10102" or later.

Point

For the instructions for which a rising edge instruction (□P) is not described, the processing time is the same as an ON execution instruction.
(Example) MOVP instruction, WANDP instruction etc.

• When using Q03UDVCPU, Q04UDVCPU, Q04UDPVCPU, Q06UDVCPU, Q06UDPVCPU, Q13UDVCPU, Q13UDPVCPU, Q26UDVCPU, and Q26UDPVCPU

Category	Instruction	Condition (device)		Processing time (μs)					
				Q03UDVCPU		Q04UDVCPU, Q04UDPVCPU		Q06UDVCPU, Q06UDPVCPU, Q13UDVCPU, Q13UDPVCPU, Q26UDVCPU, Q26UDPVCPU	
				Min.	Max.	Min.	Max.	Min.	Max.
Sequence instruction	LD LDI AND ANI OR ORI	When executed		0.0019	0.0078	0.0019	0.0078	0.0019	0.0078
	LDP LDF ANDP ANDF ORP ORF	When executed		0.0058	0.015	0.0058	0.015	0.0058	0.015
	LDPI LDFI	When executed		0.0058	0.015	0.0058	0.015	0.0058	0.015
	ANDPI ANDFI ORPI ORFI	When executed		0.0058	0.015	0.0058	0.015	0.0058	0.015
	OUT	When not changed		0.0039	0.0078	0.0039	0.0078	0.0039	0.0078
		When changed		0.0039	0.0078	0.0039	0.0078	0.0039	0.0078
	SET RST	When not executed		0.0039	0.0078	0.0039	0.0078	0.0039	0.0078
		When executed	When not changed	0.0039	0.0078	0.0039	0.0078	0.0039	0.0078
			When changed	0.0039	0.0078	0.0039	0.0078	0.0039	0.0078
Basic instruction	LD=	In conductive status		0.0098	0.023	0.0098	0.023	0.0098	0.023
		In non-conductive status		0.0098	0.023	0.0098	0.023	0.0098	0.023
	AND=	When not executed		0.0078	0.023	0.0078	0.023	0.0078	0.023
		When executed	In conductive status	0.0098	0.023	0.0098	0.023	0.0098	0.023
			In non-conductive status	0.0098	0.023	0.0098	0.023	0.0098	0.023
	OR=	When not executed		0.0078	0.023	0.0078	0.023	0.0078	0.023
		When executed	In conductive status	0.0098	0.023	0.0098	0.023	0.0098	0.023
			In non-conductive status	0.0098	0.023	0.0098	0.023	0.0098	0.023
	LD<>	In conductive status		0.0098	0.023	0.0098	0.023	0.0098	0.023
		In non-conductive status		0.0098	0.023	0.0098	0.023	0.0098	0.023
	AND<>	When not executed		0.0078	0.023	0.0078	0.023	0.0078	0.023
		When executed	In conductive status	0.0098	0.023	0.0098	0.023	0.0098	0.023
			In non-conductive status	0.0098	0.023	0.0098	0.023	0.0098	0.023
	OR<>	When not executed		0.0078	0.023	0.0078	0.023	0.0078	0.023
		When executed	In conductive status	0.0098	0.023	0.0098	0.023	0.0098	0.023
			In non-conductive status	0.0098	0.023	0.0098	0.023	0.0098	0.023

Category	Instruction	Condition (device)		Processing time (μs)					
				Q03UDVCPU		Q04UDVCPU, Q04UDPVCPU		Q06UDVCPU, Q06UDPVCPU, Q13UDVCPU, Q13UDPVCPU, Q26UDVCPU, Q26UDPVCPU	
				Min.	Max.	Min.	Max.	Min.	Max.
Basic instruction	LD>	In conductive status		0.0098	0.023	0.0098	0.023	0.0098	0.023
		In non-conductive status		0.0098	0.023	0.0098	0.023	0.0098	0.023
	AND>	When not executed		0.0078	0.023	0.0078	0.023	0.0078	0.023
		When executed	In conductive status	0.0098	0.023	0.0098	0.023	0.0098	0.023
			In non-conductive status	0.0098	0.023	0.0098	0.023	0.0098	0.023
	OR>	When not executed		0.0078	0.023	0.0078	0.023	0.0078	0.023
		When executed	In conductive status	0.0098	0.023	0.0098	0.023	0.0098	0.023
			In non-conductive status	0.0098	0.023	0.0098	0.023	0.0098	0.023
	LD<=	In conductive status		0.0098	0.023	0.0098	0.023	0.0098	0.023
		In non-conductive status		0.0098	0.023	0.0098	0.023	0.0098	0.023
	AND<=	When not executed		0.0078	0.023	0.0078	0.023	0.0078	0.023
		When executed	In conductive status	0.0098	0.023	0.0098	0.023	0.0098	0.023
			In non-conductive status	0.0098	0.023	0.0098	0.023	0.0098	0.023
	OR<=	When not executed		0.0078	0.023	0.0078	0.023	0.0078	0.023
		When executed	In conductive status	0.0098	0.023	0.0098	0.023	0.0098	0.023
			In non-conductive status	0.0098	0.023	0.0098	0.023	0.0098	0.023
	LD<	In conductive status		0.0098	0.023	0.0098	0.023	0.0098	0.023
		In non-conductive status		0.0098	0.023	0.0098	0.023	0.0098	0.023
	AND<	When not executed		0.0078	0.023	0.0078	0.023	0.0078	0.023
		When executed	In conductive status	0.0098	0.023	0.0098	0.023	0.0098	0.023
			In non-conductive status	0.0098	0.023	0.0098	0.023	0.0098	0.023
	OR<	When not executed		0.0078	0.023	0.0078	0.023	0.0078	0.023
		When executed	In conductive status	0.0098	0.023	0.0098	0.023	0.0098	0.023
			In non-conductive status	0.0098	0.023	0.0098	0.023	0.0098	0.023
	LD>=	In conductive status		0.0098	0.023	0.0098	0.023	0.0098	0.023
		In non-conductive status		0.0098	0.023	0.0098	0.023	0.0098	0.023
	AND>=	When not executed		0.0078	0.023	0.0078	0.023	0.0078	0.023
		When executed	In conductive status	0.0098	0.023	0.0098	0.023	0.0098	0.023
			In non-conductive status	0.0098	0.023	0.0098	0.023	0.0098	0.023

Category	Instruction	Condition (device)		Processing time (μs)					
				Q03UDVCPU		Q04UDVCPU, Q04UDPVCPU		Q06UDVCPU, Q06UDPVCPU, Q13UDVCPU, Q13UDPVCPU, Q26UDVCPU, Q26UDPVCPU	
				Min.	Max.	Min.	Max.	Min.	Max.
Basic instruction	OR>=	When not executed		0.0078	0.023	0.0078	0.023	0.0078	0.023
		When executed	In conductive status	0.0098	0.023	0.0098	0.023	0.0098	0.023
			In non-conductive status	0.0098	0.023	0.0098	0.023	0.0098	0.023
	LDD=	In conductive status		0.0098	0.023	0.0098	0.023	0.0098	0.023
		In non-conductive status		0.0098	0.023	0.0098	0.023	0.0098	0.023
	ANDD=	When not executed		0.0078	0.023	0.0078	0.023	0.0078	0.023
		When executed	In conductive status	0.0098	0.023	0.0098	0.023	0.0098	0.023
			In non-conductive status	0.0098	0.023	0.0098	0.023	0.0098	0.023
	ORD=	When not executed		0.0078	0.023	0.0078	0.023	0.0078	0.023
		When executed	In conductive status	0.0098	0.023	0.0098	0.023	0.0098	0.023
			In non-conductive status	0.0098	0.023	0.0098	0.023	0.0098	0.023
	LDD<>	In conductive status		0.0098	0.023	0.0098	0.023	0.0098	0.023
		In non-conductive status		0.0098	0.023	0.0098	0.023	0.0098	0.023
	ANDD<>	When not executed		0.0078	0.023	0.0078	0.023	0.0078	0.023
		When executed	In conductive status	0.0098	0.023	0.0098	0.023	0.0098	0.023
			In non-conductive status	0.0098	0.023	0.0098	0.023	0.0098	0.023
	ORD<>	When not executed		0.0078	0.023	0.0078	0.023	0.0078	0.023
		When executed	In conductive status	0.0098	0.023	0.0098	0.023	0.0098	0.023
			In non-conductive status	0.0098	0.023	0.0098	0.023	0.0098	0.023
	LDD>	In conductive status		0.0098	0.023	0.0098	0.023	0.0098	0.023
		In non-conductive status		0.0098	0.023	0.0098	0.023	0.0098	0.023
	ANDD>	When not executed		0.0078	0.023	0.0078	0.023	0.0078	0.023
		When executed	In conductive status	0.0098	0.023	0.0098	0.023	0.0098	0.023
			In non-conductive status	0.0098	0.023	0.0098	0.023	0.0098	0.023
	ORD>	When not executed		0.0078	0.023	0.0078	0.023	0.0078	0.023
		When executed	In conductive status	0.0098	0.023	0.0098	0.023	0.0098	0.023
			In non-conductive status	0.0098	0.023	0.0098	0.023	0.0098	0.023
	LDD<=	In conductive status		0.0098	0.023	0.0098	0.023	0.0098	0.023
		In non-conductive status		0.0098	0.023	0.0098	0.023	0.0098	0.023

Category	Instruction	Condition (device)		Processing time (μs)					
				Q03UDVCPU		Q04UDVCPU, Q04UDPVCPU		Q06UDVCPU, Q06UDPVCPU, Q13UDVCPU, Q13UDPVCPU, Q26UDVCPU, Q26UDPVCPU	
				Min.	Max.	Min.	Max.	Min.	Max.
Basic instruction	ANDD<=	When not executed		0.0078	0.023	0.0078	0.023	0.0078	0.023
		When executed	In conductive status	0.0098	0.023	0.0098	0.023	0.0098	0.023
			In non-conductive status	0.0098	0.023	0.0098	0.023	0.0098	0.023
	ORD<=	When not executed		0.0078	0.023	0.0078	0.023	0.0078	0.023
		When executed	In conductive status	0.0098	0.023	0.0098	0.023	0.0098	0.023
			In non-conductive status	0.0098	0.023	0.0098	0.023	0.0098	0.023
	LDD<	In conductive status		0.0098	0.023	0.0098	0.023	0.0098	0.023
		In non-conductive status		0.0098	0.023	0.0098	0.023	0.0098	0.023
	ANDD<	When not executed		0.0078	0.023	0.0078	0.023	0.0078	0.023
		When executed	In conductive status	0.0098	0.023	0.0098	0.023	0.0098	0.023
			In non-conductive status	0.0098	0.023	0.0098	0.023	0.0098	0.023
	ORD<	When not executed		0.0078	0.023	0.0078	0.023	0.0078	0.023
		When executed	In conductive status	0.0098	0.023	0.0098	0.023	0.0098	0.023
			In non-conductive status	0.0098	0.023	0.0098	0.023	0.0098	0.023
	LDD>=	In conductive status		0.0098	0.023	0.0098	0.023	0.0098	0.023
		In non-conductive status		0.0098	0.023	0.0098	0.023	0.0098	0.023
	ANDD>=	When not executed		0.0078	0.023	0.0078	0.023	0.0078	0.023
		When executed	In conductive status	0.0098	0.023	0.0098	0.023	0.0098	0.023
			In non-conductive status	0.0098	0.023	0.0098	0.023	0.0098	0.023
	ORD>=	When not executed		0.0078	0.023	0.0078	0.023	0.0078	0.023
		When executed	In conductive status	0.0098	0.023	0.0098	0.023	0.0098	0.023
			In non-conductive status	0.0098	0.023	0.0098	0.023	0.0098	0.023
	+ (S) (D)	When executed		0.0098	0.023	0.0098	0.023	0.0098	0.023
	+ (S1) (S2) (D)	When executed		0.011	0.031	0.011	0.031	0.011	0.031
	- (S) (D)	When executed		0.0098	0.023	0.0098	0.023	0.0098	0.023
	- (S1) (S2) (D)	When executed		0.011	0.031	0.011	0.031	0.011	0.031
	D+ (S) (D)	When executed		0.0098	0.023	0.0098	0.023	0.0098	0.023
	D+ (S1) (S2) (D)	When executed		0.011	0.031	0.011	0.031	0.011	0.031
	D- (S) (D)	When executed		0.0098	0.023	0.0098	0.023	0.0098	0.023
	D- (S1) (S2) (D)	When executed		0.011	0.031	0.011	0.031	0.011	0.031
	* (S1) (S2) (D)	When executed		0.015	0.023	0.015	0.023	0.015	0.023
	/ (S1) (S2) (D)	When executed		0.023	0.023	0.023	0.023	0.023	0.023

Category	Instruction	Condition (device)		Processing time (μs)					
				Q03UDVCPU		Q04UDVCPU, Q04UDPVCPU		Q06UDVCPU, Q06UDPVCPU, Q13UDVCPU, Q13UDPVCPU, Q26UDVCPU, Q26UDPVCPU	
				Min.	Max.	Min.	Max.	Min.	Max.
Basic instruction	D* (S1) (S2) (D)	When executed		0.023	0.023	0.023	0.023	0.023	0.023
	D/ (S1) (S2) (D)	When executed		0.033	0.054	0.033	0.054	0.033	0.054
	B+ (S) (D)	When executed		1.400	9.100	1.400	9.100	1.400	9.100
	B+ (S1) (S2) (D)	When executed		2.100	7.400	2.100	7.400	2.100	7.400
	B- (S) (D)	When executed		1.400	9.000	1.400	9.000	1.400	9.000
	B- (S1) (S2) (D)	When executed		2.100	7.400	2.100	7.400	2.100	7.400
	B* (S1) (S2) (D)	When executed		1.600	10.900	1.600	10.900	1.600	10.900
	B/ (S1) (S2) (D)	When executed		1.700	10.200	1.700	10.200	1.700	10.200
	E+ (S) (D)	Single precision	(S)=0, (D)=0	0.013	0.023	0.013	0.023	0.013	0.023
			(S)=2^{127}, (D)=2^{127}	0.013	0.023	0.013	0.023	0.013	0.023
	E+ (S1) (S2) (D)	Single precision	(S1)=0, (S2)=0	0.015	0.031	0.015	0.031	0.015	0.031
			(S1)=2^{127}, (S2)=2^{127}	0.015	0.031	0.015	0.031	0.015	0.031
	E- (S) (D)	Single precision	(S)=0, (D)=0	0.013	0.023	0.013	0.023	0.013	0.023
			(S)=2^{127}, (D)=2^{127}	0.013	0.023	0.013	0.023	0.013	0.023
	E- (S1) (S2) (D)	Single precision	(S1)=0, (S2)=0	0.015	0.031	0.015	0.031	0.015	0.031
			(S1)=2^{127}, (S2)=2^{127}	0.015	0.031	0.015	0.031	0.015	0.031
	E* (S1) (S2) (D)	Single precision	(S1)=0, (S2)=0	0.013	0.023	0.013	0.023	0.013	0.023
			(S1)=2^{127}, (S2)=2^{127}	0.013	0.023	0.013	0.023	0.013	0.023
	E/ (S1) (S2) (D)	Single precision	(S1)=2^{127}, (S2)=2^{127}	0.060	0.060	0.060	0.060	0.060	0.060
	INC	When executed		0.0078	0.015	0.0078	0.015	0.0078	0.015
	DINC	When executed		0.0078	0.015	0.0078	0.015	0.0078	0.015
	DEC	When executed		0.0078	0.015	0.0078	0.015	0.0078	0.015
	DDEC	When executed		0.0078	0.015	0.0078	0.015	0.0078	0.015
	BCD	When executed		0.013	0.015	0.013	0.015	0.013	0.015
	DBCD	When executed		0.021	0.019	0.021	0.019	0.021	0.019
	BIN	When executed		0.0098	0.015	0.0098	0.015	0.0098	0.015
	DBIN	When executed		0.0098	0.015	0.0098	0.015	0.0098	0.015
	FLT	Single precision	(S)=0	0.0098	0.015	0.0098	0.015	0.0098	0.015
			(S)=7FFFH	0.0098	0.015	0.0098	0.015	0.0098	0.015
	DFLT	Single precision	(S)=0	0.0098	0.015	0.0098	0.015	0.0098	0.015
			(S)=7FFFFFFFH	0.0098	0.015	0.0098	0.015	0.0098	0.015
	INT	Single precision	(S)=0	0.0098	0.015	0.0098	0.015	0.0098	0.015
			(S)=32766.5	0.0098	0.015	0.0098	0.015	0.0098	0.015
	DINT	Single precision	(S)=0	0.0098	0.015	0.0098	0.015	0.0098	0.015
			(S)=1234567890.3	0.0098	0.015	0.0098	0.015	0.0098	0.015
	MOV	—		0.0039	0.015	0.0039	0.015	0.0039	0.015
	DMOV	—		0.0039	0.015	0.0039	0.015	0.0039	0.015
	EMOV	—		0.0039	0.015	0.0039	0.015	0.0039	0.015

Category	Instruction	Condition (device)		Processing time (μs)					
				Q03UDVCPU		Q04UDVCPU, Q04UDPVCPU		Q06UDVCPU, Q06UDPVCPU, Q13UDVCPU, Q13UDPVCPU, Q26UDVCPU, Q26UDPVCPU	
				Min.	Max.	Min.	Max.	Min.	Max.
Basic instruction	CML	—		0.0058	0.015	0.0058	0.015	0.0058	0.015
	DCML	—		0.0058	0.015	0.0058	0.015	0.0058	0.015
	BMOV	n=1	SM237=OFF	2.400	7.500	2.400	7.500	2.400	7.500
			SM237=ON	1.200	1.400	1.200	1.400	1.200	1.400
		n=96	SM237=OFF	2.900	8.000	2.900	8.000	2.900	8.000
			SM237=ON	2.000	2.200	2.000	2.200	2.000	2.200
	FMOV	n=1	SM237=OFF	2.000	7.000	2.000	7.000	2.000	7.000
			SM237=ON	1.000	1.200	1.000	1.200	1.000	1.200
		n=96	SM237=OFF	2.700	7.700	2.700	7.700	2.700	7.700
			SM237=ON	2.000	2.200	2.000	2.200	2.000	2.200
	XCH	—		0.700	1.900	0.700	1.900	0.700	1.900
	DXCH	—		0.700	2.200	0.700	2.200	0.700	2.200
	DFMOV	n=1	SM237=OFF	2.000	7.500	2.000	7.500	2.000	7.500
			SM237=ON	1.200	2.000	1.200	2.000	1.200	2.000
		n=96	SM237=OFF	3.400	7.200	3.400	7.200	3.400	7.200
			SM237=ON	2.800	3.400	2.800	3.400	2.800	3.400
	CJ	—		1.200	8.300	1.200	8.300	1.200	8.300
	SCJ	—		1.200	8.300	1.200	8.300	1.200	8.300
	JMP	—		1.200	8.300	1.200	8.300	1.200	8.300
Application instruction	WAND (S) (D)	When executed		0.0078	0.023	0.0078	0.023	0.0078	0.023
	WAND (S1) (S2) (D)	When executed		0.0098	0.031	0.0098	0.031	0.0098	0.031
	DAND (S) (D)	When executed		0.0078	0.023	0.0078	0.023	0.0078	0.023
	DAND (S1) (S2) (D)	When executed		0.0098	0.031	0.0098	0.031	0.0098	0.031
	WOR (S) (D)	When executed		0.0078	0.023	0.0078	0.023	0.0078	0.023
	WOR (S1) (S2) (D)	When executed		0.0098	0.031	0.0098	0.031	0.0098	0.031
	DOR (S) (D)	When executed		0.0078	0.023	0.0078	0.023	0.0078	0.023
	DOR (S1) (S2) (D)	When executed		0.0098	0.031	0.0098	0.031	0.0098	0.031
	WXOR (S) (D)	When executed		0.0078	0.023	0.0078	0.023	0.0078	0.023
	WXOR (S1) (S2) (D)	When executed		0.0098	0.031	0.0098	0.031	0.0098	0.031
	DXOR (S) (D)	When executed		0.0078	0.023	0.0078	0.023	0.0078	0.023
	DXOR (S1) (S2) (D)	When executed		0.0098	0.031	0.0098	0.031	0.0098	0.031

Category	Instruction	Condition (device)	Processing time (μs)					
			Q03UDVCPU		Q04UDVCPU, Q04UDPVCPU		Q06UDVCPU, Q06UDPVCPU, Q13UDVCPU, Q13UDPVCPU, Q26UDVCPU, Q26UDPVCPU	
			Min.	Max.	Min.	Max.	Min.	Max.
Application instruction	WXNR (S) (D)	When executed	0.0078	0.023	0.0078	0.023	0.0078	0.023
	WXNR (S1) (S2) (D)	When executed	0.0098	0.031	0.0098	0.031	0.0098	0.031
	DXNR (S) (D)	When executed	0.0078	0.023	0.0078	0.023	0.0078	0.023
	DXNR (S1) (S2) (D)	When executed	0.0098	0.031	0.0098	0.031	0.0098	0.031
	ROR (D) n	n=1	0.011	0.023	0.011	0.023	0.011	0.023
		n=15	0.011	0.023	0.011	0.023	0.011	0.023
	RCR (D) n	n=1	0.015	0.031	0.015	0.031	0.015	0.031
		n=15	0.015	0.031	0.015	0.031	0.015	0.031
	ROL (D) n	n=1	0.011	0.023	0.011	0.023	0.011	0.023
		n=15	0.011	0.023	0.011	0.023	0.011	0.023
	RCL (D) n	n=1	0.015	0.031	0.015	0.031	0.015	0.031
		n=15	0.015	0.031	0.015	0.031	0.015	0.031
	DROR (D) n	n=1	0.011	0.023	0.011	0.023	0.011	0.023
		n=31	0.011	0.023	0.011	0.023	0.011	0.023
	DRCR (D) n	n=1	0.015	0.031	0.015	0.031	0.015	0.031
		n=31	0.015	0.031	0.015	0.031	0.015	0.031
	DROL (D) n	n=1	0.011	0.023	0.011	0.023	0.011	0.023
		n=31	0.011	0.023	0.011	0.023	0.011	0.023
	DRCL (D) n	n=1	0.015	0.031	0.015	0.031	0.015	0.031
		n=31	0.015	0.031	0.015	0.031	0.015	0.031
	SFR (D) n	n=1	0.011	0.023	0.011	0.023	0.011	0.023
		n=15	0.011	0.023	0.011	0.023	0.011	0.023
	SFL (D) n	n=1	0.011	0.023	0.011	0.023	0.011	0.023
		n=15	0.011	0.023	0.011	0.023	0.011	0.023
	DSFR (D) n	n=1	1.000	6.300	1.000	6.300	1.000	6.300
		n=96	8.100	14.100	8.100	14.100	8.100	14.100
	DSFL (D) n	n=1	1.000	6.300	1.000	6.300	1.000	6.300
		n=96	8.100	14.100	8.100	14.100	8.100	14.100
	SUM	(S)=0	1.300	4.900	1.300	4.900	1.300	4.900
		(S)=FFFFH	1.300	4.900	1.300	4.900	1.300	4.900
	SEG	When executed	1.000	4.800	1.000	4.800	1.000	4.800
	FOR	—	0.0058	0.015	0.0058	0.015	0.0058	0.015
	CALL Pn	Internal file pointer	1.200	1.500	1.200	1.500	1.200	1.500
		Common pointer	3.700	12.800	3.700	12.800	3.700	12.800
	CALL Pn (S1) to (S5)	—	9.000	29.500	9.000	29.500	9.000	29.500

Point

For the instructions for which a rising edge instruction (□P) is not described, the processing time is the same as an ON execution instruction.
(Example) MOVP instruction, WANDP instruction etc.

■Table of the time to be added when file register, extended data register, extended link register, and module access device are used

• When using Q00UJCPU, Q00UCPU, Q01UCPU and Q02UCPU

Device name		Data	Device specification location	Addition time (μs)			
				Q00UJCPU	Q00UCPU	Q01UCPU	Q02UCPU
File register (R)	When standard RAM is used	Bit	Source	0.100	0.100	0.100	0.100
			Destination	0.220	0.220	0.220	0.220
		Word	Source	0.100	0.100	0.100	0.100
			Destination	0.100	0.100	0.100	0.100
		Double word	Source	0.200	0.200	0.200	0.200
			Destination	0.200	0.200	0.200	0.200
	When SRAM card is used (Q2MEM-1MBS, Q2MEM-2MBS)	Bit	Source	—	—	—	0.220
			Destination	—	—	—	0.420
		Word	Source	—	—	—	0.220
			Destination	—	—	—	0.180
		Double word	Source	—	—	—	0.440
			Destination	—	—	—	0.380
	When SRAM card is used (Q3MEM-4MBS, Q3MEM-8MBS)	Bit	Source	—	—	—	0.160
			Destination	—	—	—	0.320
		Word	Source	—	—	—	0.160
			Destination	—	—	—	0.140
		Double word	Source	—	—	—	0.320
			Destination	—	—	—	0.300
File register (ZR), extended data register (D), extended link register (W)	When standard RAM is used	Bit	Source	0.220	0.180	0.160	0.140
			Destination	0.360	0.320	0.300	0.280
		Word	Source	0.220	0.180	0.160	0.140
			Destination	0.220	0.180	0.160	0.140
		Double word	Source	0.320	0.280	0.260	0.240
			Destination	0.320	0.280	0.260	0.240
	When SRAM card is used (Q2MEM-1MBS, Q2MEM-2MBS)	Bit	Source	—	—	—	0.260
			Destination	—	—	—	0.480
		Word	Source	—	—	—	0.260
			Destination	—	—	—	0.220
		Double word	Source	—	—	—	0.480
			Destination	—	—	—	0.420
	When SRAM card is used (Q3MEM-4MBS, Q3MEM-8MBS)	Bit	Source	—	—	—	0.200
			Destination	—	—	—	0.380
		Word	Source	—	—	—	0.200
			Destination	—	—	—	0.180
		Double word	Source	—	—	—	0.360
			Destination	—	—	—	0.340
Module access device (Multiple CPU high speed transmission area) (U3En\G10000)		Bit	Source	—	—	—	—
			Destination	—	—	—	—
		Word	Source	—	—	—	—
			Destination	—	—	—	—
		Double word	Source	—	—	—	—
			Destination	—	—	—	—

• When using Q03UD(E)CPU, Q04UD(E)HCPU, Q06UD(E)HCPU, Q10UD(E)HCPU, Q13UD(E)HCPU, Q20UD(E)HCPU, Q26UD(E)HCPU, Q50UDEHCPU, and Q100UDEHCPU

Device name		Data	Device specification location	Addition time (μs)			
				Q03UD(E)CPU	Q04UD(E)HCPU, Q06UD(E)HCPU	Q10UD(E)HCPU, Q13UD(E)HCPU, Q20UD(E)HCPU, Q26UD(E)HCPU	Q50UDEHCPU, Q100UDEHCPU
File register (R)	When standard RAM is used	Bit	Source	0.100	0.048	0.048	0.048
			Destination	0.100	0.038	0.038	0.038
		Word	Source	0.100	0.048	0.048	0.048
			Destination	0.100	0.038	0.038	0.038
		Double word	Source	0.200	0.095	0.095	0.095
			Destination	0.200	0.086	0.086	0.086
	When SRAM card is used (Q2MEM-1MBS, Q2MEM-2MBS)	Bit	Source	0.220	0.200	0.200	0.200
			Destination	0.180	0.162	0.162	0.162
		Word	Source	0.220	0.200	0.200	0.200
			Destination	0.180	0.162	0.162	0.162
		Double word	Source	0.440	0.399	0.399	0.399
			Destination	0.380	0.361	0.361	0.361
	When SRAM card is used (Q3MEM-4MBS, Q3MEM-8MBS)	Bit	Source	0.160	0.152	0.152	0.152
			Destination	0.140	0.133	0.133	0.133
		Word	Source	0.160	0.152	0.152	0.152
			Destination	0.140	0.133	0.133	0.133
		Double word	Source	0.320	0.304	0.304	0.304
			Destination	0.300	0.295	0.295	0.295
File register (ZR), extended data register (D), extended link register (W)	When standard RAM is used	Bit	Source	0.120	0.057	0.057	0.057
			Destination	0.120	0.048	0.048	0.048
		Word	Source	0.120	0.057	0.057	0.057
			Destination	0.120	0.048	0.048	0.048
		Double word	Source	0.220	0.105	0.105	0.105
			Destination	0.220	0.095	0.095	0.095
	When SRAM card is used (Q2MEM-1MBS, Q2MEM-2MBS)	Bit	Source	0.240	0.209	0.209	0.209
			Destination	0.200	0.171	0.171	0.171
		Word	Source	0.240	0.209	0.209	0.209
			Destination	0.200	0.171	0.171	0.171
		Double word	Source	0.460	0.409	0.409	0.409
			Destination	0.400	0.371	0.371	0.371
	When SRAM card is used (Q3MEM-4MBS, Q3MEM-8MBS)	Bit	Source	0.180	0.162	0.162	0.162
			Destination	0.160	0.143	0.143	0.143
		Word	Source	0.180	0.162	0.162	0.162
			Destination	0.160	0.143	0.143	0.143
		Double word	Source	0.340	0.314	0.314	0.314
			Destination	0.320	0.304	0.304	0.304
Module access device (Multiple CPU high speed transmission area) (U3En\G10000)		Bit	Source	0.220	0.181	0.181	0.181
			Destination	0.140	0.105	0.105	0.105
		Word	Source	0.220	0.181	0.181	0.181
			Destination	0.140	0.105	0.105	0.105
		Double word	Source	0.500	0.437	0.437	0.437
			Destination	0.340	0.285	0.285	0.285

• When using Q03UDVCPU, Q04UDVCPU, Q04UDPVCPU, Q06UDVCPU, Q06UDPVCPU, Q13UDVCPU, Q13UDPVCPU, Q26UDVCPU, and Q26UDPVCPU

Device name		Data	Device specification location	Addition time (μs)		
				Q03UDVCPU	Q04UDVCPU, Q04UDPVCPU	Q06UDVCPU, Q06UDPVCPU, Q13UDVCPU, Q13UDPVCPU, Q26UDVCPU, Q26UDPVCPU
File register (R)	When the extended SRAM cassette is not used	Bit	Source	0.074	0.043	0.043
			Destination	0.023	0.023	0.023
		Word	Source	0.074	0.043	0.043
			Destination	0.023	0.023	0.023
		Double word	Source	0.148	0.085	0.085
			Destination	0.044	0.044	0.044
	When the extended SRAM cassette is used	Bit	Source	0.099	0.099	0.099
			Destination	0.028	0.028	0.028
		Word	Source	0.099	0.099	0.099
			Destination	0.028	0.028	0.028
		Double word	Source	0.198	0.198	0.198
			Destination	0.054	0.054	0.054
File register (ZR), extended data register (D), extended link register (W)	When the extended SRAM cassette is not used	Bit	Source	0.074	0.043	0.043
			Destination	0.023	0.023	0.023
		Word	Source	0.074	0.043	0.043
			Destination	0.023	0.023	0.023
		Double word	Source	0.148	0.085	0.085
			Destination	0.044	0.044	0.044
	When the extended SRAM cassette is used	Bit	Source	0.099	0.099	0.099
			Destination	0.028	0.028	0.028
		Word	Source	0.099	0.099	0.099
			Destination	0.028	0.028	0.028
		Double word	Source	0.198	0.198	0.198
			Destination	0.054	0.054	0.054
Module access device (Multiple CPU high speed transmission area) (U3En\G10000)		Bit	Source	0.042	0.042	0.042
			Destination	0.049	0.049	0.049
		Word	Source	0.042	0.042	0.042
			Destination	0.049	0.049	0.049
		Double word	Source	0.092	0.092	0.092
			Destination	0.095	0.095	0.095

■Table of the time to be added when F/T(ST)/C device is used in OUT/SET/RST instruction

• When using Q00UJCPU, Q00UCPU, Q01UCPU and Q02UCPU

Instruction name	Device name	Condition		Addition time (μs)			
				Q00UJCPU	Q00UCPU	Q01UCPU	Q02UCPU
OUT	F	When not executed		2.900	2.900	2.900	2.100
		When executed		116.000	116.000	116.000	68.800
	T(ST), C	When not executed		0.360	0.240	0.180	0.120
		When executed	After time up	0.360	0.240	0.180	0.120
			When added	0.360	0.240	0.180	0.120
SET	F	When not executed		0.120	0.080	0.006	0.004
		When executed		116.000	116.000	116.000	68.600
RST	F	When not executed		0.120	0.080	0.006	0.004
		When executed		55.800	55.800	55.800	26.500
	T(ST), C	When not executed		0.360	0.240	0.180	0.120
		When executed		0.360	0.240	0.180	0.120

• When using Q03UD(E)HCPU, Q04UD(E)HCPU, Q06UD(E)HCPU, Q10UD(E)HCPU, Q13UD(E)HCPU, Q20UD(E)HCPU, Q26UD(E)HCPU, Q50UDEHCPU, and Q100UDEHCPU

Instruction name	Device name	Condition		Addition time (μs)			
				Q03UD(E)CPU	Q04UD(E)HCPU, Q06UD(E)HCPU	Q10UD(E)HCPU, Q13UD(E)HCPU, Q20UD(E)HCPU, Q26UD(E)HCPU	Q50UDEHCPU, Q100UDEHCPU
OUT	F	When not executed		1.940	1.570	1.570	1.570
		When executed		39.930	38.090	38.090	38.090
	T(ST), C	When not executed		0.060	0.030	0.030	0.030
		When executed	After time up	0.060	0.030	0.030	0.030
SET	F	When not executed		0.000	0.000	0.000	0.000
		When executed		42.900	40.600	40.600	40.600
RST	F	When not executed		0.000	0.000	0.000	0.000
		When executed		45.260	36.600	36.600	36.600
	T(ST), C	When not executed		0.060	0.030	0.030	0.030
		When executed		0.060	0.030	0.030	0.030

A

• When using Q03UDVCPU, Q04UDVCPU, Q04UDPVCPU, Q06UDVCPU, Q06UDPVCPU, Q13UDVCPU, Q13UDPVCPU, Q26UDVCPU, and Q26UDPVCPU

Instruction name	Device name	Condition		Addition time (μs)					
				Q03UDVCPU		Q04UDVCPU, Q04UDPVCPU		Q06UDVCPU, Q06UDPVCPU, Q13UDVCPU, Q13UDPVCPU, Q26UDVCPU, Q26UDPVCPU	
				Min.	Max.	Min.	Max.	Min.	Max.
OUT	F	When not executed		1.100	3.900	1.100	3.900	1.100	3.900
		When executed		15.000	49.000	15.000	49.000	15.000	49.000
	T(ST), C	When not executed		0.011		0.011		0.011	
		When executed	After time up	0.011		0.011		0.011	
			When added	0.011		0.011		0.011	
SET	F	When not executed		0.005		0.005		0.005	
		When executed		14.000	49.000	14.000	49.000	14.000	49.000
RST	F	When not executed		0.005		0.005		0.005	
		When executed		7.900	26.000	7.900	26.000	7.900	26.000
	T(ST), C	When not executed		0.011		0.011		0.011	
		When executed		0.011		0.011		0.011	

Processing time of instructions other than subset instruction

The following table shows the processing time of instructions other than subset instructions.

Point

- The processing time shown in "Table of the processing time of instructions other than subset instructions" applies when the device used in an instruction does not meet the device condition for subset processing. (For device condition that does not trigger subset processing, refer to Page 109 Subset processing.) For instructions not shown in the following table, refer to Page 928 Subset instruction processing time.
- When using the file resister (R, ZR), extended data register (D), extended link register (W), module access device (Un\G□, U3En\G0 to G4095) or link direct device (Jn\□), add the processing time shown in Page 1031 Table of the time to be added when file register, extended data register, extended link register, module access device, and link direct device are used to that of each instruction.
- Since the processing time of an instruction varies depending on that of the cash function, both the minimum and maximum values are described in the table.

■Table of the processing time of instructions other than subset instructions

- When using Q00UJCPU, Q00UCPU, Q01UCPU and Q02UCPU

Category	Instruction	Condition (device)	Processing time (μs)							
			Q00UJCPU		Q00UCPU		Q01UCPU		Q02UCPU	
			Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.
Sequence instruction	ANB ORB MPS MRD MPP	—	0.120		0.080		0.060		0.040	
	INV	When not executed	0.120		0.080		0.060		0.040	
		When executed								
	MEP MEF	When not executed	0.120		0.080		0.060		0.040	
		When executed								
	EGP EGF	When not executed	0.120		0.080		0.060		0.040	
		When executed								
	PLS	—	1.800	1.900	1.800	1.900	1.800	1.900	1.300	1.600
	PLF	—	1.800	1.900	1.800	1.900	1.800	1.900	1.600	1.700
	FF	When not executed	0.240		0.160		0.120		0.080	
		When executed	1.700	1.800	1.700	1.800	1.700	1.800	1.200	1.500
	DELTA	When not executed	0.240		0.160		0.120		0.080	
		When executed	4.000	14.700	4.000	14.700	4.000	14.700	2.800	3.600
	SFT	When not executed	0.240		0.160		0.120		0.800	
		When executed	1.800	12.600	1.800	12.600	1.800	12.600	1.600	6.600
	MC	—	0.240		0.160		0.120		0.080	
	MCR	—	0.120		0.080		0.060		0.040	
	FEND END	Error check performed	250.000	250.000	250.000	250.000	250.000	250.000	175.000	252.000
		No error check performed	250.000	250.000	250.000	250.000	250.000	250.000	175.000	221.000
	STOP	—	—		—		—		—	
	NOP NOPLF PAGE	—	0.120		0.080		0.060		0.040	

A

Category	Instruction	Condition (device)			Processing time (μs)							
					Q00UJCPU		Q00UCPU		Q01UCPU		Q02UCPU	
					Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.
Basic instruction	LDE=	Single precision	In conductive status		4.400	20.900	4.400	20.900	4.400	20.900	4.700	10.100
			In non-conductive status		4.400	20.900	4.400	20.900	4.400	20.900	4.700	10.100
	ANDE=	Single precision	When not executed		0.360		0.240		0.180		0.120	
			When executed	In conductive status	4.200	19.600	4.200	19.600	4.200	19.600	4.200	12.500
				In non-conductive status	4.200	19.600	4.200	19.600	4.200	19.600	4.400	11.900
	ORE=	Single precision	When not executed		0.360		0.240		0.180		0.120	
			When executed	In conductive status	4.200	17.400	4.200	17.400	4.200	17.400	4.600	10.800
				In non-conductive status	4.200	17.400	4.200	17.400	4.200	17.400	4.500	9.800
	LDE<>	Single precision	In conductive status		4.400	20.900	4.400	20.900	4.400	20.900	4.700	7.700
			In non-conductive status		4.400	20.900	4.400	20.900	4.400	20.900	4.600	8.200
	ANDE<>	Single precision	When not executed		0.360		0.240		0.180		0.120	
			When executed	In conductive status	4.200	19.600	4.200	19.600	4.200	19.600	4.300	14.200
				In non-conductive status	4.200	19.600	4.200	19.600	4.200	19.600	4.400	14.200
	ORE<>	Single precision	When not executed		0.360		0.240		0.180		0.120	
			When executed	In conductive status	4.200	17.400	4.200	17.400	4.200	17.400	4.600	6.700
				In non-conductive status	4.200	17.400	4.200	17.400	4.200	17.400	4.400	6.600
	LDE>	Single precision	In conductive status		4.400	20.900	4.400	20.900	4.400	20.900	4.700	13.700
			In non-conductive status		4.400	20.900	4.400	20.900	4.400	20.900	4.600	13.700
	ANDE>	Single precision	When not executed		0.360		0.240		0.180		0.120	
			When executed	In conductive status	4.200	19.600	4.200	19.600	4.200	19.600	4.300	8.100
				In non-conductive status	4.200	19.600	4.200	19.600	4.200	19.600	4.200	8.100
	ORE>	Single precision	When not executed		0.360		0.240		0.180		0.120	
			When executed	In conductive status	4.200	17.400	4.200	17.400	4.200	17.400	4.600	8.500
				In non-conductive status	4.200	17.400	4.200	17.400	4.200	17.400	4.400	8.100
	LDE<=	Single precision	In conductive status		4.400	20.900	4.400	20.900	4.400	20.900	4.700	11.100
			In non-conductive status		4.400	20.900	4.400	20.900	4.400	20.900	4.700	9.600
	ANDE<=	Single precision	When not executed		0.360		0.240		0.180		0.120	
			When executed	In conductive status	4.200	19.600	4.200	19.600	4.200	19.600	4.100	7.800
				In non-conductive status	4.200	19.600	4.200	19.600	4.200	19.600	4.400	8.200
	ORE<=	Single precision	When not executed		0.360		0.240		0.180		0.120	
			When executed	In conductive status	4.200	17.400	4.200	17.400	4.200	17.400	4.500	10.300
				In non-conductive status	4.200	17.400	4.200	17.400	4.200	17.400	4.400	9.800

Category	Instruction	Condition (device)			Processing time (μs)							
					Q00UJCPU		Q00UCPU		Q01UCPU		Q02UCPU	
					Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.
Basic instruction	LDE<	Single precision	In conductive status		4.400	20.900	4.400	20.900	4.400	20.900	4.700	11.500
			In non-conductive status		4.400	20.900	4.400	20.900	4.400	20.900	4.700	10.900
	ANDE<	Single precision	When not executed		0.360		0.240		0.180		0.120	
			When executed	In conductive status	4.200	19.600	4.200	19.600	4.200	19.600	4.300	9.200
				In non-conductive status	4.200	19.600	4.200	19.600	4.200	19.600	4.400	9.400
	ORE<	Single precision	When not executed		0.360		0.240		0.180		0.120	
			When executed	In conductive status	4.200	17.400	4.200	17.400	4.200	17.400	4.600	10.400
				In non-conductive status	4.200	17.400	4.200	17.400	4.200	17.400	4.400	9.800
	LDE>=	Single precision	In conductive status		4.400	20.900	4.400	20.900	4.400	20.900	4.700	12.200
			In non-conductive status		4.400	20.900	4.400	20.900	4.400	20.900	4.700	11.800
	ANDE>=	Single precision	When not executed		0.360		0.240		0.180		0.120	
			When executed	In conductive status	4.200	19.600	4.200	19.600	4.200	19.600	4.100	6.700
				In non-conductive status	4.200	19.600	4.200	19.600	4.200	19.600	4.400	7.000
	ORE>=	Single precision	When not executed		0.360		0.240		0.180		0.120	
			When executed	In conductive status	4.200	17.400	4.200	17.400	4.200	17.400	4.600	14.000
				In non-conductive status	4.200	17.400	4.200	17.400	4.200	17.400	4.500	14.300
	LDED=	Double precision	In conductive status		4.700	37.400	4.700	37.400	4.700	37.400	4.200	21.000
			In non-conductive status		4.700	37.400	4.700	37.400	4.700	37.400	5.100	21.900
	ANDED=	Double precision	When not executed		0.360		0.240		0.180		0.120	
			When executed	In conductive status	4.500	34.700	4.500	34.700	4.500	34.700	3.800	17.800
				In non-conductive status	4.500	34.700	4.500	34.700	4.500	34.700	4.100	18.100
	ORED=	Double precision	When not executed		0.360		0.240		0.180		0.120	
			When executed	In conductive status	4.700	33.200	4.700	33.200	4.700	33.200	4.100	23.800
				In non-conductive status	4.700	33.200	4.700	33.200	4.700	33.200	4.900	25.500
	LDED<>	Double precision	In conductive status		4.700	37.400	4.700	37.400	4.700	37.400	5.100	23.500
			In non-conductive status		4.700	37.400	4.700	37.400	4.700	37.400	4.200	22.600
	ANDED<>	Double precision	When not executed		0.360		0.240		0.180		0.120	
			When executed	In conductive status	4.500	34.700	4.500	34.700	4.500	34.700	4.000	18.800
				In non-conductive status	4.500	34.700	4.500	34.700	4.500	34.700	4.000	18.700
	ORED<>	Double precision	When not executed		0.360		0.240		0.180		0.120	
			When executed	In conductive status	4.700	33.200	4.700	33.200	4.700	33.200	5.000	25.200
				In non-conductive status	4.700	33.200	4.700	33.200	4.700	33.200	4.100	23.400

Category	Instruction	Condition (device)			Processing time (μs)							
					Q00UJCPU		Q00UCPU		Q01UCPU		Q02UCPU	
					Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.
Basic instruction	LDED>	Double precision	In conductive status		4.700	37.400	4.700	37.400	4.700	37.400	5.100	25.100
			In non-conductive status		4.700	37.400	4.700	37.400	4.700	37.400	4.200	23.400
	ANDED>	Double precision	When not executed		0.360		0.240		0.180		0.120	
			When executed	In conductive status	4.500	34.700	4.500	34.700	4.500	34.700	4.000	19.500
				In non-conductive status	4.500	34.700	4.500	34.700	4.500	34.700	4.100	19.700
	ORED>	Double precision	When not executed		0.360		0.240		0.180		0.120	
			When executed	In conductive status	4.700	33.200	4.700	33.200	4.700	33.200	5.000	24.200
				In non-conductive status	4.700	33.200	4.700	33.200	4.700	33.200	4.900	25.800
	LDED<=	Double precision	In conductive status		4.700	37.400	4.700	37.400	4.700	37.400	4.200	22.500
			In non-conductive status		4.700	37.400	4.700	37.400	4.700	37.400	4.200	13.500
	ANDED<=	Double precision	When not executed		0.360		0.240		0.180		0.120	
			When executed	In conductive status	4.500	34.700	4.500	34.700	4.500	34.700	4.000	19.600
				In non-conductive status	4.500	34.700	4.500	34.700	4.500	34.700	4.100	19.700
	ORED<=	Double precision	When not executed		0.360		0.240		0.180		0.120	
			When executed	In conductive status	4.700	33.200	4.700	33.200	4.700	33.200	5.000	26.300
				In non-conductive status	4.700	33.200	4.700	33.200	4.700	33.200	5.000	25.200
	LDED<	Double precision	In conductive status		4.700	37.400	4.700	37.400	4.700	37.400	5.100	25.000
			In non-conductive status		4.700	37.400	4.700	37.400	4.700	37.400	4.200	24.100
	ANDED<	Double precision	When not executed		0.360		0.240		0.180		0.120	
			When executed	In conductive status	4.500	34.700	4.500	34.700	4.500	34.700	4.000	19.400
				In non-conductive status	4.500	34.700	4.500	34.700	4.500	34.700	4.100	19.700
	ORED<	Double precision	When not executed		0.360		0.240		0.180		0.120	
			When executed	In conductive status	4.700	33.200	4.700	33.200	4.700	33.200	5.000	25.100
				In non-conductive status	4.700	33.200	4.700	33.200	4.700	33.200	5.000	25.100
	LDED>=	Double precision	In conductive status		4.700	37.400	4.700	37.400	4.700	37.400	4.200	13.100
			In non-conductive status		4.700	37.400	4.700	37.400	4.700	37.400	4.300	13.100
	ANDED>=	Double precision	When not executed		0.360		0.240		0.180		0.120	
			When executed	In conductive status	4.500	34.700	4.500	34.700	4.500	34.700	3.900	19.500
				In non-conductive status	4.500	34.700	4.500	34.700	4.500	34.700	4.100	19.800
	ORED>=	Double precision	When not executed		0.360		0.240		0.180		0.120	
			When executed	In conductive status	4.700	33.200	4.700	33.200	4.700	33.200	5.000	25.100
				In non-conductive status	4.700	33.200	4.700	33.200	4.700	33.200	4.200	18.500

Category	Instruction	Condition (device)		Processing time (μs)							
				Q00UJCPU		Q00UCPU		Q01UCPU		Q02UCPU	
				Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.
Basic instruction	LD$=	In conductive status		8.300	38.500	8.300	38.500	8.300	38.500	5.500	14.900
		In non-conductive status		8.300	38.500	8.300	38.500	8.300	38.500	5.500	15.600
	AND$=	When not executed		0.360		0.240		0.180		0.120	
		When executed	In conductive status	7.200	37.300	7.200	37.300	7.200	37.300	5.200	13.800
			In non-conductive status	7.200	37.300	7.200	37.300	7.200	37.300	5.300	14.500
	OR$=	When not executed		0.360		0.240		0.180		0.120	
		When executed	In conductive status	7.500	36.600	7.500	36.600	7.500	36.600	5.500	14.900
			In non-conductive status	7.500	36.600	7.500	36.600	7.500	36.600	5.300	14.600
	LD$<>	In conductive status		8.300	39.300	8.300	39.300	8.300	39.300	5.600	15.200
		In non-conductive status		8.300	39.300	8.300	39.300	8.300	39.300	5.600	15.400
	AND$<>	When not executed		0.360		0.240		0.180		0.120	
		When executed	In conductive status	8.000	38.200	8.000	38.200	8.000	38.200	4.300	21.500
			In non-conductive status	8.000	38.200	8.000	38.200	8.000	38.200	4.500	23.400
	OR$<>	When not executed		0.360		0.240		0.180		0.120	
		When executed	In conductive status	8.300	37.300	8.300	37.300	8.300	37.300	5.400	17.700
			In non-conductive status	8.300	37.300	8.300	37.300	8.300	37.300	5.300	19.400
	LD$>	In conductive status		8.300	41.600	8.300	41.600	8.300	41.600	6.400	19.200
		In non-conductive status		8.300	41.600	8.300	41.600	8.300	41.600	5.600	20.100
	AND$>	When not executed		0.360		0.240		0.180		0.120	
		When executed	In conductive status	8.000	38.100	8.000	38.100	8.000	38.100	4.500	15.400
			In non-conductive status	8.000	38.100	8.000	38.100	8.000	38.100	4.600	15.300
	OR$>	When not executed		0.360		0.240		0.180		0.120	
		When executed	In conductive status	8.200	35.700	8.200	35.700	8.200	35.700	5.400	20.000
			In non-conductive status	8.200	35.700	8.200	35.700	8.200	35.700	5.400	22.100
	LD$<=	In conductive status		8.300	39.200	8.300	39.200	8.300	39.200	5.800	12.800
		In non-conductive status		8.300	39.200	8.300	39.200	8.300	39.200	6.300	13.900

A

Category	Instruction	Condition (device)		Processing time (μs)							
				Q00UJCPU		Q00UCPU		Q01UCPU		Q02UCPU	
				Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.
Basic instruction	AND$<=	When not executed		0.360		0.240		0.180		0.120	
		When executed	In conductive status	7.100	36.500	7.100	36.500	7.100	36.500	6.000	16.000
			In non-conductive status	7.100	36.500	7.100	36.500	7.100	36.500	6.100	16.200
	OR$<=	When not executed		0.360		0.240		0.180		0.120	
		When executed	In conductive status	7.400	35.600	7.400	35.600	7.400	35.600	4.700	14.600
			In non-conductive status	7.400	35.600	7.400	35.600	7.400	35.600	4.600	14.400
	LD$<	In conductive status		7.400	40.000	7.400	40.000	7.400	40.000	4.800	17.000
		In non-conductive status		7.400	40.000	7.400	40.000	7.400	40.000	5.500	18.000
	AND$<	When not executed		0.360		0.240		0.180		0.120	
		When executed	In conductive status	8.000	37.300	8.000	37.300	8.000	37.300	5.900	13.400
			In non-conductive status	8.000	37.300	8.000	37.300	8.000	37.300	6.200	14.500
	OR$<	When not executed		0.360		0.240		0.180		0.120	
		When executed	In conductive status	8.300	35.600	8.300	35.600	8.300	35.600	6.200	18.700
			In non-conductive status	8.300	35.600	8.300	35.600	8.300	35.600	5.400	19.700
	LD$>=	In conductive status		7.400	38.300	7.400	38.300	7.400	38.300	4.800	10.000
		In non-conductive status		7.400	38.300	7.400	38.300	7.400	38.300	5.500	11.200
	AND$>=	When not executed		0.360		0.240		0.180		0.120	
		When executed	In conductive status	7.200	37.300	7.200	37.300	7.200	37.300	4.400	21.600
			In non-conductive status	7.200	37.300	7.200	37.300	7.200	37.300	4.500	21.800
	OR$>=	When not executed		0.360		0.240		0.180		0.120	
		When executed	In conductive status	8.200	36.400	8.200	36.400	8.200	36.400	5.400	15.400
			In non-conductive status	8.200	36.400	8.200	36.400	8.200	36.400	5.300	15.300
	BKCMP= (S1) (S2) (D) n	n=1		15.300	36.100	15.300	36.100	15.300	36.100	8.200	22.600
		n=96		64.500	85.500	64.500	85.500	64.500	85.500	57.400	72.500
	BKCMP<> (S1) (S2) (D) n	n=1		15.300	36.100	15.300	36.100	15.300	36.100	8.200	22.500
		n=96		66.600	87.500	66.600	87.500	66.600	87.500	59.500	74.500
	BKCMP> (S1) (S2) (D) n	n=1		15.300	36.100	15.300	36.100	15.300	36.100	8.200	23.100
		n=96		66.600	87.500	66.600	87.500	66.600	87.500	59.500	74.400
	BKCMP<= (S1) (S2) (D) n	n=1		15.300	36.100	15.300	36.100	15.300	36.100	8.200	22.500
		n=96		64.500	85.500	64.500	85.500	64.500	85.500	57.400	72.400

Category	Instruction	Condition (device)		Processing time (μs)							
				Q00UJCPU		Q00UCPU		Q01UCPU		Q02UCPU	
				Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.
Basic instruction	BKCMP< (S1) (S2) (D) n	n=1		15.300	36.100	15.300	36.100	15.300	36.100	8.300	23.000
		n=96		66.600	87.500	66.600	87.500	66.600	87.500	59.500	74.500
	BKCMP>= (S1) (S2) (D) n	n=1		15.300	36.100	15.300	36.100	15.300	36.100	8.200	22.500
		n=96		64.500	85.500	64.500	85.500	64.500	85.500	57.400	72.400
	DBKCMP= (S1) (S2) (D) n	n=1		15.800	36.300	15.800	36.300	15.800	36.300	9.350	29.000
		n=96		64.900	85.700	64.900	85.700	64.900	85.700	60.700	78.400
	DBKCMP<> (S1) (S2) (D) n	n=1		15.700	36.300	15.700	36.300	15.700	36.300	9.350	28.900
		n=96		67.000	87.700	67.000	87.700	67.000	87.700	62.500	80.300
	DBKCMP> (S1) (S2) (D) n	n=1		15.800	36.300	15.800	36.300	15.800	36.300	9.350	29.000
		n=96		67.000	87.700	67.000	87.700	67.000	87.700	62.600	80.300
	DBKCMP<= (S1) (S2) (D) n	n=1		15.700	36.300	15.700	36.300	15.700	36.300	9.350	29.000
		n=96		64.800	85.700	64.800	85.700	64.800	85.700	60.800	78.400
	DBKCMP< (S1) (S2) (D) n	n=1		15.800	36.300	15.800	36.300	15.800	36.300	9.350	29.000
		n=96		67.000	87.700	67.000	87.700	67.000	87.700	62.700	80.400
	DBKCMP>= (S1) (S2) (D) n	n=1		15.700	36.300	15.700	36.300	15.700	36.300	9.300	29.000
		n=96		64.800	85.700	64.800	85.700	64.800	85.700	60.700	78.400
	DB+ (S) (D)	When executed		5.750	13.300	5.750	13.300	5.750	13.300	4.900	7.500
	DB+ (S1) (S2) (D)	When executed		5.650	13.200	5.650	13.200	5.650	13.200	5.200	11.000
	DB- (S) (D)	When executed		5.750	12.700	5.750	12.700	5.750	12.700	4.900	10.200
	DB- (S1) (S2) (D)	When executed		5.650	12.600	5.650	12.600	5.650	12.600	5.200	8.600
	DB* (S1) (S2) (D)	When executed		8.750	40.200	8.750	40.200	8.750	40.200	8.300	22.200
	DB/ (S1) (S2) (D)	When executed		5.750	21.500	5.750	21.500	5.750	21.500	6.100	19.200
	ED+ (S) (D)	Double precision	(S)=0, (D)=0	4.500	26.700	4.500	26.700	4.500	26.700	4.800	16.800
			(S)=2^{1023}, (D)=2^{1023}	5.800	32.900	5.800	32.900	5.800	32.900	4.800	16.800
	ED+ (S1) (S2) (D)	Double precision	(S1)=0, (S2)=0	5.450	35.400	5.450	35.400	5.450	35.400	7.100	20.100
			(S1)=2^{1023}, (S2)=2^{1023}	6.750	41.400	6.750	41.400	6.750	41.400	7.100	20.100
	ED- (S) (D)	Double precision	(S)=0, (D)=0	5.200	25.900	5.200	25.900	5.200	25.900	5.000	17.300
			(S)=2^{1023}, (D)=2^{1023}	6.000	27.700	6.000	27.700	6.000	27.700	5.000	17.300
	ED- (S1) (S2) (D)	Double precision	(S1)=0, (S2)=0	5.550	32.900	5.550	32.900	5.550	32.900	6.000	16.300
			(S1)=2^{1023}, (S2)=2^{1023}	5.750	33.900	5.750	33.900	5.750	33.900	6.000	16.300
	ED* (S1) (S2) (D)	Double precision	(S1)=0, (S2)=0	5.550	34.400	5.550	34.400	5.550	34.400	10.500	22.300
			(S1)=2^{1023}, (S2)=2^{1023}	5.950	39.100	5.950	39.100	5.950	39.100	10.500	22.300
	ED/ (S1) (S2) (D)	Double precision	(S1)=2^{1023}, (S2)=2^{1023}	8.050	44.200	8.050	44.200	8.050	44.200	7.500	27.200
	BK+ (S1) (S2) (D) n	n=1		13.500	28.500	13.500	28.500	13.500	28.500	12.100	19.700
		n=96		63.100	78.200	63.100	78.200	63.100	78.200	61.700	69.300
	BK- (S1) (S2) (D) n	n=1		13.500	28.500	13.500	28.500	13.500	28.500	12.100	20.600
		n=96		63.100	78.200	63.100	78.200	63.100	78.200	61.700	70.200
	DBK+ (S1) (S2) (D) n	n=1		10.100	24.200	10.100	24.200	10.100	24.200	7.050	19.200
		n=96		59.800	73.900	59.800	73.900	59.800	73.900	59.400	68.900
	DBK- (S1) (S2) (D) n	n=1		10.100	24.200	10.100	24.200	10.100	24.200	7.050	19.900
		n=96		59.800	73.900	59.800	73.900	59.800	73.900	59.400	69.600

Category	Instruction	Condition (device)		Processing time (μs)							
				Q00UJCPU		Q00UCPU		Q01UCPU		Q02UCPU	
				Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.
Basic instruction	$+ (S) (D)	—		15.400	64.300	15.400	64.300	15.400	64.300	14.400	34.000
	$+ (S1) (S2) (D)	—		19.700	71.000	19.700	71.000	19.700	71.000	9.200	22.900
	FLTD	Double precision	(S)=0	3.100	19.600	3.100	19.600	3.100	19.600	4.000	8.900
			(S)=7FFFH	3.350	19.900	3.350	19.900	3.350	19.900	3.400	9.000
	DFLTD	Double precision	(S)=0	3.200	20.400	3.200	20.400	3.200	20.400	4.100	10.800
			(S)=7FFFFFFFH	3.450	20.500	3.450	20.500	3.450	20.500	3.600	10.800
	INTD	Double precision	(S)=0	3.200	22.900	3.200	22.900	3.200	22.900	3.500	9.300
			(S)=32766.5	4.100	34.300	4.100	34.300	4.100	34.300	5.100	19.500
	DINTD	Double precision	(S)=0	3.200	23.000	3.200	23.000	3.200	23.000	2.600	6.800
			(S)=1234567890.3	4.050	33.500	4.050	33.500	4.050	33.500	3.400	11.700
	DBL	When executed		3.300	5.900	3.300	5.900	3.300	5.900	2.700	3.800
	WORD	When executed		3.000	7.250	3.000	7.250	3.000	7.250	2.900	7.000
	GRY	When executed		3.350	7.500	3.350	7.500	3.350	7.500	2.700	6.100
	DGRY	When executed		3.000	7.200	3.000	7.200	3.000	7.200	2.900	4.600
	GBIN	When executed		4.600	9.700	4.600	9.700	4.600	9.700	4.000	8.200
	DGBIN	When executed		5.550	10.700	5.550	10.700	5.550	10.700	5.500	8.000
	NEG	When executed		3.300	6.850	3.300	6.850	3.300	6.850	2.400	4.100
	DNEG	When executed		3.050	5.700	3.050	5.700	3.050	5.700	2.500	4.300
	ENEG	Floating point = 0		3.100	7.350	3.100	7.350	3.100	7.350	2.500	3.400
		Floating point = -1.0		3.350	11.700	3.350	11.700	3.350	11.700	2.700	4.500
	EDNEG	Floating point = 0		3.000	21.200	3.000	21.200	3.000	21.200	2.200	3.500
		Floating point = -1.0		3.100	22.900	3.100	22.900	3.100	22.900	2.400	3.500
	BKBCD (S) (D) n	n=1		8.700	27.600	8.700	27.600	8.700	27.600	9.700	22.000
		n=96		84.200	104.000	84.200	104.000	84.200	104.000	74.200	86.500
	BKBIN (S) (D) n	n=1		8.450	28.100	8.450	28.100	8.450	28.100	8.900	16.300
		n=96		56.100	75.800	56.100	75.800	56.100	75.800	58.500	65.100
	ECON	—		3.100	21.300	3.100	21.300	3.100	21.300	4.300	6.800
	EDCON	—		5.050	24.000	5.050	24.000	5.050	24.000	2.800	5.400
	EDMOV	—		2.900	22.900	2.900	22.900	2.900	22.900	3.200	7.800
	$MOV	Character string to be transferred = 0		6.250	30.100	6.250	30.100	6.250	30.100	4.500	13.900
		Character string to be transferred = 32		15.500	39.300	15.500	39.300	15.500	39.300	15.400	17.500
	BXCH (D1) (D2) n	n=1		8.400	20.900	8.400	20.900	8.400	20.900	8.700	15.200
		n=96		67.100	79.900	67.100	79.900	67.100	79.900	67.200	74.000
	SWAP	—		3.300	3.550	3.300	3.550	3.300	3.550	2.400	2.700
	GOEND	—		0.550		0.550		0.550		0.500	
	DI	—		2.800	8.400	2.800	8.400	2.800	8.400	1.800	2.200
	EI	—		4.300	12.300	4.300	12.300	4.300	12.300	3.100	3.800
	IMASK	—		12.900	40.600	12.900	40.600	12.900	40.600	9.800	25.000
	IRET	—		1.000		1.000		1.000		1.000	
	RFS X n	n=1		7.500	26.500	7.500	26.500	7.500	26.500	4.300	16.100
		n=96		11.400	30.400	11.400	30.400	11.400	30.400	11.400	23.700
	RFS Y n	n=1		7.300	26.300	7.300	26.300	7.300	26.300	3.800	10.000
		n=96		10.900	29.900	10.900	29.900	10.900	29.900	8.500	15.200
	UDCNT1	—		1.500	7.100	1.500	7.100	1.500	7.100	1.000	2.000
	UDCNT2	—		1.500	6.300	1.500	6.300	1.500	6.300	1.000	4.000

Category	Instruction	Condition (device)		Processing time (μs)							
				Q00UJCPU		Q00UCPU		Q01UCPU		Q02UCPU	
				Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.
Basic instruction	TTMR	—		5.300	20.900	5.300	20.900	5.300	20.900	3.900	6.100
	STMR	—		8.900	49.800	8.900	49.800	8.900	49.800	7.200	30.000
	ROTC	—		52.300	52.600	52.300	52.600	52.300	52.600	15.200	16.100
	RAMP	—		7.400	30.900	7.400	30.900	7.400	30.900	5.900	18.300
	SPD	—		1.500	6.300	1.500	6.300	1.500	6.300	1.000	2.800
	PLSY	—		6.400	7.100	6.400	7.100	6.400	7.100	3.500	4.700
	PWM	—		3.900	4.600	3.900	4.600	3.900	4.600	3.400	3.400
	MTR	—		10.100	61.400	10.100	61.400	10.100	61.400	20.500	28.400
Application instruction	BKAND (S1) (S2) (D) n	n=1		13.600	28.500	13.600	28.500	13.600	28.500	12.100	20.100
		n=96		63.200	78.200	63.200	78.200	63.200	78.200	57.400	63.200
	BKOR (S1) (S2) (D) n	n=1		13.500	28.500	13.500	28.500	13.500	28.500	7.700	13.200
		n=96		63.100	78.200	63.100	78.200	63.100	78.200	57.400	62.800
	BKXOR (S1) (S2) (D) n	n=1		13.600	28.300	13.600	28.300	13.600	28.300	7.800	13.200
		n=96		63.100	78.000	63.100	78.000	63.100	78.000	57.300	62.800
	BKXNR (S1) (S2) (D) n	n=1		13.500	28.300	13.500	28.300	13.500	28.300	7.800	14.100
		n=96		63.100	78.000	63.100	78.000	63.100	78.000	57.400	62.900
	BSFR (D) n	n=1		5.050	21.100	5.050	21.100	5.050	21.100	3.700	6.300
		n=96		9.000	34.800	9.000	34.800	9.000	34.800	10.200	12.800
	BSFL (D) n	n=1		4.800	19.100	4.800	19.100	4.800	19.100	4.500	8.900
		n=96		8.550	34.300	8.550	34.300	8.550	34.300	10.100	14.300
	SFTBR (D) n1 n2	n1=16, n2=1		10.300	46.500	10.300	46.500	10.300	46.500	8.800	43.400
		n1=16, n2=15		10.300	46.400	10.300	46.400	10.300	46.400	8.750	43.400
	SFTBL (D) n1 n2	n1=16, n2=1		10.500	49.800	10.500	49.800	10.500	49.800	8.050	45.100
		n1=16, n2=15		10.500	49.800	10.500	49.800	10.500	49.800	8.050	45.100
	SFTWR (D) n1 n2	n1=16, n2=1		7.950	24.000	7.950	24.000	7.950	24.000	6.500	22.800
		n1=16, n2=15		7.950	24.100	7.950	24.100	7.950	24.100	6.500	22.800
	SFTWL (D) n1 n2	n1=16, n2=1		8.700	23.600	8.700	23.600	8.700	23.600	7.350	23.600
		n1=16, n2=15		8.650	23.700	8.650	23.700	8.650	23.700	7.300	23.700
	BSET (D) n	n=1		4.550	4.750	4.550	4.750	4.550	4.750	3.000	3.400
		n=15		4.550	4.750	4.550	4.750	4.550	4.750	3.000	3.500
	BRST (D) n	n=1		4.600	4.750	4.600	4.750	4.600	4.750	3.000	3.400
		n=15		4.600	4.750	4.600	4.750	4.600	4.750	3.000	3.400
	TEST	When executed		7.250	13.200	7.250	13.200	7.250	13.200	4.400	6.900
	DTEST	When executed		6.950	12.900	6.950	12.900	6.950	12.900	4.500	7.000
	BKRST (S) n	n=1		7.350	11.600	7.350	11.600	7.350	11.600	4.300	5.200
		n=96		10.100	22.600	10.100	22.600	10.100	22.600	6.500	13.200
	SER (S1) (S2) (D) n	n=1	All match	6.650	6.800	6.650	6.800	6.650	6.800	5.000	5.300
			None match	6.650	6.800	6.650	6.800	6.650	6.800	5.000	5.300
		n=96	All match	34.000	42.300	34.000	42.300	34.000	42.300	32.300	35.900
			None match	34.000	42.300	34.000	42.300	34.000	42.300	32.400	35.900
	DSER (S1) (S2) (D) n	n=1	All match	8.000	16.300	8.000	16.300	8.000	16.300	6.800	10.200
			None match	8.000	16.300	8.000	16.300	8.000	16.300	6.800	10.200
		n=96	All match	54.100	62.600	54.100	62.600	54.100	62.600	52.800	56.300
			None match	54.100	62.600	54.100	62.600	54.100	62.600	52.800	56.300
	DSUM (S) (D)	(S)=0		4.100	4.200	4.100	4.200	4.100	4.200	3.700	4.100
		(S)=FFFFFFFFH		4.100	4.200	4.100	4.200	4.100	4.200	3.800	4.100
	DECO (S) (D) n	n=2		8.850	23.000	8.850	23.000	8.850	23.000	6.000	16.400
		n=8		13.600	36.600	13.600	36.600	13.600	36.600	8.100	15.200

Category	Instruction	Condition (device)		Processing time (μs)							
				Q00UJCPU		Q00UCPU		Q01UCPU		Q02UCPU	
				Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.
Application instruction	ENCO (S) (D) n	n=2	M1=ON	7.650	11.900	7.650	11.900	7.650	11.900	5.300	6.300
			M4=ON	7.500	11.700	7.500	11.700	7.500	11.700	5.200	6.200
		n=8	M1=ON	14.600	27.800	14.600	27.800	14.600	27.800	10.400	17.900
			M256=ON	10.600	23.700	10.600	23.700	10.600	23.700	5.700	13.300
	DIS (S) (D) n	n=1		6.500	14.800	6.500	14.800	6.500	14.800	5.000	10.900
		n=4		6.900	15.200	6.900	15.200	6.900	15.200	5.400	11.300
	UNI (S) (D) n	n=1		6.800	15.100	6.800	15.100	6.800	15.100	5.500	8.900
		n=4		7.500	15.900	7.500	15.900	7.500	15.900	6.200	9.600
	NDIS	When executed		4.750	18.700	4.750	18.700	4.750	18.700	11.000	16.300
	NUNI	When executed		4.750	18.700	4.750	18.700	4.750	18.700	10.600	16.000
	WTOB (S) (D) n	n=1		6.600	14.900	6.600	14.900	6.600	14.900	5.000	6.500
		n=96		37.700	46.100	37.700	46.100	37.700	46.100	36.000	38.400
	BTOW (S) (D) n	n=1		7.350	15.600	7.350	15.600	7.350	15.600	5.100	6.100
		n=96		32.100	40.500	32.100	40.500	32.100	40.500	29.900	32.000
	MAX (S) (D) n	n=1		8.250	24.900	8.250	24.900	8.250	24.900	4.300	6.900
		n=96		34.200	51.600	34.200	51.600	34.200	51.600	32.000	34.300
	MIN (S) (D) n	n=1		8.250	24.800	8.250	24.800	8.250	24.800	4.400	6.800
		n=96		34.200	51.600	34.200	51.600	34.200	51.600	30.300	34.800
	DMAX (S) (D) n	n=1		6.800	34.900	6.800	34.900	6.800	34.900	4.800	14.200
		n=96		60.300	89.200	60.300	89.200	60.300	89.200	56.400	68.000
	DMIN (S) (D) n	n=1		7.600	35.700	7.600	35.700	7.600	35.700	4.800	9.300
		n=96		59.400	90.000	59.400	90.000	59.400	90.000	55.400	62.800
	SORT (S1) n (S2) (D1) (D2)	n=1, (S2)=1		9.400	28.900	9.400	28.900	9.400	28.900	6.200	24.900
		n=96, (S2)=16		31.500	74.000	31.500	74.000	31.500	74.000	27.500	70.100
	DSORT (S1) n (S2) (D1) (D2)	n=1, (S2)=1		9.400	29.000	9.400	29.000	9.400	29.000	6.200	25.900
		n=96, (S2)=16		37.800	81.000	37.800	81.000	37.800	81.000	33.100	78.900
	WSUM (S) (D) n	n=1		6.700	15.000	6.700	15.000	6.700	15.000	4.800	6.200
		n=96		28.900	37.100	28.900	37.100	28.900	37.100	26.900	28.700
	DWSUM (S) (D) n	n=1		8.600	26.800	8.600	26.800	8.600	26.800	5.500	7.000
		n=96		56.200	74.700	56.200	74.700	56.200	74.700	53.000	56.300
	MEAN (S) (D) n	n=1		5.850	19.800	5.850	19.800	5.850	19.800	4.300	17.300
		n=96		17.300	38.200	17.300	38.200	17.300	38.200	16.000	35.500
	DMEAN (S) (D) n	n=1		6.900	23.300	6.900	23.300	6.900	23.300	5.750	21.900
		n=96		29.400	49.900	29.400	49.900	29.400	49.900	29.200	48.600
	NEXT	—		1.000	1.100	1.000	1.100	1.000	1.100	0.980	1.400
	BREAK	—		4.700	25.000	4.700	25.000	4.700	25.000	21.300	17.900
	RET	Return to original program		4.100	19.500	4.100	19.500	4.100	19.500	2.000	3.000
		Return to other program		4.700	16.700	4.700	16.700	4.700	16.700	2.300	4.900
	FCALL Pn	Internal file pointer		5.400	5.400	5.400	5.400	5.400	5.400	3.300	5.300
		Common pointer		7.600	30.500	7.600	30.500	7.600	30.500	4.900	6.600
	FCALL Pn (S1) to (S5)	—		50.400	62.700	50.400	62.700	50.400	62.700	19.800	23.700
	ECALL * Pn *: Program name	—		105.000	214.000	105.000	214.000	105.000	214.000	75.700	134.000
	ECALL * Pn (S1) to (S5) *: Program name	—		164.000	271.000	164.000	271.000	164.000	271.000	109.000	173.000

Category	Instruction	Condition (device)	Processing time (μs)							
			Q00UJCPU		Q00UCPU		Q01UCPU		Q02UCPU	
			Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.
Application instruction	EFCALL * Pn *: Program name	—	105.000	214.000	105.000	214.000	105.000	214.000	76.200	134.000
	EFCALL * Pn (S1) to (S5) *: Program name	—	164.000	271.000	164.000	271.000	164.000	271.000	90.500	170.000
	XCALL	—	5.100	6.700	5.100	6.700	5.100	6.700	3.800	6.400
	FIFW	Number of data points = 0	6.100	14.200	6.100	14.200	6.100	14.200	3.700	10.100
		Number of data points = 96	6.100	14.200	6.100	14.200	6.100	14.200	3.800	5.200
	FIFR	Number of data points = 1	7.500	15.600	7.500	15.600	7.500	15.600	4.400	5.800
		Number of data points = 96	37.000	45.000	37.000	45.000	37.000	45.000	33.500	35.200
	FPOP	Number of data points = 1	7.600	15.600	7.600	15.600	7.600	15.600	4.400	10.800
		Number of data points = 96	7.600	15.600	7.600	15.600	7.600	15.600	4.400	10.800
	FINS	Number of data points = 0	6.900	15.000	6.900	15.000	6.900	15.000	5.000	10.700
		Number of data points = 96	36.600	44.700	36.600	44.700	36.600	44.700	4.400	10.900
	FDEL	Number of data points = 1	8.000	16.100	8.000	16.100	8.000	16.100	4.900	11.300
		Number of data points = 96	37.300	45.500	37.300	45.500	37.300	45.500	34.200	35.900
	FROM n1 n2 (D) n3	n3=1	17.400	74.700	17.400	74.700	17.400	74.700	12.100	71.300
		n3=1000	406.000	498.500	406.000	498.500	406.000	498.500	402.600	495.100
	DFRO n1 n2 (D) n3	n3=1	19.600	85.600	19.600	85.600	19.600	85.600	14.600	81.800
		n3=500	406.000	498.500	406.000	498.500	406.000	498.500	402.600	495.100
	TO n1 n2 (S) n3	n3=1	16.400	69.600	16.400	69.600	16.400	69.600	11.700	63.400
		n3=1000	381.300	471.200	381.300	471.200	381.300	471.200	375.900	464.300
	DTO n1 n2 (S) n3	n3=1	18.600	85.100	18.600	85.100	18.600	85.100	14.200	78.500
		n3=500	381.300	471.200	381.300	471.200	381.300	471.200	375.900	464.300

A

Category	Instruction	Condition (device)	Processing time (μs)							
			Q00UJCPU		Q00UCPU		Q01UCPU		Q02UCPU	
			Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.
Application instruction	CCOM COM	When selecting I/O refresh only	18.100	89.100	18.100	89.100	18.100	89.100	12.800	79.000
		When selecting CC-Link refresh only (master station side)	33.300	132.000	33.300	132.000	33.300	132.000	24.900	119.000
		When selecting CC-Link refresh only (local station side)	33.300	132.000	33.300	132.000	33.300	132.000	24.900	119.000
		• When selecting MELSECNET/H refresh only (control station side) • When selecting CC-Link IE Controller Network refresh only (control station side)	78.600	231.000	78.600	231.000	78.600	231.000	54.000	212.000
		• When selecting MELSECNET/H refresh only (normal station side) • When selecting CC-Link IE Controller Network refresh only (normal station side)	78.600	231.000	78.600	231.000	78.600	231.000	54.000	212.000
		When selecting CC-Link IE Field Network refresh only (master station side)	32.000	127.000	32.000	127.000	32.000	127.000	22.000	118.000
		When selecting CC-Link IE Field Network refresh only (local station side)	32.000	127.000	32.000	127.000	32.000	127.000	22.000	118.000
		When selecting intelli auto refresh only	18.100	89.000	18.100	89.000	18.100	89.000	12.800	79.000
		When selecting I/O outside the group only (Input only)	15.700	71.600	15.700	71.600	15.700	71.600	8.600	76.500
		When selecting I/O outside the group only (Output only)	40.200	152.000	40.200	152.000	40.200	152.000	26.300	135.000
		When selecting I/O outside the group only (Both I/O)	45.800	153.000	45.800	153.000	45.800	153.000	26.100	135.000
		When selecting refresh of multiple CPU high speed transmission area only	—	—	—	—	—	—	—	—
		When selecting communication with external devices only	18.200	89.000	18.200	89.000	18.200	89.000	7.250	54.300
	LEDR	No display → no display	1.500	7.100	1.500	7.100	1.500	7.100	5.100	5.100
		LED instruction execution → no display	38.900	109.000	38.900	109.000	38.900	109.000	35.700	89.200
	BINDA (S) (D)	(S)=1	5.600	13.900	5.600	13.900	5.600	13.900	4.900	6.500
		(S)=-32768	7.800	16.200	7.800	16.200	7.800	16.200	7.200	8.700
	DBINDA (S) (D)	(S)=1	6.200	14.500	6.200	14.500	6.200	14.500	5.700	7.100
		(S)=-2147483648	11.000	19.200	11.000	19.200	11.000	19.200	10.400	12.200
	BINHA (S) (D)	(S)=1	5.050	13.400	5.050	13.400	5.050	13.400	4.400	5.900
		(S)=FFFFH	5.050	13.400	5.050	13.400	5.050	13.400	4.400	5.800
	DBINHA (S) (D)	(S)=1	5.600	13.900	5.600	13.900	5.600	13.900	5.200	6.700
		(S)=FFFFFFFFH	5.600	13.900	5.600	13.900	5.600	13.900	5.100	6.500
	BCDDA (S) (D)	(S)=1	4.850	13.200	4.850	13.200	4.850	13.200	4.300	5.800
		(S)=9999	5.300	13.600	5.300	13.600	5.300	13.600	4.700	6.100
	DBCDDA (S) (D)	(S)=1	5.300	13.600	5.300	13.600	5.300	13.600	4.800	6.300
		(S)=99999999	6.200	14.500	6.200	14.500	6.200	14.500	5.600	7.100
	DABIN (S) (D)	(S)=1	7.000	18.500	7.000	18.500	7.000	18.500	6.500	9.000
		(S)=-32768	6.950	18.500	6.950	18.500	6.950	18.500	6.300	8.900
	DDABIN (S) (D)	(S)=1	9.450	21.000	9.450	21.000	9.450	21.000	9.400	12.000
		(S)=-2147483648	9.450	21.000	9.450	21.000	9.450	21.000	9.100	11.600
	HABIN (S) (D)	(S)=1	5.650	17.100	5.650	17.100	5.650	17.100	4.900	7.500
		(S)=FFFFH	5.750	17.300	5.750	17.300	5.750	17.300	5.100	8.100

Category	Instruction	Condition (device)		Processing time (μs)							
				Q00UJCPU		Q00UCPU		Q01UCPU		Q02UCPU	
				Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.
Application instruction	DHABIN (S) (D)	(S)=1		6.800	18.200	6.800	18.200	6.800	18.200	6.000	8.500
		(S)=FFFFFFFFH		7.100	18.600	7.100	18.600	7.100	18.600	6.300	8.900
	DABCD (S) (D)	(S)=1		5.650	17.200	5.650	17.200	5.650	17.200	5.000	7.500
		(S)=9999		5.700	17.200	5.700	17.200	5.700	17.200	5.000	7.500
	DDABCD (S) (D)	(S)=1		6.850	18.300	6.850	18.300	6.850	18.300	6.200	8.800
		(S)=99999999		6.850	18.300	6.850	18.300	6.850	18.300	6.200	8.800
	COMRD	—		185.000	188.000	185.000	188.000	185.000	188.000	97.300	97.400
	LEN	1 character		4.700	16.200	4.700	16.200	4.700	16.200	4.100	6.600
		96 characters		20.600	32.900	20.600	32.900	20.600	32.900	19.800	22.400
	STR	—		9.800	36.500	9.800	36.500	9.800	36.500	6.900	14.400
	DSTR	—		12.100	40.400	12.100	40.400	12.100	40.400	10.200	20.800
	VAL	—		12.200	40.900	12.200	40.900	12.200	40.900	9.800	23.900
	DVAL	—		19.400	45.600	19.400	45.600	19.400	45.600	14.000	33.100
	ESTR	—		29.700	87.800	29.700	87.800	29.700	87.800	22.100	52.400
	EVAL	Decimal point format all 2-digit specification		23.900	70.400	23.900	70.400	23.900	70.400	23.300	36.500
		Exponent format all 6-digit specification		23.700	70.300	23.700	70.300	23.700	70.300	23.300	36.400
	ASC (S) (D) n	n=1		10.200	41.800	10.200	41.800	10.200	41.800	5.600	19.700
		n=96		31.900	66.600	31.900	66.600	31.900	66.600	30.200	44.700
	HEX (S) (D) n	n=1		8.600	43.400	8.600	43.400	8.600	43.400	7.500	23.100
		n=96		77.100	115.000	77.100	115.000	77.100	115.000	37.500	53.300
	RIGHT (S) (D) n	n=1		10.900	29.600	10.900	29.600	10.900	29.600	7.600	11.400
		n=96		41.400	60.300	41.400	60.300	41.400	60.300	36.300	46.000
	LEFT (S) (D) n	n=1		10.600	29.300	10.600	29.300	10.600	29.300	6.500	16.100
		n=96		41.300	60.200	41.300	60.200	41.300	60.200	36.200	46.200
	MIDR	—		11.700	30.600	11.700	30.600	11.700	30.600	9.500	19.100
	MIDW	—		12.400	24.000	12.400	24.000	12.400	24.000	10.300	18.200
	INSTR	No match		22.000	38.200	22.000	38.200	22.000	38.200	19.300	29.000
		Match	Head	13.300	29.600	13.300	29.600	13.300	29.600	10.300	20.000
			End	21.900	38.100	21.900	38.100	21.900	38.100	51.100	60.800
	EMOD	—		11.600	24.000	11.600	24.000	11.600	24.000	10.300	15.300
	EREXP	—		19.700	28.000	19.700	28.000	19.700	28.000	19.300	22.300
	STRINS (S) (D) n	(S)=128/(D)=40/n=1		47.000	102.000	47.000	102.000	47.000	102.000	44.300	96.700
		(S)=128/(D)=40/n=48		70.100	134.000	70.100	134.000	70.100	134.000	58.800	112.000
	STRDEL (S) (D) n	(S)=128/(D)=40/n=1		46.400	93.600	46.400	93.600	46.400	93.600	39.000	78.100
		(S)=128/(D)=40/n=48		44.500	70.600	44.500	70.600	44.500	70.600	36.000	69.200
	SIN	Single precision		6.400	13.900	6.400	13.900	6.400	13.900	4.500	9.900
	COS	Single precision		6.100	13.500	6.100	13.500	6.100	13.500	4.300	8.200
	TAN	Single precision		8.300	15.000	8.300	15.000	8.300	15.000	5.100	7.200
	ASIN	Single precision		7.300	15.600	7.300	15.600	7.300	15.600	6.100	13.700
	ACOS	Single precision		8.100	16.500	8.100	16.500	8.100	16.500	6.800	11.100
	ATAN	Single precision		5.350	12.000	5.350	12.000	5.350	12.000	4.000	6.900
	SIND	Double precision		13.400	51.300	13.400	51.300	13.400	51.300	9.600	26.000
	COSD	Double precision		14.700	51.700	14.700	51.700	14.700	51.700	10.000	26.900
	TAND	Double precision		17.400	54.400	17.400	54.400	17.400	54.400	11.400	25.300
	ASIND	Double precision		22.600	60.300	22.600	60.300	22.600	60.300	12.100	30.800
	ACOSD	Double precision		19.700	60.000	19.700	60.000	19.700	60.000	11.700	28.000
	ATAND	Double precision		15.000	51.800	15.000	51.800	15.000	51.800	9.700	22.000
	RAD	Single precision		3.200	10.300	3.200	10.300	3.200	10.300	2.500	4.800

Category	Instruction	Condition (device)		Processing time (μs)							
				Q00UJCPU		Q00UCPU		Q01UCPU		Q02UCPU	
				Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.
Application instruction	RADD	Double precision		5.200	43.100	5.200	43.100	5.200	43.100	4.100	16.400
	DEG	Single precision		3.200	11.500	3.200	11.500	3.200	11.500	2.500	4.700
	DEGD	Double precision		5.150	43.800	5.150	43.800	5.150	43.800	5.000	18.100
	SQR	Single precision		3.900	12.300	3.900	12.300	3.900	12.300	3.500	9.300
	SQRD	Double precision		7.000	45.700	7.000	45.700	7.000	45.700	5.700	25.400
	EXP (S) (D)	Single precision	(S)=-10	6.350	13.800	6.350	13.800	6.350	13.800	4.000	13.000
			(S)=1	6.350	13.800	6.350	13.800	6.350	13.800	4.000	13.000
	EXPD (S) (D)	Double precision	(S)=-10	15.800	52.700	15.800	52.700	15.800	52.700	8.800	27.600
			(S)=1	15.400	52.500	15.400	52.500	15.400	52.500	8.500	27.300
	LOG (S) (D)	Single precision	(S)=1	5.800	14.900	5.800	14.900	5.800	14.900	4.100	8.100
			(S)=10	7.450	16.500	7.450	16.500	7.450	16.500	6.200	10.300
	LOGD (S) (D)	Double precision	(S)=1	11.000	48.900	11.000	48.900	11.000	48.900	9.500	28.300
			(S)=10	12.600	51.300	12.600	51.300	12.600	51.300	11.100	29.900
	SCL (S1) (S2) (D)	SM750=ON	Point No.1 < (S1) < Point No.2	14.900	50.100	14.900	50.100	14.900	50.100	14.700	48.000
			Point No.9 < (S1) < Point No.10	15.800	50.900	15.800	50.900	15.800	50.900	19.600	50.400
		SM750=OFF	Point No.1 < (S1) < Point No.2	13.900	53.100	13.900	53.100	13.900	53.100	13.700	51.000
			Point No.9 < (S1) < Point No.10	16.600	56.600	16.600	56.600	16.600	56.600	20.400	56.200
	DSCL (S1) (S2) (D)	SM750=ON	Point No.1 < (S1) < Point No.2	13.400	52.400	13.400	52.400	13.400	52.400	12.800	50.300
			Point No.9 < (S1) < Point No.10	14.200	54.100	14.200	54.100	14.200	54.100	17.300	53.500
		SM750=OFF	Point No.1 < (S1) < Point No.2	12.300	53.200	12.300	53.200	12.300	53.200	11.500	51.100
			Point No.9 < (S1) < Point No.10	15.000	57.600	15.000	57.600	15.000	57.600	18.100	57.100
	RSET	Standard RAM		6.800	26.900	6.800	26.900	6.800	26.900	3.000	16.400
		SRAM		—	—	—	—	—	—	3.000	16.400
	QDRSET	SRAM card to standard RAM		—	—	—	—	—	—	230.000	327.000
		Standard RAM to SRAM card		—	—	—	—	—	—	997.000	1066.000
	QCDSET	SRAM card to standard ROM		—	—	—	—	—	—	525.000	690.000
		Standard ROM to SRAM card		—	—	—	—	—	—	490.000	655.000
	DATERD	—		5.600	27.800	5.600	27.800	5.600	27.800	5.100	14.700
	DATEWR	—		7.800	42.100	7.800	42.100	7.800	42.100	7.100	23.000
	DATE+	No digit increase		14.200	41.200	14.200	41.200	14.200	41.200	6.500	13.100
		Digit increase		14.200	41.200	14.200	41.200	14.200	41.200	5.700	21.200
	DATE-	No digit increase		15.100	41.200	15.100	41.200	15.100	41.200	6.500	11.500
		Digit increase		15.100	41.200	15.100	41.200	15.100	41.200	5.700	17.200
	SECOND	—		5.800	20.500	5.800	20.500	5.800	20.500	2.600	5.900
	HOUR	—		6.200	22.500	6.200	22.500	6.200	22.500	3.000	5.300
	RND	—		1.950	5.450	1.950	5.450	1.950	5.450	1.200	2.300
	SRND	—		2.750	4.550	2.750	4.550	2.750	4.550	1.400	2.400
	BSQR (S) (D)	(S)=0		2.500	6.800	2.500	6.800	2.500	6.800	1.800	3.300
		(S)=9999		6.400	15.500	6.400	15.500	6.400	15.500	5.100	8.800
	BDSQR (S) (D)	(S)=0		2.600	6.050	2.600	6.050	2.600	6.050	1.900	3.700
		(S)=99999999		8.450	17.600	8.450	17.600	8.450	17.600	7.500	10.900
	BSIN	—		11.500	32.800	11.500	32.800	11.500	32.800	8.700	20.200
	BCOS	—		10.400	32.500	10.400	32.500	10.400	32.500	7.800	14.400

Category	Instruction	Condition (device)			Processing time (μs)							
					Q00UJCPU		Q00UCPU		Q01UCPU		Q02UCPU	
					Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.
Application instruction	BTAN	—			12.100	33.700	12.100	33.700	12.100	33.700	9.000	17.000
	BASIN	—			13.300	32.800	13.300	32.800	13.300	32.800	12.200	15.100
	BACOS	—			13.400	33.700	13.400	33.700	13.400	33.700	13.100	14.900
	BATAN	—			12.600	31.400	12.600	31.400	12.600	31.400	11.400	15.700
	POW (S1) (S2) (D)	Single precision	(S1)=12.3E+5 (S2)=3.45E+0		12.200	22.100	12.200	22.100	12.200	22.100	8.950	19.500
	POWD (S1) (S2) (D)	Double precision	(S1)=12.3E+5 (S2)=3.45E+0		27.300	61.000	27.300	61.000	27.300	61.000	19.400	55.200
	LOG10	Single precision			8.200	16.500	8.200	16.500	8.200	16.500	5.950	14.800
	LOG10D	Double precision			15.100	48.000	15.100	48.000	15.100	48.000	12.400	46.500
	LIMIT	—			5.350	5.500	5.350	5.500	5.350	5.500	5.200	5.400
	DLIMIT	—			6.000	6.150	6.000	6.150	6.000	6.150	5.700	5.900
	BAND	—			5.450	12.400	5.450	12.400	5.450	12.400	5.400	6.300
	DBAND	—			6.050	11.900	6.050	11.900	6.050	11.900	5.800	6.900
	ZONE	—			6.250	10.700	6.250	10.700	6.250	10.700	5.200	11.100
	DZONE	—			6.000	11.900	6.000	11.900	6.000	11.900	5.700	10.800
	LDDT<=	Comparison of specified date		In conductive status	8.200	25.500	8.200	25.500	6.500	25.500	7.400	23.400
				In non-conductive status	8.200	25.500	8.200	25.500	6.500	25.500	7.400	23.400
		Comparison of current date		In conductive status	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.200
				In non-conductive status	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.200
	ANDDT<=	When not executed			0.480		0.320		0.240		0.160	
		Comparison of specified date		In conductive status	8.200	25.500	8.200	25.500	6.500	25.500	7.200	23.400
				In non-conductive status	8.200	25.500	8.200	25.500	6.500	25.500	7.200	23.400
		Comparison of current date		In conductive status	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.200
				In non-conductive status	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.200
	ORDT<=	When not executed			0.480		0.320		0.240		0.160	
		Comparison of specified date		In conductive status	8.200	25.500	8.200	25.500	6.500	25.500	7.400	23.300
				In non-conductive status	8.200	25.500	8.200	25.500	6.500	25.500	7.400	23.300
		Comparison of current date		In conductive status	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.000
				In non-conductive status	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.000

Category	Instruction	Condition (device)		Processing time (μs)							
				Q00UJCPU		Q00UCPU		Q01UCPU		Q02UCPU	
				Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.
Application instruction	LDDT<	Comparison of specified date	In conductive status	8.200	25.500	8.200	25.500	6.500	25.500	7.400	23.400
			In non-conductive status	8.200	25.500	8.200	25.500	6.500	25.500	7.400	23.400
		Comparison of current date	In conductive status	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.200
			In non-conductive status	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.200
	ANDDT<	When not executed		0.480		0.320		0.240		0.160	
		Comparison of specified date	In conductive status	8.200	25.500	8.200	25.500	6.500	25.500	7.200	23.400
			In non-conductive status	8.200	25.500	8.200	25.500	6.500	25.500	7.200	23.400
		Comparison of current date	In conductive status	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.200
			In non-conductive status	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.200
	ORDT<	When not executed		0.480		0.320		0.240		0.160	
		Comparison of specified date	In conductive status	8.200	25.500	8.200	25.500	6.500	25.500	7.400	23.300
			In non-conductive status	8.200	25.500	8.200	25.500	6.500	25.500	7.400	23.300
		Comparison of current date	In conductive status	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.000
			In non-conductive status	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.000
	LDDT>=	Comparison of specified date	In conductive status	8.200	25.500	8.200	25.500	6.500	25.500	7.400	23.400
			In non-conductive status	8.200	25.500	8.200	25.500	6.500	25.500	7.400	23.400
		Comparison of current date	In conductive status	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.200
			In non-conductive status	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.200
	ANDDT>=	When not executed		0.480		0.320		0.240		0.160	
		Comparison of specified date	In conductive status	8.200	25.500	8.200	25.500	6.500	25.500	7.200	23.400
			In non-conductive status	8.200	25.500	8.200	25.500	6.500	25.500	7.200	23.400
		Comparison of current date	In conductive status	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.200
			In non-conductive status	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.200

Category	Instruction	Condition (device)		Processing time (μs)							
				Q00UJCPU		Q00UCPU		Q01UCPU		Q02UCPU	
				Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.
Application instruction	ORDT>=	When not executed		0.480		0.320		0.240		0.160	
		Comparison of specified date	In conductive status	8.200	25.500	8.200	25.500	6.500	25.500	7.400	23.300
			In non-conductive status	8.200	25.500	8.200	25.500	6.500	25.500	7.400	23.300
		Comparison of current date	In conductive status	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.000
			In non-conductive status	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.000
	LDDT=	Comparison of specified date	In conductive status	8.200	25.500	8.200	25.500	6.500	25.500	7.400	23.400
			In non-conductive status	8.200	25.500	8.200	25.500	6.500	25.500	8.200	25.500
		Comparison of current date	In conductive status	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.200
			In non-conductive status	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.200
	ANDDT=	When not executed		0.480		0.320		0.240		0.160	
		Comparison of specified date	In conductive status	8.200	25.500	8.200	25.500	6.500	25.500	7.200	23.400
			In non-conductive status	8.200	25.500	8.200	25.500	6.500	25.500	7.200	23.400
		Comparison of current date	In conductive status	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.200
			In non-conductive status	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.200
	ORDT=	When not executed		0.480		0.320		0.240		0.160	
		Comparison of specified date	In conductive status	8.200	25.500	8.200	25.500	6.500	25.500	7.400	23.300
			In non-conductive status	8.200	25.500	8.200	25.500	6.500	25.500	7.400	23.300
		Comparison of current date	In conductive status	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.200
			In non-conductive status	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.000

Category	Instruction	Condition (device)		Processing time (μs)							
				Q00UJCPU		Q00UCPU		Q01UCPU		Q02UCPU	
				Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.
Application instruction	LDDT<>	Comparison of specified date	In conductive status	8.200	25.500	8.200	25.500	6.500	25.500	7.400	23.400
			In non-conductive status	8.200	25.500	8.200	25.500	6.500	25.500	7.400	23.400
		Comparison of current date	In conductive status	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.200
			In non-conductive status	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.200
	ANDDT<>	When not executed		0.480		0.320		0.240		0.160	
		Comparison of specified date	In conductive status	8.200	25.500	8.200	25.500	6.500	25.500	7.200	23.400
			In non-conductive status	8.200	25.500	8.200	25.500	6.500	25.500	7.200	23.400
		Comparison of current date	In conductive status	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.200
			In non-conductive status	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.200
	ORDT<>	When not executed		0.480		0.320		0.240		0.160	
		Comparison of specified date	In conductive status	8.200	25.500	8.200	25.500	6.500	25.500	7.400	23.300
			In non-conductive status	8.200	25.500	8.200	25.500	6.500	25.500	7.400	23.300
		Comparison of current date	In conductive status	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.000
			In non-conductive status	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.000
	LDDT>	Comparison of specified date	In conductive status	8.200	25.500	8.200	25.500	6.500	25.500	7.400	23.400
			In non-conductive status	8.200	25.500	8.200	25.500	6.500	25.500	7.400	23.400
		Comparison of current date	In conductive status	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.200
			In non-conductive status	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.200
	ANDDT>	When not executed		0.480		0.320		0.240		0.160	
		Comparison of specified date	In conductive status	8.200	25.500	8.200	25.500	8.200	25.500	7.200	23.400
			In non-conductive status	8.200	25.500	8.200	25.500	8.200	25.500	7.200	23.400
		Comparison of current date	In conductive status	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.200
			In non-conductive status	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.200

Category	Instruction	Condition (device)		Processing time (μs)							
				Q00UJCPU		Q00UCPU		Q01UCPU		Q02UCPU	
				Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.
Application instruction	ORDT>	When not executed		0.480		0.320		0.240		0.160	
		Comparison of specified date	In conductive status	8.200	25.500	8.200	25.500	8.200	25.500	7.400	23.300
			In non-conductive status	8.200	25.500	8.200	25.500	8.200	25.500	7.400	23.300
		Comparison of current date	In conductive status	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.000
			In non-conductive status	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.000
	LDTM<=	Comparison of specified clock	In conductive status	8.200	25.500	8.200	25.500	6.500	25.500	7.300	23.300
			In non-conductive status	8.200	25.500	8.200	25.500	6.500	25.500	7.300	23.300
		Comparison of current clock	In conductive status	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.100
			In non-conductive status	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.100
	ANDTM<=	When not executed		0.480		0.320		0.240		0.160	
		Comparison of specified clock	In conductive status	8.200	25.500	8.200	25.500	6.500	25.500	7.000	23.000
			In non-conductive status	8.200	25.500	8.200	25.500	6.500	25.500	7.000	23.000
		Comparison of current clock	In conductive status	6.500	23.100	6.500	23.100	6.500	23.100	5.600	21.900
			In non-conductive status	6.500	23.100	6.500	23.100	6.500	23.100	5.600	21.900
	ORTM<=	When not executed		0.480		0.320		0.240		0.160	
		Comparison of specified clock	In conductive status	8.200	25.500	8.200	25.500	6.500	25.500	7.300	23.200
			In non-conductive status	8.200	25.500	8.200	25.500	6.500	25.500	7.300	23.200
		Comparison of current clock	In conductive status	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.000
			In non-conductive status	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.000
	LDTM<	Comparison of specified clock	In conductive status	8.200	25.500	8.200	25.500	6.500	25.500	7.300	23.300
			In non-conductive status	8.200	25.500	8.200	25.500	6.500	25.500	7.300	23.300
		Comparison of current clock	In conductive status	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.100
			In non-conductive status	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.100

A

Category	Instruction	Condition (device)		Processing time (μs)							
				Q00UJCPU		Q00UCPU		Q01UCPU		Q02UCPU	
				Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.
Application instruction	ANDTM<	When not executed		0.480		0.320		0.240		0.160	
		Comparison of specified clock	In conductive status	8.200	25.500	8.200	25.500	6.500	25.500	7.000	23.000
			In non-conductive status	8.200	25.500	8.200	25.500	6.500	25.500	7.000	23.000
		Comparison of current clock	In conductive status	6.500	23.100	6.500	23.100	6.500	23.100	5.600	21.900
			In non-conductive status	6.500	23.100	6.500	23.100	6.500	23.100	5.600	21.900
	ORTM<	When not executed		0.480		0.320		0.240		0.160	
		Comparison of specified clock	In conductive status	8.200	25.500	8.200	25.500	6.500	25.500	7.300	23.200
			In non-conductive status	8.200	25.500	8.200	25.500	6.500	25.500	7.300	23.200
		Comparison of current clock	In conductive status	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.000
			In non-conductive status	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.000
	LDTM>=	Comparison of specified clock	In conductive status	8.200	25.500	8.200	25.500	6.500	25.500	7.300	23.300
			In non-conductive status	8.200	25.500	8.200	25.500	6.500	25.500	7.300	23.300
		Comparison of current clock	In conductive status	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.100
			In non-conductive status	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.100
	ANDTM>=	When not executed		0.480		0.320		0.240		0.160	
		Comparison of specified clock	In conductive status	8.200	25.500	8.200	25.500	6.500	25.500	7.000	23.000
			In non-conductive status	8.200	25.500	8.200	25.500	6.500	25.500	7.000	23.000
		Comparison of current clock	In conductive status	6.500	23.100	6.500	23.100	6.500	23.100	5.600	21.900
			In non-conductive status	6.500	23.100	6.500	23.100	6.500	23.100	5.600	21.900
	ORTM>=	When not executed		0.480		0.320		0.240		0.160	
		Comparison of specified clock	In conductive status	8.200	25.500	8.200	25.500	6.500	25.500	7.000	23.000
			In non-conductive status	8.200	25.500	8.200	25.500	6.500	25.500	7.000	23.000
		Comparison of current clock	In conductive status	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.100
			In non-conductive status	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.100

Category	Instruction	Condition (device)		Processing time (μs)							
				Q00UJCPU		Q00UCPU		Q01UCPU		Q02UCPU	
				Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.
Application instruction	LDTM=	Comparison of specified time	In conductive status	8.200	25.500	8.200	25.500	6.500	25.500	7.300	23.300
			In non-conductive status	8.200	25.500	8.200	25.500	6.500	25.500	7.300	23.300
		Comparison of current time	In conductive status	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.100
			In non-conductive status	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.100
	ANDTM=	When not executed		0.480		0.320		0.240		0.160	
		Comparison of specified clock	In conductive status	8.200	25.500	8.200	25.500	6.500	25.500	7.000	23.000
			In non-conductive status	8.200	25.500	8.200	25.500	6.500	25.500	7.000	23.000
		Comparison of current clock	In conductive status	6.500	23.100	6.500	23.100	6.500	23.100	5.600	21.900
			In non-conductive status	6.500	23.100	6.500	23.100	6.500	23.100	5.600	21.900
	ORTM=	When not executed		0.480		0.320		0.240		0.160	
		Comparison of specified clock	In conductive status	8.200	25.500	8.200	25.500	6.500	25.500	7.300	23.200
			In non-conductive status	8.200	25.500	8.200	25.500	6.500	25.500	7.300	23.200
		Comparison of current clock	In conductive status	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.000
			In non-conductive status	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.000
	LDTM<>	Comparison of specified clock	In conductive status	8.200	25.500	8.200	25.500	6.500	25.500	7.300	23.300
			In non-conductive status	8.200	25.500	8.200	25.500	6.500	25.500	7.300	23.300
		Comparison of current clock	In conductive status	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.100
			In non-conductive status	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.100
	ANDTM<>	When not executed		0.480		0.320		0.240		0.160	
		Comparison of specified clock	In conductive status	8.200	25.500	8.200	25.500	6.500	25.500	7.000	23.000
			In non-conductive status	8.200	25.500	8.200	25.500	6.500	25.500	7.000	23.000
		Comparison of current clock	In conductive status	6.500	23.100	6.500	23.100	6.500	23.100	5.600	21.900
			In non-conductive status	6.500	23.100	6.500	23.100	6.500	23.100	5.600	21.900

Category	Instruction	Condition (device)		Processing time (μs)							
				Q00UJCPU		Q00UCPU		Q01UCPU		Q02UCPU	
				Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.
Application instruction	ORTM<>	When not executed		0.480		0.320		0.240		0.160	
		Comparison of specified clock	In conductive status	8.200	25.500	8.200	25.500	6.500	25.500	7.300	23.200
			In non-conductive status	8.200	25.500	8.200	25.500	6.500	25.500	7.300	23.200
		Comparison of current clock	In conductive status	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.000
			In non-conductive status	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.000
	LDTM>	Comparison of specified clock	In conductive status	8.200	25.500	8.200	25.500	6.500	25.500	7.300	23.300
			In non-conductive status	8.200	25.500	8.200	25.500	6.500	25.500	7.300	23.300
		Comparison of current clock	In conductive status	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.100
			In non-conductive status	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.100
	ANDTM>	When not executed		0.480		0.320		0.240		0.160	
		Comparison of specified clock	In conductive status	8.200	25.500	8.200	25.500	6.500	25.500	7.000	23.000
			In non-conductive status	8.200	25.500	8.200	25.500	6.500	25.500	7.000	23.000
		Comparison of current clock	In conductive status	6.500	23.100	6.500	23.100	6.500	23.100	5.600	21.900
			In non-conductive status	6.500	23.100	6.500	23.100	6.500	23.100	5.600	21.900
	ORTM>	When not executed		0.480		0.320		0.240		0.160	
		Comparison of specified clock	In conductive status	8.200	25.500	8.200	25.500	6.500	25.500	7.300	23.200
			In non-conductive status	8.200	25.500	8.200	25.500	6.500	25.500	7.300	23.200
		Comparison of current clock	In conductive status	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.000
			In non-conductive status	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.000
	S.DATERD	—		9.250	51.000	9.250	51.000	9.250	51.000	7.500	23.400
	S.DATE+	No digit increase		16.800	75.400	16.800	75.400	16.800	75.400	9.100	23.400
		Digit increase		16.800	75.400	16.800	75.400	16.800	75.400	8.900	22.200
	S.DATE-	No digit increase		17.600	75.300	17.600	75.300	17.600	75.300	9.000	22.200
		Digit increase		16.900	75.300	16.900	75.300	16.900	75.300	9.800	22.100
	PSTOP	—		82.200	199.000	82.200	199.000	82.200	199.000	61.400	84.500
	POFF	—		82.600	198.000	82.600	198.000	82.600	198.000	121.000	246.000
	PSCAN	—		83.600	200.000	83.600	200.000	83.600	200.000	126.000	232.000
	WDT	—		2.900	12.000	2.900	12.000	2.900	12.000	1.300	3.000
	DUTY	—		7.700	27.500	7.700	27.500	7.700	27.500	4.900	24.300
	TIMCHK	—		5.350	24.500	5.350	24.500	5.350	24.500	7.400	23.300
	ZRRDB	File register of standard RAM		4.100	4.200	4.100	4.200	4.100	4.200	2.400	2.600
		File register of SRAM card		—	—	—	—	—	—	2.500	2.800

Category	Instruction	Condition (device)		Processing time (μs)							
				Q00UJCPU		Q00UCPU		Q01UCPU		Q02UCPU	
				Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.
Application instruction	ZRWRB	File register of standard RAM		5.400	5.500	5.400	5.500	5.400	5.500	3.100	3.300
		File register of SRAM card		—	—	—	—	—	—	3.300	3.600
	ADRSET	—		2.400	6.650	2.400	6.650	2.400	6.650	4.200	4.900
	ZPUSH	—		9.200	20.500	9.200	20.500	9.200	20.500	6.900	14.000
	ZPOP	—		9.000	15.500	9.000	15.500	9.000	15.500	7.500	12.500
	UNIRD n1 (D) n2	n2=1		6.000	33.100	6.000	33.100	6.000	33.100	4.000	29.100
		n2=16		16.500	43.600	16.500	43.600	16.500	43.600	12.500	37.600
	TYPERD	—		48.50	141.30	43.50	139.90	43.40	139.80	32.40	134.20
	TRACE	Start		174.000	174.000	174.000	174.000	174.000	174.000	96.600	103.000
	TRACER	—		5.100	15.500	5.100	15.500	5.100	15.500	3.800	13.600
	RBMOV (S) (D) n	When standard RAM is used	1 point	—	—	12.200	34.900	12.200	34.900	9.400	31.300
			1000 points	—	—	121.500	145.100	121.500	145.100	118.500	141.300
		When SRAM card is used	1 point	—	—	—	—	—	—	9.400	31.400
			1000 points	—	—	—	—	—	—	178.500	201.300
	SP.FWRITE	—		—	—	—	—	—	—	87.000	144.000
	SP.FREAD	—		—	—	—	—	—	—	127.000	140.000
	SP.DEVST	—		125.000	125.000	125.000	125.000	125.000	125.000	8.100	98.000
	S.DEVLD	—		18.300	36.700	18.300	36.700	18.300	36.700	13.000	43.000
Data link instruction	S.ZCOM	When mounting CC-Link module (master station side)		29.400	91.700	29.400	91.700	29.400	91.700	20.600	55.000
		When mounting CC-Link module (local station side)		29.500	91.600	29.500	91.600	29.500	91.600	20.600	66.100
		• When selecting MELSECNET/H refresh only (control station side) • When selecting CC-Link IE Controller Network refresh only (control station side)		79.900	214.000	79.900	214.000	79.900	214.000	102.000	180.000
		• When selecting MELSECNET/H refresh only (normal station side) • When selecting CC-Link IE Controller Network refresh only (normal station side)		79.900	214.000	79.900	214.000	79.900	214.000	55.600	168.100
		When selecting CC-Link IE Field Network refresh only (master station side)		60.000	167.000	60.000	167.000	60.000	167.000	51.000	154.000
		When selecting CC-Link IE Field Network refresh only (local station side)		60.000	167.000	60.000	167.000	60.000	167.000	51.000	154.000
	S.RTREAD	—		12.600	65.000	12.600	65.000	12.600	65.000	8.700	60.500
	S.RTWRITE	—		13.300	67.100	13.300	67.100	13.300	67.100	9.300	65.000
Multiple CPU dedicated instruction	S.TO n1 n2 n3 n4 (D)	Writing to host CPU shared memory	n4=1	64.600	78.100	64.600	78.100	64.600	78.100	64.600	78.100
			n4=320	115.000	126.000	115.000	126.000	115.000	126.000	154.000	126.000
	TO n1 n2 (S) n3	Writing to host CPU shared memory	n3=1	12.700	62.200	12.700	62.200	12.700	62.200	8.300	58.200
			n3=320	63.500	112.300	63.500	112.300	63.500	112.300	56.200	107.800
	DTO n1 n2 (S) n3	Writing to host CPU shared memory	n3=1	13.500	62.300	13.500	62.300	13.500	62.300	8.600	58.300
			n3=320	112.900	160.800	112.900	160.800	112.900	160.800	106.800	157.300
	FROM n1 n2 (D) n3	Reading from host CPU shared memory	n3=1	12.100	58.700	12.100	58.700	12.100	58.700	8.400	52.600
			n3=320	56.000	101.700	56.000	101.700	56.000	101.700	51.700	96.600
		Reading from other CPU shared memory	n3=1	24.400	82.900	24.400	82.900	24.400	82.900	16.600	37.000
			n3=320	152.000	243.000	152.000	243.000	152.000	243.000	153.000	185.000
			n3=1000	418.000	518.000	418.000	518.000	418.000	518.000	432.000	485.000

A

Category	Instruction	Condition (device)		Processing time (μs)							
				Q00UJCPU		Q00UCPU		Q01UCPU		Q02UCPU	
				Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.
Multiple CPU dedicated instruction	DFRO n1 n2 (D) n3	Reading from host CPU shared memory	n3=1	12.100	58.700	12.100	58.700	12.100	58.700	8.800	53.400
			n3=320	97.400	143.700	97.400	143.700	97.400	143.700	94.900	139.600
		Reading from other CPU shared memory	n3=1	24.800	94.200	24.800	94.200	24.800	94.200	16.600	47.300
			n3=320	276.000	367.000	276.000	367.000	276.000	367.000	278.000	339.000
			n3=1000	799.000	892.000	799.000	892.000	799.000	892.000	841.000	892.000

Point

For the instructions for which a rising edge instruction (□P) is not described, the processing time is the same as an ON execution instruction.
(Example) WORDP instruction, TOP instruction etc.

• When using Q03UD(E)HCPU, Q04UD(E)HCPU, Q06UD(E)HCPU, Q10UD(E)HCPU, Q13UD(E)HCPU, Q20UD(E)HCPU, Q26UD(E)HCPU, Q50UDEHCPU, and Q100UDEHCPU

Category	Instruction	Condition (device)			Processing time (μs)							
					Q03UD(E)CPU		Q04UD(E)HCPU, Q06UD(E)HCPU		Q10UD(E)HCPU, Q13UD(E)HCPU, Q20UD(E)HCPU, Q26UD(E)HCPU		Q50UDEHCPU, Q100UDEHCPU	
					Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.
Sequence instruction	ANB ORB MPS MRD MPP	—			0.020		0.0095		0.0095		0.0095	
	INV	When not executed			0.020		0.0095		0.0095		0.0095	
		When executed										
	MEP MEF	When not executed			0.020		0.0095		0.0095		0.0095	
		When executed										
	EGP EGF	When not executed			0.020		0.0095		0.0095		0.0095	
		When executed										
	PLS	—			1.300	1.600	0.890	1.100	0.890	1.100	0.890	1.100
	FF	When not executed			0.040		0.0185		0.0185		0.0185	
		When executed			1.200	1.500	0.790	0.910	0.790	0.910	0.790	0.910
	DELTA	When not executed			0.040		0.0185		0.0185		0.0185	
		When executed			2.800	3.600	2.400	3.200	2.400	3.200	2.400	3.200
	SFT	When not executed			0.040		0.0185		0.0185		0.0185	
		When executed			1.600	3.300	1.100	2.700	1.100	2.700	1.100	2.700
	MC	—			0.040		0.0185		0.0185		0.0185	
	MCR	—			0.040		0.0185		0.0185		0.0185	
	FEND END	Error check performed			108.000	130.000	75.800	89.300	75.800	89.300	75.800	89.300
		No error check performed			107.000	124.000	75.800	89.800	75.800	89.800	75.800	89.800
	NOP NOPLF PAGE	—			0.020		0.0095		0.0095		0.0095	
Basic instruction	LDE=	Single precision	In conductive status		3.700	4.700	3.300	4.300	0.0285		0.0285	
			In non-conductive status		3.800	5.000	3.400	4.500				
	ANDE=	Single precision	When not executed		0.060		0.0285		0.0285		0.0285	
			When executed	In conductive status	3.300	5.800	3.000	5.100				
				In non-conductive status	3.500	5.600	3.000	5.200				
	ORE=	Single precision	When not executed		0.060		0.0285		0.0285		0.0285	
			When executed	In conductive status	3.600	4.500	3.200	4.200				
				In non-conductive status	3.500	4.800	3.200	4.300				
	LDE<>	Single precision	In conductive status		4.000	4.700	3.600	4.200	0.0285		0.0285	
			In non-conductive status		3.900	4.500	3.500	4.000				
	ANDE<>	Single precision	When not executed		0.060		0.0285		0.0285		0.0285	
			When executed	In conductive status	3.300	5.100	3.000	4.800				
				In non-conductive status	3.500	5.000	3.100	4.600				

A

Category	Instruction	Condition (device)			Processing time (μs)							
					Q03UD(E)CPU		Q04UD(E)HCPU, Q06UD(E)HCPU		Q10UD(E)HCPU, Q13UD(E)HCPU, Q20UD(E)HCPU, Q26UD(E)HCPU		Q50UDEHCPU, Q100UDEHCPU	
					Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.
Basic instruction	ORE<>	Single precision	When not executed		0.060		0.0285		0.0285		0.0285	
			When executed	In conductive status	3.600	6.000	3.300	5.500				
				In non-conductive status	3.500	5.800	3.100	5.300				
	LDE>	Single precision	In conductive status		3.800	5.000	3.300	4.600	0.0285		0.0285	
			In non-conductive status		3.700	4.900	3.300	4.400				
	ANDE>	Single precision	When not executed		0.060		0.0285		0.0285		0.0285	
			When executed	In conductive status	3.500	4.700	3.100	4.200				
				In non-conductive status	3.600	4.500	3.100	4.000				
	ORE>	Single precision	When not executed		0.060		0.0285		0.0285		0.0285	
			When executed	In conductive status	3.600	5.100	3.300	4.600				
				In non-conductive status	3.500	4.800	3.200	4.500				
	LDE<=	Single precision	In conductive status		3.800	5.600	3.400	5.200	0.0285		0.0285	
			In non-conductive status		3.800	5.600	3.400	5.100				
	ANDE<=	Single precision	When not executed		0.060		0.0285		0.0285		0.0285	
			When executed	In conductive status	3.200	4.600	2.800	4.200				
				In non-conductive status	3.500	5.000	3.100	4.500				
	ORE<=	Single precision	When not executed		0.060		0.0285		0.0285		0.0285	
			When executed	In conductive status	3.700	5.800	3.400	5.400				
				In non-conductive status	3.800	5.700	3.300	5.300				
	LDE<	Single precision	In conductive status		4.000	5.400	3.500	4.900	0.0285		0.0285	
			In non-conductive status		4.000	5.200	3.500	4.900				
	ANDE<	Single precision	When not executed		0.060		0.0285		0.0285		0.0285	
			When executed	In conductive status	3.400	4.600	3.000	4.200				
				In non-conductive status	3.500	4.900	3.100	4.400				
	ORE<	Single precision	When not executed		0.060		0.0285		0.0285		0.0285	
			When executed	In conductive status	3.600	5.200	3.300	4.900				
				In non-conductive status	3.400	4.900	3.200	4.500				

Category	Instruction	Condition (device)			Processing time (μs)							
					Q03UD(E)CPU		Q04UD(E)HCPU, Q06UD(E)HCPU		Q10UD(E)HCPU, Q13UD(E)HCPU, Q20UD(E)HCPU, Q26UD(E)HCPU		Q50UDEHCPU, Q100UDEHCPU	
					Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.
Basic instruction	LDE>=	Single precision	In conductive status		3.800	6.000	3.300	5.500	0.0285		0.0285	
			In non-conductive status		3.800	5.900	3.400	5.400				
	ANDE>=	Single precision	When not executed		0.060		0.0285		0.0285		0.0285	
			When executed	In conductive status	3.200	4.800	2.900	4.600				
				In non-conductive status	3.500	5.400	3.100	5.100				
	ORE>=	Single precision	When not executed		0.060		0.0285		0.0285		0.0285	
			When executed	In conductive status	3.600	5.200	3.300	4.700				
				In non-conductive status	3.500	5.200	3.200	4.700				
	LDED=	Double precision	In conductive status		4.100	7.700	3.500	7.200	3.500	7.200	3.500	7.200
			In non-conductive status		4.300	8.100	3.800	7.400	3.800	7.400	3.800	7.400
	ANDED=	Double precision	When not executed		0.060		0.0285		0.0285		0.0285	
			When executed	In conductive status	3.600	7.600	3.200	7.000	3.200	7.000	3.200	7.000
				In non-conductive status	3.900	7.700	3.400	7.400	3.400	7.400	3.400	7.400
	ORED=	Double precision	When not executed		0.060		0.0285		0.0285		0.0285	
			When executed	In conductive status	3.800	8.800	3.400	8.300	3.400	8.300	3.400	8.300
				In non-conductive status	4.000	9.300	3.700	8.800	3.700	8.800	3.700	8.800
	LDED<>	Double precision	In conductive status		4.400	8.200	3.900	7.700	3.900	7.700	3.900	7.700
			In non-conductive status		4.100	7.900	3.500	7.500	3.500	7.500	3.500	7.500
	ANDED<>	Double precision	When not executed		0.060		0.0285		0.0285		0.0285	
			When executed	In conductive status	3.800	7.600	3.300	7.200	3.300	7.200	3.300	7.200
				In non-conductive status	3.800	7.700	3.400	7.300	3.400	7.300	3.400	7.300
	ORED<>	Double precision	When not executed		0.060		0.0285		0.0285		0.0285	
			When executed	In conductive status	4.100	9.300	3.700	8.900	3.700	8.900	3.700	8.900
				In non-conductive status	3.800	8.900	3.400	8.400	3.400	8.400	3.400	8.400
	LDED>	Double precision	In conductive status		4.300	8.100	3.800	7.500	3.800	7.500	3.800	7.500
			In non-conductive status		4.100	7.800	3.500	7.200	3.500	7.200	3.500	7.200

A

Category	Instruction	Condition (device)			Processing time (μs)							
					Q03UD(E)CPU		Q04UD(E)HCPU, Q06UD(E)HCPU		Q10UD(E)HCPU, Q13UD(E)HCPU, Q20UD(E)HCPU, Q26UD(E)HCPU		Q50UDEHCPU, Q100UDEHCPU	
					Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.
Basic instruction	ANDED>	Double precision	When not executed		0.060		0.0285		0.0285		0.0285	
			When executed	In conductive status	3.800	7.700	3.300	7.300	3.300	7.300	3.300	7.300
				In non-conductive status	4.000	7.900	3.500	7.500	3.500	7.500	3.500	7.500
	ORED>	Double precision	When not executed		0.060		0.0285		0.0285		0.0285	
			When executed	In conductive status	4.100	9.300	3.700	8.800	3.700	8.800	3.700	8.800
				In non-conductive status	4.100	9.300	3.700	8.800	3.700	8.800	3.700	8.800
	LDED<=	Double precision	In conductive status		4.000	8.000	3.500	7.400	3.500	7.400	3.500	7.400
			In non-conductive status		4.100	9.400	3.600	8.800	3.600	8.800	3.600	8.800
	ANDED<=	Double precision	When not executed		0.060		0.0285		0.0285		0.0285	
			When executed	In conductive status	3.800	7.700	3.300	7.200	3.300	7.200	3.300	7.200
				In non-conductive status	3.900	7.700	3.500	7.400	3.500	7.400	3.500	7.400
	ORED<=	Double precision	When not executed		0.060		0.0285		0.0285		0.0285	
			When executed	In conductive status	4.100	9.600	3.700	9.200	3.700	9.200	3.700	9.200
				In non-conductive status	4.100	9.600	3.700	9.200	3.700	9.200	3.700	9.200
	LDED<	Double precision	In conductive status		4.300	8.300	3.800	7.600	3.800	7.600	3.800	7.600
			In non-conductive status		3.700	7.900	3.500	7.400	3.500	7.400	3.500	7.400
	ANDED<	Double precision	When not executed		0.060		0.0285		0.0285		0.0285	
			When executed	In conductive status	3.800	7.800	3.300	7.300	3.300	7.300	3.300	7.300
				In non-conductive status	3.900	7.900	3.400	3.900	3.400	3.900	3.400	3.900
	ORED<	Double precision	When not executed		0.060		0.0285		0.0285		0.0285	
			When executed	In conductive status	4.100	9.600	3.700	9.200	3.700	9.200	3.700	9.200
				In non-conductive status	4.000	9.600	3.700	9.200	3.700	9.200	3.700	9.200
	LDED>=	Double precision	In conductive status		4.100	9.600	3.600	9.000	3.600	9.000	3.600	9.000
			In non-conductive status		4.100	9.600	3.600	8.900	3.600	8.900	3.600	8.900
	ANDED>=	Double precision	When not executed		0.060		0.0285		0.0285		0.0285	
			When executed	In conductive status	3.800	7.900	3.400	7.400	3.400	7.400	3.400	7.400
				In non-conductive status	3.900	8.100	3.400	7.500	3.400	7.500	3.400	7.500

Category	Instruction	Condition (device)			Processing time (μs)							
					Q03UD(E)CPU		Q04UD(E)HCPU, Q06UD(E)HCPU		Q10UD(E)HCPU, Q13UD(E)HCPU, Q20UD(E)HCPU, Q26UD(E)HCPU		Q50UDEHCPU, Q100UDEHCPU	
					Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.
Basic instruction	ORED>=	Double precision	When not executed		0.060		0.0285		0.0285		0.0285	
			When executed	In conductive status	4.100	9.600	3.700	9.200	3.700	9.200	3.700	9.200
				In non-conductive status	4.000	7.200	3.600	6.600	3.600	6.600	3.600	6.600
	LD$=	In conductive status			5.300	8.900	4.700	8.100	4.700	8.100	4.700	8.100
		In non-conductive status			4.700	9.000	4.200	8.200	4.200	8.200	4.200	8.200
	AND$=	When not executed			0.060		0.0285		0.0285		0.0285	
		When executed		In conductive status	4.400	6.800	3.900	6.400	3.900	6.400	3.900	6.400
				In non-conductive status	4.500	6.700	4.000	6.300	4.000	6.300	4.000	6.300
	OR$=	When not executed			0.060		0.0285		0.0285		0.0285	
		When executed		In conductive status	5.100	8.200	4.200	7.600	4.200	7.600	4.200	7.600
				In non-conductive status	5.000	8.100	4.000	7.200	4.000	7.200	4.000	7.200
	LD$<>	In conductive status			4.800	8.100	4.300	7.500	4.300	7.500	4.300	7.500
		In non-conductive status			4.700	8.400	4.200	7.800	4.200	7.800	4.200	7.800
	AND$<>	When not executed			0.060		0.0285		0.0285		0.0285	
		When executed		In conductive status	4.300	5.500	4.100	5.100	4.100	5.100	4.100	5.100
				In non-conductive status	4.500	5.900	4.400	5.400	4.400	5.400	4.400	5.400
	OR$<>	When not executed			0.060		0.0285		0.0285		0.0285	
		When executed		In conductive status	5.200	7.300	4.100	6.700	4.100	6.700	4.100	6.700
				In non-conductive status	5.100	7.200	4.100	6.700	4.100	6.700	4.100	6.700
	LD$>	In conductive status			4.800	7.200	4.300	6.700	4.300	6.700	4.300	6.700
		In non-conductive status			4.800	7.700	4.200	7.100	4.200	7.100	4.200	7.100
	AND$>	When not executed			0.060		0.0285		0.0285		0.0285	
		When executed		In conductive status	4.500	7.100	4.000	6.700	4.000	6.700	4.000	6.700
				In non-conductive status	4.600	7.600	4.300	7.000	4.300	7.000	4.300	7.000
	OR$>	When not executed			0.060		0.0285		0.0285		0.0285	
		When executed		In conductive status	5.100	6.800	4.300	6.200	4.300	6.200	4.300	6.200
				In non-conductive status	5.200	7.200	4.300	6.600	4.300	6.600	4.300	6.600

Category	Instruction	Condition (device)		Processing time (μs)							
				Q03UD(E)CPU		Q04UD(E)HCPU, Q06UD(E)HCPU		Q10UD(E)HCPU, Q13UD(E)HCPU, Q20UD(E)HCPU, Q26UD(E)HCPU		Q50UDEHCPU, Q100UDEHCPU	
				Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.
Basic instruction	LD$<=	In conductive status		5.000	6.300	4.400	5.700	4.400	5.700	4.400	5.700
		In non-conductive status		4.800	6.400	4.200	5.800	4.200	5.800	4.200	5.800
	AND$<=	When not executed		0.060		0.0285		0.0285		0.0285	
		When executed	In conductive status	4.600	7.600	4.100	7.200	4.100	7.200	4.100	7.200
			In non-conductive status	4.700	7.700	4.200	7.300	4.200	7.300	4.200	7.300
	OR$<=	When not executed		0.060		0.0285		0.0285		0.0285	
		When executed	In conductive status	4.700	7.700	4.400	7.200	4.400	7.200	4.400	7.200
			In non-conductive status	4.600	7.600	4.400	7.100	4.400	7.100	4.400	7.100
	LD$<	In conductive status		4.800	8.100	4.500	7.500	4.500	7.500	4.500	7.500
		In non-conductive status		5.000	8.300	4.500	7.900	4.500	7.900	4.500	7.900
	AND$<	When not executed		0.060		0.0285		0.0285		0.0285	
		When executed	In conductive status	4.500	7.100	4.000	6.600	4.000	6.600	4.000	6.600
			In non-conductive status	4.900	7.500	4.400	7.100	4.400	7.100	4.400	7.100
	OR$<	When not executed		0.060		0.0285		0.0285		0.0285	
		When executed	In conductive status	5.100	7.800	4.100	7.200	4.100	7.200	4.100	7.200
			In non-conductive status	5.000	8.100	4.100	7.600	4.100	7.600	4.100	7.600
	LD$>=	In conductive status		4.800	6.700	4.500	6.200	4.500	6.200	4.500	6.200
		In non-conductive status		5.000	6.700	4.400	6.300	4.400	6.300	4.400	6.300
	AND$>=	When not executed		0.060		0.0285		0.0285		0.0285	
		When executed	In conductive status	4.400	6.800	4.100	6.300	4.100	6.300	4.100	6.300
			In non-conductive status	4.500	7.000	4.200	6.600	4.200	6.600	4.200	6.600
	OR$>=	When not executed		0.060		0.0285		0.0285		0.0285	
		When executed	In conductive status	5.400	6.600	4.100	5.800	4.100	5.800	4.100	5.800
			In non-conductive status	5.300	6.300	4.100	5.700	4.100	5.700	4.100	5.700
	BKCMP= (S1) (S2) (D) n	n=1		8.200	10.700	7.500	10.000	7.500	10.000	7.500	10.000
		n=96		57.400	61.800	46.400	48.700	46.400	48.700	46.400	48.700
	BKCMP<> (S1) (S2) (D) n	n=1		8.200	10.700	7.500	10.000	7.500	10.000	7.500	10.000
		n=96		59.500	63.300	45.600	50.400	45.600	50.400	45.600	50.400

Category	Instruction	Condition (device)		Processing time (μs)							
				Q03UD(E)CPU		Q04UD(E)HCPU, Q06UD(E)HCPU		Q10UD(E)HCPU, Q13UD(E)HCPU, Q20UD(E)HCPU, Q26UD(E)HCPU		Q50UDEHCPU, Q100UDEHCPU	
				Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.
Basic instruction	BKCMP> (S1) (S2) (D) n	n=1		8.200	10.800	7.500	10.100	7.500	10.100	7.500	10.100
		n=96		59.500	63.400	47.700	50.500	47.700	50.500	47.700	50.500
	BKCMP<= (S1) (S2) (D) n	n=1		8.200	10.600	7.500	10.000	7.500	10.000	7.500	10.000
		n=96		57.400	61.700	46.400	49.000	46.400	49.000	46.400	49.000
	BKCMP< (S1) (S2) (D) n	n=1		8.300	10.600	7.500	10.000	7.500	10.000	7.500	10.000
		n=96		59.500	63.600	47.600	50.500	47.600	50.500	47.600	50.500
	BKCMP>= (S1) (S2) (D) n	n=1		8.200	10.900	7.500	10.000	7.500	10.000	7.500	10.000
		n=96		57.400	62.000	46.400	48.900	46.400	48.900	46.400	48.900
	DB* (S1) (S2) (D)	When executed		8.300	12.100	8.100	11.600	8.100	11.600	8.100	11.600
	DB/ (S1) (S2) (D)	When executed		6.100	9.100	5.800	8.800	5.800	8.800	5.800	8.800
	DBKCMP= (S1) (S2) (D) n	n=1		9.250	14.000	8.600	13.000	8.600	13.000	8.600	13.000
		n=96		60.700	67.500	47.900	52.800	47.900	52.800	47.900	52.800
	DBKCMP<> (S1) (S2) (D) n	n=1		9.250	14.000	8.600	13.000	8.600	13.000	8.600	13.000
		n=96		60.700	67.500	47.900	52.800	47.900	52.800	47.900	52.800
	DBKCMP> (S1) (S2) (D) n	n=1		9.250	14.000	8.600	13.000	8.600	13.000	8.600	13.000
		n=96		60.700	67.500	47.900	52.800	47.900	52.800	47.900	52.800
	DBKCMP<= (S1) (S2) (D) n	n=1		9.250	14.000	8.600	13.000	8.600	13.000	8.600	13.000
		n=96		60.700	67.500	47.900	52.800	47.900	52.800	47.900	52.800
	DBKCMP< (S1) (S2) (D) n	n=1		9.250	14.000	8.600	13.000	8.600	13.000	8.600	13.000
		n=96		60.700	67.500	47.900	52.800	47.900	52.800	47.900	52.800
	DBKCMP>= (S1) (S2) (D) n	n=1		9.250	14.000	8.600	13.000	8.600	13.000	8.600	13.000
		n=96		60.700	67.500	47.900	52.800	47.900	52.800	47.900	52.800
	DB+ (S) (D)	When executed		4.900	7.000	4.600	6.400	4.600	6.400	4.600	6.400
	DB+ (S1) (S2) (D)	When executed		5.200	7.300	4.800	6.700	4.800	6.700	4.800	6.700
	DB- (S) (D)	When executed		4.900	6.600	4.700	6.000	4.700	6.000	4.700	6.000
	DB- (S1) (S2) (D)	When executed		5.200	7.500	4.800	6.600	4.800	6.600	4.800	6.600
	ED+ (S) (D)	Double precision	(S)=0, (D)=0	4.800	8.000	4.300	7.200	4.300	7.200	4.300	7.200
			(S)=2^{1023}, (D)=2^{1023}	4.800	8.000	4.300	7.200	4.300	7.200	4.300	7.200
	ED+ (S1) (S2) (D)	Double precision	(S1)=0, (S2)=0	5.500	9.800	4.800	9.200	4.800	9.200	4.800	9.200
			(S1)=2^{1023}, (S2)=2^{1023}	5.500	9.800	4.800	9.200	4.800	9.200	4.800	9.200
	ED- (S) (D)	Double precision	(S)=0, (D)=0	5.000	8.200	4.400	7.500	4.400	7.500	4.400	7.500
			(S)=2^{1023}, (D)=2^{1023}	5.000	8.200	4.400	7.500	4.400	7.500	4.400	7.500
	ED- (S1) (S2) (D)	Double precision	(S1)=0, (S2)=0	4.400	8.100	3.800	7.500	3.800	7.500	3.800	7.500
			(S1)=2^{1023}, (S2)=2^{1023}	4.400	8.100	3.800	7.500	3.800	7.500	3.800	7.500
	ED* (S1) (S2) (D)	Double precision	(S1)=0, (S2)=0	5.800	9.500	5.100	8.800	5.100	8.800	5.100	8.800
			(S1)=2^{1023}, (S2)=2^{1023}	5.800	9.500	5.100	8.800	5.100	8.800	5.100	8.800
	ED/ (S1) (S2) (D)	Double precision	(S1)=2^{1023}, (S2)=2^{1023}	6.600	10.600	5.900	10.000	5.900	10.000	5.900	10.000
	BK+ (S1) (S2) (D) n	n=1		9.100	11.200	8.500	10.600	8.500	10.600	8.500	10.600
		n=96		60.700	62.900	44.600	47.000	44.600	47.000	44.600	47.000

Category	Instruction	Condition (device)		Processing time (μs)							
				Q03UD(E)CPU		Q04UD(E)HCPU, Q06UD(E)HCPU		Q10UD(E)HCPU, Q13UD(E)HCPU, Q20UD(E)HCPU, Q26UD(E)HCPU		Q50UDEHCPU, Q100UDEHCPU	
				Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.
Basic instruction	BK- (S1) (S2) (D) n	n=1		9.700	12.000	8.900	11.300	8.900	11.300	8.900	11.300
		n=96		61.300	63.600	45.600	47.900	45.600	47.900	45.600	47.900
	DBK+ (S1) (S2) (D) n	n=1		7.000	10.700	6.450	9.950	6.450	9.950	6.450	9.950
		n=96		59.400	63.100	43.700	47.500	43.700	47.500	43.700	47.500
	DBK- (S1) (S2) (D) n	n=1		7.000	10.700	6.450	9.950	6.450	9.950	6.450	9.950
		n=96		59.400	63.100	43.700	47.500	43.700	47.500	43.700	47.500
	$+ (S) (D)	—		8.800	14.600	8.100	13.900	8.100	13.900	8.100	13.900
	$+ (S1) (S2) (D)	—		7.300	11.100	6.500	10.300	6.500	10.300	6.500	10.300
	FLTD	Double precision	(S)=0	2.300	5.000	1.800	4.700	1.800	4.700	1.800	4.700
			(S)=7FFFH	2.500	5.200	2.200	4.800	2.200	4.800	2.200	4.800
	DFLTD	Double precision	(S)=0	2.400	5.200	2.000	4.900	2.000	4.900	2.000	4.900
			(S)=7FFFFFFFH	2.700	5.400	2.300	5.100	2.300	5.100	2.300	5.100
	INTD	Double precision	(S)=0	2.700	4.100	2.200	4.100	2.200	4.100	2.200	4.100
			(S)=32766.5	3.700	5.900	3.200	5.600	3.200	5.600	3.200	5.600
	DINTD	Double precision	(S)=0	2.600	3.900	2.200	3.400	2.200	3.400	2.200	3.400
			(S)=1234567890.3	3.400	5.600	3.000	5.100	3.000	5.100	3.000	5.100
	DBL	When executed		2.700	3.400	2.300	2.700	2.300	2.700	2.300	2.700
	WORD	When executed		2.900	4.300	2.600	3.600	2.600	3.600	2.600	3.600
	GRY	When executed		2.700	3.900	2.300	3.400	2.300	3.400	2.300	3.400
	DGRY	When executed		2.900	3.500	2.500	3.000	2.500	3.000	2.500	3.000
	GBIN	When executed		4.000	4.800	3.800	4.300	3.800	4.300	3.800	4.300
	DGBIN	When executed		5.500	6.100	5.000	5.900	5.000	5.900	5.000	5.900
	NEG	When executed		2.400	3.900	2.000	3.300	2.000	3.300	2.000	3.300
	DNEG	When executed		2.500	3.700	2.500	3.300	2.500	3.300	2.500	3.300
	ENEG	Floating point = 0		2.500	3.300	2.300	2.800	2.300	2.800	2.300	2.800
		Floating point = -1.0		2.700	4.500	2.500	3.900	2.500	3.900	2.500	3.900
	EDNEG	Floating point = 0		2.200	3.500	1.800	3.100	1.800	3.100	1.800	3.100
		Floating point = -1.0		2.400	3.500	1.900	3.000	1.900	3.000	1.900	3.000
	BKBCD (S) (D) n	n=1		6.600	8.900	5.900	8.200	5.900	8.200	5.900	8.200
		n=96		71.300	74.100	61.000	63.400	61.000	63.400	61.000	63.400
	BKBIN (S) (D) n	n=1		6.500	9.800	5.600	9.300	5.600	9.300	5.600	9.300
		n=96		56.300	59.500	49.200	52.500	49.200	52.500	49.200	52.500
	ECON	—		2.600	5.400	2.100	4.500	2.100	4.500	2.100	4.500
	EDCON	—		2.800	5.400	2.500	5.400	2.500	5.400	2.500	5.400
	EDMOV	—		2.300	5.500	1.700	5.000	1.700	5.000	1.700	5.000
	$MOV	Character string to be transferred = 0		4.000	6.300	3.400	5.600	3.400	5.600	3.400	5.600
		Character string to be transferred = 32		14.600	16.500	11.400	13.300	11.400	13.300	11.400	13.300
	BXCH (D1) (D2) n	n=1		6.200	7.900	5.500	7.300	5.500	7.300	5.500	7.300
		n=96		67.000	68.800	47.300	49.300	47.300	49.300	47.300	49.300
	SWAP	—		2.400	2.700	1.900	2.200	1.900	2.200	1.900	2.200
	GOEND	—		0.500		0.500		0.500		0.500	
	DI	—		1.800	2.200	1.500	1.800	1.500	1.800	1.500	1.800
	EI	—		3.100	3.800	3.000	3.300	3.000	3.300	3.000	3.300
	IMASK	—		9.800	13.300	7.200	10.500	7.200	10.500	7.200	10.500
	IRET	—		1.000		1.000		1.000		1.000	

Category	Instruction	Condition (device)	Processing time (μs)							
			Q03UD(E)CPU		Q04UD(E)HCPU, Q06UD(E)HCPU		Q10UD(E)HCPU, Q13UD(E)HCPU, Q20UD(E)HCPU, Q26UD(E)HCPU		Q50UDEHCPU, Q100UDEHCPU	
			Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.
Basic instruction	RFS X n	n=1	4.200	5.900	3.700	5.600	3.700	5.600	3.700	5.600
		n=96	11.400	13.800	10.700	12.400	10.700	12.400	10.700	12.400
	RFS Y n	n=1	3.800	4.800	3.400	4.800	3.400	4.800	3.400	4.800
		n=96	8.500	9.500	8.100	8.900	8.100	8.900	8.100	8.900
	UDCNT1	—	0.900	1.500	0.500	0.983	0.500	0.983	0.500	0.983
	UDCNT2	—	0.900	1.700	0.600	1.300	0.600	1.300	0.600	1.300
	TTMR	—	3.900	6.100	3.400	5.400	3.400	5.400	3.400	5.400
	STMR	—	6.800	13.500	5.800	12.500	5.800	12.500	5.800	12.500
	ROTC	—	9.000	10.500	8.000	9.400	8.000	9.400	8.000	9.400
	RAMP	—	5.900	8.800	5.200	8.400	5.200	8.400	5.200	8.400
	SPD	—	0.900	1.900	0.500	1.400	0.500	1.400	0.500	1.400
	PLSY	—	1.900	2.200	1.500	1.800	1.500	1.800	1.500	1.800
	PWM	—	1.200	1.600	0.900	1.200	0.900	1.200	0.900	1.200
	MTR	—	10.400	19.800	9.400	10.000	9.400	10.000	9.400	10.000

Category	Instruction	Condition (device)		Processing time (μs)							
				Q03UD(E)CPU		Q04UD(E)HCPU, Q06UD(E)HCPU		Q10UD(E)HCPU, Q13UD(E)HCPU, Q20UD(E)HCPU, Q26UD(E)HCPU		Q50UDEHCPU, Q100UDEHCPU	
				Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.
Application instruction	BKAND (S1) (S2) (D) n	n=1		9.000	11.700	8.300	11.000	8.300	11.000	8.300	11.000
		n=96		57.400	63.100	43.800	47.300	43.800	47.300	43.800	47.300
	BKOR (S1) (S2) (D) n	n=1		7.700	10.000	7.700	9.500	7.700	9.500	7.700	9.500
		n=96		57.400	61.900	44.300	45.800	44.300	45.800	44.300	45.800
	BKXOR (S1) (S2) (D) n	n=1		7.800	10.100	7.300	9.200	7.300	9.200	7.300	9.200
		n=96		57.300	61.500	43.800	45.800	43.800	45.800	43.800	45.800
	BKXNR (S1) (S2) (D) n	n=1		7.800	9.600	7.600	8.900	7.600	8.900	7.600	8.900
		n=96		57.400	61.400	43.900	45.300	43.900	45.300	43.900	45.300
	BSFR (D) n	n=1		3.700	5.400	3.200	4.800	3.200	4.800	3.200	4.800
		n=96		6.900	9.000	5.800	7.700	5.800	7.700	5.800	7.700
	BSFL (D) n	n=1		4.100	5.900	3.400	5.100	3.400	5.100	3.400	5.100
		n=96		7.100	9.100	6.000	7.900	6.000	7.900	6.000	7.900
	SFTBR (D) n1 n2	n1=16, n2=1		7.950	17.500	7.600	16.900	7.600	16.900	7.600	16.900
		n1=16, n2=15		7.950	17.500	7.550	16.900	7.550	16.900	7.550	16.900
	SFTBL (D) n1 n2	n1=16, n2=1		7.950	17.900	7.500	17.400	7.500	17.400	7.500	17.400
		n1=16, n2=15		7.900	17.800	7.500	17.300	7.500	17.300	7.500	17.300
	SFTWR (D) n1 n2	n1=16, n2=1		5.950	10.600	4.600	8.700	4.600	8.700	4.600	8.700
		n1=16, n2=15		5.900	10.600	4.600	8.700	4.600	8.700	4.600	8.700
	SFTWL (D) n1 n2	n1=16, n2=1		5.950	10.700	4.550	8.700	4.550	8.700	4.550	8.700
		n1=16, n2=15		5.950	10.700	4.600	8.800	4.600	8.800	4.600	8.800
	BSET (D) n	n=1		3.000	3.400	2.500	2.800	2.500	2.800	2.500	2.800
		n=15		3.000	3.500	2.500	2.800	2.500	2.800	2.500	2.800
	BRST (D) n	n=1		3.000	3.400	2.600	2.800	2.600	2.800	2.600	2.800
		n=15		3.000	3.400	2.500	2.800	2.500	2.800	2.500	2.800
	TEST	When executed		4.400	5.300	3.700	4.700	3.700	4.700	3.700	4.700
	DTEST	When executed		4.500	5.400	3.900	4.800	3.900	4.800	3.900	4.800
	BKRST (S) n	n=1		4.300	4.600	3.700	4.100	3.700	4.100	3.700	4.100
		n=96		6.000	6.800	5.100	6.000	5.100	6.000	5.100	6.000
	SER (S1) (S2) (D) n	n=1	All match	4.900	5.300	4.200	4.600	4.200	4.600	4.200	4.600
			None match	5.000	5.300	4.200	4.600	4.200	4.600	4.200	4.600
		n=96	All match	32.300	32.900	25.900	26.300	25.900	26.300	25.900	26.300
			None match	32.400	32.900	25.900	26.300	25.900	26.300	25.900	26.300
	DSER (S1) (S2) (D) n	n=1	All match	6.100	6.500	5.400	5.700	5.400	5.700	5.400	5.700
			None match	6.200	6.600	5.500	5.900	5.500	5.900	5.500	5.900
		n=96	All match	52.800	54.200	41.200	41.800	41.200	41.800	41.200	41.800
			None match	52.800	54.200	41.200	41.800	41.200	41.800	41.200	41.800
	DSUM (S) (D)	(S)=0		3.700	4.100	3.300	3.600	3.300	3.600	3.300	3.600
		(S)=FFFFFFFFH		3.800	4.100	3.200	3.700	3.200	3.700	3.200	3.700
	DECO (S) (D) n	n=2		6.000	7.500	5.300	6.900	5.300	6.900	5.300	6.900
		n=8		8.100	9.300	6.800	7.800	6.800	7.800	6.800	7.800
	ENCO (S) (D) n	n=2	M1=ON	5.300	5.700	4.700	5.100	4.700	5.100	4.700	5.100
			M4=ON	5.200	5.700	4.600	5.000	4.600	5.000	4.600	5.000
		n=8	M1=ON	10.400	11.400	9.000	10.000	9.000	10.000	9.000	10.000
			M256=ON	5.700	6.800	5.100	6.100	5.100	6.100	5.100	6.100
	DIS (S) (D) n	n=1		4.400	5.300	3.800	4.600	3.800	4.600	3.800	4.600
		n=4		4.800	5.700	4.000	5.000	4.000	5.000	4.000	5.000

Category	Instruction	Condition (device)	Processing time (μs)							
			Q03UD(E)CPU		Q04UD(E)HCPU, Q06UD(E)HCPU		Q10UD(E)HCPU, Q13UD(E)HCPU, Q20UD(E)HCPU, Q26UD(E)HCPU		Q50UDEHCPU, Q100UDEHCPU	
			Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.
Application instruction	UNI (S) (D) n	n=1	5.000	5.300	3.500	4.800	3.500	4.800	3.500	4.800
		n=4	5.600	6.000	4.000	5.100	4.000	5.100	4.000	5.100
	NDIS	When executed	11.000	13.100	11.000	13.200	11.000	13.200	11.000	13.200
	NUNI	When executed	10.600	12.700	7.300	13.200	7.300	13.200	7.300	13.200
	WTOB (S) (D) n	n=1	5.000	6.500	4.400	5.800	4.400	5.800	4.400	5.800
		n=96	36.000	38.400	28.200	29.300	28.200	29.300	28.200	29.300
	BTOW (S) (D) n	n=1	5.100	6.100	4.600	5.500	4.600	5.500	4.600	5.500
		n=96	29.900	32.000	22.800	23.800	22.800	23.800	22.800	23.800
	MAX (S) (D) n	n=1	4.300	6.900	4.000	6.100	4.000	6.100	4.000	6.100
		n=96	31.200	33.500	24.700	27.000	24.700	27.000	24.700	27.000
	MIN (S) (D) n	n=1	4.400	6.800	4.000	6.000	4.000	6.000	4.000	6.000
		n=96	30.300	34.800	26.500	28.300	26.500	28.300	26.500	28.300
	DMAX (S) (D) n	n=1	4.800	9.100	4.800	8.100	4.800	8.100	4.800	8.100
		n=96	56.400	62.200	47.100	49.600	47.100	49.600	47.100	49.600
	DMIN (S) (D) n	n=1	4.800	6.800	4.300	5.900	4.300	5.900	4.300	5.900
		n=96	55.400	60.200	45.400	47.400	45.400	47.400	45.400	47.400
	SORT (S1) n (S2) (D1) (D2)	n=1, (S2)=1	6.200	9.300	5.600	8.800	5.600	8.800	5.600	8.800
		n=96, (S2)=16	28.200	38.500	22.200	32.200	22.200	32.200	22.200	32.200
	DSORT (S1) n (S2) (D1) (D2)	n=1, (S2)=1	6.200	11.600	5.600	10.900	5.600	10.900	5.600	10.900
		n=96, (S2)=16	34.700	45.300	26.700	36.900	26.700	36.900	26.700	36.900
	WSUM (S) (D) n	n=1	4.800	6.200	4.200	5.500	4.200	5.500	4.200	5.500
		n=96	26.900	28.700	21.300	22.300	21.300	22.300	21.300	22.300
	DWSUM (S) (D) n	n=1	5.500	7.000	4.800	6.100	4.800	6.100	4.800	6.100
		n=96	53.000	56.300	42.700	44.000	42.700	44.000	42.700	44.000
	MEAN (S) (D) n	n=1	4.300	8.650	3.900	7.800	3.900	7.800	3.900	7.800
		n=96	16.000	21.400	12.900	18.000	12.900	18.000	12.900	18.000
	DMEAN (S) (D) n	n=1	5.700	10.600	5.300	9.950	5.300	9.950	5.300	9.950
		n=96	29.200	35.200	23.000	28.800	23.000	28.800	23.000	28.800
	NEXT	—	0.940	1.400	0.770	1.200	0.770	1.200	0.770	1.200
	BREAK	—	10.400	5.500	9.100	5.000	9.100	5.000	9.100	5.000
	RET	Return to original program	2.000	3.000	1.600	2.600	1.600	2.600	1.600	2.600
		Return to other program	2.300	3.700	2.000	3.100	2.000	3.100	2.000	3.100
	FCALL Pn	Internal file pointer	3.100	4.400	2.700	3.600	2.700	3.600	2.700	3.600
		Common pointer	4.000	5.700	3.600	5.100	3.600	5.100	3.600	5.100
	FCALL Pn (S1) to (S5)	—	19.300	21.500	16.500	18.600	16.500	18.600	16.500	18.600
	ECALL * Pn *: Program name	—	70.300	82.300	65.900	77.600	65.900	77.600	65.900	77.600
	ECALL * Pn (S1) to (S5) *: Program name	—	101.000	114.000	91.800	105.000	91.800	105.000	91.800	105.000
	EFCALL * Pn *: Program name	—	70.700	82.800	66.200	78.100	66.200	78.100	66.200	78.100
	EFCALL * Pn (S1) to (S5) *: Program name	—	86.500	107.000	78.800	91.600	78.800	91.600	78.800	91.600

Category	Instruction	Condition (device)	Processing time (μs)							
			Q03UD(E)CPU		Q04UD(E)HCPU, Q06UD(E)HCPU		Q10UD(E)HCPU, Q13UD(E)HCPU, Q20UD(E)HCPU, Q26UD(E)HCPU		Q50UDEHCPU, Q100UDEHCPU	
			Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.
Application instruction	XCALL	—	3.800	5.700	3.700	5.200	3.700	5.200	3.700	5.200
	COM CCOM	When selecting I/O refresh only	12.800	29.100	12.400	28.600	12.400	28.600	12.400	28.600
		When selecting CC-Link refresh only (master station side)	16.000	39.500	15.500	39.100	15.500	39.100	15.500	39.100
		When selecting CC-Link refresh only (local station side)	16.100	39.500	15.500	39.100	15.500	39.100	15.500	39.100
		• When selecting MELSECNET/H refresh only (control station side) • When selecting CC-Link IE Controller Network refresh only (control station side)	34.700	70.400	34.400	69.800	34.400	69.800	34.400	69.800
		• When selecting MELSECNET/H refresh only (normal station side) • When selecting CC-Link IE Controller Network refresh only (normal station side)	34.700	70.400	34.400	69.800	34.400	69.800	34.400	69.800
		When selecting CC-Link IE Field Network refresh only (master station side)	17.000	38.800	16.600	38.000	16.600	38.000	16.600	38.000
		When selecting CC-Link IE Field Network refresh only (local station side)	17.000	38.800	16.600	38.000	16.600	38.000	16.600	38.000
		When selecting intelli auto refresh only	12.800	33.200	12.800	33.200	12.800	33.200	12.800	33.200
		When selecting I/O outside the group only (Input only)	7.900	21.100	7.700	20.700	7.700	20.700	7.700	20.700
		When selecting I/O outside the group only (Output only)	16.900	44.800	16.500	44.200	16.500	44.200	16.500	44.200
		When selecting I/O outside the group only (Both I/O)	22.600	52.600	22.400	52.600	22.400	52.600	22.400	52.600
		When selecting refresh of multiple CPU high speed transmission area only	13.000	33.800	12.700	33.200	12.700	33.200	12.700	33.200
		When selecting communication with external devices only	7.250	18.800	7.100	18.500	7.100	18.500	7.100	18.500
	FIFW	Number of data points = 0	3.700	5.300	3.200	4.600	3.200	4.600	3.200	4.600
		Number of data points = 96	3.800	4.400	3.300	3.800	3.300	3.800	3.300	3.800
	FIFR	Number of data points = 1	4.300	5.000	3.800	4.400	3.800	4.400	3.800	4.400
		Number of data points = 96	33.500	35.500	24.800	25.700	24.800	25.700	24.800	25.700
	FPOP	Number of data points = 1	4.300	5.900	3.800	5.300	3.800	5.300	3.800	5.300
		Number of data points = 96	4.300	5.900	3.700	5.400	3.700	5.400	3.700	5.400
	FINS	Number of data points = 0	4.800	5.900	3.700	5.300	3.700	5.300	3.700	5.300
		Number of data points = 96	4.300	5.900	3.700	5.300	3.700	5.300	3.700	5.300
	FDEL	Number of data points = 1	4.900	6.500	4.200	5.800	4.200	5.800	4.200	5.800
		Number of data points = 96	34.200	35.900	25.400	25.900	25.400	25.900	25.400	25.900
	FROM n1 n2 (D) n3	n3=1	10.800	24.100	10.700	23.600	10.700	23.600	10.700	23.600
		n3=1000	392.600	413.300	390.900	410.200	390.900	410.200	390.900	410.200
	DFRO n1 n2 (D) n3	n3=1	13.600	27.700	12.600	26.700	12.600	26.700	12.600	26.700
		n3=500	392.600	413.300	390.900	410.200	390.900	410.200	390.900	410.200
	TO n1 n2 (S) n3	n3=1	10.200	21.900	9.600	21.300	9.600	21.300	9.600	21.300
		n3=1000	373.700	394.100	372.500	390.800	372.500	390.800	372.500	390.800
	DTO n1 n2 (S) n3	n3=1	13.000	26.700	12.000	25.700	12.000	25.700	12.000	25.700
		n3=500	373.700	394.100	372.500	390.800	372.500	390.800	372.500	390.800

Category	Instruction	Condition (device)		Processing time (μs)							
				Q03UD(E)CPU		Q04UD(E)HCPU, Q06UD(E)HCPU		Q10UD(E)HCPU, Q13UD(E)HCPU, Q20UD(E)HCPU, Q26UD(E)HCPU		Q50UDEHCPU, Q100UDEHCPU	
				Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.
Application instruction	LEDR	No display → no display		2.400	2.600	1.900	2.000	1.900	2.000	1.900	2.000
		LED instruction execution → no display		28.100	39.400	24.400	35.800	24.400	35.800	24.400	35.800
	BINDA (S1) (D)	(S)=1		4.900	6.500	4.300	5.600	4.300	5.600	4.300	5.600
		(S)=-32768		7.200	8.700	6.500	8.000	6.500	8.000	6.500	8.000
	DBINDA (S) (D)	(S)=1		5.700	7.100	4.900	6.300	4.900	6.300	4.900	6.300
		(S)=-2147483648		10.400	12.000	9.600	11.000	9.600	11.000	9.600	11.000
	BINHA (S) (D)	(S)=1		4.400	5.900	3.800	5.200	3.800	5.200	3.800	5.200
		(S)=FFFFH		4.400	5.800	3.700	5.200	3.700	5.200	3.700	5.200
	DBINHA (S) (D)	(S)=1		5.200	6.700	4.600	6.000	4.600	6.000	4.600	6.000
		(S)=FFFFFFFFH		5.100	6.500	4.600	6.000	4.600	6.000	4.600	6.000
	BCDDA (S) (D)	(S)=1		4.300	5.800	3.600	5.000	3.600	5.000	3.600	5.000
		(S)=9999		4.700	6.100	4.100	5.400	4.100	5.400	4.100	5.400
	DBCDDA (S) (D)	(S)=1		4.800	6.300	4.000	5.500	4.000	5.500	4.000	5.500
		(S)=99999999		5.600	7.100	4.900	6.300	4.900	6.300	4.900	6.300
	DABIN (S) (D)	(S)=1		6.500	8.500	5.800	7.800	5.800	7.800	5.800	7.800
		(S)=-32768		6.300	8.300	5.600	7.700	5.600	7.700	5.600	7.700
	DDABIN (S) (D)	(S)=1		9.400	11.500	8.500	10.500	8.500	10.500	8.500	10.500
		(S)=-2147483648		9.100	11.200	8.100	10.200	8.100	10.200	8.100	10.200
	HABIN (S) (D)	(S)=1		4.900	7.100	4.400	6.400	4.400	6.400	4.400	6.400
		(S)=FFFFH		5.100	7.300	4.600	6.500	4.600	6.500	4.600	6.500
	DHABIN (S) (D)	(S)=1		6.000	8.100	5.300	7.300	5.300	7.300	5.300	7.300
		(S)=FFFFFFFFH		6.300	8.500	5.600	7.700	5.600	7.700	5.600	7.700
	DABCD (S) (D)	(S)=1		5.000	7.100	4.400	6.300	4.400	6.300	4.400	6.300
		(S)=9999		5.000	7.100	4.300	6.300	4.300	6.300	4.300	6.300
	DDABCD (S) (D)	(S)=1		6.200	8.300	5.500	7.400	5.500	7.400	5.500	7.400
		(S)=99999999		6.200	8.300	5.500	7.500	5.500	7.500	5.500	7.500
	COMRD	—		51.600	52.400	50.900	51.200	50.900	51.200	50.900	51.200
	LEN	1 character		4.100	6.200	3.600	5.500	3.600	5.500	3.600	5.500
		96 characters		19.800	22.200	16.800	18.700	16.800	18.700	16.800	18.700
	STR	—		6.900	11.100	6.600	10.400	6.600	10.400	6.600	10.400
	DSTR	—		10.200	12.500	9.600	11.500	9.600	11.500	9.600	11.500
	VAL	—		9.800	14.200	8.900	13.000	8.900	13.000	8.900	13.000
	DVAL	—		14.000	18.700	12.700	16.800	12.700	16.800	12.700	16.800
	ESTR	—		18.700	24.100	17.900	23.100	17.900	23.100	17.900	23.100
	LDTM=	Comparison of specified clock	In conductive status	7.300	11.500	6.700	10.800	6.700	10.800	6.700	10.800
			In non-conductive status	7.300	11.500	6.700	10.800	6.700	10.800	6.700	10.800
		Comparison of current clock	In conductive status	5.800	9.900	5.400	9.500	5.400	9.500	5.400	9.500
			In non-conductive status	5.800	9.900	5.400	9.500	5.400	9.500	5.400	9.500

A

Category	Instruction	Condition (device)		Processing time (μs)							
				Q03UD(E)CPU		Q04UD(E)HCPU, Q06UD(E)HCPU		Q10UD(E)HCPU, Q13UD(E)HCPU, Q20UD(E)HCPU, Q26UD(E)HCPU		Q50UDEHCPU, Q100UDEHCPU	
				Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.
Application instruction	ANDTM=	When not executed		0.008		0.038		0.038		0.038	
		Comparison of specified clock	In conductive status	7.000	11.500	6.300	10.800	6.300	10.800	6.300	10.800
			In non-conductive status	7.000	11.500	6.300	10.800	6.300	10.800	6.300	10.800
		Comparison of current clock	In conductive status	5.500	9.900	5.100	9.500	5.100	9.500	5.100	9.500
			In non-conductive status	5.500	9.900	5.100	9.500	5.100	9.500	5.100	9.500
	ORTM=	When not executed		0.008		0.038		0.038		0.038	
		Comparison of specified clock	In conductive status	7.300	11.500	6.600	10.800	6.600	10.800	6.600	10.800
			In non-conductive status	7.300	11.500	6.600	10.800	6.600	10.800	6.600	10.800
		Comparison of current clock	In conductive status	5.900	9.900	5.300	9.500	5.300	9.500	5.300	9.500
			In non-conductive status	5.900	9.900	5.300	9.500	5.300	9.500	5.300	9.500
	LDTM<>	Comparison of specified clock	In conductive status	7.300	11.500	6.700	10.800	6.700	10.800	6.700	10.800
			In non-conductive status	7.300	11.500	6.700	10.800	6.700	10.800	6.700	10.800
		Comparison of current clock	In conductive status	5.800	9.900	5.400	9.500	5.400	9.500	5.400	9.500
			In non-conductive status	5.800	9.900	5.400	9.500	5.400	9.500	5.400	9.500
	ANDTM<>	When not executed		0.008		0.038		0.038		0.038	
		Comparison of specified clock	In conductive status	7.000	11.500	6.300	10.800	6.300	10.800	6.300	10.800
			In non-conductive status	7.000	11.500	6.300	10.800	6.300	10.800	6.300	10.800
		Comparison of current clock	In conductive status	5.500	9.900	5.100	9.500	5.100	9.500	5.100	9.500
			In non-conductive status	5.500	9.900	5.100	9.500	5.100	9.500	5.100	9.500

Category	Instruction	Condition (device)		Processing time (μs)							
				Q03UD(E)CPU		Q04UD(E)HCPU, Q06UD(E)HCPU		Q10UD(E)HCPU, Q13UD(E)HCPU, Q20UD(E)HCPU, Q26UD(E)HCPU		Q50UDEHCPU, Q100UDEHCPU	
				Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.
Application instruction	ORTM<>	When not executed		0.008		0.038		0.038		0.038	
		Comparison of specified clock	In conductive status	7.300	11.500	6.600	10.800	6.600	10.800	6.600	10.800
			In non-conductive status	7.300	11.500	6.600	10.800	6.600	10.800	6.600	10.800
		Comparison of current clock	In conductive status	5.900	9.900	5.300	9.500	5.300	9.500	5.300	9.500
			In non-conductive status	5.900	9.900	5.300	9.500	5.300	9.500	5.300	9.500
	LDTM>	Comparison of specified clock	In conductive status	7.300	11.500	6.700	10.800	6.700	10.800	6.700	10.800
			In non-conductive status	7.300	11.500	6.700	10.800	6.700	10.800	6.700	10.800
		Comparison of current clock	In conductive status	5.800	9.900	5.400	9.500	5.400	9.500	5.400	9.500
			In non-conductive status	5.800	9.900	5.400	9.500	5.400	9.500	5.400	9.500
	ANDTM>	When not executed		0.008		0.038		0.038		0.038	
		Comparison of specified clock	In conductive status	7.000	11.500	6.300	10.800	6.300	10.800	6.300	10.800
			In non-conductive status	7.000	11.500	6.300	10.800	6.300	10.800	6.300	10.800
		Comparison of current clock	In conductive status	5.500	9.900	5.100	9.500	5.100	9.500	5.100	9.500
			In non-conductive status	5.500	9.900	5.100	9.500	5.100	9.500	5.100	9.500
	ORTM>	When not executed		0.008		0.038		0.038		0.038	
		Comparison of specified clock	In conductive status	7.300	11.500	6.600	10.800	6.600	10.800	6.600	10.800
			In non-conductive status	7.300	11.500	6.600	10.800	6.600	10.800	6.600	10.800
		Comparison of current clock	In conductive status	5.900	9.900	5.300	9.500	5.300	9.500	5.300	9.500
			In non-conductive status	5.900	9.900	5.300	9.500	5.300	9.500	5.300	9.500

Category	Instruction	Condition (device)		Processing time (μs)							
				Q03UD(E)CPU		Q04UD(E)HCPU, Q06UD(E)HCPU		Q10UD(E)HCPU, Q13UD(E)HCPU, Q20UD(E)HCPU, Q26UD(E)HCPU		Q50UDEHCPU, Q100UDEHCPU	
				Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.
Application instruction	LDTM<=	Comparison of specified clock	In conductive status	7.300	11.500	6.700	10.800	6.700	10.800	6.700	10.800
			In non-conductive status	7.300	11.500	6.700	10.800	6.700	10.800	6.700	10.800
		Comparison of current clock	In conductive status	5.800	9.900	5.400	9.500	5.400	9.500	5.400	9.500
			In non-conductive status	5.800	9.900	5.400	9.500	5.400	9.500	5.400	9.500
	ANDTM<=	When not executed		0.008		0.038		0.038		0.038	
		Comparison of specified clock	In conductive status	7.000	11.500	6.300	10.800	6.300	10.800	6.300	10.800
			In non-conductive status	7.000	11.500	6.300	10.800	6.300	10.800	6.300	10.800
		Comparison of current clock	In conductive status	5.500	9.900	5.100	9.500	5.100	9.500	5.100	9.500
			In non-conductive status	5.500	9.900	5.100	9.500	5.100	9.500	5.100	9.500
	ORTM<=	When not executed		0.008		0.038		0.038		0.038	
		Comparison of specified clock	In conductive status	7.300	11.500	6.600	10.800	6.600	10.800	6.600	10.800
			In non-conductive status	7.300	11.500	6.600	10.800	6.600	10.800	6.600	10.800
		Comparison of current clock	In conductive status	5.900	9.900	5.300	9.500	5.300	9.500	5.300	9.500
			In non-conductive status	5.900	9.900	5.300	9.500	5.300	9.500	5.300	9.500
	LDTM<	Comparison of specified clock	In conductive status	7.300	11.500	6.700	10.800	6.700	10.800	6.700	10.800
			In non-conductive status	7.300	11.500	6.700	10.800	6.700	10.800	6.700	10.800
		Comparison of current clock	In conductive status	5.800	9.900	5.400	9.500	5.400	9.500	5.400	9.500
			In non-conductive status	5.800	9.900	5.400	9.500	5.400	9.500	5.400	9.500

Category	Instruction	Condition (device)		Processing time (μs)							
				Q03UD(E)CPU		Q04UD(E)HCPU, Q06UD(E)HCPU		Q10UD(E)HCPU, Q13UD(E)HCPU, Q20UD(E)HCPU, Q26UD(E)HCPU		Q50UDEHCPU, Q100UDEHCPU	
				Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.
Application instruction	ANDTM<	When not executed		0.008		0.038		0.038		0.038	
		Comparison of specified clock	In conductive status	7.000	11.500	6.300	10.800	6.300	10.800	6.300	10.800
			In non-conductive status	7.000	11.500	6.300	10.800	6.300	10.800	6.300	10.800
		Comparison of current clock	In conductive status	5.500	9.900	5.100	9.500	5.100	9.500	5.100	9.500
			In non-conductive status	5.500	9.900	5.100	9.500	5.100	9.500	5.100	9.500
	ORTM<	When not executed		0.008		0.038		0.038		0.038	
		Comparison of specified clock	In conductive status	7.300	11.500	6.600	10.800	6.600	10.800	6.600	10.800
			In non-conductive status	7.300	11.500	6.600	10.800	6.600	10.800	6.600	10.800
		Comparison of current clock	In conductive status	5.900	9.900	5.300	9.500	5.300	9.500	5.300	9.500
			In non-conductive status	5.900	9.900	5.300	9.500	5.300	9.500	5.300	9.500
	LDTM>=	Comparison of specified clock	In conductive status	7.300	11.500	6.700	10.800	6.700	10.800	6.700	10.800
			In non-conductive status	7.300	11.500	6.700	10.800	6.700	10.800	6.700	10.800
		Comparison of current clock	In conductive status	5.800	9.900	5.400	9.500	5.400	9.500	5.400	9.500
			In non-conductive status	5.800	9.900	5.400	9.500	5.400	9.500	5.400	9.500
	ANDTM>=	When not executed		0.008		0.038		0.038		0.038	
		Comparison of specified clock	In conductive status	7.000	11.500	6.300	10.800	6.300	10.800	6.300	10.800
			In non-conductive status	7.000	11.500	6.300	10.800	6.300	10.800	6.300	10.800
		Comparison of current clock	In conductive status	5.500	9.900	5.100	9.500	5.100	9.500	5.100	9.500
			In non-conductive status	5.500	9.900	5.100	9.500	5.100	9.500	5.100	9.500

Category	Instruction	Condition (device)		Processing time (μs)							
				Q03UD(E)CPU		Q04UD(E)HCPU, Q06UD(E)HCPU		Q10UD(E)HCPU, Q13UD(E)HCPU, Q20UD(E)HCPU, Q26UD(E)HCPU		Q50UDEHCPU, Q100UDEHCPU	
				Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.
Application instruction	ORTM>=	When not executed		0.008		0.038		0.038		0.038	
		Comparison of specified clock	In conductive status	7.300	11.500	6.600	10.800	6.600	10.800	6.600	10.800
			In non-conductive status	7.300	11.500	6.600	10.800	6.600	10.800	6.600	10.800
		Comparison of current clock	In conductive status	5.900	9.900	5.300	9.500	5.300	9.500	5.300	9.500
			In non-conductive status	5.900	9.900	5.300	9.500	5.300	9.500	5.300	9.500
	LDDT=	Comparison of specified date	In conductive status	7.400	11.400	6.800	10.900	6.800	10.900	6.800	10.900
			In non-conductive status	7.400	11.600	6.800	10.900	6.800	10.900	6.800	10.900
		Comparison of current date	In conductive status	5.900	10.000	5.500	9.700	5.500	9.700	5.500	9.700
			In non-conductive status	5.900	10.100	5.500	9.700	5.500	9.700	5.500	9.700
	ANDDT=	When not executed		0.008		0.038		0.038		0.038	
		Comparison of specified date	In conductive status	7.200	11.400	6.500	10.700	6.500	10.700	6.500	10.700
			In non-conductive status	7.200	11.400	6.500	10.700	6.500	10.700	6.500	10.700
		Comparison of current date	In conductive status	5.700	9.900	5.300	9.300	5.300	9.300	5.300	9.300
			In non-conductive status	5.700	9.900	5.300	9.300	5.300	9.300	5.300	9.300
	ORDT=	When not executed		0.008		0.038		0.038		0.038	
		Comparison of specified date	In conductive status	7.400	11.500	6.700	10.800	6.700	10.800	6.700	10.800
			In non-conductive status	7.400	11.500	6.700	10.800	6.700	10.800	6.700	10.800
		Comparison of current date	In conductive status	5.900	10.000	5.400	9.600	5.400	9.600	5.400	9.600
			In non-conductive status	5.900	10.000	5.400	9.600	5.400	9.600	5.400	9.600

Category	Instruction	Condition (device)		Processing time (μs)							
				Q03UD(E)CPU		Q04UD(E)HCPU, Q06UD(E)HCPU		Q10UD(E)HCPU, Q13UD(E)HCPU, Q20UD(E)HCPU, Q26UD(E)HCPU		Q50UDEHCPU, Q100UDEHCPU	
				Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.
Application instruction	LDDT<>	Comparison of specified date	In conductive status	7.400	11.400	6.800	10.900	6.800	10.900	6.800	10.900
			In non-conductive status	7.400	11.600	6.800	10.900	6.800	10.900	6.800	10.900
		Comparison of current date	In conductive status	5.900	10.000	5.500	9.700	5.500	9.700	5.500	9.700
			In non-conductive status	5.900	10.100	5.500	9.700	5.500	9.700	5.500	9.700
	ANDDT<>	When not executed		0.008		0.038		0.038		0.038	
		Comparison of specified date	In conductive status	7.200	11.400	6.500	10.700	6.500	10.700	6.500	10.700
			In non-conductive status	7.200	11.400	6.500	10.700	6.500	10.700	6.500	10.700
		Comparison of current date	In conductive status	5.700	9.900	5.300	9.300	5.300	9.300	5.300	9.300
			In non-conductive status	5.700	9.900	5.300	9.300	5.300	9.300	5.300	9.300
	ORDT<>	When not executed		0.008		0.038		0.038		0.038	
		Comparison of specified date	In conductive status	7.400	11.500	6.700	10.800	6.700	10.800	6.700	10.800
			In non-conductive status	7.400	11.500	6.700	10.800	6.700	10.800	6.700	10.800
		Comparison of current date	In conductive status	5.900	10.000	5.400	9.600	5.400	9.600	5.400	9.600
			In non-conductive status	5.900	10.000	5.400	9.600	5.400	9.600	5.400	9.600
	LDDT>	Comparison of specified date	In conductive status	7.400	11.400	6.800	10.900	6.800	10.900	6.800	10.900
			In non-conductive status	7.400	11.600	6.800	10.900	6.800	10.900	6.800	10.900
		Comparison of current date	In conductive status	5.900	10.000	5.500	9.700	5.500	9.700	5.500	9.700
			In non-conductive status	5.900	10.100	5.500	9.700	5.500	9.700	5.500	9.700

Category	Instruction	Condition (device)		Processing time (μs)							
				Q03UD(E)CPU		Q04UD(E)HCPU, Q06UD(E)HCPU		Q10UD(E)HCPU, Q13UD(E)HCPU, Q20UD(E)HCPU, Q26UD(E)HCPU		Q50UDEHCPU, Q100UDEHCPU	
				Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.
Application instruction	ANDDT>	When not executed		0.008		0.038		0.038		0.038	
		Comparison of specified date	In conductive status	7.200	11.400	6.500	10.700	6.500	10.700	6.500	10.700
			In non-conductive status	7.200	11.400	6.500	10.700	6.500	10.700	6.500	10.700
		Comparison of current date	In conductive status	5.700	9.900	5.300	9.300	5.300	9.300	5.300	9.300
			In non-conductive status	5.700	9.900	5.300	9.300	5.300	9.300	5.300	9.300
	ORDT>	When not executed		0.008		0.038		0.038		0.038	
		Comparison of specified date	In conductive status	7.400	11.500	6.700	10.800	6.700	10.800	6.700	10.800
			In non-conductive status	7.400	11.500	6.700	10.800	6.700	10.800	6.700	10.800
		Comparison of current date	In conductive status	5.900	10.000	5.400	9.600	5.400	9.600	5.400	9.600
			In non-conductive status	5.900	10.000	5.400	9.600	5.400	9.600	5.400	9.600
	LDDT<=	Comparison of specified date	In conductive status	7.400	11.400	6.800	10.900	6.800	10.900	6.800	10.900
			In non-conductive status	7.400	11.600	6.800	10.900	6.800	10.900	6.800	10.900
		Comparison of current date	In conductive status	5.900	10.000	5.500	9.700	5.500	9.700	5.500	9.700
			In non-conductive status	5.900	10.100	5.500	9.700	5.500	9.700	5.500	9.700
	ANDDT<=	When not executed		0.008		0.038		0.038		0.038	
		Comparison of specified date	In conductive status	7.200	11.400	6.500	10.700	6.500	10.700	6.500	10.700
			In non-conductive status	7.200	11.400	6.500	10.700	6.500	10.700	6.500	10.700
		Comparison of current date	In conductive status	5.700	9.900	5.300	9.300	5.300	9.300	5.300	9.300
			In non-conductive status	5.700	9.900	5.300	9.300	5.300	9.300	5.300	9.300

Category	Instruction	Condition (device)		Processing time (μs)							
				Q03UD(E)CPU		Q04UD(E)HCPU, Q06UD(E)HCPU		Q10UD(E)HCPU, Q13UD(E)HCPU, Q20UD(E)HCPU, Q26UD(E)HCPU		Q50UDEHCPU, Q100UDEHCPU	
				Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.
Application instruction	ORDT<=	When not executed		0.008		0.038		0.038		0.038	
		Comparison of specified date	In conductive status	7.400	11.500	6.700	10.800	6.700	10.800	6.700	10.800
			In non-conductive status	7.400	11.500	6.700	10.800	6.700	10.800	6.700	10.800
		Comparison of current date	In conductive status	5.900	10.000	5.400	9.600	5.400	9.600	5.400	9.600
			In non-conductive status	5.900	10.000	5.400	9.600	5.400	9.600	5.400	9.600
	LDDT<	Comparison of specified date	In conductive status	7.400	11.400	6.800	10.900	6.800	10.900	6.800	10.900
			In non-conductive status	7.400	11.600	6.800	10.900	6.800	10.900	6.800	10.900
		Comparison of current date	In conductive status	5.900	10.000	5.500	9.700	5.500	9.700	5.500	9.700
			In non-conductive status	5.900	10.100	5.500	9.700	5.500	9.700	5.500	9.700
	ANDDT<	When not executed		0.008		0.038		0.038		0.038	
		Comparison of specified date	In conductive status	7.200	11.400	6.500	10.700	6.500	10.700	6.500	10.700
			In non-conductive status	7.200	11.400	6.500	10.700	6.500	10.700	6.500	10.700
		Comparison of current date	In conductive status	5.700	9.900	5.300	9.300	5.300	9.300	5.300	9.300
			In non-conductive status	5.700	9.900	5.300	9.300	5.300	9.300	5.300	9.300
	ORDT<	When not executed		0.008		0.038		0.038		0.038	
		Comparison of specified date	In conductive status	7.400	11.500	6.700	10.800	6.700	10.800	6.700	10.800
			In non-conductive status	7.400	11.500	6.700	10.800	6.700	10.800	6.700	10.800
		Comparison of current date	In conductive status	5.900	10.000	5.400	9.600	5.400	9.600	5.400	9.600
			In non-conductive status	5.900	10.000	5.400	9.600	5.400	9.600	5.400	9.600

Category	Instruction	Condition (device)		Processing time (μs)							
				Q03UD(E)CPU		Q04UD(E)HCPU, Q06UD(E)HCPU		Q10UD(E)HCPU, Q13UD(E)HCPU, Q20UD(E)HCPU, Q26UD(E)HCPU		Q50UDEHCPU, Q100UDEHCPU	
				Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.
Application instruction	LDDT>=	Comparison of specified date	In conductive status	7.400	11.400	6.800	10.900	6.800	10.900	6.800	10.900
			In non-conductive status	7.400	11.600	6.800	10.900	6.800	10.900	6.800	10.900
		Comparison of current date	In conductive status	5.900	10.000	5.500	9.700	5.500	9.700	5.500	9.700
			In non-conductive status	5.900	10.100	5.500	9.700	5.500	9.700	5.500	9.700
	ANDDT>=	When not executed		0.008		0.038		0.038		0.038	
		Comparison of specified date	In conductive status	7.200	11.400	6.500	10.700	6.500	10.700	6.500	10.700
			In non-conductive status	7.200	11.400	6.500	10.700	6.500	10.700	6.500	10.700
		Comparison of current date	In conductive status	5.700	9.900	5.300	9.300	5.300	9.300	5.300	9.300
			In non-conductive status	5.700	9.900	5.300	9.300	5.300	9.300	5.300	9.300
	ORDT>=	When not executed		0.008		0.038		0.038		0.038	
		Comparison of specified date	In conductive status	7.400	11.500	6.700	10.800	6.700	10.800	6.700	10.800
			In non-conductive status	7.400	11.500	6.700	10.800	6.700	10.800	6.700	10.800
		Comparison of current date	In conductive status	5.900	10.000	5.400	9.600	5.400	9.600	5.400	9.600
			In non-conductive status	5.900	10.000	5.400	9.600	5.400	9.600	5.400	9.600
	EVAL	Decimal point format all 2-digit specification		23.300	30.400	22.800	29.000	22.800	29.000	22.800	29.000
		Exponent format all 6-digit specification		23.300	30.500	22.500	29.000	22.500	29.000	22.500	29.000
	ASC (S) (D) n	n=1		5.600	9.000	5.400	8.300	5.400	8.300	5.400	8.300
		n=96		28.700	32.100	25.200	28.400	25.200	28.400	25.200	28.400
	HEX (S) (D) n	n=1		6.000	9.700	5.400	9.000	5.400	9.000	5.400	9.000
		n=96		35.600	39.800	31.300	35.000	31.300	35.000	31.300	35.000
	RIGHT (S) (D) n	n=1		7.600	9.400	6.600	7.300	6.600	7.300	6.600	7.300
		n=96		36.300	40.000	29.200	31.600	29.200	31.600	29.200	31.600
	LEFT (S) (D) n	n=1		6.500	8.900	5.900	8.200	5.900	8.200	5.900	8.200
		n=96		36.200	39.700	29.200	31.500	29.200	31.500	29.200	31.500
	MIDR	—		9.500	12.100	8.100	10.300	8.100	10.300	8.100	10.300
	MIDW	—		10.300	12.000	8.800	10.200	8.800	10.200	8.800	10.200

Category	Instruction	Condition (device)		Processing time (μs)							
				Q03UD(E)CPU		Q04UD(E)HCPU, Q06UD(E)HCPU		Q10UD(E)HCPU, Q13UD(E)HCPU, Q20UD(E)HCPU, Q26UD(E)HCPU		Q50UDEHCPU, Q100UDEHCPU	
				Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.
Application instruction	INSTR	No match		19.300	21.800	16.600	18.400	16.600	18.400	16.600	18.400
		Match	Head	10.300	12.800	9.100	10.900	9.100	10.900	9.100	10.900
			End	51.100	54.200	42.700	44.900	42.700	44.900	42.700	44.900
	EMOD	—		10.300	11.800	9.600	11.000	9.600	11.000	9.600	11.000
	EREXP	—		19.300	21.000	18.800	20.100	18.800	20.100	18.800	20.100
	STRINS (S) (D) n	(S)=128/(D)=40/n=1		41.100	54.200	35.300	47.600	35.300	47.600	35.300	47.600
		(S)=128/(D)=40/n=48		56.700	81.400	48.600	61.700	48.600	61.700	48.600	61.700
	STRDEL (S) (D) n	(S)=128/(D)=40/n=1		39.000	49.500	34.800	44.600	34.800	44.600	34.800	44.600
		(S)=128/(D)=40/n=48		36.000	45.200	29.200	38.100	29.200	38.100	29.200	38.100
	SIN	Single precision		4.500	6.200	4.100	5.700	4.100	5.700	4.100	5.700
	COS	Single precision		4.300	6.000	4.000	5.600	4.000	5.600	4.000	5.600
	TAN	Single precision		5.100	7.200	5.100	6.700	5.100	6.700	5.100	6.700
	ASIN	Single precision		6.100	8.900	5.900	8.500	5.900	8.500	5.900	8.500
	ACOS	Single precision		6.800	9.300	6.700	8.900	6.700	8.900	6.700	8.900
	ATAN	Single precision		4.000	6.500	3.900	6.000	3.900	6.000	3.900	6.000
	SIND	Double precision		8.800	14.300	8.500	13.800	8.500	13.800	8.500	13.800
	COSD	Double precision		9.300	15.100	8.800	14.600	8.800	14.600	8.800	14.600
	TAND	Double precision		11.200	16.900	10.800	16.500	10.800	16.500	10.800	16.500
	ASIND	Double precision		12.000	17.100	11.600	16.600	11.600	16.600	11.600	16.600
	ACOSD	Double precision		11.700	16.500	11.200	16.200	11.200	16.200	11.200	16.200
	ATAND	Double precision		9.500	14.200	9.100	13.800	9.100	13.800	9.100	13.800
	RAD	Single precision		2.500	4.800	2.100	4.300	2.100	4.300	2.100	4.300
	RADD	Double precision		4.000	9.600	3.600	9.200	3.600	9.200	3.600	9.200
	DEG	Single precision		2.500	4.700	2.200	4.400	2.200	4.400	2.200	4.400
	DEGD	Double precision		4.300	9.000	3.800	9.000	3.800	9.000	3.800	9.000
	SQR	Single precision		3.000	4.600	2.600	4.300	2.600	4.300	2.600	4.300
	SQRD	Double precision		5.600	11.500	5.200	11.000	5.200	11.000	5.200	11.000
	EXP (S) (D)	Single precision	(S)=-10	4.000	6.100	3.800	5.500	3.800	5.500	3.800	5.500
			(S)=1	4.000	6.100	3.800	5.600	3.800	5.600	3.800	5.600
	EXPD (S) (D)	Double precision	(S)=-10	8.700	13.900	8.200	13.500	8.200	13.500	8.200	13.500
			(S)=1	8.400	13.600	8.000	13.200	8.000	13.200	8.000	13.200
	LOG (S) (D)	Single precision	(S)=1	4.100	6.900	3.800	6.400	3.800	6.400	3.800	6.400
			(S)=10	5.600	8.200	5.200	7.700	5.200	7.700	5.200	7.700
	LOGD (S) (D)	Double precision	(S)=1	8.100	13.000	7.700	12.500	7.700	12.500	7.700	12.500
			(S)=10	9.700	14.800	9.200	14.300	9.200	14.300	9.200	14.300
	RND	—		1.200	2.300	0.800	1.800	0.800	1.800	0.800	1.800
	SRND	—		1.400	2.400	1.100	2.000	1.100	2.000	1.100	2.000
	BSQR (S) (D)	(S)=0		1.800	3.300	1.600	2.800	1.600	2.800	1.600	2.800
		(S)=9999		5.100	8.800	5.100	8.000	5.100	8.000	5.100	8.000
	BDSQR (S) (D)	(S)=0		1.900	3.400	1.500	3.000	1.500	3.000	1.500	3.000
		(S)=99999999		7.500	10.200	7.500	9.900	7.500	9.900	7.500	9.900
	BSIN	—		8.600	15.100	8.100	14.500	8.100	14.500	8.100	14.500
	BCOS	—		7.800	14.400	7.800	13.700	7.800	13.700	7.800	13.700
	BTAN	—		9.000	13.800	9.000	13.300	9.000	13.300	9.000	13.300
	BASIN	—		10.600	13.400	10.100	12.800	10.100	12.800	10.100	12.800
	BACOS	—		11.600	14.400	11.100	14.100	11.100	14.100	11.100	14.100
	BATAN	—		9.800	11.700	9.100	10.900	9.100	10.900	9.100	10.900

A

Category	Instruction	Condition (device)		Processing time (μs)							
				Q03UD(E)CPU		Q04UD(E)HCPU, Q06UD(E)HCPU		Q10UD(E)HCPU, Q13UD(E)HCPU, Q20UD(E)HCPU, Q26UD(E)HCPU		Q50UDEHCPU, Q100UDEHCPU	
				Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.
Application instruction	POW (S1) (S2) (D)	Single precision	(S1)=12.3E+5 (S2)=3.45E+0	8.750	11.400	8.400	10.900	8.400	10.900	8.400	10.900
	POWD (S1) (S2) (D)	Double precision	(S1)=12.3E+5 (S2)=3.45E+0	18.600	27.200	18.200	26.500	18.200	26.500	18.200	26.500
	LOG10	—		5.900	8.550	5.700	8.050	5.700	8.050	5.700	8.050
	LOG10D	—		11.500	19.400	11.100	18.600	11.100	18.600	11.100	18.600
	LIMIT	—		2.800	3.100	2.400	2.700	2.400	2.700	2.400	2.700
	DLIMIT	—		3.200	3.500	2.800	3.000	2.800	3.000	2.800	3.000
	BAND	—		3.000	4.300	2.700	3.800	2.700	3.800	2.700	3.800
	DBAND	—		3.600	5.100	3.300	4.600	3.300	4.600	3.300	4.600
	ZONE	—		3.000	4.700	2.600	4.300	2.600	4.300	2.600	4.300
	DZONE	—		3.400	5.000	3.000	4.600	3.000	4.600	3.000	4.600
	SCL (S1) (S2) (D)	SM750= ON	Point No.1 < (S1) < Point No.2	13.200	23.600	12.300	22.500	12.300	22.500	12.300	22.500
			Point No.9 < (S1) < Point No.10	13.300	23.600	12.600	22.700	12.600	22.700	12.600	22.700
		SM750= OFF	Point No.1 < (S1) < Point No.2	12.000	23.100	11.400	22.200	11.400	22.200	11.400	22.200
			Point No.9 < (S1) < Point No.10	14.100	25.300	12.800	23.900	12.800	23.900	12.800	23.900
	DSCL (S1) (S2) (D)	SM750= ON	Point No.1 < (S1) < Point No.2	12.800	23.800	11.900	23.000	11.900	23.000	11.900	23.000
			Point No.9 < (S1) < Point No.10	12.900	23.900	12.100	23.000	12.100	23.000	12.100	23.000
		SM750= OFF	Point No.1 < (S1) < Point No.2	11.500	22.400	10.900	21.500	10.900	21.500	10.900	21.500
			Point No.9 < (S1) < Point No.10	13.800	24.900	12.700	23.600	12.700	23.600	12.700	23.600
	SCL2 (S1) (S2) (D)	SM750= ON	Point No.1 < (S1) < Point No.2	12.700	24.200	11.900	23.300	11.900	23.300	11.900	23.300
			Point No.9 < (S1) < Point No.10	12.900	24.600	12.100	23.300	12.100	23.300	12.100	23.300
		SM750= OFF	Point No.1 < (S1) < Point No.2	12.300	23.400	11.500	22.600	11.500	22.600	11.500	22.600
			Point No.9 < (S1) < Point No.10	13.700	25.000	12.600	23.900	12.600	23.900	12.600	23.900
	DSCL2 (S1) (S2) (D)	SM750= ON	Point No.1 < (S1) < Point No.2	12.600	23.800	11.800	22.900	11.800	22.900	11.800	22.900
			Point No.9 < (S1) < Point No.10	13.000	23.900	12.200	22.800	12.200	22.800	12.200	22.800
		SM750= OFF	Point No.1 < (S1) < Point No.2	11.500	22.400	11.000	21.400	11.000	21.400	11.000	21.400
			Point No.9 < (S1) < Point No.10	13.900	24.900	12.800	23.600	12.800	23.600	12.800	23.600
	RSET	Standard RAM		3.000	6.300	2.700	5.900	2.700	5.900	2.700	5.900
		SRAM card		3.000	6.400	2.600	5.800	2.600	5.800	2.600	5.800
	DATE-	No digit increase		5.800	8.500	4.600	7.000	4.600	7.000	4.600	7.000
		Digit increase		5.700	7.400	4.600	6.500	4.600	6.500	4.600	6.500
	SECOND	—		2.600	3.900	2.200	3.400	2.200	3.400	2.200	3.400
	HOUR	—		2.900	4.800	2.400	4.300	2.400	4.300	2.400	4.300
	QDRSET	SRAM card to standard RAM		120.000	134.000	115.000	134.000	115.000	134.000	115.000	134.000
		Standard RAM to SRAM card		533.000	560.000	520.000	553.000	520.000	553.000	520.000	553.000

Category	Instruction	Condition (device)		Processing time (μs)							
				Q03UD(E)CPU		Q04UD(E)HCPU, Q06UD(E)HCPU		Q10UD(E)HCPU, Q13UD(E)HCPU, Q20UD(E)HCPU, Q26UD(E)HCPU		Q50UDEHCPU, Q100UDEHCPU	
				Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.
Application instruction	QCDSET	SRAM card to standard ROM		306.000	346.000	305.000	346.000	305.000	346.000	305.000	346.000
		Standard ROM to SRAM card		311.000	342.000	300.000	334.000	300.000	334.000	300.000	334.000
	DATERD	—		3.200	5.000	2.500	4.200	2.500	4.200	2.500	4.200
	DATEWR	—		4.900	9.700	4.100	8.900	4.100	8.900	4.100	8.900
	DATE+	No digit increase		5.100	8.000	4.700	6.600	4.700	6.600	4.700	6.600
		Digit increase		5.700	8.000	4.600	6.500	4.600	6.500	4.600	6.500
	S.DATERD	—		5.600	8.000	4.800	7.100	4.800	7.100	4.800	7.100
	S.DATE+	No digit increase		9.100	11.700	7.400	10.000	7.400	11.000	7.400	11.000
		Digit increase		8.900	11.800	7.400	10.000	7.400	11.000	7.400	11.000
	S.DATE-	No digit increase		9.000	12.000	7.400	10.300	7.400	10.300	7.400	10.300
		Digit increase		9.000	12.200	7.500	10.200	7.500	10.200	7.500	10.200
	PSTOP	—		61.400	84.500	56.600	79.800	56.600	79.800	56.600	79.800
	POFF	—		61.800	84.500	57.200	79.800	57.200	79.800	57.200	79.800
	PSCAN	—		64.900	84.700	60.100	79.900	60.100	79.900	60.100	79.900
	WDT	—		1.300	2.700	1.100	2.400	1.100	2.400	1.100	2.400
	DUTY	—		4.900	10.100	4.800	9.600	4.800	9.600	4.800	9.600
	TIMCHK	—		3.800	5.300	3.500	4.700	3.500	4.700	3.500	4.700
	ZRRDB	File register of standard RAM		2.400	2.600	1.800	2.100	1.800	2.100	1.800	2.100
		File register of SRAM card		2.500	2.800	2.000	2.300	2.000	2.300	2.000	2.300
	ZRWRB	File register of standard RAM		3.000	3.300	2.400	2.700	2.400	2.700	2.400	2.700
		File register of SRAM card		3.200	3.600	2.600	3.000	2.600	3.000	2.600	3.000
	ADRSET	—		2.600	3.100	2.100	2.600	2.100	2.600	2.100	2.600
	ZPUSH	—		6.900	9.200	5.800	7.500	5.800	7.500	5.800	7.500
	ZPOP	—		7.500	8.800	5.800	6.400	5.800	6.400	5.800	6.400
	UNIRD n1 (D) n2	n2=1		4.000	8.400	3.700	8.000	3.700	8.000	3.700	8.000
		n2=16		12.500	17.000	12.200	16.600	12.200	16.600	12.200	16.600
	TYPERD	—		29.800	53.000	29.500	52.300	29.500	52.300	29.500	52.300
	TRACE	Start		46.600	48.300	43.800	44.700	43.800	44.700	43.800	44.700
	TRACER	—		3.300	6.800	2.600	6.000	2.600	6.000	2.600	6.000
	RBMOV (S) (D) n	When standard RAM is used	1 point	11.300	16.800	9.200	15.100	9.200	15.100	9.200	15.100
			1000 points	120.700	127.100	61.000	68.600	61.000	68.600	61.000	68.600
		When SRAM card is used	1 point	11.200	16.700	9.400	15.600	9.400	15.600	9.400	15.600
			1000 points	180.700	187.100	165.000	172.600	165.000	172.600	165.000	172.600
	SP.FWRITE	—		4.100	55.600	3.700	53.700	3.700	53.700	3.700	53.700
	SP.FREAD	—		4.600	54.600	4.000	53.000	4.000	53.000	4.000	53.000
	SP.DEVST	—		4.500	36.500	4.000	34.500	4.000	34.500	4.000	34.500
	S.DEVLD	—		11.000	17.800	10.000	17.000	10.000	17.000	10.000	17.000

A

Category	Instruction	Condition (device)	Processing time (μs)							
			Q03UD(E)CPU		Q04UD(E)HCPU, Q06UD(E)HCPU		Q10UD(E)HCPU, Q13UD(E)HCPU, Q20UD(E)HCPU, Q26UD(E)HCPU		Q50UDEHCPU, Q100UDEHCPU	
			Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.
Data link instruction	S.ZCOM	When mounting CC-Link module (master station side)	19.600	26.500	19.300	26.000	19.300	26.000	19.300	26.000
		When mounting CC-Link module (local station side)	19.600	26.500	19.100	26.200	19.100	26.200	19.100	26.200
		• When selecting MELSECNET/H refresh only (control station side) • When selecting CC-Link IE Controller Network refresh only (control station side)	53.500	73.500	53.000	72.700	53.000	72.700	53.000	72.700
		• When selecting MELSECNET/H refresh only (normal station side) • When selecting CC-Link IE Controller Network refresh only (normal station side)	29.800	61.100	29.800	60.800	29.800	60.800	29.800	60.800
		When selecting CC-Link IE Field Network refresh only (master station side)	31.500	60.000	31.000	58.000	31.000	58.000	31.000	58.000
		When selecting CC-Link IE Field Network refresh only (local station side)	31.500	60.000	31.000	58.000	31.000	58.000	31.000	58.000
	S.RTREAD	—	8.200	20.500	7.400	19.000	7.400	19.000	7.400	19.000
	S.RTWRITE	—	8.700	21.500	8.300	19.800	8.300	19.800	8.300	19.800
	S.REFDVWRB	When the refresh device as the write target exists at Transfer 1	85.300	99.400	84.800	97.500	84.800	97.500	84.800	97.500
		When the refresh device as the write target exists at Transfer 256	515.700	528.500	501.600	518.200	501.600	518.200	501.600	518.200
	S.REFDVWRW	When the refresh device as the write target exists at Transfer 1	85.000	99.300	84.600	97.400	84.600	97.400	84.600	97.400
		When the refresh device as the write target exists at Transfer 256	527.000	539.500	517.300	529.000	517.300	529.000	517.300	529.000
	S.REFDVRDB	When the refresh device as the read target exists at Transfer 1	83.100	99.100	82.000	97.200	82.000	97.200	82.000	97.200
		When the refresh device as the read target exists at Transfer 256	514.600	527.700	499.800	517.400	499.800	517.400	499.800	517.400
	S.REFDVRDW	When the refresh device as the read target exists at Transfer 1	83.100	99.100	82.000	97.200	82.000	97.200	82.000	97.200
		When the refresh device as the read target exists at Transfer 256	526.100	539.400	517.400	528.900	517.400	528.900	517.400	528.900

Category	Instruction	Condition (device)		Processing time (μs)							
				Q03UD(E)CPU		Q04UD(E)HCPU, Q06UD(E)HCPU		Q10UD(E)HCPU, Q13UD(E)HCPU, Q20UD(E)HCPU, Q26UD(E)HCPU		Q50UDEHCPU, Q100UDEHCPU	
				Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.
Multiple CPU Dedicated Instruction	S.TO n1 n2 n3 n4 (D)	Writing to host CPU shared memory	n4=1	34.700	34.900	33.500	34.400	33.500	34.400	33.500	34.400
			n4=320	85.900	87.600	75.200	75.500	75.200	75.500	75.200	75.500
	TO n1 n2 (S) n3	Writing to host CPU shared memory	n3=1	4.700	23.800	4.500	23.300	4.500	23.300	4.500	23.300
			n3=320	57.500	76.200	47.100	64.500	47.100	64.500	47.100	64.500
	DTO n1 n2 (S) n3	Writing to host CPU shared memory	n3=1	5.300	23.800	5.800	23.300	5.800	23.300	5.800	23.300
			n3=320	111.300	128.400	91.500	108.500	91.500	108.500	91.500	108.500
	FROM n1 n2 (D) n3	Reading from host CPU shared memory	n3=1	5.000	23.800	4.300	23.300	4.300	23.300	4.300	23.300
			n3=320	51.400	65.600	44.400	60.700	44.400	60.700	44.400	60.700
		Reading from other CPU shared memory	n3=1	11.600	17.700	10.600	13.900	10.600	13.900	10.600	13.900
			n3=320	142.000	160.000	142.000	149.000	142.000	149.000	142.000	149.000
			n3=1000	431.000	463.000	422.000	448.000	422.000	448.000	422.000	448.000
	DFRO n1 n2 (D) n3	Reading from host CPU shared memory	n3=1	6.600	23.800	5.600	23.300	5.600	23.300	5.600	23.300
			n3=320	96.400	113.200	83.600	100.800	83.600	100.800	83.600	100.800
		Reading from other CPU shared memory	n3=1	12.900	20.800	12.200	17.100	12.200	17.100	12.200	17.100
			n3=320	277.000	299.000	274.000	291.000	274.000	291.000	274.000	291.000
			n3=1000	838.000	860.000	835.000	857.000	835.000	857.000	835.000	857.000
Multiple CPU high-speed transmission dedicated instruction	D.DDWR n (S1) (S2) (D1) (D2)	Writing devices to another CPU	n=1	82.000	133.000	80.000	130.000	80.000	130.000	80.000	130.000
			n=16	91.000	142.000	89.000	139.000	89.000	139.000	89.000	139.000
			n=96*1	136.000	189.000	132.000	182.000	132.000	182.000	132.000	182.000
	DP.DDWR n (S1) (S2) (D1) (D2)		n=1	82.000	133.000	80.000	130.000	80.000	130.000	80.000	130.000
			n=16	91.000	142.000	89.000	139.000	89.000	139.000	89.000	139.000
			n=96*1	136.000	189.000	132.000	182.000	132.000	182.000	132.000	182.000
	D.DDRD n (S1) (S2) (D1) (D2)	Reading devices from another CPU	n=1	82.000	133.000	80.000	130.000	80.000	130.000	80.000	130.000
			n=16	82.000	133.000	80.000	130.000	80.000	130.000	80.000	130.000
			n=96*1	82.000	133.000	80.000	130.000	80.000	130.000	80.000	130.000
	DP.DDRD n (S1) (S2) (D1) (D2)		n=1	82.000	133.000	80.000	130.000	80.000	130.000	80.000	130.000
			n=16	82.000	133.000	80.000	130.000	80.000	130.000	80.000	130.000
			n=96*1	82.000	133.000	80.000	130.000	80.000	130.000	80.000	130.000

*1 Can be used only for the Q03UDCPU, Q04UDHCPU, Q06UDHCPU, Q13UDHCPU, and Q26UDHCPU whose first five digits of serial number is "10012" or later.

Point

For the instructions for which a rising edge instruction (□P) is not described, the processing time is the same as an ON execution instruction.
(Example) WORDP instruction, TOP instruction etc.

A

• When using Q03UDVCPU, Q04UDVCPU, Q04UDPVCPU, Q06UDVCPU, Q06UDPVCPU, Q13UDVCPU, Q13UDPVCPU, Q26UDVCPU, and Q26UDPVCPU

Category	Instruction	Condition (device)	Processing time (μs)					
			Q03UDVCPU		Q04UDVCPU, Q04UDPVCPU		Q06UDVCPU, Q06UDPVCPU, Q13UDVCPU, Q13UDPVCPU, Q26UDVCPU, Q26UDPVCPU	
			Min.	Max.	Min.	Max.	Min.	Max.
Sequence instruction	ANB ORB MPS MRD MPP	—	0.0019	0.0078	0.0019	0.0078	0.0019	0.0078
	INV	When not executed	0.0019	0.0078	0.0019	0.0078	0.0019	0.0078
		When executed	0.0019	0.0078	0.0019	0.0078	0.0019	0.0078
	MEP MEF	When not executed	0.0039	0.0078	0.0039	0.0078	0.0039	0.0078
		When executed	0.0039	0.0078	0.0039	0.0078	0.0039	0.0078
	EGP EGF	When not executed	0.0039	0.0078	0.0039	0.0078	0.0039	0.0078
		When executed	0.0039	0.0078	0.0039	0.0078	0.0039	0.0078
	PLS	—	0.0078	0.015	0.0078	0.015	0.0078	0.015
	PLF	—	0.0078	0.015	0.0078	0.015	0.0078	0.015
	FF	When not executed	0.0078	0.015	0.0078	0.015	0.0078	0.015
		When executed	0.0078	0.015	0.0078	0.015	0.0078	0.015
	DELTA	When not executed	0.012		0.012		0.012	
		When executed	1.600	5.600	1.600	5.600	1.600	5.600
	SFT	When not executed	0.012		0.012		0.012	
		When executed	1.100	5.000	1.100	5.000	1.100	5.000
	MC	—	0.0078	0.015	0.0078	0.015	0.0078	0.015
	MCR	—	0.0078	0.015	0.0078	0.015	0.0078	0.015
	FEND END	Error check performed	60.000	60.000	60.000	60.000	60.000	60.000
		No error check performed	60.000	60.000	60.000	60.000	60.000	60.000
	NOP NOPLF PAGE	—	0.0019	0.0078	0.0019	0.0078	0.0019	0.0078

Category	Instruction	Condition (device)			Processing time (μs)					
					Q03UDVCPU		Q04UDVCPU, Q04UDPVCPU		Q06UDVCPU, Q06UDPVCPU, Q13UDVCPU, Q13UDPVCPU, Q26UDVCPU, Q26UDPVCPU	
					Min.	Max.	Min.	Max.	Min.	Max.
Basic instruction	LDE=	Single precision	In conductive status		0.0098	0.023	0.0098	0.023	0.0098	0.023
			In non-conductive status		0.0098	0.023	0.0098	0.023	0.0098	0.023
	ANDE=	Single precision	When not executed		0.012		0.012		0.012	
			When executed	In conductive status	0.0098	0.023	0.0098	0.023	0.0098	0.023
				In non-conductive status	0.0098	0.023	0.0098	0.023	0.0098	0.023
	ORE=	Single precision	When not executed		0.0078	0.023	0.0078	0.023	0.0078	0.023
			When executed	In conductive status	0.0098	0.023	0.0098	0.023	0.0098	0.023
				In non-conductive status	0.0098	0.023	0.0098	0.023	0.0098	0.023
	LDE<>	Single precision	In conductive status		0.0098	0.023	0.0098	0.023	0.0098	0.023
			In non-conductive status		0.0098	0.023	0.0098	0.023	0.0098	0.023
	ANDE<>	Single precision	When not executed		0.0078	0.023	0.0078	0.023	0.0078	0.023
			When executed	In conductive status	0.0098	0.023	0.0098	0.023	0.0098	0.023
				In non-conductive status	0.0098	0.023	0.0098	0.023	0.0098	0.023
	ORE<>	Single precision	When not executed		0.0078	0.023	0.0078	0.023	0.0078	0.023
			When executed	In conductive status	0.0098	0.023	0.0098	0.023	0.0098	0.023
				In non-conductive status	0.0098	0.023	0.0098	0.023	0.0098	0.023

Category	Instruction	Condition (device)			Processing time (μs)					
					Q03UDVCPU		Q04UDVCPU, Q04UDPVCPU		Q06UDVCPU, Q06UDPVCPU, Q13UDVCPU, Q13UDPVCPU, Q26UDVCPU, Q26UDPVCPU	
					Min.	Max.	Min.	Max.	Min.	Max.
Basic instruction	LDE>	Single precision	In conductive status		0.0098	0.023	0.0098	0.023	0.0098	0.023
			In non-conductive status		0.0098	0.023	0.0098	0.023	0.0098	0.023
	ANDE>	Single precision	When not executed		0.0078	0.023	0.0078	0.023	0.0078	0.023
			When executed	In conductive status	0.0098	0.023	0.0098	0.023	0.0098	0.023
				In non-conductive status	0.0098	0.023	0.0098	0.023	0.0098	0.023
	ORE>	Single precision	When not executed		0.0078	0.023	0.0078	0.023	0.0078	0.023
			When executed	In conductive status	0.0098	0.023	0.0098	0.023	0.0098	0.023
				In non-conductive status	0.0098	0.023	0.0098	0.023	0.0098	0.023
	LDE<=	Single precision	In conductive status		0.0098	0.023	0.0098	0.023	0.0098	0.023
			In non-conductive status		0.0098	0.023	0.0098	0.023	0.0098	0.023
	ANDE<=	Single precision	When not executed		0.0078	0.023	0.0078	0.023	0.0078	0.023
			When executed	In conductive status	0.0098	0.023	0.0098	0.023	0.0098	0.023
				In non-conductive status	0.0098	0.023	0.0098	0.023	0.0098	0.023
	ORE<=	Single precision	When not executed		0.0078	0.023	0.0078	0.023	0.0078	0.023
			When executed	In conductive status	0.0098	0.023	0.0098	0.023	0.0098	0.023
				In non-conductive status	0.0098	0.023	0.0098	0.023	0.0098	0.023
	LDE<	Single precision	In conductive status		0.0098	0.023	0.0098	0.023	0.0098	0.023
			In non-conductive status		0.0098	0.023	0.0098	0.023	0.0098	0.023
	ANDE<	Single precision	When not executed		0.0078	0.023	0.0078	0.023	0.0078	0.023
			When executed	In conductive status	0.0098	0.023	0.0098	0.023	0.0098	0.023
				In non-conductive status	0.0098	0.023	0.0098	0.023	0.0098	0.023
	ORE<	Single precision	When not executed		0.0078	0.023	0.0078	0.023	0.0078	0.023
			When executed	In conductive status	0.0098	0.023	0.0098	0.023	0.0098	0.023
				In non-conductive status	0.0098	0.023	0.0098	0.023	0.0098	0.023
	LDE>=	Single precision	In conductive status		0.0098	0.023	0.0098	0.023	0.0098	0.023
			In non-conductive status		0.0098	0.023	0.0098	0.023	0.0098	0.023
	ANDE>=	Single precision	When not executed		0.0078	0.023	0.0078	0.023	0.0078	0.023
			When executed	In conductive status	0.0098	0.023	0.0098	0.023	0.0098	0.023
				In non-conductive status	0.0098	0.023	0.0098	0.023	0.0098	0.023
	ORE>=	Single precision	When not executed		0.0078	0.023	0.0078	0.023	0.0078	0.023
			When executed	In conductive status	0.0098	0.023	0.0098	0.023	0.0098	0.023
				In non-conductive status	0.0098	0.023	0.0098	0.023	0.0098	0.023
	LDED=	Double precision	In conductive status		1.800	6.900	1.800	6.900	1.800	6.900
			In non-conductive status		1.800	7.100	1.800	7.100	1.800	7.100

Category	Instruction	Condition (device)			Processing time (μs)					
					Q03UDVCPU		Q04UDVCPU, Q04UDPVCPU		Q06UDVCPU, Q06UDPVCPU, Q13UDVCPU, Q13UDPVCPU, Q26UDVCPU, Q26UDPVCPU	
					Min.	Max.	Min.	Max.	Min.	Max.
Basic instruction	ANDED=	Double precision	When not executed		0.018		0.018		0.018	
			When executed	In conductive status	1.700	6.600	1.700	6.600	1.700	6.600
				In non-conductive status	1.700	6.600	1.700	6.600	1.700	6.600
	ORED=	Double precision	When not executed		0.018		0.018		0.018	
			When executed	In conductive status	1.700	7.100	1.700	7.100	1.700	7.100
				In non-conductive status	1.800	7.100	1.800	7.100	1.800	7.100
	LDED<>	Double precision	In conductive status		1.800	7.100	1.800	7.100	1.800	7.100
			In non-conductive status		1.800	6.900	1.800	6.900	1.800	6.900
	ANDED<>	Double precision	When not executed		0.018		0.018		0.018	
			When executed	In conductive status	1.700	6.200	1.700	6.200	1.700	6.200
				In non-conductive status	1.700	6.600	1.700	6.600	1.700	6.600
	ORED<>	Double precision	When not executed		0.018		0.018		0.018	
			When executed	In conductive status	1.800	6.500	1.800	6.500	1.800	6.500
				In non-conductive status	1.800	6.700	1.800	6.700	1.800	6.700
	LDED>	Double precision	In conductive status		1.800	7.300	1.800	7.300	1.800	7.300
			In non-conductive status		1.800	7.200	1.800	7.200	1.800	7.200
	ANDED>	Double precision	When not executed		0.018		0.018		0.018	
			When executed	In conductive status	1.700	6.700	1.700	6.700	1.700	6.700
				In non-conductive status	1.700	6.800	1.700	6.800	1.700	6.800
	ORED>	Double precision	When not executed		0.018		0.018		0.018	
			When executed	In conductive status	1.700	6.700	1.700	6.700	1.700	6.700
				In non-conductive status	1.800	7.200	1.800	7.200	1.800	7.200
	LDED<=	Double precision	In conductive status		1.800	7.100	1.800	7.100	1.800	7.100
			In non-conductive status		1.700	7.100	1.700	7.100	1.700	7.100
	ANDED<=	Double precision	When not executed		0.018		0.018		0.018	
			When executed	In conductive status	1.700	6.700	1.700	6.700	1.700	6.700
				In non-conductive status	1.800	6.500	1.800	6.500	1.800	6.500
	ORED<=	Double precision	When not executed		0.018		0.018		0.018	
			When executed	In conductive status	1.700	6.700	1.700	6.700	1.700	6.700
				In non-conductive status	1.800	6.500	1.800	6.500	1.800	6.500
	LDED<	Double precision	In conductive status		1.800	7.100	1.800	7.100	1.800	7.100
			In non-conductive status		1.800	7.100	1.800	7.100	1.800	7.100

A

Category	Instruction	Condition (device)			Processing time (μs)					
					Q03UDVCPU		Q04UDVCPU, Q04UDPVCPU		Q06UDVCPU, Q06UDPVCPU, Q13UDVCPU, Q13UDPVCPU, Q26UDVCPU, Q26UDPVCPU	
					Min.	Max.	Min.	Max.	Min.	Max.
Basic instruction	ANDED<	Double precision	When not executed		0.018		0.018		0.018	
			When executed	In conductive status	1.700	6.700	1.700	6.700	1.700	6.700
				In non-conductive status	1.700	6.400	1.700	6.400	1.700	6.400
	ORED<	Double precision	When not executed		0.018		0.018		0.018	
			When executed	In conductive status	1.700	7.100	1.700	7.100	1.700	7.100
				In non-conductive status	1.700	7.100	1.700	7.100	1.700	7.100
	LDED>=	Double precision	In conductive status		1.800	7.100	1.800	7.100	1.800	7.100
			In non-conductive status		1.800	7.100	1.800	7.100	1.800	7.100
	ANDED>=	Double precision	When not executed		0.018		0.018		0.018	
			When executed	In conductive status	1.700	6.400	1.700	6.400	1.700	6.400
				In non-conductive status	1.700	6.400	1.700	6.400	1.700	6.400
	ORED>=	Double precision	When not executed		0.018		0.018		0.018	
			When executed	In conductive status	1.700	7.100	1.700	7.100	1.700	7.100
				In non-conductive status	1.700	7.100	1.700	7.100	1.700	7.100
	LD$=	In conductive status			1.400	3.800	1.400	3.800	1.400	3.800
		In non-conductive status			1.400	3.800	1.300	3.800	1.300	3.800
	AND$=	When not executed			0.018		0.018		0.018	
		When executed	In conductive status		1.300	3.700	1.300	3.800	1.300	3.800
			In non-conductive status		1.400	3.900	1.400	3.900	1.400	3.900
	OR$=	When not executed			0.018		0.018		0.018	
		When executed	In conductive status		1.300	3.900	1.400	3.900	1.400	3.900
			In non-conductive status		1.300	3.800	1.300	3.800	1.300	3.800
	LD$<>	In conductive status			1.400	3.800	1.400	3.800	1.400	3.800
		In non-conductive status			1.400	3.800	1.400	3.800	1.400	3.800
	AND$<>	When not executed			0.018		0.018		0.018	
		When executed	In conductive status		1.300	3.700	1.300	3.700	1.300	3.700
			In non-conductive status		1.400	3.900	1.400	3.900	1.400	3.900
	OR$<>	When not executed			0.018		0.018		0.018	
		When executed	In conductive status		1.300	3.900	1.300	3.900	1.300	3.900
			In non-conductive status		1.300	3.800	1.300	3.800	1.300	3.800
	LD$>	In conductive status			1.400	3.800	1.400	3.800	1.400	3.800
		In non-conductive status			1.300	3.800	1.300	3.800	1.300	3.800

Category	Instruction	Condition (device)		Processing time (μs)					
				Q03UDVCPU		Q04UDVCPU, Q04UDPVCPU		Q06UDVCPU, Q06UDPVCPU, Q13UDVCPU, Q13UDPVCPU, Q26UDVCPU, Q26UDPVCPU	
				Min.	Max.	Min.	Max.	Min.	Max.
Basic instruction	AND$>	When not executed		0.018		0.018		0.018	
		When executed	In conductive status	1.300	3.800	1.300	3.800	1.300	3.800
			In non-conductive status	1.300	3.900	1.300	3.900	1.300	3.900
	OR$>	When not executed		0.018		0.018		0.018	
		When executed	In conductive status	1.300	3.900	1.300	3.900	1.300	3.900
			In non-conductive status	1.300	3.700	1.300	3.700	1.300	3.700
	LD$<=	In conductive status		1.400	3.900	1.400	3.900	1.400	3.900
		In non-conductive status		1.400	3.800	1.400	3.800	1.400	3.800
	AND$<=	When not executed		0.018		0.018		0.018	
		When executed	In conductive status	1.300	3.800	1.300	3.800	1.300	3.800
			In non-conductive status	1.400	3.900	1.400	3.900	1.400	3.900
	OR$<=	When not executed		0.018		0.018		0.018	
		When executed	In conductive status	1.400	3.900	1.400	3.900	1.400	3.900
			In non-conductive status	1.300	3.700	1.300	3.700	1.300	3.700
	LD$<	In conductive status		1.400	3.900	1.400	3.900	1.400	3.900
		In non-conductive status		1.400	3.800	1.400	3.800	1.400	3.800
	AND$<	When not executed		0.018		0.018		0.018	
		When executed	In conductive status	1.300	3.700	1.300	3.700	1.300	3.700
			In non-conductive status	1.400	3.900	1.400	3.900	1.400	3.900
	OR$<	When not executed		0.018		0.018		0.018	
		When executed	In conductive status	1.300	3.900	1.300	3.900	1.300	3.900
			In non-conductive status	1.300	3.800	1.300	3.800	1.300	3.800
	LD$>=	In conductive status		1.400	3.800	1.400	3.800	1.400	3.800
		In non-conductive status		1.300	3.800	1.300	3.800	1.300	3.800
	AND$>=	When not executed		0.018		0.018		0.018	
		When executed	In conductive status	1.300	3.700	1.300	3.700	1.300	3.700
			In non-conductive status	1.300	3.900	1.300	3.900	1.300	3.900
	OR$>=	When not executed		0.018		0.018		0.018	
		When executed	In conductive status	1.300	3.800	1.300	3.800	1.300	3.800
			In non-conductive status	1.300	3.800	1.300	3.800	1.300	3.800

Category	Instruction	Condition (device)	Processing time (μs)					
			Q03UDVCPU		Q04UDVCPU, Q04UDPVCPU		Q06UDVCPU, Q06UDPVCPU, Q13UDVCPU, Q13UDPVCPU, Q26UDVCPU, Q26UDPVCPU	
			Min.	Max.	Min.	Max.	Min.	Max.
Basic instruction	BKCMP= (S1) (S2) (D) n	n=1	3.200	20.600	3.200	20.600	3.200	20.600
		n=96	16.800	34.000	16.800	34.000	16.800	34.000
	BKCMP<> (S1) (S2) (D) n	n=1	3.200	20.600	3.200	20.600	3.200	20.600
		n=96	16.800	34.000	16.800	34.000	16.800	34.000
	BKCMP> (S1) (S2) (D) n	n=1	3.200	20.800	3.200	20.800	3.200	20.800
		n=96	16.700	34.400	16.700	34.400	16.700	34.400
	BKCMP<= (S1) (S2) (D) n	n=1	3.200	21.000	3.200	21.000	3.200	21.000
		n=96	16.600	34.300	16.600	34.300	16.600	34.300
	BKCMP< (S1) (S2) (D) n	n=1	3.200	21.000	3.200	21.000	3.200	21.000
		n=96	16.800	34.400	16.800	34.400	16.800	34.400
	BKCMP>= (S1) (S2) (D) n	n=1	3.200	20.600	3.200	20.600	3.200	20.600
		n=96	16.800	34.000	16.800	34.000	16.800	34.000
	DBKCMP= (S1) (S2) (D) n	n=1	3.500	22.200	3.500	22.200	3.500	22.200
		n=96	17.100	35.700	17.100	35.700	17.100	35.700
	DBKCMP<> (S1) (S2) (D) n	n=1	3.500	22.700	3.500	22.700	3.500	22.700
		n=96	17.000	36.300	17.000	36.300	17.000	36.300
	DBKCMP> (S1) (S2) (D) n	n=1	3.500	22.700	3.500	22.700	3.500	22.700
		n=96	17.100	36.300	17.100	36.300	17.100	36.300
	DBKCMP<= (S1) (S2) (D) n	n=1	3.500	22.600	3.500	22.600	3.500	22.600
		n=96	17.000	36.000	17.000	36.000	17.000	36.000
	DBKCMP< (S1) (S2) (D) n	n=1	3.500	22.700	3.500	22.700	3.500	22.700
		n=96	17.100	36.500	17.100	36.500	17.100	36.500
	DBKCMP>= (S1) (S2) (D) n	n=1	3.500	22.700	3.500	22.700	3.500	22.700
		n=96	16.900	36.000	16.900	36.000	16.900	36.000
	CMP	—	2.300	7.800	2.300	7.800	2.300	7.800
	DCMP	—	2.300	7.800	2.300	7.800	2.300	7.800
	ZCP	—	2.600	8.500	2.600	8.500	2.600	8.500
	DZCP	—	2.600	8.500	2.600	8.500	2.600	8.500
	ECMP	—	2.600	10.000	2.600	10.000	2.600	10.000
	EZCP	—	3.000	10.800	3.000	10.800	3.000	10.800
	EDCMP	—	2.700	12.900	2.700	12.900	2.700	12.900
	EDZCP	—	3.200	14.300	3.200	14.300	3.200	14.300
	DB+ (S) (D)	When executed	2.000	8.400	2.000	8.400	2.000	8.400
	DB+ (S1) (S2) (D)	When executed	2.200	8.400	2.200	8.400	2.200	8.400
	DB- (S) (D)	When executed	2.000	8.200	2.000	8.200	2.000	8.200
	DB- (S1) (S2) (D)	When executed	2.200	8.600	2.200	8.600	2.200	8.600
	DB* (S1) (S2) (D)	When executed	2.700	12.200	2.700	12.200	2.700	12.200
	DB/ (S1) (S2) (D)	When executed	2.500	11.100	2.500	11.100	2.500	11.100

Category	Instruction	Condition (device)		Processing time (μs)					
				Q03UDVCPU		Q04UDVCPU, Q04UDPVCPU		Q06UDVCPU, Q06UDPVCPU, Q13UDVCPU, Q13UDPVCPU, Q26UDVCPU, Q26UDPVCPU	
				Min.	Max.	Min.	Max.	Min.	Max.
Basic instruction	ED+ (S) (D)	Double precision	(S)=0, (D)=0	1.700	6.800	1.700	6.800	1.700	6.800
			(S)=2^{1023}, (D)=2^{1023}	2.000	9.300	2.000	9.300	2.000	9.300
	ED+ (S1) (S2) (D)	Double precision	(S1)=0, (S2)=0	1.900	8.400	1.900	8.400	1.900	8.400
			(S1)=2^{1023}, (S2)=2^{1023}	2.100	10.500	2.100	10.500	2.100	10.500
	ED- (S) (D)	Double precision	(S)=0, (D)=0	1.700	8.000	1.700	8.000	1.700	8.000
			(S)=2^{1023}, (D)=2^{1023}	1.800	8.100	1.800	8.100	1.800	8.100
	ED- (S1) (S2) (D)	Double precision	(S1)=0, (S2)=0	1.900	8.100	1.900	8.100	1.900	8.100
			(S1)=2^{1023}, (S2)=2^{1023}	1.900	8.400	1.900	8.400	1.900	8.400
	ED* (S1) (S2) (D)	Double precision	(S1)=0, (S2)=0	1.900	8.700	1.900	8.700	1.900	8.700
			(S1)=2^{1023}, (S2)=2^{1023}	2.200	10.800	2.200	10.800	2.200	10.800
	ED/ (S1) (S2) (D)	Double precision	(S1)=2^{1023}, (S2)=2^{1023}	2.200	10.800	2.200	10.800	2.200	10.800
	BK+ (S1) (S2) (D) n	n=1		2.900	14.500	2.900	14.500	2.900	14.500
		n=96		17.000	28.700	17.000	28.700	17.000	28.700
	BK- (S1) (S2) (D) n	n=1		2.800	14.200	2.800	14.200	2.800	14.200
		n=96		17.000	28.400	17.000	28.400	17.000	28.400
	DBK+ (S1) (S2) (D) n	n=1		3.000	16.300	3.000	16.300	3.000	16.300
		n=96		17.100	30.500	17.100	30.500	17.100	30.500
	DBK- (S1) (S2) (D) n	n=1		3.000	16.300	3.000	16.300	3.000	16.300
		n=96		17.100	30.500	17.100	30.500	17.100	30.500
	$+ (S) (D)	—		1.900	6.200	1.900	6.200	1.900	6.200
	$+ (S1) (S2) (D)	—		2.300	7.800	2.300	7.800	2.300	7.800
	FLTD	Double precision	(S)=0	1.400	4.500	1.400	4.500	1.400	4.500
			(S)=7FFFH	1.400	4.400	1.400	4.400	1.400	4.400
	DFLTD	Double precision	(S)=0	1.400	4.300	1.400	4.300	1.400	4.300
			(S)=7FFFFFFFH	1.400	4.500	1.400	4.500	1.400	4.500
	INTD	Double precision	(S)=0	1.400	5.300	1.400	5.300	1.400	5.300
			(S)=32766.5	1.400	6.700	1.400	6.700	1.400	6.700
	DINTD	Double precision	(S)=0	1.400	5.300	1.400	5.300	1.400	5.300
			(S)=1234567890.3	1.500	6.500	1.500	6.500	1.500	6.500
	DBL	When executed		0.0039	0.015	0.0039	0.015	0.0039	0.015
	WORD	When executed		0.0078	0.015	0.0078	0.015	0.0078	0.015
	GRY	When executed		0.0078	0.015	0.0078	0.015	0.0078	0.015
	DGRY	When executed		0.0078	0.015	0.0078	0.015	0.0078	0.015
	GBIN	When executed		0.0078	0.015	0.0078	0.015	0.0078	0.015
	DGBIN	When executed		0.0078	0.015	0.0078	0.015	0.0078	0.015
	NEG	When executed		0.0078	0.015	0.0078	0.015	0.0078	0.015
	DNEG	When executed		0.0078	0.015	0.0078	0.015	0.0078	0.015

Category	Instruction	Condition (device)	Processing time (μs)					
			Q03UDVCPU		Q04UDVCPU, Q04UDPVCPU		Q06UDVCPU, Q06UDPVCPU, Q13UDVCPU, Q13UDPVCPU, Q26UDVCPU, Q26UDPVCPU	
			Min.	Max.	Min.	Max.	Min.	Max.
Basic instruction	ENEG	Floating point = 0	1.200	2.800	1.200	2.800	1.200	2.800
		Floating point = -1.0	1.300	3.300	1.300	3.300	1.300	3.300
	EDNEG	Floating point = 0	1.300	5.500	1.300	5.500	1.300	5.500
		Floating point = -1.0	1.300	6.200	1.300	6.200	1.300	6.200
	BKBCD (S) (D) n	n=1	2.200	12.800	2.200	12.800	2.200	12.800
		n=96	23.400	34.100	23.400	34.100	23.400	34.100
	BKBIN (S) (D) n	n=1	2.100	11.700	2.100	11.700	2.100	11.700
		n=96	16.600	26.200	16.600	26.200	16.600	26.200
	ECON	—	1.500	6.300	1.500	6.300	1.500	6.300
	EDCON	—	1.600	5.300	1.600	5.300	1.600	5.300
	EDMOV	—	0.0078	0.015	0.0078	0.015	0.0078	0.015
	$MOV	Character string to be transferred = 0	1.800	13.500	1.800	13.500	1.800	13.500
		Character string to be transferred = 32	3.900	15.400	3.900	15.400	3.900	15.400
	BXCH (D1) (D2) n	n=1	2.100	10.300	2.100	10.300	2.100	10.300
		n=96	16.200	24.400	16.200	24.400	16.200	24.400
	SWAP	—	1.200	2.200	1.200	2.200	1.200	2.200
	SMOV	—	3.100	12.300	3.100	12.300	3.100	12.300
	GOEND	—	0.580	2.100	0.580	2.100	0.580	2.100
	DI	—	2.200	3.400	2.200	3.400	2.200	3.400
	EI	—	3.100	6.400	3.100	6.400	3.100	6.400
	IMASK	—	4.100	8.000	4.100	8.000	4.100	8.000
	IRET	—	1.000	2.900	1.000	2.900	1.000	2.900
	RFS X n	n=1	2.200	10.100	2.200	10.100	2.200	10.100
		n=96	3.600	12.200	3.600	12.200	3.600	12.200
	RFS Y n	n=1	2.100	10.800	2.100	10.800	2.100	10.800
		n=96	3.600	13.200	3.600	13.200	3.600	13.200
	UDCNT1	—	1.000	1.700	1.000	1.700	1.000	1.700
	UDCNT2	—	1.000	2.000	1.000	2.000	1.000	2.000
	TTMR	—	1.800	8.000	1.800	8.000	1.800	8.000
	STMR	—	2.800	11.300	2.800	11.300	2.800	11.300
	ROTC	—	5.400	9.700	5.400	9.700	5.400	9.700
	RAMP	—	3.100	12.000	3.100	12.000	3.100	12.000
	SPD	—	1.000	1.800	1.000	1.800	1.000	1.800
	PLSY	—	1.100	2.100	1.100	2.100	1.100	2.100
	PWM	—	1.100	1.600	1.100	1.600	1.100	1.600
	MTR	—	3.000	14.500	3.000	14.500	3.000	14.500

Category	Instruction	Condition (device)		Processing time (μs)					
				Q03UDVCPU		Q04UDVCPU, Q04UDPVCPU		Q06UDVCPU, Q06UDPVCPU, Q13UDVCPU, Q13UDPVCPU, Q26UDVCPU, Q26UDPVCPU	
				Min.	Max.	Min.	Max.	Min.	Max.
Application instruction	BKAND (S1) (S2) (D) n	n=1		2.800	14.700	2.800	14.700	2.800	14.700
		n=96		17.000	28.800	17.000	28.800	17.000	28.800
	BKOR (S1) (S2) (D) n	n=1		2.900	14.300	2.900	14.300	2.900	14.300
		n=96		17.000	28.400	17.000	28.400	17.000	28.400
	BKXOR (S1) (S2) (D) n	n=1		2.800	14.600	2.800	14.600	2.800	14.600
		n=96		17.000	28.700	17.000	28.700	17.000	28.700
	BKXNR (S1) (S2) (D) n	n=1		2.900	14.600	2.900	14.600	2.900	14.600
		n=96		17.400	29.100	17.400	29.100	17.400	29.100
	BSFR (D) n	n=1		1.700	5.300	1.700	5.300	1.700	5.300
		n=96		2.300	9.600	2.300	9.600	2.300	9.600
	BSFL (D) n	n=1		1.700	5.300	1.700	5.300	1.700	5.300
		n=96		2.300	9.300	2.300	9.300	2.300	9.300
	SFTBR (D) n1 n2	n1=16, n2=1		3.300	16.200	3.300	16.200	3.300	16.200
		n1=16, n2=15		3.300	16.100	3.300	16.100	3.300	16.100
	SFTBL (D) n1 n2	n1=16, n2=1		3.300	16.000	3.300	16.000	3.300	16.000
		n1=16, n2=15		3.300	16.000	3.300	16.000	3.300	16.000
	SFTWR (D) n1 n2	n1=16, n2=1		2.000	11.900	2.000	11.900	2.000	11.900
		n1=16, n2=15		2.000	11.900	2.000	11.900	2.000	11.900
	SFTWL (D) n1 n2	n1=16, n2=1		2.000	11.900	2.000	11.900	2.000	11.900
		n1=16, n2=15		2.100	11.700	2.100	11.700	2.100	11.700
	SFTR (S) (D) n1 n2	n1=16, n2=1		5.600	21.400	5.600	21.400	5.600	21.400
		n1=16, n2=15		5.700	21.600	5.700	21.600	5.700	21.600
	SFTL (S) (D) n1 n2	n1=16, n2=1		5.700	22.000	5.700	22.000	5.700	22.000
		n1=16, n2=15		5.800	22.100	5.800	22.100	5.800	22.100
	WSFR (S) (D) n1 n2	n1=16, n2=1		4.400	14.200	4.400	14.200	4.400	14.200
		n1=16, n2=15		4.400	14.200	4.400	14.200	4.400	14.200
	WSFL (S) (D) n1 n2	n1=16, n2=1		4.400	14.900	4.400	14.900	4.400	14.900
		n1=16, n2=15		4.400	15.000	4.400	15.000	4.400	15.000
	BSET (D) n	n=1		0.0039	0.015	0.0039	0.015	0.0039	0.015
		n=15		0.0039	0.015	0.0039	0.015	0.0039	0.015
	BRST (D) n	n=1		0.0039	0.015	0.0039	0.015	0.0039	0.015
		n=15		0.0039	0.015	0.0039	0.015	0.0039	0.015
	TEST	When executed		0.012	0.031	0.012	0.031	0.012	0.031
	DTEST	When executed		0.019	0.031	0.019	0.031	0.019	0.031
	BKRST (S) n	n=1		1.700	9.200	1.700	9.200	1.700	9.200
		n=96		2.000	11.500	2.000	11.500	2.000	11.500
	SER (S1) (S2) (D) n	n=1	All match	2.700	8.000	2.700	8.000	2.700	8.000
			None match	2.700	8.000	2.700	8.000	2.700	8.000
		n=96	All match	9.600	18.100	9.600	18.100	9.600	18.100
			None match	9.600	18.100	9.600	18.100	9.600	18.100
	DSER (S1) (S2) (D) n	n=1	All match	2.800	10.600	2.800	10.600	2.800	10.600
			None match	2.800	10.600	2.800	10.600	2.800	10.600
		n=96	All match	14.400	22.000	14.400	22.000	14.400	22.000
			None match	14.400	21.900	14.400	21.900	14.400	21.900

A

Category	Instruction	Condition (device)		Processing time (μs)					
				Q03UDVCPU		Q04UDVCPU, Q04UDPVCPU		Q06UDVCPU, Q06UDPVCPU, Q13UDVCPU, Q13UDPVCPU, Q26UDVCPU, Q26UDPVCPU	
				Min.	Max.	Min.	Max.	Min.	Max.
Application instruction	DSUM (S) (D)	(S)=0		1.600	2.700	1.600	2.700	1.600	2.700
		(S)=FFFFFFFFH		1.600	2.800	1.600	2.800	1.600	2.800
	DECO (S) (D) n	n=2		2.600	12.800	2.600	12.800	2.600	12.800
		n=8		3.000	15.400	3.000	15.400	3.000	15.400
	ENCO (S) (D) n	n=2	M1=ON	2.400	8.600	2.400	8.600	2.400	8.600
			M4=ON	2.400	8.600	2.400	8.600	2.400	8.600
		n=8	M1=ON	3.800	13.400	3.800	13.400	3.800	13.400
			M256=ON	2.600	11.400	2.600	11.400	2.600	11.400
	DIS (S) (D) n	n=1		1.900	6.500	1.900	6.500	1.900	6.500
		n=4		2.000	6.700	2.000	6.700	2.000	6.700
	UNI (S) (D) n	n=1		2.200	6.700	2.200	6.700	2.200	6.700
		n=4		2.500	7.000	2.500	7.000	2.500	7.000
	NDIS	When executed		3.200	12.300	3.200	12.300	3.200	12.300
	NUNI	When executed		3.200	12.400	3.200	12.400	3.200	12.400
	WTOB (S) (D) n	n=1		2.100	7.300	2.100	7.300	2.100	7.300
		n=96		12.500	17.700	12.500	17.700	12.500	17.700
	BTOW (S) (D) n	n=1		2.000	7.000	2.000	7.000	2.000	7.000
		n=96		9.300	14.000	9.300	14.000	9.300	14.000
	MAX (S) (D) n	n=1		2.000	6.400	2.000	6.400	2.000	6.400
		n=96		9.100	13.700	9.100	13.700	9.100	13.700
	MIN (S) (D) n	n=1		2.000	7.100	2.000	7.100	2.000	7.100
		n=96		9.100	14.200	9.100	14.200	9.100	14.200
	DMAX (S) (D) n	n=1		2.200	11.000	2.200	11.000	2.200	11.000
		n=96		16.900	26.000	16.900	26.000	16.900	26.000
	DMIN (S) (D) n	n=1		2.300	11.000	2.300	11.000	2.300	11.000
		n=96		6.900	26.000	6.900	26.000	6.900	26.000
	SORT (S1) n (S2) (D1) (D2)	n=1, (S2)=1		3.000	7.700	3.000	7.700	3.000	7.700
		n=96, (S2)=16		8.100	17.500	8.100	17.500	8.100	17.500
	DSORT (S1) n (S2) (D1) (D2)	n=1, (S2)=1		3.100	8.800	3.100	8.800	3.100	8.800
		n=96, (S2)=16		9.800	20.300	9.800	20.300	9.800	20.300
	WSUM (S) (D) n	n=1		1.800	6.100	1.800	6.100	1.800	6.100
		n=96		6.600	11.000	6.600	11.000	6.600	11.000
	DWSUM (S) (D) n	n=1		2.300	11.400	2.300	11.400	2.300	11.400
		n=96		8.600	18.600	8.600	18.600	8.600	18.600
	MEAN (S) (D) n	n=1		2.100	6.800	2.100	6.800	2.100	6.800
		n=96		3.900	10.100	3.900	10.100	3.900	10.100
	DMEAN (S) (D) n	n=1		2.400	11.100	2.400	11.100	2.400	11.100
		n=96		8.700	16.100	8.700	16.100	8.700	16.100
	CCD	SM772 = OFF, Number of data blocks = 1		2.700	11.800	2.700	11.800	2.700	11.800
		SM772 = OFF, Number of data blocks = 96		11.100	20.400	11.100	20.400	11.100	20.400
		SM772 = ON, Number of data blocks = 1		2.700	11.700	2.700	11.700	2.700	11.700
		SM772 = ON, Number of data blocks = 96		10.500	19.000	10.500	19.000	10.500	19.000
	CRC	SM772 = OFF, Number of data blocks = 1		2.800	11.700	2.800	11.700	2.800	11.700
		SM772 = OFF, Number of data blocks = 96		11.700	24.400	11.700	24.400	11.700	24.400
		SM772 = ON, Number of data blocks = 1		2.700	12.600	2.700	12.600	2.700	12.600
		SM772 = ON, Number of data blocks = 96		10.900	23.700	10.900	23.700	10.900	23.700

Category	Instruction	Condition (device)	Processing time (μs)					
			Q03UDVCPU		Q04UDVCPU, Q04UDPVCPU		Q06UDVCPU, Q06UDPVCPU, Q13UDVCPU, Q13UDPVCPU, Q26UDVCPU, Q26UDPVCPU	
			Min.	Max.	Min.	Max.	Min.	Max.
Application instruction	NEXT	—	0.044		0.044		0.044	
	BREAK	—	4.000	12.000	4.000	12.000	4.000	12.000
	RET	Return to original program	0.200	0.200	0.200	0.200	0.200	0.200
		Return to other program	2.000	5.700	2.000	5.700	2.000	5.700
	FCALL Pn	Internal file pointer	1.100	1.400	1.100	1.400	1.100	1.400
		Common pointer	5.600	25.000	5.600	25.000	5.600	25.000
	FCALL Pn (S1) to (S5)	—	14.500	48.600	14.500	48.600	14.500	48.600
	ECALL * Pn *: Program name	—	48.800	86.000	48.800	86.000	48.800	86.000
	ECALL * Pn (S1) to (S5) *: Program name	—	62.000	97.900	62.000	97.900	62.000	97.900
	EFCALL * Pn *: Program name	—	50.000	90.700	50.000	90.700	50.000	90.700
	EFCALL * Pn (S1) to (S5) *: Program name	—	67.700	109.400	67.700	109.400	67.700	109.400
	XCALL	—	3.000	10.000	3.000	10.000	3.000	10.000
	COM CCOM	When selecting I/O refresh only	4.500	12.500	4.500	12.500	4.500	12.500
		When selecting CC-Link refresh only (master station side)	5.900	21.400	5.900	21.400	5.900	21.400
		When selecting CC-Link refresh only (local station side)	5.900	21.400	5.900	21.400	5.900	21.400
		• When selecting MELSECNET/H refresh only (control station side) • When selecting CC-Link IE Controller Network refresh only (control station side)	13.900	42.900	13.900	42.900	13.900	42.900
		• When selecting MELSECNET/H refresh only (normal station side) • When selecting CC-Link IE Controller Network refresh only (normal station side)	13.900	37.600	13.900	37.600	13.900	37.600
		When selecting CC-Link IE Field Network refresh only (master station side)	9.800	32.200	9.800	32.200	9.800	32.200
		When selecting CC-Link IE Field Network refresh only (local station side)	9.800	35.700	9.800	35.700	9.800	35.700
		When selecting intelli auto refresh only	5.300	15.100	5.300	15.100	5.300	15.100
		When selecting I/O outside the group only (Input only)	2.100	10.900	2.100	10.900	2.100	10.900
		When selecting I/O outside the group only (Output only)	7.500	30.500	7.500	30.500	7.500	30.500
		When selecting I/O outside the group only (Both I/O)	7.200	30.100	7.200	30.100	7.200	30.100
		When selecting refresh of multiple CPU high speed transmission area only	2.000	9.400	2.000	9.400	2.000	9.400
		When selecting communication with external devices only	7.100	28.200	7.100	28.200	7.100	28.200
		When selecting CC-Link IE Field Network Basic refresh only (master station side)	11.500	32.900	11.500	32.900	11.500	32.900

A

Category	Instruction	Condition (device)	Processing time (μs)					
			Q03UDVCPU		Q04UDVCPU, Q04UDPVCPU		Q06UDVCPU, Q06UDPVCPU, Q13UDVCPU, Q13UDPVCPU, Q26UDVCPU, Q26UDPVCPU	
			Min.	Max.	Min.	Max.	Min.	Max.
Application instruction	FIFW	Number of data points = 0	1.900	6.000	1.900	6.000	1.900	6.000
		Number of data points = 96	1.900	6.000	1.900	6.000	1.900	6.000
	FIFR	Number of data points = 1	2.000	6.500	2.000	6.500	2.000	6.500
		Number of data points = 96	7.100	12.200	7.100	12.200	7.100	12.200
	FPOP	Number of data points = 1	2.100	6.200	2.100	6.200	2.100	6.200
		Number of data points = 96	2.100	6.200	2.100	6.200	2.100	6.200
	FINS	Number of data points = 0	2.200	7.500	2.200	7.500	2.200	7.500
		Number of data points = 96	9.400	15.600	9.400	15.600	9.400	15.600
	FDEL	Number of data points = 1	2.400	7.100	2.400	7.100	2.400	7.100
		Number of data points = 96	7.500	12.800	7.500	12.800	7.500	12.800
	FROM n1 n2 (D) n3	n3=1	5.900	21.400	5.900	21.400	5.900	21.400
		n3=1000	366.700	383.200	366.700	383.200	366.700	383.200
	DFRO n1 n2 (D) n3	n3=1	6.500	22.900	6.500	22.900	6.500	22.900
		n3=500	366.700	405.100	366.700	405.100	366.700	405.100
	TO n1 n2 (S) n3	n3=1	5.100	19.100	5.100	19.100	5.100	19.100
		n3=1000	355.700	370.400	355.700	370.400	355.700	370.400
	DTO n1 n2 (S) n3	n3=1	6.900	21.700	6.900	21.700	6.900	21.700
		n3=500	355.700	370.200	355.700	370.200	355.700	370.200
	LEDR	No display → no display	0.900	3.000	0.900	3.000	0.900	3.000
		LED instruction execution → no display	8.700	26.600	8.700	26.600	8.700	26.600
	BINDA (S1) (D)	(S)=1	1.900	7.600	1.900	7.600	1.900	7.600
		(S)=-32768	2.200	7.800	2.200	7.800	2.200	7.800
	DBINDA (S) (D)	(S)=1	1.900	7.600	1.900	7.600	1.900	7.600
		(S)=-2147483648	2.300	8.000	2.300	8.000	2.300	8.000
	BINHA (S) (D)	(S)=1	1.800	7.200	1.800	7.200	1.800	7.200
		(S)=FFFFH	1.800	7.300	1.800	7.300	1.800	7.300
	DBINHA (S) (D)	(S)=1	1.800	7.100	1.800	7.100	1.800	7.100
		(S)=FFFFFFFFH	1.800	6.800	1.800	6.800	1.800	6.800
	BCDDA (S) (D)	(S)=1	1.800	7.600	1.800	7.600	1.800	7.600
		(S)=9999	1.800	7.700	1.800	7.700	1.800	7.700
	DBCDDA (S) (D)	(S)=1	1.700	7.200	1.700	7.200	1.700	7.200
		(S)=99999999	1.900	7.400	1.900	7.400	1.900	7.400
	DABIN (S) (D)	(S)=1	1.900	9.800	1.900	9.800	1.900	9.800
		(S)=-32768	1.900	9.800	1.900	9.800	1.900	9.800
	DDABIN (S) (D)	(S)=1	2.400	12.300	2.400	12.300	2.400	12.300
		(S)=-2147483648	2.400	12.400	2.400	12.400	2.400	12.400
	HABIN (S) (D)	(S)=1	1.900	10.000	1.900	10.000	1.900	10.000
		(S)=FFFFH	1.900	10.100	1.900	10.100	1.900	10.100
	DHABIN (S) (D)	(S)=1	2.100	9.900	2.100	9.900	2.100	9.900
		(S)=FFFFFFFFH	2.200	9.800	2.200	9.800	2.200	9.800
	DABCD (S) (D)	(S)=1	1.800	9.100	1.800	9.100	1.800	9.100
		(S)=9999	1.800	9.100	1.800	9.100	1.800	9.100
	DDABCD (S) (D)	(S)=1	2.100	10.100	2.100	10.100	2.100	10.100
		(S)=99999999	2.100	10.100	2.100	10.100	2.100	10.100
	COMRD	—	23.000	31.300	23.000	31.300	23.000	31.300

Category	Instruction	Condition (device)		Processing time (μs)					
				Q03UDVCPU		Q04UDVCPU, Q04UDPVCPU		Q06UDVCPU, Q06UDPVCPU, Q13UDVCPU, Q13UDPVCPU, Q26UDVCPU, Q26UDPVCPU	
				Min.	Max.	Min.	Max.	Min.	Max.
Application instruction	LEN	1 character		1.200	3.400	1.200	3.400	1.200	3.400
		96 characters		8.900	11.100	8.900	11.100	8.900	11.100
	STR	—		2.600	11.600	2.600	11.600	2.600	11.600
	DSTR	—		2.900	11.600	2.900	11.600	2.900	11.600
	VAL	—		3.700	14.100	3.700	14.100	3.700	14.100
	DVAL	—		4.400	160.000	4.400	160.000	4.400	160.000
	ESTR	—		4.200	23.700	4.200	23.700	4.200	23.700
	LDTM=	Comparison of specified clock	In conductive status	2.600	14.700	2.600	14.700	2.600	14.700
			In non-conductive status	2.600	14.300	2.600	14.300	2.600	14.300
		Comparison of current clock	In conductive status	4.900	21.600	4.900	21.600	4.900	21.600
			In non-conductive status	4.800	22.500	4.800	22.500	4.800	22.500
	ANDTM=	When not executed		0.018		0.018		0.018	
		Comparison of specified clock	In conductive status	2.500	14.400	2.500	14.400	2.500	14.400
			In non-conductive status	2.600	14.100	2.600	14.100	2.600	14.100
		Comparison of current clock	In conductive status	4.700	21.600	4.700	21.600	4.700	21.600
			In non-conductive status	4.800	22.200	4.800	22.200	4.800	22.200
	ORTM=	When not executed		0.018		0.018		0.018	
		Comparison of specified clock	In conductive status	2.600	14.800	2.600	14.800	2.600	14.800
			In non-conductive status	2.600	14.300	2.600	14.300	2.600	14.300
		Comparison of current clock	In conductive status	4.900	21.900	4.900	21.900	4.900	21.900
			In non-conductive status	4.700	22.000	4.700	22.000	4.700	22.000
	LDTM<>	Comparison of specified clock	In conductive status	2.600	14.400	2.600	14.400	2.600	14.400
			In non-conductive status	2.600	14.700	2.600	14.700	2.600	14.700
		Comparison of current clock	In conductive status	4.800	22.300	4.800	22.300	4.800	22.300
			In non-conductive status	4.800	21.800	4.800	21.800	4.800	21.800
	ANDTM<>	When not executed		0.018		0.018		0.018	
		Comparison of specified clock	In conductive status	2.600	14.200	2.600	14.200	2.600	14.200
			In non-conductive status	2.600	14.600	2.600	14.600	2.600	14.600
		Comparison of current clock	In conductive status	4.600	22.200	4.600	22.200	4.600	22.200
			In non-conductive status	4.800	21.600	4.800	21.600	4.800	21.600

Category	Instruction	Condition (device)		Processing time (μs)					
				Q03UDVCPU		Q04UDVCPU, Q04UDPVCPU		Q06UDVCPU, Q06UDPVCPU, Q13UDVCPU, Q13UDPVCPU, Q26UDVCPU, Q26UDPVCPU	
				Min.	Max.	Min.	Max.	Min.	Max.
Application instruction	ORTM<>	When not executed		0.018		0.018		0.018	
		Comparison of specified clock	In conductive status	2.600	15.000	2.600	15.000	2.600	15.000
			In non-conductive status	2.600	14.900	2.600	14.900	2.600	14.900
		Comparison of current clock	In conductive status	4.800	22.300	4.800	22.300	4.800	22.300
			In non-conductive status	4.700	21.500	4.700	21.500	4.700	21.500
	LDTM>	Comparison of specified clock	In conductive status	2.600	14.300	2.600	14.300	2.600	14.300
			In non-conductive status	2.600	14.300	2.600	14.300	2.600	14.300
		Comparison of current clock	In conductive status	4.800	22.400	4.800	22.400	4.800	22.400
			In non-conductive status	4.800	21.700	4.800	21.700	4.800	21.700
	ANDTM>	When not executed		0.018		0.018		0.018	
		Comparison of specified clock	In conductive status	2.500	14.500	2.500	14.500	2.500	14.500
			In non-conductive status	2.600	14.900	2.600	14.900	2.600	14.900
		Comparison of current clock	In conductive status	4.600	22.000	4.600	22.000	4.600	22.000
			In non-conductive status	4.800	21.800	4.800	21.800	4.800	21.800
	ORTM>	When not executed		0.018		0.018		0.018	
		Comparison of specified clock	In conductive status	2.600	14.400	2.600	14.400	2.600	14.400
			In non-conductive status	2.500	14.400	2.500	14.400	2.500	14.400
		Comparison of current clock	In conductive status	4.800	22.700	4.800	22.700	4.800	22.700
			In non-conductive status	4.700	21.800	4.700	21.800	4.700	21.800
	LDTM<=	Comparison of specified clock	In conductive status	2.600	14.400	2.600	14.400	2.600	14.400
			In non-conductive status	2.600	14.400	2.600	14.400	2.600	14.400
		Comparison of current clock	In conductive status	4.800	22.200	4.800	22.200	4.800	22.200
			In non-conductive status	4.800	22.100	4.800	22.100	4.800	22.100
	ANDTM<=	When not executed		0.018		0.018		0.018	
		Comparison of specified clock	In conductive status	2.600	14.400	2.600	14.400	2.600	14.400
			In non-conductive status	2.600	14.500	2.600	14.500	2.600	14.500
		Comparison of current clock	In conductive status	4.600	21.800	4.600	21.800	4.600	21.800
			In non-conductive status	4.700	22.400	4.700	22.400	4.700	22.400

Category	Instruction	Condition (device)		Processing time (μs)					
				Q03UDVCPU		Q04UDVCPU, Q04UDPVCPU		Q06UDVCPU, Q06UDPVCPU, Q13UDVCPU, Q13UDPVCPU, Q26UDVCPU, Q26UDPVCPU	
				Min.	Max.	Min.	Max.	Min.	Max.
Application instruction	ORTM<=	When not executed		0.018		0.018		0.018	
		Comparison of specified clock	In conductive status	2.600	14.800	2.600	14.800	2.600	14.800
			In non-conductive status	2.500	14.600	2.500	14.600	2.500	14.600
		Comparison of current clock	In conductive status	4.800	22.700	4.800	22.700	4.800	22.700
			In non-conductive status	4.700	22.400	4.700	22.400	4.700	22.400
	LDTM<	Comparison of specified clock	In conductive status	2.600	14.500	2.600	14.500	2.600	14.500
			In non-conductive status	2.600	14.500	2.600	14.500	2.600	14.500
		Comparison of current clock	In conductive status	4.800	21.900	4.800	21.900	4.800	21.900
			In non-conductive status	4.800	22.200	4.800	22.200	4.800	22.200
	ANDTM<	When not executed		0.018		0.018		0.018	
		Comparison of specified clock	In conductive status	2.500	14.500	2.500	14.500	2.500	14.500
			In non-conductive status	2.600	14.600	2.600	14.600	2.600	14.600
		Comparison of current clock	In conductive status	4.600	21.900	4.600	21.900	4.600	21.900
			In non-conductive status	4.700	22.200	4.700	22.200	4.700	22.200
	ORTM<	When not executed		0.018		0.018		0.018	
		Comparison of specified clock	In conductive status	2.600	14.800	2.600	14.800	2.600	14.800
			In non-conductive status	2.500	14.500	2.500	14.500	2.500	14.500
		Comparison of current clock	In conductive status	4.800	22.400	4.800	22.400	4.800	22.400
			In non-conductive status	4.700	22.200	4.700	22.200	4.700	22.200
	LDTM>=	Comparison of specified clock	In conductive status	2.600	14.500	2.600	14.500	2.600	14.500
			In non-conductive status	2.600	14.400	2.600	14.400	2.600	14.400
		Comparison of current clock	In conductive status	4.800	22.300	4.800	22.300	4.800	22.300
			In non-conductive status	4.800	22.200	4.800	22.200	4.800	22.200
	ANDTM>=	When not executed		0.018		0.018		0.018	
		Comparison of specified clock	In conductive status	2.600	14.500	2.600	14.500	2.600	14.500
			In non-conductive status	2.600	14.600	2.600	14.600	2.600	14.600
		Comparison of current clock	In conductive status	4.600	22.000	4.600	22.000	4.600	22.000
			In non-conductive status	4.700	21.900	4.700	21.900	4.700	21.900

Category	Instruction	Condition (device)		Processing time (μs)					
				Q03UDVCPU		Q04UDVCPU, Q04UDPVCPU		Q06UDVCPU, Q06UDPVCPU, Q13UDVCPU, Q13UDPVCPU, Q26UDVCPU, Q26UDPVCPU	
				Min.	Max.	Min.	Max.	Min.	Max.
Application instruction	ORTM>=	When not executed		0.018		0.018		0.018	
		Comparison of specified clock	In conductive status	2.600	14.800	2.600	14.800	2.600	14.800
			In non-conductive status	2.500	14.500	2.500	14.500	2.500	14.500
		Comparison of current clock	In conductive status	4.800	22.600	4.800	22.600	4.800	22.600
			In non-conductive status	4.700	22.100	4.700	22.100	4.700	22.100
	LDDT=	Comparison of specified date	In conductive status	2.700	14.900	2.700	14.900	2.700	14.900
			In non-conductive status	2.600	14.500	2.600	14.500	2.600	14.500
		Comparison of current date	In conductive status	4.800	22.600	4.800	22.600	4.800	22.600
			In non-conductive status	4.800	22.200	4.800	22.200	4.800	22.200
	ANDDT=	When not executed		0.018		0.018		0.018	
		Comparison of specified date	In conductive status	2.500	14.700	2.500	14.700	2.500	14.700
			In non-conductive status	2.500	15.200	2.500	15.200	2.500	15.200
		Comparison of current date	In conductive status	4.600	22.400	4.600	22.400	4.600	22.400
			In non-conductive status	4.700	22.600	4.700	22.600	4.700	22.600
	ORDT=	When not executed		0.018		0.018		0.018	
		Comparison of specified date	In conductive status	2.600	15.700	2.600	15.700	2.600	15.700
			In non-conductive status	2.600	15.000	2.600	15.000	2.600	15.000
		Comparison of current date	In conductive status	4.700	23.300	4.700	23.300	4.700	23.300
			In non-conductive status	4.700	22.500	4.700	22.500	4.700	22.500
	LDDT<>	Comparison of specified date	In conductive status	2.700	14.700	2.700	14.700	2.700	14.700
			In non-conductive status	2.700	14.900	2.700	14.900	2.700	14.900
		Comparison of current date	In conductive status	4.800	22.600	4.800	22.600	4.800	22.600
			In non-conductive status	4.800	22.800	4.800	22.800	4.800	22.800

Category	Instruction	Condition (device)		Processing time (μs)					
				Q03UDVCPU		Q04UDVCPU, Q04UDPVCPU		Q06UDVCPU, Q06UDPVCPU, Q13UDVCPU, Q13UDPVCPU, Q26UDVCPU, Q26UDPVCPU	
				Min.	Max.	Min.	Max.	Min.	Max.
Application instruction	ANDDT<>	When not executed		0.018		0.018		0.018	
		Comparison of specified date	In conductive status	2.600	15.100	2.600	15.100	2.600	15.100
			In non-conductive status	2.700	15.400	2.700	15.400	2.700	15.400
		Comparison of current date	In conductive status	4.600	22.100	4.600	22.100	4.600	22.100
			In non-conductive status	4.700	22.500	4.700	22.500	4.700	22.500
	ORDT<>	When not executed		0.018		0.018		0.018	
		Comparison of specified date	In conductive status	2.700	15.100	2.700	15.100	2.700	15.100
			In non-conductive status	2.600	15.100	2.600	15.100	2.600	15.100
		Comparison of current date	In conductive status	4.800	22.600	4.800	22.600	4.800	22.600
			In non-conductive status	4.700	22.800	4.700	22.800	4.700	22.800
	LDDT>	Comparison of specified date	In conductive status	2.700	14.700	2.700	14.700	2.700	14.700
			In non-conductive status	2.700	14.500	2.700	14.500	2.700	14.500
		Comparison of current date	In conductive status	4.800	22.400	4.800	22.400	4.800	22.400
			In non-conductive status	4.800	21.800	4.800	21.800	4.800	21.800
	ANDDT>	When not executed		0.018		0.018		0.018	
		Comparison of specified date	In conductive status	2.600	15.000	2.600	15.000	2.600	15.000
			In non-conductive status	2.700	15.100	2.700	15.100	2.700	15.100
		Comparison of current date	In conductive status	4.600	22.100	4.600	22.100	4.600	22.100
			In non-conductive status	4.500	22.000	4.500	22.000	4.500	22.000
	ORDT>	When not executed		0.018		0.018		0.018	
		Comparison of specified date	In conductive status	2.700	15.300	2.700	15.300	2.700	15.300
			In non-conductive status	2.600	15.000	2.600	15.000	2.600	15.000
		Comparison of current date	In conductive status	4.800	21.500	4.800	21.500	4.800	21.500
			In non-conductive status	4.700	22.400	4.700	22.400	4.700	22.400
	LDDT<=	Comparison of specified date	In conductive status	2.700	14.500	2.700	14.500	2.700	14.500
			In non-conductive status	2.700	14.500	2.700	14.500	2.700	14.500
		Comparison of current date	In conductive status	4.800	22.600	4.800	22.600	4.800	22.600
			In non-conductive status	4.800	22.600	4.800	22.600	4.800	22.600

Category	Instruction	Condition (device)		Processing time (μs)					
				Q03UDVCPU		Q04UDVCPU, Q04UDPVCPU		Q06UDVCPU, Q06UDPVCPU, Q13UDVCPU, Q13UDPVCPU, Q26UDVCPU, Q26UDPVCPU	
				Min.	Max.	Min.	Max.	Min.	Max.
Application instruction	ANDDT<=	When not executed		0.018		0.018		0.018	
		Comparison of specified date	In conductive status	2.400	14.500	2.400	14.500	2.400	14.500
			In non-conductive status	2.700	15.000	2.700	15.000	2.700	15.000
		Comparison of current date	In conductive status	4.700	21.600	4.700	21.600	4.700	21.600
			In non-conductive status	4.700	22.400	4.700	22.400	4.700	22.400
	ORDT<=	When not executed		0.018		0.018		0.018	
		Comparison of specified date	In conductive status	2.700	15.300	2.700	15.300	2.700	15.300
			In non-conductive status	2.600	15.200	2.600	15.200	2.600	15.200
		Comparison of current date	In conductive status	4.800	22.500	4.800	22.500	4.800	22.500
			In non-conductive status	4.800	21.700	4.800	21.700	4.800	21.700
	LDDT<	Comparison of specified date	In conductive status	2.700	14.500	2.700	14.500	2.700	14.500
			In non-conductive status	2.700	14.300	2.700	14.300	2.700	14.300
		Comparison of current date	In conductive status	4.800	22.700	4.800	22.700	4.800	22.700
			In non-conductive status	4.800	22.500	4.800	22.500	4.800	22.500
	ANDDT<	When not executed		0.018		0.018		0.018	
		Comparison of specified date	In conductive status	2.600	14.800	2.600	14.800	2.600	14.800
			In non-conductive status	2.600	14.900	2.600	14.900	2.600	14.900
		Comparison of current date	In conductive status	4.600	22.200	4.600	22.200	4.600	22.200
			In non-conductive status	4.700	22.100	4.700	22.100	4.700	22.100
	ORDT<	When not executed		0.018		0.018		0.018	
		Comparison of specified date	In conductive status	2.700	15.500	2.700	15.500	2.700	15.500
			In non-conductive status	2.600	15.000	2.600	15.000	2.600	15.000
		Comparison of current date	In conductive status	4.800	23.000	4.800	23.000	4.800	23.000
			In non-conductive status	4.700	22.600	4.700	22.600	4.700	22.600
	LDDT>=	Comparison of specified date	In conductive status	2.700	14.700	2.700	14.700	2.700	14.700
			In non-conductive status	2.700	14.700	2.700	14.700	2.700	14.700
		Comparison of current date	In conductive status	4.800	22.600	4.800	22.600	4.800	22.600
			In non-conductive status	4.800	22.900	4.800	22.900	4.800	22.900

Category	Instruction	Condition (device)		Processing time (μs)					
				Q03UDVCPU		Q04UDVCPU, Q04UDPVCPU		Q06UDVCPU, Q06UDPVCPU, Q13UDVCPU, Q13UDPVCPU, Q26UDVCPU, Q26UDPVCPU	
				Min.	Max.	Min.	Max.	Min.	Max.
Application instruction	ANDDT>=	When not executed		0.018		0.018		0.018	
		Comparison of specified date	In conductive status	2.600	15.000	2.600	15.000	2.600	15.000
			In non-conductive status	2.700	15.000	2.700	15.000	2.700	15.000
		Comparison of current date	In conductive status	4.600	22.100	4.600	22.100	4.600	22.100
			In non-conductive status	4.700	22.700	4.700	22.700	4.700	22.700
	ORDT>=	When not executed		0.018		0.018		0.018	
		Comparison of specified date	In conductive status	2.700	15.300	2.700	15.300	2.700	15.300
			In non-conductive status	2.600	15.000	2.600	15.000	2.600	15.000
		Comparison of current date	In conductive status	4.800	22.500	4.800	22.500	4.800	22.500
			In non-conductive status	4.700	22.700	4.700	22.700	4.700	22.700
	EVAL	Decimal point format all 2-digit specification		5.600	16.000	5.600	16.000	5.600	16.000
		Exponent format all 6-digit specification		2.100	16.300	2.100	16.300	2.100	16.300
	ASC (S) (D) n	n=1		8.100	11.000	8.100	11.000	8.100	11.000
		n=96		3.400	14.900	3.400	14.900	3.400	14.900
	HEX (S) (D) n	n=1		15.100	11.700	15.100	11.700	15.100	11.700
		n=96		3.300	17.700	3.300	17.700	3.300	17.700
	RIGHT (S) (D) n	n=1		15.100	15.400	15.100	15.400	15.100	15.400
		n=96		4.000	25.500	4.000	25.500	4.000	25.500
	LEFT (S) (D) n	n=1		4.500	16.000	4.500	16.000	4.500	16.000
		n=96		7.300	25.700	7.300	25.700	7.300	25.700
	MIDR	—		4.700	18.600	4.700	18.600	4.700	18.600
	MIDW	—		7.300	17.000	7.300	17.000	7.300	17.000
	INSTR	No match		2.900	19.200	2.900	19.200	2.900	19.200
		Match	Head	2.800	16.900	2.800	16.900	2.800	16.900
			End	16.900	19.400	16.900	19.400	16.900	19.400
	EMOD	—		19.900	12.100	19.900	12.100	19.900	12.100
	EREXP	—		15.300	13.800	15.300	13.800	15.300	13.800
	STRINS (S) (D) n	(S)=128/(D)=40/n=1		16.900	30.000	16.900	30.000	16.900	30.000
		(S)=128/(D)=40/n=48		19.900	32.500	19.900	32.500	19.900	32.500
	STRDEL (S) (D) n	(S)=128/(D)=40/n=1		15.300	25.600	15.300	25.600	15.300	25.600
		(S)=128/(D)=40/n=48		13.200	23.500	13.200	23.500	13.200	23.500

A

Category	Instruction	Condition (device)		Processing time (μs)					
				Q03UDVCPU		Q04UDVCPU, Q04UDPVCPU		Q06UDVCPU, Q06UDPVCPU, Q13UDVCPU, Q13UDPVCPU, Q26UDVCPU, Q26UDPVCPU	
				Min.	Max.	Min.	Max.	Min.	Max.
Application instruction	SIN	Single precision		1.600	6.700	1.600	6.700	1.600	6.700
	COS	Single precision		1.600	6.700	1.600	6.700	1.600	6.700
	TAN	Single precision		1.700	6.600	1.700	6.600	1.700	6.600
	ASIN	Single precision		1.500	6.700	1.500	6.700	1.500	6.700
	ACOS	Single precision		1.500	6.700	1.500	6.700	1.500	6.700
	ATAN	Single precision		1.500	5.200	1.500	5.200	1.500	5.200
	SIND	Double precision		2.600	20.500	2.600	20.500	2.600	20.500
	COSD	Double precision		2.500	19.900	2.500	19.900	2.500	19.900
	TAND	Double precision		3.000	22.700	3.000	22.700	3.000	22.700
	ASIND	Double precision		2.600	18.100	2.600	18.100	2.600	18.100
	ACOSD	Double precision		2.400	16.400	2.400	16.400	2.400	16.400
	ATAND	Double precision		2.300	16.400	2.300	16.400	2.300	16.400
	RAD	Single precision		1.300	3.500	1.300	3.500	1.300	3.500
	RADD	Double precision		1.400	3.600	1.400	3.600	1.400	3.600
	DEG	Single precision		1.800	11.500	1.800	11.500	1.800	11.500
	DEGD	Double precision		1.800	11.500	1.800	11.500	1.800	11.500
	SQR	Single precision		1.300	4.200	1.300	4.200	1.300	4.200
	SQRD	Double precision		1.900	11.500	1.900	11.500	1.900	11.500
	EXP (S) (D)	Single precision	(S)=-10	1.700	8.100	1.700	8.100	1.700	8.100
			(S)=1	1.700	8.100	1.700	8.100	1.700	8.100
	EXPD (S) (D)	Double precision	(S)=-10	2.300	17.400	2.300	17.400	2.300	17.400
			(S)=1	2.200	17.400	2.200	17.400	2.200	17.400
	LOG (S) (D)	Single precision	(S)=1	1.500	5.900	1.500	5.900	1.500	5.900
			(S)=10	1.600	7.000	1.600	7.000	1.600	7.000
	LOGD (S) (D)	Double precision	(S)=1	1.900	13.400	1.900	13.400	1.900	13.400
			(S)=10	2.400	17.400	2.400	17.400	2.400	17.400
	RND	—		0.800	2.200	0.800	2.200	0.800	2.200
	SRND	—		1.100	2.000	1.100	2.000	1.100	2.000
	BSQR (S) (D)	(S)=0		1.400	3.000	1.400	3.000	1.400	3.000
		(S)=9999		2.000	9.600	2.000	9.600	2.000	9.600
	BDSQR (S) (D)	(S)=0		1.100	2.200	1.100	2.200	1.100	2.200
		(S)=99999999		1.900	8.300	1.900	8.300	1.900	8.300
	BSIN	—		2.300	14.700	2.300	14.700	2.300	14.700
	BCOS	—		2.300	14.900	2.300	14.900	2.300	14.900
	BTAN	—		2.500	15.600	2.500	15.600	2.500	15.600
	BASIN	—		2.200	11.800	2.200	11.800	2.200	11.800
	BACOS	—		2.200	12.200	2.200	12.200	2.200	12.200
	BATAN	—		2.300	12.500	2.300	12.500	2.300	12.500
	POW (S1) (S2) (D)	Single precision	(S1)=12.3E+5 (S2)=3.45E+0	3.300	13.600	3.300	13.600	3.300	13.600
	POWD (S1) (S2) (D)	Double precision	(S1)=12.3E+5 (S2)=3.45E+0	4.400	26.600	4.400	26.600	4.400	26.600

Category	Instruction	Condition (device)		Processing time (μs)					
				Q03UDVCPU		Q04UDVCPU, Q04UDPVCPU		Q06UDVCPU, Q06UDPVCPU, Q13UDVCPU, Q13UDPVCPU, Q26UDVCPU, Q26UDPVCPU	
				Min.	Max.	Min.	Max.	Min.	Max.
Application instruction	LOG10	—		1.900	6.800	1.900	6.800	1.900	6.800
	LOG10D	—		2.500	19.000	2.500	19.000	2.500	19.000
	LIMIT	—		1.000	1.800	1.000	1.800	1.000	1.800
	DLIMIT	—		1.000	1.800	1.000	1.800	1.000	1.800
	BAND	—		2.000	4.000	2.000	4.000	2.000	4.000
	DBAND	—		2.000	4.000	2.000	4.000	2.000	4.000
	ZONE	—		2.000	3.700	2.000	3.700	2.000	3.700
	DZONE	—		2.000	3.500	2.000	3.500	2.000	3.500
	SCL (S1) (S2) (D)	SM750=ON	Point No.1 < (S1) < Point No.2	3.200	14.100	3.200	14.100	3.200	14.100
			Point No.9 < (S1) < Point No.10	3.200	14.200	3.200	14.200	3.200	14.200
		SM750=OFF	Point No.1 < (S1) < Point No.2	2.900	12.700	2.900	12.700	2.900	12.700
			Point No.9 < (S1) < Point No.10	3.400	12.700	3.400	12.700	3.400	12.700
	DSCL (S1) (S2) (D)	SM750=ON	Point No.1 < (S1) < Point No.2	3.200	14.500	3.200	14.500	3.200	14.500
			Point No.9 < (S1) < Point No.10	3.200	14.700	3.200	14.700	3.200	14.700
		SM750=OFF	Point No.1 < (S1) < Point No.2	2.700	12.200	2.700	12.200	2.700	12.200
			Point No.9 < (S1) < Point No.10	3.300	12.100	3.300	12.100	3.300	12.100
	SCL2 (S1) (S2) (D)	SM750=ON	Point No.1 < (S1) < Point No.2	3.200	14.000	3.200	14.000	3.200	14.000
			Point No.9 < (S1) < Point No.10	3.200	14.200	3.200	14.200	3.200	14.200
		SM750=OFF	Point No.1 < (S1) < Point No.2	2.900	12.300	2.900	12.300	2.900	12.300
			Point No.9 < (S1) < Point No.10	3.300	12.400	3.300	12.400	3.300	12.400
	DSCL2 (S1) (S2) (D)	SM750=ON	Point No.1 < (S1) < Point No.2	3.200	13.900	3.200	13.900	3.200	13.900
			Point No.9 < (S1) < Point No.10	3.200	13.900	3.200	13.900	3.200	13.900
		SM750=OFF	Point No.1 < (S1) < Point No.2	2.700	12.200	2.700	12.200	2.700	12.200
			Point No.9 < (S1) < Point No.10	3.300	12.200	3.300	12.200	3.300	12.200
	RSET	Standard RAM		1.600	9.300	1.600	9.300	1.600	9.300
		Extended SRAM cassette		1.600	8.300	1.600	8.300	1.600	8.300
	QDRSET	Extended SRAM cassette to standard RAM		51.800	88.600	51.800	88.600	51.800	88.600
		Standard RAM to extended SRAM cassette		53.300	88.600	53.300	88.600	53.300	88.600
	QCDSET	Extended SRAM cassette to standard ROM		930.000	1008.000	930.000	1008.000	930.000	1008.000
		Standard ROM to extended SRAM cassette		56.000	68.000	56.000	68.000	56.000	68.000
	DATERD	—		2.500	13.900	2.500	13.900	2.500	13.900
	DATEWR	—		4.100	19.100	4.100	19.100	4.100	19.100
	DATE+	No digit increase		2.400	8.500	2.400	8.500	2.400	8.500
		Digit increase		2.400	8.500	2.400	8.500	2.400	8.500
	DATE-	No digit increase		2.400	8.500	2.400	8.500	2.400	8.500
		Digit increase		2.500	8.400	2.500	8.400	2.500	8.400
	DATE2SEC	—		3.000	11.000	3.000	11.000	3.000	11.000
	SEC2DATE	—		3.200	17.500	3.200	17.500	3.200	17.500

Category	Instruction	Condition (device)	Processing time (μs)					
			Q03UDVCPU		Q04UDVCPU, Q04UDPVCPU		Q06UDVCPU, Q06UDPVCPU, Q13UDVCPU, Q13UDPVCPU, Q26UDVCPU, Q26UDPVCPU	
			Min.	Max.	Min.	Max.	Min.	Max.
Application instruction	SECOND	—	1.600	5.200	1.600	5.200	1.600	5.200
	HOUR	—	1.500	5.400	1.500	5.400	1.500	5.400
	HOURM	—	2.700	10.800	2.700	10.800	2.700	10.800
	DHOURM	—	2.900	11.800	2.900	11.800	2.900	11.800
	TCMP	—	3.300	13.500	3.300	13.500	3.300	13.500
	TZCP	—	3.700	15.400	3.700	15.400	3.700	15.400
	S.DATERD	—	4.400	19.500	4.400	19.500	4.400	19.500
	S.DATE+	No digit increase	3.400	14.800	3.400	14.800	3.400	14.800
		Digit increase	3.500	14.500	3.500	14.500	3.500	14.500
	S.DATE-	No digit increase	3.400	15.800	3.400	15.800	3.400	15.800
		Digit increase	3.400	15.500	3.400	15.500	3.400	15.500
	PSTOP	—	35.600	68.400	35.600	68.400	35.600	68.400
	POFF	—	35.400	66.800	35.400	66.800	35.400	66.800
	PSCAN	—	36.700	69.100	36.700	69.100	36.700	69.100
	WDT	—	1.000	3.300	1.000	3.300	1.000	3.300
	DUTY	—	2.500	9.700	2.500	9.700	2.500	9.700
	TIMCHK	—	2.500	6.900	2.500	6.900	2.500	6.900
	ZRRDB	File register of standard RAM	1.600	3.500	1.600	3.500	1.600	3.500
		File register of extended SRAM cassette	1.600	3.500	1.600	3.500	1.600	3.500
	ZRWRB	File register of standard RAM	1.700	4.100	1.700	4.100	1.700	4.100
		File register of extended SRAM cassette	1.700	4.100	1.700	4.100	1.700	4.100
	ADRSET	—	1.400	3.800	1.400	3.800	1.400	3.800
	ZPUSH	—	3.100	7.000	3.100	7.000	3.100	7.000
	ZPOP	—	3.000	3.900	3.000	3.900	3.000	3.900
	UNIRD n1 (D) n2	n2=1	2.100	10.100	2.100	10.100	2.100	10.100
		n2=16	4.200	12.600	4.200	12.600	4.200	12.600
	TYPERD	—	16.100	36.800	16.100	36.800	16.100	36.800
	TRACE	Start	32.200	46.200	32.200	46.200	32.200	46.200
	TRACER	—	2.600	8.800	2.600	8.800	2.600	8.800
	RBMOV (S) (D) n	When standard RAM is used: 1 point	3.500	21.400	3.500	21.400	3.500	21.400
		When standard RAM is used: 1000 points	42.200	58.000	42.200	58.000	42.200	58.000
		When extended SRAM cassette is used: 1 point	3.600	19.600	3.600	19.600	3.600	19.600
		When extended SRAM cassette is used: 1000 points	102.200	118.000	102.200	118.000	102.200	118.000
	PID	—	2.900	25.400	2.900	25.400	2.900	25.400
	SP.FREAD	—	3.500	39.400	3.500	39.400	3.500	39.400
	SP.FWRITE	—	3.500	39.500	3.500	39.500	3.500	39.500
	SP.DEVST	—	25.900	37.600	25.900	37.600	25.900	37.600
	S.DEVLD	—	3.700	20.000	3.700	20.000	3.700	20.000

Category	Instruction	Condition (device)		Processing time (μs)					
				Q03UDVCPU		Q04UDVCPU, Q04UDPVCPU		Q06UDVCPU, Q06UDPVCPU, Q13UDVCPU, Q13UDPVCPU, Q26UDVCPU, Q26UDPVCPU	
				Min.	Max.	Min.	Max.	Min.	Max.
Data link instruction	S.ZCOM	When mounting CC-Link module (master station side)		7.100	25.000	7.100	25.000	7.100	25.000
		When mounting CC-Link module (local station side)		7.100	25.200	7.100	25.200	7.100	25.200
		• When selecting MELSECNET/H refresh only (control station side) • When selecting CC-Link IE Controller Network refresh only (control station side)		18.400	48.300	18.400	48.300	18.400	48.300
		• When selecting MELSECNET/H refresh only (normal station side) • When selecting CC-Link IE Controller Network refresh only (normal station side)		18.400	43.500	18.400	43.500	18.400	43.500
		When selecting CC-Link IE Field Network refresh only (master station side)		12.000	38.500	12.000	38.500	12.000	38.500
		When selecting CC-Link IE Field Network refresh only (local station side)		11.900	41.900	11.900	41.900	11.900	41.900
	S.RTREAD	—		2.900	15.400	2.900	15.400	2.900	15.400
	S.RTWRITE	—		3.100	15.600	3.100	15.600	3.100	15.600
	S.REFDVWRB	When the refresh device as the write target exists at Transfer 1		34.500	51.000	34.500	51.000	34.500	51.000
		When the refresh device as the write target exists at Transfer 256		91.000	109.000	91.000	109.000	91.000	109.000
	S.REFDVWRW	When the refresh device as the write target exists at Transfer 1		34.500	51.000	34.500	51.000	34.500	51.000
		When the refresh device as the write target exists at Transfer 256		91.000	109.000	91.000	109.000	91.000	109.000
	S.REFDVRDB	When the refresh device as the read target exists at Transfer 1		34.500	51.000	34.500	51.000	34.500	51.000
		When the refresh device as the read target exists at Transfer 256		91.000	109.000	91.000	109.000	91.000	109.000
	S.REFDVRDW	When the refresh device as the read target exists at Transfer 1		34.500	51.000	34.500	51.000	34.500	51.000
		When the refresh device as the read target exists at Transfer 256		91.000	109.000	91.000	109.000	91.000	109.000
Multiple CPU dedicated instruction	S.TO n1 n2 n3 n4 (D)	Writing to host CPU shared memory	n4=1	14.700	30.000	14.700	30.000	14.700	30.000
			n4=320	48.700	62.900	48.700	62.900	48.700	62.900
	TO n1 n2 (S) n3	Writing to host CPU shared memory	n3=1	3.700	18.200	3.700	18.200	3.700	18.200
			n3=320	36.500	50.800	36.500	50.800	36.500	50.800
	DTO n1 n2 (S) n3	Writing to host CPU shared memory	n3=1	3.700	18.600	3.700	18.600	3.700	18.600
			n3=320	68.700	82.700	68.700	82.700	68.700	82.700
	FROM n1 n2 (D) n3	Reading from host CPU shared memory	n3=1	3.700	17.500	3.700	17.500	3.700	17.500
			n3=320	27.400	41.000	27.400	41.000	27.400	41.000
		Reading from other CPU shared memory	n3=1	5.100	28.000	5.100	28.000	5.100	28.000
			n3=320	125.600	152.100	125.600	152.100	125.600	152.100
			n3=1000	386.000	413.900	386.000	413.900	386.000	413.900
	DFRO n1 n2 (D) n3	Reading from host CPU shared memory	n3=1	3.700	17.500	3.700	17.500	3.700	17.500
			n3=320	51.500	65.000	51.500	65.000	51.500	65.000
		Reading from other CPU shared memory	n3=1	5.700	30.300	5.700	30.300	5.700	30.300
			n3=320	246.700	272.000	246.700	272.000	246.700	272.000
			n3=1000	764.500	792.800	764.500	792.800	764.500	792.800

Category	Instruction	Condition (device)		Processing time (μs)					
				Q03UDVCPU		Q04UDVCPU, Q04UDPVCPU		Q06UDVCPU, Q06UDPVCPU, Q13UDVCPU, Q13UDPVCPU, Q26UDVCPU, Q26UDPVCPU	
				Min.	Max.	Min.	Max.	Min.	Max.
Multiple CPU high-speed transmission dedicated instruction	D.DDWR n (S1) (S2) (D1) (D2)	Writing Devices to Another CPU	n=1	34.000	82.000	34.000	82.000	34.000	82.000
			n=16	37.000	84.000	37.000	84.000	37.000	84.000
			n=96	52.000	98.000	52.000	98.000	52.000	98.000
	DP.DDWR n (S1) (S2) (D1) (D2)		n=1	34.000	82.000	34.000	82.000	34.000	82.000
			n=16	37.000	84.000	37.000	84.000	37.000	84.000
			n=96	52.000	98.000	52.000	98.000	52.000	98.000
	D.DDRD n (S1) (S2) (D1) (D2)	Reads devices from another CPU.	n=1	34.000	81.000	34.000	81.000	34.000	81.000
			n=16	34.000	81.000	34.000	81.000	34.000	81.000
			n=96	34.000	81.000	34.000	81.000	34.000	81.000
	DP.DDRD n (S1) (S2) (D1) (D2)		n=1	34.000	81.000	34.000	81.000	34.000	81.000
			n=16	34.000	81.000	34.000	81.000	34.000	81.000
			n=96	34.000	81.000	34.000	81.000	34.000	81.000

Point

For the instructions for which a rising edge instruction (□P) is not described, the processing time is the same as an ON execution instruction.
(Example) WORDP instruction, TOP instruction etc.

■Table of the time to be added when file register, extended data register, extended link register, module access device, and link direct device are used

• When using Q00UJCPU, Q00UCPU, Q01UCPU and Q02UCPU

Device name		Data	Device specification location	Addition time (μs)			
				Q00UJCPU	Q00UCPU	Q01UCPU	Q02UCPU
File register (R)	When standard RAM is used	Bit	Source	0.100	0.100	0.100	0.100
			Destination	0.100	0.100	0.100	0.100
		Word	Source	0.100	0.100	0.100	0.100
			Destination	0.100	0.100	0.100	0.100
		Double word	Source	0.100	0.100	0.100	0.200
			Destination	0.100	0.100	0.100	0.200
	When SRAM card is used (Q2MEM-1MBS, Q2MEM-2MBS)	Bit	Source	—	—	—	0.220
			Destination	—	—	—	0.180
		Word	Source	—	—	—	0.220
			Destination	—	—	—	0.180
		Double word	Source	—	—	—	0.440
			Destination	—	—	—	0.380
	When SRAM card is used (Q3MEM-4MBS, Q3MEM-8MBS)	Bit	Source	—	—	—	0.160
			Destination	—	—	—	0.140
		Word	Source	—	—	—	0.160
			Destination	—	—	—	0.140
		Double word	Source	—	—	—	0.320
			Destination	—	—	—	0.300
File register (ZR), extended data register (D), extended link register (W)	When standard RAM is used	Bit	Source	0.120	0.120	0.120	0.120
			Destination	0.120	0.120	0.120	0.120
		Word	Source	0.120	0.120	0.120	0.120
			Destination	0.120	0.120	0.120	0.120
		Double word	Source	0.120	0.120	0.120	0.220
			Destination	0.120	0.120	0.120	0.220
	When SRAM card is used (Q2MEM-1MBS, Q2MEM-2MBS)	Bit	Source	—	—	—	0.240
			Destination	—	—	—	0.200
		Word	Source	—	—	—	0.240
			Destination	—	—	—	0.200
		Double word	Source	—	—	—	0.460
			Destination	—	—	—	0.400
	When SRAM card is used (Q3MEM-4MBS, Q3MEM-8MBS)	Bit	Source	—	—	—	0.180
			Destination	—	—	—	0.160
		Word	Source	—	—	—	0.180
			Destination	—	—	—	0.160
		Double word	Source	—	—	—	0.340
			Destination	—	—	—	0.320
Module access device (Un\G□, U3En\G0 to G4095)		Bit	Source	—	—	—	12.000
			Destination	—	—	—	17.300
		Word	Source	—	—	—	9.700
			Destination	—	—	—	33.000
		Double word	Source	—	—	—	24.200
			Destination	—	—	—	34.800
Link direct device (Jn\□)		Bit	Source	70.900	70.900	70.900	46.200
			Destination	120.100	120.100	120.100	75.000
		Word	Source	68.400	68.400	68.400	44.800
			Destination	53.700	53.700	53.700	33.600
		Double word	Source	75.600	75.600	75.600	60.300
			Destination	58.900	58.900	58.900	41.900

• When using Q03UD(E)HCPU, Q04UD(E)HCPU, Q06UD(E)HCPU, Q10UD(E)HCPU, Q13UD(E)HCPU, Q20UD(E)HCPU, Q26UD(E)HCPU, Q50UDEHCPU, and Q100UDEHCPU

Device name		Data	Device specification location	Addition time (μs)			
				Q03UD(E)CPU	Q04UD(E)HCPU, Q06UD(E)HCPU	Q10UD(E)HCPU, Q13UD(E)HCPU, Q20UD(E)HCPU, Q26UD(E)HCPU	Q50UDEHCPU, Q100UDEHCPU
File register (R)	When standard RAM is used	Bit	Source	0.100	0.048	0.048	0.048
			Destination	0.100	0.038	0.038	0.038
		Word	Source	0.100	0.048	0.048	0.048
			Destination	0.100	0.038	0.038	0.038
		Double word	Source	0.200	0.095	0.095	0.095
			Destination	0.200	0.086	0.086	0.086
	When SRAM card is used (Q2MEM-1MBS, Q2MEM-2MBS)	Bit	Source	0.220	0.200	0.200	0.200
			Destination	0.180	0.162	0.162	0.162
		Word	Source	0.220	0.200	0.200	0.200
			Destination	0.180	0.162	0.162	0.162
		Double word	Source	0.440	0.399	0.399	0.399
			Destination	0.380	0.361	0.361	0.361
	When SRAM card is used (Q3MEM-4MBS, Q3MEM-8MBS)	Bit	Source	0.160	0.152	0.152	0.152
			Destination	0.140	0.133	0.133	0.133
		Word	Source	0.160	0.152	0.152	0.152
			Destination	0.140	0.133	0.133	0.133
		Double word	Source	0.320	0.304	0.304	0.304
			Destination	0.300	0.295	0.295	0.295
File register (ZR), extended data register (D), extended link register (W)	When standard RAM is used	Bit	Source	0.120	0.057	0.057	0.057
			Destination	0.120	0.048	0.048	0.048
		Word	Source	0.120	0.057	0.057	0.057
			Destination	0.120	0.048	0.048	0.048
		Double word	Source	0.220	0.105	0.105	0.105
			Destination	0.220	0.095	0.095	0.095
	When SRAM card is used (Q2MEM-1MBS, Q2MEM-2MBS)	Bit	Source	0.240	0.209	0.209	0.209
			Destination	0.200	0.171	0.171	0.171
		Word	Source	0.240	0.209	0.209	0.209
			Destination	0.200	0.171	0.171	0.171
		Double word	Source	0.460	0.409	0.409	0.409
			Destination	0.400	0.371	0.371	0.371
	When SRAM card is used (Q3MEM-4MBS, Q3MEM-8MBS)	Bit	Source	0.180	0.162	0.162	0.162
			Destination	0.160	0.143	0.143	0.143
		Word	Source	0.180	0.162	0.162	0.162
			Destination	0.160	0.143	0.143	0.143
		Double word	Source	0.340	0.314	0.314	0.314
			Destination	0.320	0.304	0.304	0.304
Module access device (Un\G□, U3En\G0 to G4095)		Bit	Source	11.700	11.200	11.200	11.200
			Destination	15.400	15.300	15.300	15.300
		Word	Source	9.460	9.410	9.410	9.410
			Destination	19.000	19.000	19.000	19.000
		Double word	Source	11.000	10.900	10.900	10.900
			Destination	18.800	18.700	18.700	18.700

Device name	Data	Device specification location	Addition time (μs)			
			Q03UD(E)CPU	Q04UD(E)HCPU, Q06UD(E)HCPU	Q10UD(E)HCPU, Q13UD(E)HCPU, Q20UD(E)HCPU, Q26UD(E)HCPU	Q50UDEHCPU, Q100UDEHCPU
Link direct device (Jn\□)	Bit	Source	32.700	31.300	31.300	31.300
		Destination	52.300	51.800	51.800	51.800
	Word	Source	30.600	30.100	30.100	30.100
		Destination	28.900	28.400	28.400	28.400
	Double word	Source	38.900	38.400	38.400	38.400
		Destination	34.800	34.300	34.300	34.300

- When using Q03UDVCPU, Q04UDVCPU, Q04UDPVCPU, Q06UDVCPU, Q06UDPVCPU, Q13UDVCPU, Q13UDPVCPU, Q26UDVCPU, and Q26UDPVCPU

Device name		Data	Device specification location	Addition time (μs)		
				Q03UDVCPU	Q04UDVCPU, Q04UDPVCPU	Q06UDVCPU, Q06UDPVCPU, Q13UDVCPU, Q13UDPVCPU, Q26UDVCPU, Q26UDPVCPU
File register (R)	When the extended SRAM cassette is not used	Bit	Source	0.074	0.043	0.043
			Destination	0.023	0.023	0.023
		Word	Source	0.074	0.043	0.043
			Destination	0.023	0.023	0.023
		Double word	Source	0.148	0.085	0.085
			Destination	0.044	0.044	0.044
	When the extended SRAM cassette is used	Bit	Source	0.099	0.099	0.099
			Destination	0.028	0.028	0.028
		Word	Source	0.099	0.099	0.099
			Destination	0.028	0.028	0.028
		Double word	Source	0.198	0.198	0.198
			Destination	0.054	0.054	0.054
File register (ZR), extended data register (D), extended link register (W)	When the extended SRAM cassette is not used	Bit	Source	0.074	0.043	0.043
			Destination	0.023	0.023	0.023
		Word	Source	0.074	0.043	0.043
			Destination	0.023	0.023	0.023
		Double word	Source	0.148	0.085	0.085
			Destination	0.044	0.044	0.044
	When the extended SRAM cassette is used	Bit	Source	0.099	0.099	0.099
			Destination	0.028	0.028	0.028
		Word	Source	0.099	0.099	0.099
			Destination	0.028	0.028	0.028
		Double word	Source	0.198	0.198	0.198
			Destination	0.054	0.054	0.054
Module access device (Un\G□, U3En\G0 to G4095)		Bit	Source	8.200	8.200	8.200
			Destination	11.800	11.800	11.800
		Word	Source	9.410	9.410	9.410
			Destination	9.400	9.400	9.400
		Double word	Source	10.900	10.900	10.900
			Destination	18.700	18.700	18.700

A

Device name	Data	Device specification location	Addition time (μs)		
			Q03UDVCPU	Q04UDVCPU, Q04UDPVCPU	Q06UDVCPU, Q06UDPVCPU, Q13UDVCPU, Q13UDPVCPU, Q26UDVCPU, Q26UDPVCPU
Link direct device (Jn\□)	Bit	Source	16.200	16.200	16.200
		Destination	27.400	27.400	27.400
	Word	Source	19.800	19.800	19.800
		Destination	17.400	17.400	17.400
	Double word	Source	18.200	18.200	18.200
		Destination	17.400	17.400	17.400

Operation processing time of LCPU

The processing time for the individual instructions are shown in the table on the following pages.
Operation processing times can vary substantially depending on the nature of the sources and destinations of the instructions, and the values contained in the following tables should therefore be taken as a set of general guidelines to processing time rather than as being strictly accurate.

Subset instruction processing time

The following describes the subset instruction processing time.

Point

- The processing time shown in "Subset instruction processing time table" applies when the device used in an instruction meets the device condition for subset processing. (For device condition triggering subset processing, refer to Page 109 Subset processing.)
- When using the file resister (R, ZR), extended data register (D), or extended link register (W), add the processing time shown in Page 1042 Table of the time to be added when file register, extended data register, and extended link register are used to that of each instruction.
- When using the F, T(ST), or C device with the OUT, SET, or RST instruction, add the processing time shown in Page 1042 Table of the time to be added when F/T(ST)/C device is used in OUT/SET/RST instruction to that of each instruction.
- Since the processing time of an instruction varies depending on that of the cash function, both the minimum and maximum values are described in the table.

■Subset instruction processing time table

- When using L02SCPU, L02SCPU-P, L02CPU, L02CPU-P, L06CPU, L06CPU-P, L26CPU, L26CPU-P, L26CPU-BT, and L26CPU-PBT

Category	Instruction	Condition (device)		Processing time (μs)					
				L02SCPU, L02SCPU-P		L02CPU, L02CPU-P		L06CPU, L06CPU-P, L26CPU, L26CPU-P, L26CPU-BT, L26CPU-PBT	
				Min.	Max.	Min.	Max.	Min.	Max.
Sequence instruction	LD LDI AND ANI OR ORI LDP LDF ANDP ANDF ORP ORF	When executed		0.060		0.040		0.0095	
	LDPI LDFI	When executed		0.180		0.120		0.0285	
	ANDPI ANDFI ORPI ORFI	When executed		0.240		0.160		0.038	
	OUT	When not changed		0.060		0.040		0.0095	
		When changed							
	OUTH	When not changed		0.060		0.040		0.0095	
		When changed							
	SET RST	When not executed		0.060		0.040		0.0095	
		When executed	When not changed						
			When changed						

A

Category	Instruction	Condition (device)		Processing time (μs)					
				L02SCPU, L02SCPU-P		L02CPU, L02CPU-P		L06CPU, L06CPU-P, L26CPU, L26CPU-P, L26CPU-BT, L26CPU-PBT	
				Min.	Max.	Min.	Max.	Min.	Max.
Basic instruction	LD=	In conductive status		0.180		0.120		0.0285	
		In non-conductive status							
	AND=	When not executed		0.180		0.120		0.0285	
		When executed	In conductive status						
			In non-conductive status						
	OR=	When not executed		0.180		0.120		0.0285	
		When executed	In conductive status						
			In non-conductive status						
	LD<>	In conductive status		0.180		0.120		0.0285	
		In non-conductive status							
	AND<>	When not executed		0.180		0.120		0.0285	
		When executed	In conductive status						
			In non-conductive status						
	OR<>	When not executed		0.180		0.120		0.0285	
		When executed	In conductive status						
			In non-conductive status						
	LD>	In conductive status		0.180		0.120		0.0285	
		In non-conductive status							
	AND>	When not executed		0.180		0.120		0.0285	
		When executed	In conductive status						
			In non-conductive status						
	OR>	When not executed		0.180		0.120		0.0285	
		When executed	In conductive status						
			In non-conductive status						
	LD<=	In conductive status		0.180		0.120		0.0285	
		In non-conductive status							
	AND<=	When not executed		0.180		0.120		0.0285	
		When executed	In conductive status						
			In non-conductive status						
	OR<=	When not executed		0.180		0.120		0.0285	
		When executed	In conductive status						
			In non-conductive status						

Category	Instruction	Condition (device)		Processing time (μs)					
				L02SCPU, L02SCPU-P		L02CPU, L02CPU-P		L06CPU, L06CPU-P, L26CPU, L26CPU-P, L26CPU-BT, L26CPU-PBT	
				Min.	Max.	Min.	Max.	Min.	Max.
Basic instruction	LD<	In conductive status		0.180		0.120		0.0285	
		In non-conductive status							
	AND<	When not executed		0.180		0.120		0.0285	
		When executed	In conductive status						
			In non-conductive status						
	OR<	When not executed		0.180		0.120		0.0285	
		When executed	In conductive status						
			In non-conductive status						
	LD>=	In conductive status		0.180		0.120		0.0285	
		In non-conductive status							
	AND>=	When not executed		0.180		0.120		0.0285	
		When executed	In conductive status						
			In non-conductive status						
	OR>=	When not executed		0.180		0.120		0.0285	
		When executed	In conductive status						
			In non-conductive status						
	LDD=	In conductive status		0.180		0.120		0.0285	
		In non-conductive status							
	ANDD=	When not executed		0.180		0.120		0.0285	
		When executed	In conductive status						
			In non-conductive status						
	ORD=	When not executed		0.180		0.120		0.0285	
		When executed	In conductive status						
			In non-conductive status						
	LDD<>	In conductive status		0.180		0.120		0.0285	
		In non-conductive status							
	ANDD<>	When not executed		0.180		0.120		0.0285	
		When executed	In conductive status						
			In non-conductive status						
	ORD<>	When not executed		0.180		0.120		0.0285	
		When executed	In conductive status						
			In non-conductive status						
	LDD>	In conductive status		0.180		0.120		0.0285	
		In non-conductive status							
	ANDD>	When not executed		0.180		0.120		0.0285	
		When executed	In conductive status						
			In non-conductive status						
	ORD>	When not executed		0.180		0.120		0.0285	
		When executed	In conductive status						
			In non-conductive status						

A

Category	Instruction	Condition (device)		Processing time (μs)					
				L02SCPU, L02SCPU-P		L02CPU, L02CPU-P		L06CPU, L06CPU-P, L26CPU, L26CPU-P, L26CPU-BT, L26CPU-PBT	
				Min.	Max.	Min.	Max.	Min.	Max.
Basic instruction	LDD<=	In conductive status		0.180		0.120		0.0285	
		In non-conductive status							
	ANDD<=	When not executed		0.180		0.120		0.0285	
		When executed	In conductive status						
			In non-conductive status						
	ORD<=	When not executed		0.180		0.120		0.0285	
		When executed	In conductive status						
			In non-conductive status						
	LDD<	In conductive status		0.180		0.120		0.0285	
		In non-conductive status							
	ANDD<	When not executed		0.180		0.120		0.0285	
		When executed	In conductive status						
			In non-conductive status						
	ORD<	When not executed		0.180		0.120		0.0285	
		When executed	In conductive status						
			In non-conductive status						
	LDD>=	In conductive status		0.180		0.120		0.0285	
		In non-conductive status							
	ANDD>=	When not executed		0.180		0.120		0.0285	
		When executed	In conductive status						
			In non-conductive status						
	ORD>=	When not executed		0.180		0.120		0.0285	
		When executed	In conductive status						
			In non-conductive status						
	+ (S) (D)	When executed		0.180		0.120		0.0285	
	+ (S1) (S2) (D)	When executed		0.240		0.160		0.038	
	- (S) (D)	When executed		0.180		0.120		0.0285	
	- (S1) (S2) (D)	When executed		0.240		0.160		0.038	
	D+ (S) (D)	When executed		0.180		0.120		0.0285	
	D+ (S1) (S2) (D)	When executed		0.240		0.160		0.038	
	D- (S) (D)	When executed		0.180		0.120		0.0285	
	D- (S1) (S2) (D)	When executed		0.240		0.160		0.038	
	* (S1) (S2) (D)	When executed		0.240		0.180		0.057	
	/ (S1) (S2) (D)	When executed		0.340		0.280		0.105	
	D* (S1) (S2) (D)	When executed		0.320		0.260		0.095	
	D/ (S1) (S2) (D)	When executed		0.460		0.400		0.162	
	B+ (S) (D)	When executed		3.100	12.300	3.100	6.800	2.900	4.100
	B+ (S1) (S2) (D)	When executed		5.900	13.500	4.800	8.900	4.200	5.900
	B- (S) (D)	When executed		3.100	12.300	3.100	6.800	2.900	4.100

Category	Instruction	Condition (device)			Processing time (μs)					
					L02SCPU, L02SCPU-P		L02CPU, L02CPU-P		L06CPU, L06CPU-P, L26CPU, L26CPU-P, L26CPU-BT, L26CPU-PBT	
					Min.	Max.	Min.	Max.	Min.	Max.
Basic instruction	B- (S1) (S2) (D)	When executed			5.900	13.600	4.800	8.900	4.200	4.600
	B* (S1) (S2) (D)	When executed			3.700	12.100	3.900	7.400	3.400	4.800
	B/ (S1) (S2) (D)	When executed			4.000	14.000	3.900	8.500	3.700	5.200
	E+ (S) (D)	Single precision	(S)=0, (D)=0		0.240		0.180		0.057	
			(S)=2^{127}, (D)=2^{127}		0.240		0.180		0.057	
	E+ (S1) (S2) (D)	Single precision	(S1)=0, (S2)=0		0.300		0.220		0.0665	
			(S1)=2^{127}, (S2)=2^{127}		0.300		0.220		0.0665	
	E- (S) (D)	Single precision	(S)=0, (D)=0		0.240		0.180		0.057	
			(S)=2^{127}, (D)=2^{127}		0.240		0.180		0.057	
	E- (S1) (S2) (D)	Single precision	(S1)=0, (S2)=0		0.300		0.220		0.0665	
			(S1)=2^{127}, (S2)=2^{127}		0.300		0.220		0.0665	
	E* (S1) (S2) (D)	Single precision	(S1)=0, (S2)=0		0.240		0.180		0.057	
			(S1)=2^{127}, (S2)=2^{127}		0.240		0.180		0.057	
	E/ (S1) (S2) (D)	Single precision	(S1)=2^{127}, (S2)=2^{127}		4.900	18.900	3.900	8.500	0.285*1	
	INC	When executed			0.120		0.080		0.019	
	DINC	When executed			0.120		0.080		0.019	
	DEC	When executed			0.120		0.080		0.019	
	DDEC	When executed			0.120		0.080		0.019	
	BCD	When executed			0.200		0.160		0.057	
	DBCD	When executed			0.280		0.240		0.095	
	BIN	When executed			0.140		0.100		0.0285	
	DBIN	When executed			0.140		0.100		0.0285	
	FLT	Single precision	(S)=0		0.180		0.100		0.0475	
			(S)=7FFFH		0.180		0.140		0.0475	
	DFLT	Single precision	(S)=0		0.180		0.140		0.0475	
			(S)=7FFFFFFFH		0.180		0.140		0.0475	
	INT	Single precision	(S)=0		0.180		0.140		0.0475	
			(S)=32766.5		0.180		0.140		0.0475	
	DINT	Single precision	(S)=0		0.180		0.140		0.0475	
			(S)=1234567890.3		0.180		0.140		0.0475	
	MOV	—			0.120		0.080		0.019	
	DMOV	—			0.120		0.080		0.019	
	EMOV	—			0.120		0.080		0.019	
	CML	—			0.120		0.080		0.019	
	DCML	—			0.120		0.080		0.019	
	BMOV	SM237=ON		n=1	4.200	4.600	3.600	4.100	2.900	3.200
				n=96	4.850	5.150	4.500	4.700	3.400	3.700
		SM237=OFF		n=1	6.800	11.300	5.000	7.400	3.900	5.100
				n=96	7.450	11.900	6.000	7.900	4.400	5.700

A

Category	Instruction	Condition (device)		Processing time (μs)					
				L02SCPU, L02SCPU-P		L02CPU, L02CPU-P		L06CPU, L06CPU-P, L26CPU, L26CPU-P, L26CPU-BT, L26CPU-PBT	
				Min.	Max.	Min.	Max.	Min.	Max.
Basic instruction	FMOV	SM237=ON	n=1	4.100	4.600	5.900	6.800	2.800	3.200
			n=96	4.800	5.200	6.300	11.000	3.000	5.200
		SM237=OFF	n=1	4.600	8.250	7.000	8.000	3.400	3.800
			n=96	6.150	10.600	5.200	6.900	3.600	5.800
	XCH	—		2.250	8.100	2.100	4.100	1.800	2.300
	DXCH	—		2.400	8.200	2.200	4.200	2.100	2.900
	DFMOV	SM237=ON	n=1	2.700	2.800	2.000	3.200	1.750	1.750
			n=96	6.500	6.800	5.600	6.100	3.650	4.150
		SM237=OFF	n=1	4.000	8.150	2.900	4.600	2.250	3.150
			n=96	8.000	12.200	6.100	8.200	4.200	5.500
	CJ	—		3.500	10.100	2.100	2.900	1.100	2.400
	SCJ	—		3.500	10.100	2.100	2.900	1.100	2.400
	JMP	—		3.500	10.100	2.100	2.900	1.100	2.400
Application instruction	WAND (S) (D)	When executed		0.180		0.120		0.0285	
	WAND (S1) (S2) (D)	When executed		0.240		0.160		0.038	
	DAND (S) (D)	When executed		0.180		0.120		0.0285	
	DAND (S1) (S2) (D)	When executed		0.240		0.160		0.038	
	WOR (S) (D)	When executed		0.180		0.120		0.0285	
	WOR (S1) (S2) (D)	When executed		0.240		0.160		0.038	
	DOR (S) (D)	When executed		0.180		0.120		0.0285	
	DOR (S1) (S2) (D)	When executed		0.240		0.160		0.038	
	WXOR (S) (D)	When executed		0.180		0.120		0.0285	
	WXOR (S1) (S2) (D)	When executed		0.240		0.160		0.038	
	DXOR (S) (D)	When executed		0.180		0.120		0.0285	
	DXOR (S1) (S2) (D)	When executed		0.240		0.160		0.038	
	WXNR (S) (D)	When executed		0.180		0.120		0.0285	
	WXNR (S1) (S2) (D)	When executed		0.240		0.160		0.038	
	DXNR (S) (D)	When executed		0.180		0.120		0.0285	
	DXNR (S1) (S2) (D)	When executed		0.240		0.160		0.038	
	ROR (D) n	n=1		2.250	10.800	2.200	4.900	1.700	2.500
		n=15		2.350	10.800	2.200	4.900	1.700	2.500
	RCR (D) n	n=1		2.250	10.800	2.100	4.800	1.700	3.200
		n=15		2.250	10.800	2.100	4.800	1.700	3.200
	ROL (D) n	n=1		2.350	10.800	2.100	4.800	1.800	3.200
		n=15		2.350	10.800	2.100	4.800	1.800	3.200
	RCL (D) n	n=1		2.300	11.500	2.100	5.200	1.800	2.200
		n=15		2.300	11.500	2.100	5.200	1.800	2.200
	DROR (D) n	n=1		2.350	11.500	2.200	5.200	1.900	2.700
		n=31		2.350	11.500	2.200	5.200	1.900	2.700
	DRCR (D) n	n=1		2.350	13.300	2.200	5.900	1.900	4.200
		n=31		2.350	14.900	2.200	5.900	1.900	4.200

Category	Instruction	Condition (device)	Processing time (μs)					
			L02SCPU, L02SCPU-P		L02CPU, L02CPU-P		L06CPU, L06CPU-P, L26CPU, L26CPU-P, L26CPU-BT, L26CPU-PBT	
			Min.	Max.	Min.	Max.	Min.	Max.
Application instruction	DROL (D) n	n=1	2.350	10.800	2.200	4.900	1.800	3.300
		n=31	2.350	10.800	2.200	4.900	1.800	3.300
	DRCL (D) n	n=1	2.350	13.300	2.200	5.900	1.900	3.800
		n=31	2.350	13.300	2.200	5.900	1.900	3.800
	SFR (D) n	n=1	2.350	9.900	2.200	4.600	1.700	2.600
		n=15	2.350	9.900	2.200	4.600	1.700	2.600
	SFL (D) n	n=1	2.350	9.850	2.200	4.600	1.800	2.700
		n=15	2.350	9.850	2.200	4.600	1.800	2.700
	DSFR (D) n	n=1	3.250	15.500	2.200	6.100	2.200	4.300
		n=96	32.600	45.000	33.400	38.100	23.900	26.100
	DSFL (D) n	n=1	3.200	15.500	2.200	6.100	2.100	4.000
		n=96	32.600	45.100	33.500	38.000	23.700	25.800
	SUM	(S)=0	3.100	8.950	3.000	4.800	2.900	3.600
		(S)=FFFFH	3.000	8.850	3.000	4.900	2.900	3.600
	SEG	When executed	2.100	7.700	1.700	3.600	1.500	2.100
	FOR	—	1.500	7.500	1.300	3.200	0.870	2.100
	CALL Pn	Internal file pointer	4.800	5.400	2.600	4.000	2.300	3.600
		Common pointer	7.100	30.500	4.600	13.500	3.200	4.900
	CALL Pn (S1) to (S5)	—	50.200	62.000	31.200	36.000	26.100	29.300

*1 For the L06CPU and L06CPU-P, the minimum value is 3.900μs and the maximum value is 4.900μs.

Point

For the instructions for which a rising edge instruction (□P) is not described, the processing time is the same as an ON execution instruction.
(Example) MOVP instruction, WANDP instruction etc.

A

■Table of the time to be added when file register, extended data register, and extended link register are used

• When using L02SCPU, L02SCPU-P, L02CPU, L02CPU-P, L06CPU, L06CPU-P, L26CPU, L26CPU-P, L26CPU-BT, and L26CPU-PBT

Device name		Data	Device specification location	Addition time (μs)		
				L02SCPU, L02SCPU-P	L02CPU, L02CPU-P	L06CPU, L06CPU-P, L26CPU, L26CPU-P, L26CPU-BT, L26CPU-PBT
File register (R)	When standard RAM is used	Bit	Source	0.100	0.100	0.048
			Destination	0.220	0.220	0.038
		Word	Source	0.100	0.100	0.048
			Destination	0.100	0.100	0.038
		Double word	Source	0.200	0.200	0.095
			Destination	0.200	0.200	0.086
File register (ZR), extended data register (D), extended link register (W)	When standard RAM is used	Bit	Source	0.160	0.140	0.057
			Destination	0.320	0.280	0.048
		Word	Source	0.160	0.140	0.057
			Destination	0.160	0.140	0.048
		Double word	Source	0.260	0.240	0.105
			Destination	0.260	0.240	0.095

■Table of the time to be added when F/T(ST)/C device is used in OUT/SET/RST instruction

• When using L02SCPU, L02SCPU-P, L02CPU, L02CPU-P, L06CPU, L06CPU-P, L26CPU, L26CPU-P, L26CPU-BT, and L26CPU-PBT

Instruction name	Device name	Condition		Addition time (μs)		
				L02SCPU, L02SCPU-P	L02CPU, L02CPU-P	L06CPU, L06CPU-P, L26CPU, L26CPU-P, L26CPU-BT, L26CPU-PBT
OUT	F	When not executed		2.900	2.000	1.570
		When executed		116.000	53.100	38.090
	T(ST), C	When not executed		0.180	0.120	0.030
		When executed	After time up	0.180	0.120	0.030
			When added	0.180	0.120	0.030
SET	F	When not executed		0.060	0.040	0.010
		When executed		116.000	52.000	40.600
RST	F	When not executed		0.060	0.040	0.010
		When executed		55.800	45.700	36.600
	T(ST), C	When not executed		0.180	0.120	0.030
		When executed		0.180	0.120	0.030

Processing time of instructions other than subset instruction

The following table shows the processing time of instructions other than subset instructions.

Point

- The processing time shown in "Table of the processing time of instructions other than subset instructions" applies when the device used in an instruction does not meet the device condition for subset processing. (For device condition that does not trigger subset processing, refer to Page 109 Subset processing.) For instructions not shown in the following table, refer to Page 1035 Subset instruction processing time.
- When using the file register (R, ZR), extended data register (D), extended link register (W), module access device (Un\G□), or link direct device (Jn\□), add the processing time shown in Page 1073 Table of the time to be added when file register, extended data register, extended link register, module access device, and link direct device are used to that of each instruction.
- Since the processing time of an instruction varies depending on that of the cash function, both the minimum and maximum values are described in the table.

■Table of the processing time of instructions other than subset instructions

- When using L02SCPU, L02SCPU-P, L02CPU, L02CPU-P, L06CPU, L06CPU-P, L26CPU, L26CPU-P, L26CPU-BT, and L26CPU-PBT

Category	Instruction	Condition (device)	Processing time (μs)					
			L02SCPU, L02SCPU-P		L02CPU, L02CPU-P		L06CPU, L06CPU-P, L26CPU, L26CPU-P, L26CPU-BT, L26CPU-PBT	
			Min.	Max.	Min.	Max.	Min.	Max.
Sequence instruction	ANB ORB MPS MRD MPP	—	0.060		0.040		0.0095	
	INV	When not executed	0.060		0.040		0.0095	
		When executed						
	MEP MEF	When not executed	0.060		0.040		0.0095	
		When executed						
	EGP EGF	When not executed	0.060		0.040		0.0095	
		When executed						
	PLS	—	1.800	1.900	1.600	1.700	0.890	1.200
	PLF	—	1.800	1.900	1.600	1.700	0.890	1.200
	FF	When not executed	0.120		0.080		0.0185	
		When executed	1.700	1.800	1.500	1.500	0.790	0.910
	DELTA	When not executed	0.120		0.080		0.0185	
		When executed	4.000	14.700	2.700	6.800	2.400	3.200
	SFT	When not executed	0.120		0.080		0.0185	
		When executed	1.800	12.600	1.700	4.300	1.100	2.700
	MC	—	0.120		0.080		0.0185	
	MCR	—	0.060		0.040		0.0185	
	FEND END	Error check performed	280.000		220.000		180.000	
		No error check performed						
	STOP	—	—		—		—	
	NOP NOPLF PAGE	—	0.060		0.040		0.0095	

Category	Instruction	Condition (device)			Processing time (μs)					
					L02SCPU, L02SCPU-P		L02CPU, L02CPU-P		L06CPU, L06CPU-P, L26CPU, L26CPU-P, L26CPU-BT, L26CPU-PBT	
					Min.	Max.	Min.	Max.	Min.	Max.
Basic instruction	LDE=	Single precision	In conductive status		4.400	20.900	3.900	10.000	0.0285	
			In non-conductive status		4.400	20.900	3.900	10.000	0.0285	
	ANDE=	Single precision	When not executed		0.180		0.120		0.0285	
			When executed	In conductive status	4.200	19.600	3.400	9.300	0.0285	
				In non-conductive status	4.200	19.600	3.400	9.300	0.0285	
	ORE=	Single precision	When not executed		0.180		0.120		0.0285	
			When executed	In conductive status	4.200	17.400	3.500	8.500	0.0285	
				In non-conductive status	4.200	17.400	3.500	8.500	0.0285	
	LDE<>	Single precision	In conductive status		4.400	20.900	3.900	10.000	0.0285	
			In non-conductive status		4.400	20.900	3.900	10.000	0.0285	
	ANDE<>	Single precision	When not executed		0.180		0.120		0.0285	
			When executed	In conductive status	4.200	19.600	3.400	9.300	0.0285	
				In non-conductive status	4.200	19.600	3.400	9.300	0.0285	
	ORE<>	Single precision	When not executed		0.180		0.120		0.0285	
			When executed	In conductive status	4.200	17.400	3.500	8.500	0.0285	
				In non-conductive status	4.200	17.400	3.500	8.500	0.0285	
	LDE>	Single precision	In conductive status		4.400	20.900	3.900	10.000	0.0285	
			In non-conductive status		4.400	20.900	3.900	10.000	0.0285	
	ANDE>	Single precision	When not executed		0.180		0.120		0.0285	
			When executed	In conductive status	4.200	19.600	3.400	9.300	0.0285	
				In non-conductive status	4.200	19.600	3.400	9.300	0.0285	
	ORE>	Single precision	When not executed		0.180		0.120		0.0285	
			When executed	In conductive status	4.200	17.400	3.500	8.500	0.0285	
				In non-conductive status	4.200	17.400	3.500	8.500	0.0285	
	LDE<=	Single precision	In conductive status		4.400	20.900	3.900	10.000	0.0285	
			In non-conductive status		4.400	20.900	3.900	10.000	0.0285	

Category	Instruction	Condition (device)			Processing time (μs)					
					L02SCPU, L02SCPU-P		L02CPU, L02CPU-P		L06CPU, L06CPU-P, L26CPU, L26CPU-P, L26CPU-BT, L26CPU-PBT	
					Min.	Max.	Min.	Max.	Min.	Max.
Basic instruction	ANDE<=	Single precision	When not executed		0.180		0.120		0.0285	
			When executed	In conductive status	4.200	19.600	3.400	9.300	0.0285	
				In non-conductive status	4.200	19.600	3.400	9.300	0.0285	
	ORE<=	Single precision	When not executed		0.180		0.120		0.0285	
			When executed	In conductive status	4.200	17.400	3.500	8.500	0.0285	
				In non-conductive status	4.200	17.400	3.500	8.500	0.0285	
	LDE<	Single precision	In conductive status		4.400	20.900	3.900	10.000	0.0285	
			In non-conductive status		4.400	20.900	3.900	10.000	0.0285	
	ANDE<	Single precision	When not executed		0.180		0.120		0.0285	
			When executed	In conductive status	4.200	19.600	3.400	9.300	0.0285	
				In non-conductive status	4.200	19.600	3.400	9.300	0.0285	
	ORE<	Single precision	When not executed		0.180		0.120		0.0285	
			When executed	In conductive status	4.200	17.400	3.500	8.500	0.0285	
				In non-conductive status	4.200	17.400	3.500	8.500	0.0285	
	LDE>=	Single precision	In conductive status		4.400	20.900	3.900	10.000	0.0285	
			In non-conductive status		4.400	20.900	3.900	10.000	0.0285	
	ANDE>=	Single precision	When not executed		0.180		0.120		0.0285	
			When executed	In conductive status	4.200	19.600	3.400	9.300	0.0285	
				In non-conductive status	4.200	19.600	3.400	9.300	0.0285	
	ORE>=	Single precision	When not executed		0.180		0.120		0.0285	
			When executed	In conductive status	4.200	17.400	3.500	8.500	0.0285	
				In non-conductive status	4.200	17.400	3.500	8.500	0.0285	
	LDED=	Double precision	In conductive status		4.700	37.400	4.800	16.000	3.500	9.000
			In non-conductive status		4.700	37.400	4.800	16.000	3.500	9.000

A

Category	Instruction	Condition (device)			Processing time (μs)					
					L02SCPU, L02SCPU-P		L02CPU, L02CPU-P		L06CPU, L06CPU-P, L26CPU, L26CPU-P, L26CPU-BT, L26CPU-PBT	
					Min.	Max.	Min.	Max.	Min.	Max.
Basic instruction	ANDED=	Double precision	When not executed		0.180		0.120		0.0285	
			When executed	In conductive status	4.500	34.700	4.400	15.100	3.200	7.500
				In non-conductive status	4.500	34.700	4.400	15.100	3.200	7.500
	ORED=	Double precision	When not executed		0.180		0.120		0.0285	
			When executed	In conductive status	4.700	33.200	4.500	14.900	3.400	9.200
				In non-conductive status	4.700	33.200	4.500	14.900	3.400	9.200
	LDED<>	Double precision	In conductive status		4.700	37.400	4.800	16.000	3.500	9.000
			In non-conductive status		4.700	37.400	4.800	16.000	3.500	9.000
	ANDED<>	Double precision	When not executed		0.180		0.120		0.0285	
			When executed	In conductive status	4.500	34.700	4.400	15.100	3.200	7.500
				In non-conductive status	4.500	34.700	4.400	15.100	3.200	7.500
	ORED<>	Double precision	When not executed		0.180		0.120		0.0285	
			When executed	In conductive status	4.700	33.200	4.500	14.900	3.400	9.200
				In non-conductive status	4.700	33.200	4.500	14.900	3.400	9.200
	LDED>	Double precision	In conductive status		4.700	37.400	4.800	16.000	3.500	9.000
			In non-conductive status		4.700	37.400	4.800	16.000	3.500	9.000
	ANDED>	Double precision	When not executed		0.180		0.120		0.0285	
			When executed	In conductive status	4.500	34.700	4.400	15.100	3.200	7.500
				In non-conductive status	4.500	34.700	4.400	15.100	3.200	7.500
	ORED>	Double precision	When not executed		0.180		0.120		0.0285	
			When executed	In conductive status	4.700	33.200	4.500	14.900	3.400	9.200
				In non-conductive status	4.700	33.200	4.500	14.900	3.400	9.200
	LDED<=	Double precision	In conductive status		4.700	37.400	4.800	16.000	3.500	9.000
			In non-conductive status		4.700	37.400	4.800	16.000	3.500	9.000

Category	Instruction	Condition (device)			Processing time (μs)					
					L02SCPU, L02SCPU-P		L02CPU, L02CPU-P		L06CPU, L06CPU-P, L26CPU, L26CPU-P, L26CPU-BT, L26CPU-PBT	
					Min.	Max.	Min.	Max.	Min.	Max.
Basic instruction	ANDED<=	Double precision	When not executed		0.180		0.120		0.0285	
			When executed	In conductive status	4.500	34.700	4.400	15.100	3.200	7.500
				In non-conductive status	4.500	34.700	4.400	15.100	3.200	7.500
	ORED<=	Double precision	When not executed		0.180		0.120		0.0285	
			When executed	In conductive status	4.700	33.200	4.500	14.900	3.400	9.200
				In non-conductive status	4.700	33.200	4.500	14.900	3.400	9.200
	LDED>=	Double precision	In conductive status		4.700	37.400	4.800	16.000	3.500	9.000
			In non-conductive status		4.700	37.400	4.800	16.000	3.500	9.000
	ANDED>=	Double precision	When not executed		0.180		0.120		0.0285	
			When executed	In conductive status	4.500	34.700	4.400	15.100	3.200	7.500
				In non-conductive status	4.500	34.700	4.400	15.100	3.200	7.500
	ORED>=	Double precision	When not executed		0.180		0.120		0.0285	
			When executed	In conductive status	4.700	33.200	4.500	14.900	3.400	9.200
				In non-conductive status	4.700	33.200	4.500	14.900	3.400	9.200
	LDED>=	Double precision	In conductive status		4.700	37.400	4.800	16.000	3.500	9.000
			In non-conductive status		4.700	37.400	4.800	16.000	3.500	9.000
	ANDED>=	Double precision	When not executed		0.180		0.120		0.0285	
			When executed	In conductive status	4.500	34.700	4.400	15.100	3.200	7.500
				In non-conductive status	4.500	34.700	4.400	15.100	3.200	7.500
	ORED>=	Double precision	When not executed		0.180		0.120		0.0285	
			When executed	In conductive status	4.700	33.200	4.500	14.900	3.400	9.200
				In non-conductive status	4.700	33.200	4.500	14.900	3.400	9.200
	LD$=	In conductive status			8.300	38.500	5.600	17.100	4.200	8.200
		In non-conductive status			8.300	38.500	5.600	17.100	4.200	8.200

A

Category	Instruction	Condition (device)		Processing time (μs)					
				L02SCPU, L02SCPU-P		L02CPU, L02CPU-P		L06CPU, L06CPU-P, L26CPU, L26CPU-P, L26CPU-BT, L26CPU-PBT	
				Min.	Max.	Min.	Max.	Min.	Max.
Basic instruction	AND$=	When not executed		0.180		0.120		0.0285	
		When executed	In conductive status	7.200	37.300	5.300	16.400	3.900	7.300
			In non-conductive status	7.200	37.300	5.300	16.400	3.900	7.300
	OR$=	When not executed		0.180		0.120		0.0285	
		When executed	In conductive status	7.500	36.600	5.200	15.700	4.000	7.600
			In non-conductive status	7.500	36.600	5.200	15.700	4.000	7.600
	LD$<>	In conductive status		8.300	39.300	5.600	17.100	4.200	8.200
		In non-conductive status		8.300	39.300	5.600	17.100	4.200	8.200
	AND$<>	When not executed		0.180		0.120		0.0285	
		When executed	In conductive status	8.000	38.200	5.300	16.400	3.900	7.300
			In non-conductive status	8.000	38.200	5.300	16.400	3.900	7.300
	OR$<>	When not executed		0.180		0.120		0.0285	
		When executed	In conductive status	8.300	37.300	5.200	15.700	4.000	7.600
			In non-conductive status	8.300	37.300	5.200	15.700	4.000	7.600
	LD$>	In conductive status		8.300	41.600	5.600	17.100	4.200	8.200
		In non-conductive status		8.300	41.600	5.600	17.100	4.200	8.200
	AND$>	When not executed		0.180		0.120		0.0285	
		When executed	In conductive status	8.000	38.100	5.300	16.400	3.900	7.300
			In non-conductive status	8.000	38.100	5.300	16.400	3.900	7.300
	OR$>	When not executed		0.180		0.120		0.0285	
		When executed	In conductive status	8.200	35.700	5.200	15.700	4.000	7.600
			In non-conductive status	8.200	35.700	5.200	15.700	4.000	7.600
	LD$<=	In conductive status		8.300	39.200	5.600	17.100	4.200	8.200
		In non-conductive status		8.300	39.200	5.600	17.100	4.200	8.200

Category	Instruction	Condition (device)		Processing time (μs)					
				L02SCPU, L02SCPU-P		L02CPU, L02CPU-P		L06CPU, L06CPU-P, L26CPU, L26CPU-P, L26CPU-BT, L26CPU-PBT	
				Min.	Max.	Min.	Max.	Min.	Max.
Basic instruction	AND$<=	When not executed		0.180		0.120		0.0285	
		When executed	In conductive status	7.100	36.500	5.300	16.400	3.900	7.300
			In non-conductive status	7.100	36.500	5.300	16.400	3.900	7.300
	OR$<=	When not executed		0.180		0.120		0.0285	
		When executed	In conductive status	7.400	35.600	5.200	15.700	4.000	7.600
			In non-conductive status	7.400	35.600	5.200	15.700	4.000	7.600
	LD$<	In conductive status		7.400	40.000	5.600	17.100	4.200	8.200
		In non-conductive status		7.400	40.000	5.600	17.100	4.200	8.200
	AND$<	When not executed		0.180		0.120		0.0285	
		When executed	In conductive status	8.000	37.300	5.300	16.400	3.900	7.300
			In non-conductive status	8.000	37.300	5.300	16.400	3.900	7.300
	OR$<	When not executed		0.180		0.120		0.0285	
		When executed	In conductive status	8.300	35.600	5.200	15.700	4.000	7.600
			In non-conductive status	8.300	35.600	5.200	15.700	4.000	7.600
	LD$>=	In conductive status		7.400	38.300	5.600	17.100	4.200	8.200
		In non-conductive status		7.400	38.300	5.600	17.100	4.200	8.200
	AND$>=	When not executed		0.180		0.120		0.0285	
		When executed	In conductive status	7.200	37.300	5.300	16.400	3.900	7.300
			In non-conductive status	7.200	37.300	5.300	16.400	3.900	7.300
	OR$>=	When not executed		0.180		0.120		0.0285	
		When executed	In conductive status	8.200	36.400	5.200	15.700	4.000	7.600
			In non-conductive status	8.200	8.200	5.200	15.700	4.000	7.600

Category	Instruction	Condition (device)	Processing time (μs)					
			L02SCPU, L02SCPU-P		L02CPU, L02CPU-P		L06CPU, L06CPU-P, L26CPU, L26CPU-P, L26CPU-BT, L26CPU-PBT	
			Min.	Max.	Min.	Max.	Min.	Max.
Basic instruction	BKCMP= (S1) (S2) (D) n	n=1	15.300	36.100	9.200	15.600	7.500	10.100
		n=96	64.500	85.500	60.700	69.100	45.600	50.500
	BKCMP<> (S1) (S2) (D) n	n=1	15.300	36.100	9.200	15.600	7.500	10.100
		n=96	66.600	87.500	60.700	69.100	45.600	50.500
	BKCMP> (S1) (S2) (D) n	n=1	15.300	36.100	9.200	15.600	7.500	10.100
		n=96	66.600	87.500	60.700	69.100	45.600	50.500
	BKCMP<= (S1) (S2) (D) n	n=1	15.300	36.100	9.200	15.600	7.500	10.100
		n=96	64.500	85.500	60.700	69.100	45.600	50.500
	BKCMP< (S1) (S2) (D) n	n=1	15.300	36.100	9.200	15.600	7.500	10.100
		n=96	66.600	87.500	60.700	69.100	45.600	50.500
	BKCMP>= (S1) (S2) (D) n	n=1	15.300	36.100	9.200	15.600	7.500	10.100
		n=96	64.500	85.500	60.700	69.100	45.600	50.500
	DBKCMP= (S1) (S2) (D) n	n=1	15.800	36.300	9.700	16.400	8.600	13.000
		n=96	64.900	85.700	61.200	69.900	47.900	52.800
	DBKCMP<> (S1) (S2) (D) n	n=1	15.700	36.300	9.700	16.400	8.600	13.000
		n=96	67.000	87.700	61.200	69.900	47.900	52.800
	DBKCMP> (S1) (S2) (D) n	n=1	15.800	36.300	9.700	16.400	8.600	13.000
		n=96	67.000	87.700	61.200	69.900	47.900	52.800
	DBKCMP<= (S1) (S2) (D) n	n=1	15.700	36.300	9.700	16.400	8.600	13.000
		n=96	64.800	85.700	61.200	69.900	47.900	52.800
	DBKCMP< (S1) (S2) (D) n	n=1	15.800	36.300	9.700	16.400	8.600	13.000
		n=96	67.000	87.700	61.200	69.900	47.900	52.800
	DBKCMP>= (S1) (S2) (D) n	n=1	15.700	36.300	9.700	16.400	8.600	13.000
		n=96	64.800	85.700	61.200	69.900	47.900	52.800
	CMP	—	4.700	23.900	4.600	11.300	4.100	9.200
	CMPP	—	4.700	23.900	4.600	11.300	4.100	9.200
	DCMP	—	4.700	23.900	4.600	11.300	4.100	9.200
	DCMPP	—	4.700	23.900	4.600	11.300	4.100	9.200
	ZCP	—	5.500	26.500	5.400	12.800	4.800	10.300
	ZCPP	—	5.500	26.500	5.400	12.800	4.800	10.300
	DZCP	—	5.500	26.500	5.400	12.800	4.800	10.300
	DZCPP	—	5.500	26.500	5.400	12.800	4.800	10.300
	ECMP	—	6.500	29.600	5.500	14.800	4.500	11.800
	ECMPP	—	6.500	29.600	5.500	14.800	4.500	11.800
	EDCMP	—	7.400	42.500	6.200	19.200	5.400	15.000
	EDCMPP	—	7.400	42.500	6.200	19.200	5.400	15.000
	EZCP	—	7.900	33.000	6.600	16.900	5.800	13.300
	EZCPP	—	7.900	33.000	6.600	16.900	5.800	13.300
	EDZCP	—	9.600	48.500	8.400	22.700	7.000	17.600
	EDZCPP	—	9.600	48.500	8.400	22.700	7.000	17.600

Category	Instruction	Condition (device)		Processing time (μs)					
				L02SCPU, L02SCPU-P		L02CPU, L02CPU-P		L06CPU, L06CPU-P, L26CPU, L26CPU-P, L26CPU-BT, L26CPU-PBT	
				Min.	Max.	Min.	Max.	Min.	Max.
Basic instruction	DB+ (S) (D)	When executed		5.750	13.300	4.800	8.400	4.600	6.400
	DB+ (S1) (S2) (D)	When executed		5.650	13.200	5.100	8.700	4.800	6.700
	DB- (S) (D)	When executed		5.750	12.700	4.800	8.400	4.600	6.400
	DB- (S1) (S2) (D)	When executed		5.650	12.600	5.100	8.700	4.800	6.700
	DB* (S1) (S2) (D)	When executed		8.750	40.200	8.700	18.900	8.100	11.600
	DB/ (S1) (S2) (D)	When executed		5.750	21.500	6.100	9.100	5.800	8.800
	ED+ (S) (D)	Double precision	(S)=0, (D)=0	4.500	26.700	4.800	8.000	4.300	7.200
			(S)=2^{1023}, (D)=2^{1023}	5.800	32.900	5.400	14.900	4.300	7.200
	ED+ (S1) (S2) (D)	Double precision	(S1)=0, (S2)=0	5.450	35.400	5.500	9.800	4.800	9.200
			(S1)=2^{1023}, (S2)=2^{1023}	6.750	41.400	6.100	17.800	4.800	9.200
	ED- (S) (D)	Double precision	(S)=0, (D)=0	5.200	25.900	4.400	10.800	4.400	7.500
			(S)=2^{1023}, (D)=2^{1023}	6.000	27.700	5.400	15.500	4.400	7.500
	ED- (S1) (S2) (D)	Double precision	(S1)=0, (S2)=0	5.550	32.900	4.700	13.900	3.800	7.500
			(S1)=2^{1023}, (S2)=2^{1023}	5.750	33.900	5.700	17.200	3.800	7.500
	ED* (S1) (S2) (D)	Double precision	(S1)=0, (S2)=0	5.500	34.400	5.800	9.500	5.100	8.800
			(S1)=2^{1023}, (S2)=2^{1023}	5.950	39.100	5.900	17.600	5.100	8.800
	ED/ (S1) (S2) (D)	Double precision	(S1)=2^{1023}, (S2)=2^{1023}	8.050	44.200	7.300	18.700	5.900	10.000
	BK+ (S1) (S2) (D) n	n=1		13.500	28.500	9.100	11.200	8.500	10.600
		n=96		63.100	78.200	60.500	66.200	44.600	47.900
	BK- (S1) (S2) (D) n	n=1		13.500	28.500	9.700	12.000	8.900	11.300
		n=96		63.100	78.200	60.500	66.200	44.600	47.900
	DBK+ (S1) (S2) (D) n	n=1		10.100	24.200	7.500	12.400	6.450	9.950
		n=96		59.800	73.900	59.900	65.200	43.700	47.500
	DBK- (S1) (S2) (D) n	n=1		10.100	24.200	7.500	12.400	6.450	9.950
		n=96		59.800	73.900	59.900	65.200	43.700	47.500
	$+ (S) (D)	—		15.400	64.300	11.200	24.700	8.100	13.900
	$+ (S1) (S2) (D)	—		19.700	71.000	7.900	16.600	6.500	10.300
	FLTD	Double precision	(S)=0	3.100	19.600	2.800	9.400	1.800	4.700
			(S)=7FFFH	3.350	19.900	3.300	9.600	2.200	4.800
	DFLTD	Double precision	(S)=0	3.200	20.400	2.900	9.100	2.000	4.900
			(S)=7FFFFFFFH	3.450	20.500	3.400	9.300	2.300	5.100
	INTD	Double precision	(S)=0	3.200	22.900	3.500	8.700	2.200	4.100
			(S)=32766.5	4.100	34.300	4.100	12.900	3.200	5.600
	DINTD	Double precision	(S)=0	3.200	23.000	3.200	9.500	2.200	3.400
			(S)=1234567890.3	4.050	33.500	4.100	13.400	3.000	5.100

A

Category	Instruction	Condition (device)	Processing time (μs)					
			L02SCPU, L02SCPU-P		L02CPU, L02CPU-P		L06CPU, L06CPU-P, L26CPU, L26CPU-P, L26CPU-BT, L26CPU-PBT	
			Min.	Max.	Min.	Max.	Min.	Max.
Basic instruction	DBL	When executed	3.300	5.900	2.500	4.400	2.300	2.700
	WORD	When executed	3.000	7.250	2.800	3.900	2.600	3.600
	GRY	When executed	3.350	7.500	2.700	4.300	2.300	3.000
	DGRY	When executed	3.000	7.200	2.700	4.300	2.300	3.000
	GBIN	When executed	4.600	9.700	4.000	6.400	3.800	4.300
	DGBIN	When executed	5.550	10.700	5.000	6.900	5.000	5.900
	NEG	When executed	3.300	6.850	2.100	4.400	2.000	3.300
	DNEG	When executed	3.050	5.700	2.500	3.700	2.500	3.300
	ENEG	Floating point = 0	3.100	7.350	2.500	3.300	2.300	2.800
		Floating point = -1.0	3.350	11.700	2.800	5.600	2.500	3.900
	EDNEG	Floating point = 0	3.000	21.200	3.000	8.800	1.800	3.100
		Floating point = -1.0	3.100	22.900	2.700	9.400	1.900	3.000
	BKBCD (S) (D) n	n=1	8.700	27.600	6.000	13.400	5.900	8.200
		n=96	84.200	104.000	83.300	91.400	61.000	63.400
	BKBIN (S) (D) n	n=1	8.450	28.100	6.500	9.800	5.600	9.300
		n=96	56.100	75.800	55.400	62.900	49.200	52.500
	ECON	—	3.100	21.300	3.000	9.800	2.100	4.500
	EDCON	—	5.050	24.000	3.300	10.300	2.500	5.400
	EDMOV	—	2.900	22.900	2.700	8.500	1.700	5.000
	$MOV	Character string to be transferred = 0	6.250	30.100	4.400	12.300	3.400	5.600
		Character string to be transferred = 32	15.500	39.300	14.000	21.900	11.400	13.300
	BXCH (D1) (D2) n	n=1	8.400	20.900	6.200	7.900	5.500	7.300
		n=96	67.100	79.900	67.300	71.400	47.300	49.300
	SWAP	—	3.300	3.550	2.400	2.700	1.900	2.200
	SMOV	—	7.400	31.500	7.300	16.500	7.200	13.200
	GOEND	—	0.550		0.700		0.500	
	DI	—	2.800	8.400	2.100	4.000	1.500	1.800

Category	Instruction	Condition (device)	Processing time (μs)					
			L02SCPU, L02SCPU-P		L02CPU, L02CPU-P		L06CPU, L06CPU-P, L26CPU, L26CPU-P, L26CPU-BT, L26CPU-PBT	
			Min.	Max.	Min.	Max.	Min.	Max.
Application instruction	EI	—	4.300	12.300	3.600	6.300	3.000	3.300
	IMASK	—	12.900	40.600	11.800	20.500	7.200	10.500
	IRET	—	1.000		1.400		1.000	
	RFS X n	n=1	7.500	26.500	5.900	12.500	3.700	5.600
		n=96	11.400	30.400	12.900	19.300	10.700	12.400
	RFS Y n	n=1	7.300	26.300	5.100	11.500	3.400	4.800
		n=96	10.900	29.900	8.600	15.300	8.100	8.900
	UDCNT1	—	1.500	7.100	6.200	16.400	5.100	12.300
	UDCNT2	—	1.500	6.300	6.300	16.800	5.400	12.500
	TTMR	—	5.300	20.900	4.500	9.500	3.400	5.400
	STMR	—	8.900	49.800	7.800	21.400	5.800	12.500
	ROTC	—	52.300	52.600	20.900	21.500	8.000	9.400
	RAMP	—	7.400	30.900	6.700	14.600	5.200	8.400
	SPD	—	1.500	6.300	5.400	14.800	4.900	11.200
	PLSY	—	6.400	7.100	10.500	10.500	7.900	7.900
	PWM	—	3.900	4.600	10.100	10.100	7.500	7.500
	MTR	—	10.100	61.400	14.700	25.100	9.400	10.000
	BKAND (S1) (S2) (D) n	n=1	13.600	28.500	9.000	11.700	8.300	11.000
		n=96	63.200	78.200	60.600	66.400	43.800	47.300
	BKOR (S1) (S2) (D) n	n=1	13.500	28.500	7.900	14.000	7.700	9.500
		n=96	63.100	78.200	60.700	66.500	44.300	45.800
	BKXOR (S1) (S2) (D) n	n=1	13.600	28.300	8.800	13.800	7.300	9.200
		n=96	63.100	78.000	61.300	66.300	43.800	45.800
	BKXNR (S1) (S2) (D) n	n=1	13.500	28.300	8.400	13.900	7.600	8.900
		n=96	63.100	78.000	60.900	66.700	43.900	45.300
	BSFR (D) n	n=1	5.050	21.100	3.600	9.500	3.200	4.800
		n=96	9.000	34.800	6.500	15.900	5.800	7.700
	BSFL (D) n	n=1	4.800	19.100	3.600	9.300	3.400	5.100
		n=96	8.550	34.300	6.300	15.800	6.000	7.900
	SFTBR (D) n1 n2	n1=16, n2=1	10.300	46.500	8.100	21.000	7.500	17.400
		n1=16, n2=15	10.300	46.500	8.100	22.100	7.500	17.300
	SFTBL (D) n1 n2	n1=16, n2=1	10.500	49.800	8.100	21.000	7.500	17.400
		n1=16, n2=15	10.500	49.800	8.100	22.100	7.500	17.300
	SFTWR (D) n1 n2	n1=16, n2=1	7.950	24.000	6.200	13.100	4.500	8.700
		n1=16, n2=15	7.950	24.000	6.100	13.100	4.600	8.800
	SFTWL (D) n1 n2	n1=16, n2=1	8.700	23.600	6.200	13.100	4.500	8.700
		n1=16, n2=15	8.650	23.700	6.100	13.100	4.600	8.800
	SFTR (S) (D) n1 n2	n1=16, n2=1	15.900	79.400	13.900	35.500	12.800	27.600
		n1=16, n2=15	16.300	80.200	14.400	35.600	13.200	27.500

Category	Instruction	Condition (device)		Processing time (μs)					
				L02SCPU, L02SCPU-P		L02CPU, L02CPU-P		L06CPU, L06CPU-P, L26CPU, L26CPU-P, L26CPU-BT, L26CPU-PBT	
				Min.	Max.	Min.	Max.	Min.	Max.
Application instruction	SFTL (S) (D) n1 n2	n1=16, n2=1		15.900	78.300	13.900	35.200	12.600	27.400
		n1=16, n2=15		16.300	78.800	14.200	35.500	13.000	27.700
	WSFR (S) (D) n1 n2	n1=16, n2=1		14.000	51.400	11.700	25.300	9.800	20.400
		n1=16, n2=15		14.400	51.400	12.000	25.500	10.100	21.000
	WSFL (S) (D) n1 n2	n1=16, n2=1		14.200	51.600	11.900	25.600	10.300	20.800
		n1=16, n2=15		14.500	52.200	12.100	25.900	10.300	21.000
	BSET (D) n	n=1		4.550	4.750	2.800	3.100	2.500	2.800
		n=15		4.550	4.750	2.800	3.100	2.500	2.800
	BRST (D) n	n=1		4.600	4.750	2.800	3.100	2.500	2.800
		n=15		4.600	4.750	2.800	3.100	2.500	2.800
	TEST	When executed		7.250	13.200	4.700	6.100	3.700	4.800
	DTEST	When executed		6.950	12.900	4.700	6.100	3.700	4.800
	BKRST (S) n	n=1		7.350	11.600	4.300	5.700	3.700	4.100
		n=96		10.100	22.600	6.200	10.000	5.100	6.000
	SER (S1) (S2) (D) n	n=1	All match	6.650	6.800	4.800	5.300	4.200	4.600
			None match	6.650	6.800	4.700	5.300	4.200	4.600
		n=96	All match	34.000	42.300	33.200	35.900	25.900	26.300
			None match	34.000	42.300	33.200	35.900	25.900	26.300
	DSER (S1) (S2) (D) n	n=1	All match	8.000	16.300	6.500	9.000	5.400	5.700
			None match	8.000	16.300	6.500	9.000	5.500	5.900
		n=96	All match	54.100	62.600	54.800	57.500	41.200	41.800
			None match	54.100	62.600	54.700	57.500	41.200	41.800
	DSUM (S) (D)	(S)=0		4.100	4.200	3.400	3.700	3.200	3.700
		(S)=FFFFFFFFH		4.100	4.200	3.400	3.700	3.200	3.700
	DECO (S) (D) n	n=2		8.850	23.000	6.000	10.700	5.300	6.900
		n=8		13.600	36.600	9.500	16.700	6.800	7.800
	ENCO (S) (D) n	n=2	M1=ON	7.650	11.900	5.400	6.900	4.700	5.100
			M4=ON	7.500	11.700	5.300	6.600	4.600	5.000
		n=8	M1=ON	14.600	27.800	10.700	14.000	9.000	10.000
			M256=ON	10.600	23.700	7.000	11.100	5.100	6.100
	DIS (S) (D) n	n=1		6.500	14.800	4.600	7.000	3.800	4.600
		n=4		6.900	15.200	4.900	7.300	4.000	5.000
	UNI (S) (D) n	n=1		6.800	15.100	5.000	7.300	3.500	4.800
		n=4		7.500	15.900	5.700	8.300	4.000	5.100
	NDIS	When executed		4.750	18.700	11.200	15.200	11.000	13.200
	NUNI	When executed		4.750	18.700	10.600	12.700	7.300	13.200
	WTOB (S) (D) n	n=1		6.600	14.900	5.400	8.100	4.400	5.800
		n=96		37.700	46.100	38.400	40.900	28.200	29.300
	BTOW (S) (D) n	n=1		7.350	15.600	5.300	8.200	4.600	5.500
		n=96		32.100	40.500	31.700	34.200	22.800	23.800
	MAX (S) (D) n	n=1		8.250	24.900	5.400	11.900	4.000	6.100
		n=96		34.200	51.600	34.200	41.100	24.700	27.000

Category	Instruction	Condition (device)	Processing time (μs)					
			L02SCPU, L02SCPU-P		L02CPU, L02CPU-P		L06CPU, L06CPU-P, L26CPU, L26CPU-P, L26CPU-BT, L26CPU-PBT	
			Min.	Max.	Min.	Max.	Min.	Max.
Application instruction	MIN (S) (D) n	n=1	8.250	24.800	6.100	12.000	4.000	6.000
		n=96	34.200	51.600	32.900	39.300	26.500	28.300
	DMAX (S) (D) n	n=1	6.800	34.900	6.000	14.800	4.800	8.100
		n=96	60.300	89.200	61.100	69.500	47.100	49.600
	DMIN (S) (D) n	n=1	7.600	35.700	6.000	14.800	4.300	5.900
		n=96	59.400	90.000	57.000	67.000	45.400	47.400
	SORT (S1) n (S2) (D1) (D2)	n=1, (S2)=1	9.400	28.900	6.800	13.700	5.600	8.800
		n=96, (S2)=16	31.500	74.000	31.300	46.800	24.300	34.400
	DSORT (S1) n (S2) (D1) (D2)	n=1, (S2)=1	9.400	29.000	6.800	14.300	5.600	10.900
		n=96, (S2)=16	37.800	81.000	34.900	49.700	26.200	36.700
	WSUM (S) (D) n	n=1	6.700	15.000	5.000	7.300	4.200	5.500
		n=96	28.900	37.100	28.100	30.700	21.300	22.300
	DWSUM (S) (D) n	n=1	8.600	26.800	6.100	11.300	4.800	6.100
		n=96	56.200	74.700	56.200	62.100	42.700	44.000
	MEAN (S) (D) n	n=1	5.850	19.800	4.400	10.400	3.900	7.800
		n=96	17.300	38.200	16.100	24.500	12.900	18.000
	DMEAN (S) (D) n	n=1	6.900	23.300	6.000	12.500	5.300	9.950
		n=96	29.400	49.900	34.000	42.000	23.000	28.800
	CCD	SM772 = OFF, Number of data blocks = 1	8.600	37.100	7.200	17.600	5.600	14.000
		SM772 = OFF, Number of data blocks = 96	56.300	90.500	47.000	75.600	39.100	63.200
		SM772 = ON, Number of data blocks = 1	7.700	36.200	6.500	17.300	5.400	13.900
		SM772 = ON, Number of data blocks = 96	53.700	86.300	44.700	72.100	37.300	60.200
	CRC	SM772 = OFF, Number of data blocks = 1	8.900	41.400	7.500	19.100	5.900	15.100
		SM772 = OFF, Number of data blocks = 96	76.700	97.000	64.000	77.300	57.900	69.600
		SM772 = ON, Number of data blocks = 1	7.800	39.900	6.800	18.700	5.600	14.800
		SM772 = ON, Number of data blocks = 96	59.700	91.800	59.100	76.000	52.900	63.400
	NEXT	—	1.000	1.100	0.940	1.400	0.770	1.200
	BREAK	—	4.700	25.000	3.500	10.200	3.100	7.600
	RET	Return to original program	4.100	19.500	2.900	8.800	1.600	2.600
		Return to other program	4.700	16.700	3.200	10.500	2.000	3.100
	FCALL Pn	Internal file pointer	5.400	5.400	3.600	3.800	2.700	3.600
		Common pointer	7.600	30.500	5.300	13.500	3.600	5.100
	FCALL Pn (S1) to (S5)	—	50.400	62.700	20.900	30.300	16.500	18.600
	ECALL * Pn *: Program name	—	105.000	214.000	72.700	109.000	65.900	77.600
	ECALL * Pn (S1) to (S5) *: Program name	—	164.000	271.000	101.400	141.400	91.800	105.000

Category	Instruction	Condition (device)	Processing time (μs)					
			L02SCPU, L02SCPU-P		L02CPU, L02CPU-P		L06CPU, L06CPU-P, L26CPU, L26CPU-P, L26CPU-BT, L26CPU-PBT	
			Min.	Max.	Min.	Max.	Min.	Max.
Application instruction	EFCALL * Pn *: Program name	—	105.000	214.000	72.800	109.600	66.200	78.100
	EFCALL * Pn (S1) to (S5) *: Program name	—	164.000	271.000	101.900	141.500	78.800	91.600
	XCALL	—	5.100	6.700	5.200	14.600	3.700	5.200
	CCOM COM	When selecting I/O refresh only	18.100	89.100	8.400	14.600	12.600	17.200
		When selecting CC-Link refresh only (master station side)	33.300	132.000	10.500	29.400	10.100	22.000
		When selecting CC-Link refresh only (local station side)	33.300	132.000	10.500	29.400	10.100	22.000
		When selecting CC-Link IE Field Network refresh only (master station side)	32.000	127.000	17.000	49.500	16.600	38.000
		When selecting CC-Link IE Field Network refresh only (local station side)	32.000	127.000	17.000	49.500	16.600	38.000
		When selecting MELSECNET/H refresh only (control station side)	—	—	—	—	—	—
		When selecting MELSECNET/H refresh only (normal station side)	—	—	—	—	—	—
		When selecting intelli auto refresh only	18.100	89.000	7.900	14.400	7.400	11.900
		When selecting I/O outside the group only (Input only)	—	—	—	—	—	—
		When selecting I/O outside the group only (Output only)	—	—	—	—	—	—
		When selecting I/O outside the group only (Both I/O)	—	—	—	—	—	—
		When selecting refresh of multiple CPU high speed transmission area only	—	—	—	—	—	—
		When selecting communications with display unit	—	—	29.700	79.900	26.800	60.700
		When selecting communication with external devices only	21.300	89.000	9.500	32.800	9.200	25.200
		When selecting CC-Link IE Field Network Basic refresh only (master station side)	—	—	31.500	47.900	30.000	43.700
	FIFW	Number of data points = 0	6.100	14.200	4.200	6.700	3.200	4.600
		Number of data points = 96	6.100	14.200	4.400	6.800	3.300	3.800
	FIFR	Number of data points = 1	7.500	15.600	5.100	7.400	3.800	4.400
		Number of data points = 96	37.000	45.000	36.100	38.800	24.800	25.700
	FPOP	Number of data points = 1	7.600	15.600	4.900	7.500	3.800	5.300
		Number of data points = 96	7.600	15.600	5.000	7.500	3.700	5.400
	FINS	Number of data points = 0	6.900	15.000	5.400	7.500	3.700	5.300
		Number of data points = 96	36.600	44.700	5.000	7.400	3.700	5.300
	FDEL	Number of data points = 1	8.000	16.100	5.700	8.300	4.200	5.800
		Number of data points = 96	37.300	45.500	36.900	39.300	25.400	25.900

Category	Instruction	Condition (device)	Processing time (μs)					
			L02SCPU, L02SCPU-P		L02CPU, L02CPU-P		L06CPU, L06CPU-P, L26CPU, L26CPU-P, L26CPU-BT, L26CPU-PBT	
			Min.	Max.	Min.	Max.	Min.	Max.
Application instruction	FROM n1 n2 (D) n3	n3=1	17.400	74.700	11.600	31.000	10.700	23.600
		n3=1000	406.000	498.500	403.900	432.900	390.900	410.200
	DFRO n1 n2 (D) n3	n3=1	19.600	85.600	13.300	35.400	12.600	26.700
		n3=500	406.000	498.500	405.000	434.600	390.900	410.200
	TO n1 n2 (S) n3	n3=1	16.400	69.600	11.200	28.400	9.600	21.300
		n3=1000	381.300	471.200	381.500	410.900	372.500	390.800
	DTO n1 n2 (S) n3	n3=1	18.600	85.100	12.500	33.900	12.000	25.700
		n3=500	381.300	471.200	379.800	410.400	372.500	390.800
	LEDR	No display → no display	1.500	7.100	2.400	2.600	1.900	2.000
		LED instruction execution → no display	38.900	109.000	32.700	50.600	24.400	35.800
	BINDA (S) (D)	(S)=1	5.600	13.900	5.000	7.300	4.300	5.600
		(S)=-32768	7.800	16.200	7.400	9.800	6.500	8.000
	DBINDA (S) (D)	(S)=1	6.200	14.500	5.600	8.300	4.900	6.300
		(S)=-2147483648	11.000	19.200	10.500	12.900	9.600	11.000
	BINHA (S) (D)	(S)=1	5.050	13.400	4.500	6.900	3.700	5.200
		(S)=FFFFH	5.050	13.400	4.500	6.900	3.700	5.200
	DBINHA (S) (D)	(S)=1	5.600	13.900	5.000	7.600	4.600	6.000
		(S)=FFFFFFFFH	5.600	13.900	5.000	7.600	4.600	6.000
	BCDDA (S) (D)	(S)=1	4.850	13.200	4.300	6.700	3.600	5.000
		(S)=9999	5.300	13.600	4.800	7.100	4.100	5.400
	DBCDDA (S) (D)	(S)=1	5.300	13.600	4.900	7.200	4.000	5.500
		(S)=99999999	6.200	14.500	5.700	8.300	4.900	6.300
	DABIN (S) (D)	(S)=1	7.000	18.500	5.800	10.100	5.600	7.800
		(S)=-32768	6.950	18.500	5.800	10.100	5.600	7.800
	DDABIN (S) (D)	(S)=1	9.450	21.000	8.300	12.600	8.100	10.500
		(S)=-2147483648	9.450	21.000	8.300	12.600	8.100	10.500
	HABIN (S) (D)	(S)=1	5.650	17.100	4.500	8.800	4.400	6.500
		(S)=FFFFH	5.750	17.300	4.500	8.800	4.400	6.500
	DHABIN (S) (D)	(S)=1	6.800	18.200	5.500	10.000	5.300	7.700
		(S)=FFFFFFFFH	7.100	18.600	5.500	10.000	5.300	7.700
	DABCD (S) (D)	(S)=1	5.650	17.200	4.500	8.700	4.300	6.300
		(S)=9999	5.700	17.200	4.500	8.700	4.300	6.300
	DDABCD (S) (D)	(S)=1	6.850	18.300	5.500	9.800	5.500	7.500
		(S)=99999999	6.850	18.300	5.500	9.800	5.500	7.500
	COMRD	—	185.000	188.000	65.700	65.700	50.900	51.200
	LEN	1 character	4.700	16.200	3.900	7.800	3.600	5.500
		96 characters	20.600	32.900	19.700	23.900	16.800	18.700
	STR	—	9.800	36.500	7.500	16.700	6.600	10.400
	DSTR	—	12.100	40.400	10.200	19.700	9.600	11.500
	VAL	—	12.200	40.900	9.800	19.900	8.900	13.000
	DVAL	—	19.400	45.600	12.700	23.900	12.700	16.800
	ESTR	—	29.700	87.800	21.200	43.400	17.900	23.100

A

Category	Instruction	Condition (device)		Processing time (μs)					
				L02SCPU, L02SCPU-P		L02CPU, L02CPU-P		L06CPU, L06CPU-P, L26CPU, L26CPU-P, L26CPU-BT, L26CPU-PBT	
				Min.	Max.	Min.	Max.	Min.	Max.
Application instruction	EVAL	Decimal point format all 2-digit specification		23.900	70.400	28.300	41.000	22.500	29.00
		Exponent format all 6-digit specification		23.700	70.300	28.300	41.000	22.500	29.00
	ASC (S) (D) n	n=1		10.200	41.800	6.200	17.100	5.400	8.300
		n=96		31.900	66.600	30.300	42.100	25.200	28.400
	HEX (S) (D) n	n=1		8.600	43.400	5.400	16.000	5.400	9.000
		n=96		77.100	115.000	42.400	54.900	31.300	35.000
	RIGHT (S) (D) n	n=1		10.900	29.600	7.400	13.900	6.600	7.300
		n=96		41.400	60.300	39.300	45.800	29.200	31.600
	LEFT (S) (D) n	n=1		10.600	29.300	6.900	13.400	5.900	8.200
		n=96		41.300	60.200	39.300	45.800	29.200	31.500
	MIDR	—		11.700	30.600	10.200	16.500	8.100	10.300
	MIDW	—		12.400	24.000	10.700	14.900	8.800	10.200
	INSTR	No match		22.000	38.200	20.000	25.600	16.600	18.400
		Match	Head	13.300	29.600	11.000	16.500	9.100	10.900
			End	21.900	38.100	53.900	60.000	42.700	44.900
	EMOD	—		11.600	24.000	11.200	15.100	9.600	11.000
	EREXP	—		19.700	28.000	20.400	22.900	18.800	20.100
	STRINS (S) (D) n	(S)=128/(D)=40/n=1		47.000	102.000	45.300	63.400	35.300	47.600
		(S)=128/(D)=40/n=48		70.100	134.000	63.200	81.900	48.600	61.700
	STRDEL (S) (D) n	(S)=128/(D)=40/n=1		46.400	93.600	39.000	53.500	34.800	44.600
		(S)=128/(D)=40/n=48		44.500	70.600	40.800	50.400	29.200	38.100
	SIN	Single precision		6.400	13.900	5.000	8.400	4.100	5.700
	COS	Single precision		6.100	13.500	5.200	8.000	4.000	5.600
	TAN	Single precision		8.300	15.000	6.100	9.200	5.100	6.700
	ASIN	Single precision		7.300	15.600	6.900	10.900	5.900	8.500
	ACOS	Single precision		8.100	16.500	7.800	11.000	6.700	8.900
	ATAN	Single precision		5.350	12.000	4.700	7.300	3.900	6.000
	SIND	Double precision		13.400	51.300	9.400	22.300	8.500	13.800
	COSD	Double precision		14.700	51.700	10.000	22.300	8.800	14.600
	TAND	Double precision		17.400	54.400	12.200	24.900	10.800	16.500
	ASIND	Double precision		22.600	60.300	12.800	25.900	11.600	16.600
	ACOSD	Double precision		19.700	60.000	12.600	25.900	11.200	16.200
	ATAND	Double precision		15.000	51.800	10.500	22.900	9.100	13.800
	RAD	Single precision		3.200	10.300	3.000	6.400	2.100	4.300
	RADD	Double precision		5.200	43.100	5.200	16.900	3.600	9.200
	DEG	Single precision		3.200	11.500	2.900	6.600	2.200	4.400
	DEGD	Double precision		5.150	43.800	5.200	16.800	3.800	9.000
	SQR	Single precision		3.900	12.300	3.600	7.200	2.600	4.300
	SQRD	Double precision		7.000	45.700	6.200	19.100	5.200	11.000
	EXP (S) (D)	Single precision	(S)=-10	6.350	13.800	4.700	7.500	3.800	5.600
			(S)=1	6.350	13.800	4.700	7.500	3.800	5.600
	EXPD (S) (D)	Double precision	(S)=-10	15.800	52.700	9.300	22.100	8.000	13.500
			(S)=1	15.400	52.500	9.300	22.100	8.000	13.500
	LOG (S) (D)	Single precision	(S)=1	5.800	14.900	4.700	8.800	3.800	6.400
			(S)=10	7.450	16.500	6.300	10.400	5.200	7.700
	LOGD (S) (D)	Double precision	(S)=1	11.000	48.900	8.600	21.100	7.700	12.500
			(S)=10	12.600	51.300	10.200	23.000	9.200	14.300

Category	Instruction	Condition (device)			Processing time (μs)					
					L02SCPU, L02SCPU-P		L02CPU, L02CPU-P		L06CPU, L06CPU-P, L26CPU, L26CPU-P, L26CPU-BT, L26CPU-PBT	
					Min.	Max.	Min.	Max.	Min.	Max.
Application instruction	RND	—			1.950	5.450	1.500	2.500	0.800	1.800
	SRND	—			2.750	4.550	1.800	2.900	1.100	2.000
	BSQR (S) (D)	(S)=0			2.500	6.800	2.700	4.400	1.500	3.000
		(S)=9999			6.400	15.500	6.100	12.500	5.100	8.000
	BDSQR (S) (D)	(S)=0			2.600	6.050	2.700	4.400	1.500	3.000
		(S)=99999999			8.450	17.600	8.500	15.200	7.500	9.900
	BSIN	—			11.500	32.800	9.500	21.500	8.100	14.500
	BCOS	—			10.400	32.500	9.500	21.400	7.800	13.700
	BTAN	—			12.100	33.700	10.400	22.600	9.000	13.300
	BASIN	—			13.300	32.800	11.800	23.600	10.100	12.800
	BACOS	—			13.400	33.700	13.100	23.700	11.100	14.100
	BATAN	—			12.600	31.400	11.100	21.500	9.100	10.900
	POW (S1) (S2) (D)	Single precision	(S1)=12.3E+5 (S2)=3.45E+0		12.200	22.100	9.600	13.300	8.400	10.900
	POWD (S1) (S2) (D)	Double precision	(S1)=12.3E+5 (S2)=3.45E+0		27.300	61.000	18.900	30.600	18.200	26.500
	LOG10	Single precision			8.200	16.500	6.000	9.600	5.700	8.050
	LOG10D	Double precision			15.100	48.000	11.900	22.900	11.100	18.600
	LIMIT	—			5.350	5.500	4.000	4.000	2.400	2.700
	DLIMIT	—			6.000	6.150	4.400	4.400	2.800	3.000
	BAND	—			5.450	12.400	4.500	6.600	2.700	3.800
	DBAND	—			6.050	11.900	4.800	6.900	3.300	4.600
	ZONE	—			6.250	10.700	4.200	6.100	2.600	4.300
	DZONE	—			6.000	11.900	4.700	6.900	3.000	4.600
	LDDT=	Comparison of specified date		In conductive status	8.200	25.500	7.700	14.200	6.800	10.900
				In non-conductive status	8.200	25.500	7.700	14.200	6.800	10.900
		Comparison of current date		In conductive status	6.500	23.100	6.400	12.800	5.500	9.700
				In non-conductive status	6.500	23.100	6.400	12.800	5.500	9.700
	ANDDT=	When not executed			0.240		0.160		0.038	
		Comparison of specified date		In conductive status	8.200	25.500	7.300	14.000	6.500	10.700
				In non-conductive status	8.200	25.500	7.300	14.000	6.500	10.700
		Comparison of current date		In conductive status	6.500	23.100	6.100	12.700	5.300	9.300
				In non-conductive status	6.500	23.100	6.100	12.700	5.300	9.300

A

Category	Instruction	Condition (device)		Processing time (μs)					
				L02SCPU, L02SCPU-P		L02CPU, L02CPU-P		L06CPU, L06CPU-P, L26CPU, L26CPU-P, L26CPU-BT, L26CPU-PBT	
				Min.	Max.	Min.	Max.	Min.	Max.
Application instruction	ORDT=	When not executed		0.240		0.160		0.038	
		Comparison of specified date	In conductive status	8.200	25.500	7.400	14.400	6.700	10.800
			In non-conductive status	8.200	25.500	7.400	14.400	6.700	10.800
		Comparison of current date	In conductive status	6.500	23.100	6.000	12.800	5.400	9.600
			In non-conductive status	6.500	23.100	6.000	12.800	5.400	9.600
	LDDT<>	Comparison of specified date	In conductive status	8.200	25.500	7.700	14.200	6.800	10.900
			In non-conductive status	8.200	25.500	7.700	14.200	6.800	10.900
		Comparison of current date	In conductive status	6.500	23.100	6.400	12.800	5.500	9.700
			In non-conductive status	6.500	23.100	6.400	12.800	5.500	9.700
	ANDDT<>	When not executed		0.240		0.160		0.038	
		Comparison of specified date	In conductive status	8.200	25.500	7.300	14.000	6.500	10.700
			In non-conductive status	8.200	25.500	7.300	14.000	6.500	10.700
		Comparison of current date	In conductive status	6.500	23.100	6.100	12.700	5.300	9.300
			In non-conductive status	6.500	23.100	6.100	12.700	5.300	9.300
	ORDT<>	When not executed		0.240		0.160		0.038	
		Comparison of specified date	In conductive status	8.200	25.500	7.400	14.400	6.700	10.800
			In non-conductive status	8.200	25.500	7.400	14.400	6.700	10.800
		Comparison of current date	In conductive status	6.500	23.100	6.000	12.800	5.400	9.600
			In non-conductive status	6.500	23.100	6.000	12.800	5.400	9.600

Category	Instruction	Condition (device)		Processing time (μs)					
				L02SCPU, L02SCPU-P		L02CPU, L02CPU-P		L06CPU, L06CPU-P, L26CPU, L26CPU-P, L26CPU-BT, L26CPU-PBT	
				Min.	Max.	Min.	Max.	Min.	Max.
Application instruction	LDDT>	Comparison of specified date	In conductive status	8.200	25.500	7.700	14.200	6.800	10.900
			In non-conductive status	8.200	25.500	7.700	14.200	6.800	10.900
		Comparison of current date	In conductive status	6.500	23.100	6.400	12.800	5.500	9.700
			In non-conductive status	6.500	23.100	6.400	12.800	5.500	9.700
	ANDDT>	When not executed		0.240		0.160		0.038	
		Comparison of specified date	In conductive status	8.200	25.500	7.300	14.000	6.500	10.700
			In non-conductive status	8.200	25.500	7.300	14.000	6.500	10.700
		Comparison of current date	In conductive status	6.500	23.100	6.100	12.700	5.300	9.300
			In non-conductive status	6.500	23.100	6.100	12.700	5.300	9.300
	ORDT>	When not executed		0.240		0.160		0.038	
		Comparison of specified date	In conductive status	8.200	25.500	7.400	14.400	6.700	10.800
			In non-conductive status	8.200	25.500	7.400	14.400	6.700	10.800
		Comparison of current date	In conductive status	6.500	23.100	6.000	12.800	5.400	9.600
			In non-conductive status	6.500	23.100	6.000	12.800	5.400	9.600
	LDDT<=	Comparison of specified date	In conductive status	8.200	25.500	7.700	14.200	6.800	10.900
			In non-conductive status	8.200	25.500	7.700	14.200	6.800	10.900
		Comparison of current date	In conductive status	6.500	23.100	6.400	12.800	5.500	9.700
			In non-conductive status	6.500	23.100	6.400	12.800	5.500	9.700

Category	Instruction	Condition (device)		Processing time (μs)					
				L02SCPU, L02SCPU-P		L02CPU, L02CPU-P		L06CPU, L06CPU-P, L26CPU, L26CPU-P, L26CPU-BT, L26CPU-PBT	
				Min.	Max.	Min.	Max.	Min.	Max.
Application instruction	ANDDT<=	When not executed		0.240		0.160		0.038	
		Comparison of specified date	In conductive status	8.200	25.500	7.300	14.000	6.500	10.700
			In non-conductive status	8.200	25.500	7.300	14.000	6.500	10.700
		Comparison of current date	In conductive status	6.500	23.100	6.100	12.700	5.300	9.300
			In non-conductive status	6.500	23.100	6.100	12.700	5.300	9.300
	ORDT<=	When not executed		0.240		0.160		0.038	
		Comparison of specified date	In conductive status	8.200	25.500	7.400	14.400	6.700	10.800
			In non-conductive status	8.200	25.500	7.400	14.400	6.700	10.800
		Comparison of current date	In conductive status	6.500	23.100	6.000	12.800	5.400	9.600
			In non-conductive status	6.500	23.100	6.000	12.800	5.400	9.600
	LDDT<	Comparison of specified date	In conductive status	8.200	25.500	7.700	14.200	6.800	10.900
			In non-conductive status	8.200	25.500	7.700	14.200	6.800	10.900
		Comparison of current date	In conductive status	6.500	23.100	6.400	12.800	5.500	9.700
			In non-conductive status	6.500	23.100	6.400	12.800	5.500	9.700
	ANDDT<	When not executed		0.240		0.160		0.038	
		Comparison of specified date	In conductive status	8.200	25.500	7.300	14.000	6.500	10.700
			In non-conductive status	8.200	25.500	7.300	14.000	6.500	10.700
		Comparison of current date	In conductive status	6.500	23.100	6.100	12.700	5.300	9.300
			In non-conductive status	6.500	23.100	6.100	12.700	5.300	9.300

Category	Instruction	Condition (device)		Processing time (μs)					
				L02SCPU, L02SCPU-P		L02CPU, L02CPU-P		L06CPU, L06CPU-P, L26CPU, L26CPU-P, L26CPU-BT, L26CPU-PBT	
				Min.	Max.	Min.	Max.	Min.	Max.
Application instruction	ORDT<	When not executed		0.240		0.160		0.038	
		Comparison of specified date	In conductive status	8.200	25.500	7.400	14.400	6.700	10.800
			In non-conductive status	8.200	25.500	7.400	14.400	6.700	10.800
		Comparison of current date	In conductive status	6.500	23.100	6.000	12.800	5.400	9.600
			In non-conductive status	6.500	23.100	6.000	12.800	5.400	9.600
	LDDT>=	Comparison of specified date	In conductive status	8.200	25.500	7.700	14.200	6.800	10.900
			In non-conductive status	8.200	25.500	7.700	14.200	6.800	10.900
		Comparison of current date	In conductive status	6.500	23.100	6.400	12.800	5.500	9.700
			In non-conductive status	6.500	23.100	6.400	12.800	5.500	9.700
	ANDDT>=	When not executed		0.240		0.160		0.038	
		Comparison of specified date	In conductive status	8.200	25.500	7.300	14.000	6.500	10.700
			In non-conductive status	8.200	25.500	7.300	14.000	6.500	10.700
		Comparison of current date	In conductive status	6.500	23.100	6.100	12.700	5.300	9.300
			In non-conductive status	6.500	23.100	6.100	12.700	5.300	9.300
	ORDT>=	When not executed		0.240		0.160		0.038	
		Comparison of specified date	In conductive status	8.200	25.500	7.400	14.400	6.700	10.800
			In non-conductive status	8.200	25.500	7.400	14.400	6.700	10.800
		Comparison of current date	In conductive status	6.500	23.100	6.000	12.800	5.400	9.600
			In non-conductive status	6.500	23.100	6.000	12.800	5.400	9.600

Category	Instruction	Condition (device)		Processing time (μs)					
				L02SCPU, L02SCPU-P		L02CPU, L02CPU-P		L06CPU, L06CPU-P, L26CPU, L26CPU-P, L26CPU-BT, L26CPU-PBT	
				Min.	Max.	Min.	Max.	Min.	Max.
Application instruction	LDTM=	Comparison of specified time	In conductive status	8.200	25.500	7.600	14.000	6.700	10.800
			In non-conductive status	8.200	25.500	7.600	14.000	6.700	10.800
		Comparison of current time	In conductive status	6.500	23.100	6.200	12.700	5.400	9.500
			In non-conductive status	6.500	23.100	6.200	12.700	5.400	9.500
	ANDTM=	When not executed		0.240		0.160		0.038	
		Comparison of specified time	In conductive status	8.200	25.500	7.200	13.900	6.300	10.800
			In non-conductive status	8.200	25.500	7.200	13.900	6.300	10.800
		Comparison of current time	In conductive status	6.500	23.100	5.900	12.500	5.100	9.500
			In non-conductive status	6.500	23.100	5.900	12.500	5.100	9.500
	ORTM=	When not executed		0.240		0.160		0.038	
		Comparison of specified time	In conductive status	8.200	25.500	7.300	14.100	6.600	10.800
			In non-conductive status	8.200	25.500	7.300	14.100	6.600	10.800
		Comparison of current time	In conductive status	6.500	23.100	6.000	12.700	5.300	9.500
			In non-conductive status	6.500	23.100	6.000	12.700	5.300	9.500
	LDTM<>	Comparison of specified time	In conductive status	8.200	25.500	7.600	14.000	6.700	10.800
			In non-conductive status	8.200	25.500	7.600	14.000	6.700	10.800
		Comparison of current time	In conductive status	6.500	23.100	6.200	12.700	5.400	9.500
			In non-conductive status	6.500	23.100	6.200	12.700	5.400	9.500

Category	Instruction	Condition (device)		Processing time (μs)					
				L02SCPU, L02SCPU-P		L02CPU, L02CPU-P		L06CPU, L06CPU-P, L26CPU, L26CPU-P, L26CPU-BT, L26CPU-PBT	
				Min.	Max.	Min.	Max.	Min.	Max.
Application instruction	ANDTM<>	When not executed		0.240		0.160		0.038	
		Comparison of specified time	In conductive status	8.200	25.500	7.200	13.900	6.300	10.800
			In non-conductive status	8.200	25.500	7.200	13.900	6.300	10.800
		Comparison of current time	In conductive status	6.500	23.100	5.900	12.500	5.100	9.500
			In non-conductive status	6.500	23.100	5.900	12.500	5.100	9.500
	ORTM<>	When not executed		0.240		0.160		0.038	
		Comparison of specified time	In conductive status	8.200	25.500	7.300	14.100	6.600	10.800
			In non-conductive status	8.200	25.500	7.300	14.100	6.600	10.800
		Comparison of current time	In conductive status	6.500	23.100	6.000	12.700	5.300	9.500
			In non-conductive status	6.500	23.100	6.000	12.700	5.300	9.500
	LDTM>	Comparison of specified time	In conductive status	8.200	25.500	7.600	14.000	6.700	10.800
			In non-conductive status	8.200	25.500	7.600	14.000	6.700	10.800
		Comparison of current time	In conductive status	6.500	23.100	6.200	12.700	5.400	9.500
			In non-conductive status	6.500	23.100	6.200	12.700	5.400	9.500
	ANDTM>	When not executed		0.240		0.160		0.038	
		Comparison of specified time	In conductive status	8.200	25.500	7.200	13.900	6.300	10.800
			In non-conductive status	8.200	25.500	7.200	13.900	6.300	10.800
		Comparison of current time	In conductive status	6.500	23.100	5.900	12.500	5.100	9.500
			In non-conductive status	6.500	23.100	5.900	12.500	5.100	9.500

Category	Instruction	Condition (device)		Processing time (μs)					
				L02SCPU, L02SCPU-P		L02CPU, L02CPU-P		L06CPU, L06CPU-P, L26CPU, L26CPU-P, L26CPU-BT, L26CPU-PBT	
				Min.	Max.	Min.	Max.	Min.	Max.
Application instruction	ORTM>	When not executed		0.240		0.160		0.038	
		Comparison of specified time	In conductive status	8.200	25.500	7.300	14.100	6.600	10.800
			In non-conductive status	8.200	25.500	7.300	14.100	6.600	10.800
		Comparison of current time	In conductive status	6.500	23.100	6.000	12.700	5.300	9.500
			In non-conductive status	6.500	23.100	6.000	12.700	5.300	9.500
	LDTM<=	Comparison of specified time	In conductive status	8.200	25.500	7.600	14.000	6.700	10.800
			In non-conductive status	8.200	25.500	7.600	14.000	6.700	10.800
		Comparison of current time	In conductive status	6.500	23.100	6.200	12.700	5.400	9.500
			In non-conductive status	6.500	23.100	6.200	12.700	5.400	9.500
	ANDTM<=	When not executed		0.240		0.160		0.038	
		Comparison of specified time	In conductive status	8.200	25.500	7.200	13.900	6.300	10.800
			In non-conductive status	8.200	25.500	7.200	13.900	6.300	10.800
		Comparison of current time	In conductive status	6.500	23.100	5.900	12.500	5.100	9.500
			In non-conductive status	6.500	23.100	5.900	12.500	5.100	9.500
	ORTM<=	When not executed		0.240		0.160		0.038	
		Comparison of specified time	In conductive status	8.200	25.500	7.300	14.100	6.600	10.800
			In non-conductive status	8.200	25.500	7.300	14.100	6.600	10.800
		Comparison of current time	In conductive status	6.500	23.100	6.000	12.700	5.300	9.500
			In non-conductive status	6.500	23.100	6.000	12.700	5.300	9.500

Category	Instruction	Condition (device)		Processing time (μs)					
				L02SCPU, L02SCPU-P		L02CPU, L02CPU-P		L06CPU, L06CPU-P, L26CPU, L26CPU-P, L26CPU-BT, L26CPU-PBT	
				Min.	Max.	Min.	Max.	Min.	Max.
Application instruction	LDTM<	Comparison of specified time	In conductive status	8.200	25.500	7.600	14.000	6.700	10.800
			In non-conductive status	8.200	25.500	7.600	14.000	6.700	10.800
		Comparison of current time	In conductive status	6.500	23.100	6.200	12.700	5.400	9.500
			In non-conductive status	6.500	23.100	6.200	12.700	5.400	9.500
	ANDTM<	When not executed		0.240		0.160		0.038	
		Comparison of specified time	In conductive status	8.200	25.500	7.200	13.900	6.300	10.800
			In non-conductive status	8.200	25.500	7.200	13.900	6.300	10.800
		Comparison of current time	In conductive status	6.500	23.100	5.900	12.500	5.100	9.500
			In non-conductive status	6.500	23.100	5.900	12.500	5.100	9.500
	ORTM<	When not executed		0.240		0.160		0.038	
		Comparison of specified time	In conductive status	8.200	25.500	7.300	14.100	6.600	10.800
			In non-conductive status	8.200	25.500	7.300	14.100	6.600	10.800
		Comparison of current time	In conductive status	6.500	23.100	6.000	12.700	5.300	9.500
			In non-conductive status	6.500	23.100	6.000	12.700	5.300	9.500
	LDTM>=	Comparison of specified time	In conductive status	8.200	25.500	7.600	14.000	6.700	10.800
			In non-conductive status	8.200	25.500	7.600	14.000	6.700	10.800
		Comparison of current time	In conductive status	6.500	23.100	6.200	12.700	5.400	9.500
			In non-conductive status	6.500	23.100	6.200	12.700	5.400	9.500

A

Category	Instruction	Condition (device)		Processing time (μs)					
				L02SCPU, L02SCPU-P		L02CPU, L02CPU-P		L06CPU, L06CPU-P, L26CPU, L26CPU-P, L26CPU-BT, L26CPU-PBT	
				Min.	Max.	Min.	Max.	Min.	Max.
Application instruction	ANDTM>=	When not executed		0.240		0.160		0.038	
		Comparison of specified time	In conductive status	8.200	25.500	7.200	13.900	6.300	10.800
			In non-conductive status	8.200	25.500	7.200	13.900	6.300	10.800
		Comparison of current time	In conductive status	6.500	23.100	5.900	12.500	5.100	9.500
			In non-conductive status	6.500	23.100	5.900	12.500	5.100	9.500
	ORTM>=	When not executed		0.240		0.160		0.038	
		Comparison of specified time	In conductive status	8.200	25.500	7.300	14.100	6.600	10.800
			In non-conductive status	8.200	25.500	7.300	14.100	6.600	10.800
		Comparison of current time	In conductive status	6.500	23.100	6.000	12.700	5.300	9.500
			In non-conductive status	6.500	23.100	6.000	12.700	5.300	9.500
	SCL (S1) (S2) (D)	SM750= ON	Point No.1 < (S1) < Point No.2	14.900	50.100	12.500	29.200	11.900	23.000
			Point No.9 < (S1) < Point No.10	15.800	50.900	13.200	29.100	12.100	23.000
		SM750= OFF	Point No.1 < (S1) < Point No.2	13.900	53.100	12.100	28.900	10.900	22.200
			Point No.9 < (S1) < Point No.10	16.600	56.600	13.900	30.900	12.700	23.900

Category	Instruction	Condition (device)		Processing time (μs)					
				L02SCPU, L02SCPU-P		L02CPU, L02CPU-P		L06CPU, L06CPU-P, L26CPU, L26CPU-P, L26CPU-BT, L26CPU-PBT	
				Min.	Max.	Min.	Max.	Min.	Max.
Application instruction	DSCL (S1) (S2) (D)	SM750=ON	Point No.1 < (S1) < Point No.2	13.400	52.400	12.500	29.200	11.900	23.000
			Point No.9 < (S1) < Point No.10	14.200	54.100	13.200	29.100	12.100	23.000
		SM750=OFF	Point No.1 < (S1) < Point No.2	12.300	53.200	12.100	28.900	10.900	22.200
			Point No.9 < (S1) < Point No.10	15.000	57.600	13.900	30.900	12.700	23.900
	SCL2 (S1) (S2) (D)	SM750=ON	Point No.1 < (S1) < Point No.2	14.200	53.300	13.400	29.700	11.800	23.300
			Point No.9 < (S1) < Point No.10	14.900	55.000	12.900	29.500	12.100	23.300
		SM750=OFF	Point No.1 < (S1) < Point No.2	15.000	53.500	12.200	29.100	11.000	22.600
			Point No.9 < (S1) < Point No.10	16.300	56.400	13.900	30.700	12.600	23.900
	DSCL2 (S1) (S2) (D)	SM750=ON	Point No.1 < (S1) < Point No.2	13.400	52.700	13.400	29.700	11.800	23.300
			Point No.9 < (S1) < Point No.10	14.200	54.300	12.900	29.500	12.100	23.300
		SM750=OFF	Point No.1 < (S1) < Point No.2	12.300	53.200	12.200	29.100	11.000	22.600
			Point No.9 < (S1) < Point No.10	15.000	57.600	13.900	30.700	12.600	23.900
	RSET	Standard RAM		6.800	26.900	3.500	11.100	2.700	5.900
	DATERD	—		5.600	27.800	4.600	11.200	2.500	4.200
	DATEWR	—		7.800	42.100	6.500	19.300	4.100	8.900
	DATE+	No digit increase		14.200	41.200	10.000	19.400	4.700	6.600
		Digit increase		14.200	41.200	9.900	19.700	4.600	6.500
	DATE-	No digit increase		15.100	41.200	9.000	17.900	4.600	7.000
		Digit increase		15.100	41.200	10.000	19.200	4.600	6.500
	SECOND	—		5.800	20.500	4.600	9.800	2.200	3.400
	HOUR	—		6.200	22.500	4.600	10.300	2.400	4.300
	HOURM	—		7.000	30.400	5.900	14.100	4.300	11.400
	DHOURM	—		7.200	31.600	6.000	14.800	4.600	11.800
	TCMP	—		9.700	45.700	7.500	20.800	6.700	16.600
	TZCP	—		11.800	54.100	9.800	24.300	8.500	19.900

A

Category	Instruction	Condition (device)		Processing time (μs)					
				L02SCPU, L02SCPU-P		L02CPU, L02CPU-P		L06CPU, L06CPU-P, L26CPU, L26CPU-P, L26CPU-BT, L26CPU-PBT	
				Min.	Max.	Min.	Max.	Min.	Max.
Application instruction	QDRSET	SRAM card to standard RAM		—	—	—	—	—	—
		Standard RAM to SRAM card		—	—	—	—	—	—
	QCDSET	SD memory card to standard ROM		—	—	690.800	736.470	1146.900	1179.500
		Standard ROM to SD memory card		—	—	6981.400	7232.070	5613.900	5653.500
	S.DATERD	—		9.250	51.000	7.800	22.500	4.800	7.100
	S.DATE+	No digit increase		16.800	75.400	15.100	34.100	7.400	10.000
		Digit increase		16.800	75.400	15.000	34.100	7.400	10.000
	S.DATE-	No digit increase		17.600	75.300	13.700	33.600	7.400	10.300
		Digit increase		16.900	75.300	13.700	33.600	7.500	10.200
	PSTOP	—		82.200	199.000	67.600	104.100	56.600	79.800
	POFF	—		82.600	198.000	66.800	103.600	57.200	79.800
	PSCAN	—		83.600	200.000	67.900	104.800	60.100	79.900
	WDT	—		2.900	12.000	1.600	4.800	1.100	2.400
	DUTY	—		7.700	27.500	4.900	10.100	4.800	9.600
	TIMCHK	—		5.350	24.500	4.100	9.100	3.500	4.700
	ZRRDB	File register of standard RAM		4.100	4.200	2.900	3.300	1.800	2.100
		File register of SRAM card		—	—	—	—	—	—
	ZRWRB	File register of standard RAM		5.400	5.500	3.600	3.800	2.400	2.700
		File register of SRAM card		—	—	—	—	—	—
	ADRSET	—		2.400	6.650	2.200	4.800	2.100	2.600
	ZPUSH	—		9.200	20.500	8.000	12.000	5.800	7.500
	ZPOP	—		9.000	15.5000	8.200	10.900	5.800	6.400
	UNIRD n1 (D) n2	n2=1		6.000	33.100	5.000	14.100	3.700	8.000
		n2=16		16.500	43.600	13.600	22.600	12.200	16.600
	TYPERD	—		43.400	139.800	32.100	67.600	29.500	52.500
	TRACE	Start		174.000	174.000	58.100	58.100	43.800	44.700
	TRACER	—		5.100	15.500	6.100	6.100	4.500	4.500
	RBMOV (S) (D) n	When standard RAM is used	1 point	—	—	—	—	—	—
			1000 points	—	—	—	—	—	—
		When SRAM card is used	1 point	—	—	—	—	—	—
			1000 points	—	—	—	—	—	—
	UMSG	Number of displayed characters = 1		—	—	7.300	17.000	7.000	13.500
		Number of displayed characters = 32		—	—	16.500	26.300	14.300	21.300
	PID	—		25.800	143.700	21.500	67.100	12.100	53.600
	SP.FWRITE	—		—	—	81.000	81.800	63.500	64.100
	SP.FREAD	—		—	—	81.100	81.700	61.600	62.500
	SP.DEVST	—		125.000	125.000	50.100	50.100	39.400	39.400
	S.DEVLD	—		18.300	36.700	12.000	27.600	10.000	17.000

Category	Instruction	Condition (device)		Processing time (μs)					
				L02SCPU, L02SCPU-P		L02CPU, L02CPU-P		L06CPU, L06CPU-P, L26CPU, L26CPU-P, L26CPU-BT, L26CPU-PBT	
				Min.	Max.	Min.	Max.	Min.	Max.
Data link instruction	S.ZCOM	When mounting CC-Link module (master station side)		29.400	91.700	23.700	48.500	19.300	26.000
		When mounting CC-Link module (local station side)		29.500	91.600	23.700	48.500	19.100	26.200
		When selecting CC-Link IE Field Network refresh only (master station side)		79.900	214.000	31.500	72.000	31.000	58.000
		When selecting CC-Link IE Field Network refresh only (local station side)		79.900	214.000	31.500	72.000	31.000	58.000
		When mounting MELSECNET/H module or CC-Link IE Controller Network module (control station side)		—	—	—	—	—	—
		When mounting MELSECNET/H module or CC-Link IE Controller Network module (normal station side)		—	—	—	—	—	—
	S.RTREAD	—		12.600	65.000	8.500	27.000	7.400	19.000
	S.RTWRITE	—		13.300	67.100	9.000	28.000	8.300	19.800
	S.REFDVWRB	When the refresh device as the write target exists at Transfer 1		—	—	90.100	98.400	98.200	118.100
		When the refresh device as the write target exists at Transfer 256		—	—	546.600	559.800	577.900	596.400
	S.REFDVWRW	When the refresh device as the write target exists at Transfer 1		—	—	88.600	97.700	97.200	116.800
		When the refresh device as the write target exists at Transfer 256		—	—	544.900	554.100	588.500	606.400
	S.REFDVRDB	When the refresh device as the read target exists at Transfer 1		—	—	89.500	97.900	96.000	117.300
		When the refresh device as the read target exists at Transfer 256		—	—	545.800	559.000	577.000	595.500
	S.REFDVRDW	When the refresh device as the read target exists at Transfer 1		—	—	88.500	97.700	95.800	116.800
		When the refresh device as the read target exists at Transfer 256		—	—	545.100	554.000	589.600	606.100
Communication protocol function instruction	S(P).CPRTCL	—		138.000	192.600	68.600	83.100	50.700	60.300
	S.TO n1 n2 n3 n4 (D)	Writing to host CPU shared memory	n4=1	—	—	—	—	—	—
			n4=320	—	—	—	—	—	—
	TO n1 n2 (S) n3	Writing to host CPU shared memory	n3=1	—	—	—	—	—	—
			n3=320	—	—	—	—	—	—
	DTO n1 n2 (S) n3	Writing to host CPU shared memory	n3=1	—	—	—	—	—	—
			n3=320	—	—	—	—	—	—
	FROM n1 n2 (D) n3	Reading from host CPU shared memory	n3=1	—	—	—	—	—	—
			n3=320	—	—	—	—	—	—
		Reading from other CPU shared memory	n3=1	—	—	—	—	—	—
			n3=320	—	—	—	—	—	—
			n3=1000	—	—	—	—	—	—
	DFRO n1 n2 (D) n3	Reading from host CPU shared memory	n3=1	—	—	—	—	—	—
			n3=320	—	—	—	—	—	—
		Reading from other CPU shared memory	n3=1	—	—	—	—	—	—
			n3=320	—	—	—	—	—	—
			n3=1000	—	—	—	—	—	—

A

Category	Instruction	Condition (device)		Processing time (μs)					
				L02SCPU, L02SCPU-P		L02CPU, L02CPU-P		L06CPU, L06CPU-P, L26CPU, L26CPU-P, L26CPU-BT, L26CPU-PBT	
				Min.	Max.	Min.	Max.	Min.	Max.
Instructions for multiple CPU high-speed transmission	D.DDWR n (S1) (S2) (D1) (D2)	Writing Devices to Another CPU	n=1	—	—	—	—	—	—
			n=16	—	—	—	—	—	—
			n=96	—	—	—	—	—	—
	DP.DDWR n (S1) (S2) (D1) (D2)		n=1	—	—	—	—	—	—
			n=16	—	—	—	—	—	—
			n=96	—	—	—	—	—	—
	D.DDRD n (S1) (S2) (D1) (D2)	Reads devices from another CPU.	n=1	—	—	—	—	—	—
			n=16	—	—	—	—	—	—
			n=96	—	—	—	—	—	—
	DP.DDRD n (S1) (S2) (D1) (D2)		n=1	—	—	—	—	—	—
			n=16	—	—	—	—	—	—
			n=96	—	—	—	—	—	—

Point

For the instructions for which a rising edge instruction (□P) is not described, the processing time is the same as an ON execution instruction.
(Example) WORDP instruction, TOP instruction etc.

■Table of the time to be added when file register, extended data register, extended link register, module access device, and link direct device are used

- When using L02SCPU, L02SCPU-P, L02CPU, L02CPU-P, L06CPU, L06CPU-P, L26CPU, L26CPU-P, L26CPU-BT, and L26CPU-PBT

Device name		Data	Device specification location	Addition time (μs)		
				L02SCPU, L02SCPU-P	L02CPU, L02CPU-P	L06CPU, L06CPU-P, L26CPU, L26CPU-P, L26CPU-BT, L26CPU-PBT
File register (R)	When standard RAM is used	Bit	Source	0.100	0.100	0.048
			Destination	0.220	0.220	0.038
		Word	Source	0.100	0.100	0.048
			Destination	0.100	0.100	0.038
		Double word	Source	0.200	0.200	0.095
			Destination	0.200	0.200	0.086
File register (ZR), extended data register (D), extended link register (W)	When standard RAM is used	Bit	Source	0.160	0.140	0.057
			Destination	0.320	0.280	0.048
		Word	Source	0.160	0.140	0.057
			Destination	0.160	0.140	0.048
		Double word	Source	0.260	0.240	0.105
			Destination	0.260	0.240	0.095
Module access device (Un\G□)		Bit	Source	15.000	11.700	11.200
			Destination	21.300	15.400	15.300
		Word	Source	10.600	9.460	9.410
			Destination	33.000	19.000	19.000
		Double word	Source	24.200	11.000	10.900
			Destination	34.800	18.800	18.700
Link direct device (Jn\□)		Bit	Source	70.900	41.600	37.900
			Destination	120.100	63.200	58.100
		Word	Source	68.400	40.700	37.500
			Destination	53.700	31.700	30.800
		Double word	Source	75.600	49.400	43.400
			Destination	58.900	39.600	37.300

A

Appendix 2 CPU Performance Comparison

Comparison of Q, LCPU with AnNCPU, AnACPU, and AnUCPU

Usable devices

Device name		QCPU			LCPU	AnUCPU	AnACPU	AnNCPU
Number of I/O points[*9]		Q00J: 256 points Q00: 1024 points Q01: 1024 points	Q00UJ: 256 points Q00U: 1024 points Q01U: 1024 points	Q02, Q02H, Q06H, Q12H, Q25H, Q02PH, Q06PH, Q12PH, Q25PH, Q12PRH, Q25PRH, Q03UD(E), Q03UDV, Q04UD(E)H, Q04UDV, Q04UDPV, Q06UD(E)H, Q06UDV, Q06UDPV, Q10UD(E)H, Q13UD(E)H, Q13UDV, Q13UDPV, Q20UD(E)H, Q26UD(E)H, Q26UDV, Q26UDPV, Q50UDEH, Q100UDEH: 4096 points Q02U: 2048 points	L02SCPU, L02SCPU-P, L02CPU, L02CPU-P: 1024 points L06CPU, L06CPU-P, L26CPU, L26CPU-P, L26CPU-BT, L26CPU-PBT: 4096 points	A2U: 512 points A2U-S1: 1024 points A3U: 2048 points A4U: 4096 points	A2A: 512 points A2A-S1: 1024 points A3A: 2048 points	A1N: 256 points A2N: 512 points A2N-S1: 1024 points A3N: 2048 points
Number of I/O device points[*8]		2048 points[*1]	8192 points[*1]		8192 points[*1]	8192 points	Same with I/O devices points of each CPU	
Internal relay		8192 points[*1]			8192 points[*1]	Total 8192 points		Total 2048 points
Latch relay		2048 points[*1]	8192 points[*1]		8192 points[*1]			
Step relay	Sequence program	—			—			—
	SFC	2048 points[*6]	8192 points		8192 points	—		
Annunciator		1024 points[*1]	2048 points[*1]		2048 points[*1]	2048 points		256 points
Edge relay		1024 points[*1]	2048 points[*1]		2048 points[*1]	—		
Link relay		2048 points[*1]	8192 points[*1]		8192 points[*1]	8192 points	4096 points	1024 points
Link special relay		1024 points	2048 points		2048 points	56 points		
Timer		512 points[*1]	2048 points[*1]		2048 points[*1]	Total 2048 points		Total 256 points
Retentive timers		0 points[*1]			0 points[*1]			
Counter		512 points[*1]	1024 points[*1]		1024 points[*1]	1024 points		256 points
Data register		11136 points[*1]	12288 points[*1]		12288 points[*1]	8192 points	6144 points	1024 points
Link register		2048 points[*1]	8192 points[*1]		8192 points[*1]	8192 points	4096 points	1024 points
Link special register		1024 points	2048 points		2048 points	56 points		
Function input		16 points (FX0 to FXF)[*7]			16 points (FX0 to FXF)[*7]	—		
Function output		16 points (FY0 to FYF)[*7]			16 points (FY0 to FYF)[*7]			
Special relay		1000 points	2048 points		2048 points	256 points		
Function register		5 points (FD0 to FD4)			5 points (FD0 to FD4)	—		
Special register		1000 points	2048 points		2048 points	256 points		
Link direct device		Designated by J□\□			Designated by J□\□	—		
Intelligent function module device		Designated by U□\G□			Designated by U□\G□			
Index register	Z	10 points (Z0 to Z9)	Other than Universal model QCPU: 16 points (Z0 to Z15) Universal model QCPU: 20 points (Z0 to Z19)		20 points (Z0 to Z19)	7 points (Z, Z1 to Z6)		1 point (Z)
	V[*2]	—			—	7 points (V, V1 to V6)		1 point (V)

Device name	QCPU			LCPU	AnUCPU	AnACPU	AnNCPU
File register	32768 points/block[*5] (R0 to R32767)	32768 points/block (R0 to R32767)[*10]		32768 points/block (R0 to R32767)	8192 points/block(R0 to R8191)		
Accumulator[*3]	—			—	2 points		
Nesting	15 points			15 points	8 points		
Pointer	300 points	512 points	4096 points	4096 points[*11]	256 points		
Interrupt pointers	128 points	128 points	256 points	256 points	32 points		
SFC blocks	126[*6]	320 points		320 points	—		
SFC transition devices	—	512 points		512 points			
Decimal constants	K - 2147483648 to K2147483647						
Hexadecimal constants	H0 to HFFFFFFFF						
Real number constants[*6]	E ± 1.17550-38 to E ± 3.40282+38				—		
Character string	"QnACPU", "ABCD"[*4]						

*1 The number of device points can be changed at the parameters.
*2 CPU uses V as an edge relay.
*3 Instructions that used accumulators with the AnNCPU, AnACPU, and AnUCPU have different formats with the QCPU.
*4 Can only be used by the $MOV instruction with the Q00JCPU, Q00CPU, and Q01CPU.
*5 The Q00JCPU does not have file registers.
*6 Applicable to products with the first five digits of the serial number 04122 or higher (Q00JCPU, Q00CPU, and QCPU).
*7 Each 5 points of FX0 to FX4 and FY0 to FY4 can be used on the programs.
*8 The number of points that can be used on the programs
*9 The number of accessible points to actual I/O modules
*10 The Q00UJCPU does not have file registers.
*11 Only when L06CPU, L06CPU-P, L26CPU, L26CPU-P, L26CPU-BT and L26CPU-PBT whose first digits of serial number are "16042" or later are used, the pointer for auto assignment device up to 32768 points can be used. (In order to use them, the PLC Parameter should be set. For the PLC Parameter, refer to the MELSEC-L CPU Module User's Manual (Function Explanation, Program Fundamentals).)

I/O control mode

I/O control mode			QCPU	LCPU	AnUCPU	AnACPU	AnNCPU
Refresh mode	—		○	○	○	○	○[*2]
	Direct I/O method	Partial refresh instructions	○	○	○	○	○
		Dedicated instruction[*1]	—	—	○	○	—
		Direct access input	○	○	—	—	—
		Direct access output	○	○	—	—	—
Direct mode			—	—	—	—	○[*2]

Symbol in table ○: Usable, —: Unusable

*1 The DOUT, DSET, and SRST instructions are direct output dedicated instructions. There are no dedicated instructions for direct input.
*2 Switching between the refresh mode and direct mode is conducted with an AnNCPU DIP switch.

A

Data that can be used by instructions

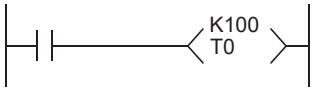
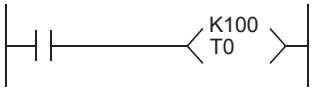
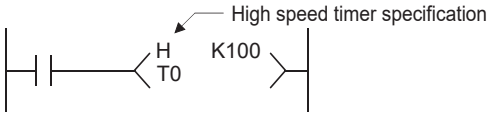
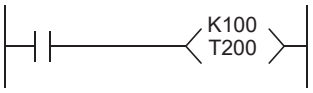
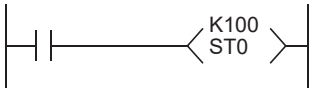
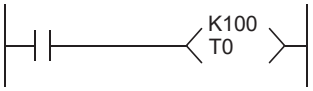
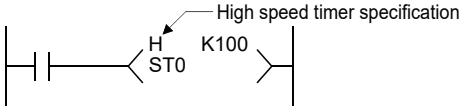
Setting data		QCPU	LCPU	AnUCPU	AnACPU	AnNCPU
Bit data	Bit device	○	○	○	○	○
	Word device	○ (Bit specification required)	○ (Bit specification required)	—	—	—
Word data	Bit device	○ (Digit specification required)	○ (Digit specification required)	○ (Digit specification required)	○ (Digit specification required)	○ (Digit specification required)
	Word device	○	○	○	○	○
Double word data	Bit device	○ (Digit specification required)	○ (Digit specification required)	○ (Digit specification required)	○ (Digit specification required)	○ (Digit specification required)
	Word device	○	○	○	○	○
Real number data		○*1	○	○	○	—
Character string data		○*2	○	—	—	—

Symbol in table ○: Usable, —: Unusable

*1 Applicable to products with the first five digits of the serial number 04122 or higher (Q00JCPU, Q00CPU, and Q01CPU).

*2 Usable with only the $MOV instruction for the Q00JCPU, Q00CPU, and Q01CPU.

Timer comparison

Function		QCPU/LCPU	AnUCPU	AnACPU	AnNCPU
Low speed timer	Measurement unit	100ms (default value) Change of measurement unit at the parameter is enabled. QCPU/LCPU: 1 to 1000ms (0.01ms unit)	Fixed at 100ms		
	Designation method				
High speed timer	Measurement unit	10ms (default value) Change of measurement unit at the parameter is enabled. QnUCPU/LCPU: 0.01 to 100ms (0.01ms unit) QCPU(Other than QnUCPU): 0.1 to 100ms (0.1ms unit)	Fixed at 10ms		
	Designation method	 High speed timer setting: Conducted by sequence program	 High speed timer setting: Conducted at parameters		
Retentive timers	Measurement unit	Same measurement unit as low speed timer	Fixed at 100ms		
	Designation method				
High-speed retentive timer	Measurement unit	Same measurement unit as high speed timer	None		
	Designation method	 High speed timer setting: Conducted by sequence program			
Setting range for set values		1 to 32767	1 to 32767		
Processing for set value 0		Momentarily ON	No maximum (does not time out)		
Index modification	Contact	Enabled (only Z0 and Z1 are usable)*2	Enabled		Disabled
	Coil	Enabled (only Z0 and Z1 are usable)*2	Disabled		Disabled
	Set value	Enabled (Z0 to Z15 are usable)*1	Disabled		Disabled
	Present value	Enabled (Z0 to Z15 are usable)*1	Enabled		Enabled
Update processing for present value		When OUT Tn instruction is executed	When END processing is done		
Contact ON/OFF processing					

*1 The Q00J/Q00/Q01CPU can use Z0 to Z9.
The Universal model QCPU/LCPU can use Z0 to Z19.

*2 The High-speed Universal model QCPU and Universal model Process CPU can use those other than Z0 and Z1.

A

■Cautions on using timers

QCPU, LCPU updates the present value of timers and turns ON/OFF the contacts of them at the execution of OUT T□ instruction.

Therefore, if "Present value ≥ Set value" when the timer coil is turned ON, the contact of that timer is turned ON.

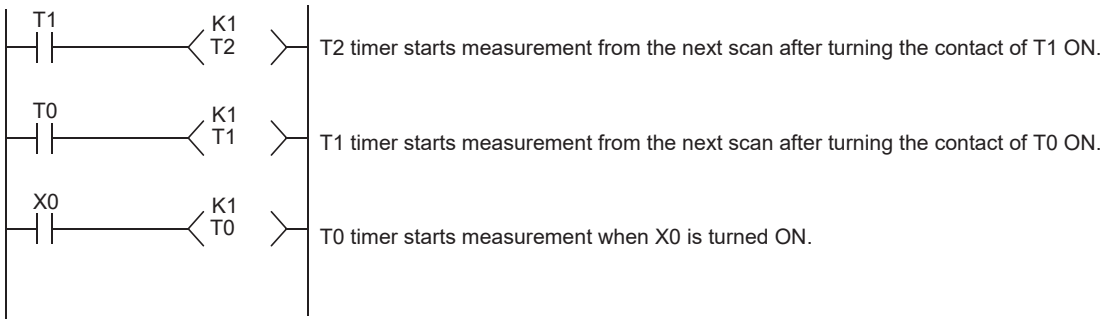
When creating a program in which the operation of the timer contact triggers the operation of another timer, create the program for the timer that operates later first.

In the following cases, all timers go ON at the same scan if the program is created in the order the timers operate.

- With high speed timers, if the set value is smaller than a scan time.
- With slow timers, if "1" is set.

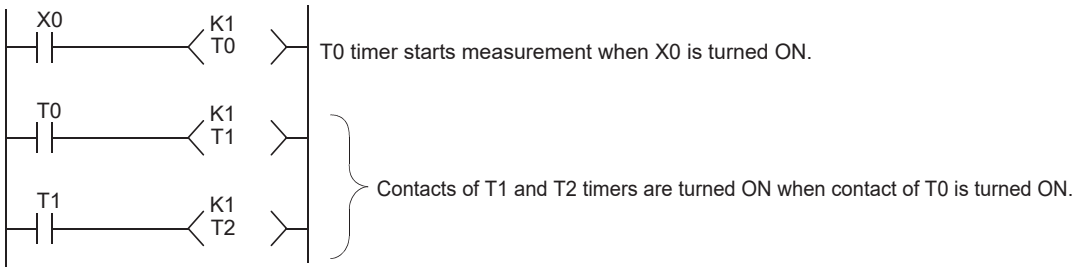
Ex.

For timers T0 to T2, the program is created in the order the timer operates later.

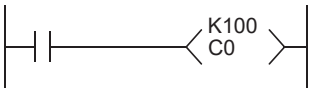
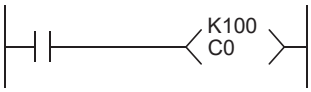


Ex.

For timers T0 to T2, the program is created in the order of timer operation.



Comparison of counters

Function		QCPU/LCPU	AnUCPU	AnACPU	AnNCPU
Designation method					
Index modification	Contact	• Enabled (only Z0 and Z1 are usable)*2	• Enabled		• Disabled
	Coil	• Enabled (only Z0 and Z1 are usable)*2	• Disabled		• Disabled
	Set value	• Disabled	• Disabled		• Disabled
	Present value	• Enabled (Z0 to Z15 are usable)*1	• Enabled		• Enabled
Update processing for present value		• When OUT Cn instruction is executed	• When END processing is done		
Contact ON/OFF processing					

*1 The Q00J/Q00/Q01CPU can use Z0 to Z9.
The Universal model QCPU/LCPU can use Z0 to Z19.

*2 The High-speed Universal model QCPU and Universal model Process CPU can use those other than Z0 and Z1.

Comparison of display instructions

Instruction	QCPU/LCPU	AnUCPU	AnACPU	AnNCPU
PR*1	• When SM701 is OFF: Output continued until 00H encountered • When SM701 is ON: 16 characters output	• When M9049 is OFF: Output continued until 00H encountered • When M9049 is ON: 16 characters output		
PRC*1	• When SM701 is OFF: 32-character comment output • When SM701 is ON: Upper 16 characters output	16-character comment output		

*1 Unusable for the Q00J/Q00/Q01CPU.

Instructions whose designation format has been changed*1

*1 Except dedicated instructions for AnACPU and AnUCPU

Because the QCPU, LCPU does not have accumulators (A0, A1), the format of AnUCPU, AnACPU and AnNCPU instructions that used accumulators has been changed.

Function	QCPU/LCPU		AnUCPU/AnACPU/AnNCPU	
	Instruction format	Remark	Instruction format	Remark
16-bit rotation to right	ROR D n	• D: Rotation data	ROR n	• Rotation data are set at A0.
	RCR D n	• D: Rotation data • SM700 is used for carry flag.	RCR n	• Rotation data are set at A0. • M9012 is used for carry flag.
16-bit rotation to left	ROL D n	• D: Rotation data	ROL n	• Rotation data are set at A0.
	RCL D n	• D: Rotation data • SM700 is used for carry flag.	RCL n	• Rotation data are set at A0. • M9012 is used for carry flag.
32-bit rotation to right	DROR D n	• D: Rotation data	DROR n	• Rotation data are set at A0 and A1.
	DRCR D n	• D: Rotation data • SM700 is used for carry flag.	DRCR n	• Rotation data are set at A0 and A1. • M9012 is used for carry flag.
32-bit rotation to left	DROL D n	• D: Rotation data	DROL n	• Rotation data are set at A0 and A1.
	DRCL D n	• D: Rotation data • SM700 is used for carry flag.	DRCL n	• Rotation data are set at A0 and A1. • M9012 is used for carry flag.
16-bit data search	SER S1 S2 D n	• Search results are stored at the D and D+1 devices.	SER S1 S2 n	• Search results are stored at A0 and A1.
32-bit data search	DSER S1 S2 D n	• Search results are stored at the D and D+1 devices.	DSER S1 S2 n	• Search results are stored at A0 and A1.
16-bit data bit check	SUM S D	• Check results are stored at the D device.	SUM S	• Check results are stored at A0.
32-bit data bit check	DSUM S D	• Check results are stored at the D device.	DSUM S	• Check results are stored at A0.
Partial refresh	RFS D n	• Dedicated instruction is added.	SEG D n	• Only when M9052 is ON
8-character ASCII conversion	$MOV (Character string) D	—	ASC (Character string) D	—
Carry flag set	SET SM700	• No dedicated instruction	STC	—
Carry flag reset	RST SM700	• No dedicated instruction	CLC	—
Jump to END instruction	GOEND	• Dedicated instruction is added.	CJ P255	• P255: END instruction designation
CHK instruction*2	CHKST CHK	• The CHKST instruction is added.	CJ Pn P254 CHK	—

*2 Unusable for the Q00J/Q00/Q01CPU/Universal model QCPU/LCPU.

A

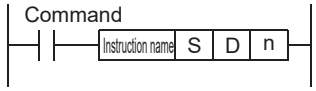
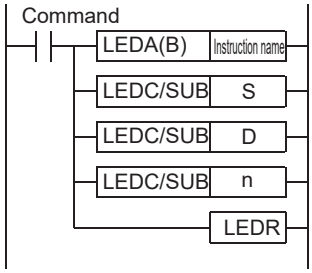
AnACPU and AnUCPU dedicated instructions

■Method of expression of dedicated instructions

Dedicated instructions based on the LEDA, LEDB, LEDC, SUB, and LEDR instructions, that are used with the AnACPU or AnUCPU have been changed for the same format as the basic instructions and the application instructions for the QCPU, LCPU.

The instructions that cannot be converted due to the absence of the corresponding instructions in the QCPU, LCPU are converted into OUT SM1255/OUT SM999 (for the Q00J/Q00/Q01CPU).

The instructions that have been converted into OUT SM1255/OUT SM999 should be replaced by other instructions or deleted.

QCPU/LCPU	AnUCPU/AnACPU
	 S, D, and n indicate data used by instructions.

■Dedicated instructions whose names have been changed

Dedicated instructions for the AnUCPU or AnACPU which have the same instruction name as is used for basic instructions and application instructions have undergone name changes in the QCPU, LCPU.

Function	QCPU/LCPU	AnUCPU/AnACPU
Floating point addition	E+	ADD
Floating point subtraction	E-	SUB
Floating point multiplication	E*	MUL
Floating point division	E/	DIV
Data dissociation	NDIS	DIS
Data association	NUNI	UNI
Updating check patterns	CHKCIR*1, CHKEND*1	CHK, CHKEND

*1 Unusable for the Q00J/Q00/Q01CPU/Universal model QCPU/LCPU.

Appendix 3 Application Program Examples

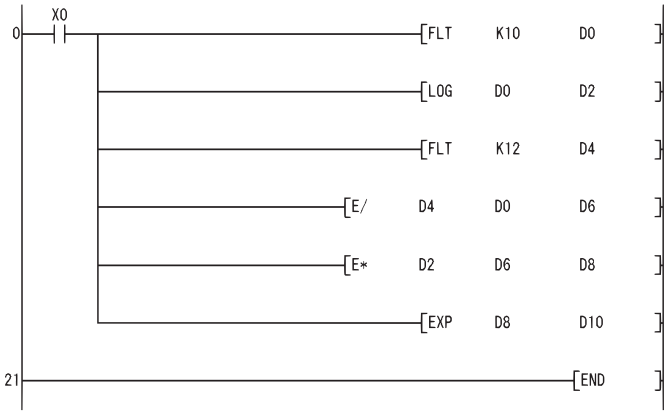
Concept of programs which perform operations of a nth power of X, a nth root X

The combination of the LOG instruction and the EXP instruction enables the operation of powers and roots (X^n, $\sqrt[n]{X}$).

Concept of programs which perform operations of X^n

X^n can be operated using $e^{(n\log_e X)}$.

For example, the operation of $10^{1.2}$ is $e^{(1.2\times\log_e 10)}$, which is represented in the form of a sequence program as shown below.



Converts 10 into a real number format data and stores the result in D0 and D1.

Executes Loge10 operation and stores the result in D2 and D3.

Converts 12 into a real number format data and stores the result into D4 and D5.

Divides D4 and D5 (12) by D0 and D1 (10), and stores the result (1, 2) in D6 and D7 (1, 2).

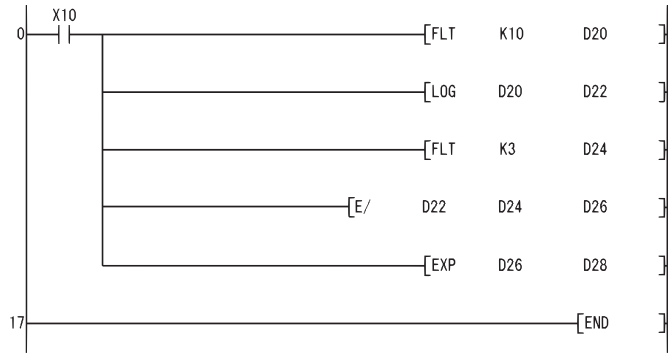
Multiplies D2 and D3 (Loge10) by D6 and D7 (1, 2) and stores the result in D8 and D9.

Executes Loge(D8, D9) operation and stores the result in D10 and D11.

Concept of program which performs operation of $\sqrt[n]{X}$

$\sqrt[n]{X}$ can be operated using $e^{(\frac{1}{n}\log_e X)}$.

For example, the operation of $\sqrt[3]{10}$ is $e^{(\frac{1}{3}\times\log_e 10)}$, which is represented in the form of a sequence program as shown below.



Converts 10 into a real number format data and stores the result in D20 and D21.

Executes Loge10 operation and stores the result in D22 and D23.

Converts 3 into a real number type data and stores the result in D24 and D25.

Divides D22 and D23 (Loge10) by D24 and D25 (3) and stores the result in D26 and D27.

Executes Loge(D26, D27) operation and stores the result in D28 and D29.

A

MEMO

INDEX

I

R

S

T

U

INSTRUCTION INDEX

Symbols

A

B

C

I

D

E

F

G

H

I

J

K

L

M

N

O

P

Q

R

S

T

U

V

W

X

Z

I

REVISIONS

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

Print date	*Manual number	Revision
December 2008 ⋮ December 2014	SH(NA)-080809ENG-A ⋮ SH(NA)-080809ENG-R	Due to the transition to the e-Manual, the details of revision have been deleted.
March 2015	SH(NA)-080809ENG-S	Complete revision (layout change)
January 2017	SH(NA)-080809ENG-T	Partial correction
August 2017	SH(NA)-080809ENG-U	■Added or modified parts Section 2.5, 6.8, 7.11, 7.15, Appendix 1
October 2018	SH(NA)-080809ENG-V	■Added or modified parts TERMS,Section 2.4, 2.5, 3.2, 3.3, 5.1, 6.5, 7.1, 7.11, 7.15, 7.19, Appendix 1
March 2021	SH(NA)-080809ENG-W	■Added or modified parts Section 6.8, 7.6, 7.12, Appendix 1
October 2021	SH(NA)-080809ENG-X	■Added or modified parts Section 7.5

Japanese manual version SH-080804-AD

WARRANTY

Please confirm the following product warranty details before using this product.

1. Gratis Warranty Term and Gratis Warranty Range

If any faults or defects (hereinafter "Failure") found to be the responsibility of Mitsubishi occurs during use of the product within the gratis warranty term, the product shall be repaired at no cost via the sales representative or Mitsubishi Service Company.

However, if repairs are required onsite at domestic or overseas location, expenses to send an engineer will be solely at the customer's discretion. Mitsubishi shall not be held responsible for any re-commissioning, maintenance, or testing on-site that involves replacement of the failed module.

[Gratis Warranty Term]

The gratis warranty term of the product shall be for one year after the date of purchase or delivery to a designated place. Note that after manufacture and shipment from Mitsubishi, the maximum distribution period shall be six (6) months, and the longest gratis warranty term after manufacturing shall be eighteen (18) months. The gratis warranty term of repair parts shall not exceed the gratis warranty term before repairs.

[Gratis Warranty Range]

(1) The range shall be limited to normal use within the usage state, usage methods and usage environment, etc., which follow the conditions and precautions, etc., given in the instruction manual, user's manual and caution labels on the product.

(2) Even within the gratis warranty term, repairs shall be charged for in the following cases.

1. Failure occurring from inappropriate storage or handling, carelessness or negligence by the user. Failure caused by the user's hardware or software design.
2. Failure caused by unapproved modifications, etc., to the product by the user.
3. When the Mitsubishi product is assembled into a user's device, Failure that could have been avoided if functions or structures, judged as necessary in the legal safety measures the user's device is subject to or as necessary by industry standards, had been provided.
4. Failure that could have been avoided if consumable parts (battery, backlight, fuse, etc.) designated in the instruction manual had been correctly serviced or replaced.
5. Failure caused by external irresistible forces such as fires or abnormal voltages, and Failure caused by force majeure such as earthquakes, lightning, wind and water damage.
6. Failure caused by reasons unpredictable by scientific technology standards at time of shipment from Mitsubishi.
7. Any other failure found not to be the responsibility of Mitsubishi or that admitted not to be so by the user.

2. Onerous repair term after discontinuation of production

(1) Mitsubishi shall accept onerous product repairs for seven (7) years after production of the product is discontinued. Discontinuation of production shall be notified with Mitsubishi Technical Bulletins, etc.

(2) Product supply (including repair parts) is not available after production is discontinued.

3. Overseas service

Overseas, repairs shall be accepted by Mitsubishi's local overseas FA Center. Note that the repair conditions at each FA Center may differ.

4. Exclusion of loss in opportunity and secondary loss from warranty liability

Regardless of the gratis warranty term, Mitsubishi shall not be liable for compensation to:

(1) Damages caused by any cause found not to be the responsibility of Mitsubishi.

(2) Loss in opportunity, lost profits incurred to the user by Failures of Mitsubishi products.

(3) Special damages and secondary damages whether foreseeable or not, compensation for accidents, and compensation for damages to products other than Mitsubishi products.

(4) Replacement by the user, maintenance of on-site equipment, start-up test run and other tasks.

5. Changes in product specifications

The specifications given in the catalogs, manuals or technical documents are subject to change without prior notice.

TRADEMARKS

Microsoft, Microsoft Access, Excel, SQL Server, Visual Basic, Visual C++, Visual Studio, Windows, Windows NT, Windows Server, Windows Vista, and Windows XP are either registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries.

The company names, system names and product names mentioned in this manual are either registered trademarks or trademarks of their respective companies.

In some cases, trademark symbols such as '™' or '®' are not specified in this manual.

SH(NA)-080809ENG-X(2110)KWIX
MODEL: QCPU-P-KY-E
MODEL CODE: 13JW10

MITSUBISHI ELECTRIC CORPORATION

HEAD OFFICE : TOKYO BUILDING, 2-7-3 MARUNOUCHI, CHIYODA-KU, TOKYO 100-8310, JAPAN
NAGOYA WORKS : 1-14 , YADA-MINAMI 5-CHOME , HIGASHI-KU, NAGOYA , JAPAN