Panasonic

PROGRAMMABLE CONTROLLER

FP7 CPU Unit

Command Reference Manual

WUME-FP7CPUPGR-02

2013.12 | panasonic.net/id/pidsx/global

Introduction

Thank you for buying a Panasonic product. Before you use the product, please carefully read the installation instructions and the users manual, and understand their contents in detail to use the product properly.

Types of Manual

- There are different types of users manual for the FP7 series, as listed below. Please refer to a relevant manual for the unit and purpose of your use.
- The manuals can be downloaded on our website:
  http://industrial.panasonic.com/ac/e/dl_center/manual/

Unit name or purpose of use	Manual name	Manual code
FP7 Power Supply Unit	FP7 CPU Unit Users Manual (Hardware)	WUME-FP7CPUH
FP7 CPU Unit	FP7 CPU Unit Command Reference Manual	WUME-FP7CPUPGR
	FP7 CPU Unit Users Manual (Logging Trace Function)	WUME-FP7CPULOG
	FP7 CPU Unit Users Manual (Security Function)	WUME-FP7CPUSEC
Instructions for Built-in LAN Port	FP7 CPU Unit Users Manual (LAN Port Communication)	WUME-FP7LAN
Instructions for Built-in COM Port	FP7 series Users Manual (SCU communication)	WUME-FP7COM
FP7 Extension Cassette (Communication) (RS-232C/RS485 type)		
FP7 Extension Cassette (Communication) (Ethernet type)	FP7 series Users Manual (Communication cassette Ethernet type)	WUME-FP7CCET
FP7 Extension (Function) Cassette Analog Cassette	FP7 Analog Cassette Users Manual	WUME-FP7FCA (Upcoming)
FP7 Digital Input/Output Unit	FP7 Digital Input/Output Unit Users Manual	WUME-FP7DIO
FP7 Analog Input Unit	FP7 Analog Input Unit Users Manual	WUME-FP7AIH
FP7 Analog Output Unit	FP7 Analog Output Unit Users Manual	WUME-FP7AOH
FP7 High-speed counter Unit	FP7 High-speed counter Unit Users Manual	WUME-FP7HSC
FP7 Pulse Output Unit	FP7 Pulse Output Unit Users Manual	WUME-FP7PG (Upcoming)
FP7 Positioning Unit	FP7 Positioning Unit Users Manual	WUME-FP7POSP
FP7 Serial Communication Unit	FP7 series Users Manual (SCU communication)	WUME-FP7COM
PHLS System	PHLS System Users Manual	WUME-PHLS
Programming Software FPWIN GR7	FPWIN GR7 Introduction Guidance	WUME-FPWINGR7

Table of Contents

1

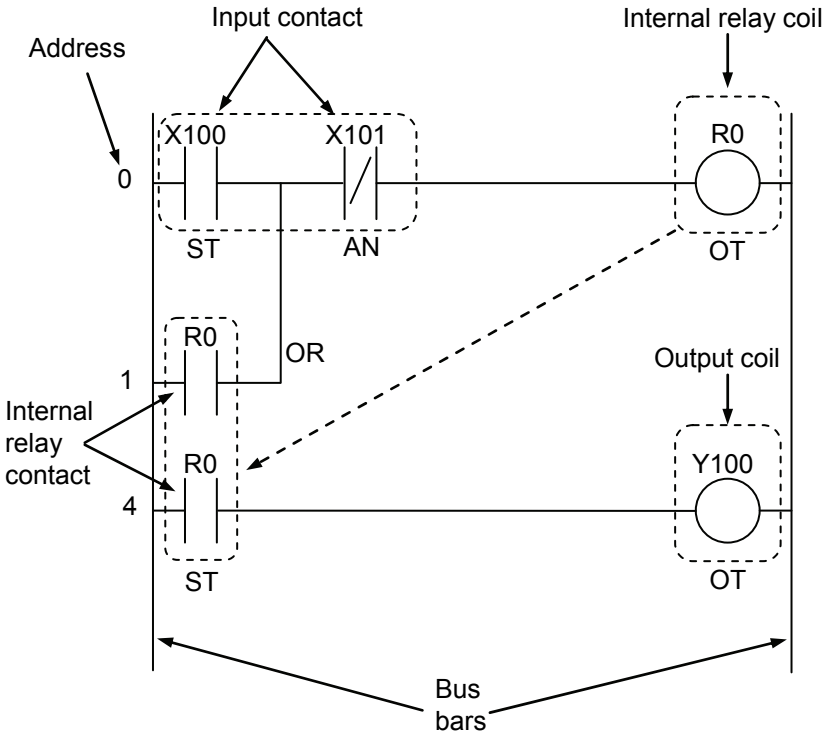
Overview of Instructions

1-1 Structure of Instructions

Structural patterns of basic instructions

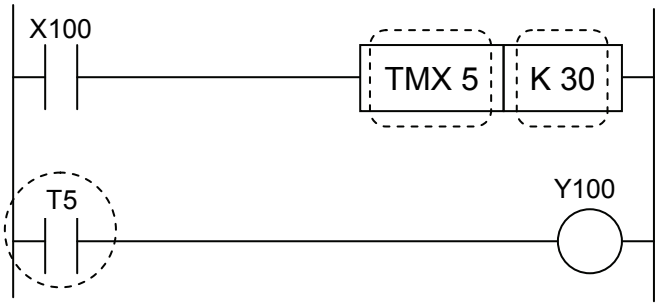
■ Sequence basic instructions

- Basic instructions are a group of the most essential instructions based on the relay sequence circuit, used for logic operation by the unit of bits. As indicated below, they are expressed by a combination of relay coils and contacts.
- Available relays vary by instruction. Refer to explanations for respective instructions.



■ Basic function instructions

- These are a group of instructions that execute timer, counter, shift register and other basic functions.



In this example, a 0.1-second timer with the timer number 5 is set to 3.0 seconds. It starts counting time when X100 is on, and the contact T5 turns on when the timer reaches 3.0 seconds.

■ Control instructions

- These are a group of instructions that determine the order and flow for executing a program. Conditions can be set to modify parts of a program to be executed, or to execute necessary parts only.
- Major control instructions include the following.

Mnemonic	Name	Functions
MC-MCE	Master control relay	When execution conditions are off, output between MC and MCE are turned off.
JP/LBL	Jump label	When execution conditions are on, execution between JP and LBL is skipped. This cuts down on time for program execution.
LOOP/LBL	Loop label	When execution conditions are on, the part of a program between LBL and LOOP is repeated for specified times.

■ Step ladder instructions

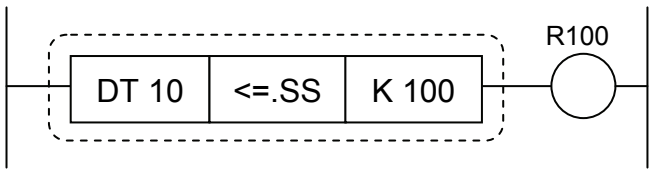
- Handle a section specified by SSTP to STPE as an independent "process", and operate sequential execution or branch execution.

■ Subroutine instructions

- Call and execute subroutines when necessary. Subrountines are programs for repeated execution of operations, etc., as specified by SBL and RET.

■ Data comparison instructions

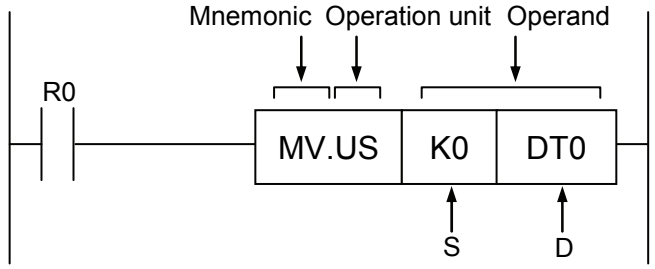
These are a group of instructions for comparing two sets of data. They operate as a contact that turns on and off in accordance with the comparison result.



Structural patterns of high-level instructions

■ Structure of high-level instructions

- Mnemonics indicate the specifics of operation (e.g. data transfer, arithmetic operation).
- Operation units indicate the units for operation triggered by instructions (e.g. US, SS, UL, SL, SF, DF).
- Operands indicate the targets of operation and/or methods for operation processes. There are three types of operands: [S], [D], and [n]. The number and types of operands that need to be specified vary by instruction.



■ Types of operands

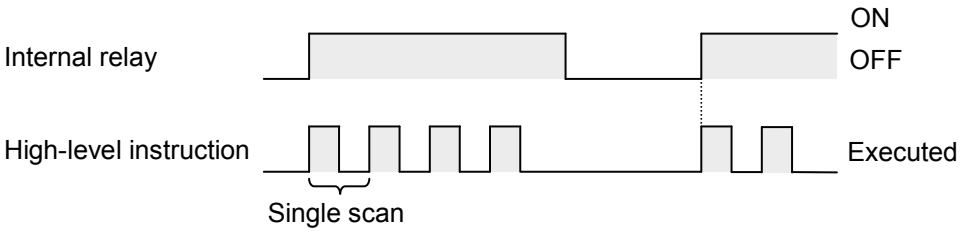
Symbol	Type	Functions
S	**Source**	Specify data targeted by operation and/or methods for processing.
D	**Destination**	Specify the storage location for operation result.
n	**Number**	Numerical data that specify data targeted by operation and/or methods for processing.

■ High-level instructions executed in every scan and only at the leading edge

- There are high-level instructions executed in every scan, and those executed only at the leading edge.

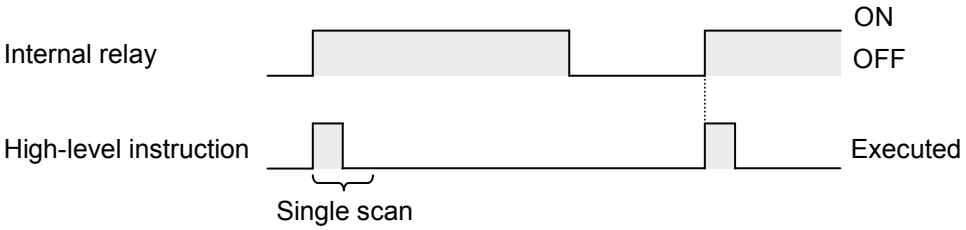
(1) High-level instructions executed in every scan

This type of instructions are repeated in every scan as long as the internal relay is established.



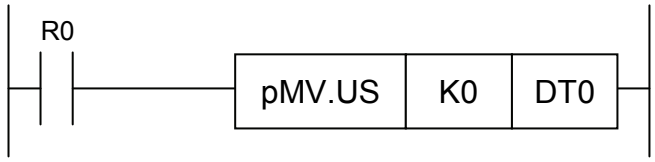
(2) High-level instructions executed only at the leading edge

This type of instructions are repeated in every scan as long as the internal relay is established.



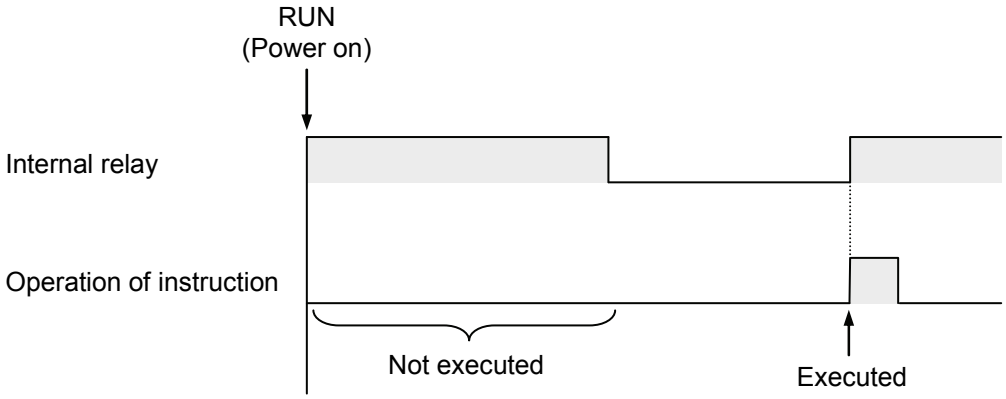
■ Input of high-level instructions executed only at the leading edge

- In order to input a high-level instruction executed only at the leading edge, using the tool software FPWINGR, press down [Shift] key + [F6] key, and then select a desired instruction in the instruction list dialog box.
- "(P)" is prefixed to a high-level instruction executed only at the leading edge.



■ Operation of high-level instructions executed only at the leading edge

- While the internal relay as the execution condition for the relevant instruction remains on, the instruction is only executed when it is switched on, and not executed afterwards.
- If the internal relay is already established before switching to the RUN mode or powering on in the RUN mode, the high-level instruction executed only at the leading edge is not executed in the first scan.



- When a high-level instruction executed only at the leading edge (a "P" instruction) is to be used in combination with instructions for changing the order of instruction execution, such as MC - MCE instructions and JP - LBL instructions ((1) to (6) below), operations vary by the timings for instruction execution and count input.

(1) MC - MCE instructions

(2) JP - LBL instructions

(3) LOOP - LBL instructions

(4) CNDE instructions

(5) Stepladder instructions

(6) Subroutine instructions

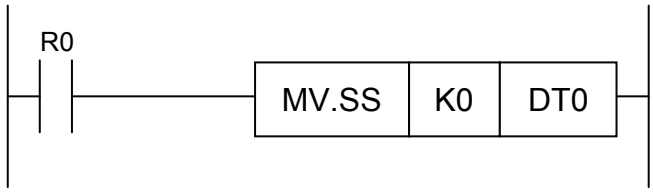
◆ **REFERENCE**

- **For details, see "4-7 Leading Edge Detection Method".**
- **When a high-level instruction executed only at the leading edge is combined with an AND stack instruction or a POP stack instruction, make sure that they are described correctly. For details, see "4-8 Precautions for Programming".**

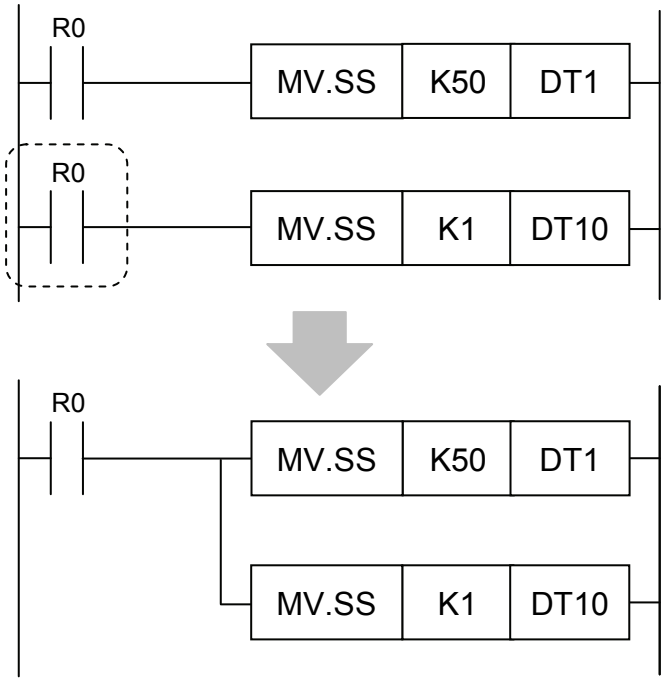
■ High-level instructions and internal relays

- High-level instructions are always used in pair with internal relays.
- When the operation result of the relay sequence circuit specified for an internal relay is on, the high-level instruction is executed.

Example) When the internal relay R0 is on, the MV instruction is executed, and K0 is transferred to DT0.



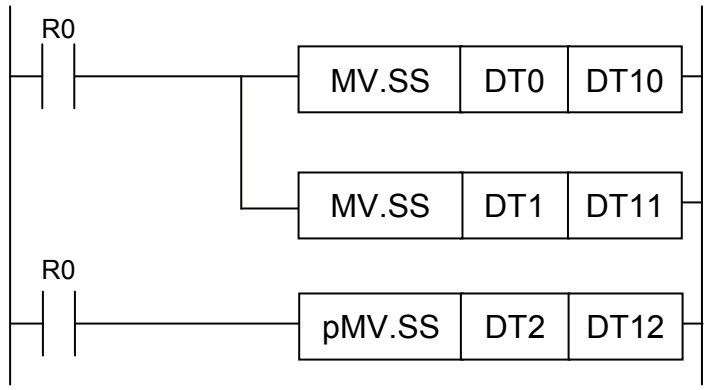
- When high-level instructions with the same internal relay are used consecutively, the internal relay can be omitted from the second instruction onwards.



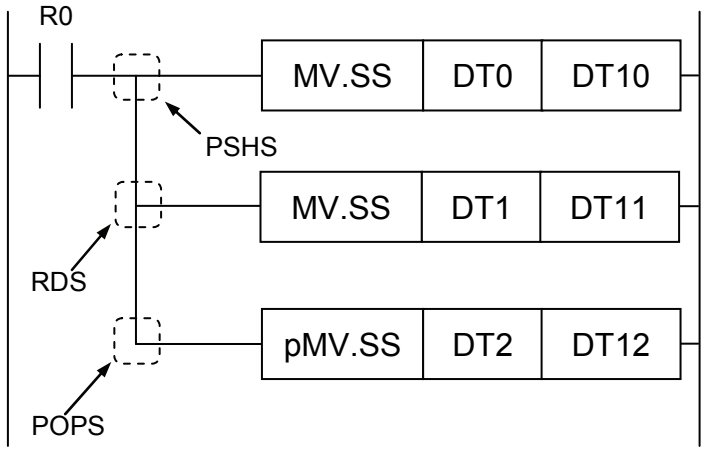
■ Precautions for omitting internal relays

- If both high-level instructions executed in every scan and those executed only at the leading edge are used with the same internal relay, a program should be written as follows.

Example 1) Separately describe high-level instructions executed in every scan and those executed only at the leading edge.



Example 2) Use PUSH, RDS, and/or POPS instructions.



1-2 Operation Unit

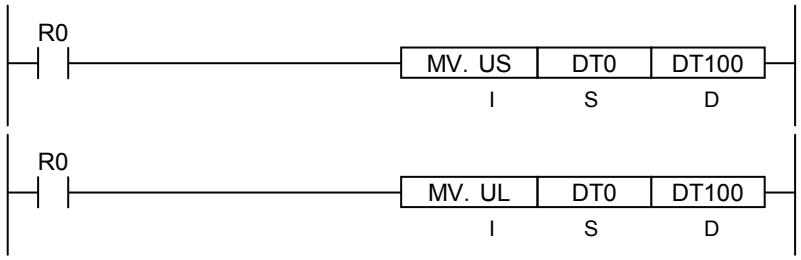
■ What are operation units?

- They specify basic units for operation triggered by respective instructions.
- Operation result of the same instruction may vary by the specified operation unit. The range of operand targeted by operation and/or the no. of words to store the operation result also vary.
- Available operation units vary by instruction. In some instructions, operation units are not specified. Refer to explanations for respective instructions. Operation units are indicated with "i".

Example 1) Differing operation units for a transfer instruction

When the operation unit is US (unsigned 16-bit data), the value of DT0 is transferred to DT100.

When the operation unit is UL(unsigned 32-bit data), the values of DT0 to DT1 are transferred to DT100 to DT101.

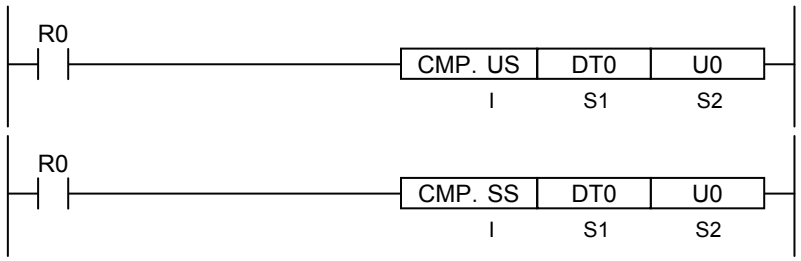


Example 2) Differing operation units for a comparison instruction

When binary data "1111 1111 1111 1111" are stored in DT0:

If the operation unit is US (unsigned 16-bit data), the value of (S1) is handled as a positive number “65535”, resulting in (S1) > (S2), and the comparison flag SRA turns on.

If the operation unit is SS (signed 16-bit data), the value of (S1) is handled as a negative number “-1”, resulting in (S1) < (S2), and the comparison flag SRC turns on.



■ Types of operation units

Symbol	Name	Available range
bit	1-bit data	0,1
US	Unsigned 16-bit data	0 to 65535
SS	Signed 16-bit data	-32768 to +32767
UL	Unsigned 32-bit data	0 to 4294967295
SL	Signed 32-bit data	-2147483548 to +2147483547
SF	Single precision floating point real number data	-1.175494E-38 to -3.402823E+38 0 +1.175494E-38 to +3.402823E+38
DF	Double precision floating point real number data	-2.2250738585072014E-308 to -1.7976931348623158E+308 0 +2.2250738585072014E-308 to +1.7976931348623158E+308

■ Operation units and available constants (●: Available)

Operation unit			K constant		U constant		H constant		Real numbers		String
Name	Size	preceding sign	16 bit	32 bit	16 bit	32 bit	16 bit	32 bit	SF	DF	""
US	16bit	Off	–	–	●	–	●	–	–	–	–
SS		On	●	–	–	–	●	–	–	–	–
UL	32bit	Off	–	–	–	●	–	●	–	–	–
SL		On	–	●	–	–	–	●	–	–	–
SF	32bit	On	–	–	–	–	–	–	●	–	–
DF	64bit	On	–	–	–	–	–	–	–	●	–

1-3 List of Operation Devices

■ List of operation devices

No. of bits	Symbol		Name	Points	Range	Explanation
	Global	Local				
1	**X**	**_X**	External input	8192 points	0 to 511F	Turns on or off based on external input.
	Y	**_Y**	External output	8192 points	0 to 511F	Externally outputs on or off state
	R	**_R**	Internal relay	32768 points	0 to 2047F	Relay that turns on or off only within a program.
	L	**_L**	Link relay	16384 points	0 to 1023F	Shared relay used for a PLC link
	T	**_T**	Timer	4096 points	0 to 4095	Turns on when the timer reaches the specified time.
	C	**_C**	Counter	1024 points	0 to 1023	Turns on when the counter reaches the specified value.
	SR	**-**	System relay	Approx. 1120 points		Relay that turns on or off based on specific conditions, and is used as a flag, etc.
	P	**-**	Pulse relay	4096 points	0 to 255F	Relay that turns on only in the first scan after the execution condition is switched on
	E	**-**	Error alarm relay	4096 points	0 to 4095	Relay for having the memory store given error conditions allocated by the user
	IN	**-**	Direct input	Max. 1008 points	0 to 62F	Relay for input/output processing during operation, independent of normal I/O refresh
	OT	**-**	Direct output	Max. 1008 points	0 to 62F	
16	**WX**	**_WX**	External input	512 words	0 to 511	Code for speciyfying 16 external input points as one word (16 bits) of data.
	WY	**_WY**	External output	512 words	0 to 511	Code for specifying 16 external output points as one word (16 bits) of data.
	WR	**_WR**	Internal relay	2048 words	0 to 2047	Code for specifying 16 internal relay points as one word (16 bits) of data.
	WL	**_WL**	Link relay	1024 words	0 to 1023	Code for specifying 16 link relay points as one word (16 bits) of data.
	DT	**_DT**	Data register	Up to 1 M words	0 to 999423	Data memory used in a program
	LD	**_LD**	Link register	16384 words	0 to 16383	Shared data memory used for a PLC link
	UM	**-**	Unit memory	Up to 512 kw / unit	0 to 524287	Device for accessing the unit memory of a high-function unit; The size varies by unit, and is allocated by default.
	SD	**-**	System data	Approx. 80 words	–	Data memory for storing specific data. Various settings and error codes are stored.
	WI	**-**	Direct input	Up to 62 words	0 to 62	Access for unit input
	WO	**-**	Direct output	Up to 62 words	0 to 62	Access for unit output
32	**TS**	**_TS**	Timer set value	4096 double words	0 to 4095	Data memory for storing the timer target value; It corresponds to the timer number.
	TE	**_TE**	Timer elapsed value	4096 double words	0 to 4095	Data memory for storing the timer elapsed value; It corresponds to the timer number.
	CS	**_CS**	Counter set value	1024 double words	0 to 1023	Data memory for storing the counter target value; It corresponds to the counter number.
	CE	**_CE**	Counter elapsed value	1024 double words	0 to 1023	Data memory for storing the counter elapsed value; It corresponds to the counter number.
	I0 to IE	**-**	Index modification register	15 double words	I0 to IE	Register for modifying memory area addresses and/or constants

Note 1) Values in the table indicate the number of devices that can be used in a program. The actual number of input/output points that can be used vary by configuration.

Note 2) Operation devices are categorized into "hold type", where a status immediately before power failure or switching to the PROG. mode is retained, and "non-hold type", where the status is reset. Areas of the non-hold type are cleared to zero when the unit is powered on or at switching between PROG and RUN.

Types of operation devices	Setting of hold or non-hold type
Internal relay (R), data register (DT), link relay (L), link register (LD)	Can be specified as a hold or non-hold type, using the tool software. However, data registers that can be used as hold type are a maximum of 262,144 words.
Counter (C), counter set value (CS), counter set value (CE), error alarm relay (E)	Hold type
Input (X), output (Y), timer (T), timer set value (TS), timer set value (TE), pulse relay (P), direct input (IN), direct output (OT), index register (I), unit memory (UM), system data (SD)	Non-hold type

Note 3) Devices for direct input (IN), direct output (OT), and unit memory (UM) are used by specifying the slot numbers and memory addresses to be controlled using instructions.

Note 4) The number of data registers (DT) that can be used vary by the type of CPU unit and the setting of memory configuration.

Unit type	Memory type	Memory selection patterns				
		1	2	3	4	5
CPS4*	Data register capacity (words)	65,536	131,072	262,144	524,288	999,424
CPS3*	Data register capacity (words)	131,072	262,144	425,985	589,824	

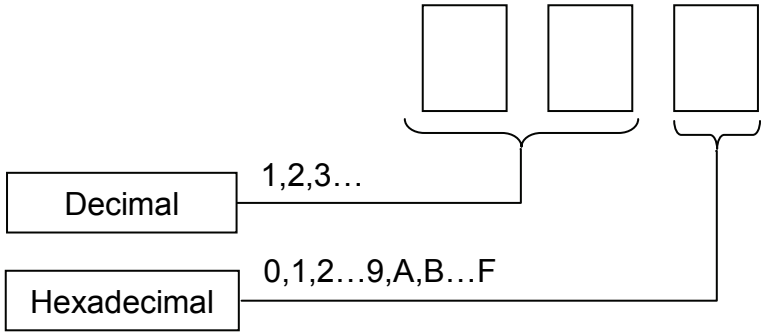
■ List of constants

Symbol	Name	No. of operation bits	Range
K	Signed decimal constants	16	-32768 to +32767
		32	-2147483548 to +2147483547
U	Unsigned decimal constants	16	0 to 65535
		32	0 to 4294967295
H	Hexadecimal constants	16	0 to FFFF
		32	0 to FFFF FFFF
SF	Single precision floating point real number constants	32	-1.175494E-38 to -3.402823E+38 0 +1.175494E-38 to +3.402823E+38
DF	Double precision floating point real number constants	64	-2.2250738585072014E-308 to -1.7976931348623158E+308 0 +2.2250738585072014E-308 to +1.7976931348623158E+308
""	Character constants	8	Up to 256 characters

1-4 Specification of Device Numbers

■ **External input (X), external output (Y), internal relay (R), link relay (L), pulse relay (P), system relay (SR), direct input (IN), and direct output (OT):**

Since relays may be handled in units of 16 points, their numbers should be expressed as a combination of decimal and hexadecimal numbers.



[Example] External input (X)

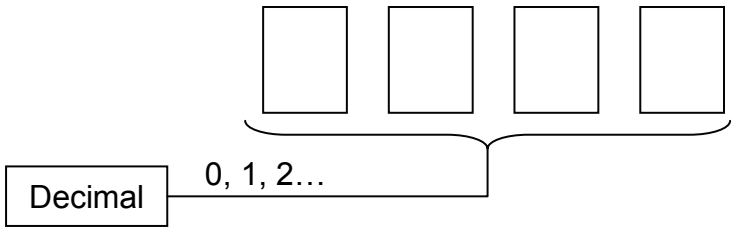
X 0, X 1 ····· X F
X 10, X 11 ····· X 1F
X 20, X 21 ····· X 2F
⋮ ⋮

■ **Relay number of external input/output**

- Only external inputs (X's) that are actually allocated to input contacts can be used.
- Only external outputs (Y's) that are actually allocated to output contacts can be used. Y's that are not allocated can be used as internal relays.
- Allocation of numbers are determined by the combination of units.

■ **Timer contact (T), counter contact (C), and error alarm relay (E)**

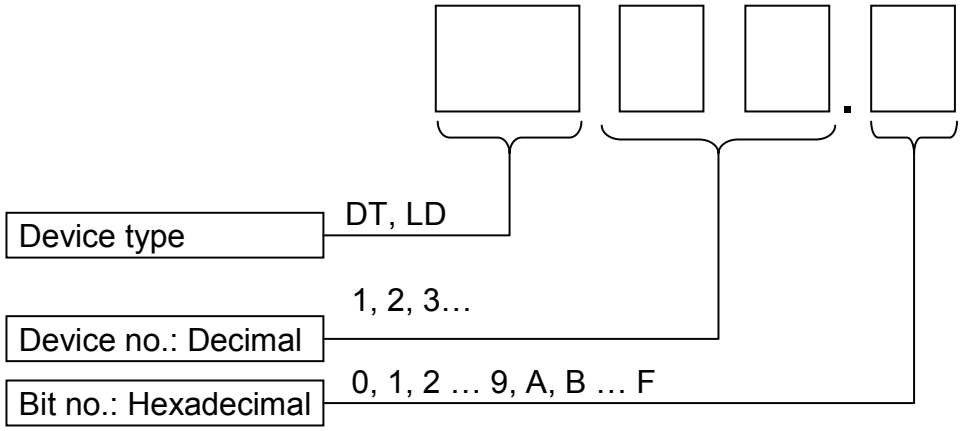
- The numbers of timer/counter contacts correspond to the numbers of timers/counters, and should be specified by decimal numbers.
- The numbers of error alarm relays should be specified by decimal numbers.



T 0,T 1 ····· T4095
C 0,C 1 ····· C1023
E0, E1 ····· E4095

■ Bit specification of word devices (DT*.n, LD*.n)

- As for data register DT and link register LD, specific bits in 16-bit data can be extracted and used as bit data, by specifying the device type, device number, and bit number.
- The device number and the bit number should be separated by a period (.).
- See explanations for respective instructions to check available instructions.



[Example] Word device DT＊.n

DT0.0, DT0.1, ------- DT0.F,

DT10.0, DT10.1, ---- DT10.F

DT20.0, DT20.1, ---- DT20.F

⋮

■ How to count memory area numbers

The numbers of respective memory areas (e.g. data register DT) should be specified by decimal numbers. (This excludes index registers.)

[Example] Data register DT

DT0, DT1,----------DT9

DT10, DT11, -------DT19

⋮

1-5 Explanations about Relays

X External input

■ Functions of external input (X)

- External inputs are used to feed in signals sent from external input devices (e.g. limit switch, photoelectric switch) connected to input points.

■ Restrictions on use

- External inputs that are not allocated cannot be used.
- The on or off state of external inputs cannot be switched by programmable operation.
- There are no restrictions on the number of times one external input relay is programmed.

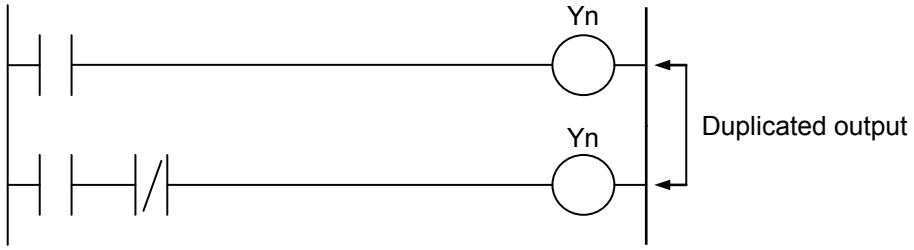
Y External output

■ Functions of external output (Y)

- External outputs are used to send the results of program operation as signals, to the loads connected to output points (e.g. solenoids, displays). The on or off state of an external output is sent as a control signal.

■ Restrictions on use

- External outputs that are not allocated can be used as internal relays. (However, they cannot be specified for hold-type devices.)
- When used as contacts, there are no restrictions on the number of times that can be used.
- As a rule, when this type of relay is specified as an output destination for operation results of OT and KP instructions, its use is limited to once in a program. (Inhibition of duplicate output)
- Duplicate output may be permitted through "Select operation" > "Duplicate output authorization" in the "CPU configuration" dialog box using the tool software FPWIN GR7.
- Use in the SET or RST instruction is not regarded as duplicate output.



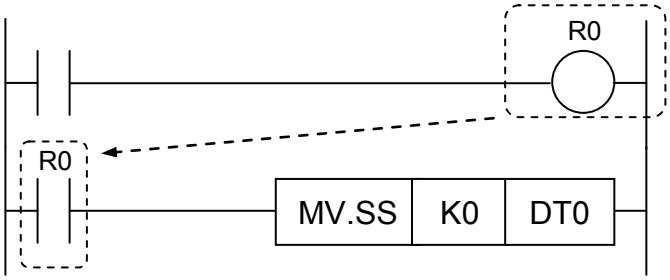
◆ **NOTE**

- **Even when a project total check is conducted using the tool software FPWINGR7, the instruction used at the start is not indicated. Instead, the second and later outputs that are regarded as duplicate use are indicated.**

R Internal relay

■ Functions of internal relay (R)

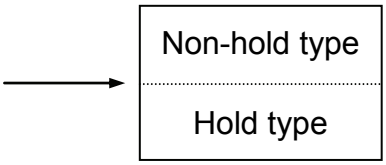
- An internal relay only operates within a program. Its on or off state is not externally output.
- Once the operation result is output and switched on (Coil: ON), the same relay used as a contact is also switched on.



■ Hold relay and non-hold relay

- Internal relays are categorized into two types, depending on their operation following powering off or switching from the RUN to PROG. mode:
  1) "Hold relays", which memorize the on or off state immediately before stopping, and can restart operation in the same state following restoration; and
  2) "Non-hold relays", where the state is reset following a stop.
- Hold relays and non-hold relays can be set through "Setting of the no. of local devices to be used (in total)" > "Global hold-type start no." in the "Memory configuration" dialog box using the tool software FPWIN GR7.
- If the beginning of a hold type relay is specified using a word number, relays before that point will be non-hold types, and subsequent relays will be hold types.
- Setting of hold type and non-hold type is available for both global devices and local devices.

"Memory configuration" >
"Setting of the no. of local devices to be used (in total)" >
"Global holding start number"



■ Restrictions on use

- When used as contacts, there are no restrictions on the number of times that can be used.
- As a rule, when this type of relay is specified as an output destination for operation results of OT and KP instructions, its use is limited to once in a program. (Inhibition of duplicate output)
- Duplicate output may be permitted through "Select operation" > "Duplicate output authorization" in the "CPU configuration" dialog box. Use in the SET or RST instruction is not regarded as duplicate output.

SR System relay

■ Functions of system relay (SR)

- System relays turn on or off under specific conditions.
- Its on or off state is not externally output. It only operates within a program.

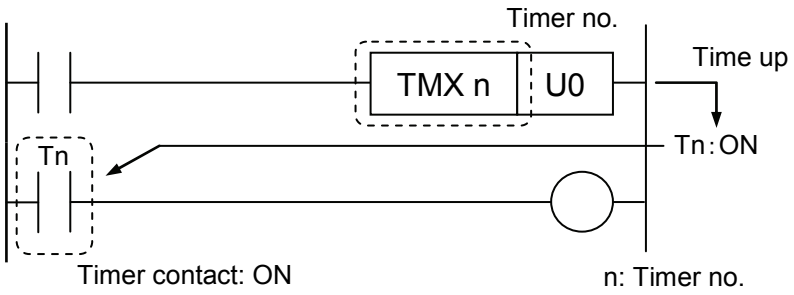
■ Types of system relay (SR)

Classification	Functions	Example
Operation status flag	Operation status is indicated by on or off.	Under operation (RUN) (SR20), under forced input/output (SR29), results of comparison instruction (SRA to SRC)
Error flags	Turn on when an error occurs and notify abnormality.	Operation errors (SR7, SR8)
Relays that turn on or off under specific conditions	Can be used with selected necessary conditions in a program.	Constantly ON relay (SR10), relay that turn ON/OFF in every scan (SR12) Initial pulse relay (SR13 to SR14), clock pulse (SR18 to SR1E)

T Timer

■ Functions of timer (T)

- When a timer is activated and the set time elapses, the timer contact with the same number as the timer turns on.
- When the timer is in the time-up state and the timer execution condition turns off, the timer contact turns off. All timers are categorized as non-hold type.



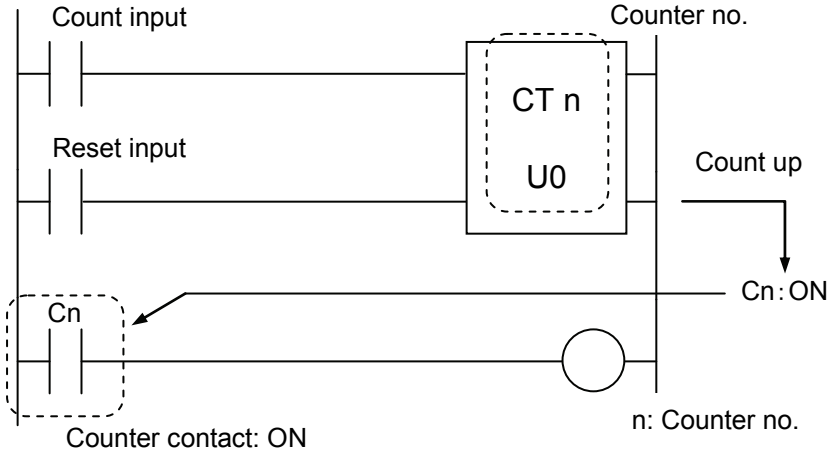
■ Restrictions on use

- When used as contacts, there are no restrictions on the number of times that can be used.

C Counter

■ Functions of counter (C)

- When the decrement-type preset counter is activated and the counter value reaches zero, the counter contact with the same number as the counter turns on.
- When the counter's reset input is turned on, the counter contact turns off.
- All counters are categorized as hold type.



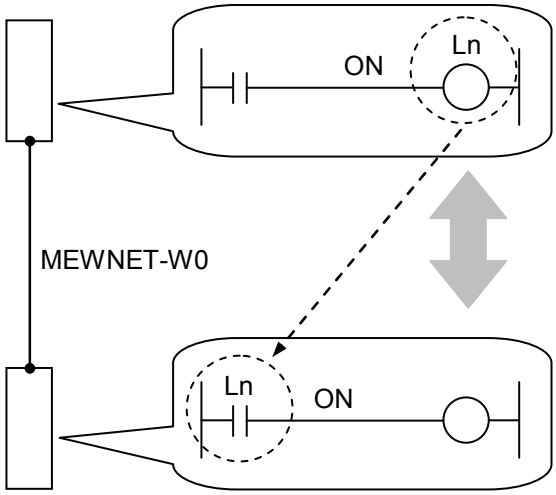
■ Restrictions on use

- When used as contacts, there are no restrictions on the number of times that can be used.

L Link relay

■ Functions of link relay (L)

- Link relays are shared relays used for a PLC link, when multiple controllers are connected via network.
- When an operation result is output to a link relay (coil) in a PLC, the information is sent to another PLC that is connected to the same network, and the operation result is reflected into the link relay (contact) of the same number.
- In this way, bit information can be exchanged between PLCs using a link relay.



■ Range of link relays used

- Range of link relays used varies by the type of network and the combination of units.
- The range of use and points should be specified for each network.

■ Setting of hold relay and non-hold relay

- Link relays are categorized into two types, depending on their operation following powering off or switching from the RUN to PROG. mode:
  1) "Hold relays", which memorize the on or off state immediately before stopping, and can restart operation in the same state following restoration; and
  2) "Non-hold relays", where the state is reset following a stop.
- The ranges of hold relays and non-hold relays can be set through "Setting of the no. of local devices to be used (in total)" > "Global hold-type start no." in the "Memory configuration" dialog box using the tool software FPWIN GR7.
- If the beginning of hold type relays is specified using a word number, relays before that point will be specified as non-hold type, and the subsequent relays will be specified as hold type. For example, if the "Global hold-type start no." is set at 10 under "Memory configuration" > "Setting of the no. of local devices to be used (in total)", then L0 to L9F are specified as non-hold type, and L100 to L63F as hold type.
- Note that, when link relays are used for reception, the retaining operation is not enabled even if they are specified as hold type in the system configuration.
- Setting of hold type and non-hold type is available for both global devices and local devices.

■ Restrictions on use

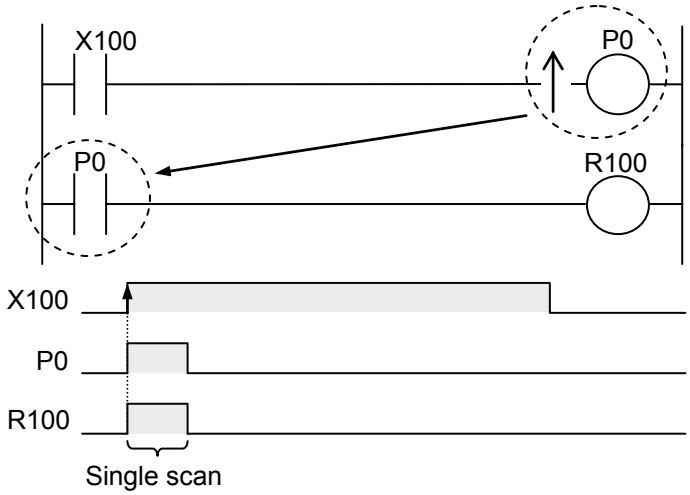
- When used as contacts, there are no restrictions on the number of times that can be used.
- As a rule, when this type of relay is specified as an output destination for operation results of OT and KP instructions, its use is limited to once in a program. (Inhibition of duplicate output)
- Duplicate output may be permitted through "Select operation" > "Duplicate output authorization" in the "CPU configuration" dialog box. Use in the SET or RST instruction is not regarded as duplicate output.

P Pulse relay

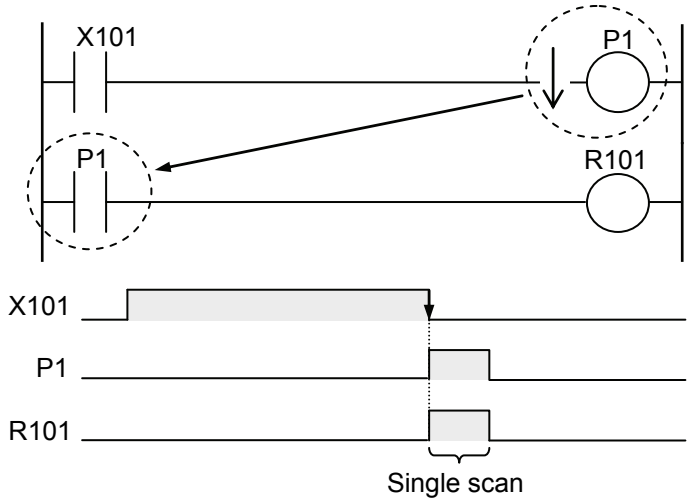
■ Functions of pulse relay (P)

- A pulse relay goes on for one scan only. Its on or off state is not externally output. It only operates within a program.
- A pulse relay only goes on when a leading edge start instruction (OT↑) or a trailing edge start instruction (OT↓) is executed.
- When used as the trigger, a pulse relay only operates during one scan when leading edge or trailing edge is detected.

[Example 1] Operates at the leading edge of Input X100



[Example 2] Operates at the trailing edge of Input X101



■ Restrictions on use

- Pulse relays are cleared when the power is turned off.
- As a rule, when this type of relay is specified as an output destination for an OT↑ or OT↓ instruction, its use is limited to once in a program.. (Inhibition of duplicate output)
- When used as contacts, there are no restrictions on the number of times that can be used.
- A pulse relay cannot be specified as an output destination for an OT, KP, SET, RST or ALT instruction.
- A word unit pulse relay (WP) cannot be specified as a storage location for a high-level instruction.

E Error alarm relay

■ Functions of error alarm relay (E)

- Error alarm relays are used to feed back error conditions freely assigned by the user to internal relays, and to store them in memory.
- Error alarm relays are turned on and off using the SET and RST instructions in the user program.
- When an error alarm relay goes on, the number of error alarm relays which are on, the relay numbers, and the data of the calendar timer which went on first are stored in a memory area in the CPU unit.

System data register SD no.	Description	
SD50, SD51	Year/month data	Calendar timer data when error alarm relays went on first
SD52, SD53	Day/hour data	
SD54, SD55	Minute/second data	
SD60	No. of error alarm relays in the ON state	
SD61 to SD79	Numbers of error alarm relays in the ON state	

- Information for up to 500 error alarm relays can be stored in the memory area. Relays that can be monitored or operated by the user, however, are those in the range from SD61 to SD79 only.

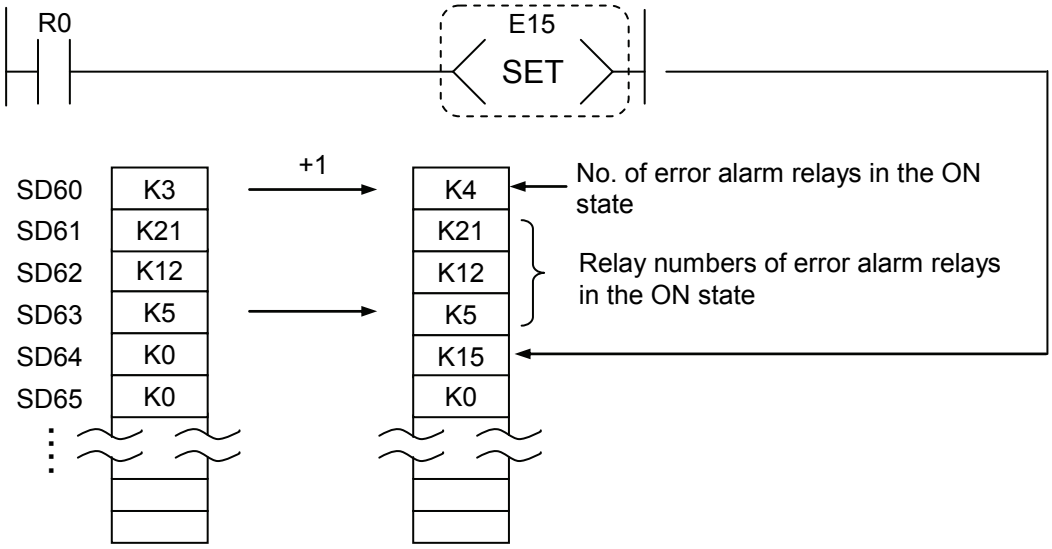
■ Restrictions on use

- An error alarm relay (E) cannot be specified as an output destination for an OT, KP, or ALT instruction.
- An error alarm relay (E) can be turned on and off in multiple locations in the program, using the SET and RST instructions. However, no check is carried out for duplicateuse.

■ Program for setting (turning on) an error alarm relays

- The SET instruction should be used as follows, to turn on error alarm relays in the error alarm conditions.
- Error alarm relays are held even if the error condition goes off.

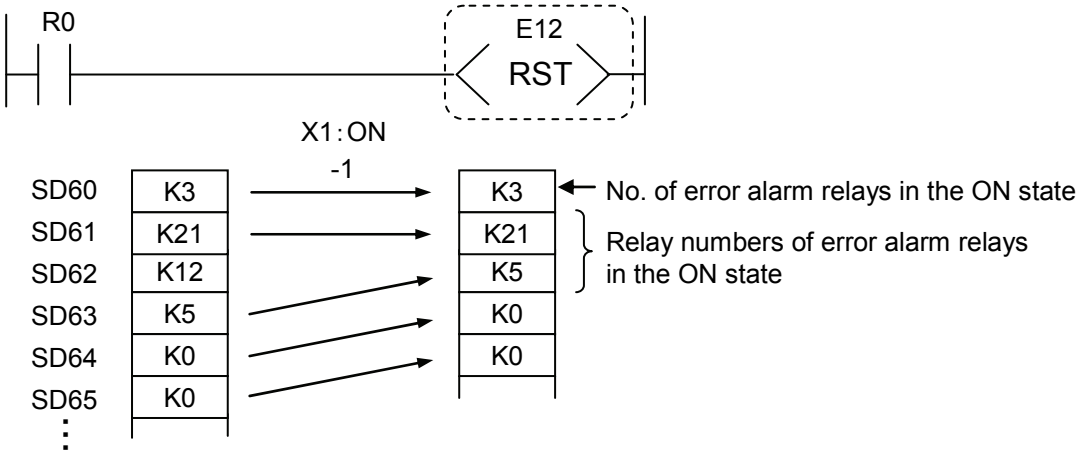
[Example] R0 goes on when an error occurs



■ Reset program for a given error alarm relay

- When an error has been corrected, the RST instruction should be used to turn off the error alarm relay.

[Example] R0 goes on when an error is corrected



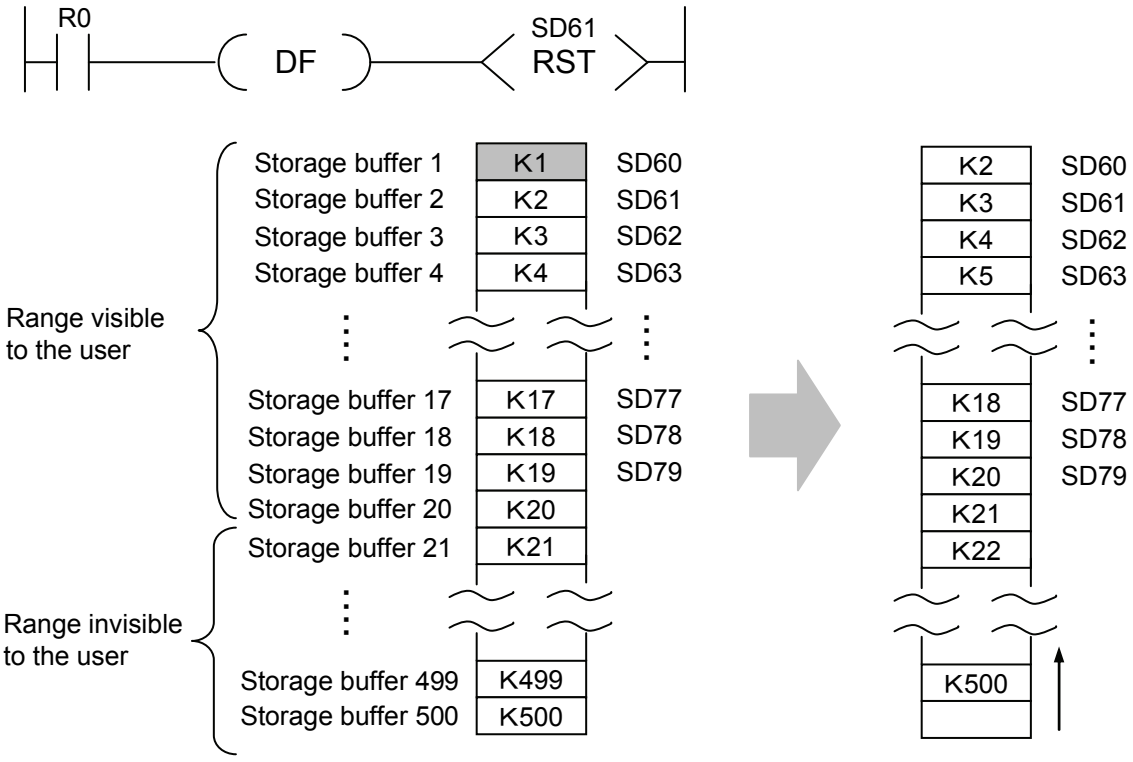
■ Clearing all buffer areas

- In order to reset all error alarm relays, specify system data SD60 using the RST instruction in the same method as described in the next page.

■ Clearing buffer areas and starting data

- Of the areas in which relay numbers are stored, only SD60 and SD61 can be cleared by directly specifying system data using with the RST instruction.
- If SD60 is specified, all error information in the buffer is cleared. If SD61 is specified, the relay no. at the start of buffer areas is cleared, and the buffers are carried up as indicated below.

[Example] Content of SD61 is deleted with the RST instruction



IN Direct input

■ Functions of direct input (IN)

- While all external inputs (X's) are read and updated at the execution of an input/output refresh, it is possible to read and update external inputs during execution of operation by using direct input (IN).
- This is useful in a control that requires high-speed response.
- Direct inputs should be specified in combination with slot numbers S1, S2, and so on in a program.

OT Direct output

■ Functions of direct output (OT)

- While all external outputs (Y's) are output at the execution of an input/output refresh, it is possible to output operation results (on or off) to the present point during execution of operation by using direct ouput (OT).
- This is useful in a control that requires high-speed response.
- Direct o should be specified in combination with slot numbers S1, S2, and so on in a program.

1-6 Explanations about Memory Areas

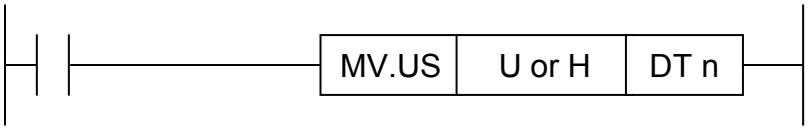
DT Data register

■ Functions of data register (DT)

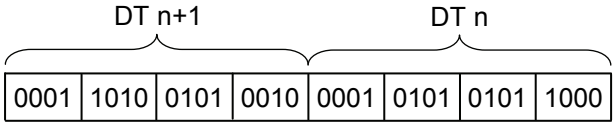
- Data registers are memory areas which are handled in word (16-bit) units, and are used to store data such as numerical data configured of 16 bits.

DTn | 0 0 0 1 | 1 0 1 0 | 0 1 0 1 | 1 0 0 0 |

[Example of program to write numerical values into DTn]



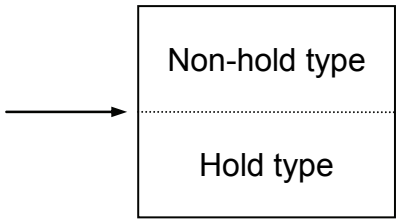
- When 32-bit (double word) data is handled in data registers, use two data registers as a set. In a program, the number of the data register for the lower 16 bits is specified.



■ Hold type data and non-hold type data

- Data registers are categorized into two types, depending on their operation following powering off or switching from the RUN to PROG. mode:
  1) "Hold type data registers", which memorize the on or off state immediately before stopping, and can restart operation in the same state following restoration; and
  2) "Non-hold type data registers", where the state is reset following a stop.
- Setting of hold type and non-hold type is available for both global devices and local devices.
- Hold type and non-hold type can be set through "Setting of the no. of local devices to be used (in total)" > "Global hold-type start no." in the "Memory configuration" dialog box using the tool software FPWIN GR7. If the beginning of hold type data registers is specified using a word number, data registers before that point will be specified as non-hold type, and the subsequent data registers will be specified as hold type.

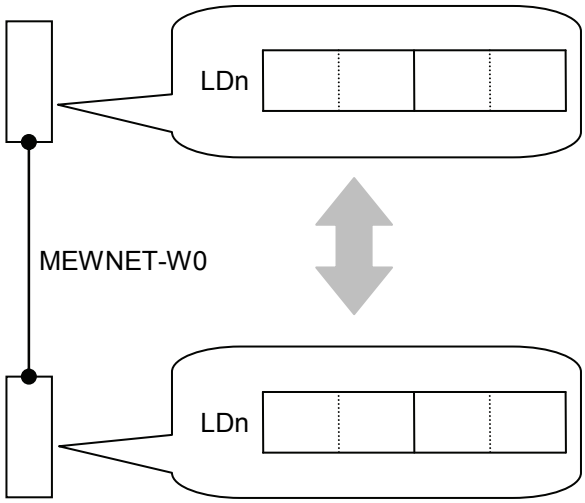
"Memory configuration" >
"Setting of the no. of local devices to be used (in total)" >
"Global holding start number"



LD Link register

■ Functions of link register (LD)

- Link registers are shared data memories used for a PLC link, when multiple control units are connected via network.
- When data are written into a link register in a PLC, the information is stored in the link register of the same number in another PLC that is connected to the same network.
- In this way, data can be exchanged between PLCs only by writing data, if a link register is used.



■ Range of link registers

- Range of link registers used varies by the type of network and the combination of units.
- The range of use and points should be specified for each network.

■ Setting of hold type register and non-hold type register

- Link registers are categorized into two types, depending on their operation following powering off or switching from the RUN to PROG. mode:
  1) "Hold type registers", which retain the content immediately before stopping, until operation is restarted; and
  2) "Non-hold type registers", where the state is reset following a stop.
- The ranges of hold type and non-hold type can be set through "Setting of the no. of local devices to be used (in total)" > "Global hold-type start no." in the "Memory configuration" dialog box.
- Note that, when link registers are used for reception, the retaining operation is not enabled even if they are specified as hold type in the configuration.
- Setting of hold type and non-hold type is available for both global devices and local devices.

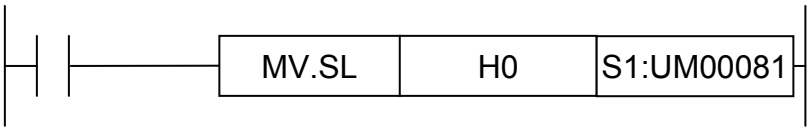
UM Unit memory

■ Functions of unit memory (UM)

- Unit memories are used when data are exchanged between CPU/high-function units.
- Data amount and allocation of unit memories vary by the type of unit.
- Unit memory addresses should be specified by five-digit hexadecimal numbers. (H 0 to H 7FFFF (Max.))
- Unit memories should be specified in combination with slot numbers S1, S2, and so on in a program.

■ Example of use of unit memory (UM)

Constant H0 is transferred to H 00081, which represents the address of a unit memory (UM) for Slot No. 1.



SD System data

■ Functions of system data (SD)

- System data represent memory areas that store specified contents.
- They are divided into call-only areas, call-and-write areas, and areas used by the system.
- Major types of system data are as follows.

■ Types of system data (SD)

Classification	Functions
Environmental settings, operation status	The operation statuses of PLC, specified with configuration data and various types of instructions, are stored. Example) Scan time
Error contents	Information of units where errors occurred, etc. are stored. Example) Self-diagnostic error codes, slot numbers of units where errors occurred, addresses where operation errors occurred
Calendar timer	The year, month, day, hour, minute, second, and day of the week tracked by the calendar timer are stored here.

WX, WY, WR, WL, WI, WO

■ Functions of WX, WY, WR, WL, WI, WO

- Relays (X, Y, R, L, IN, OT) can be handled as blocks of 16 points.
- Pulse relays (P) and error alarm relays (E) cannot be handled in word units.
- These are one-word (16-bit) memory areas, thus they can be treated as data memory.
- The composition of the one-word memory areas is as follows. Numbers correspond to the respective areas.

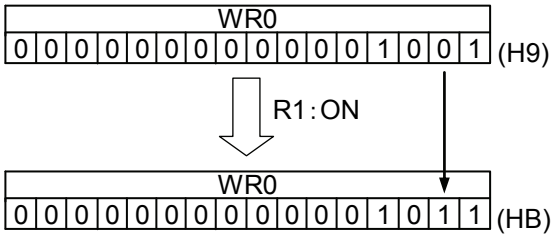
WR0																
	RF	RE	RD	RC	RB	RA	R9	R8	R7	R6	R5	R4	R3	R2	R1	R0
WR1																
	R1F	R1E	•	•	•	•	•	•	•	•	•	•	•	R12	R11	R10
WR2																
	R2F	R2E	•	•	•	•	•	•	•	•	•	•	•	R22	R21	R20

■ Example of use of WX, WY, WR, WL, WI, WO

- WX and WY can be used for reading inputs from high-function units and for external input/output in word units.
- WR can also be used as a shift register.
- All of these relays can be used to monitor 16-bit words.

■ Precautions on use

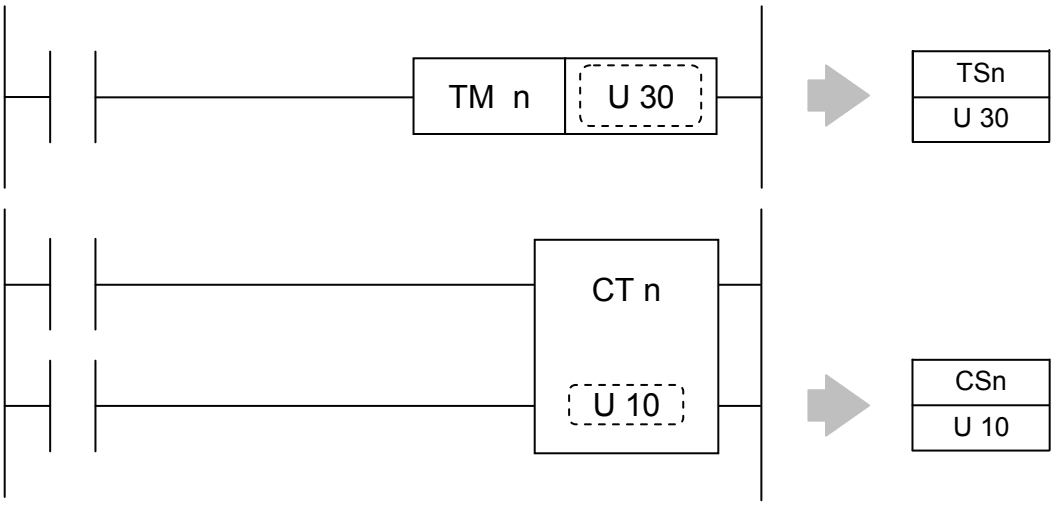
- If an on or off status of one of the relays composing the memory is changed, the memory area value will also be changed.



TS, CS Timer/Counter set value register

■ Functions of set value area (TS, CS)

- Timer set value (TS) and counter set value (CS) are stored in the set value area with the same number as the relevant timer and counter.
- A decimal number or the set value area (TS/CS) number should be used when the TM or CT instruction is entered in the program.
- A set value area (TS/CS) is a 32-bit memory area which stores a decimal number from 0 to 4,294,967,295.



■ Making use of set value area (TS, CS)

- Even during the RUN mode, a set value for a timer or counter can be changed by rewriting the corresponding set value area.
- The value of a set value area can be rewritten using a user program (e.g. data transfer instruction).
- It can also be read or written using a programming tool.

■ Reference:

- TS and TE corresponds with a timer number, and CS and CE with a counter number, on a one-on-one basis.

Timer/counter no.	Set value area	Elapsed value area
T0 T1 ⋮ T4095	TS0 TS1 ⋮ TS4095	TE0 TE1 ⋮ TE4095
C0 C1 ⋮ C1023	CS0 CS1 ⋮ CS1023	CE0 CE1 ⋮ CE1023

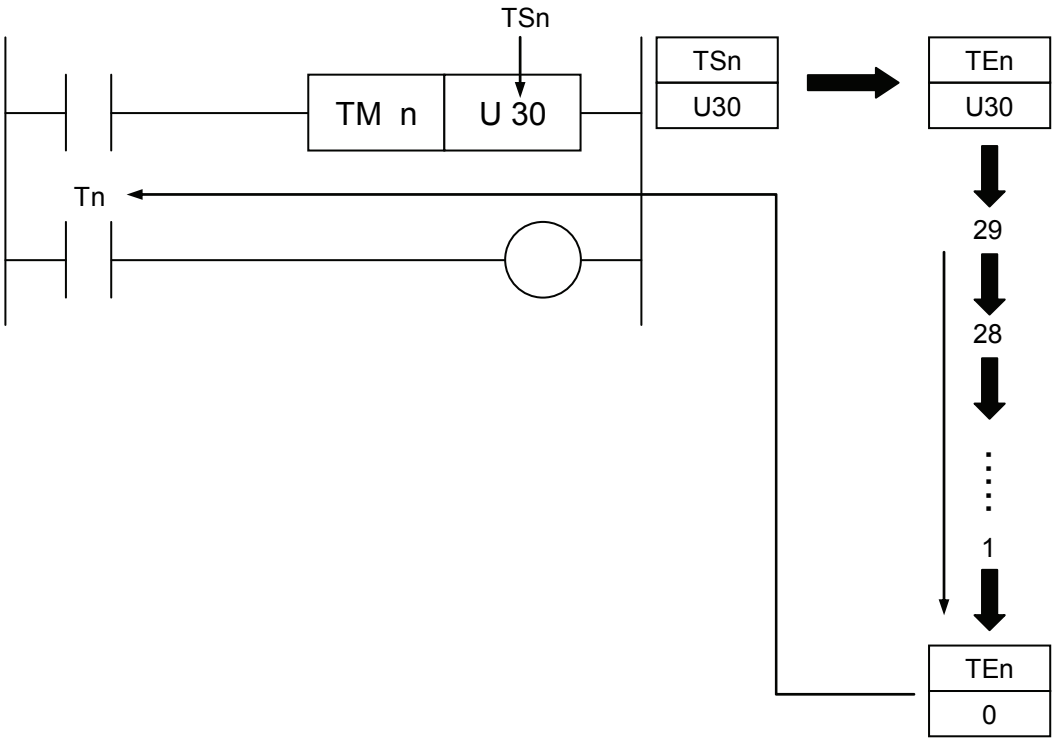
◆ **NOTES**

- **Timer/counter settings for the FP7 series should be specified with unsigned constants (U constants).**
- **The set value area (TS, CS) occupies a 32-bit area.**

TE, CE Timer/Counter elapsed value register

■ Functions of elapsed value area (TE, CE)

- While a timer or counter is operating, the elapsed value is stored in the elapsed value area (TE/CE) with the same number as the timer or counter.
- When the value of the elapsed value area (TE/CE) reaches zero, the timer or counter contact with the same number turns on.
- An elapsed value area (TE/CE) is a 32-bit memory area which stores a decimal number from 0 to 4,294,967,295.



■ Making use of elapsed value area (TE, CE)

- The elapsed value of a timer or counter in operation can be changed to prolong or shorten the operation.
- The value of a elapsed value area can be rewritten using a user program (e.g. data transfer instruction).
- It can also be read or written using a programming tool.

■ Precautions during programming

- Timer/counter settings for the FP7 series should be specified with unsigned constants (U constants).

◆ NOTE

- **The elapsed value area (TE, CE) occupies a 32-bit area.**

I0 to IE Index register

■ Functions of index register

- Index registers are used to indirectly specify constants and memory area ddresses.
- Operation to change an address or constant using the value of index register is called "index modification".
- Fifteen 32-bit registers are available (I0 to IE).

■ Precautions for the use of index register

- If the result of address modification exceeds the size of the relevant memory area, an operation error occurs.
  Example) A negative or excessively large value for an address results from modification

■ Items that can be modified with an index register

- No. of memory area used for a basic instruction
- Slot no. and memory area no.
- No. of memory area used for a high-level instruction
- Value of constant specified for a high-level instruction
- K constant (16 bits, 32 bits)
- U constant (16 bits, 32 bits)
- H constant (16 bits, 32 bits)

■ Items that cannot be modified with an index register

- Floating point data
- Some instructions may not be index modifiable. Check the section "Types of available memory areas" in the explanation pages for respective instructions.

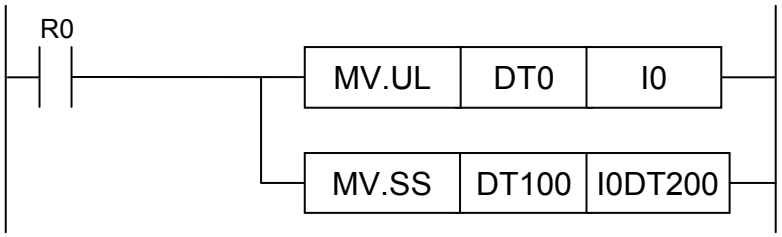
◆ **NOTE**

- **The index register (I0 to IE) occupies a 32-bit area.**

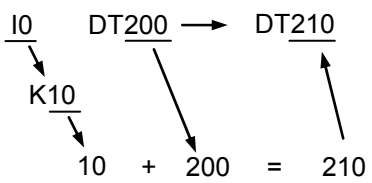
■ Methods for index modification

[Example 1] Modifying a transfer destination

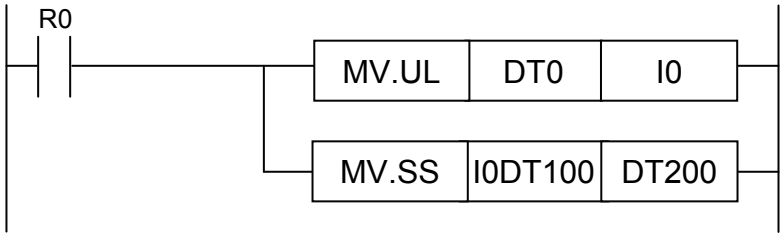
The address of destination data register DT varies by the value of DT0.



Example) If the value of DT0 is K10, the value of DT100 is written into DT210.

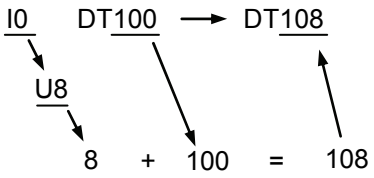


[Example 2] Modifying a transfer source



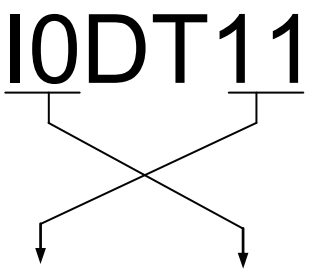
The address of source data register DT varies by the value of DT0.

Example) If the value of DT0 is U8, the value of DT108 is transferred to DT200.



■ Modification of address

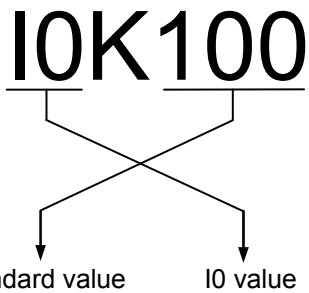
Address = Standard address + Value of I0 to IE



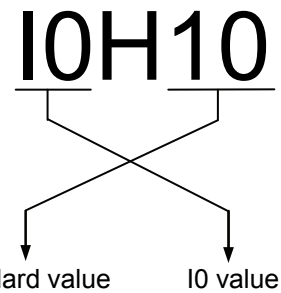
Standard address		I0 value		Targeted address
11	+	K0	=	DT11
11	+	K10	=	DT21
11	+	K-10	=	DT1

■ Modification of constant

Constant = Standard value + Value of I0 to IE



Standard value		I0 value		Constant
K100	+	K0	=	K100
K100	+	K10	=	K100
K100	+	K-10	=	K90

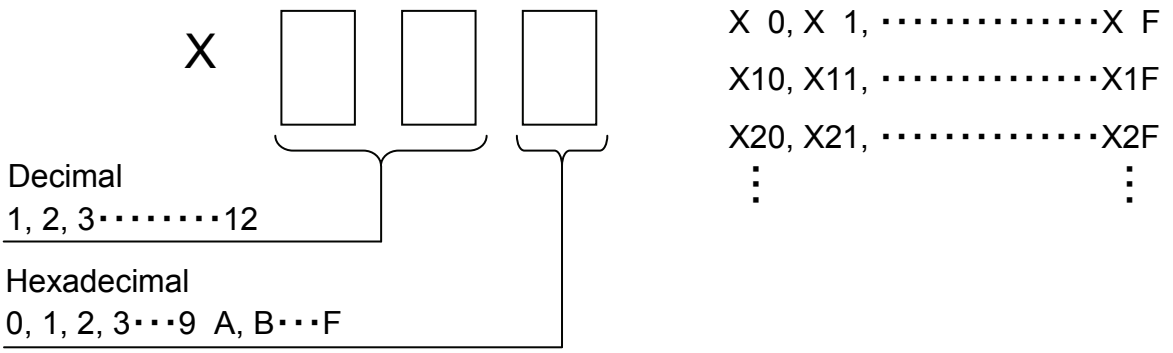


Standard value		I0 value		Constant
H10	+	HA	=	H1A
H10	+	H10	=	H20

■ Precautions for index modification of relay numbers

- For the external input relay (X), external output relay (Y), and internal relay (R), when using index modification on relay numbers, be aware that the last digit of the relay number is hexadecimal while higher digits are decimal.

[Example] External input (X)



■ Examples of index modification

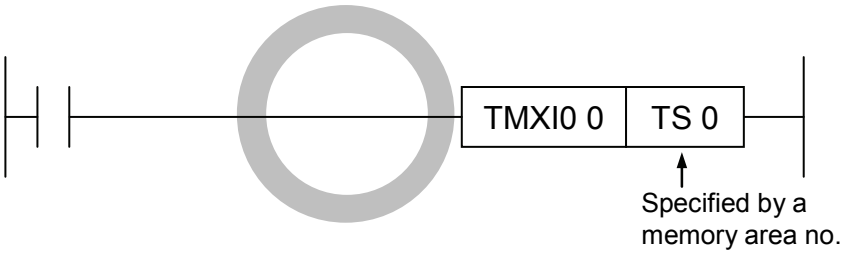
Value of index register		Post-modification relay no.
K	H	
0	0	X0
1	1	X1
⋮	⋮	⋮
9	9	X9
10	A	XA
⋮	⋮	⋮
15	F	XF
16	10	X10
⋮	⋮	⋮
31	1F	X1F
⋮	⋮	⋮
159	9F	X9F
160	A0	X100
161	A1	X101
⋮	⋮	⋮
255	FF	X15F
256	100	X160
257	101	X161
⋮	⋮	⋮
265	10A	X16A
267	10B	X16B
⋮	⋮	⋮

■ Modification of basic instruction no.

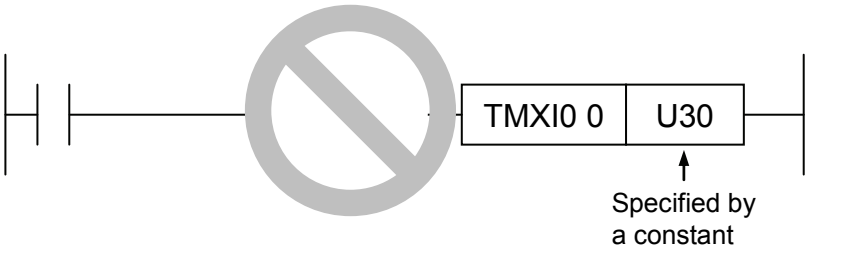
Object	Example of modification	
Timer no.	Modify TML20	TML I020
Counter no.	Modify CT200	CT I0200
Shift register no.	Modify SRWR0	SR I0WR0
Label no. specification in a jump instruction	Modify JP1	JP I01
Label no. specification in a loop instruction	Modify LOOP5	LOOP I05
Subroutine program no.	Modify CALL10	CALL I010

■ Restrictions on modification of timer/counter no.

- Timer numbers and counter numbers can be modified only when a memory area is specified for the set value.



- Modification cannot be done if the set value is specified with a constant.



1-7 Explanations about Constants

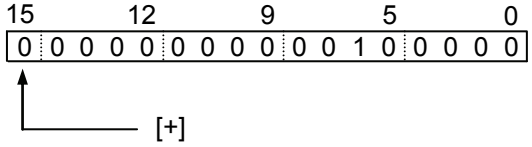
K Signed decimal constant

■ Functions of signed decimal constant (K)

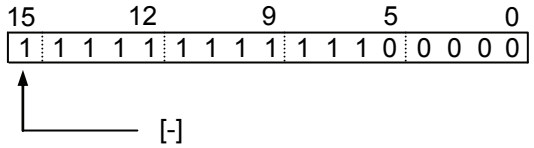
- This is binary data that have been converted into a decimal format. When entering a decimal constant, specify the value by entering a "K" at the beginning.
- Decimal constants are primarily used to specify data sizes and quantities.
- Inside the PLC, the decimal constant (K) is processed as BIN data in units of 16-bit words, as shown below.
- The positive/negative sign is determined by the prefix bit (sign bit). A “0” indicates a positive sign (+), and a “1” indicates a negative sign (-).
- Data are normally handled in units of one word (16 bits). However, they may also be handled in units of two words (32 bits). In this case, as well, the prefix bit serves as the sign bit.

■ Format of signed decimal constant (K)

[Example] Decimal "+32" (K32)



[Example] Decimal "-32" (K-32)



■ Range that can be specified using a decimal constant (K)

Operation	Available range
16-bit operation	K-32,768 to K32,767
32-bit operation	-2,147,483,648 to K2,147,483,647

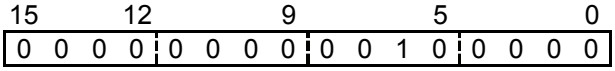
U Unsigned decimal constant

■ Functions of unsigned decimal constant (U)

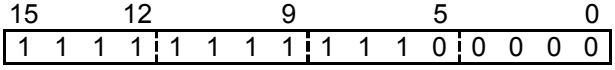
- This is binary data that have been converted into a decimal format. When entering and reading a decimal constant, specify the value by entering a "U" at the beginning.
- Decimal constants are primarily used to specify data sizes and quantities such as set values for timer.
- Inside the PLC, the decimal constant (U) is processed as BIN data in units of 16 bits, as shown below.
- Because an unsigned constant (U) represents a value with 16 bits only, it cannot handle a negative value.
- Data are normally handled in units of one word (16 bits). However, they may also be handled in units of two words (32 bits).

■ Format of unsigned decimal constant (U)

[Example] Decimal "+32" (U32)



[Example] Decimal "+65504" (U65504)



■ Range that can be specified using a decimal constant (U)

Operation	Available range
16-bit operation	U0 to U65,535
32-bit operation	U0 to U4,294,967,295

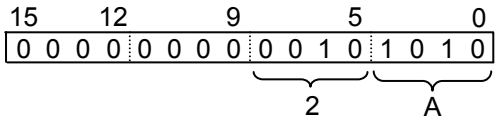
H Hexadecimal constant

■ Functions of hexadecimal constant (H)

- This is binary data that have been converted into a hexadecimal format. When entering or reading a hexadecimal constant, specify the value by entering a "H" at the beginning.
- Hexadecimal constants are primarily used to specify an ordering of 1's and 0's in 16-bit data, such as system data settings and specification of control data for high-level instructions. Hexadecimal constants are also used to specify BCD data.
- Inside the PLC, the hexadecimal constant (H) is processed as BIN data in units of 16 bits, as shown below.
- Data are normally handled in units of one word (16 bits). However, they may also be handled in units of two words (32 bits).

■ Format of hexadecimal constant (H)

[Example] Hexadecimal "2A" (i.e. H2A)



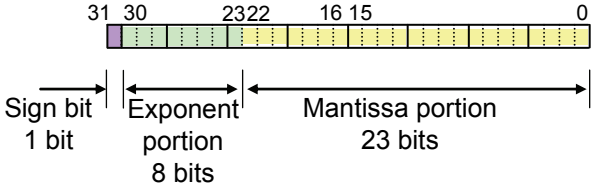
■ Range that can be specified using a hexadecimal constant (H)

Operation	Available range
16-bit operation	H0 to HFFFF
32-bit operation	H0 to HFFFFFFFF

SF Single precision floating point real number constant

■ Format of single precision floating point data

- In accordance with IEEE754, the single precision floating point format consists of 1 sign bit, 8 exponential bits, and 23 mantissa bits.
- Because operation instructions for handling real numbers, as well as instructions for conversion between real number integers, are available, it is not necessary to take data formats into account while programming.



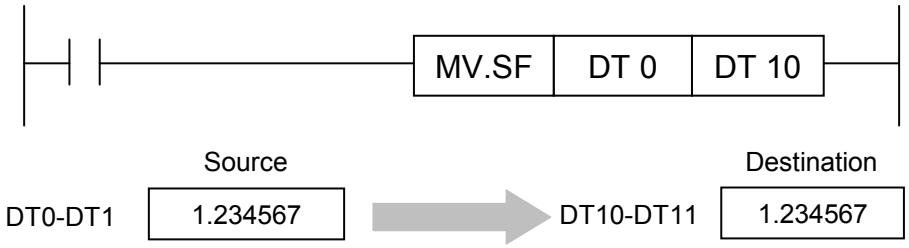
■ Range of single precision floating point real number constants

Operation	Available range
32-bit operation	Negative range: -1.175494E-38 to -3.402823E+38 0 Positive range: 3.402823E+38 to 1.175494E-38

■ Storage area for single precision floating point real number constants

- A storage area for data converted into real number, using the single precision floating point real number constant operation instruction, uses a 2-word, 32-bit area for each datum.
- In order to manipulate an area that stores real number data in a transfer instruction, etc., specify SF as the operation unit.

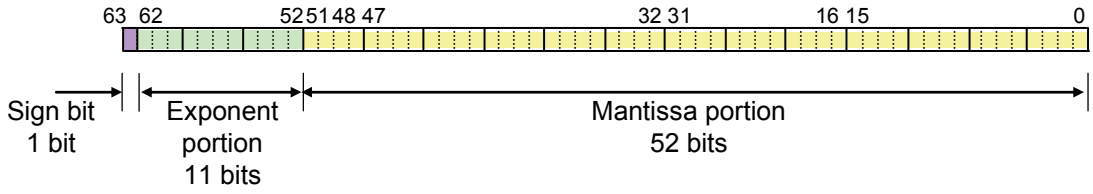
[Example] If SF is specified as the operation unit in an instruction code, two-word data are entered into DT.



DF Double precision floating point real number constant

■ Format of double precision floating point data

- In accordance with IEEE754, the double precision floating point format consists of 1 sign bit, 11 exponential bits, and 52 mantissa bits.
- Because operation instructions for handling real numbers, as well as instructions for conversion between real number integers, are available, it is not necessary to take data formats into account while programming.



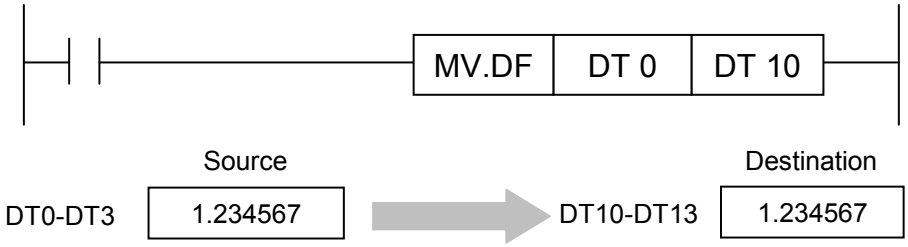
■ Range of double precision floating point real number constants

Operation	Available range
64-bit operation	Negative range: -2.2250738585072014E-308 to -1.7976931348623158E+308 0 Positive range: +2.2250738585072014E-308～+1.7976931348623158E+308

■ Storage area for double precision floating point real number constants

- A storage area for data converted into real number, using the double precision floating point real number constant operation instruction, uses a 4-word, 64-bit area for each datum.
- In order to manipulate an area that stores real number data in a transfer instruction, etc., specify DF as the operation unit.

[Example] If DF is specified as the operation unit in an instruction code, four-word data are entered into DT.



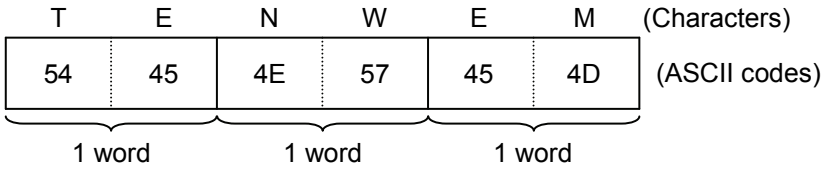
"" Character constant

■ Functions of character constant ("")

- A character constant handles binary data as an ASCII code. When entering a character constant, bracket the relevant character with "".
- Setting of a character constant is possible using the SSET (Conversion: Character Constant → ASCII Code) instruction. This instruction can only be entered using the tool software.
- Inside the PLC, the character constant ("") is stored as BIN data in the specified memory area, as shown below.

■ Format of character constant ("")

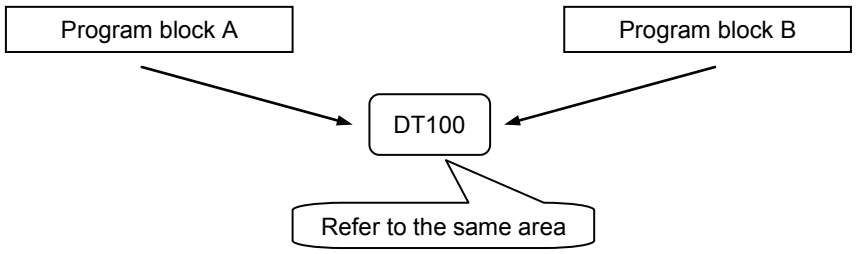
[Example] Enter a character constant "MEWNET"



1-8 Global Devices and Local Devices

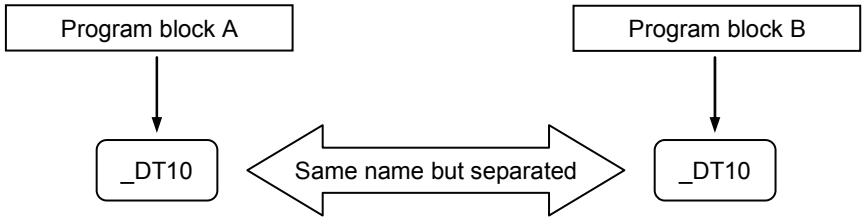
■ Global devices

- While a global device has a unique number throughout the entire program, a local device has a unique number inside each program block.
- For example, "Global device DT100" refers to the same data memory in program block A or in program block B. Therefore, data of DT100 are overwritten if the relevant information is revised in either A or B.
- A global device is used for shared use of multiple program blocks (PBs) and/or operation memories (e.g. external input/output).



■ Local devices

On the other hand, "local device dt10" refers to differing data memories in program block A and in program block B. Therefore, data of dt10 are not overwritten even if information under the same device no. is revised in the other program block.



■ List of available memory areas for local device (●: available)

Bit device			16-bit device		32-bit device
X, Y, R, L, T, C	P, E, SR, IN, OT	DT.n, LD.n	WX, WY, WR, WL, DT, LD	SD, WI, WO, UM	TS, CS,TE, CE
●		●	●		●

1-9 Range of Data that can be Handled Inside the PLC

■ 16-bit data

Data handled inside the PLC (binary 16-bit)	Decimal conversion		Hexadecimal conversion
	Unsigned	Signed	
0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1	U32767	K32767	H7FFF
⋮	⋮	⋮	⋮
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1	U 1	K 1	H 1
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	U 0	K 0	H 0
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1	U65535	K -1	HFFFF
⋮	⋮	⋮	⋮
1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	U32768	K-32768	H8000

■ 32-bit data

Data handled inside the PLC (binary 32-bit)	Decimal conversion		Hexadecimal conversion
	Unsigned	Signed	
0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1	U2147483547	K2147483547	H7FFFFFFF
⋮	⋮	⋮	⋮
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1	U 1	K 1	H 1
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	U 0	K 0	H 0
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1	U4294967295	K -1	HFFFFFFFF
⋮	⋮	⋮	⋮
1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	U2147438648	K-2147438648	H80000000

■ Range of data that can be handled inside the PLC

Type of operation			Available operation range
Binary operation	16 bits	Unsigned	U 0 to U 65,535
		Signed	K -32,768 to K 32767
	32 bits	Unsigned	U 0 to U 4,294967295
		Signed	K -2,147,483,548 to K 2,147,483,54
BCD operation	16 bits	Unsigned	H 0 to H 9999
	32 bits	Unsigned	H 0 to H 99999999

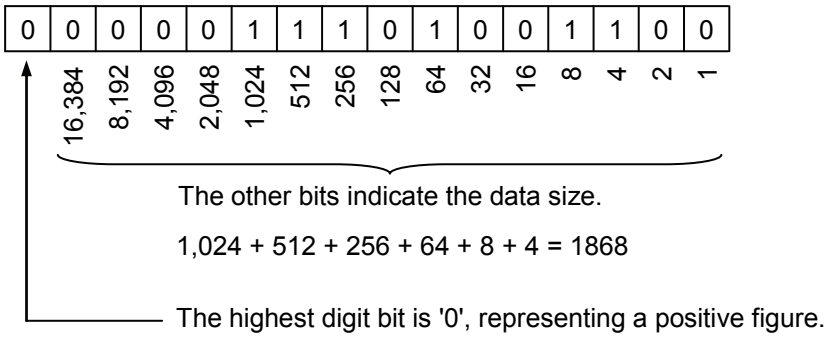
Note) In any of these cases, overflow or underflow results if the range specified above is not satisfied.

■ Representation of a decimal number inside the PLC

- A decimal number is handled as 16-bit or 32-bit binary data.
- Signed data

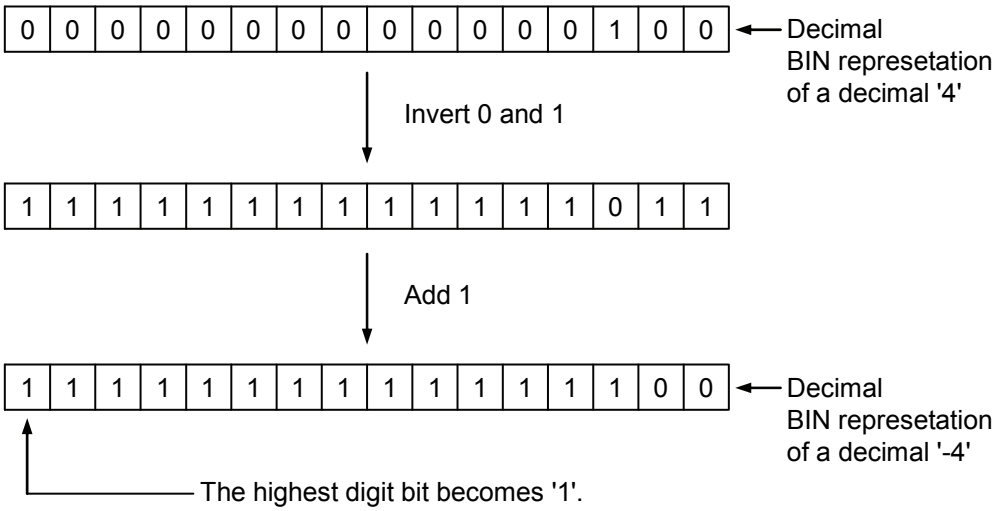
(1) The highest 1 bit is the sign bit. Its value is "0" for a positive number, and "1" for a negative number.
(2) In the case of a positive number, the other 15 bits represent the data size.

[Example] Representation of decimal "1868"



(3) A negative number is represented using a complement of "2". A complement of "2" is binary data given by inverting each of the BIN bits (0⇔1) for the corresponding positive value, and then adding 1 to the resulting data.

[Example] Representation of decimal "-4"



1-10 Overflow and Underflow

■ What are overflow and underflow?

- A value resulting from operation instruction sometimes fails to satisfy the range that can be handled.
- "Overflow" means that the max. value is exceeded, and "underflow" signifies that the min. value is not reached.

■ Overflow and underflow in binary operation

- When the following range is not satisfied, overflow or underflow occurs.

[16-bit operation]

(Overflow occurs if the max. value is exceeded)

	Unsigned		Signed	
Max. value	U65535	HFFFF	K32767	H7FFF
	⋮	⋮	⋮	⋮
	U32768	H8000	K 1	H 1
	U32767	H7FFF	K 0	H 0
	⋮	⋮	K -1	HFFFF
	U 1	H 1	⋮	⋮
Min. value	U 0	H 0	K-32768	H8000

(Underflow occurs if the min. value is not reached)

[32-bit operation]

(Overflow occurs if the max. value is exceeded)

	Unsigned		Signed	
Max. value	U4294967295	HFFFFFFFF	K2147483647	H7FFFFFFF
	⋮	⋮	⋮	⋮
	U2147483648	H80000000	K 1	H 1
	U2147483647	H7FFFFFFF	K 0	H 0
	⋮	⋮	K -1	HFFFFFFFF
	U 1	H 1	⋮	⋮
Min. value	U 0	H 0	K-2147483648	H80000000

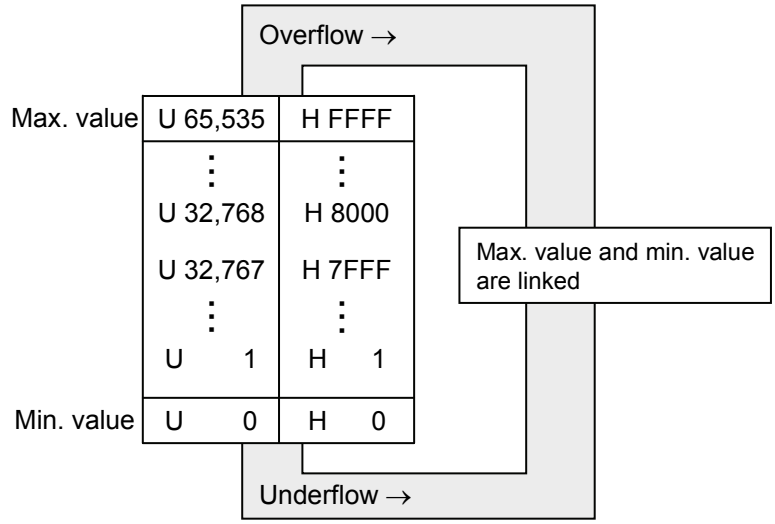
(Underflow occurs if the min. value is not reached)

■ Values at the time of overflow and underflow

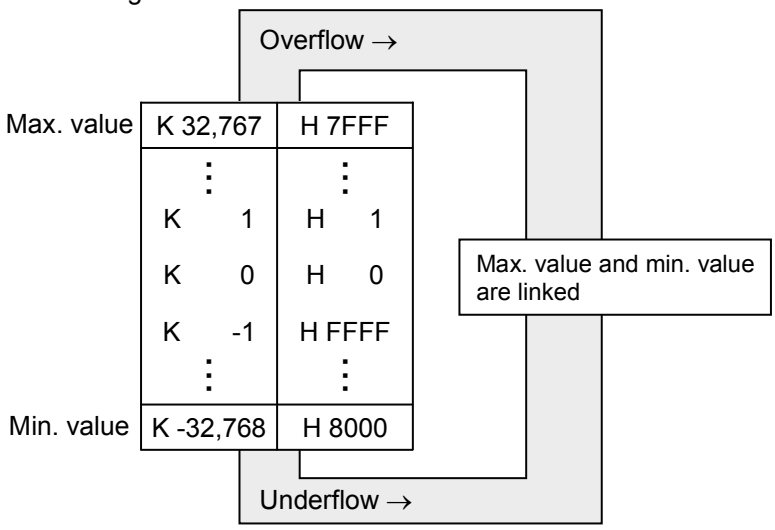
All values that can be handled during operation exist in a "loop" status, where the max. value and the min. value are linked, as indicated in the figures below.

Example 1) 16-bit binary operation

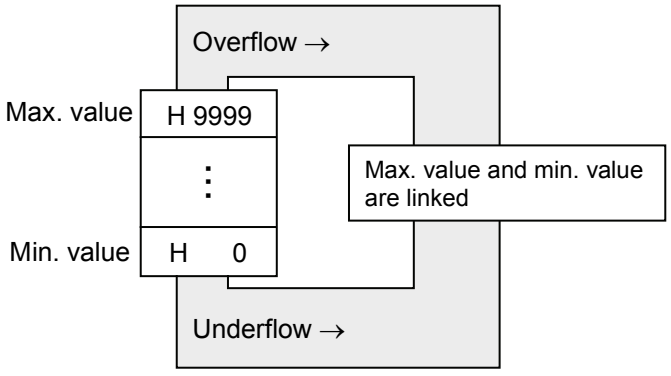
・Unsigned



・Signed



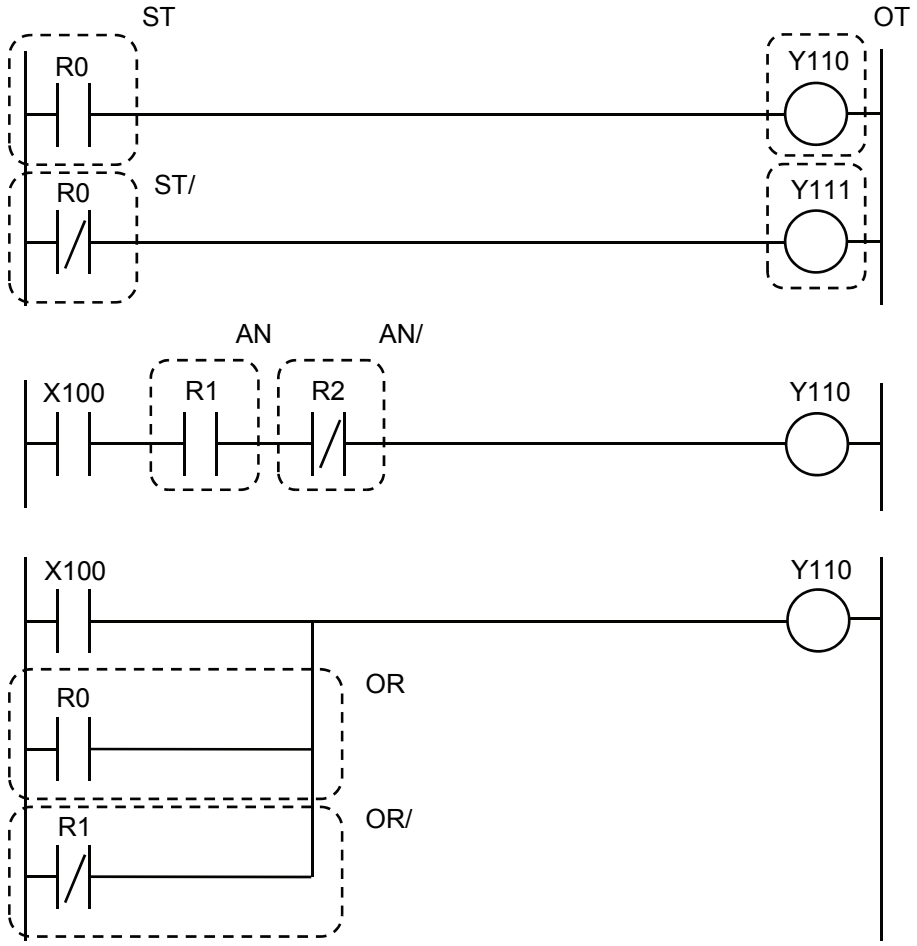
Example 2) 16-bit BCD operation



2
Basic instructions

ST, ST/, OT (START, START NOT, OUT)
AN, AN/ (AND, AND NOT)
OR, OR/ (OR, OR NOT)

■ **Ladder diagram**



■ **Available Devices (●: Available)**

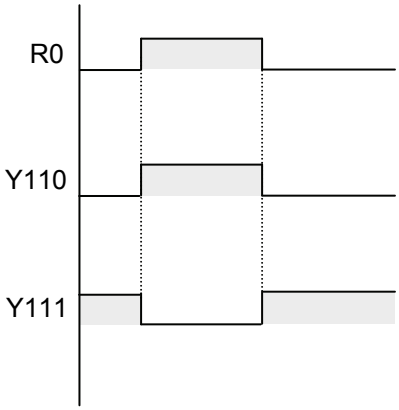
Operands		Bit device											Bit specification of word device		Index modifier
		X	Y	R	L	T	C	P	E	S	IN	OT	DT.b	LD.b	
ST ST/ AN, AN/ OR, OR/	bit	●	●	●	●	●	●	●	●	●	●	●	●	●	●
OT		●	●	●	●				●			●	●	●	●

■ Description

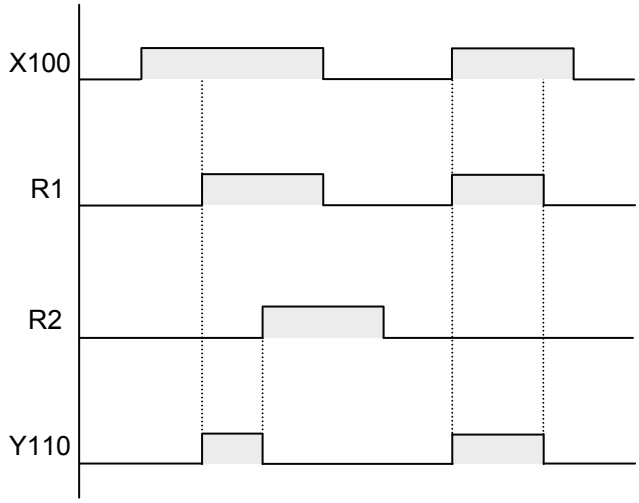
Type of instruction	Operation
ST	Begins a logic operation by treating the input contact specified by the ST instruction as a Form A (normally open).
ST/	Begins a logic operation by treating the input contact specified by the ST/ instruction as a Form B (normally closed).
AN	Executes an AND operation with the preceding operation result in serial connection by treating the input contact specified by the AN instruction as a Form A (normally open).
AN/	Executes an AND operation with the preceding operation result in serial connection by treating the input contact specified by the AN/ instruction as a Form B (normally closed).
OR	Executes an OR operation with the preceding operation result in parallel connection by treating the input contact specified by the OR instruction as a Form A (normally open).
OR/	Executes an OR operation with the preceding operation result in parallel connection by treating the input contact specified by the OR/ instruction as a Form B (normally closed).
OT/	The OT instruction outputs the operation result to the specified coil.

■ Example of Operation

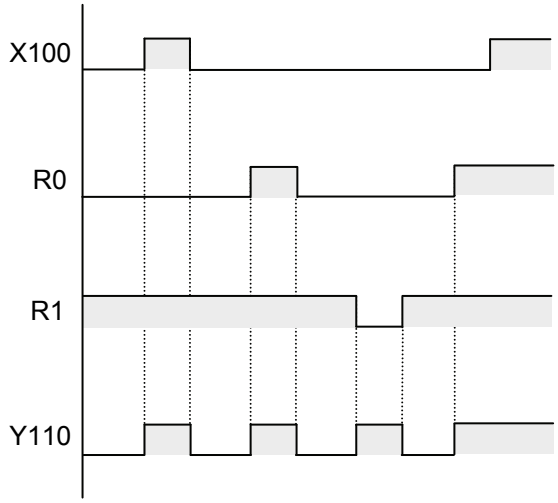
(1) Program operation for "ST", "ST/", and "OT" in ladder diagram
Outputs to Y110 when R0 is on and to Y111 when R0 is off.



(2) Program operation for "AN" and "AN/" in ladder diagram
Outputs to Y110 when X100 is on, R1 is on, and R2 is off.



(3) Program operation for "OR" and "OR/" in ladder diagram
Outputs to Y110 when X100 is on, R0 is on, or R1 is off.



■ Precautions during programming

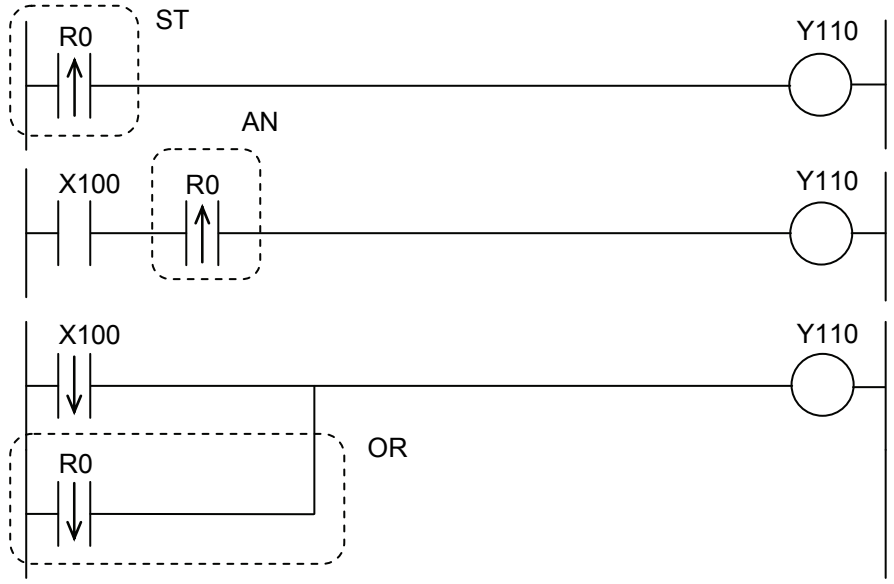
- The ST, ST/, OR, and OR/ instructions are initiated from the bus bar.
- The AN, AN/, OR, and OR/ instructions can be used in series.
- When an external switch (e.g., emergency stop switch) is a Form B (normally closed), be sure to use the ST instruction in the program.

ST↑, ST↓ (Leading and Trailing Contact Instructions)

AN↑, AN↓ (Leading and Trailing Contact Instructions)

OR↑, OR↓ (Leading and Trailing Contact Instructions)

■ Ladder diagram



■ Available Devices (●: Available)

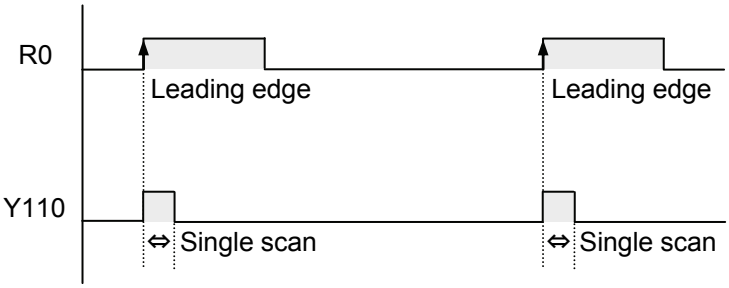
Operands	Bit device											Bit specification of word device		Index modifier
	X	Y	R	L	T	C	P	E	S	IN	OT	DT.b	LD.b	
bit	●	●	●	●	●	●	●	●	●	●	●	●	●	

■ Description

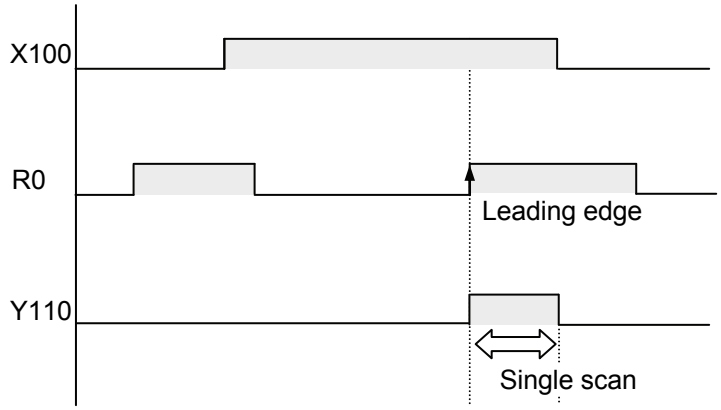
Type of instruction	Operation
ST↑	Continuity only exists for a single scan where a signal changes from the off state to the on state (i.e., rises). Begins a logic operation by treating the input contact as a Form A (normally open) or Form B (normally closed).
AN↑	Continuity only exists for a single scan where a signal changes from the off state to the on state (i.e., rises). Executes an AND operation with the preceding operation result in serial connection by treating the input contact as a Form A (normally open) or Form B (normally closed).
OR↑	Continuity only exists for a single scan where a signal changes from the off state to the on state (i.e., rises). Executes an OR operation with the preceding operation result in parallel connection by treating the input contact as a Form A (normally open) or Form B (normally closed).
ST↓	Continuity only exists for a single scan where a signal changes from the off state to the on state (i.e., rises). Begins a logic operation by treating the input contact as a Form A (normally open) or Form B (normally closed).
AN↓	Continuity only exists for a single scan where a signal changes from the off state to the on state (i.e., rises). Executes an AND operation with the preceding operation result in serial connection by treating the input contact as a Form A (normally open) or Form B (normally closed).
OR↓	Continuity only exists for a single scan where a signal changes from the off state to the on state (i.e., rises). Executes an OR operation with the preceding operation result in parallel connection by treating the input contact as a Form A (normally open) or Form B (normally closed).

■ Example of Operation

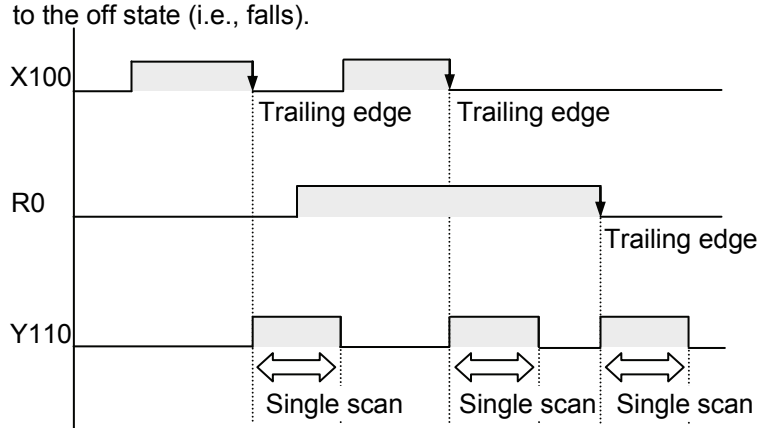
(1) Program operation for "ST↑" in ladder representation
Outputs to Y110 only for a single scan where R0 changes from the off state to the on state (i.e., rises).



(2) Program operation for "AN ↑" in ladder diagram
Outputs to Y110 only for a single scan when R0 changes from off to on (rises) while X100 is on.

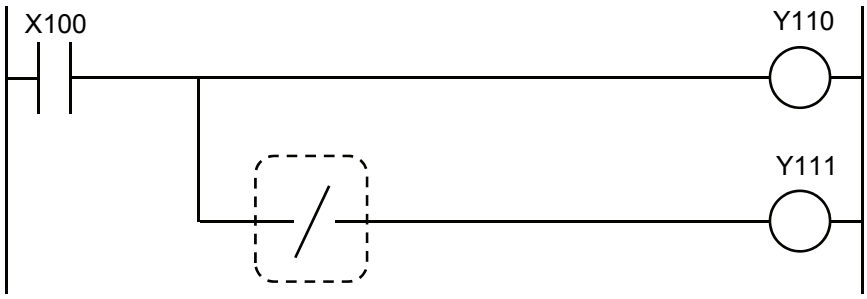


(3) Program operation for "OR ↑" in ladder diagram.
Outputs to Y110 only for a single scan where X100 or R0 changes from the on state to the off state (i.e., falls).



/ (NOT)

■ Ladder diagram

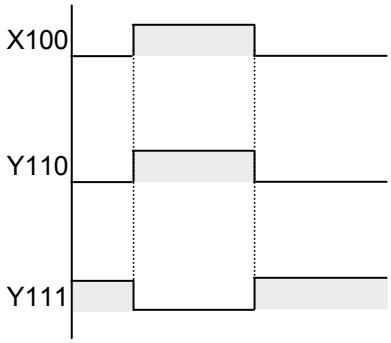


■ Description

- The / instruction inverts the preceding operation result.

■ Example of Operation

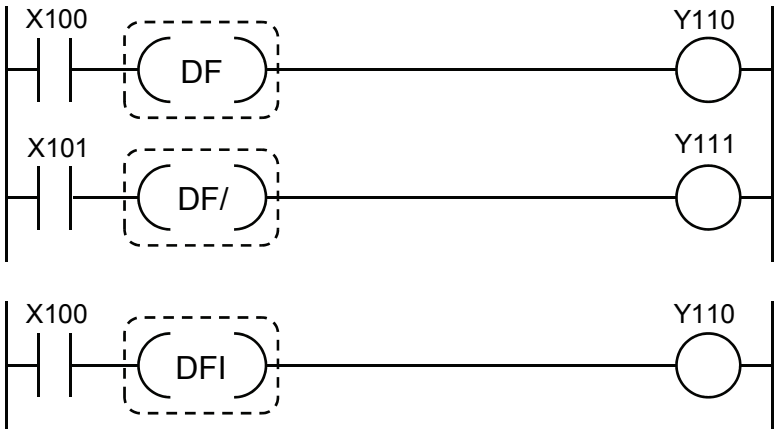
When X100 is on, Y110 is on and Y111 is off.
When X100 is on, Y110 is on and Y111 is off.



DF, DF/ (Leading Edge Differential, Trailing Edge Differential)

DFI (Leading Edge Differential (Initial Execution Type))

■ Ladder diagram



■ Description

Type of instruction	Operation
DF	The DF instruction only generates output (differential output) for a single scan where the execution condition changes from the off state to the on state (i.e., rises).
DF/	The DF/ instruction only generates output (differential output) for a single scan where the execution condition changes from the on state to the of state (i.e., falls).
DFI	The DFI instruction only generates output (differential output) for a single scan where the execution condition changes from the off state to the on state (i.e., rises).

■ Precautions during programming

- There are no restrictions on the number of times the differential instruction (DF, DF/, or DFI) can be used.
- The differential instruction only detects changes in the on/off states of a contact.
- The DF and DF/ instructions do not generate output if the execution condition is already satisfied (i.e., on state) when the operation mode is switched to RUN or the power is turned on in the RUN mode.
- The DFI instruction generates output (differential output) for the first single scan even if the execution condition is already satisfied when the RUN mode is initiated.
- If the execution condition can be already satisfied (i.e., on state) when the operation mode is switched to RUN or the power is turned on in the RUN mode, output will not be obtained for the first single scan by the DF instruction. Use the DFI instruction instead in this case.
- Be careful when using a differential instruction with an instruction (1 to 6 below) which changes the order of instruction execution such as MC, MCE, JP or LBL.

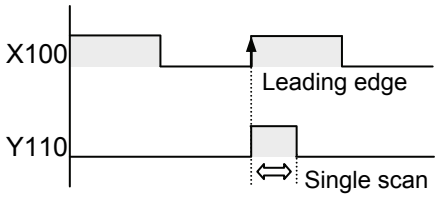
1) MC and MCE instructions
2) JP and LBL instructions
3) LOOP and LBL instructions
4) CNDE instructions
5) Step ladder instructions
6) Subroutine instructions

- When using a differential instruction with an AND stack (ANS) instruction or pop stack (POPS) instruction, be sure to write the code correctly.

■ Example of Operation

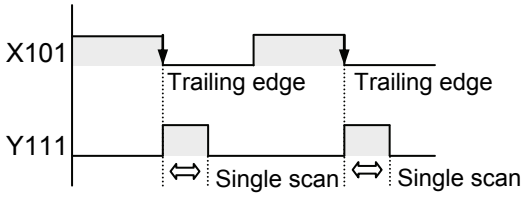
(1) Program operation for "DF" in ladder diagram

Outputs to Y110 only for a single scan where X100 changes from the off state to the on state (i.e., rises).



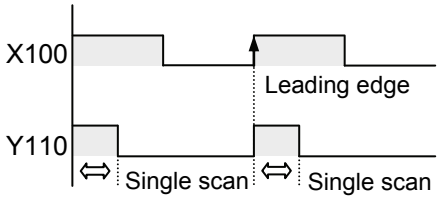
(2) Program operation for "DF/" in ladder diagram

Outputs to Y111 only for a single scan where X101 changes from the on state to the off state (i.e., falls).



(3) Program operation for "DFI" in ladder diagram

Outputs to Y110 only for a single scan where X100 changes from the off state to the on state (i.e., rises).

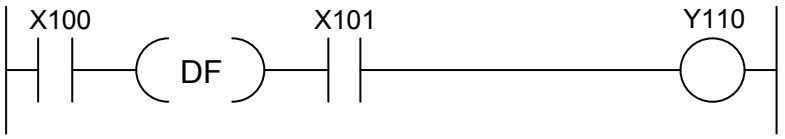


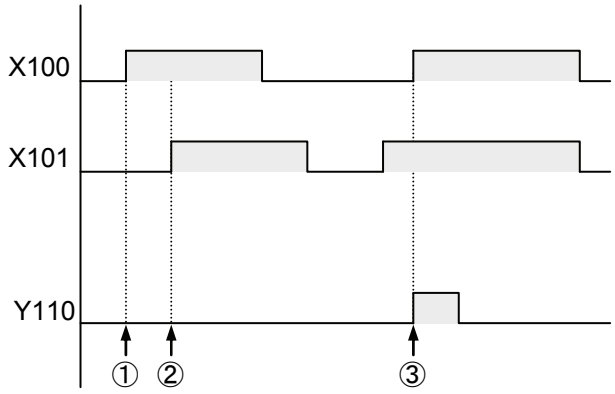
■ Sample program

- A circuit shown below operates as described below.

(1) Example 1 using DF instruction

→ When the leading edge differential is set between input information (X100, X101)





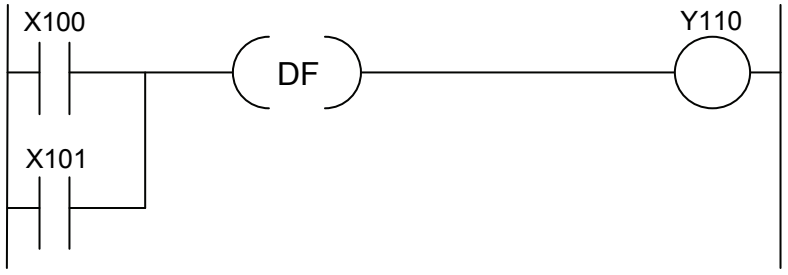
1) While X101 is off, Y110 remains off even when X100 rises.

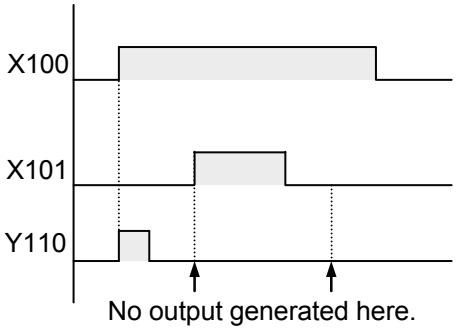
2) While X100 is on, Y110 remains off even when X101 rises.

3) While X101 is on, Y110 becomes on for a single scan when X100 rises.

(2) Example 2 using DF instruction

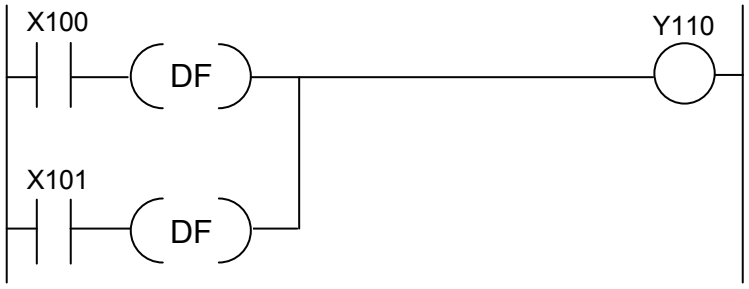
→ When the leading edge differential is set after parallel connection of input information (X100, X101)

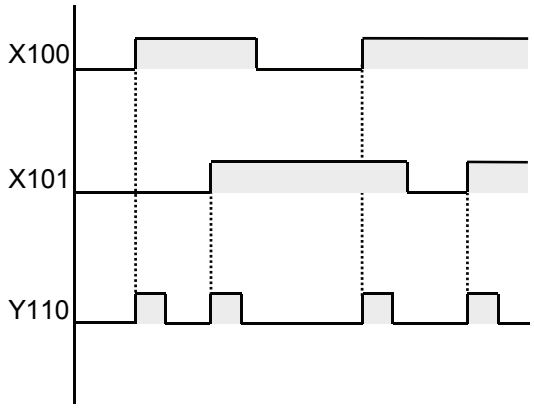




(3) Example 3 using DF instruction

→ When the leading edge differential is set for each input information (X100, X101)



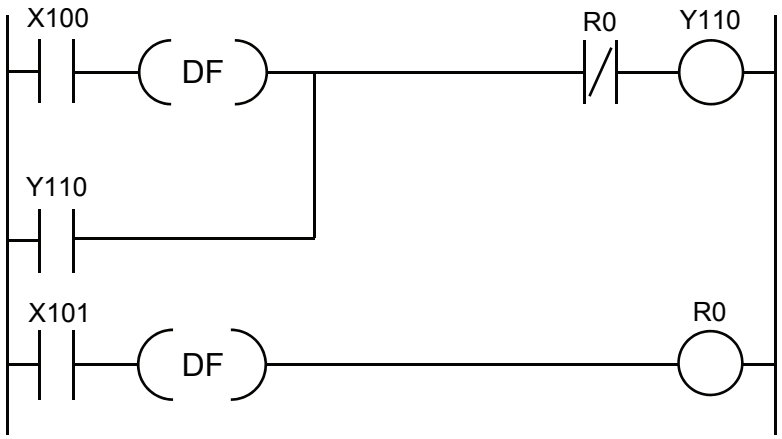


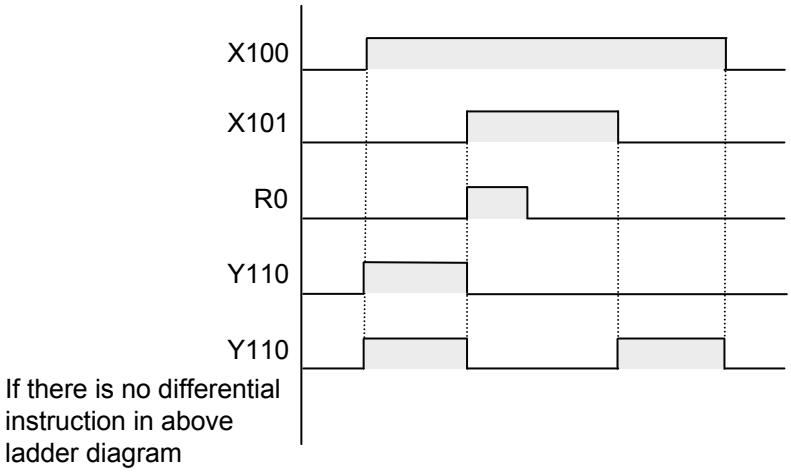
■ Application example of differential instructions

- Using differential instructions makes it easy to create and adjust programs.

<Application example for self-holding circuit>

- Differential instructions are convenient when input signals are long.



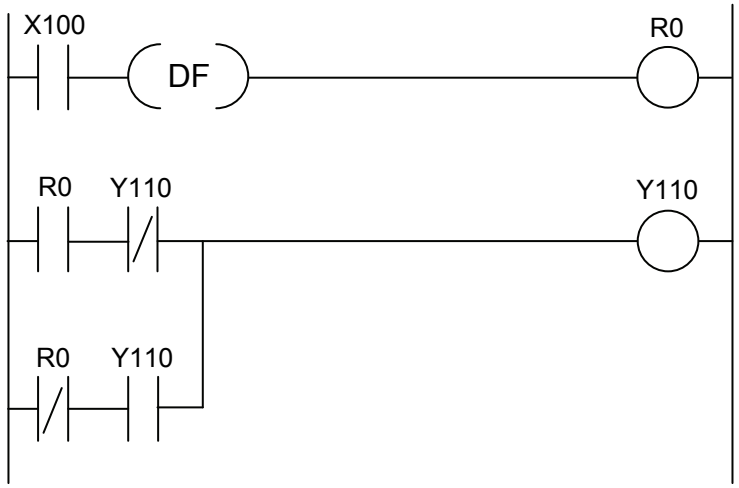


If there is no differential instruction in above ladder diagram

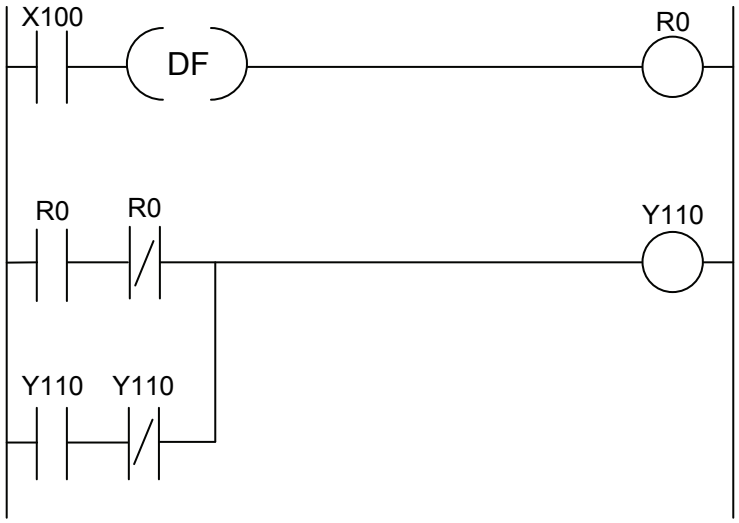
<Application example for alternating circuit>

- Differential instructions can also be applied to an alternating circuit which uses a single signal to hold and release the circuit.

<Example 1>

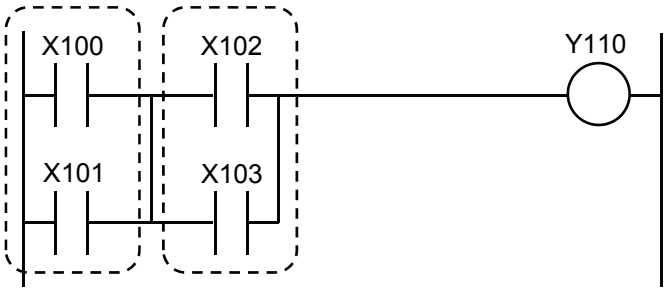


<Example 2>



ANS (AND Stack)

■ Ladder diagram



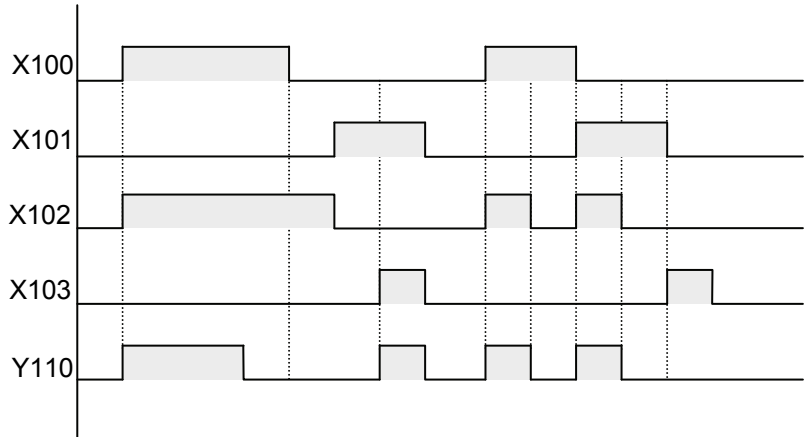
■ Description

- Connects blocks, connected in parallel, in serial.
- Each block should start with the ST instruction.

■ Example of Operation

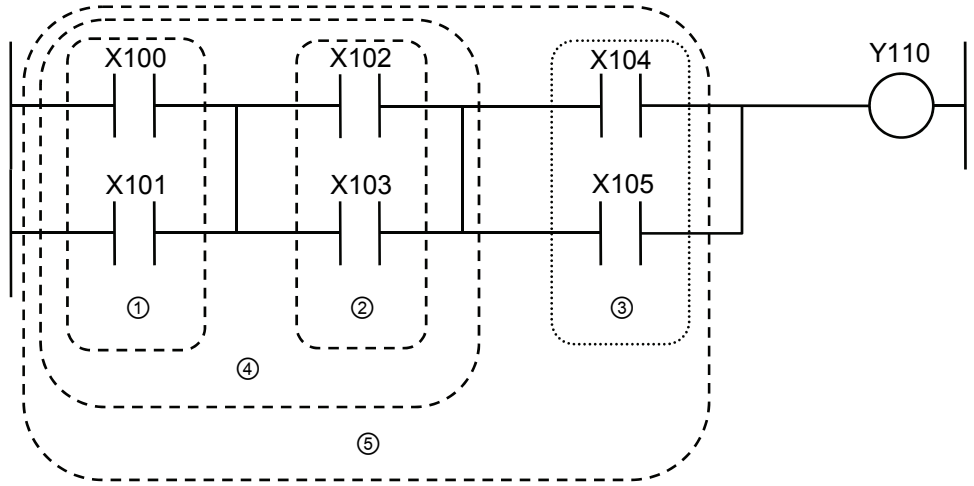
X100 or X101 is on and X102 or X103 is on, ouputs to Y110.

(X100 AND X101) OR (X102 AND X103) → Y110



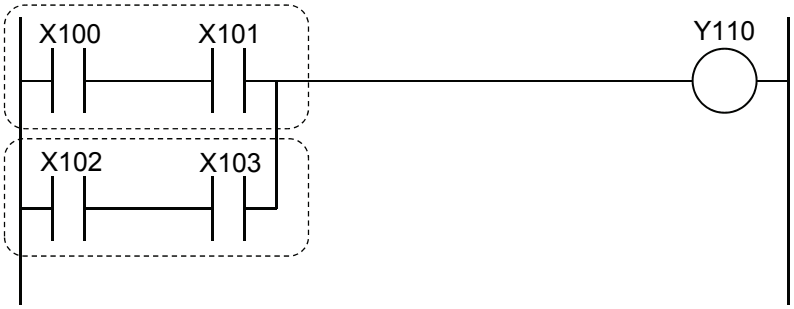
■ When blocks are sequential

- When blocks are sequential, divide them as follows.



ORS (OR Stack)

■ Ladder diagram



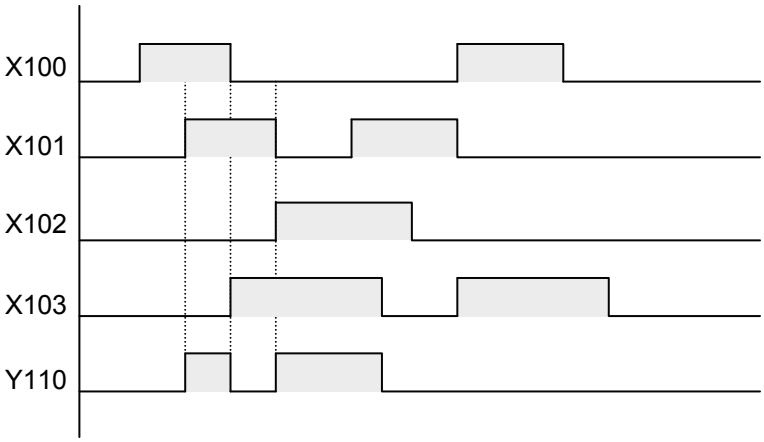
■ Description

- Connects blocks, connected in serial, in parallel.
- Each block should start with the ST instruction.

■ Example of Operation

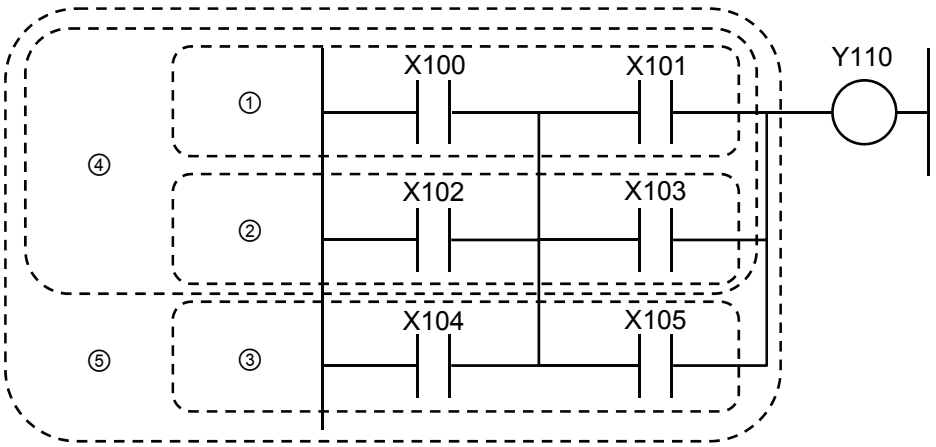
X100 and X101 are on or X102 and X103 are on, outputs to Y110.

(X100 OR X101) AND (X102 OR X103) → Y110



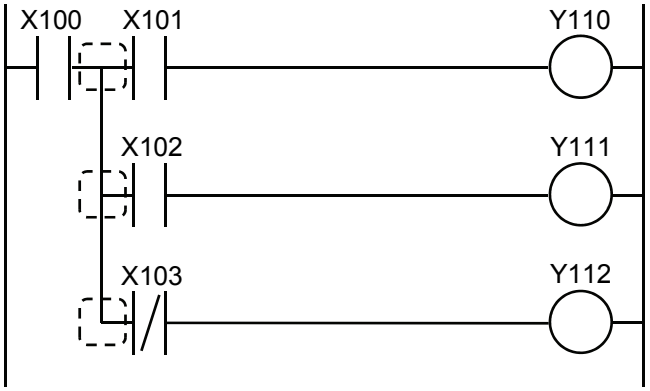
■ When blocks are sequential

- When blocks are sequential, divide them as follows.



PUSHS (Push stack)
RDS (Read stack)
POPS (Pop stack)

■ Ladder diagram



■ Description

Type of instruction	Operation
PUSHS	Saves the preceding operation result in this instruction, and continues with the operation from the next step.
RDS	Reads the operation result saved by the PUSHS instruction, and continues with the operation using it from the next step.
POPS	Reads the operation result saved by the PUSHS instruction, continues with the operation using it from the next step, and resets the operation result saved by the PUSHS instruction.

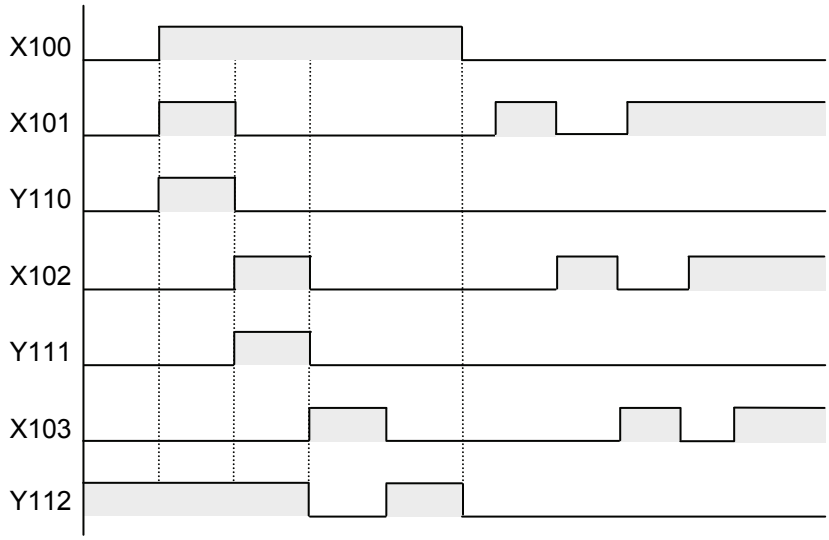
- Saves a single operation result, and read it and performs multiple operations.
- This instruction is used to branch from a single contact and connect to further contacts.

■ Example of Operation

1) When X100 is on, the PUSH instruction saves the operation result and outputs it to Y110 if X101 is on.

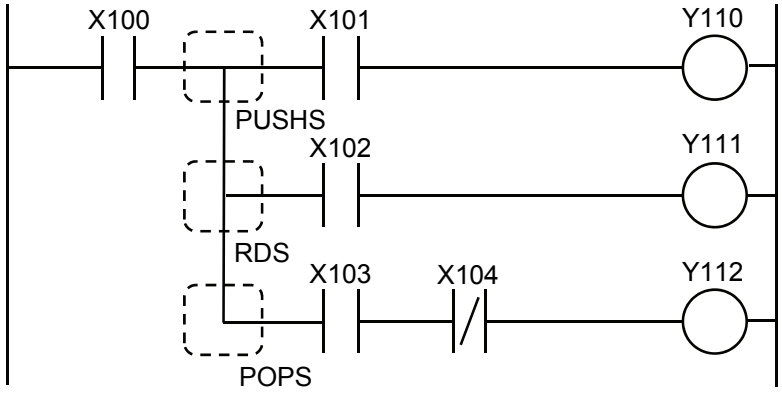
2) The RDS instruction reads the operation result and outputs it to Y111 if X102 is on.

3) The POP instruction reads the operation result and outputs it to Y112 if X103 is off, and rests the operation result saved by the PUSH instruction.

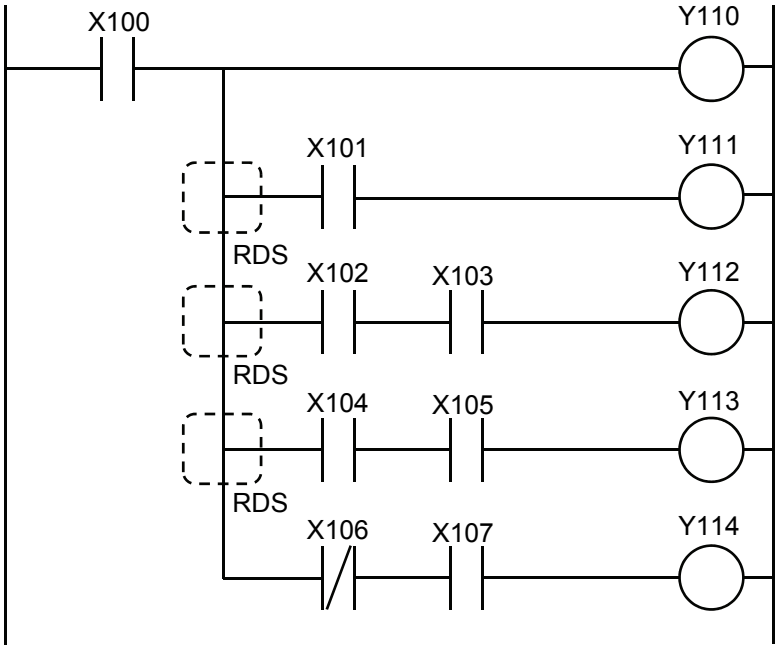


■ Precautions during programming

- Use the RDS instruction when the operation result will be further used and the POPS instruction if it will not be used any more. (Be sure to use the POPS instruction in both cases.)

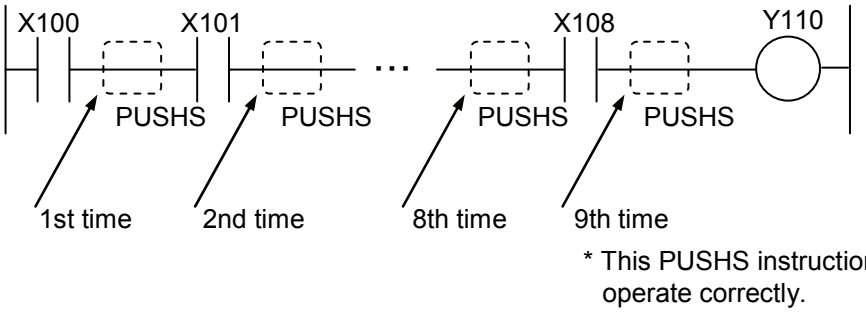


- The RDS instruction can be used repeatedly for an unlimited number of times.



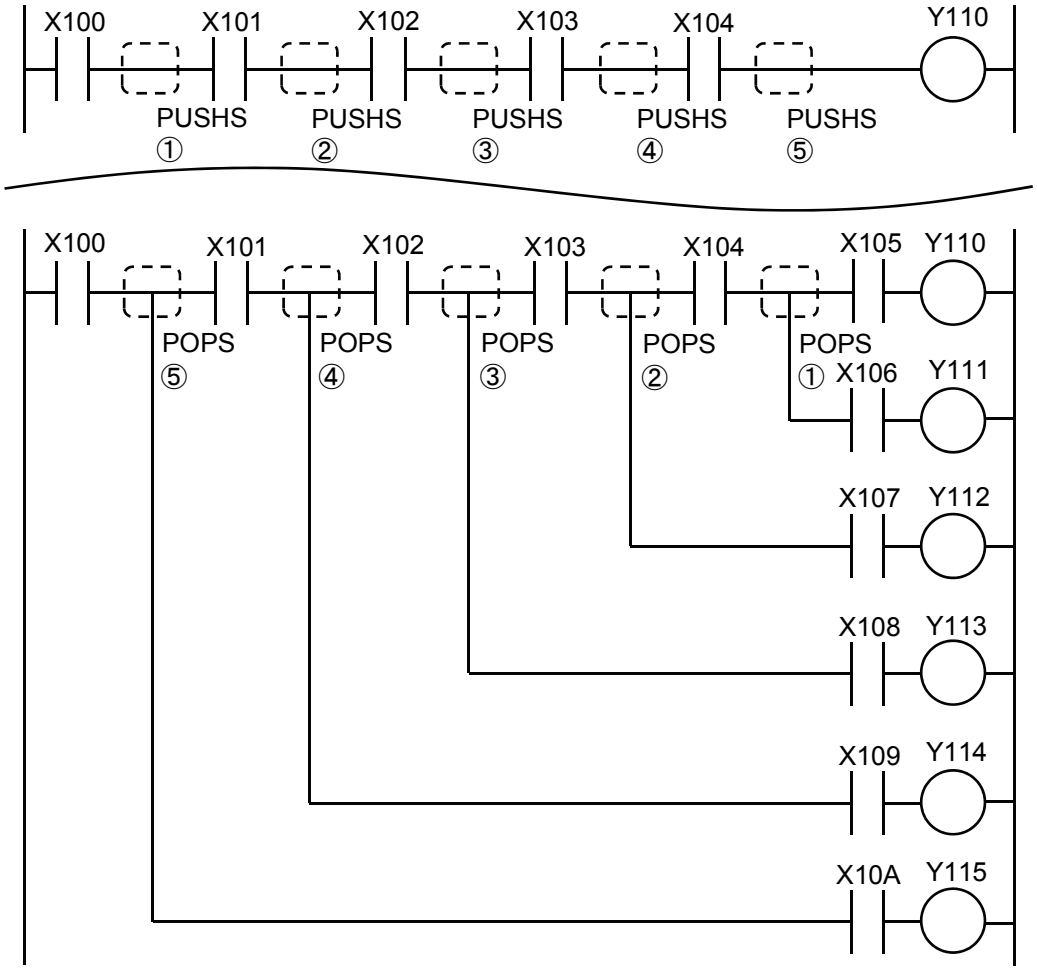
■ Precaution when using PUSHS instruction repeatedly

- There is a limit on the number of times the PUSHS instruction can be used repeatedly. The maximum number of times it can be used repeatedly before using the next POP instruction is eight.
- If the number of times for repeated use exceeds the limit, the program will not operate correctly.



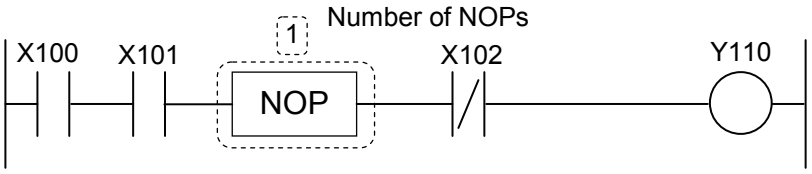
* This PUSHS instruction does not operate correctly.

- If the POPS instruction is used while the PUSHS instruction is used repeatedly, the operation results will be read with the result saved by the last PUSHS instruction first. The numbers in the figure correspond to the instruction results.



NOP (Nop)

■ Ladder diagram



■ Description

- Does not affect the preceding operation results. The program will operate in the same manner whether or not the NOP instruction is used.
- The NOP instruction may be used to make it easy to view the program code when reviewing and/or modifying it.
- To erase an instruction without changing the program address, write the NOP instruction over it.
- To send the address of a program portion without modifying the program, insert the NOP instruction.
- The NOP instruction is useful, for example, to divide a long program into several blocks.

■ Sample program

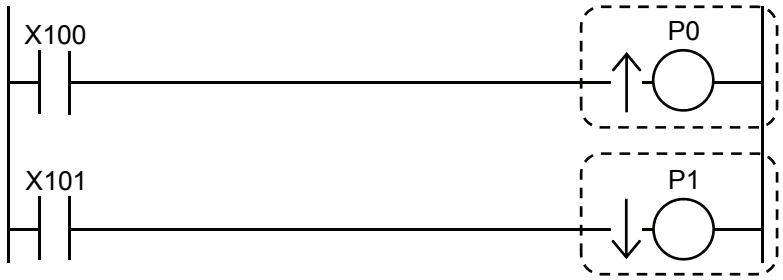
addresses		addresses		
0	ST X100	0	ST X100	
1	AN X101	1	AN X101	
2	AN/ X102	2	NOP	← NOP instruction is inserted here and the addresses change.
3	OT Y100	3	AN/ X102	
		4	OT Y100	

■ Deleting NOP instruction

- After creating a program, all NOP instructions contained in it can be deleted using a programming tool.

↑OT, ↓OT (Leading, Trailing Edge Out)

■ Ladder diagram



■ Available Devices (●: Available)

Operands	Bit device											Bit specification of word device		Index modifier
	X	Y	R	L	T	C	P	E	S	IN	OT	DT.b	LD.b	
bit							●							●

■ Description

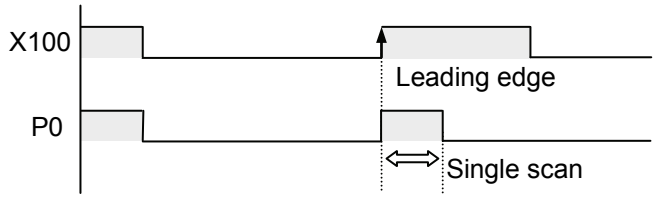
Type of instruction	Operation
↑OT	Generates output only for a single scan where the preceding operation result changes from the off state to the on state (i.e., rises).
↓OT	Generates output only for a single scan where the preceding operation result changes from the on state to the off state (i.e., falls).

■ Example of Operation

(1) Program operation for "↑OT" in ladder diagram

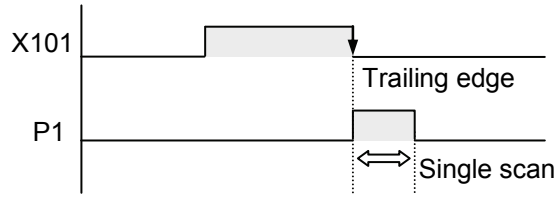
Outputs to the pulse relay P0 only for a single scan where X100 changes from the off state to the on state (i.e., rises).

Also outputs to P0 even if X100 is on for the first scan.



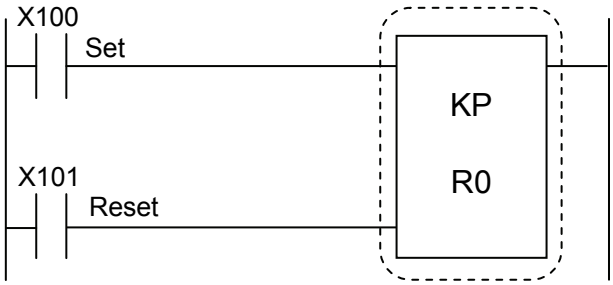
(2) Program operation for "↓OT" in ladder diagram

Outputs to the pulse relay P1 only for a single scan where X101 changes from the on state to the of state (i.e., falls).



KP (Keep)

■ Ladder diagram



■ Available Devices (●: Available)

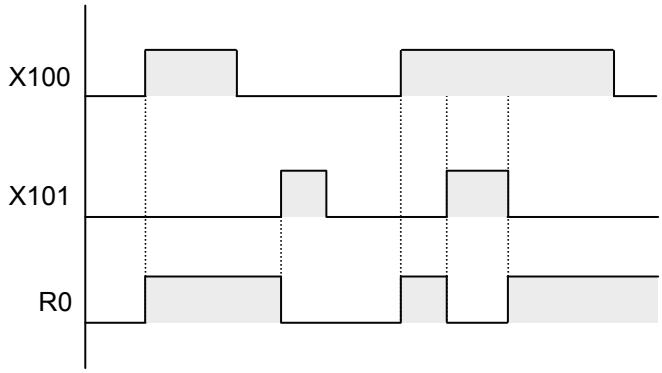
Operands	Bit device											Bit specification of word device		Index modifier
	X	Y	R	L	T	C	P	E	S	IN	OT	DT.b	LD.b	
bit	●	●	●	●				●			●	●	●	●

■ Description

- Turns on the specified coil output when the set input (X100) turns on and holds the on state. Release the output state when the reset input (X101) turns on.
- While holding the state, holds the output state regardless the on and off states of the set input (X100) until the the reset input (X101) turns on.
- If the set input (X100) and the reset input (X101) turn on simultaneously, the reset input (X101) will take precedence.

■ Example of Operation

1) Turns on the specified coil (R0) output when X100 turns on and holds the on state.
2) Release the output state when X101 turns on.

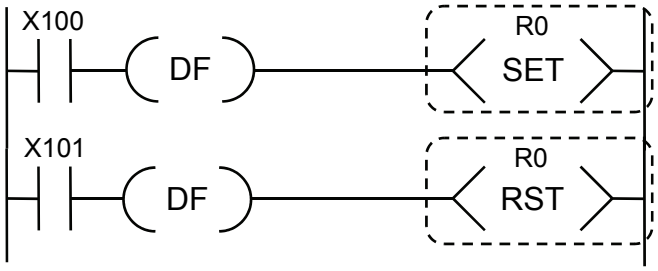


■ Precautions during programming

- The output destination holds the state even while the MC instruction is in operation.
- The state will be reset when the operation mode is switched from RUN to PROG. or the power is turned off. However, this is not the case when an internal relay set to the hold type is specified as the output destination.

SET, RST (Set, Reset)

■ Ladder diagram



■ Available Devices (●: Available)

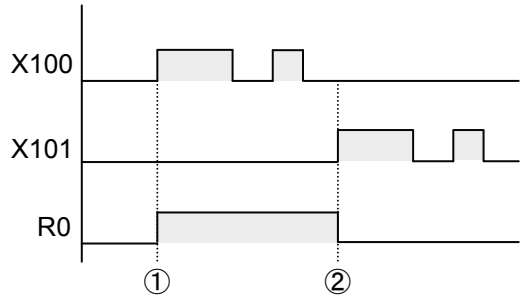
Operands	Bit device											Bit specification of word device		Index modifier
	X	Y	R	L	T	C	P	E	S	IN	OT	DT.b	LD.b	
bit	●	●	●	●				●			●	●	●	●

■ Description

Type of instruction	Operation
SET	Turns on the output when the execution condition is on and holds the state regardless of changes of the execution condition.
RST	Turns off the output coil when the execution condition is on and holds the off state regardless of changes of the execution condition.

■ Example of Operation

1) When X100 turns on, R0 turns on and its state is held.
2) When X101 turns on, R1 turns off and its state is held.

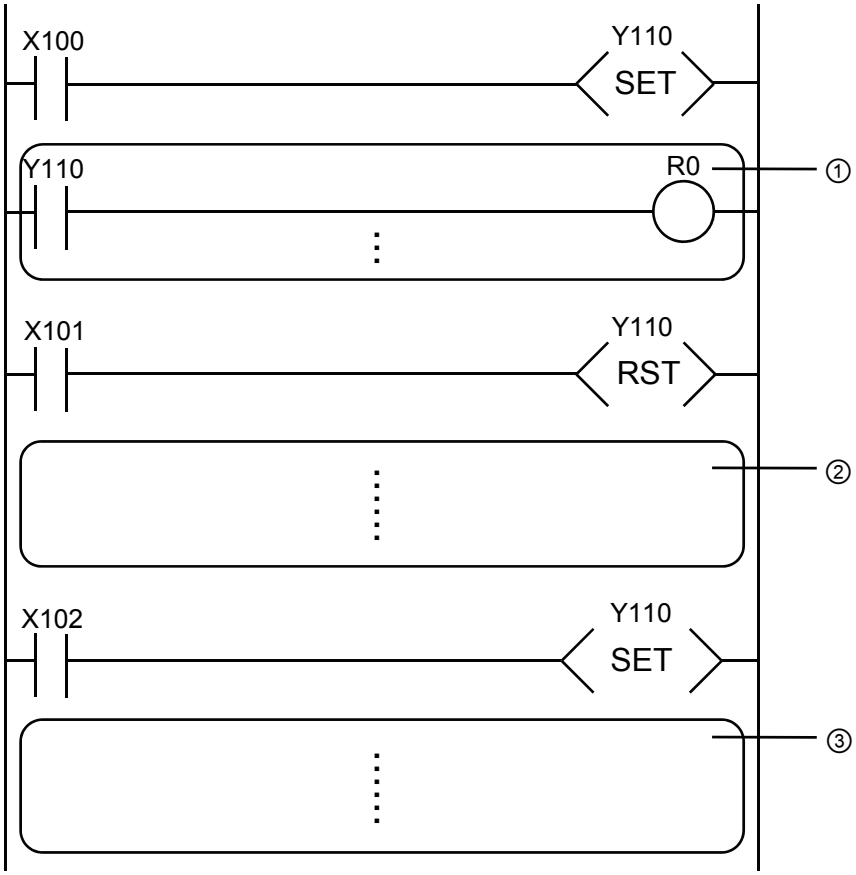


■ Precautions during programming

- The same output coil can be specified as the output destination of the SET and RST instructions for any number of times.
- Using relays with the SET and RST instructions does not result in duplicated output. It will also not be handled as a syntax error by the total check function.
- The RST instruction can be used to turn off the relay.
- The output destination of the SET instruction holds the state even while the MC instruction is in operation.
- The output destination of the SET instruction will be reset when the operation mode is switched from RUN to PROG. or the power is turned off. However, this is not the case when an internal relay set to the hold type is specified as the output destination.
- A pulse relay (P) cannot be specified as an output destination of the SET and RST instructions.
- The error alarm buffer can be all cleared by RST SD60.
- The first entry of the error alarm buffer can be cleared by RST SD61.

■ Processing mechanism of SET and RST instructions

- While operations are processed, the output contents is rewritten for each step.
- Since I/O refresh is done when the ED instruction is executed, data actually output depends on the final operation result.
- In order to output an operation result while processing is in progress, use the direct output (OT) instruction.



If X100 to X102 are all on in the above program,

1) Processing is done with Y110 to be on.

2) Processing is done with Y110 to be off.

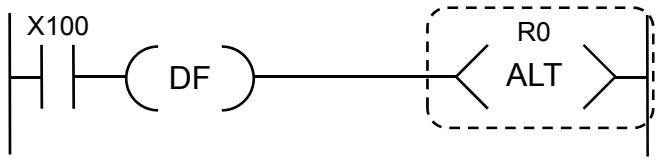
3) Processing is done with Y110 to be on.

■ Use SET and RST instructions with differential instructions

- Putting the differential DF instructions before the SET and RST instructions makes it easy to create and adjust the program.
- This is especially effective if the same output destination is used in many places within the program.

ALT (Alternative out)

■ Ladder diagram



■ Available Devices (●: Available)

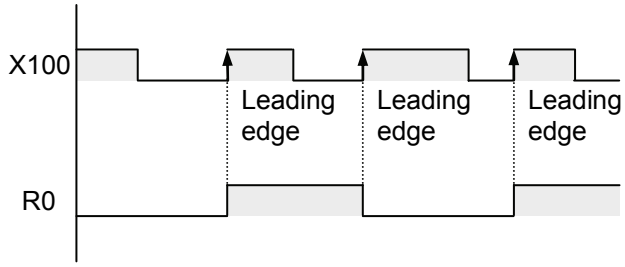
Operands	Bit device											Bit specification of word device		Index modifier
	X	Y	R	L	T	C	P	E	S	IN	OT	DT.b	LD.b	
bit	●	●	●	●				●			●	●	●	

■ Description

- Inverts the on/off state of the specified coil when the preceding operation result changes from the off state to the on state (i.e., rises).
- The on/off state of the specified coil is held until the ALT instruction specifying it rises next time. (Flip-flop control)

■ Example of Operation

On/off states of output R0 is inverted whenever X100 changes from the off to on states (rises).



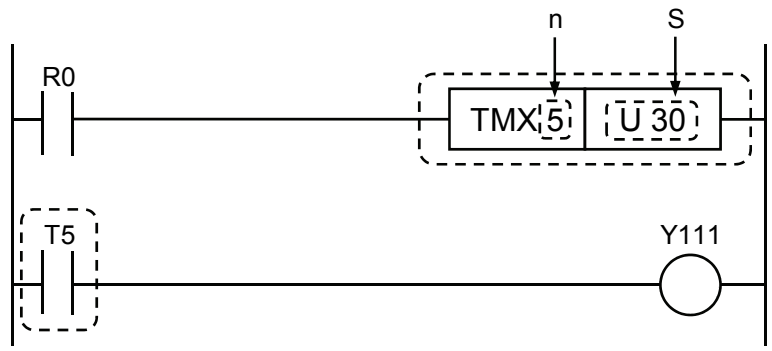
■ Precautions during programming

- The ALT instruction detects rising from off to on of the input and inverts the output.
- While the input remains on, the output is inverted at the rising edge only and not inverted later.
- The output will not be inverted for the first scan if the input is already on when the operation mode is switched to RUN or the power is turned on in the RUN mode.
- Be careful when using this instruction with an instruction (1 to 6 below) which changes the order of instruction execution such as MC, MCE, JP or LBL, because the operation of the instruction may change depending on the instruction execution and input timing.

1) MC and MCE instructions
2) JP and LBL instructions
3) LOOP and LBL instructions
4) CNDE instructions
5) Step ladder instructions
6) Subroutine instructions

TM (Timer)

■ Ladder diagram



■ Operand list

Operand	Explanation
n	Timer No.
S	Timer set value

■ Available Devices (●: Available)

Operands	16-bit device											32-bit device			Integer			Real numbers		String	Index modifier *2
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS *1	TE CE	IX	K	U	H	SF	DF	" "	
n						●										●					●
S	●	●	●			●	●	●				●				●					●

*1: Only TS can be specified for TM instruction.
*2: Only 16-bit device, 32-bit device, or integer constants can be modified (but not the real number or string constants).

■ Description

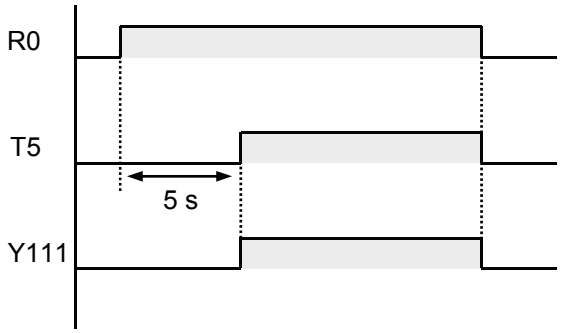
- The timer is the non-hold type which is reset when the power is turned off or the operation mode is switched from RUN to PROG.
- When the execution condition is on, the timer starts decrementing from the set time [S]. When the elapsed value reaches 0, the timer contact [Tn] (n is the timer contact number) turns on.
- When the execution condition turns off while the timer is decrementing, the timer stops and resets the elapsed value (clears it to zero).
- The OT instruction can be written immediately after the timer coil.

■ Regarding the specification of timer time

- The timer time is calculated as "timer unit" x "timer set value".
- The timer set value [S] is specified within the range between U1 to U4294967295, using a decimal constant.

"TMS" is specified within the range from 0.00001 to 42949.67295 seconds, in 0.00001 seconds.
"TML" is specified within the range from 0.001 to 4294967.295 seconds, in 0.001 seconds.
"TMR" is specified within the range from 0.01 to 42949672.95 seconds, in 0.01 seconds.
"TMX" is specified within the range from 0.1 to 429496729.5 seconds, in 0.1 seconds.
"TMY" is specified within the range from 1 to 4294967295 seconds, in 1 seconds.

■ Example of Operation



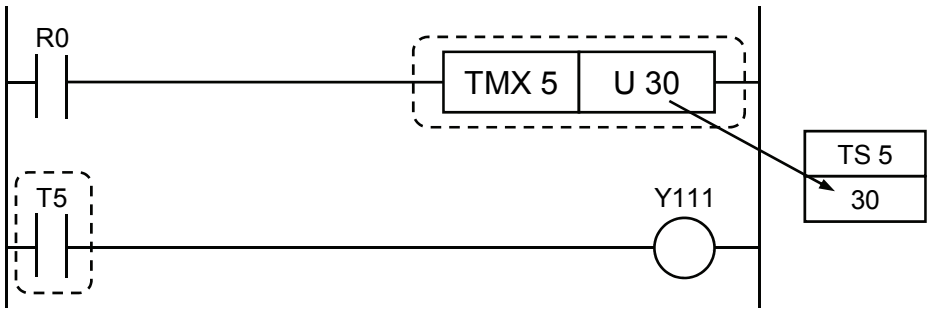
■ Precautions during programming

- The timer set value area TS and timer elapsed value area TE both occupy 32-bit areas. This is also true when a device such as DT is used in the operand [S]. Be careful not to overwrite it by another program.
- Since decrementing occurs during an operation, create the program so that decrementing occurs once during a single scan time. If multiple operations occur during a single scan due to an interrupt handler program or jump/loop instruction, or decrementing never occurs, the correct result will not be obtained.
- When using a timer instruction with an AND stack (ANS) instruction or pop stack (POPS) instruction, be sure to write the code correctly.

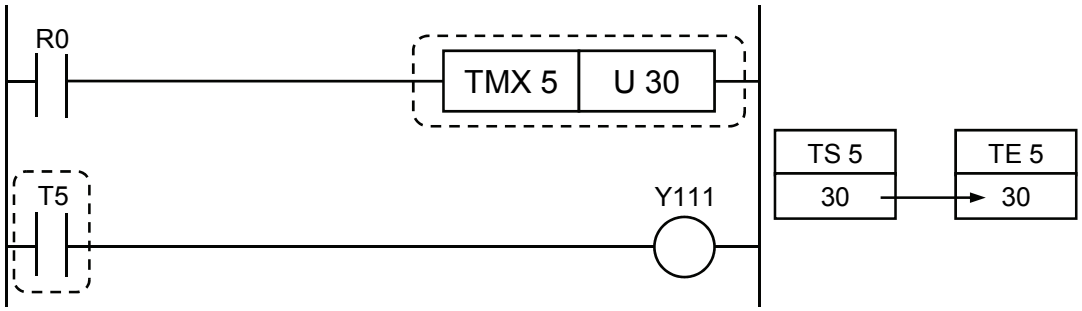
■ Mechanism of timer operation

- This is an example when the U constant is used to specify the set value. Refer to the next page for an example of operation with the set value area number specified.

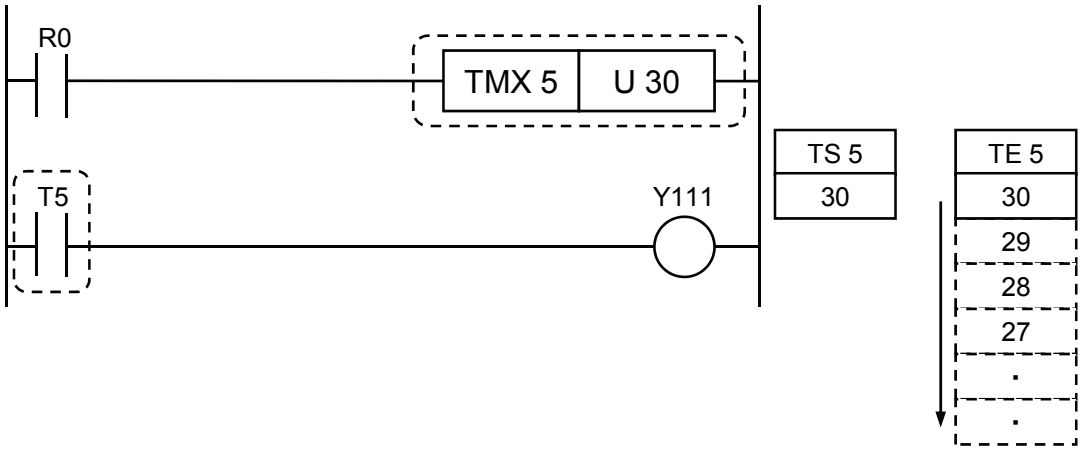
1) When the operation mode is switched to RUN or the power is turned on in the RUN mode, the timer set value is transferred to the set value area "TS" with the same number.



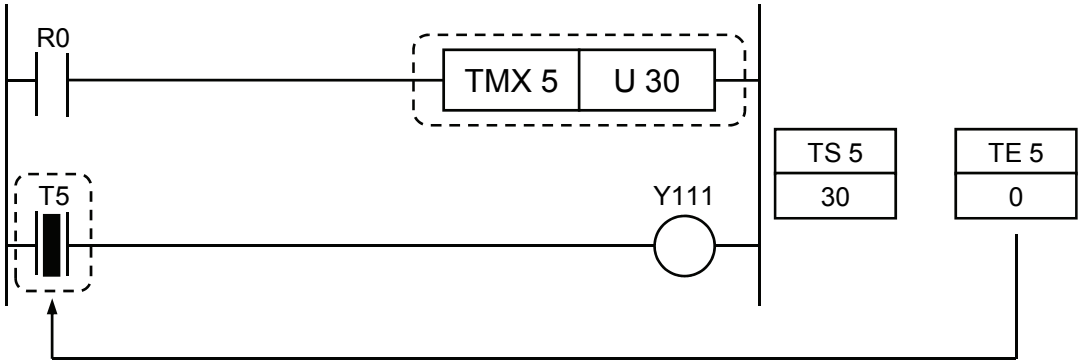
2) When the timer execution condition changes from off to on (i.e., rises), the timer set value is transferred from the set value area "TS" to the elapsed value area "TE" with the same number. (This is also true when the operation mode is switched to RUN while the execution condition is on.)



3) For each scan, the value in the elapsed time value area "TE" decrements if the execution condition is on.

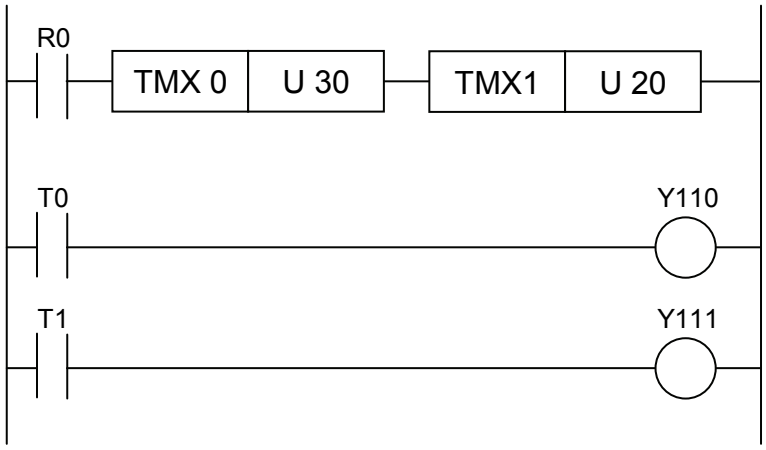


4) When the value in the elapsed value area "TE" becomes 0, the timer contact "T" with the same number turns on.

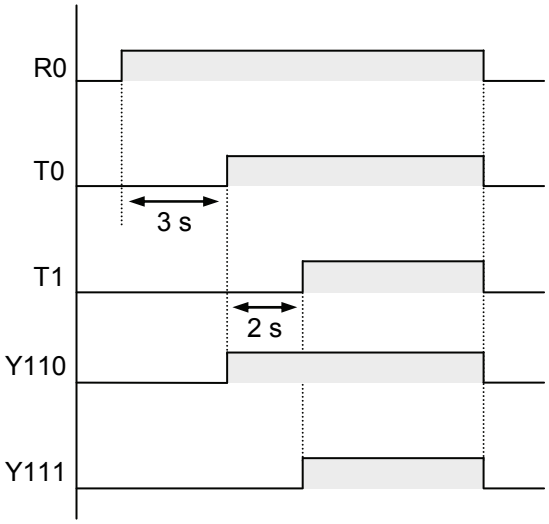


■ Application example of timer instructions <serial connection of timers>

● Ladder diagram

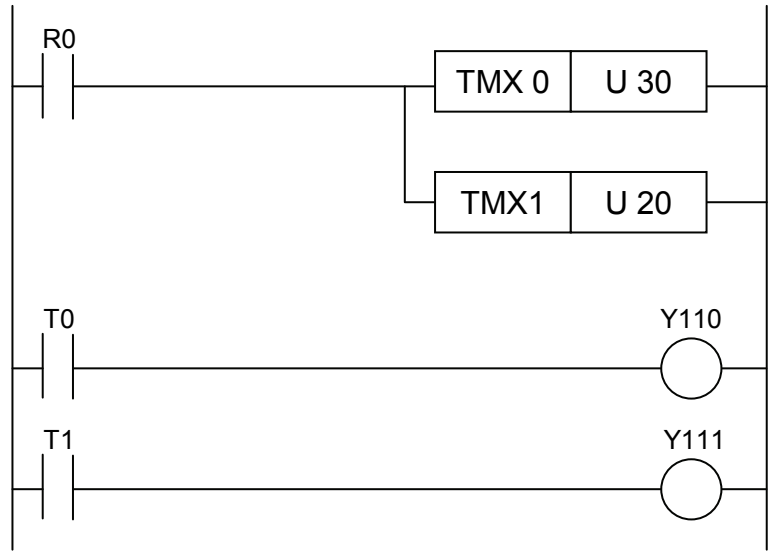


● Time chart

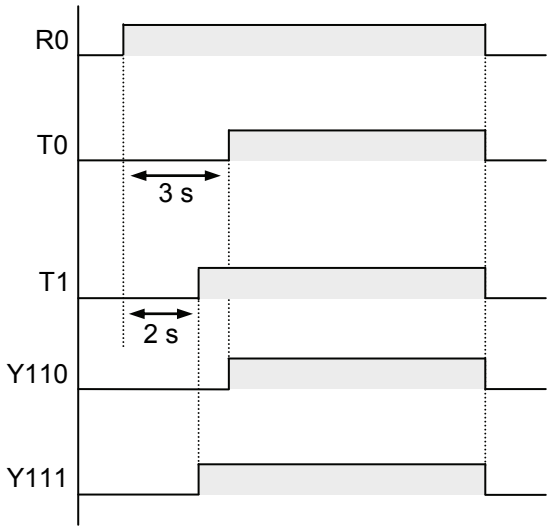


■ **Application example of timer instructions <parallel connection of timers>**

● Ladder diagram



● Time chart

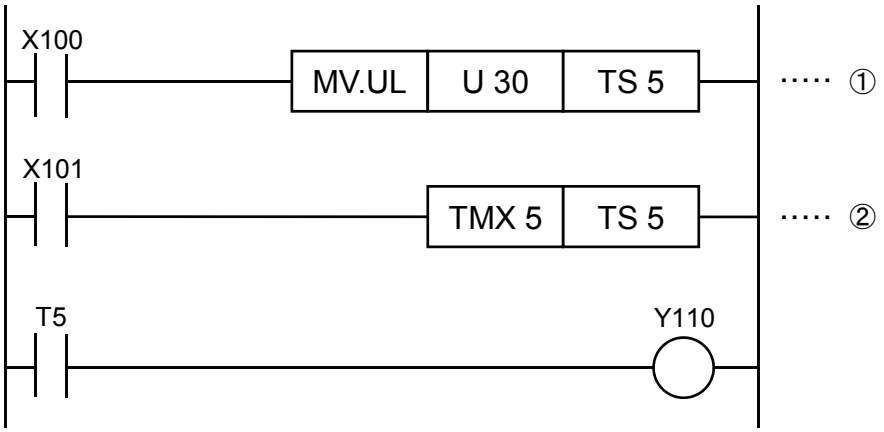


■ Regarding how to directly specify the set value area number to the timer set value

- The following program directly sets the timer set value to the timer set value area.
- Be sure to specify the same number as the timer number in [n] for the set value area "TS".
- The above program with TS5 specified to the set value operates as follows.

1) When the execution condition X100 is on, the data transfer instruction "MV" is executed to start decrementing with U30 set to TS5.

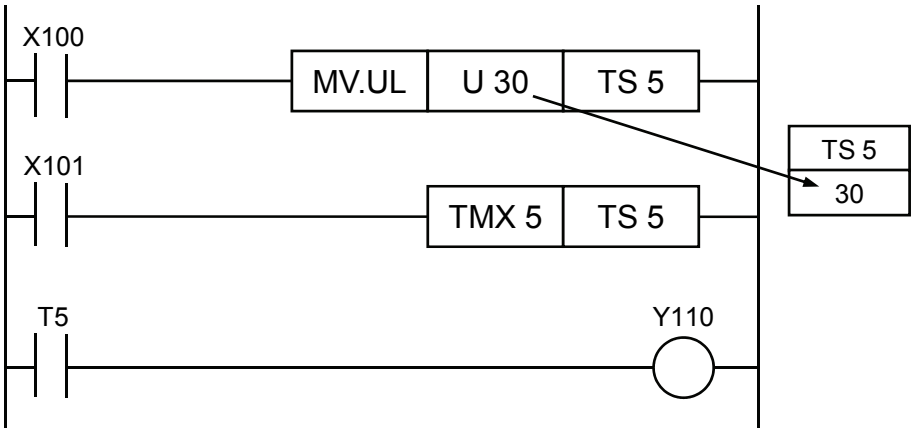
2) When the execution condition X101 turns on, decrementing starts with 30 as the set value.



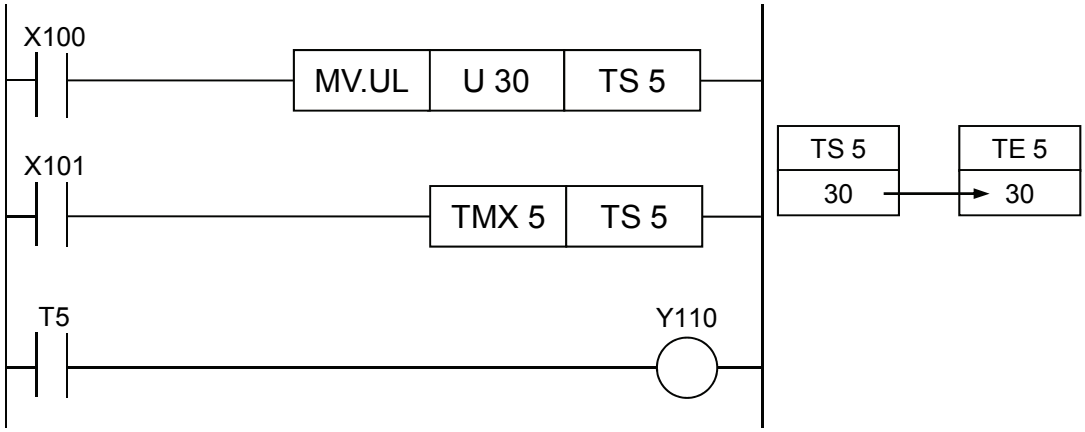
- Even when the value in the set value area "TS" is changed while decrementing, the decrement operation continues with the value before change.
- The new value will be used for the timer operation after the current decrement operation is completed or interrupted, and then the execution condition changes from off to on.
- The set value area "TS" is normally the non-hold type which is reset when the power is turned off or the operation mode is switched from RUN to PROG.

■ Mechanism of timer operation when the set value area number is specified directly

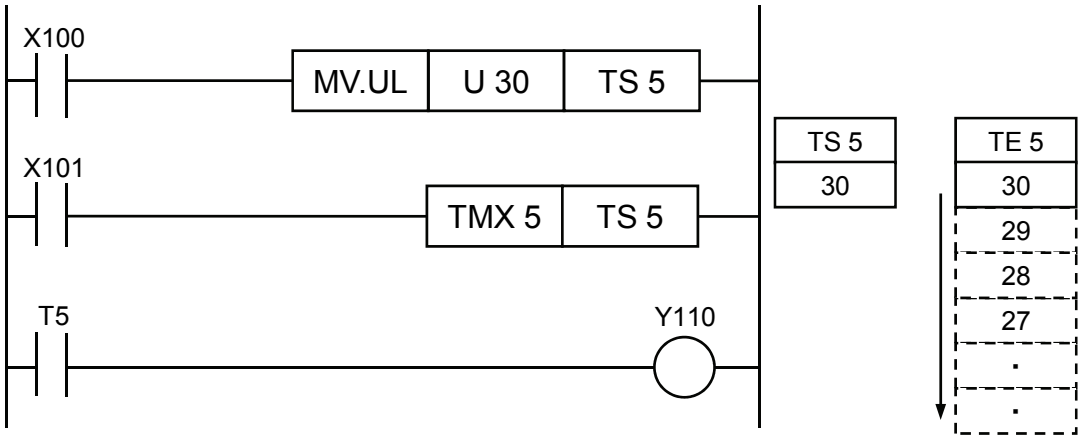
1) When the execution condition of the high-level instruction is on, the value is set in the set value area "TS". The following shows an example of using the MV instruction.



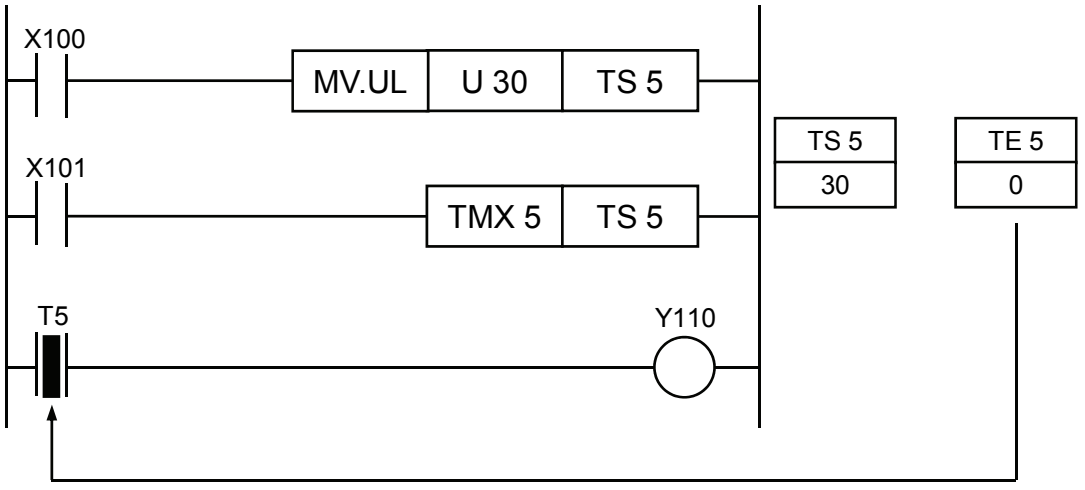
2) When the timer execution condition changes from off to on (i.e., rises), the timer set value is transferred from the set value area "TS" to the elapsed value area "TE" with the same number. (This is also true when the operation mode is switched to RUN while the execution mode is on.)



3) For each scan, the value in the elapsed time value area "TE" decrements if the execution condition is on.



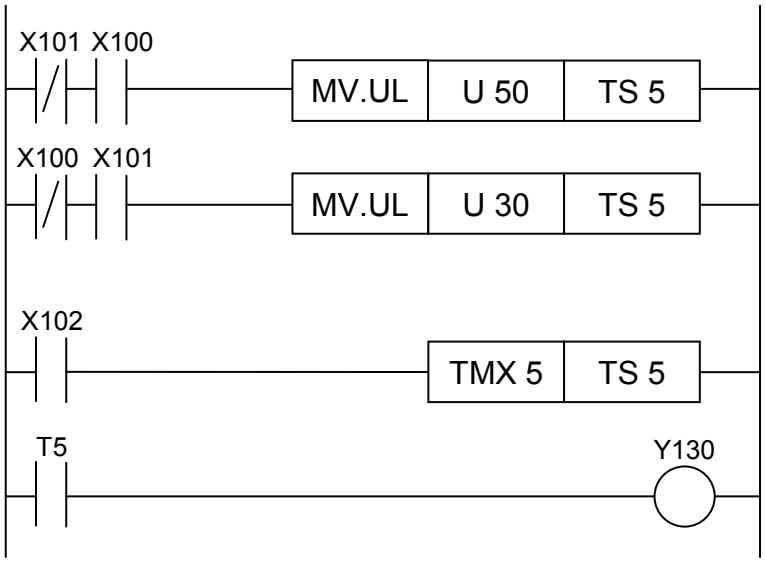
4) When the value in the elapsed value area "TE" becomes 0, the timer contact "T" with the same number turns on.



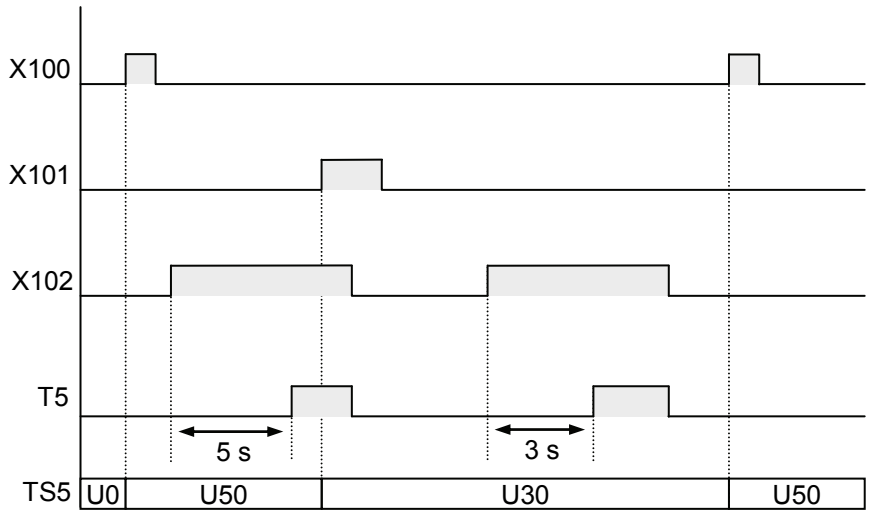
■ **Application example when the set value area number is specified directly**

<Example> Switching set values according to the condition

● Ladder diagram

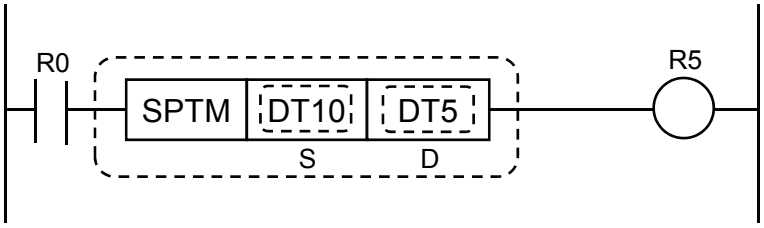


● Time chart



SPTM (unsigned 32-bit incremental auxiliary timer)

■ Ladder diagram



■ Operand list

Operand	Explanation
S	Timer set value
D	Timer elapsed value

■ Available Devices (●: Available)

Operands	16-bit device											32-bit device			Integer			Real numbers		String	Index modifier *3
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS *1	TE CE *2	IX	K	U	H	SF	DF	" "	
S	●	●	●	●			●	●	●			●				●	●				●
D	●	●	●	●			●	●	●				●								●

*1: CS cannot be specified in the first operand [S].
*2: CE cannot be specified in the second operand [D].
*3: Only 16-bit device, 32-bit device, or integer constants can be modified (but not the real number or string constants).

■ Description

- Operates as an on-delay timer with 0.01 second resolution. When the internal relay is on, the timer decrements from the set value. When the elapsed value [D] becomes 0, the system relay SRD turns on. (It is off while the internal relay is off as well as the timer is decrementing.)
- When the internal relay is turned off, the elapsed value area is cleared to zero. The relay used for the OT instruction turns off.
- The system relay SRD also turns on when the timer becomes time-up. The system relay SRD can also be used as a timer contact. (It is off while the internal relay is off as well as the timer is decrementing.)

■ Regarding the specification of timer time

- The timer time is calculated as 0.01 x "timer set value".
- The timer set value is specified within the range from U1(H1) to U4294967295(H7FFFFFFF), using a U constant.

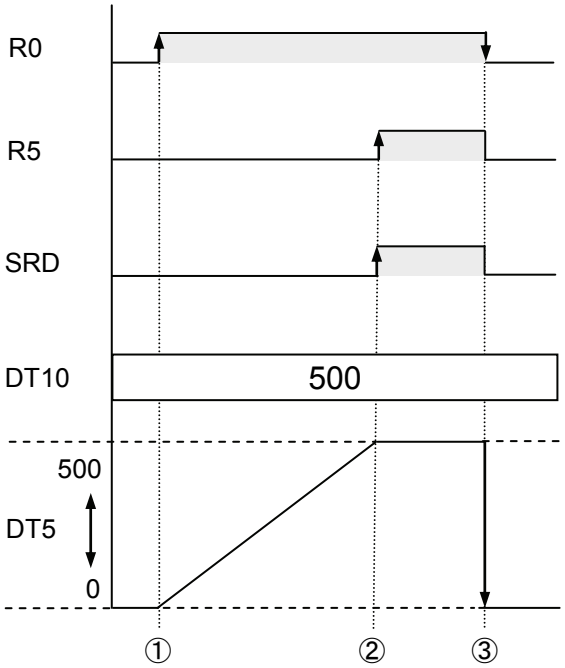
"SPTM" is specified within the range from 0.01 to 42949672.95 seconds, in 0.01 seconds.

Example) When the set value is U500, the set time is 0.01 x 500 = 5 seconds.

■ Example of Operation

When executed with U500 in the set value [S]: DT10

1) Timer operation starts when R0 turns on.
   "0" is transferred to the the elapsed value area: DT5.
2) When the value of the elapsed value area: DT5 reaches the value in the set value area: DT10 (U500), the system relay SRD and output coil R0 turn on.
3) When R0 turns off, the timer operation stops and "0" is transferred to the elapsed value area: DT5.

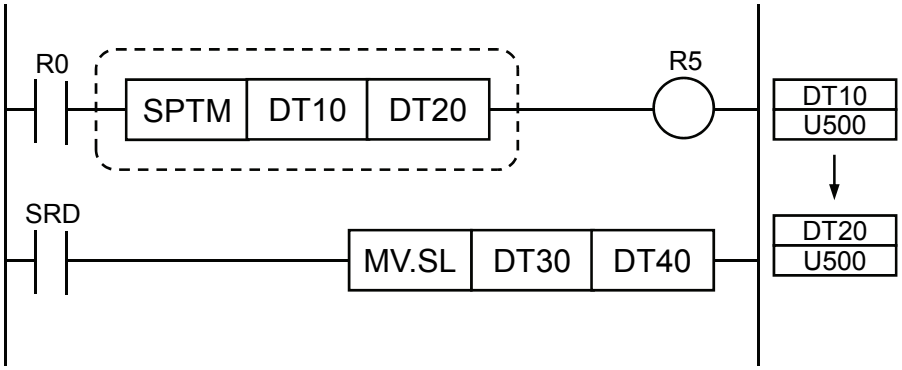


■ Precautions for programming

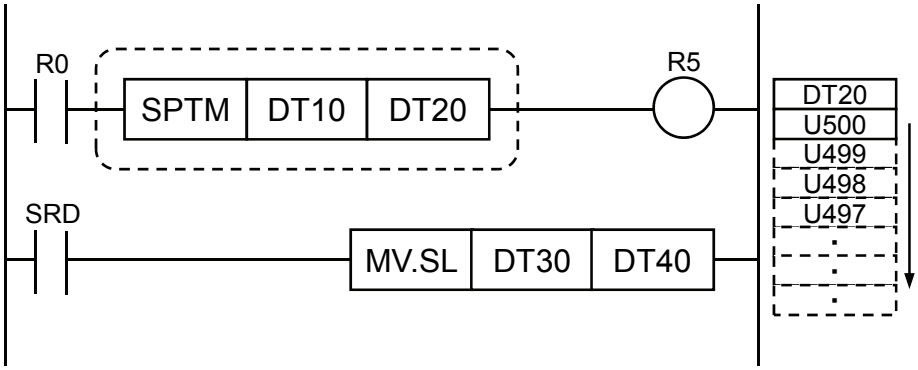
- Make sure that the operation memory areas used for other timer/counter instructions or high-level instructions are not used for the timer set and elapsed value areas.
- Since decrementing occurs during an operation, create the program so that decrementing occurs once during a single scan time. If multiple operations occur during a single scan due to an interrupt handler program or jump/loop instruction, or decrementing never occurs, the correct result will not be obtained.

■ Mechanism of auxiliary timer operation

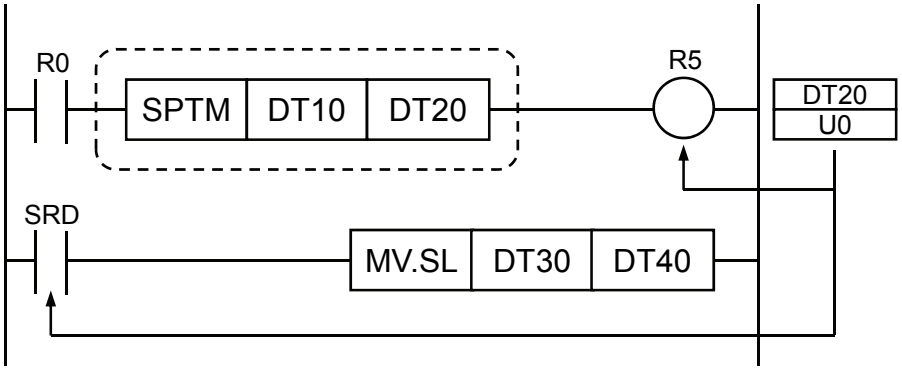
1) When the internal relay changes from off to on, the set value in [S] is transferred to the elapsed area [D].



2) For each scan, the value in the elapsed time value area [D] decrements if the internal relay is on.



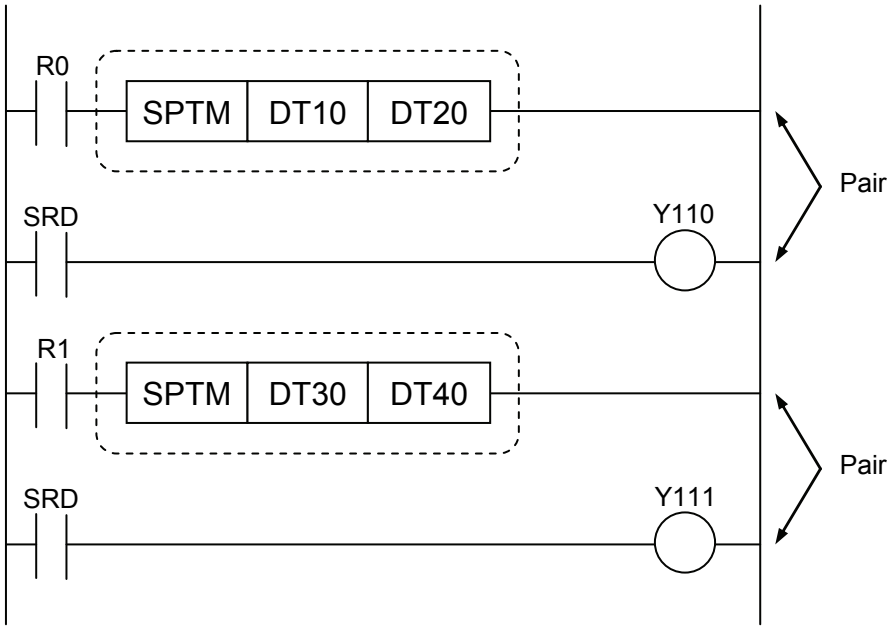
3) When the value in the elapsed value area [D] becomes 0, the relay turns on with the succeeding OT instruction. The system relay SRD also turns on.



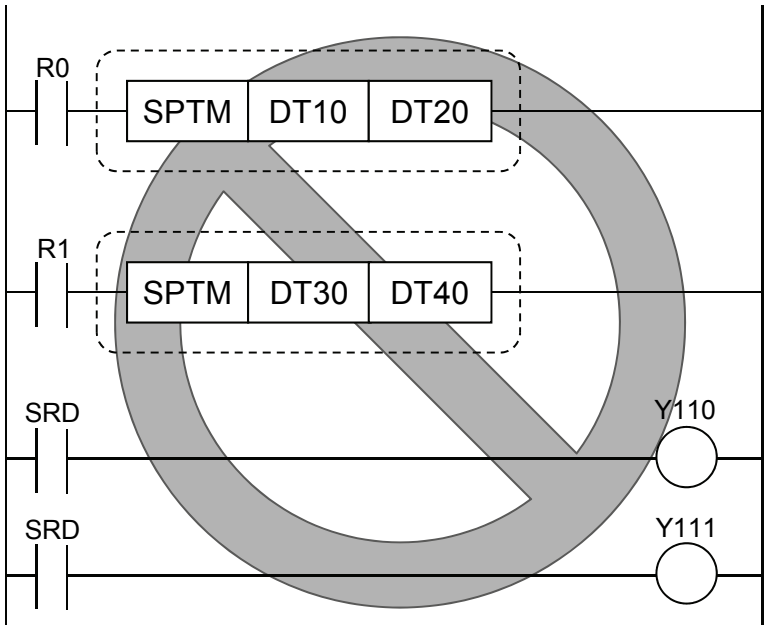
■ Precaution when using system relay SRD

- When using multiple auxiliary timers with the SRD, be sure use SRD on the line following the auxiliary timer instruction.

Example:

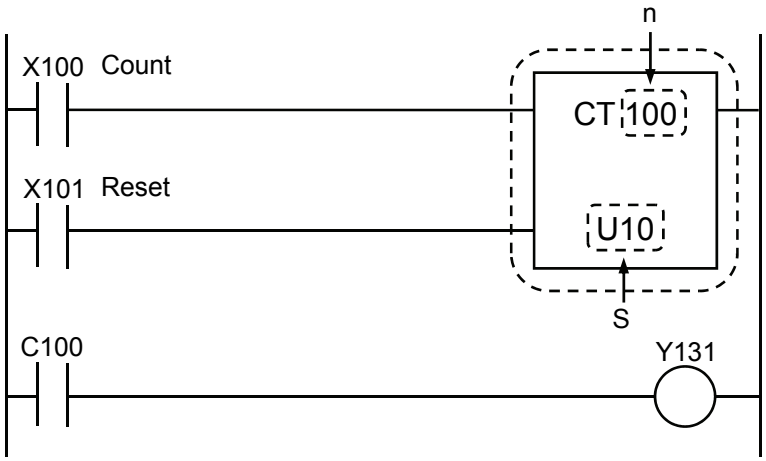


- Operation will be incorrect if code is written as follows.



CT (Down Counter)

■ Ladder diagram



■ Operand list

Operand	Explanation
n	Counter No.
S	Counter set value

■ Available Devices (●: Available)

Operands	16-bit device											32-bit device			Integer			Real numbers		String	Index modifier *2
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS *1	TE CE	IX	K	U	H	SF	DF	" "	
n																●					●
S	●	●	●	●			●	●				●				●					●

*1: Only CS can be specified for CT instruction.
*2: Only 16-bit device, 32-bit device, or integer constants can be modified (but not the real number or string constants).

■ Description

- All counters are decremental preset counters.
- When the reset input rises from on to off, the value in the set value area "CS" is preset to the elapsed value area "CE".
- When the rest input is on, the elapsed value is reset to zero.
- When the count input changes from off to on, the counter decrements from the set value. When the elapsed value becomes 0, output to the counter contact is generated.
- If the count input and the reset input turn on simultaneously, the reset input will take precedence
- If the count input rises and the reset input falls simultaneously, the count input will be ignored and only preset is performed.
- The OT instruction can be written immediately after the counter instruction.

■ Regarding count value setting

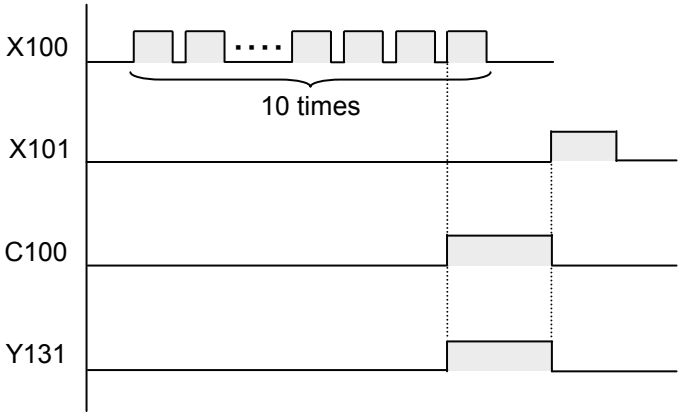
- The count value is specified within the range from U1 to 4294967295, using a decimal (U) constant.

■ Example of Operation

When counter number [n]=100, set value [S]=10

1) When X100 turns on for 10 times, C100 turns on and Y131 turns on.

2) When X101 turns on, the elapsed value is reset.

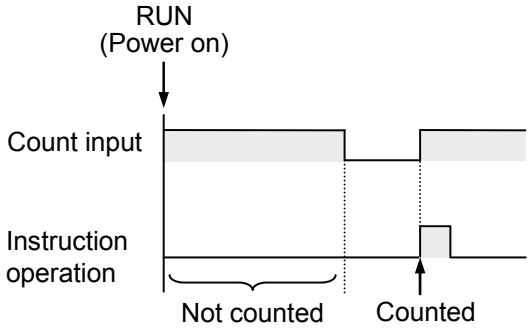


■ Precautions during programming

- The counter set value area CS and counter elapsed value area CE both occupy 32-bit areas. This is also true when a device such as DT is used in the operand [S]. Be careful not to overwrite it by another program.
- When using a counter instruction with an AND stack instruction or pop stack instruction, be sure to write the code correctly.

■ Precaution on count input detection

- The CT instruction detects rising from off to on of the count input and decrements the set value.
- While the count input remains on, counting is performed at the rising edge only and not performed later.
- The set value will not be decremented for the first scan if the count input is already on when the operation mode is switched to RUN or the power is turned on in the RUN mode.



- Be careful when using this instruction with an instruction (1 to 6 below) which changes the order of instruction execution such as MC, MCE, JP or LBL, because the operation of the instruction may change depending on the instruction execution and count input timing.

1) MC and MCE instructions

2) JP and LBL instructions

3) LOOP and LBL instructions

4) CNDE instructions

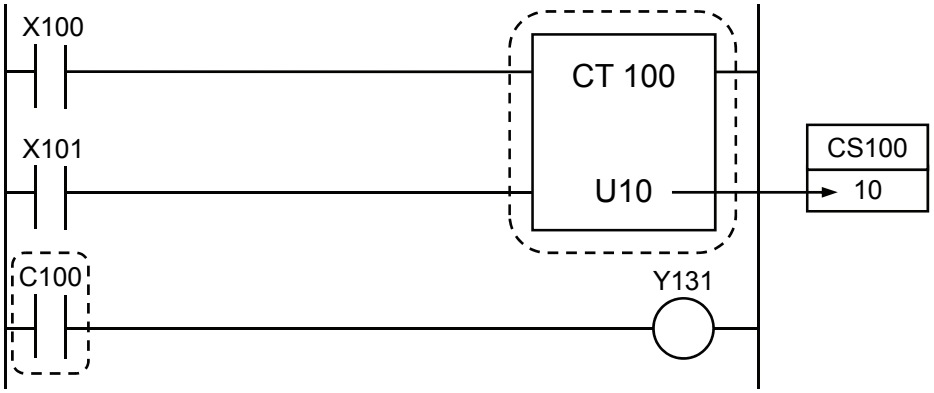
5) Step ladder instructions

6) Subroutine instructions

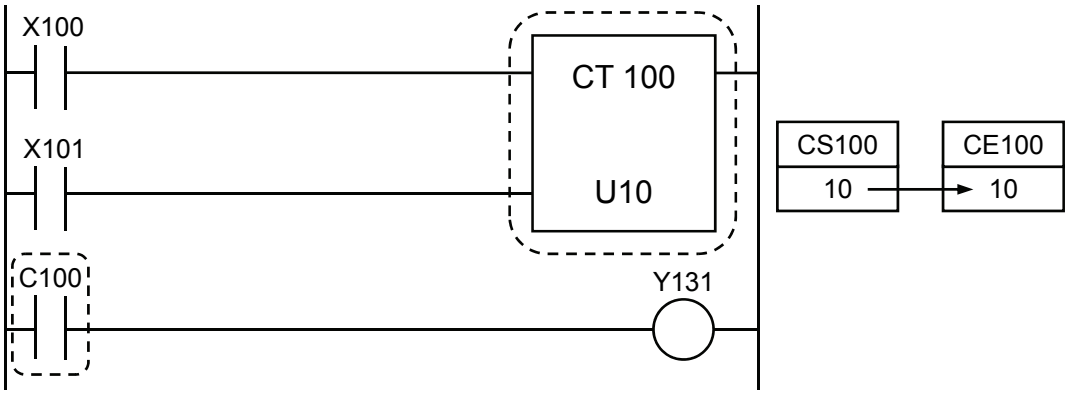
■ Mechanism of down counter operation

- This is an example when the U constant is used to specify the set value. Refer to the next page for an example of operation with the set value area number specified. (This example assumes that 100 is set to the counter.)

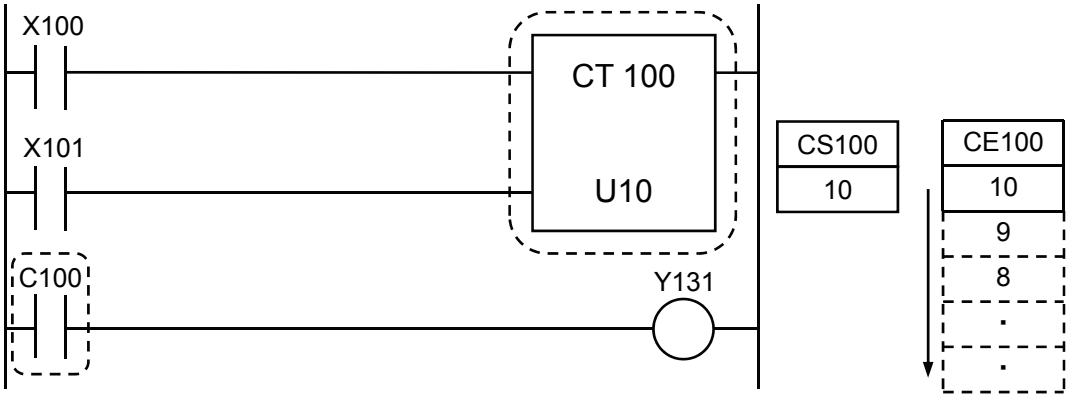
1) When the operation mode is switched to RUN or the power is turned on in the RUN mode, the counter set value is transferred to the set value area "CS" with the same number.



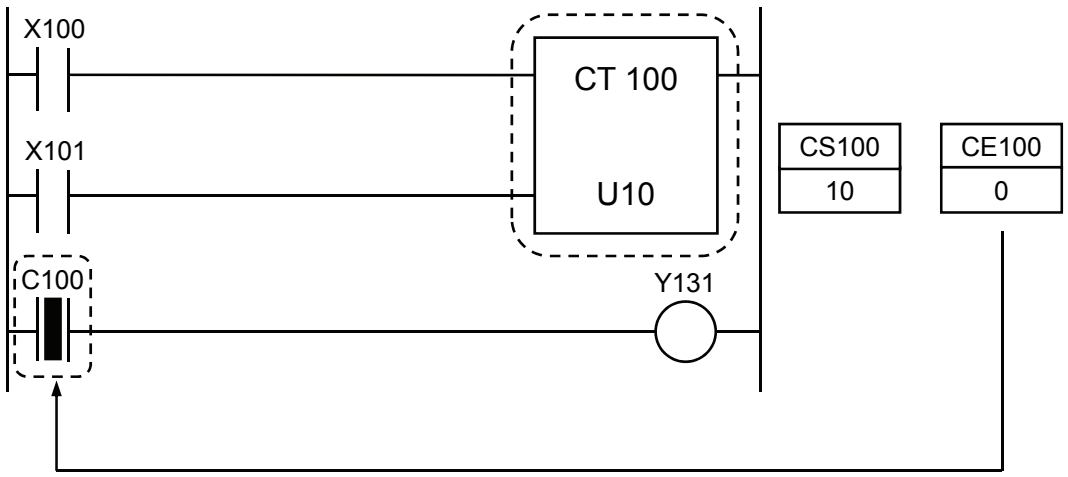
2) At the falling edge of the reset input, the value in the set value area "CS" is preset to the elapsed value area "CE".



3) Whenever the count input X100 turns on, the value in the elapsed area "CE" is decremented.



4) When the value in the elapsed value area "CE" becomes 0, the counter contact "C" with the same number turns on.



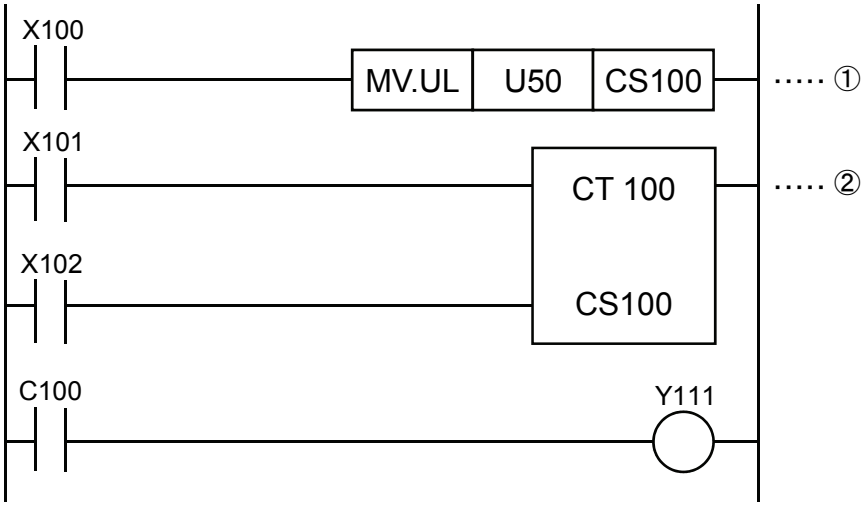
■ How to specify the set value area number directly

- The above program with CS100 specified to the set value operates as follows.

1) When the execution condition X100 is on, the data transfer instruction "MV" is executed to start decrementing with U30 set to CS100.

2) When the count input X101 is on, decrementing starts from the set value 30.

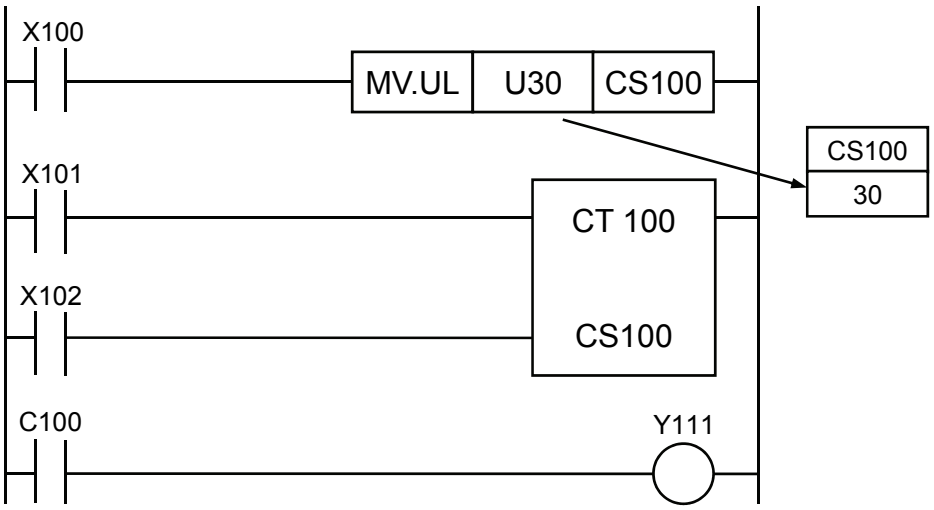
- Be sure to specify the same address as the counter number in [S] for the set value area "CS".



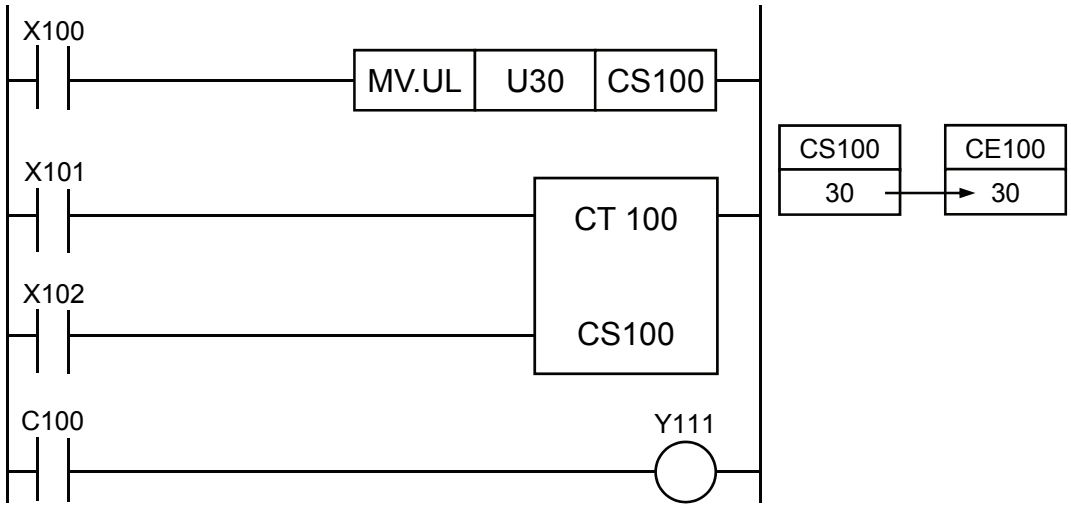
- Even when the value in the set value area "CS" is changed while decrementing, the decrement operation continues with the value before change. The new value will be used for the counter operation when the counter input changes from off to on next time.

■ Mechanism of down counter operation when the set value area number is specified directly

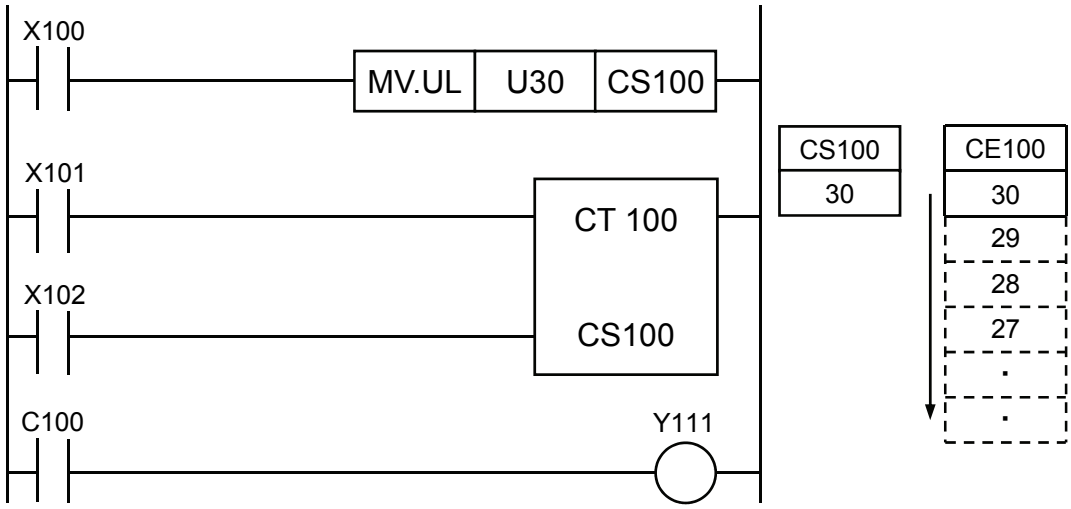
1) When the execution condition of the high-level instruction is on, the value is set in the set value area "CS". The following shows an example of using the MV instruction.



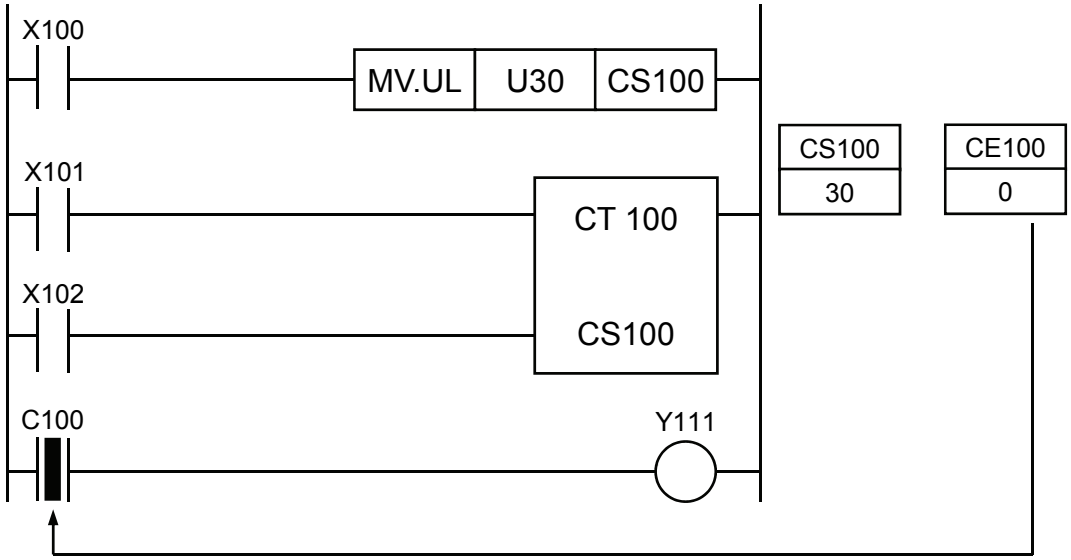
2) At the falling edge of the reset input, the value in the set value area "CS" is preset to the elapsed value area "CE".



3) Whenever the count input X101 turns on, the value in the elapsed area "CE" is decremented.



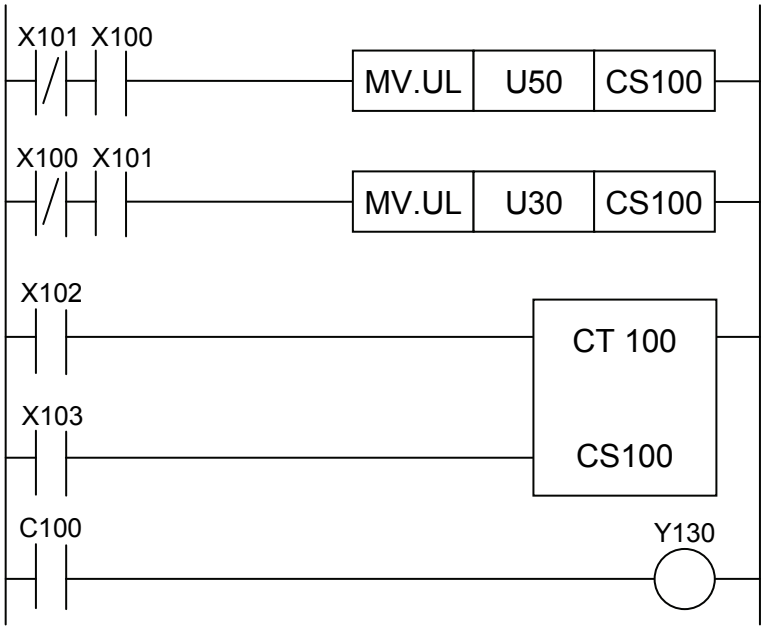
4) When the value in the elapsed value area "CE" becomes 0, the counter contact "C" with the same number turns on.



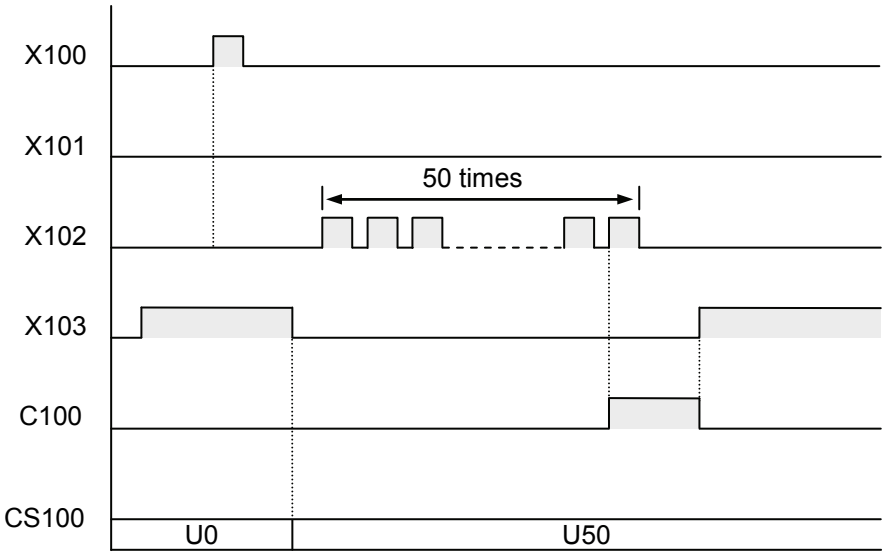
■ **Application example when the set value area number is specified directly**

<Example> Switching set values according to the condition

● Ladder diagram

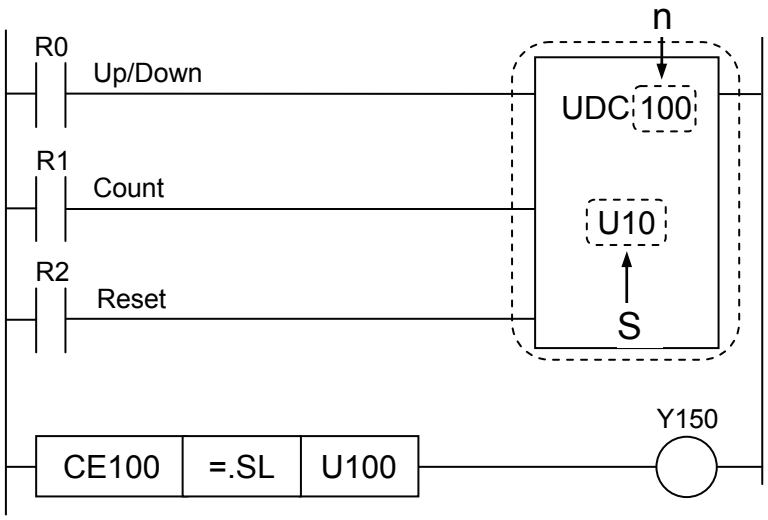


● Time chart



UDC (Up/Down Counter)

■ Ladder diagram



■ Operand list

Operand	Explanation
n	Counter No.
S	Counter set value

■ Available Devices (●: Available)

Operands	16-bit device											32-bit device			Integer			Real numbers		String	Index modifier *2
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS *1	TE CE	IX	K	U	H	SF	DF	" "	
n																●					●
S	●	●	●	●			●	●				●				●					●

*1: Only CS can be specified for CT instruction.
*2: Only 16-bit device, 32-bit device, or integer constants can be modified (but not the real number or string constants).

■ Description

- This counter can increment (up count) or decrement (down count) depending on the on/off state of the relay specified by the up/down input.
- It increments (+1) when the up/down input is on and decrements (-1) when it is off. The elapsed value is stored in the [CEn] area.
- The preset value in [S] is transferred to [CEn] when the reset input changes from on to off.
- When the count input changes from off to on, the counter starts counting from the value set in [CEn].
- When the reset input turns on, the elapsed value in [CEn] will be cleared.
- The count result can be judged by comparing the elapsed value in [CEn] with an arbitrary set value by the data comparison instruction.
- Be sure to execute the data comparison instruction immediately after the UDC instruction.

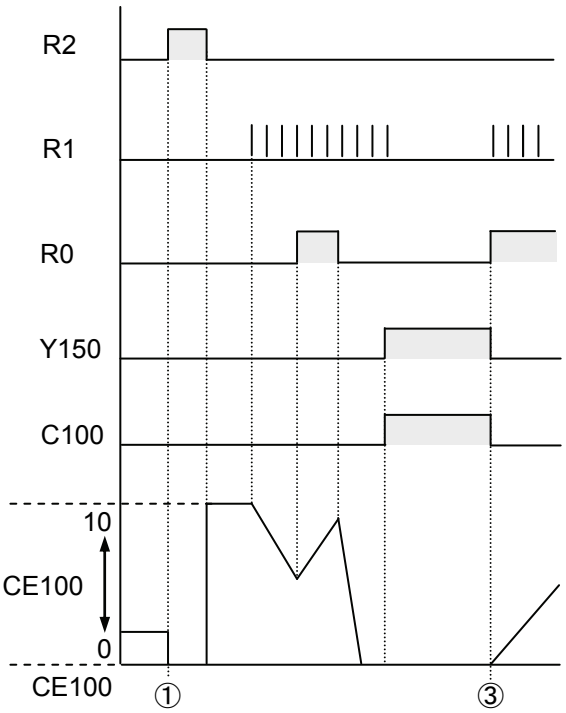
■ Regarding count value setting

- The count value is specified within the range from U1 to 4294967295, using a decimal (U) constant.

■ Example of Operation

1) When the reset input R2 changes from on to off, the set value: U100 is transferred to the elapsed value: CE100. This value is used as the target.
2) When R1 turns on while R0 is off, 1 is subtracted from the elapsed value: CE100 (decremental counting).

   When R1 turns on while R0 is on, 1 is added to the elapsed value: CE100 (incremental counting).
3) The counter elapsed value: CE100 and U0 are compared, and if CE100=U0, the external output Y150 turns on.

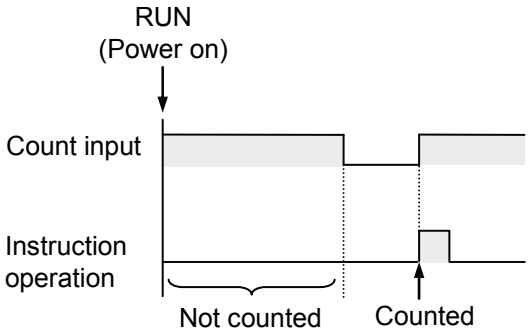


■ Precautions during programming

- If a hold type memory area is specified as the elapsed value area, the held value is treated in accordance with the held value.
- Note that the set value is not automatically preset to the elapsed value area at the start of the operation. To preset the value, turn on then off the reset input.
- When using a UDC instruction with an AND stack (ANS) instruction or pop stack (POPS) instruction, be sure to write the code correctly.

■ Precaution on count input detection

- The UDC instruction detects rising from off to on of the count input and increments or decrements the set value.
- While the count input remains on, counting is performed at the rising edge only and not performed later.
- The set value will not be incremented or decremented for the first scan if the count input is already on when the operation mode is switched to RUN or the power is turned on in the RUN mode.

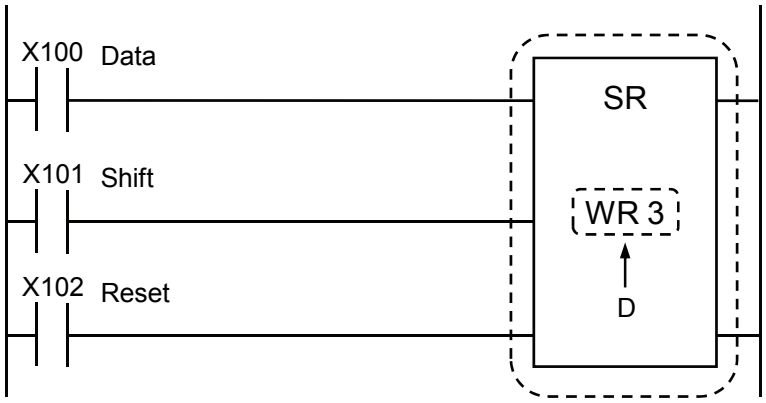


- Be careful when using this instruction with an instruction (1 to 6 below) which changes the order of instruction execution such as MC, MCE, JP or LBL, because the operation of the instruction may change depending on the instruction execution and count input timing.

1) MC and MCE instructions
2) JP and LBL instructions
3) LOOP and LBL instructions
4) CNDE instructions
5) Step ladder instructions
6) Subroutine instructions

SR (Shift Register)

■ Ladder diagram



■ Operand list

Operand	Explanation
D	Shifted device

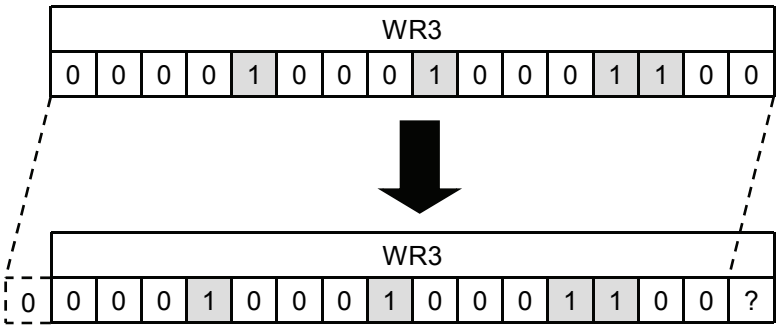
■ Available Devices (●: Available)

Operand	16-bit device											32-bit device			Integer			Real numbers		String	Index modifier
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX	K	U	H	SF	DF	" "	
D			●																		

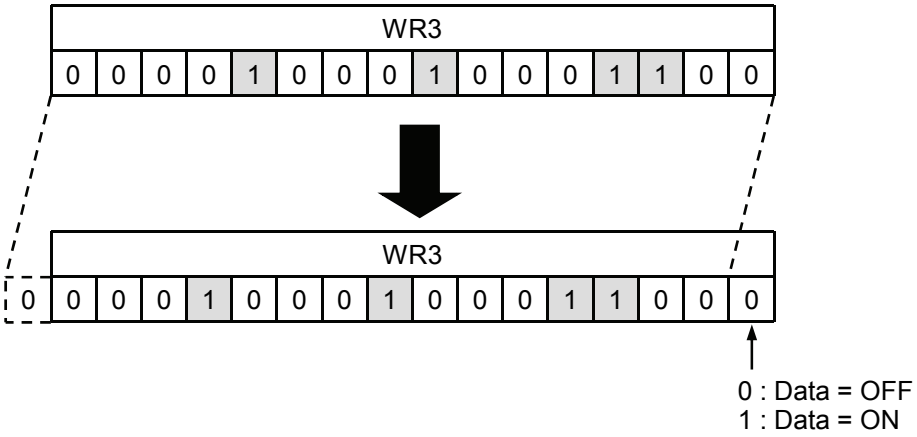
■ Description

- Moves (i.e., shifts) the contents of the specified register WR (specified operation unit) to the left by one bit.

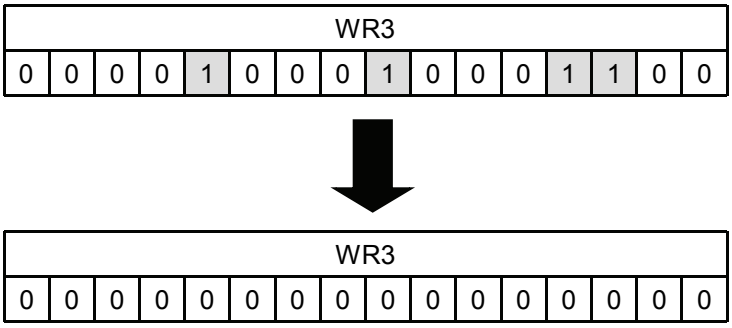
1) When the shift input turns on (rises), shifts the contents of WR to the left by one bit.



2) Upon shifting, sets 1 or 0 to the blank bit (lowest bit) if the data input is on or off, respectively.

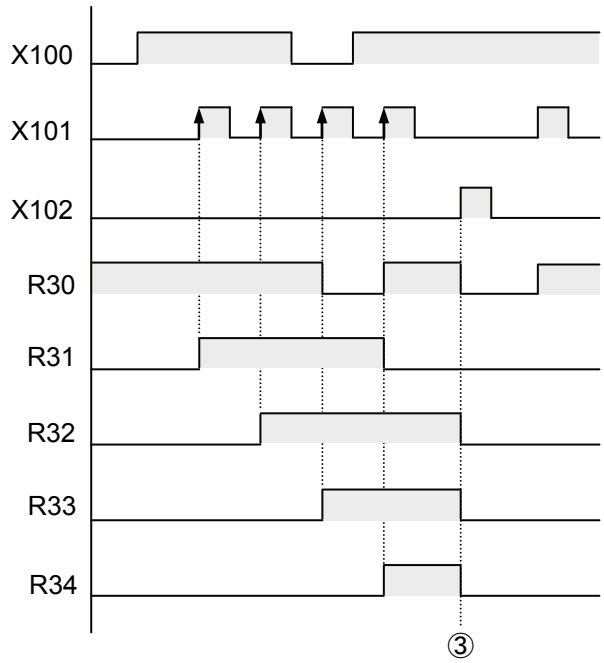


3) When the reset input turns on, the specified register contents will be cleared.



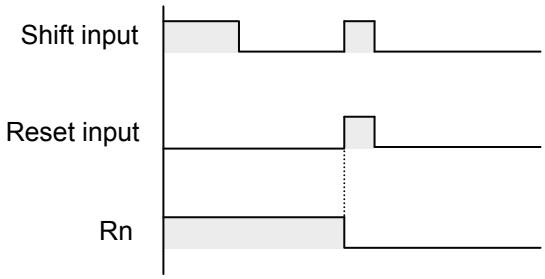
■ Example of Operation

1) When X101 turns on while X102 is off, the contents of WR3 (internal relays R30 to R3F) are shifted by 1 bit to the left.
2) In the bit that has become blank due to left shifting (R30), 1 is set when X100 is on and 0 is set when X100 is off.
3) When X102 turns on, the contents of WR3 are reset to 0.



■ Precautions during programming

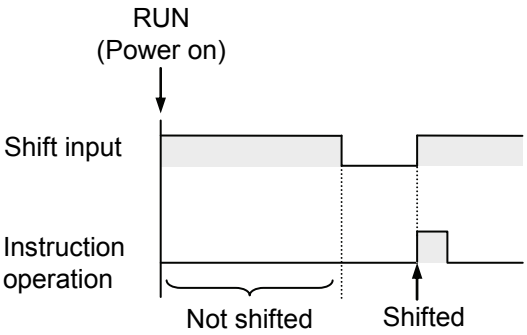
- The data input, shift input, and reset input are required for the SR instruction.
- If the reset input and the shift input turn on simultaneously, the reset input will take precedence.



- Note that if a hold-type memory area is specified for the shift register, it will not be automatically reset upon power on.
- When using a shift register instruction with an AND stack (ANDS) instruction or pop stack (POPS) instruction, be sure to write the code correctly.

■ Precaution on shift input detection

- The SR instruction detects rising from off to on of the shift input and shift the register contents.
- While the shift input remains on, shifting is performed at the rising edge only and not performed later.
- The register contents will not be shifted for the first scan if the shift input is already on when the operation mode is switched to RUN or the power is turned on in the RUN mode.



- Be careful when using this instruction with an instruction (1 to 6 below) which changes the order of instruction execution such as MC, MCE, JP or LBL, because the operation of the instruction may change depending on the instruction execution and shift input timing.

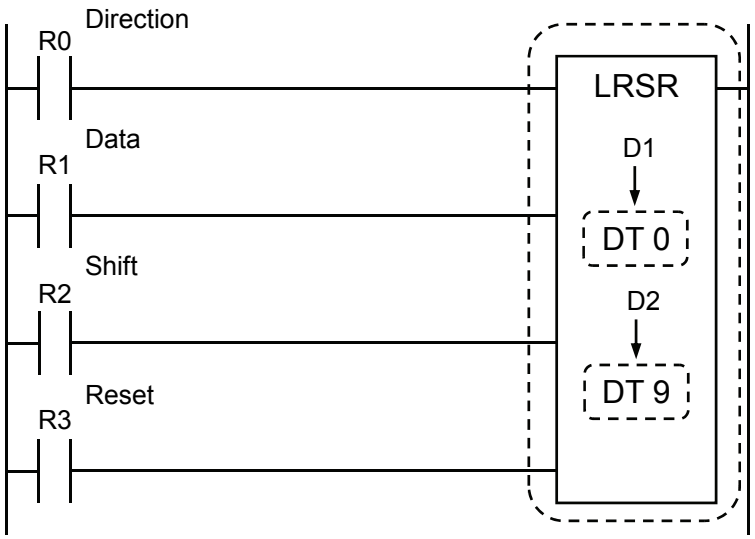
1) MC and MCE instructions
2) JP and LBL instructions
3) LOOP and LBL instructions
4) CNDE instructions
5) Step ladder instructions
6) Subroutine instructions

■ Related instructions

- The left/right shift register (LRSR) instruction is also provided in addition to this instruction for the shift register. You can also use the data shift or data rotate instruction to implement the same operation.

LRSR (Left/Right Shift Register)

■ Ladder diagram



■ Operand list

Operand	Explanation
D1	Shift start position
D2	Shift end position

■ Available Devices (●: Available)

Operands	16-bit device											32-bit device			Integer			Real numbers		String	Index modifier
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX	K	U	H	SF	DF	" "	
D1	●	●	●	●			●	●	●		●										
D2	●	●	●	●			●	●	●		●										

■ Description

- Left/right shift is a shift register which shifts 1 bit of the specified data area to the left (to the higher bit position) or to the right (to the lower bit position), depending on the on/off state of the relay specified as the left/right shift input.
- It shifts to the left or right when the left/right shift input is on or off, respectively.
- The variables [D1] and [D2] have to be of the same area type. Be sure that [D1] is equal to or smaller than [D2].

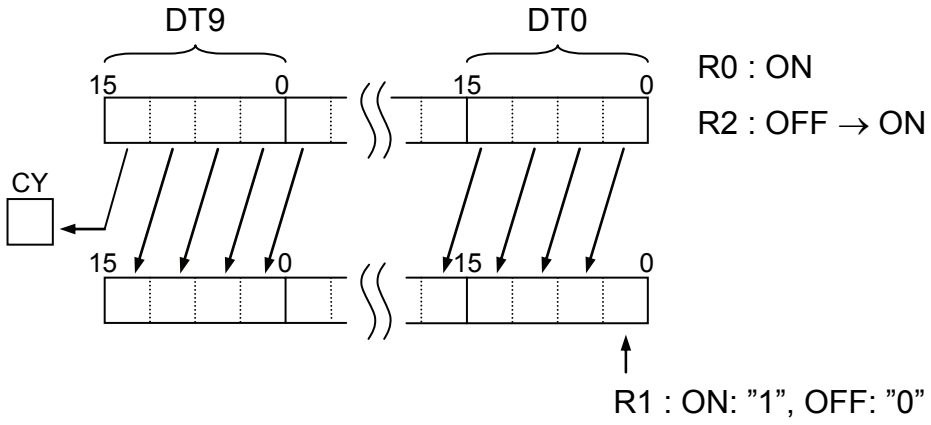
■ Operation of LRSR (Left/Right Shift Register)

1) When the shift input changes from off to on (with the reset input off), shifts the contents of the area specified by [D1] and [D2] to the left or right by one bit.

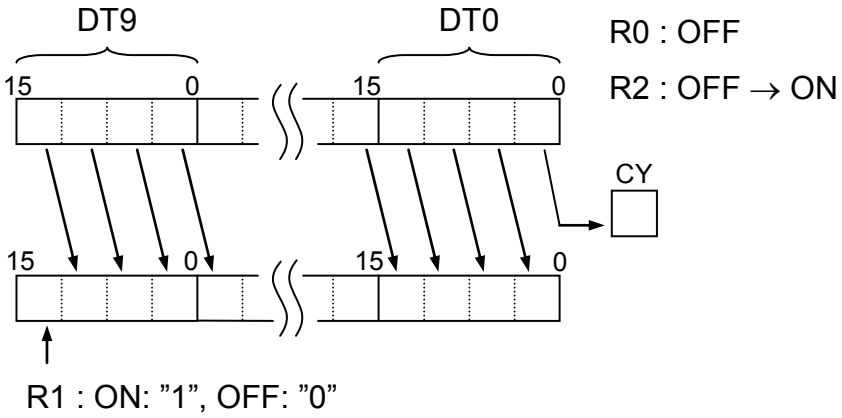
2) Upon shifting, sets 1 or 0 to the bit (highest bit or lowest bit) which has become blank due to shifting if the data input is on or off, respectively. In addition, the bit pushed out due to shifting (the highest bit for left shift and the lowest bit for right shift) is set in the system relay SR9 (carry flag).

3) When the reset input turns on, the specified area contents will be cleared.

■ **Operation example: Left shift**



■ **Operation example: Right shift**

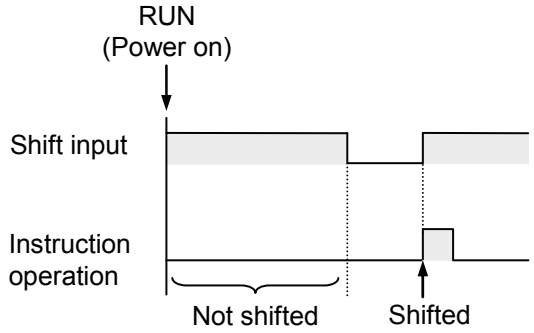


■ **Flag conditions**

Name	Explanation
SR7 SR8 (ER)	Turns on when [D1] address > [D2] address.
SR9(CY)	Turns on when the value pushed out due to shifting is "1".

■ Precaution on shift input/output

- The LRSR instruction detects rising from off to on of the shift input and shift the register contents.
- While the shift input remains on, shifting is performed at the rising edge only and not performed later.
- The register contents will not be shifted for the first scan if the shift input is already on when the operation mode is switched to RUN or the power is turned on in the RUN mode.



- Be careful when using this instruction with an instruction (1 to 6 below) which changes the order of instruction execution such as MC, MCE, JP or LBL, because the operation of the instruction may change depending on the instruction execution and shift input timing.

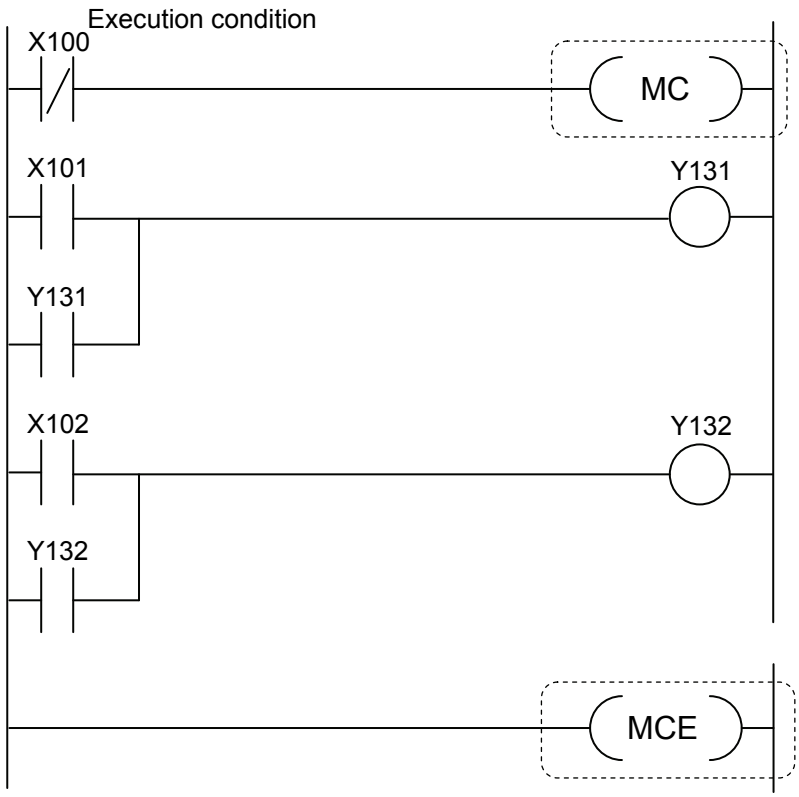
1) MC and MCE instructions
2) JP and LBL instructions
3) LOOP and LBL instructions
4) CNDE instructions
5) Step ladder instructions
6) Subroutine instructions

■ Precautions during programming

- When using a LRSR instruction with an AND stack (ANS) instruction or pop stack (POPS) instruction, be sure to write the code correctly.

MC (Master Control Relay)
MCE (Master Control Relay End)

■ **Ladder diagram**



■ Description

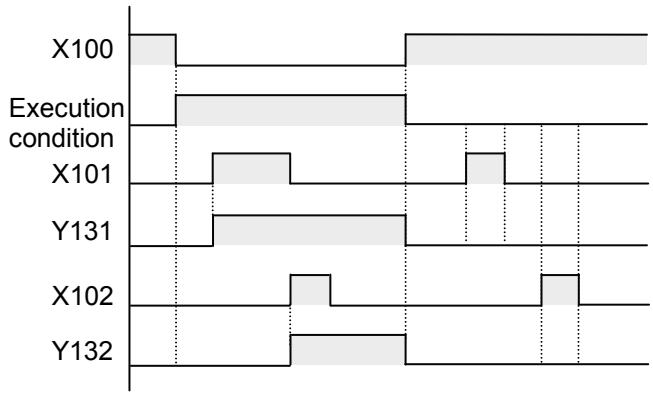
- When the execution condition is on, runs the program code between the MC and MCE instructions.
- When the execution condition is off, the state of each input and output relays is as follows.

Type of instruction	Operation
OT	All off
KP	State is held
SET	
RST	
TM	Reset
CT	Intermediate result is held
SR	
Differential instruction	Operates in the same way as differential instructions used between MC and MCE Refer to Operation of differential instructions between MC and MCE.
Other instructions	Not executed

- Be careful when using an instruction which detects the leading edge of the execution condition and runs (1 - 6 below), including a differential instruction.

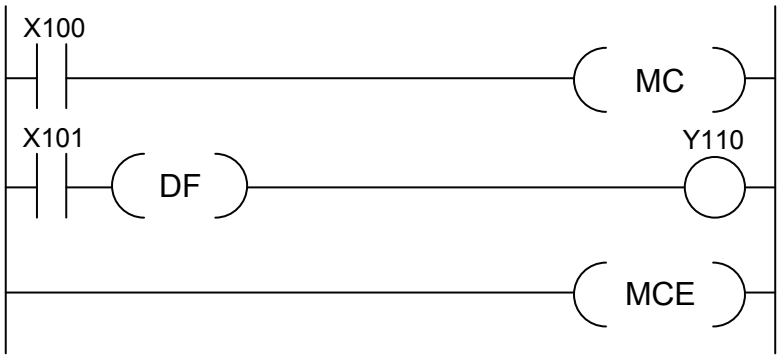
1) DF (leading edge differential) instruction
2) Count input for CT (counter) instruction
3) Count input for UDC (up-down counter) instruction
4) Shift input for SR (shift register) instruction
5) Shift input for LRSR (left and right shift register)
6) Differential execution type high-level instruction (instruction specified by p and instruction name)

■ Example of Operation

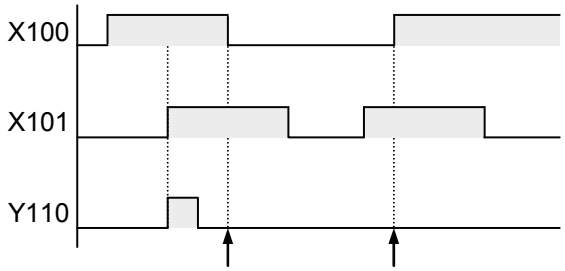


■ Operation of differential instructions between MC and MCE

- If the differential instruction is used between MC and MCE, the output to be obtained depends on the execution condition of MC and the input timing of the differential instruction as mentioned below.



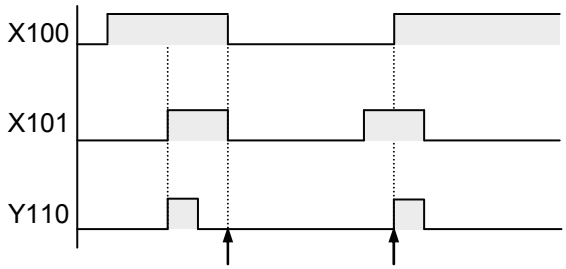
<Timing Chart 1>



Last differential instruction execution

Since the execution condition X101 for the differential instruction has not changed from the last execution, no differential output is obtained.

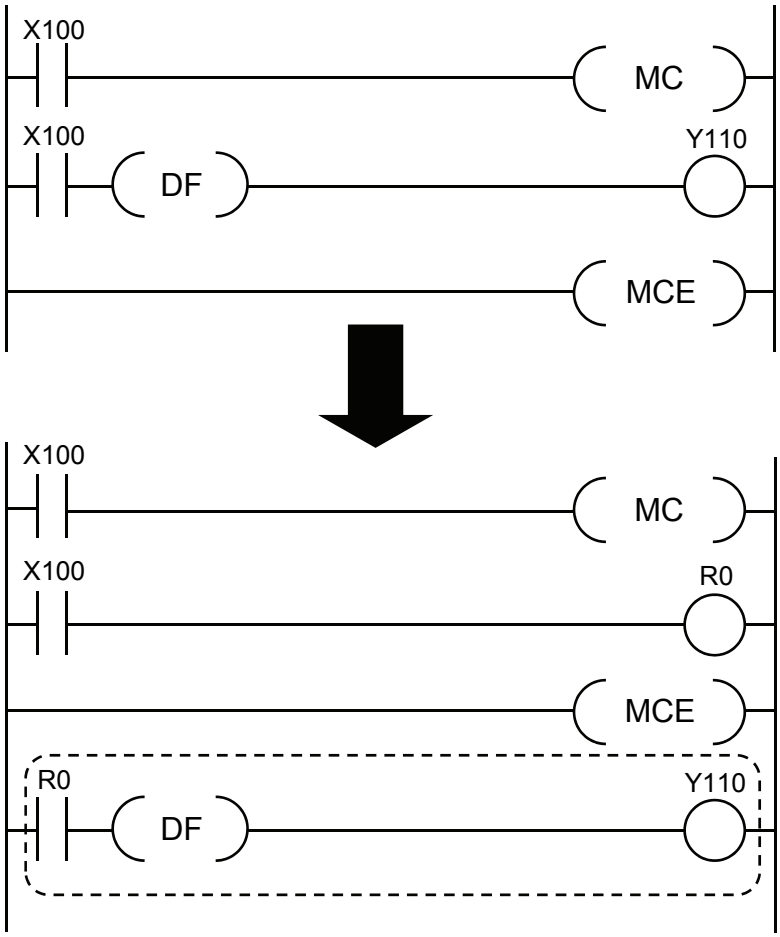
<Timing Chart 2>



Last differential instruction execution

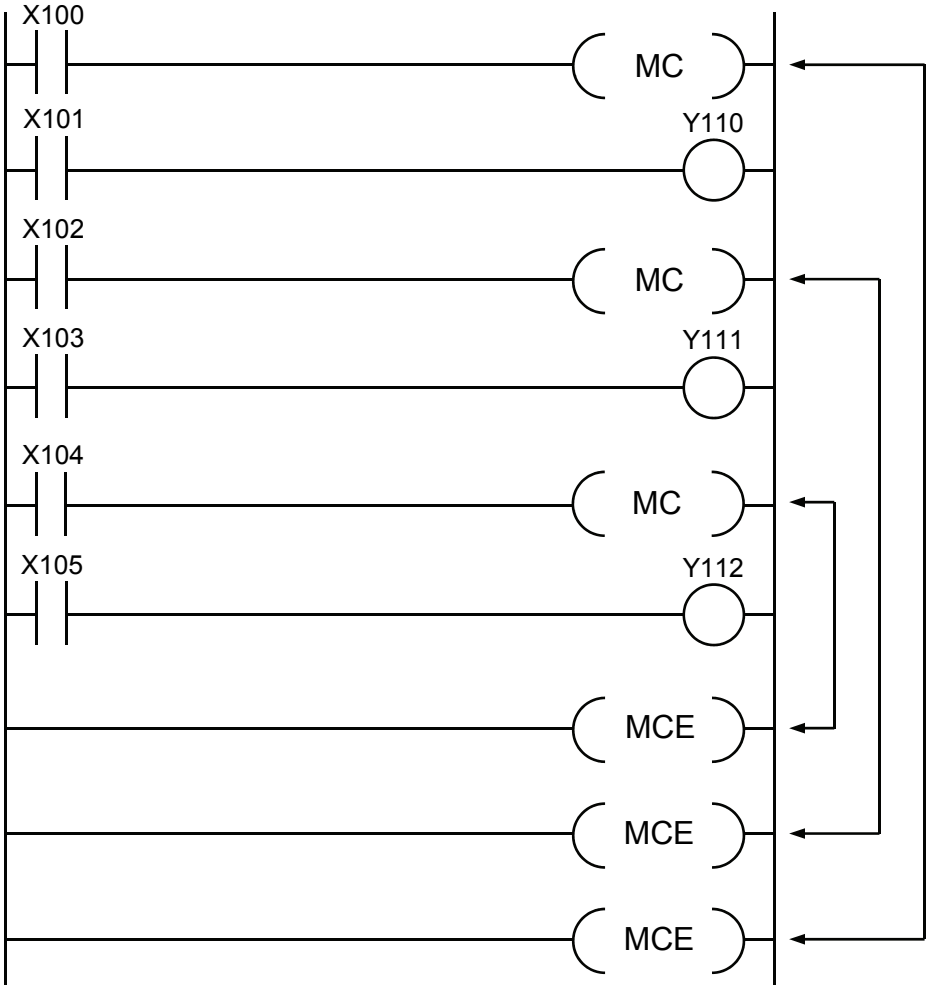
Since the execution condition X101 for the differential instruction has changed from off to on from the last execution, differential output is obtained.

- If the same execution condition is specified for the MC and differential instructions, no output will be generated. If the output is necessary, be sure to write the differential instruction outside the range between MC and MCE.



■ Precautions for Programming

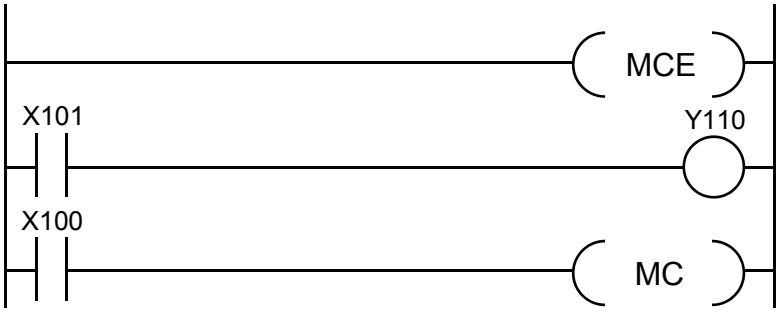
- A pair of the MC and MCE instructions can be nested within another pair of the MC and MCE instructions. (Up to 30 levels of nesting are allowed for MC and MCE.)



- The program code will not be executed in following cases:

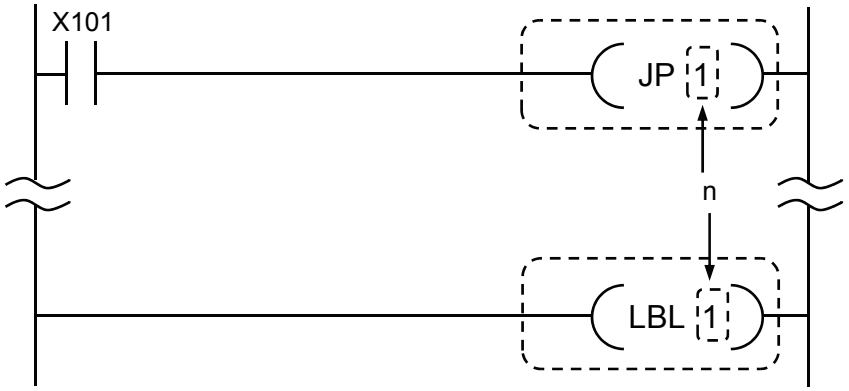
1) MC or MCE is missing in a pair.

2) The order of MC and MCE is reversed.



JP, LBL (Jump, Label)

■ Ladder diagram



■ Operand list

Operands	Explanation
n	Label number (relative jump pointer to LBL)

■ Available Devices (●: Available)

Operands	16-bit device											32-bit device			Integer			Real numbers		String	Index modifier
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX	K	U *1	H	SF	DF	" "	
n																●					

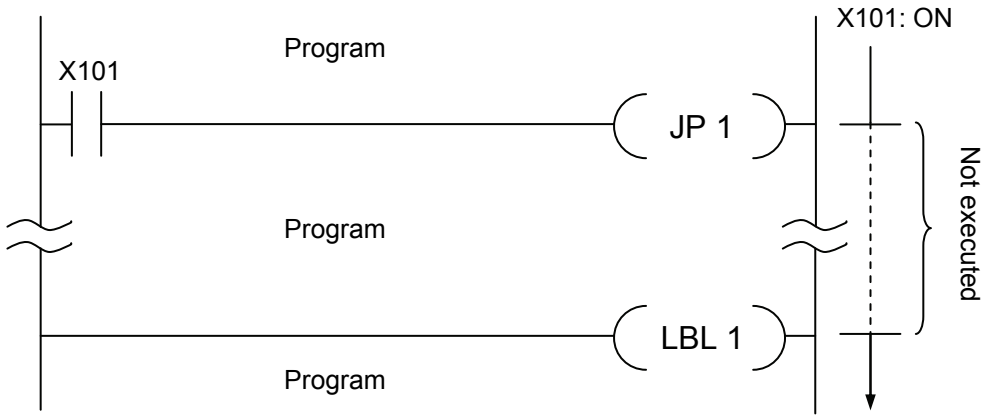
*1: Can be specified only when the operation unit is unsigned integer (US, UL).

■ Description

- When the execution condition is on, jumps to the label with the specified number (LBL instruction).
- The program continues running from the instruction after the target label.

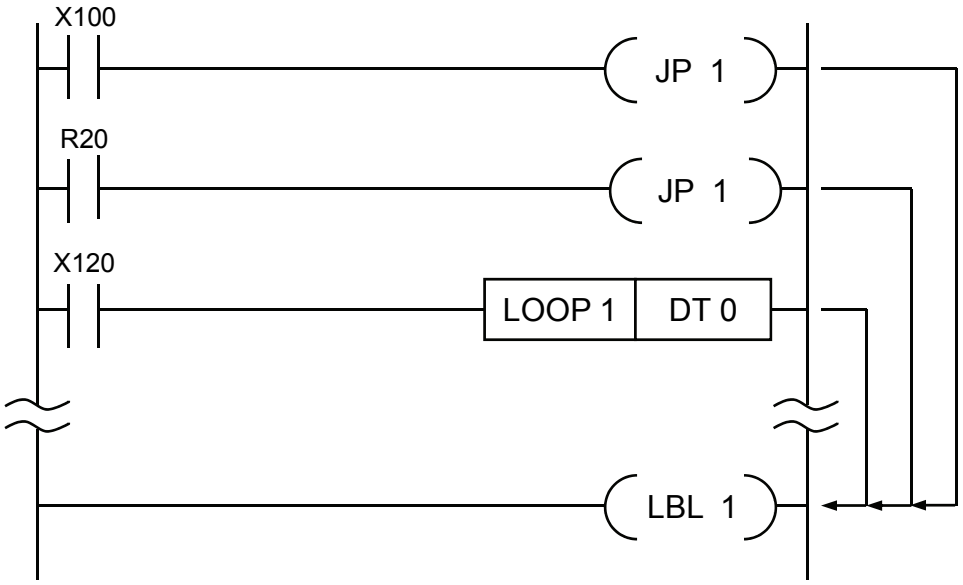
■ Example of Operation

- When the execution condition X101 is on, the program jumps to the label 1.



■ Precautions during programming

- The JP instruction specifying the same label number can be used for several times.

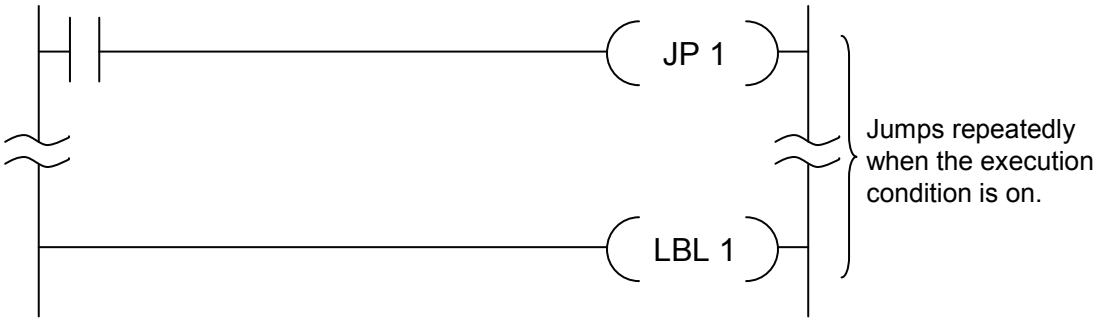


- The LBL instruction specifying the same number can only be written once in a single program.
- If the target label is not programmed, a syntax error will occur.
- Note that if the label is written at an address before the JP instruction, the program cannot end scanning and a WDT error may occur.
- The JP and LBL instructions cannot be used in the step ladder area (within the range from the SSTP to STPE instructions).
- It is not possible to jump from the main program to a subprogram (subroutine or interrupt program after the ED instruction), from a subprogram to the main program, or between subprograms.
- Be careful when using an instruction which detects the leading edge of the execution condition and runs (1 - 6 below), including a differential instruction.

1) DF (leading edge differential) instruction
2) Count input for CT (counter) instruction
3) Count input for UDC (up-down counter) instruction
4) Shift input for SR (shift register) instruction
5) Shift input for LRSR (left and right shift register)
6) Differential execution type high-level instruction (instruction specified by p and instruction name)

■ Regarding operation of TM, CT, and SR instructions between JP and LBL instruction

- If the LBL instruction is located at an address after the JP instruction, each instruction is processed as follows when the JP instruction is executed.

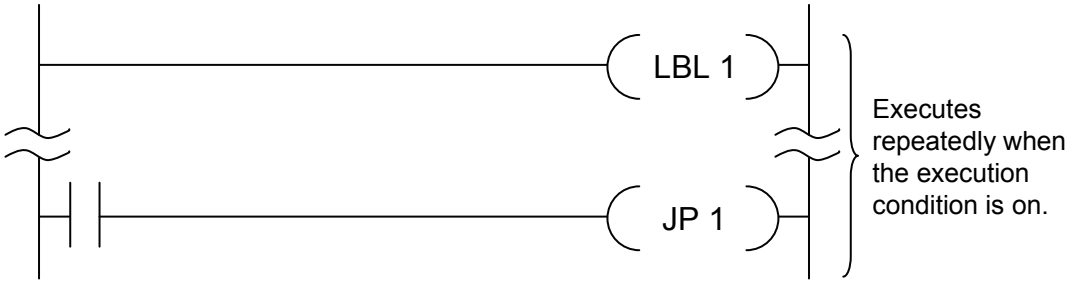


1) TM instruction: Not counted. Note that time is not guaranteed if counting does not occur during a single scan.

2) CT instruction: Not counted even if the count input is on. The elapsed value is held.

3) SR instruction: Not shifted even if the shift input is on. The specified register contents are held.

- If the LBL instruction is located at an address before the JP instruction, each instruction is processed as follows when the JP instruction is executed.



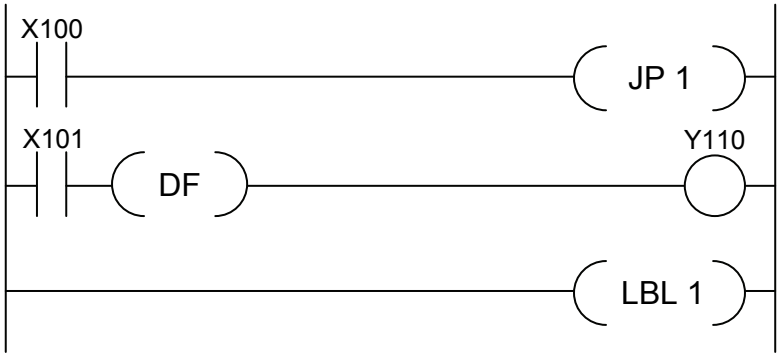
1) TM instruction: Time is not guaranteed because counting occurs several times during a single scan. (Note)

2) CT instruction: Operates normally if the count input does not change its state during the scan.

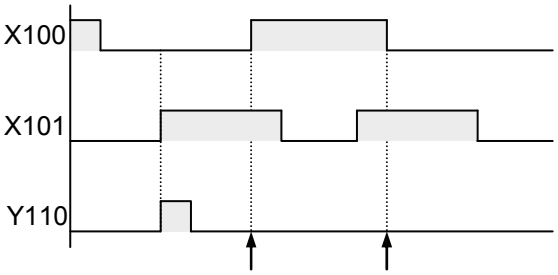
3) SR instruction: Operates normally if the shift input does not change its state during the scan.

■ Operation of differential instruction between JP and LBL instructions

- If the differential instruction is used between JP and LBL, the output to be obtained depends on the execution condition of JP and the input timing of the differential instruction as mentioned below.



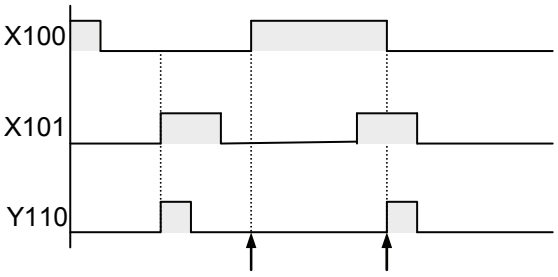
<Timing Chart 1>



Final timing with no execution of the last JP instruction

Since the execution condition X101 for the differential instruction has not changed from the final timing with no execution of the last JP instruction, no differential output is obtained.

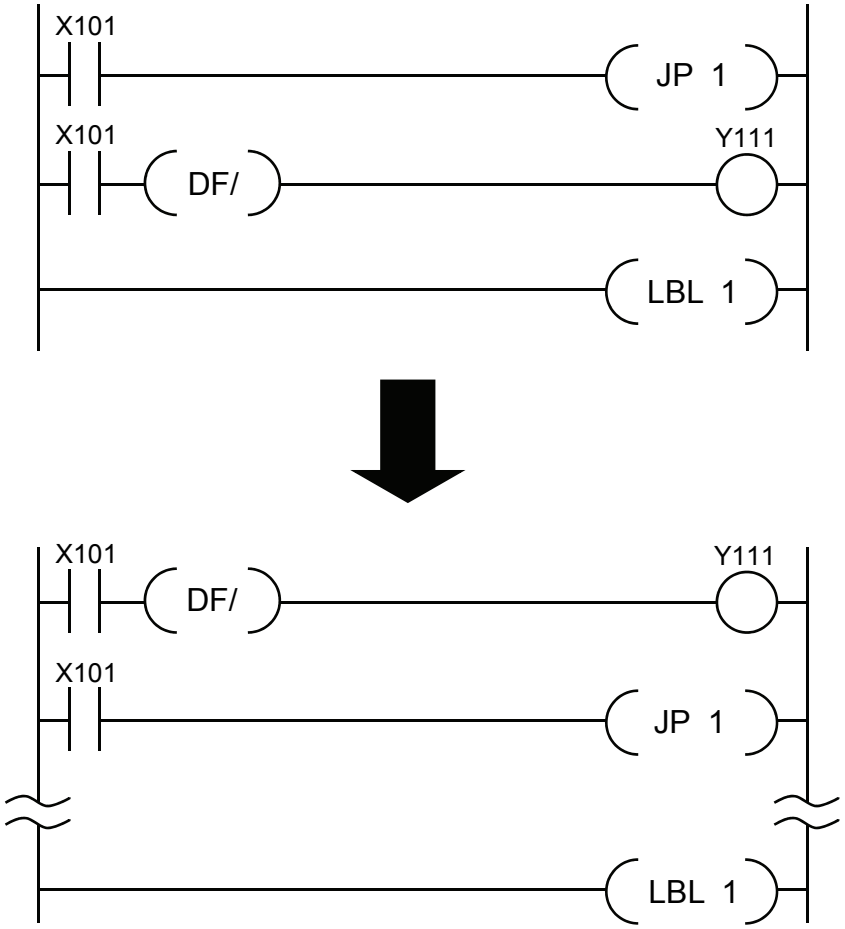
<Timing Chart 2>



Final timing with no execution of the last JP instruction

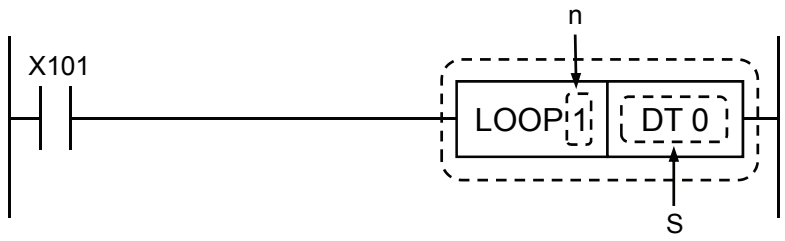
Since the execution condition X101 for the differential instruction has changed from off to on from the final timing with no execution of the last JP instruction, differential output is obtained.

- If the same execution condition is used for the JP and differential instructions, rising (or falling) of the execution condition for the differential instruction will not be detected. If the differential output is necessary, be sure to write the differential instruction outside the range between JP and LBL.



LOOP, LBL (LOOP, Label)

■ Ladder diagram



■ Operand list

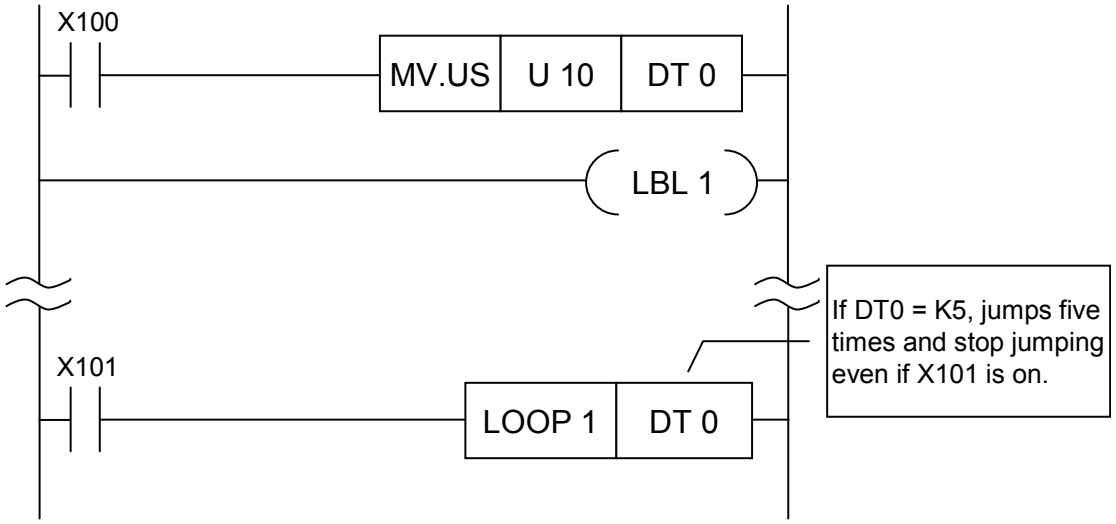
Operand	Explanation
n	Label number
S	Loop count

■ Available Devices (●: Available)

Operands	16-bit device											32-bit device			Integer			Real numbers		String	Index modifier
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX	K	U	H	SF	DF	" "	
n																●					
S	●	●	●	●			●	●													●

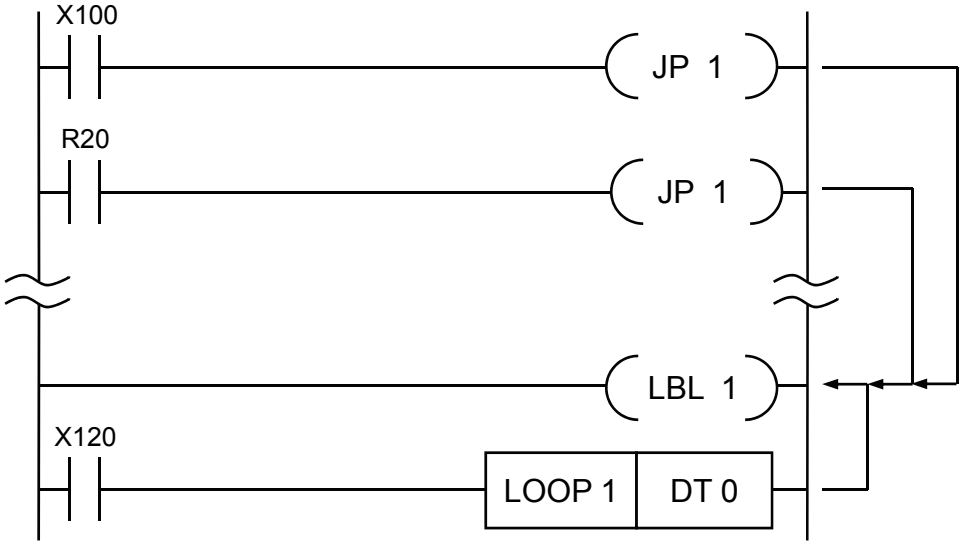
■ Description

- If the condition is on, 1 is subtracted from the value in [S]. If the result is not 0, the instruction jumps to the label with the specified number (LBL instruction).
- The program continues running from the instruction after the target label.
- The LOOP instruction specifies the number of times to execute the code. When the code has been executed for the number of times specified in [S], the program will not jump to the specified label even if the execution condition is on.



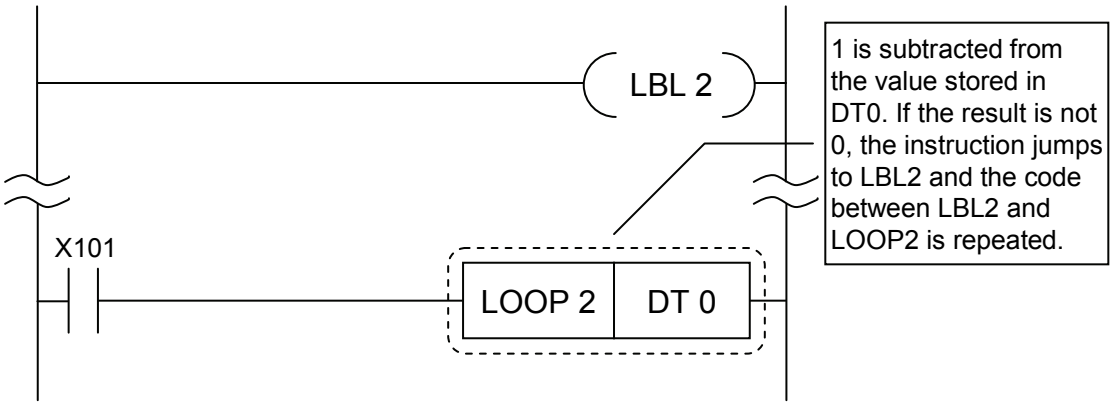
- If the value in the memory area specified in [S] is 0 for the first time, the program does not jump to the specified label but performs the next process.

- The label is shared between the JP and LOOP instructions. The label can be used as the jump target for any number of times from any instructions.



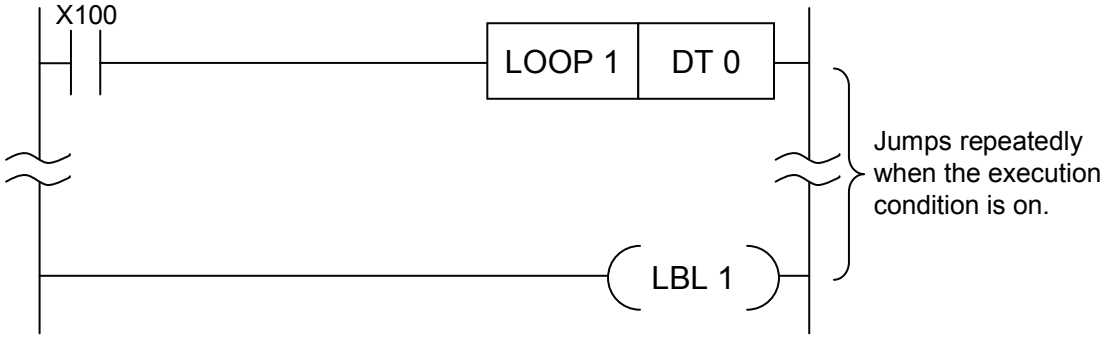
- The LBL instruction specifying the same number can only be written once in a single program.
- If the target label is not programmed, a syntax error will occur.

■ Example of Operation



■ Regarding operation of TM, CT, and SR instructions between LOOP and LBL instruction

- If the LBL instruction is located at an address after the LOOP instruction, each instruction is processed as follows when the LOOP instruction is executed.

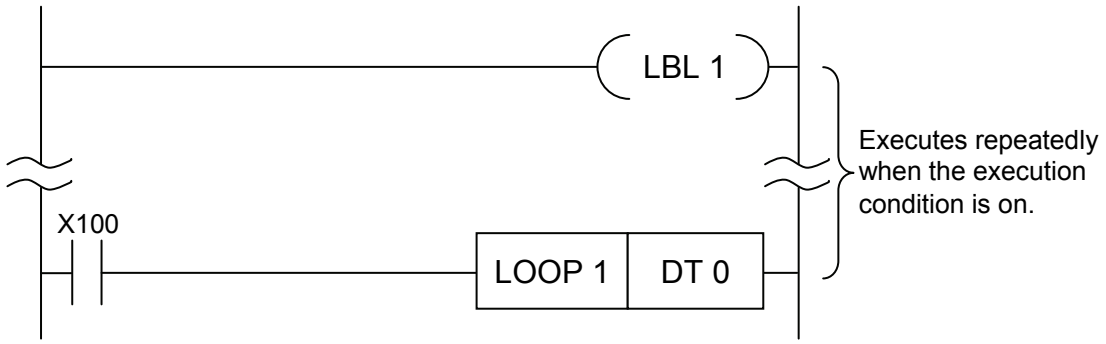


1) TM instruction: Not counted. Note that time is not guaranteed if counting does not occur during a single scan.

2) CT instruction: Not counted even if the count input is on. The elapsed value is held.

3) SR instruction: Not shifted even if the shift input is on. The specified register contents are held.

- If the LBL instruction is located at an address before the LOOP instruction, each instruction is processed as follows when the LOOP instruction is executed.



1) TM instruction: Time is not guaranteed because counting occurs several times during a single scan.

2) CT instruction: Operates normally if the count input does not change its state during the scan.

3) SR instruction: Operates normally if the shift input does not change its state during the scan.

■ Precautions during programming

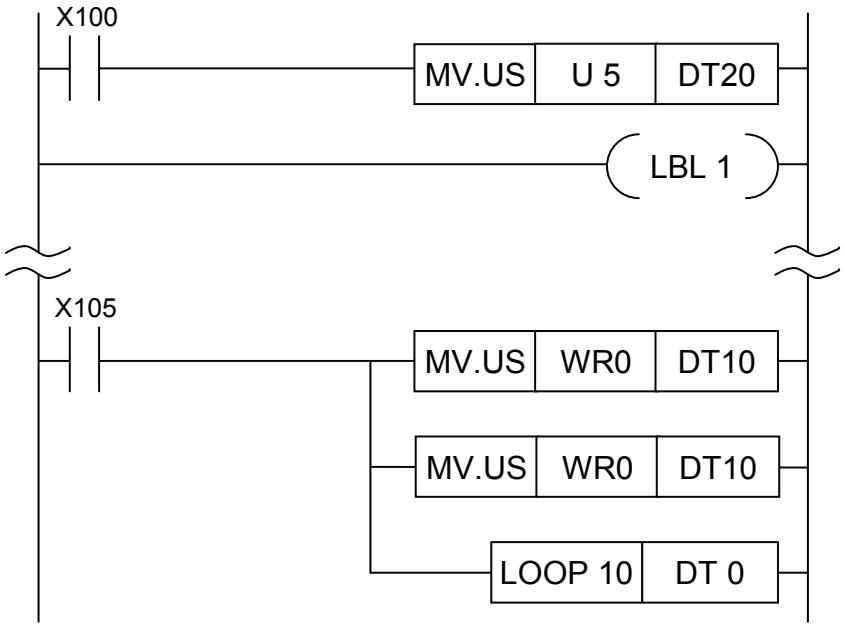
- When writing a label before the LOOP instruction, pay attention to the following:

1) Be sure to write the instruction for setting the loop count before LBL-LOOP instructions.

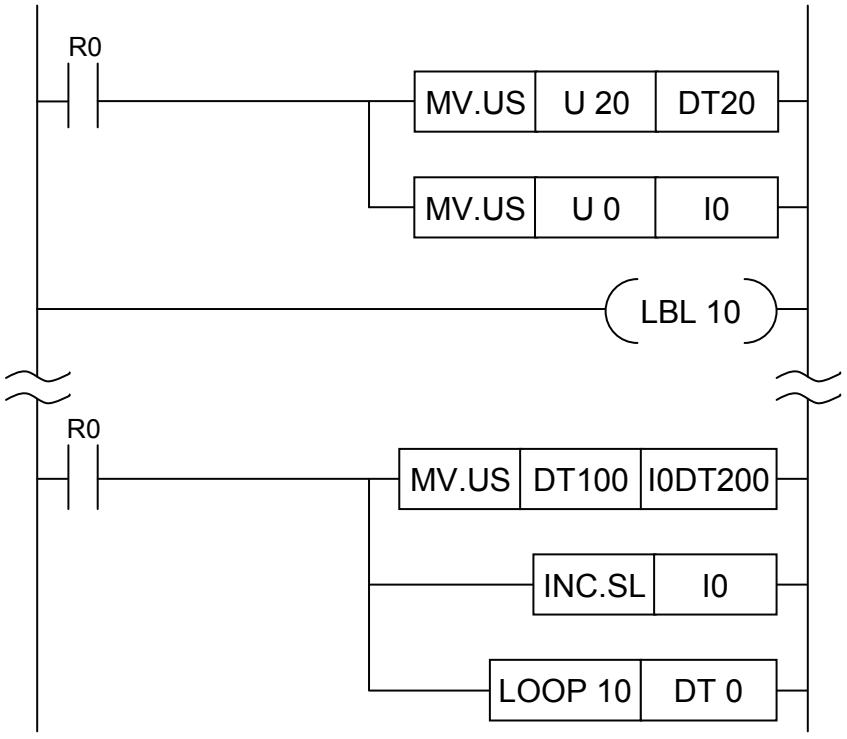
2) Be sure to write the instructions repeated between the LBL and LOOP instructions so that they are executed with the same execution condition as the LOOP instruction.

3) While repeating the program code, a single scan may exceed the WDT monitor time limit and a WDT error may occur.

<Example 1> Repeats two MV instructions for five times when X105 is on.



<Example 2> Transfers the DT100 value to DT200 to DT219.



- The LOOP and LBL instructions cannot be used in the step ladder area (within the range from the SSTP to STPE instructions).
- It is not possible to jump from the main program to a subprogram (subroutine or interrupt program after the ED instruction), from a subprogram to the main program, or between subprograms.
- Be careful when using an instruction which detects the leading edge of the execution condition and runs (1 - 6 below), including a differential instruction.

1) DF (leading edge differential) instruction

2) Count input for CT (counter) instruction

3) Count input for UDC (up-down counter) instruction

4) Shift input for SR (shift register) instruction

5) Shift input for LRSR (left and right shift register)

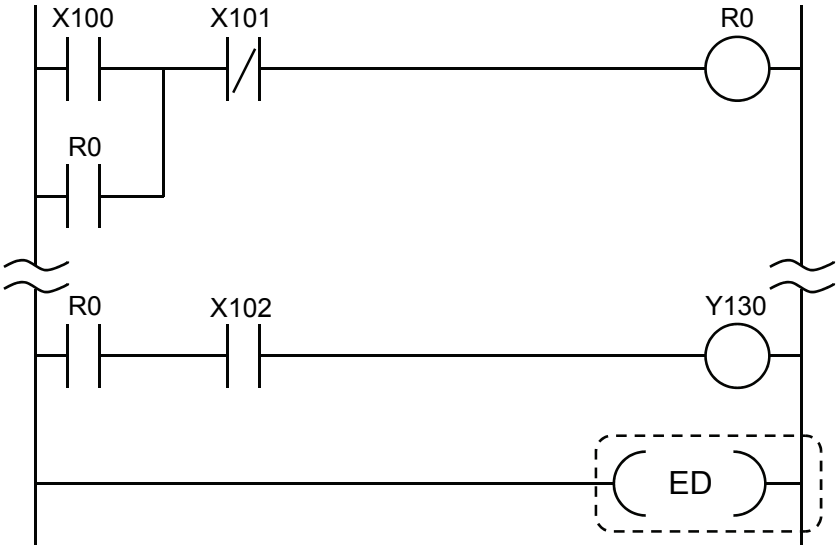
6) Differential execution type high-level instruction (instruction specified by p and instruction name)

■ Flag conditions

Name	Explanation
SR7 SR8 (ER)	On when the value in [S] is negative (highest bit is 1)

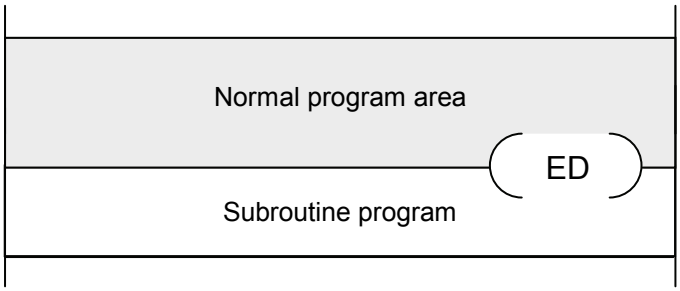
ED (End)

■ Ladder diagram



■ Description

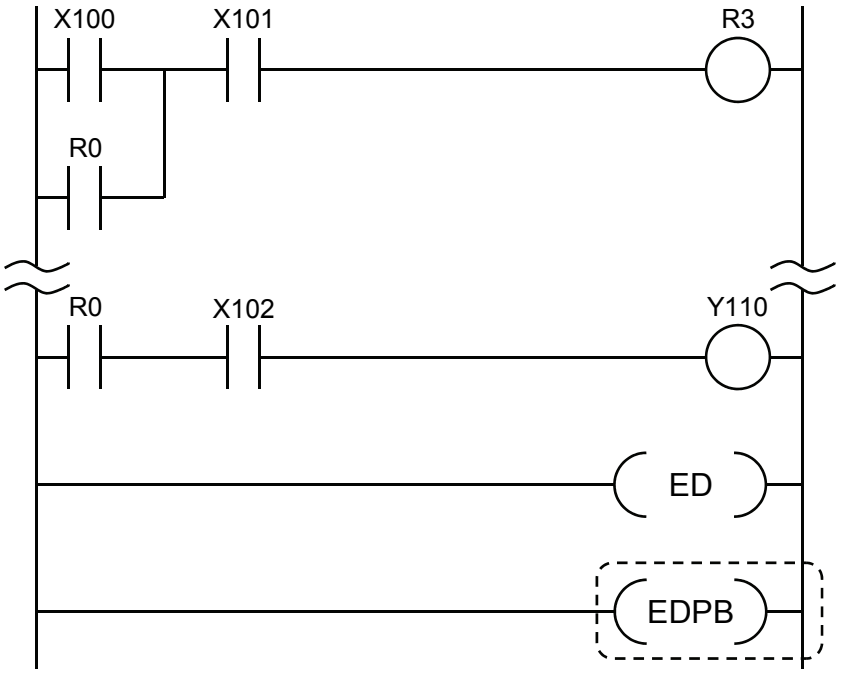
- Write the ED instruction at the end of the normal program area.



- The program area is divided into the normal program area (main program) and the "subroutine" and "interrupt program" areas (subprograms) by this instruction.
- Be sure to write the subroutine and interrupt program after the ED instruction.

EDPB (End Program Block)

■ **Ladder diagram**

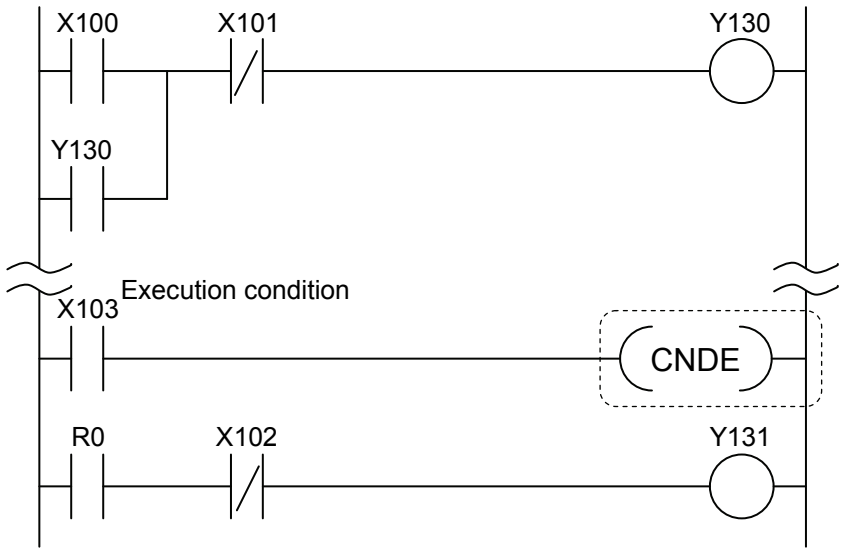


■ **Description**

- Indicates the end of PB (program block).

CNDE (Conditional End)

■ Ladder diagram



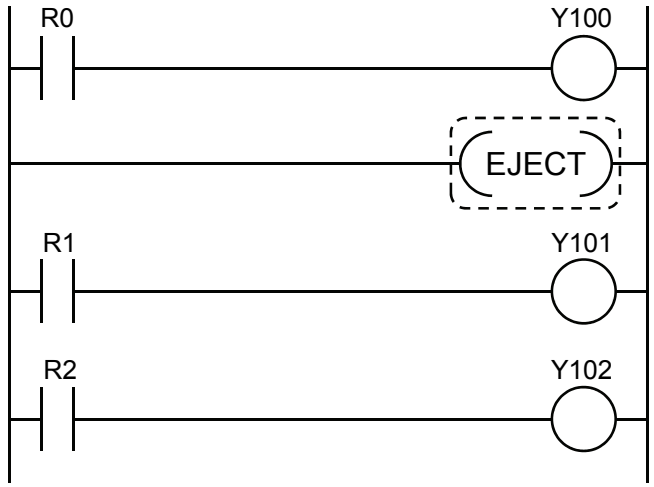
■ Description

- Ends the operation of the program at a specified address.
- When the execution condition turns on, the program terminates operation and begins I/O processing. Then the program returns to the first address.
- The process timing can be adjusted by beginning the process whenever the necessary program scan finishes.
- The CNDE instruction cannot be used in a subprogram (e.g., subroutine). Use it in the main program area.
- The CNDE instruction can be used for any number of times in the main program.
- Be careful when using an instruction which detects the leading edge of the execution condition and runs (1 - 6 below), including a differential instruction.

1) DF (leading edge differential) instruction
2) Count input for CT (counter) instruction
3) Count input for UDC (up-down counter) instruction
4) Shift input for SR (shift register) instruction
5) Shift input for LRSR (left and right shift register)
6) Differential execution type high-level instruction (instruction specified by p and instruction name)

EJECT (Eject)

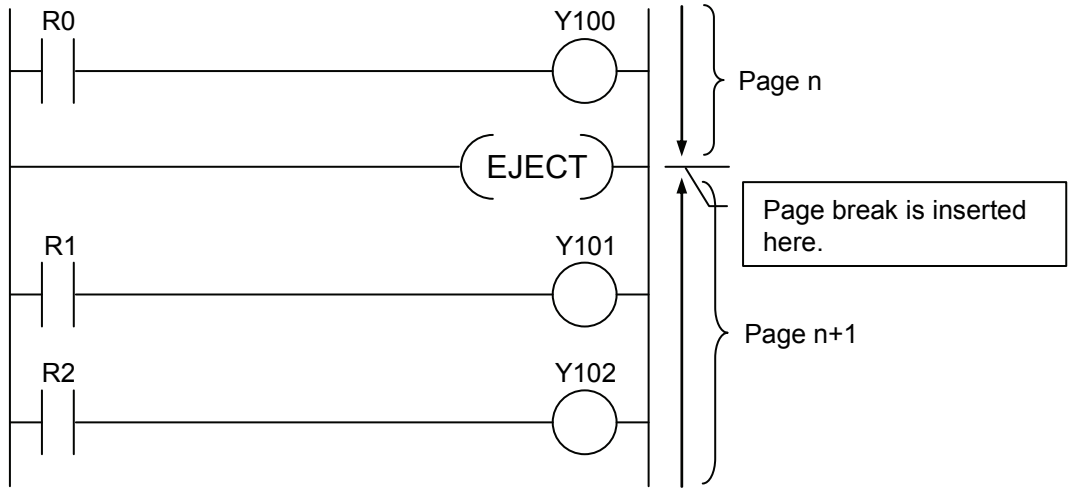
■ Ladder diagram



■ Description

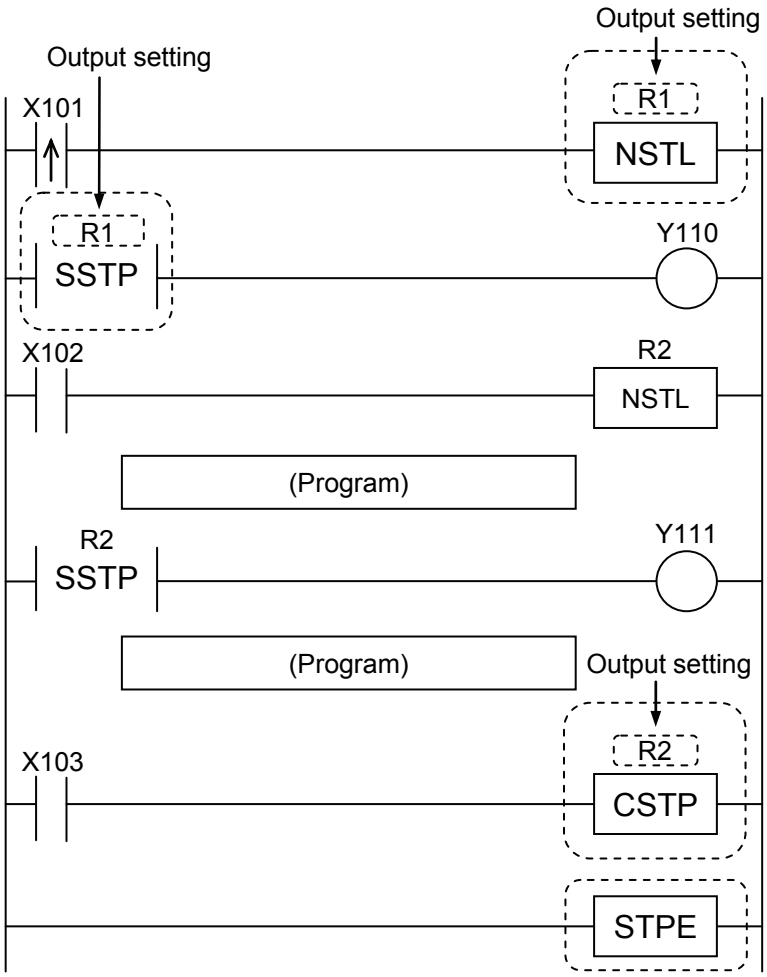
- When printing out a program code created with a software tool, page break will be added where this instruction is inserted.
- Similarly to the NOP instruction, no program processing will occur.

■ Example of Operation



SSTP (Start Step)
NSTL (Next Step)
CSTP (Clear Step)
STPE (Step End)

■ **Ladder diagram**



■ **Available Devices (●: Available)**

Operands	Bit device											Bit specification of word device		Index modifier
	X	Y	R	L	T	C	P	E	S	IN	OT	DT.b	LD.b	
bit	●	●	●	●								●	●	

■ Description

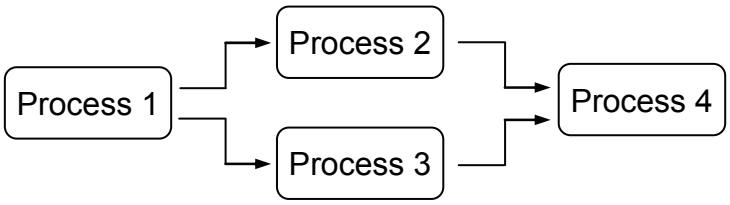
- The NSTL instruction starts and executes a process which begins with the SSTP instruction with the specified number.
- Program code between the SSTP instruction and the next SSTP instruction or the STPE instruction is handled as a single process.
- This allows for easy implementation of sequence control, selective branch control, as well as parallel branch/join control.

1) Sequence control

Sequentially switches and executes necessary processes as necessary.

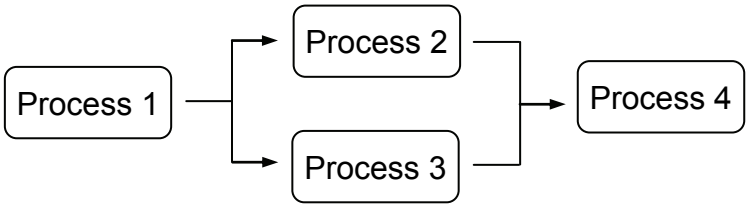
2) Selective branch control

Selects and executes processes as necessary.

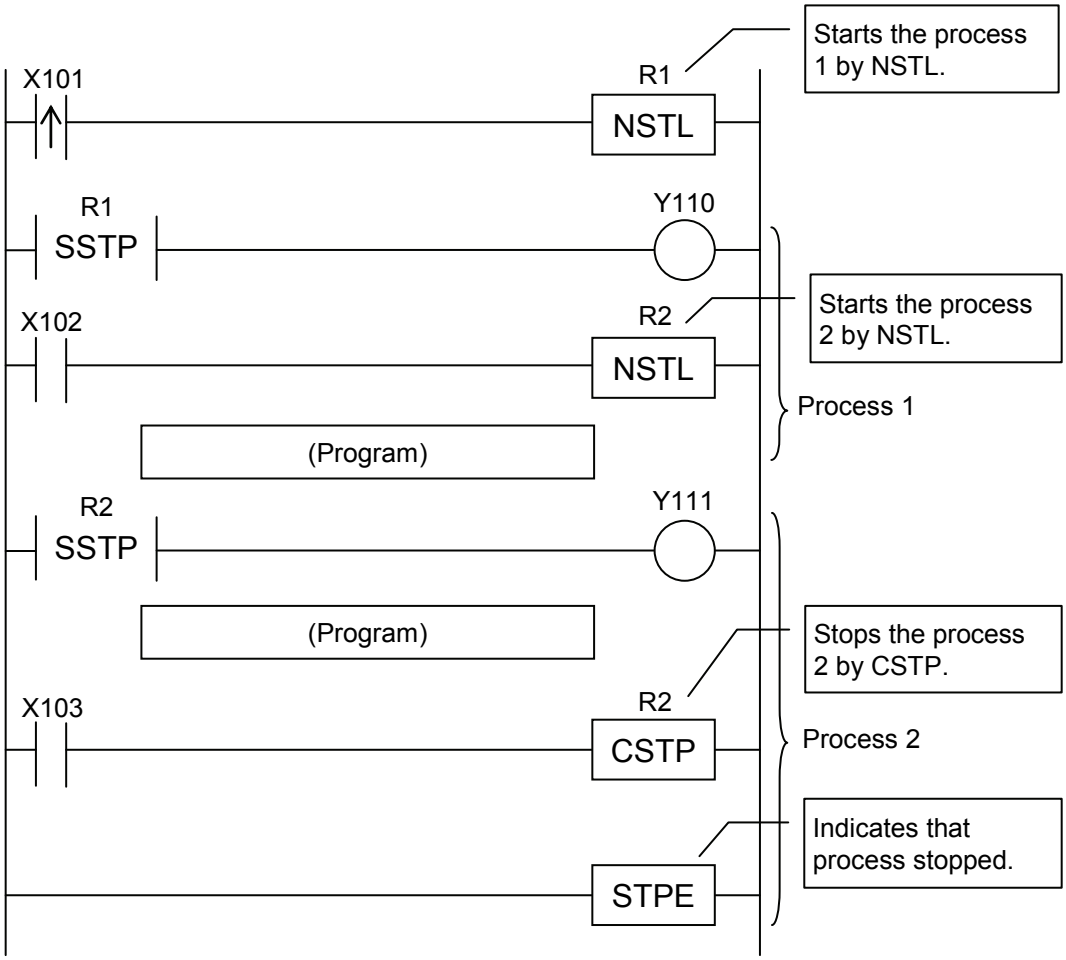


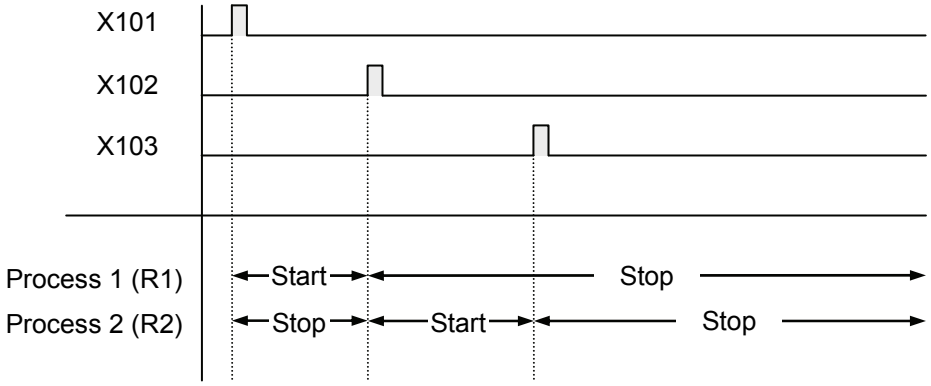
3) Parallel branch/join control

Executes multiple processes simultaneously. When all the processes running simultaneously complete, next processes can be executed.



■ Example of Operation





■ SSTP Start Step Instruction

- Indicates "Start of output number [n]". Be sure to write the "SSTP n" at the beginning of the program of the output number [n].
- Program code between "SSTP n" and the next "SSTP" or "STPE" is handled as the area with the output number [n].
- Processes with the same number cannot be defined.
- The OUT instruction can be connected directly from the bas bar immediately after the SSTP instruction.
- The SSTP instruction cannot be written in a subprogram (subroutine).
- Program code between the first SSTP instruction and STPE instruction is referred to as a "step ladder area" and is controlled as a process. Other areas are referred to as "normal ladder areas".
- There is a type of system relay which turns on only for a single scan when a process with a step ladder starts. (SR15: Step ladder initial pulse relay) This relay can be used to perform a process only for a single scan after starting a process (e.g., resetting a counter).

■ NSTL Next Step Instruction (Executed Every Scan Type)

- The NSTL R instruction starts a process specified by the relay number [R].
- The execution condition of the Next Step instruction is the start condition of the process.
- For the process that starts first, write the Next Step instruction in the normal ladder area.
- The process can be started from the normal ladder area as well as a running process.
- However, when a Next Step instruction which starts a process within another process is executed, the running process that includes this instruction will be automatically cleared, and the specified process will be started.
- Be careful when using the NSTL instruction with an instruction (1 to 6 below) which changes the order of instruction execution such as MC, MCE, JP or LBL, because the operation of the instruction may change depending on the instruction execution and input timing.

1) MC and MCE instructions
2) JP and LBL instructions
3) LOOP and LBL instructions
4) CNDE instructions
5) Step ladder instructions
6) Subroutine instructions

- When using a NTSL instruction with an AND stack (ANS) instruction or pop stack (POPS) instruction, be sure to write the code correctly.

■ CSTP Clear Step Instruction

- When the CSTPn instruction is executed, the process specified by "n" is cleared. Use this instruction to clear the final process or each process running in parallel in parallel branch/join control.
- The process can be cleared from the normal ladder area as well as a running process.

■ STPE Step End Instruction

- Indicates the end of the step ladder area. Be sure to write this instruction at the end of the final process. The final process is defined by the SSTP and STPE instructions.
- Only one STPE instruction can be written in each program. (It cannot be written in a subprogram such as a subroutine or interrupt program.)

■ Precautions during programming

- Processes need not be written in the order of their numbers.
- Note that the following instructions cannot be used in the step ladder area.

1) Jump instructions (JP and LBL)

2) Loop instructions (LOOP and LBL)

3) Master control instructions (MC and MCE)

4) Subroutine instructions (SBL and RET)

5) ED instruction

6) CNDE instruction

Note) The CALL instruction can be used in the step ladder area.

- In order to clear all processes at once, use the master control relay to program the code.
- Processes need not be started in the order of their numbers. Multiple processes can be started simultaneously.
- If an output of a process which is not started is forced to turn on or off, it will remain the forced state even when enforcement is released.

■ Step ladder operation

- The program in the normal ladder area and programs in processes started by the Next Step instruction (NSTL) are executed. Programs in processes not started are ignored.
- When a process starts and the first scan is in progress, the step initial pulse relay (SR15) turns on. It will turn off for the second and later scans. This can be used to reset a counter or shift register.
- The start and stop states of processes are stored in the following special data registers.

■ Precaution on clearing process

- When the Next Step instruction is executed within the program of a running process, the running process will be cleared. However, the process will be actually cleared during the next scan. Therefore, two processes may be running simultaneously for a single scan when transiting between them. If there are two outputs which must not be on at the same time, be sure to provide an interlock to prevent them from tuning on simultaneously. If these outputs can turn on simultaneously even when an interlock is provided by the program due to a delay in hardware response, take a counteraction on hardware to consider the delay.
- When a process is cleared, the instructions used in the process will operate as follows.
- Be careful when using an instruction which detects the leading edge of the execution condition and runs (1 - 6 below), including a differential instruction.

1) DF (leading edge differential) instruction

2) Count input for CT (counter) instruction

3) Count input for UDC (up-down counter) instruction

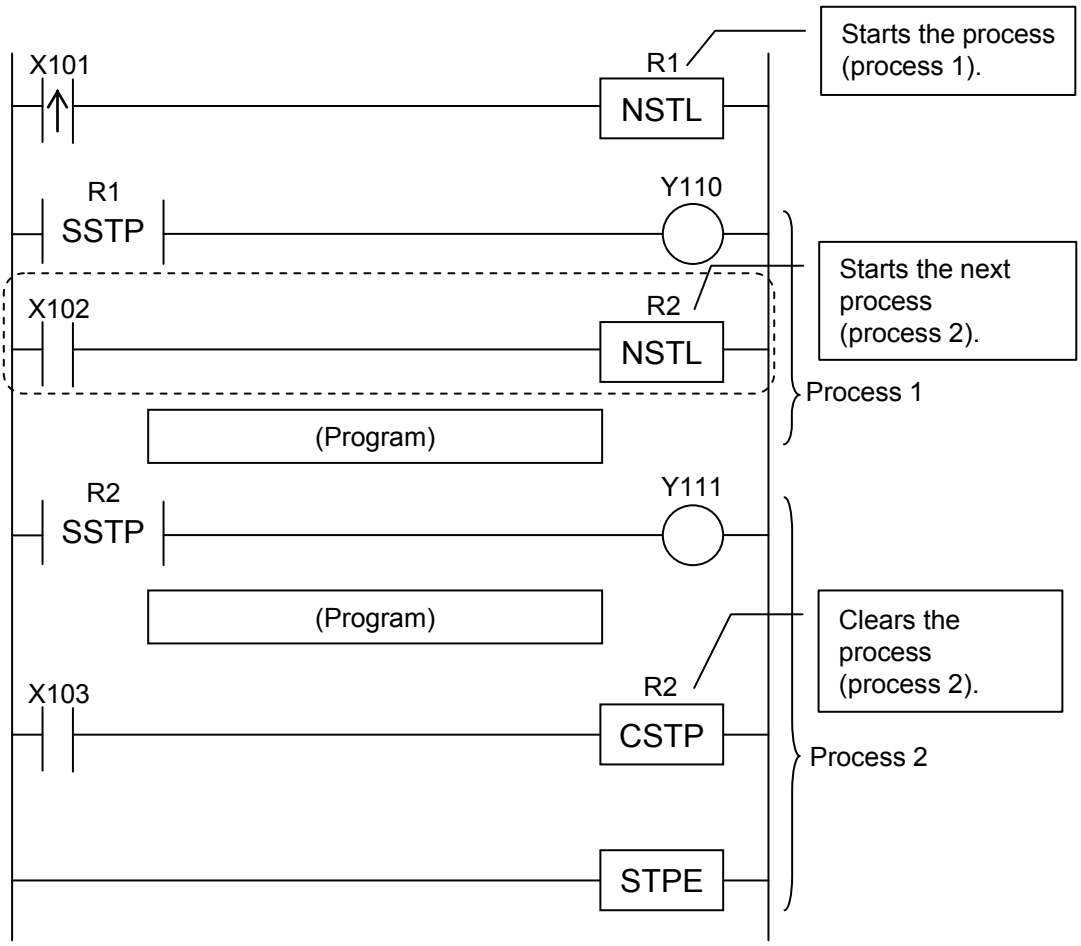
4) Shift input for SR (shift register) instruction

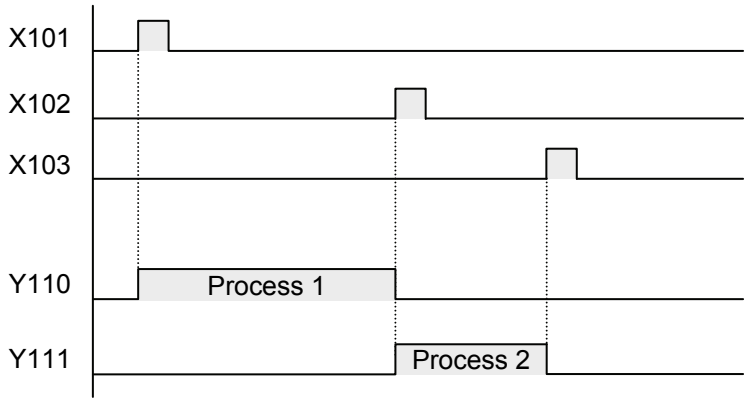
5) Shift input for LRSR (left and right shift register)

6) Differential execution type high-level instruction (instruction specified by p and instruction name)

■ Example (1) Sequential control of processes

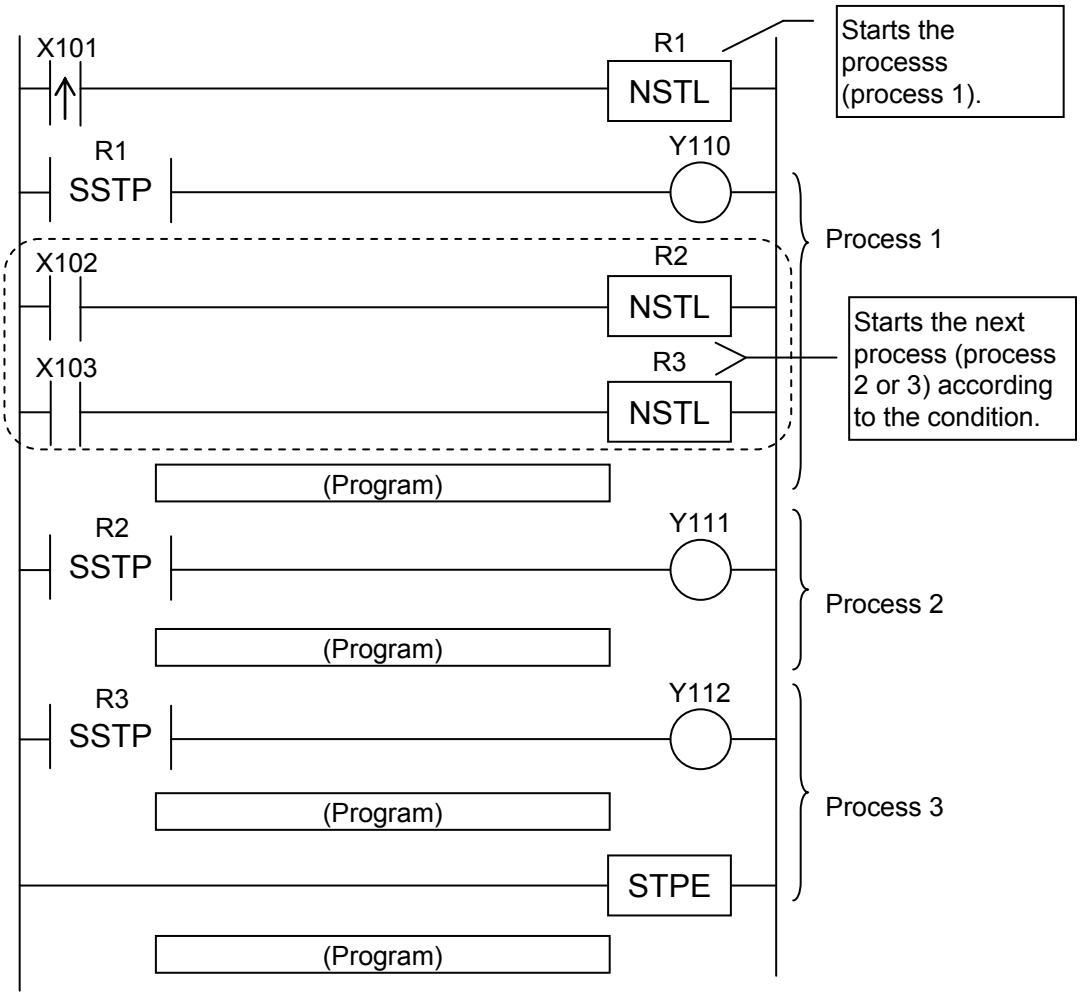
- This program repeats a process until its task completes, then moves to the next process.
- Write an instruction to start the next process within the current process. When the start instruction is executed, the next process is started and the current process is cleared.
- Processes need not be executed in the order of their numbers. It is even possible to start a previous process if necessary.

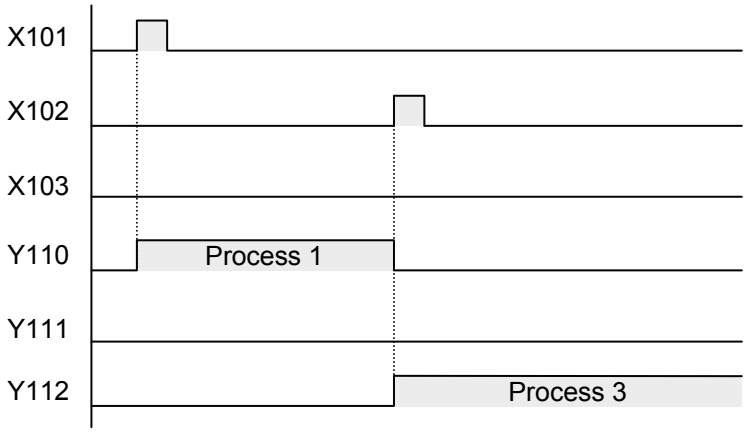




■ Example (2) Selective branch control of processes

- This program selects the next process to execute according to the task contents or result of the current process. Each process is repeated until its task completes.
- Write an instruction to start the next process within the current process. The program selects the next process and moves to it according to the execution condition.

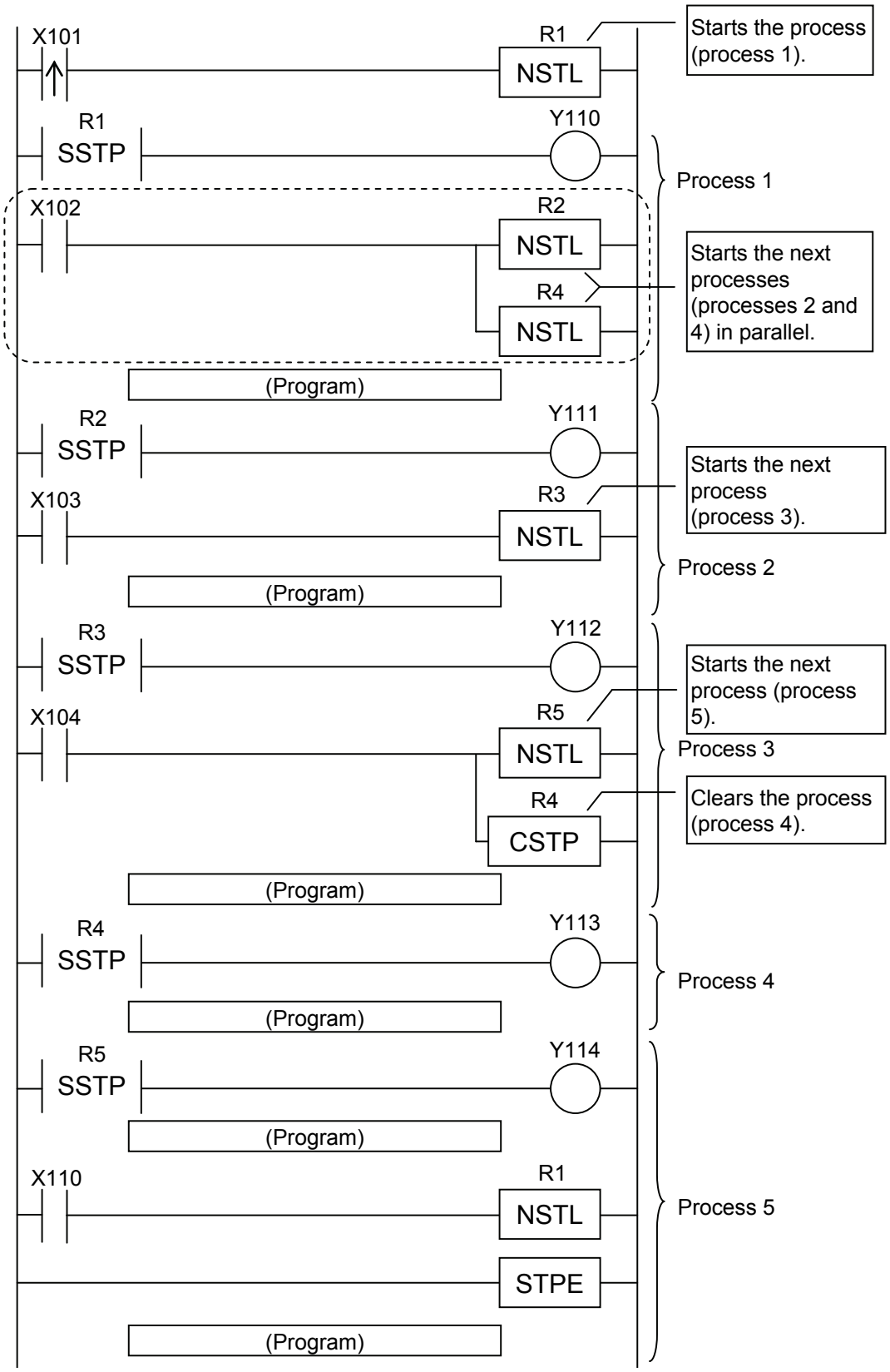


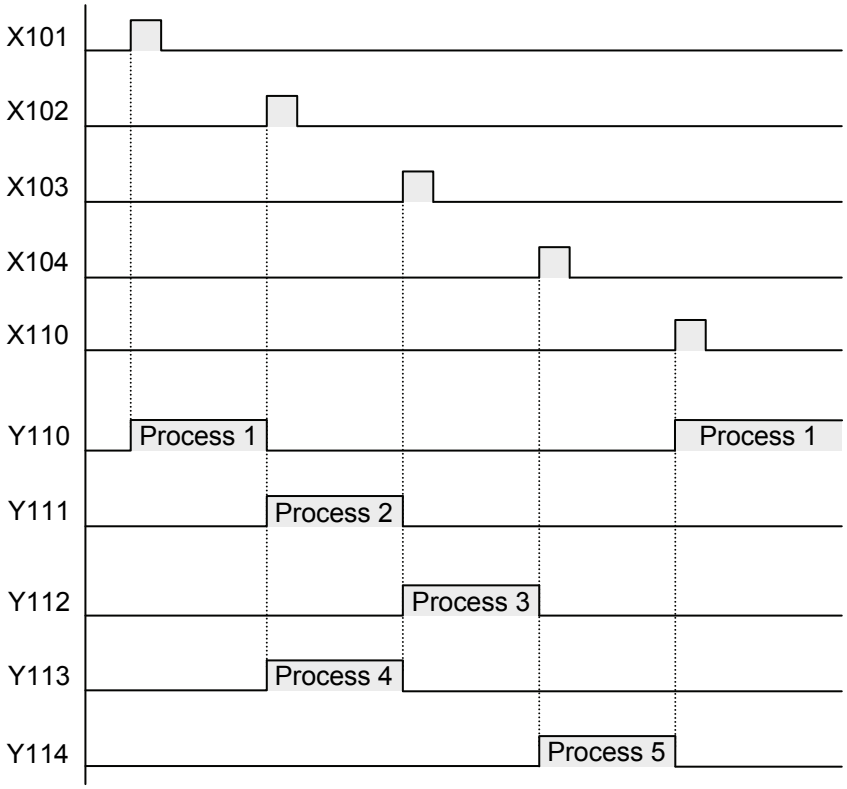


■ Example (3) Parallel branch/join control of processes

- This program starts multiple processes simultaneously. When all of the branched processes complete their tasks, they join together and move to the next process.
- Within a program in one process, write the instruction to move multiple processes in series for a single execution condition.

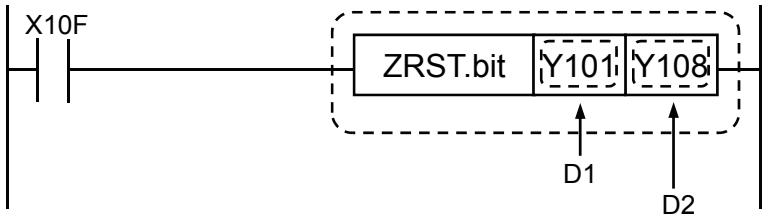
- To join the processes, include the flags indicating the states of other processes in the move condition to the next process. When joining processes and starting the next process, clear any process not cleared yet.




X101
X102
X103
X104
X110
Y110
Process 1
Process 1
Y111
Process 2
Y112
Process 3
Y113
Process 4
Y114
Process 5

ZRST (Block Clear)

■ Ladder diagram



■ Available operation unit (○: Available -: Not available)

Operation unit	bit	US	SS	UL	SL	SF	DF
i	●						

■ Operand list

Operands	Explanation
D1	Process clear start number
D2	Process clear end number

■ Available Devices (●: Available)

Operands	Bit device											Bit specification of word device		Index modifier
	X	Y	R	L	T	C	P	E	S	IN	OT	DT.b	LD.b	
D1	●	●	●	●							●	●	●	●
D2	●	●	●	●							●	●	●	●

■ Description

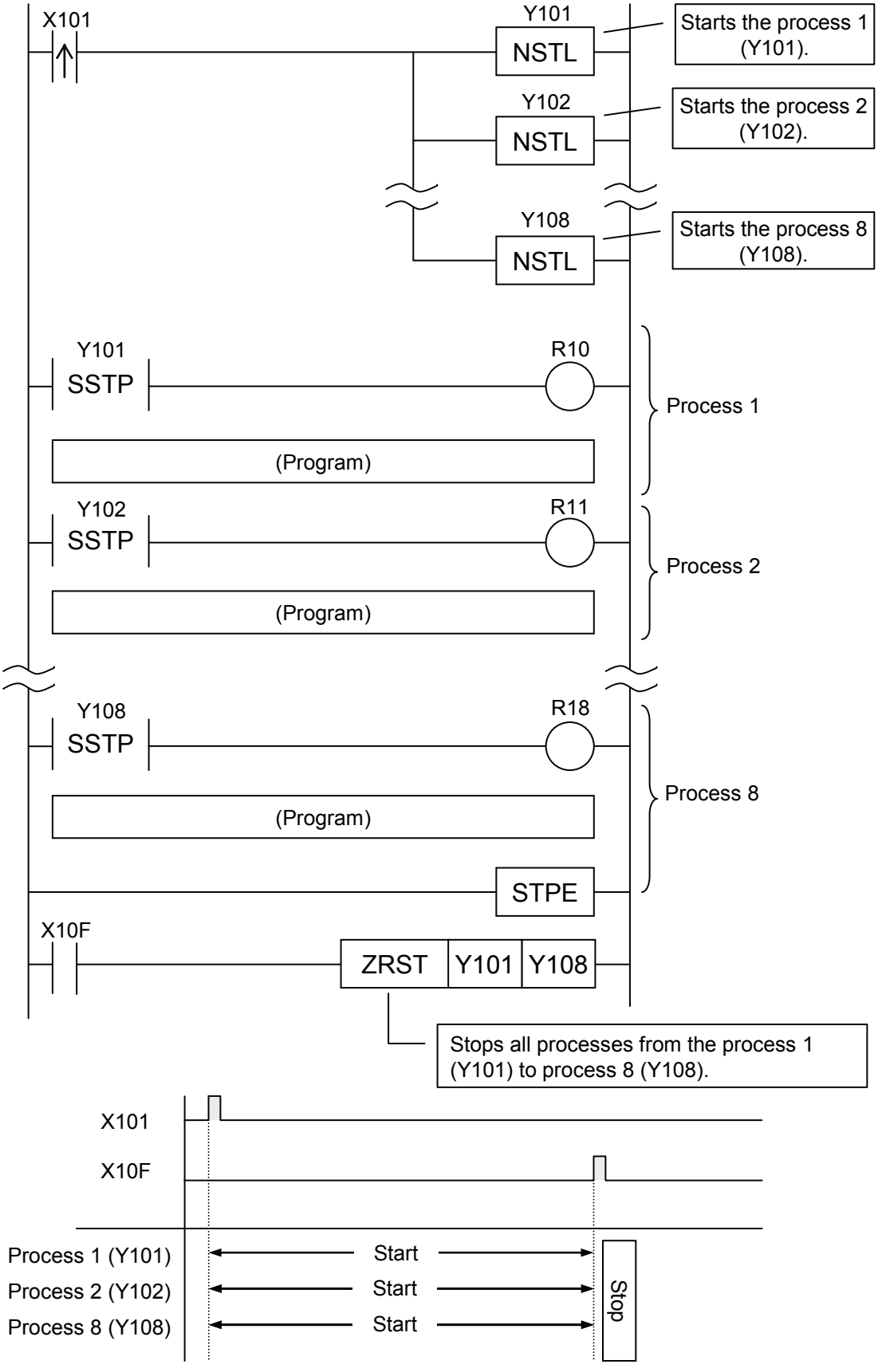
- When the ZRST instruction is executed, all the running processes within the range from the process [D1] and [D2] will be cleared.
- It can also be used to reset (clear to zero) the range from the area (bit address) specified in [D1] to the area (bit address) specified in [D2].

(Refer to "Chapter 3 High-Level Instruction ZRST" for the details.)

■ Precautions during programming

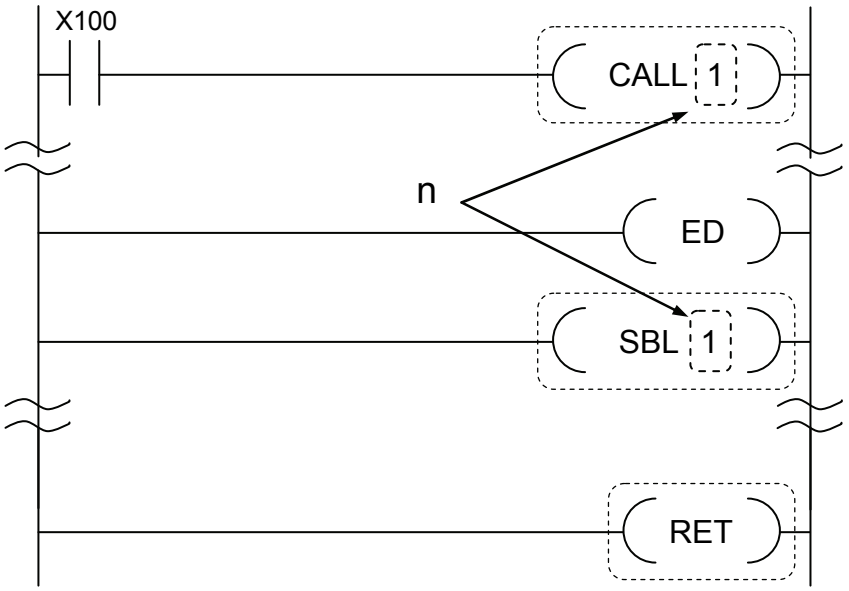
- Be sure that [D1] is smaller than [D2].
- This instruction can be executed from the normal ladder area as well as a running process.

■ Example of Operation



CALL (Subroutine Call)
SBL (Subroutine Label)
RET (Subroutine Return)

■ Ladder diagram



■ Operand list

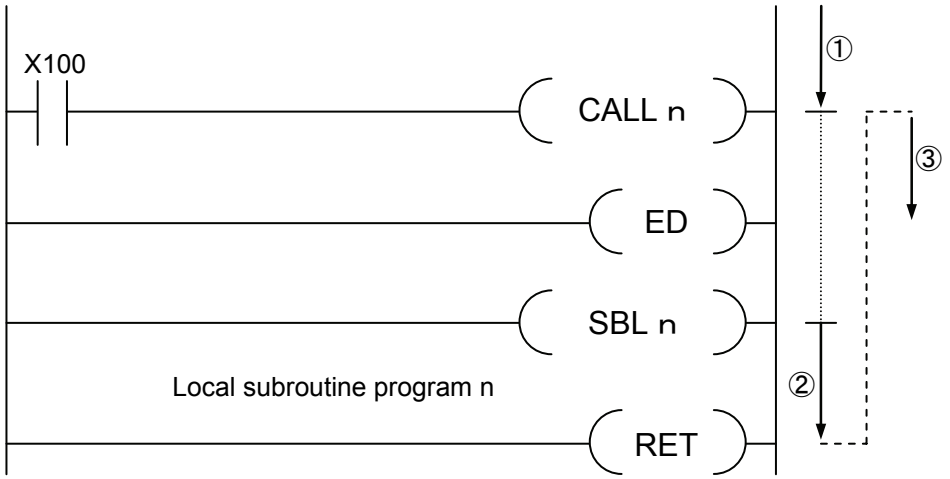
Operands	Explanation
n	Local subroutine program number within the same PB Available data specification range : 0 to 65535 (It is recommended to specify closely from 0.)

■ Available Devices (●: Available)

Operands	16-bit device											32-bit device			Integer			Real numbers		String	Index modifier
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX	K	U	H	SF	DF	" "	
n																●					

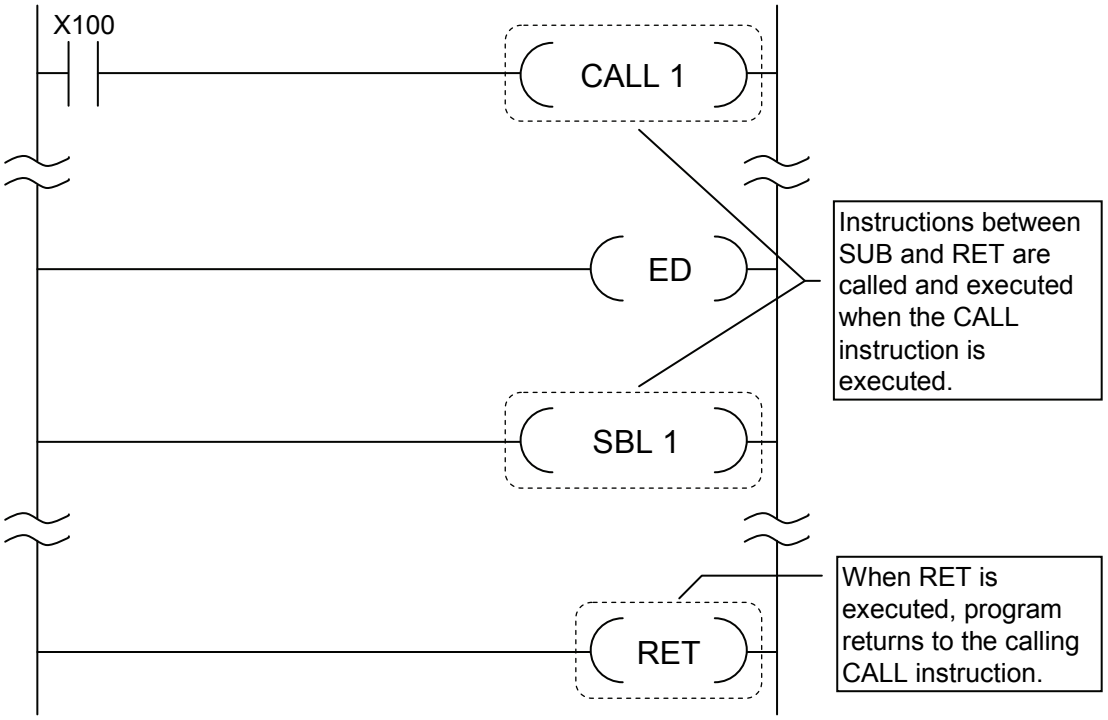
■ Description

- When the execution condition is on, the CALL instruction is executed to call the local subroutine program beginning with the SBL instruction with the specified number. When the execution condition is off, nothing occurs.
- When the subroutine program is processed up to the RET instruction, the program will return to the address next to the CALL instruction in the main program, and continue with processing of the main program.



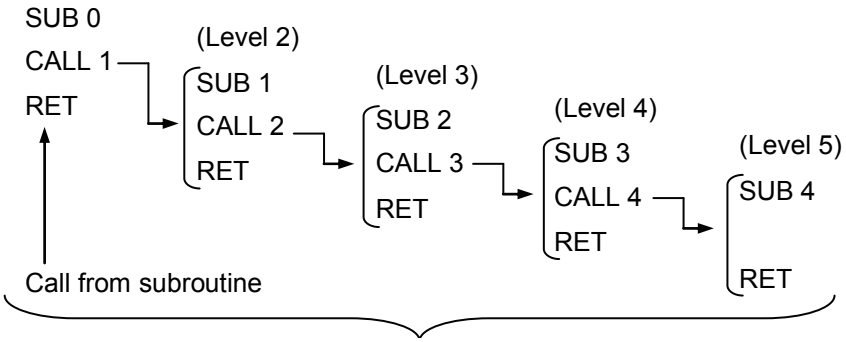
When "CALL n" is executed, ① to ③ will be executed in this order.

■ Example of Operation



■ Syntax of subroutine program

- "Subroutine program n" represents the program code between the SBL n instruction and the RET instruction. Be sure to write it to an address which follows the ED instruction.
- The CALL n instruction can be written in another subroutine program or step ladder, in addition to the main program. The CALL instruction with the same number can be written repeatedly.
- Subroutines can be nested in up to five levels.



Example of 5-level nesting

■ Precautions during programming

- Be careful when using an instruction which detects rising of the execution condition and runs (1 - 6 below) within a subroutine.

1) DF (leading edge differential) instruction
2) Count input for CT (counter) instruction
3) Count input for UDC (up-down counter) instruction
4) Shift input for SR (shift register) instruction
5) Shift input for LRSR (left and right shift register)
6) Differential execution type high-level instruction (instruction specified by p and instruction name)

- In the SBL instruction, specify the values of "n" closely from 0.

■ Operation when execution condition of CALL instruction is off

- When the execution condition of the call instruction turns off, the operation of the current subroutine stops (it is also true for calls from the master control or step ladder). In this case, the instructions used in the subroutine will operate as follows.

Type of instruction	Operation
OT	State is held
KP	
SET	
RST	
TM	Not counted. Note that time is not guaranteed if counting does not occur during a single scan.
CT	Intermediate result is held
SR	
Differential instruction	Operates in the same way as differential instructions used between MC and MCE Refer to Operation of differential instructions between MC and MCE.
Other instructions	Not executed

■ Flag conditions

Name	Explanation
SR7 SR8 (ER)	Turns on when the 16th subroutine executes the CALL instruction while subroutines are nested in 16 levels.

FCAL (Output Off Type Local Subroutine Call)

■ Ladder diagram

■ Operand list

Operands	Explanation
n	Local subroutine number (MC processing occurs only for a single scan when the execution condition is off.)

■ Available Devices (●: Available)

Operands	16-bit device											32-bit device			Integer			Real numbers		String	Index modifier
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX	K	U *1	H	SF	DF	" "	
n																●					

*1: Can be specified only when the operation unit is unsigned integer (US, UL).

■ Description

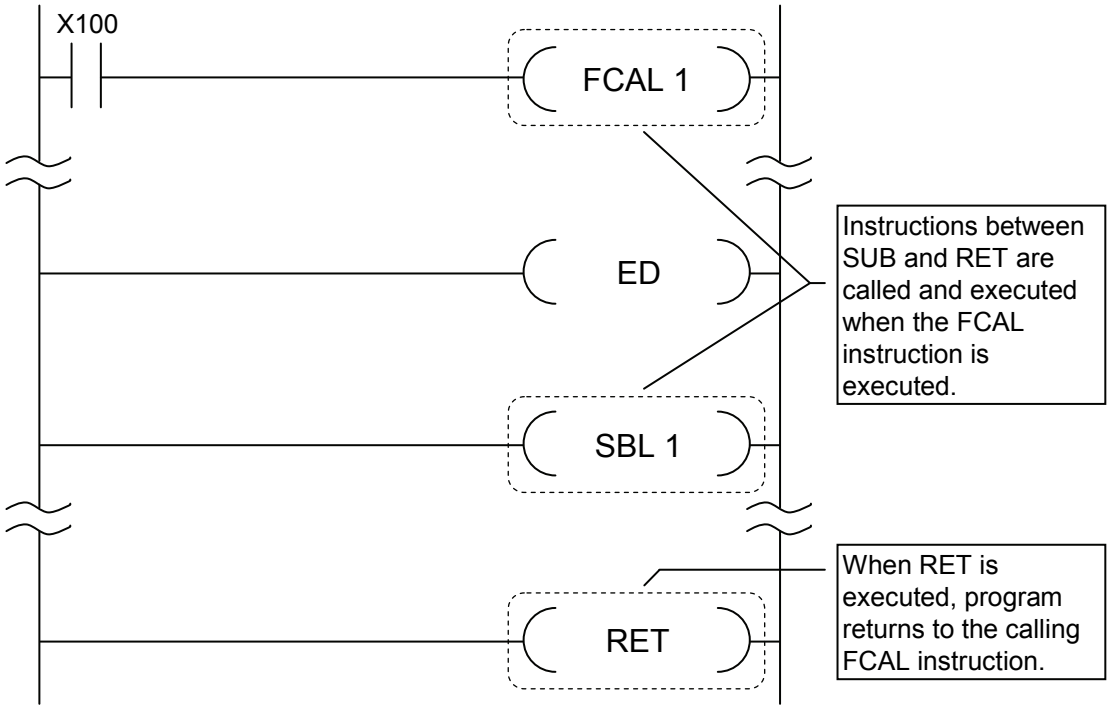
- The operation and syntax are the same as the normal local subroutine call instructions. However, there are the following exceptions.

■ Operation when execution condition of FCAL instruction is off

- When the execution condition of a subroutine turns off, the operation of the subroutine will not be executed and the instructions used in it will operate as follows:

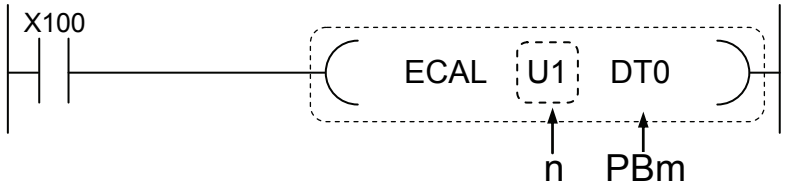
Type of instruction	Operation
OT	All off. Different operation from the CALL instruction.
KP	State is held
SET	
RST	
TM	Reset. Different operation from the CALL instruction.
CT	Intermediate result is held
SR	
Differential instruction	Operates in the same way as differential instructions used between MC and MCE Refer to Operation of differential instructions between MC and MCE.
Other instructions	Not executed

■ Example of Operation



ECAL (Subroutine Call (with PB No. Specification))

■ Ladder diagram



■ Operand list

Operands	Explanation
n	Subroutine number: 0 to 65535/1 PB
PBm	Target PB number: PB number where the subroutine specified by n is stored

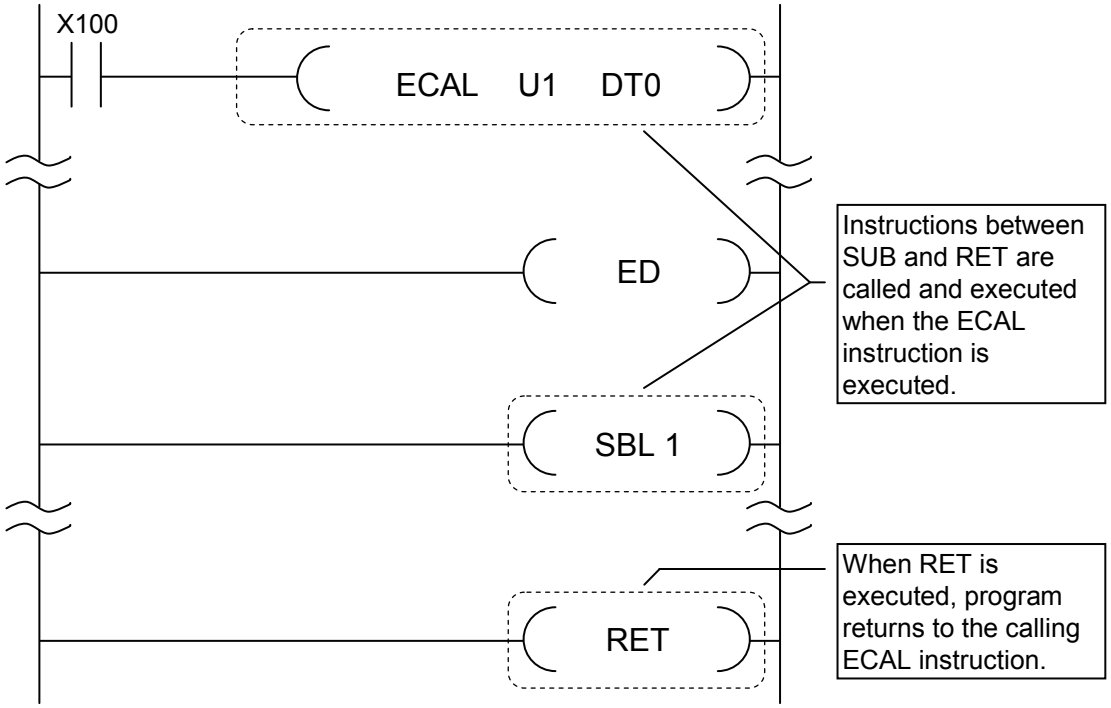
■ Available Devices (●: Available)

Operands	16-bit device											32 Bit device			Integer			Real numbers		String	Index modifier
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX	K	U	H	SF	DF	" "	
n																●					
PBm	●	●	●	●			●	●								●					

■ Description

- The operation and syntax are the same as the normal local subroutine call instructions.
- When the execution condition is on, the SBLn subroutine of PBm is called. (Branches to SBLn subroutine of PBm.)
- When the RET instruction is executed, control will return to the address after the ECAL instruction that called the subroutine.
- The local device used by the called subroutine is the local device of the called PBm.
- Be careful that executing many subroutine calls will increase the program code.
- The program just does not execute subroutines while their execution conditions are off. Outputs and execution states of the subroutines are all held.
- However, when a subroutine is called by the EFCAL forced call instruction, output within the subroutine will turn off when the execution condition turns off. This operation is similar to the master control operation.

■ Example of Operation

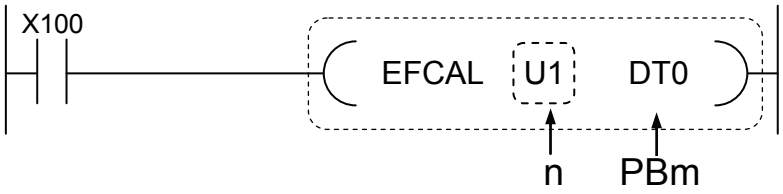


■ Precautions on use

- A subroutine can be called from within another subroutine.
- Up to 16 levels of nesting is possible when calling subroutines from within other subroutines. An operation error will occur when the nesting level becomes 17.

EFCAL (Forced Output Off Subroutine Call (with PB No. Specification))

■ Ladder diagram



■ Operand list

Operands	Explanation
n	Number: 0 to 65535/1 PB
PBm	Target PB number: PB number where the subroutine specified by n is stored

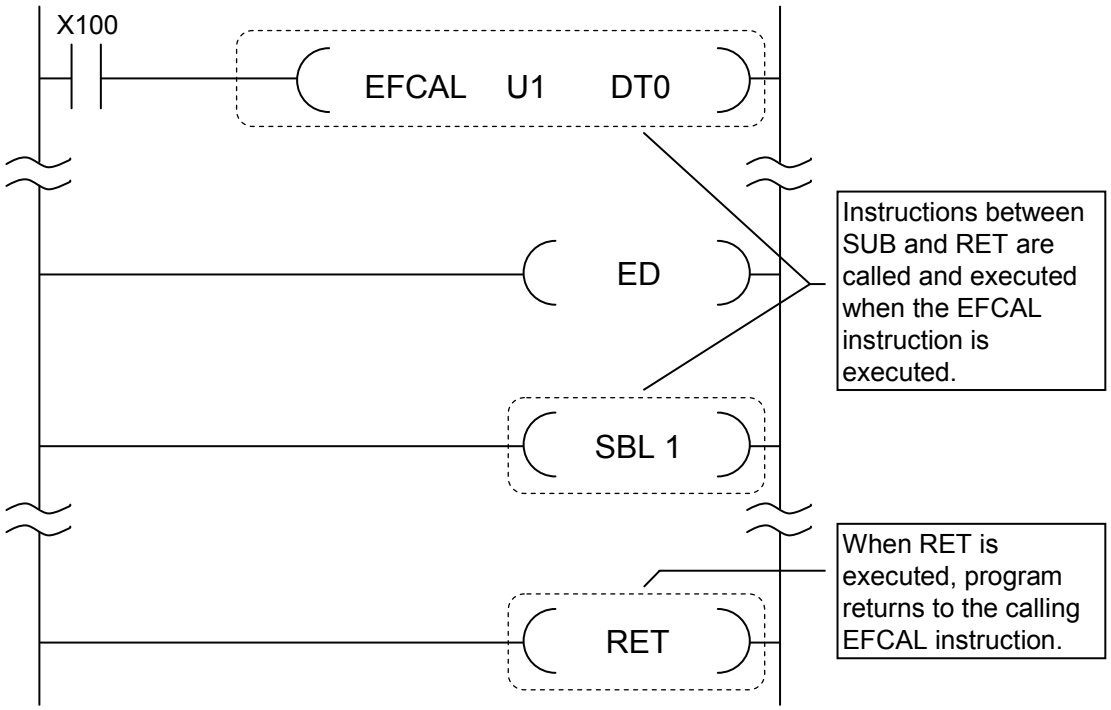
■ Available Devices (●: Available)

Operands	16-bit device											32-bit device			Integer			Real numbers		String	Index modifier
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX	K	U	H	SF	DF	" "	
n																●					
PBm	●	●	●	●			●	●								●					

■ Description

- The operation and syntax are the same as the normal local subroutine call instructions.
- When the execution condition is on, the SBLn subroutine of PBm is called. (Branches to SBLn subroutine of PBm.)
- When the RET instruction is executed, control will return to the address after the EFCAL instruction that called the subroutine.
- The local device used by the called subroutine is the local device of the called PBm.
- Be careful that executing many subroutine calls will increase the program code.
- When a subroutine is called by the EFCAL forced call instruction, output within the subroutine will turn off when the execution condition turns off. This operation is similar to the master control operation.

■ Example of Operation

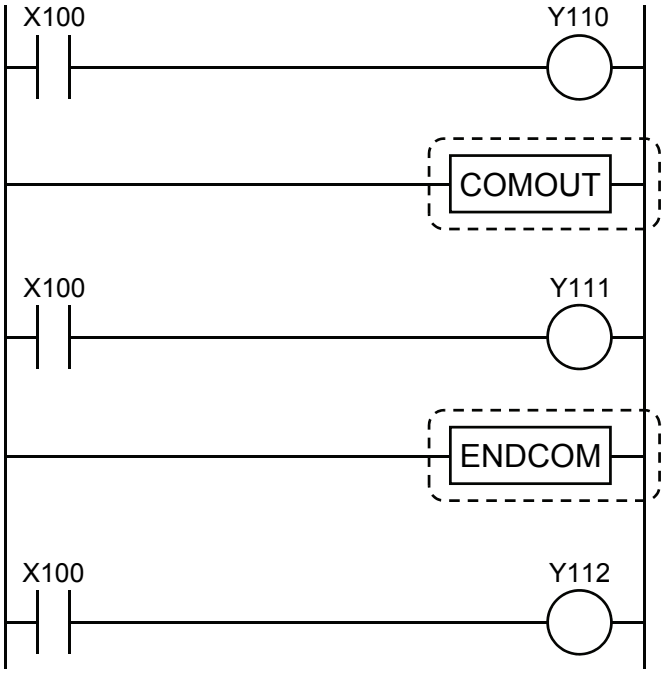


■ Precautions on use

- A subroutine can be called from within another subroutine.
- Up to 16 levels of nesting is possible when calling subroutines from within other subroutines. An operation error will occur when the nesting level becomes 17.

COMOUT (Comment Out)
ENDCOM (Comment Out End)

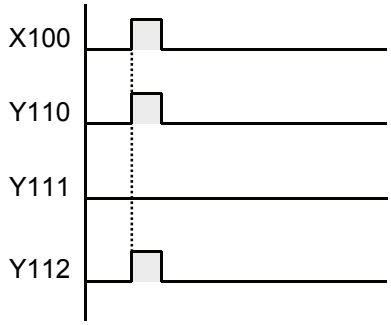
■ **Ladder diagram**



■ **Description**

- Comments out text between the COMOUT and ENDCOM instructions.

■ **Example of Operation**

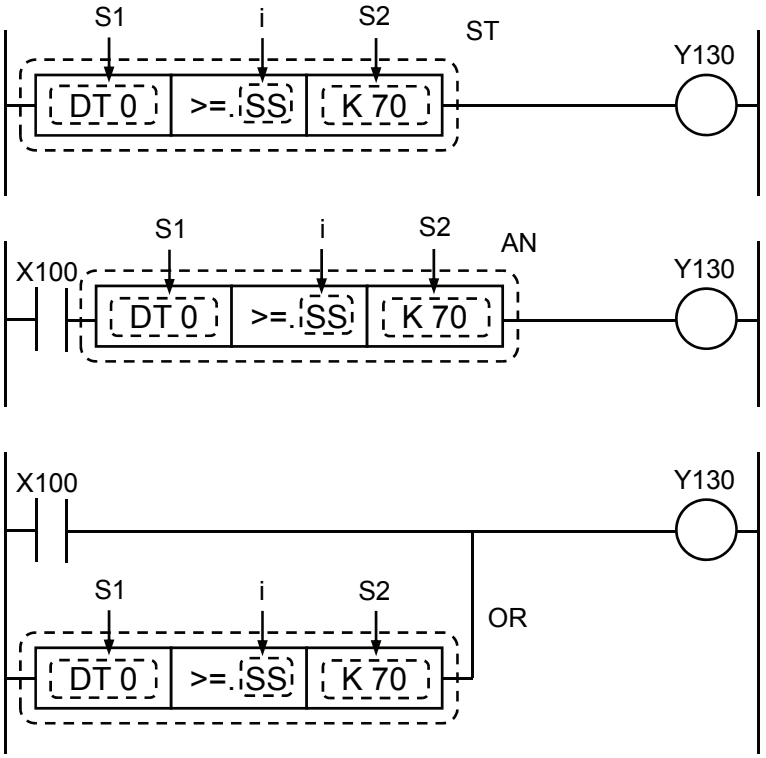


ST=, ST<>, ST>, ST>=, STPB No., ST<= (Data Comparison (Start))

AN=, AN<>, AN>, AN>=, AN<, AN<= (Data Comparison (AND))

OR=, OR<>, OR>, OR>=, OR<, OR<= (Data Comparison (OR))

■ **Ladder diagram**



■ **Available operation unit (○: Available -: Not available)**

Operation unit	bit	US	SS	UL	SL	SF	DF
i		●	●	●	●	●	●

■ **Operand list**

Operand	Explanation
S1	Comparison data 1
S2	Comparison data 2

■ Available Devices (●: Available)

Operands	16-bit device											32-bit device *1			Integer			Real numbers		String	Index modifier *2
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX *3	K *4	U *5	H *6	SF *7	DF *8	" "	
S1	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●
S2	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●

*1: Cannot be specified when the operation unit is 16-bit integer (SS, US).
*2: Only 16-bit device, 32-bit device, or integer constants can be modified (but not the real number or string constants).
*3: Index register (I0 to IE)
*4: Can be specified only when the operation unit is signed integer (SS, SL).
*5: Can be specified only when the operation unit is unsigned integer (US, UL).
*6: Can be specified only when the operation unit is integer (US, SS, UL, SL).
*7: Can be specified only when the operation unit is single-precision floating-point real number (SF).
*8: Can be specified only when the operation unit is double-precision floating-point real number (DF).

■ Description

Type of instruction	Operation
ST= ST<> ST> ST>= ST< ST<=	Compares signed data specified in [S1] with signed data specified in [S2]. Begins a logic operation as a contact electrically connected when the comparison result is the specified state (=, <, >, ・・・).
AN= AN> AN> AN>= AN< AN<=	Compares signed data specified in [S1] with signed data specified in [S2]. Connects in serial as a contact electrically connected when the comparison result is the specified state (=, <, >, ・・・).
OR= OR> OR< OR>= OR< OR<=	Compares signed data specified in [S1] with signed data specified in [S2]. Connects in parallel as a contact electrically connected when the comparison result is the specified state (=, <, >, ・・・).

■ Comparison result and operation

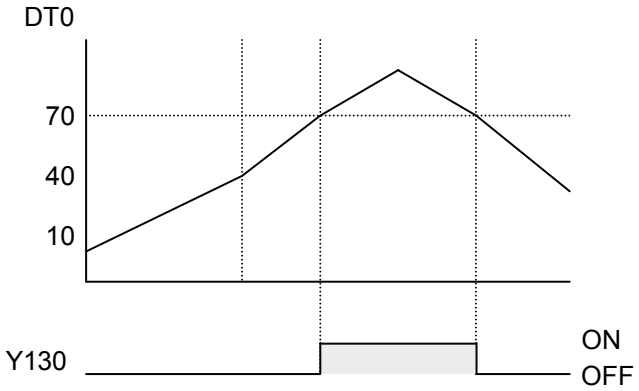
Comparison Instruction \ Relationship between [S1] and [S2]	[S1] < [S2]	[S1] = [S2]	[S1] > [S2]
ST=,AN,OR=	OFF	ON	OFF
ST<>,AN<>,OR<>	ON	OFF	ON
ST>,AN>,OR>	OFF	OFF	ON
ST>=,AN>=,OR>=	OFF	ON	ON
ST<>,AN<>,OR<>	ON	OFF	OFF
ST<=,AN<=,OR<=	ON	ON	OFF

Note) "< >" represents "≠".
">=" represents "≥".
"<=" represents "≤".

■ Example of Operation

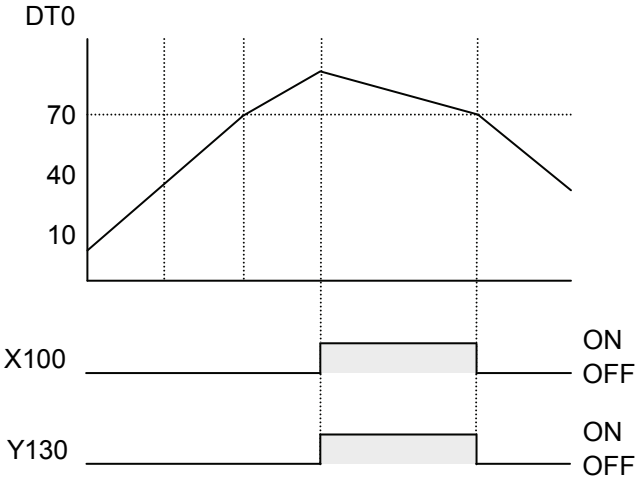
(1) Program operation for "ST >=" in ladder representation

Data register DT0 value and K70 are compared and if DT0 > K70, the external output Y130 turns on.



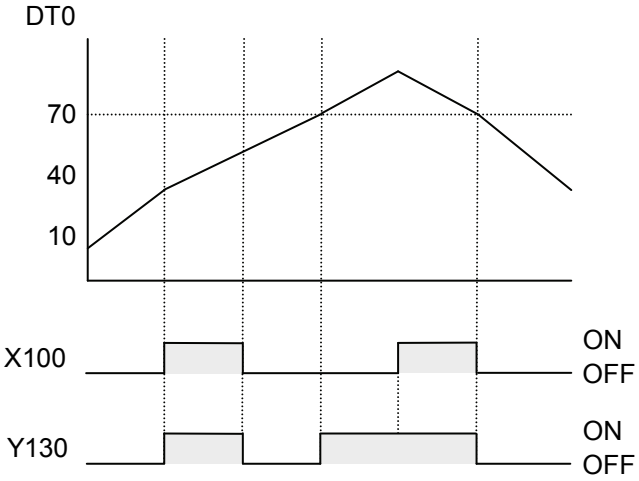
(2) Program operation for "AN >=" in ladder representation

If the external output X100 is on and the data register DT0 value and K70 are compared and if DT0 > K70, the external output Y130 turns on.



(3) Program operation for "OR >=" in ladder representation

If the external output X100 is on or the data register DT0 value and K70 are compared and if DT0 > K70, the external output Y130 turns on.



■ Precautions on use

- ST=, ST<>, ST>, ST>=, ST<, ST<=, OR=, OR<>, OR>, OR>=, OR<, and OR<=instructions are initiated from the bus bar.
- AN=, AN<>, AN>, AN>=, AN<, AN<=, OR=, OR<>, OR>, OR>=, OR<, and OR<=instructions can be used in series.
- Since BCD data is assumed to be a negative value during comparison if the highest bit is 1, the comparison result may become incorrect. In such a case, use the BIN instruction to convert the data into binary before comparison.

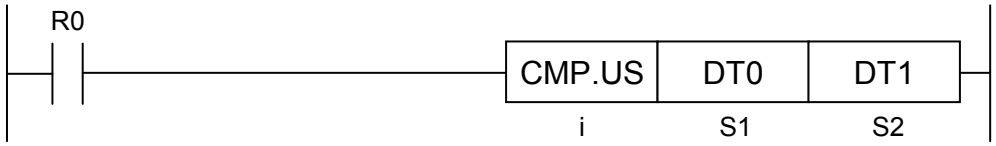
■ Flag conditions

Name	Explanation
SR7 SR8 (ER)	On if the specified address using the index modifier exceeds a limit.

3
High-level Instructions

CMP (Data Compare)

■ Ladder diagram



■ Available operation units (●: Available)

Operation unit	bit	US	SS	UL	SL	SF	DF
i		●	●	●	●	●	●

■ List of operands

Operand	Explanation
S1	Compared data 1 (device address or constant)
S2	Compared data 2 (device address or constant)

■ Available devices (●: Available)

Operand	16-bit device											32-bit device *1			Integers			Real numbers		String	Index modifier *2
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX *3	K *4	U *5	H *6	SF *7	DF *8	" "	
S1	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●		●
S2	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●		●

*1: Operation unit: 16-bit integers (SS, US) cannot be specified.
*2: Only 16-bit devices, 32-bit devices, and integer constants can be modified (real number constants, and character constants cannot be specified).
*3: Index register (I0 to IE)
*4: Only operation unit: signed integers (SS, SL) can be specified.
*5: Only operation unit: unsigned integers (US, UL) can be specified.
*6: Only operation unit: integers (US, SS, UL, SL) can be specified.
*7: Only operation unit: single-precision, floating-point real numbers (SF) can be specified.
*8: Only operation unit: double-precision, floating-point real numbers (DF) can be specified.

■ Outline of operation

- According to the operation unit [i], compare the data stored in the areas respectively headed by [S1] and [S2].
- The comparison result is output to the system relays SRA to SRC (assessment flags for the comparison instruction).

■ Process details

● Results of comparison flags (SR (system relays)) are as follows, based on relationship between [S1] and [S2].

Relationship between [S1] and [S2]	Comparison flags (SR (system relays))		
	SRA	SRB	SRC
	>	=	<
[S1] < [S2]	OFF	OFF	ON
[S1] = [S2]	OFF	ON	OFF
[S1] > [S2]	ON	OFF	OFF

Example 1) Operation unit: Unsigned 16 bits (US) (SRC(<) ON)

[i]…US

[S1]…DT0 [S2]…DT1

	Hexadecimal	Unsigned decimal	Signed decimal
DT0	H 6000	**K 24576**	K 24576
DT1	H 8500	**K 34048**	K -31488

Flag operations during execution

	SRA(>)	SRB(=)	SRC(<)
DT0 < DT1	OFF	OFF	ON

Example 2) Operation unit: Signed 16 bits (SS) (SRA(>) ON)

[i]…SS

[S1]…DT0 [S2]…DT1

	Hexadecimal	Unsigned decimal	Signed decimal
DT0	H 6000	K 24576	**K 24576**
DT1	H 8500	K 34048	**K -31488**

Flag operations during execution

	SRA(>)	SRB(=)	SRC(<)
DT0 > DT1	ON	OFF	OFF

Example 3) Operation unit: Unsigned 16 bits (US) (SRB(=) ON)

[i]…US

[S1]…DT0 [S2]…DT1

	Hexadecimal	Unsigned decimal	Signed decimal
DT0	H 1234	**K 4660**	K 4660
DT1	H 1234	**K 4660**	K 4660

Flag operations during execution

	SRA(>)	SRB(=)	SRC(<)
DT0 = DT1	OFF	ON	OFF

Example 4) Operation unit: Signed 32 bits (SL) (SRC(<) ON)

[i]…SL
[S1]…I0 [S2]…TS0

	Hexadecimal	Unsigned decimal	Signed decimal
I0	H 85000000	K 2231369728	**K -2063597568**
TS0	H 60000000	K 1610612736	**K 1610612736**

Flag operations during execution

	SRA(>)	SRB(=)	SRC(<)
I0 < TS0	OFF	OFF	ON

Example 5) Operation unit: Single-precision, floating-point real number (SF) (SRA(>) ON)

[i]…SF
[S1]…DT0 [S2]…LD0

	Value (real number decimal)
DT0・DT1	**SF 1.234E + 00**
LD0・LD1	**SF -1.234E + 00**

Flag operations during execution

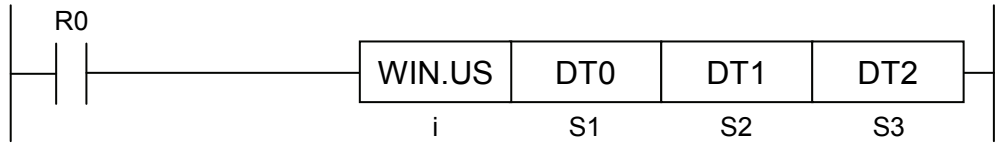
	SRA(>)	SRB(=)	SRC(<)
DT0・DT1 > LD0・LD1	ON	OFF	OFF

■ Flag operations

Name	Explanation
SR7 SR8 (ER)	To be set in the case of out-of-range in indirect access (index modification).
	To be set when a non-real number is specified for [S1] or [S2], and the operation unit is a real number (SF).
SRA (>)	Varies depending on the comparison result.
SRB (=)	
SRC (<)	

WIN (Band Compare)

■ Ladder diagram



■ Available operation units (●: Available)

Operation unit	bit	US	SS	UL	SL	SF	DF
i		●	●	●	●	●	●

■ List of operands

Operand	Explanation
S1	Compared data (device address or constant)
S2	Lower limit (device address or constant)
S3	Upper limit (device address or constant)

■ Available devices (●: Available)

Operand	16-bit device											32-bit device *1			Integers			Real numbers		String	Index modifier *2
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX *3	K *4	U *5	H *6	SF *7	DF *8	" "	
S1	●	●	●	●			●	●	●	●	●	●	●	●	●	●	●	●	●		●
S2	●	●	●	●			●	●	●	●	●	●	●	●	●	●	●	●	●		●
S3	●	●	●	●			●	●	●	●	●	●	●	●	●	●	●	●	●		●

*1: Operation unit: 16-bit integers (SS, US) cannot be specified.
*2: Only 16-bit devices, 32-bit devices, and integer constants can be modified (real number constants, and character constants cannot be specified).
*3: Index register (I0 to IE)
*4: Only operation unit: signed integers (SS, SL) can be specified.
*5: Only operation unit: unsigned integers (US, UL) can be specified.
*6: Only operation unit: integers (US, SS, UL, SL) can be specified.
*7: Only operation unit: single-precision, floating-point real numbers (SF) can be specified.
*8: Only operation unit: double-precision, floating-point real numbers (DF) can be specified.

■ Outline of operation

- According to the operation unit [i], compare [S1] (compared data) with a range delimited with [S2] (lower limit) and [S3] (upper limit).
- The comparison result is output to the system relays SRA to SRC (assessment flags for the comparison instruction).

■ Process details

● Results of the flags (SR (system relays)) are as follows, based on relationship between [S1] and [S2] and [S3].

Relationship between [S1] and [S2] and [S3]	Comparison flags (SR (system relays))		
	SRA	SRB	SRC
	>	=	<
[S1] < [S2]	OFF	OFF	ON
[S2] ≦ [S1] ≦ [S3]	OFF	ON	OFF
[S1] > [S3]	ON	OFF	OFF

Example 1) Operation unit: Unsigned 16 bits (US) (SRB(=) ON)

[i]…US
[S1]…DT0 [S2]…DT1 [S3]…DT2

	Hexadecimal	Unsigned decimal	Signed decimal
DT0	H 8000	**K 32768**	K -32767
DT1	H 7000	**K 28672**	K 28672
DT2	H 9000	**K 36864**	K -28671

Flag operations during execution

	SRA(>)	SRB(=)	SRC(<)
DT1 < DT0 < DT2	OFF	ON	OFF

Example 2) Operation unit: Unsigned 16 bits (SRC(<) ON)

[i]…US
[S1]…DT0 [S2]…DT1 [S3]…DT2

	Hexadecimal	Unsigned decimal	Signed decimal
DT0	H 6000	**K 24576**	K 24576
DT1	H 7000	**K 28672**	K 28672
DT2	H 9000	**K 36864**	K -28671

Flag operations during execution

	SRA(>)	SRB(=)	SRC(<)
DT0 < DT1	OFF	OFF	ON

Example 3) Operation unit: Signed 16 bits (SS) (operation error)

[i]…SS
[S1]...DT0 [S2]…DT1 [S3]…DT2

	Hexadecimal	Unsigned decimal	Signed decimal
DT0	H 6000	K 32768	**K -32767**
DT1	H 7000	K 28672	**K 28672**
DT2	H 9000	K 36864	**K -28671**

Flag operations during execution

	SRA(>)	SRB(=)	SRC(<)
DT1 > LD2	OFF	OFF	OFF

* Operation error because of [S2] > [S3] (Set SR7 (latest error) and SR8 (hold error))

Example 4) Operation unit: Single-precision, floating-point real number (SF) (SRA(>) ON)

[i]…SF
[S1]…DT0 [S2]…LD0 [S3]…LD2

	Value (real number decimal)
DT0・DT1	SF 8.000E + 02
LD0・LD1	SF 5.000E + 02
LD2・LD3	SF 7.000E + 02

Flag operations during execution

	SRA(>)	SRB(=)	SRC(<)
DT0・DT1 > LD2・LD3	ON	OFF	OFF

■ Precautions during programming

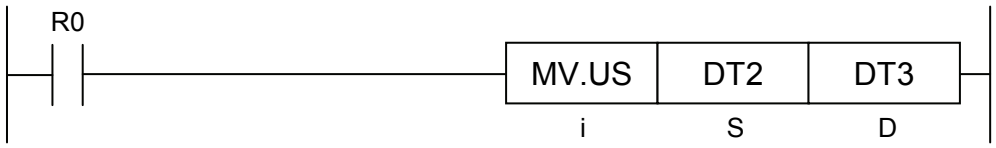
- In the case of a direct address and an index modification address, ensure that [S3] ≥ [S2].

■ Flag operations

Name	Explanation
SR7 SR8 (ER)	To be set in the case of out-of-range in indirect access (index modification).
	To be set when [S2] > [S3].
	To be set when a non-real number is specified for [S1], [S2] or [S3], and the operation unit is a real number (SF).
SRA (>)	Varies depending on the comparison result.
SRB(＝)	
SRC (<)	

MV (Data Transfer)

■ Ladder diagram



■ Available operation units (●: Available)

Operation unit	bit	US	SS	UL	SL	SF	DF
i		●	●	●	●	●	●

■ List of operands

Operand	Explanation
S	Source device address or constant
D	Destination device address

■ Available devices (●: Available)

Operand	16-bit device											32-bit device *1			Integers			Real numbers		String	Index modifier *2
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX *3	K *4	U *5	H *6	SF *7	DF *8	" "	
S	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●		●
D	●	●	●	●			●	●	●		●	●	●	●							●

*1: Operation unit: 16-bit integers (SS, US) cannot be specified.
*2: Only 16-bit devices, 32-bit devices, and integer constants can be modified (real number constants, and character constants cannot be specified).
*3: Index register (I0 to IE)
*4: Only operation unit: signed integers (SS, SL) can be specified.
*5: Only operation unit: unsigned integers (US, UL) can be specified.
*6: Only operation unit: integers (US, SS, UL, SL) can be specified.
*7: Only operation unit: single-precision, floating-point real numbers (SF) can be specified.
*8: Only operation unit: single-precision, floating-point real numbers (DF) can be specified.

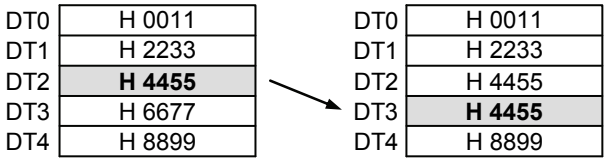
■ Outline of operation

- Transfer the operation unit data specified by [i] from the device address or constant specified by [S] to the device address specified by [D].

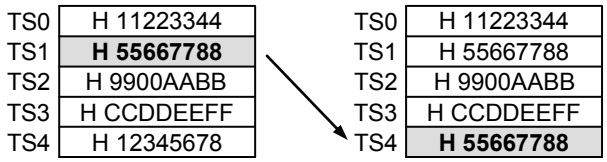
[S] → [D]

■ Process details

Example 1) Operation unit: 16 bits (US, SS)
[i]…US,SS
[S] …DT2 [D] …DT3



Example 2) Operation unit: 32 bits (UL, SL, SF)
[i]…UL,SL,SF
[S] …TS1 [D] …TS4

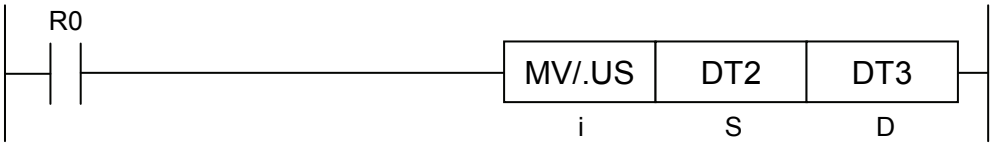


■ Flag operations

Name	Explanation
SR7 SR8 (ER)	To be set in the case of out-of-range in indirect access (index modification).

MV/ (Inversion and Transfer)

■ Ladder diagram



■ Available operation units (●: Available)

Operation unit	bit	US	SS	UL	SL	SF	DF
i		●	●	●	●	●	●

■ List of operands

Operand	Explanation
S	Source device address or constant
D	Destination device address

■ Available devices (●: Available)

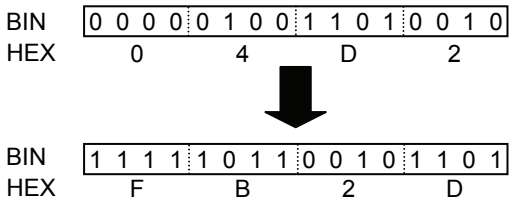
Operand	16-bit device											32-bit device *1			Integers			Real numbers		String	Index modifier *2
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX *3	K *4	U *5	H *6	SF *7	DF *8	" "	
S	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●		●
D	●	●	●	●			●	●	●		●	●	●	●							●

*1: Operation unit: 16-bit integers (SS, US) cannot be specified.
*2: Only 16-bit devices, 32-bit devices, and integer constants can be modified (real number constants, and character constants cannot be specified).
*3: Index register (I0 to IE)
*4: Only operation unit: signed integers (SS, SL) can be specified.
*5: Only operation unit: unsigned integers (US, UL) can be specified.
*6: Only operation unit: integers (US, SS, UL, SL) can be specified.
*7: Only operation unit: single-precision, floating-point real numbers (SF) can be specified.
*8: Only operation unit: double-precision, floating-point real numbers (DF) can be specified.

■ Outline of operation

- Logically invert and transfer the specified operation unit data [i] from the device address or constant specified by [S] to the device address specified by [D].

/[S] → [D]

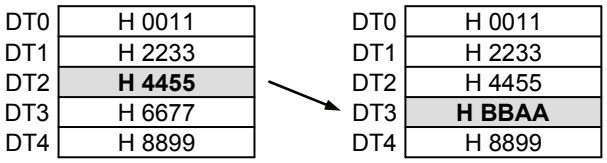


■ Process details

Example 1) Operation unit: 16 bits (US, SS)

[i]…US,SS

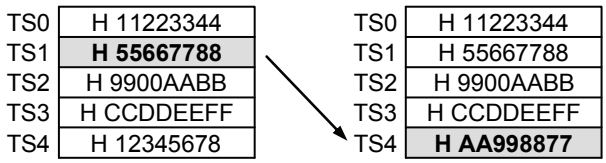
[S] …DT2 [D] …DT3



Example 2) Operation unit: 32 bits (UL, SL, SF)

[i]…UL,SL,SF

[S] …TS1 [D] …TS4

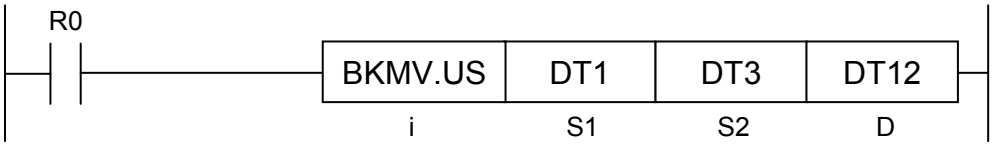


■ Flag operations

Name	Explanation
SR7 SR8 (ER)	To be set in the case of out-of-range in indirect access (index modification).

BKMV (Block Transfer)

■ Ladder diagram



■ Available operation units (●: Available)

Operation unit	bit	US	SS	UL	SL	SF	DF
i		●	●	●	●	●	●

■ List of operands

Operand	Explanation
S1	Starting device address of the source data
S2	End device address of the source data
D	Starting device address of the data destination

■ Available devices (●: Available)

Operand	16-bit device											32-bit device *1			Integers			Real numbers		String	Index modifier *2
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX *3	K	U	H	SF	DF	" "	
S1	●	●	●	●	●	●	●	●	●	●	●	●	●	●							●
S2	●	●	●	●	●	●	●	●	●	●	●	●	●	●							●
D	●	●	●	●			●	●	●		●	●	●	●							●

*1: Operation unit: 16-bit integers (SS, US) cannot be specified.
*2: Only 16-bit devices, and 32-bit devices can be modified (integer constants, real number constants, and character constants cannot be specified).
*3: Index register (I0 to IE)

■ Outline of operation

- Transfer, in a block, the data in the areas specified by [S1] and [S2] to the area starting with [D] onward.

■ Process details

Example 1) Operation unit: 16 bits (US, SS)
[i]…US,SS
[S1]…DT1 [S2]…DT3 [D]…DT12

DT0	H 0011		DT10	H 0011
DT1	**H 2233**		DT11	H 2233
DT2	**H 4455**	→	DT12	**H 2233**
DT3	**H 6677**	→	DT13	**H 4455**
DT4	H 8899	→	DT14	**H 6677**

Example 2) Operation unit: 32 bits (UL, SL, SF)
[i]…UL,SL,SF
[S1]…TS1 [S2]…TS4 [D]…DT10

TS0	H 11223344		DT10・DT11	**H 55667788**
TS1	**H 55667788**	→	DT12・DT13	**H 9900AABB**
TS2	**H 9900AABB**	→	DT14・DT15	**H CCDDEEFF**
TS3	**H CCDDEEFF**	→	DT16・DT17	**H 12345678**
TS4	**H 12345678**	→	DT18・DT19	H 24680ACE

■ Precautions during programming

- In the case of a direct address and an index modification address, specify the same type of device for [S1] and [S2]. At the same time, specify [S2] ≥ [S1].
- The data are transferred by operation unit, ending with the device containing [S2].

Example 1) Device address of [S2] comes to a higher word (operation unit = 32 bits)

[S1]…DT2 [S2]…DT6 [D]…DT12

DT0・DT1	H 11223344		DT10・DT11	H 11111111
DT2・DT3	**H 55667788**	→	DT12・DT13	**H 55667788**
DT4・DT5	**H 9900AABB**	→	DT14・DT15	**H 9900AABB**
DT6・DT7	**H CCDDEEFF**	→	DT16・DT17	**H CCDDEEFF**
DT8・DT9	H 12345678		DT18・DT19	H 55555555

Example 2) Device address of [S2] comes to a lower word (operation unit = 32 bits)

[S1]…DT2 [S2]…DT7 [D]…DT12

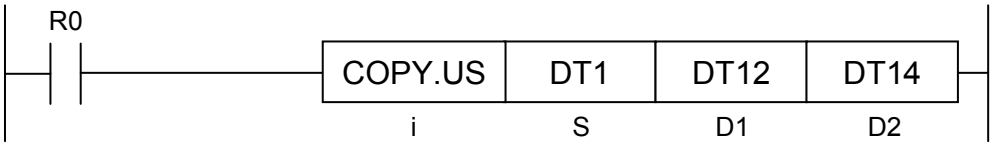
DT0・DT1	H 11223344		DT10・DT11	H 11111111
DT2・DT3	**H 55667788**	→	DT12・DT13	**H 55667788**
DT4・DT5	**H 9900AABB**	→	DT14・DT15	**H 9900AABB**
DT6・DT7	**H CCDDEEFF**	→	DT16・DT17	**H CCDDEEFF**
DT8・DT9	H 12345678		DT18・DT19	H 55555555

■ Flag operations

Name	Explanation
SR7	To be set in the case of out-of-range in indirect access (index modification).
SR8	To be set when [S1] > [S2].
(ER)	To be set when the destination range is out of the available range.

COPY (Block Copy)

■ Ladder diagram



■ Available operation units (●: Available)

Operation unit	bit	US	SS	UL	SL	SF	DF
i		●	●	●	●	●	●

■ List of operands

Operand	Explanation
S	Device address or constant of the source data
D1	Starting device address of the destination area
D2	End device address of the destination area

■ Available devices (●: Available)

Operand	16-bit device											32-bit device			Integers			Real numbers		String	Index modifier
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX *3	K *4	U *5	H *6	SF *7	DF *8	" "	
S	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●		●
D1	●	●	●	●			●	●	●		●	●	●	●							●
D2	●	●	●	●			●	●	●		●	●	●	●							●

*1: Operation unit: 16-bit integers (SS, US) cannot be specified.
*2: Only 16-bit devices, 32-bit devices, and integer constants can be modified (real number constants, and character constants cannot be specified).
*3: Index register (I0 to IE)
*4: Only operation unit: signed integers (SS, SL) can be specified.
*5: Only operation unit: unsigned integers (US, UL) can be specified.
*6: Only operation unit: integers (US, SS, UL, SL) can be specified.
*7: Only operation unit: single-precision, floating-point real numbers (SF) can be specified.
*8: Only operation unit: double-precision, floating-point real numbers (DF) can be specified.

■ Outline of operation

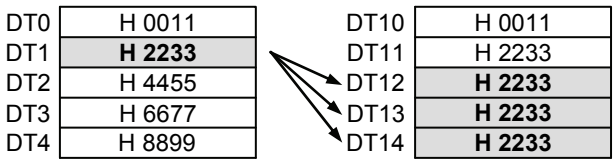
- The data specified by [S] are copied to the areas [D1] and [D2].

■ Process details

Example 1) Operation unit: 16 bits (US, SS)

[i]…US,SS

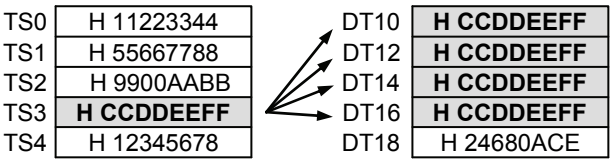
[S]…DT1 [D1]…DT12 [D2]…DT14



Example 2) Operation unit: 32 bits (UL, SL, SF)

[i]…UL,SL,SF

[S]…TS3 [D1]…DT10 [D2]…DT16

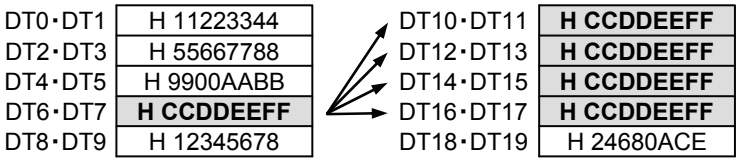


■ Precautions during programming

- In the case of a direct address and an index modification address, specify the same type of device for [D1] and [D2]. At the same time, specify [D2] ≥ [D1].
- The data are transferred by operation unit, ending with the device containing [D2].

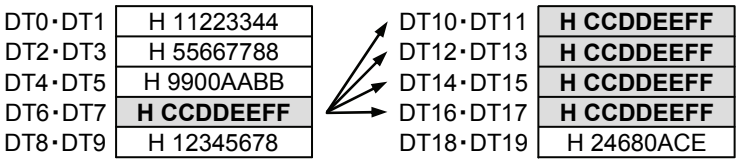
Example 1) Device address of [D2] comes to a higher word (operation unit = 32 bits)

[S]…DT6 [D1]…DT10 [D2]…DT17



Example 2) Device address of [D2] comes to a lower word (operation unit = 32 bits)

[S]…DT6 [D1]…DT10 [D2]…DT16

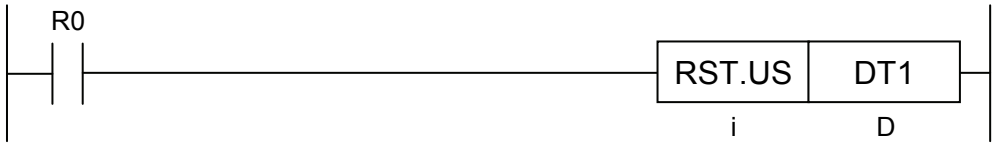


■ Flag operations

Name	Explanation
SR7	To be set in the case of out-of-range in indirect access (index modification).
SR8 (ER)	To be set when [D1] > [D2].

RST (Reset)

■ Ladder diagram



■ Available operation units (●: Available)

Operation unit	bit	US	SS	UL	SL	SF	DF
i		●	●	●	●	●	●

■ List of operands

Operand	Explanation
D	Targets of reset

■ Available devices (●: Available)

Operand	16-bit device											32-bit device *1			Integers			Real numbers		String	Index modifier *2
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX *3	K	U	H	SF	DF	" "	
D	●	●	●	●		*4	●	●	●		●	●	●	●							●

*1: Operation unit: 16-bit integers (SS, US) cannot be specified.
*2: Only 16-bit devices, and 32-bit devices can be modified (integer constants, real number constants, and character constants cannot be specified).
*3: Index register (I0 to IE)
*4: Only SD60/SD61 is permitted.

■ Outline of operation

- If the operation unit is 16 bits (US, SS) or 32 bits (UL, SL, SF), the area specified by [D] is reset (cleared to zero).

■ Process details

Example 1) Operation unit: 16 bits (US, SS)
[i]…US,SS
[D]…DT1

DT0	H 0011	→	DT0	H 0011
DT1	**H 2233**		DT1	**H 0000**
DT2	H 4455		DT2	H 4455
DT3	H 6677		DT3	H 6677
DT4	H 8899		DT4	H 8899

Example 2) Operation unit: 32 bits (UL, SL, SF)
[i]…UL,SL,SF
[D]…TS3

TS0	H 11223344	→	TS0	H 11223344
TS1	H 55667788		TS1	H 55667788
TS2	H 9900AABB		TS2	H 9900AABB
TS3	**H CCDDEEFF**		TS3	**H 00000000**
TS4	H 12345678		TS4	H 12345678

* In the case of SF, the value becomes '0.00e+00' when all the 32 bits are '0' ('0's for the sign, the exponent, and the mantissa)

■ Flag operations

Name	Explanation
SR7 SR8 (ER)	To be set in the case of out-of-range in indirect access (index modification).

BTM (Bit Block Transfer)

■ Ladder diagram

■ Available operation units (●: Available)

Operation unit	bit	US	SS	UL	SL	SF	DF
i	●						

■ List of operands

Operand	Explanation
S1	Starting bit address of the source data
S2	End bit address of the source data
D	Starting bit address of the data destination

■ Available devices (●: Available)

Operand	Bit device											Bit specification of the word device		Index modifier
	X	Y	R	L	T	C	P	E	S	IN	OT	DT.b	LD.b	
S1	●	●	●	●	●	●	●	●	●	●	●	●	●	●
S2	●	●	●	●	●	●	●	●	●	●	●	●	●	●
D	●	●	●	●							●	●	●	●

■ Outline of operation

- Bit transfer, from the area (bit address) specified by [S1] through the area (bit address) specified by [S2], to the area specified by [D].

■ Process details

Example 1) Transfer X1 through X8 to Y6 through YD
[S1]…X1 [S2]…X8 [D]…Y6

	WX0															
bit	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0
BIN	0	1	1	0	1	0	0	**1**	**1**	**1**	**0**	**0**	**0**	**0**	**1**	1

	WY0															
bit	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0
BIN	0	0	**1**	**1**	**1**	**0**	**0**	**0**	**0**	**1**	0	0	0	0	0	0

Example 2) Transfer X1 through X8 to YD through Y14
[S1]…X1 [S2]…X8 [D]…YD

	WX0															
bit	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0
BIN	0	1	1	0	1	0	0	**1**	**1**	**1**	**0**	**0**	**0**	**0**	**1**	1

	WY1																WY0											
bit	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0
BIN	0	0	0	0	0	0	0	0	0	0	0	**1**	**1**	**1**	**0**	**0**	**0**	**0**	**1**	0	0	0	0	0	0	0	0	0	0	0	0	0

■ Precautions during programming

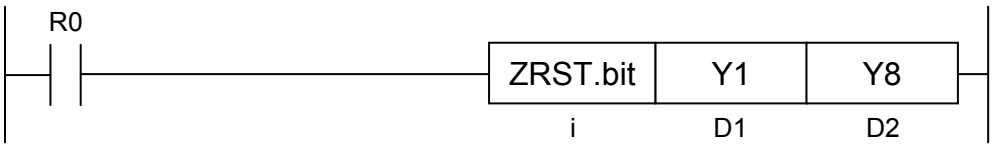
- In the case of a direct address and an index modification address, specify the same type of device for [S1] and [S2]. At the same time, specify [S2] ≥ [S1].

■ Flag operations

Name	Explanation
SR7	To be set in the case of out-of-range in indirect access (index modification).
SR8	To be set when [S1] > [S2].
(ER)	To be set when the destination range is out of the available range.

ZRST (Block Clear)

■ Ladder diagram



■ Available operation units (●: Available)

Operation unit	bit	US	SS	UL	SL	SF	DF
i	●						

■ List of operands

Operand	Explanation
D1	Starting bit address of the reset data
D2	End bit address of the reset data

■ Available devices (●: Available)

Operand	Bit device											Bit specification of the word device			Index modifier
	X	Y	R	L	T	C	P	E	S	IN	OT	DT.b	LD.b	SD.b	
D1	●	●	●	●							●	●	●		●
D2	●	●	●	●							●	●	●		●

■ Outline of operation

- Clear to zero (reset) from the area (bit address) specified by [D1] through the area (bit address) specified by [D2].
- This can also be used for a package clearance of processes that are starting up from Process [D1] to Process [D2] in the stepladder.

■ Process details

Example 1) Reset Y1 through Y8
[D1]…Y1 [D2]…Y8

	WY0															
bit	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0
BIN	1	1	1	1	1	1	1	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	1

	WY0															
bit	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0
BIN	1	1	1	1	1	1	1	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	1

Example 2) Reset YD through Y14
[D1]…YD [D2]…Y14

	WY1																WY0											
bit	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0
BIN	1	1	1	1	1	1	1	1	1	1	1	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	1	1	1	1	1	1	1	1	1	1	1	1	1

	WY1																WY0											
bit	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0
BIN	1	1	1	1	1	1	1	1	1	1	1	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	1	1	1	1	1	1	1	1	1	1	1	1	1

■ Precautions during programming

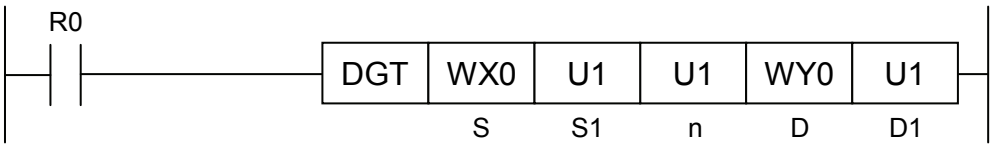
- In the case of a direct address and an index modification address, specify the same type of device for [D1] and [D2]. At the same time, specify [D2] ≥ [D1].

■ Flag operations

Name	Explanation
SR7 SR8 (ER)	To be set in the case of out-of-range in indirect access (index modification).
	To be set when [D1] > [D2].

DGT (Digit Data Transfer)

■ Ladder diagram



■ Available operation units (●: Available)

- No operation units

■ List of operands

Operand	Explanation
S	Device address or constant of the source data
S1	Transfer starting digit in the source (Data settable range: 0 to 3)
n	Digits to be transferred (Data settable range: 1 to 4)
D	Device address of the destination data
D1	Transfer starting digit in the destination (Data settable range: 0 to 3)

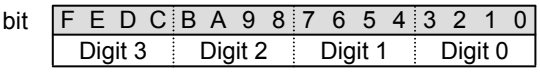
■ Available devices (●: Available)

Operand	16-bit device											32-bit device			Integers			Real numbers		String	Index modifier
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX	K	U	H	SF	DF	" "	
S	●	●	●	●	●	●	●	●	●	●	●				●	●	●				●
S1(*1)	●	●	●	●			●	●	●	●	●					●	●				●
n(*1)	●	●	●	●			●	●	●	●	●					●	●				●
D	●	●	●	●			●	●	●		●										●
D1(*1)	●	●	●	●			●	●	●	●	●					●	●				●

*1: To be handled as a 16-bit unsigned integer (US), regardless of operation unit.
*2: Only 16-bit devices, and integer constants can be modified (32-bit devices, real number constants, and character constants cannot be specified).

■ Outline of operation

- Transfer [n] digits from the [S1]th digit of the area specified by [S], to the [D1] digit of the 16-bit data specified by [D].
- Transfer starts with the 0th digit, 1st digit, 2nd digit, and 3rd digit by every four bits from the lower level.



■ Process details

Example 1) Transfer from Digit 1 to Digit 1

[S]…WX0 [S1]…U1(H1)
[n]…U1(H1)
[D]…WY0 [D1]…U1(H1)

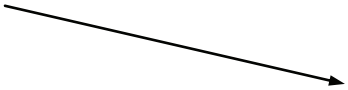
	X															
bit	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0
BIN	0	1	1	0	1	0	0	1	**1**	**1**	**0**	**0**	0	0	1	1

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

Example 2) Shift by one digit and transfer

[S]…WX0 [S1]…U3(H3)
[n]…U1(H1)
[D]…WY0 [D1]…U0(H0)

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



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

Example 3) Transfer multiple digits in parallel

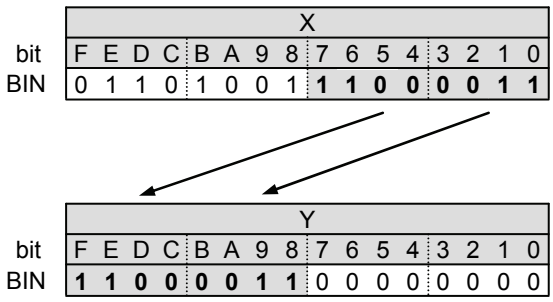
[S]…WX0 [S1]…U2(H2)
[n]…U2(H2)
[D]…WY0 [D1]…U2(H2)

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

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

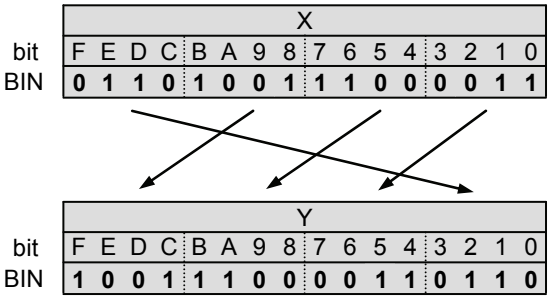
Example 4) Shift and transfer multiple digits

[S]…WX0 [S1]…U0(H0)
[n]…U2(H2)
[D]…WY0 [D1]…U2(H2)



Example 5) Transfer four digits

[S]…WX0 [S1]…U0(H0)
[n]…U4(H4)
[D]…WY0 [D1]…U1(H1)

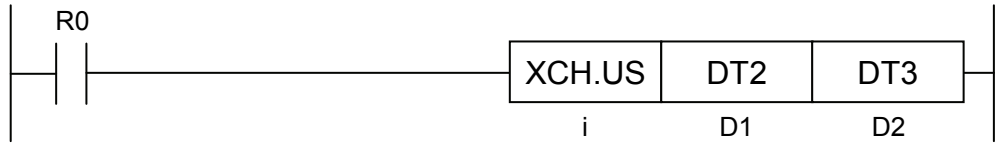


■ Flag operations

Name	Explanation
SR7	To be set in the case of out-of-range in indirect access (index modification).
SR8 (ER)	To be set when the operands [S1], [n], and/or [D1] are out of the specified range.

XCH (Data Exchange)

■ Ladder diagram



■ Available operation units (●: Available)

Operation unit	bit	US	SS	UL	SL	SF	DF
i		●	●	●	●	●	●

■ List of operands

Operand	Explanation
D1	Device address of exchanged data 1
D2	Device address of exchanged data 2

■ Available devices (●: Available)

Operand	16-bit device											32-bit device *1			Integers			Real numbers		String	Index modifier *2
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX *3	K	U	H	SF	DF	" "	
S1	●	●	●	●			●	●	●		●	●	●	●							●
S2	●	●	●	●			●	●	●		●	●	●	●							●

*1: Operation unit: 16-bit integers (SS, US) cannot be specified.
*2: Only 16-bit devices, and 32-bit devices can be modified (integer constants only, real number constants, and character constants cannot be specified).
*3: Index register (I0 to IE)

■ Outline of operation

- Exchange the data of the device address specified by [D1] and the device address specified by [D2] according to the operation unit [i].

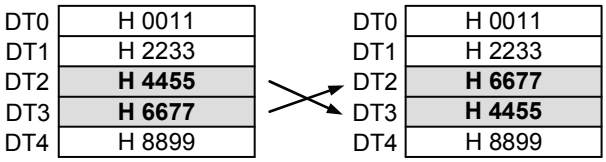
[D1] ⇔ [D2]

■ Process details

Example 1) Operation unit: 16 bits (US, SS)

[i]…US,SS

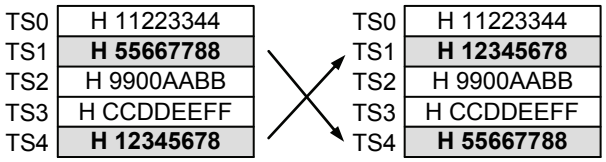
[D1]…DT2 [D2]…DT3



Example 2) Operation unit: 32 bits (UL, SL, SF)

[i]…UL,SL,SF

[D1]…TS1 [D2]…TS4



■ Precautions during programming

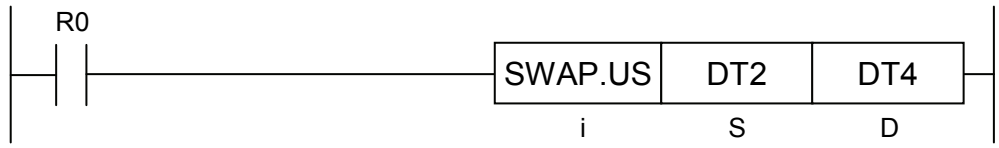
- Ensure that the ranges of the exchanged data do not overlap.

■ Flag operations

Name	Explanation
SR7 SR8 (ER)	To be set in the case of out-of-range in indirect access (index modification).

SWAP (Exchange of Higher Bytes and Lower Bytes)

■ Ladder diagram



■ Available operation units (●: Available)

Operation unit	bit	US	SS	UL	SL	SF	DF
i		●	●				

■ List of operands

Operand	Explanation
S	Device address of the source where higher bytes and lower bytes should be exchanged
D	Device address of destination of the exchanged data

■ Available devices (●: Available)

Operand	16-bit device											32-bit device			Integers			Real numbers		String	Index modifier *1
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX	K	U	H	SF	DF	" "	
S	●	●	●	●	●	●	●	●	●	●	●										●
D	●	●	●	●			●	●	●		●										●

*1: Only 16-bit devices can be modified (32-bit devices, integer constants, real number constants, and character constants cannot be specified)

■ Outline of operation

- Exchange higher bytes and lower bytes of the device address specified by [S], and transfer the resulting data to the device address specified by [D].

■ Process details

Example) Operation unit: 16 bits (US, SS)
[i]…US,SS
[S]…DT2 [D]…DT4

DT0	H 0011
DT1	H 2233
DT2	**H 4455**
DT3	H 6677
DT4	H 8899

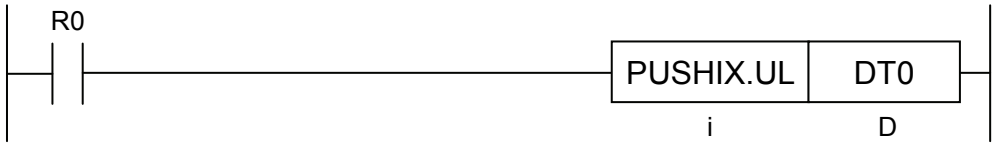
DT0	H 0011
DT1	H 2233
DT2	H 4455
DT3	H 6677
DT4	**H 5544**

■ Flag operations

Name	Explanation
SR7 SR8 (ER)	To be set in the case of out-of-range in indirect access (index modification).

PUSHIX (Index Register Backup)

■ Ladder diagram



■ Available operation units (●: Available)

Operation unit	bit	US	SS	UL	SL	SF	DF
i				●	●		

■ List of operands

Operand	Explanation
D	Start of device address of the backup destination

■ Available devices (●: Available)

Operand	16-bit device											32-bit device			Integers			Real numbers		String	Index modifier *1
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX *2	K	U	H	SF	DF	" "	
D	●	●	●	●			●	●													●

*1: Only 16-bit devices, and 32-bit devices can be modified (integer constants, real number constants, and character constants cannot be specified).
*2: Index register (I0 to IE)

■ Outline of operation

- Back up 15 data (30 words) of the index register value, starting with [D].

(I0 to IE) → ([D] to [D] + 29)

- The index register value at source does not change.
- The "PUSHIX" instruction is used to back up the index register content before switching from the main program to the sub program, in the case where index registers are used in sub routine or other sub programs.
- Please use this in combination with the "POPIX" (Index Register Recovery) instruction.

■ Process details

Example) Specify DT0 for the 1st operand [D]

			Before backup	After backup
I0	**H 00112233**	DT0	H 00000000	**H 00112233**
I1	**H 44556677**	DT2	H 00000000	**H 44556677**
I2	**H 8899AABB**	DT4	H 00000000	**H 8899AABB**
⋮	⋮	⋮	⋮	⋮
IC	**H CCDDEEFF**	DT24	H 00000000	**H CCDDEEFF**
ID	**H 12345678**	DT26	H 00000000	**H 12345678**
IE	**H 90ABCDEF**	DT28	H 00000000	**H 90ABCDEF**

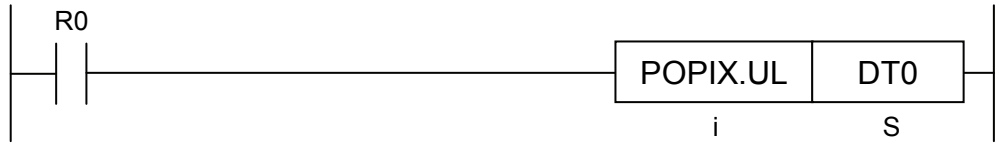
Backup

■ Flag operations

Name	Explanation
SR7	To be set in the case of out-of-range in indirect access (index modification).
SR8 (ER)	To be set when the backup destination range is out of the available range.

POPIX (Index Register Recovery)

■ Ladder diagram



■ Available operation units (●: Available)

Operation unit	bit	US	SS	UL	SL	SF	DF
i				●	●		

■ List of operands

Operand	Explanation
S	Start of device address of the recovery source

■ Available devices (●: Available)

Operand	16-bit device											32-bit device			Integers			Real numbers		String	Index modifier *1
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX *2	K	U	H	SF	DF	" "	
S	●	●	●	●			●	●													●

*1: Only 16-bit devices, and 32-bit devices can be modified (integer constants, real number constants, and character constants cannot be specified).
*2: Index register (I0 to IE)

■ Outline of operation

- Recover 15 data (30 words) for the index register value, starting with [S].
- ([S] to [S] + 29) → (I0 to IE)
- The "POPIX" instruction is used to recover the content that has been backed up using the "PUSHIX" (Index Register Backup) instruction, before switching from the sub program to the main program, in the case where index registers are used in sub routine or other sub programs.
- Please use this in combination with the "PUSHIX" (Index Register Backup) instruction.

■ Process details

Example) Specify DT0 for the 1st operand [S]

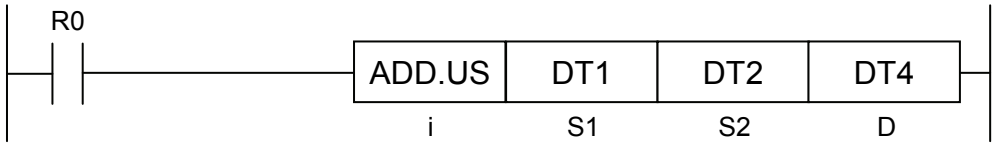
				Before recovery	After recovery
DT0	**H 00112233**	Recovery	I0	H 00000000	**H 00112233**
DT2	**H 44556677**		I1	H 00000000	**H 44556677**
DT4	**H 8899AABB**		I2	H 00000000	**H 8899AABB**
⋮	⋮		⋮	⋮	⋮
DT24	**H CCDDEEFF**		IC	H 00000000	**H CCDDEEFF**
DT26	**H 12345678**		ID	H 00000000	**H 12345678**
DT28	**H 90ABCDEF**		IE	H 00000000	**H 90ABCDEF**

■ Flag operations

Name	Explanation
SR7	To be set in the case of out-of-range in indirect access (index modification).
SR8 (ER)	To be set when the recovery source range is out of the available range.

ADD (Addition)

■ Ladder diagram



■ Available operation units (●: Available)

Operation unit	bit	US	SS	UL	SL	SF	DF
i		●	●	●	●	●	●

■ List of operands

Operand	Explanation
S1	Calculation target data 1 (device address or constant)
S2	Calculation target data 2 (device address or constant)
D	Calculation result data (device address)

■ Available devices (●: Available)

Operand	16-bit device											32-bit device *1			Integers			Real numbers		String	Index modifier *2
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX *3	K *4	U *5	H *6	SF *7	DF *8	" "	
S1	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●		●
S2	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●		●
D	●	●	●	●			●	●	●		●	●	●	●							●

*1: Operation unit: 16-bit integers (SS, US) cannot be specified.
*2: Only 16-bit devices, 32-bit devices, and integer constants can be modified (real number constants, and character constants cannot be specified).
*3: Index register (I0 to IE)
*4: Only operation unit: signed integers (SS, SL) can be specified.
*5: Only operation unit: unsigned integers (US, UL) can be specified.
*6: Only operation unit: integers (US, SS, UL, SL) can be specified.
*7: Only operation unit: single-precision, floating-point real numbers (SF) can be specified.
*8: Only operation unit: double-precision, floating-point real numbers (DF) can be specified.

■ Outline of operation

- Add the values [S1] and [S2] according to the operation unit [i].
- The calculation result is stored in the address starting with [D].

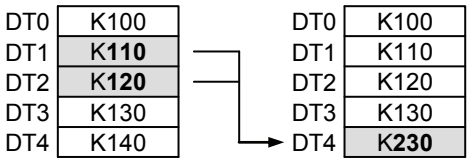
[S1] + [S2] → [D]

■ Process details

Example 1) Operation unit: 16 bits (US, SS)

[i]…US,SS

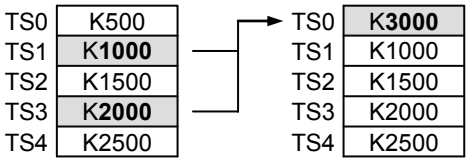
[S1]…DT1 [S2]…DT2 [D]…DT4



Example 2) Operation unit: 32 bits (UL, SL, SF)

[i]…UL,SL,SF

[S1]…TS1 [S2]…TS3 [D]…TS0

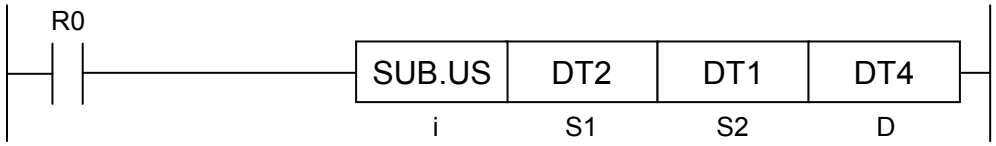


■ Flag operations

Name	Explanation
SR7	To be set in the case of out-of-range in indirect access (index modification).
SR8 (ER)	To be set when a non-real number is specified for [S1] or [S2], and the operation unit is a real number (SF).

SUB (Subtraction)

■ Ladder diagram



■ Available devices (●: Available)

Operation unit	bit	US	SS	UL	SL	SF	DF
i		●	●	●	●	●	●

■ List of operands

Operand	Explanation
S1	Calculation target data 1 (device address or constant)
S2	Calculation target data 2 (device address or constant)
D	Calculation result data (device address)

■ Available devices (●: Available)

Operand	16-bit device											32-bit device *1			Integers			Real numbers		String	Index modifier *2
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX *3	K *4	U *5	H *6	SF *7	DF *8	" "	
S1	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●		●
S2	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●		●
D	●	●	●	●			●	●	●		●	●	●	●							●

*1: Operation unit: 16-bit integers (SS, US) cannot be specified.
*2: Only 16-bit devices, 32-bit devices, and integer constants can be modified (real number constants, and character constants cannot be specified).
*3: Index register (I0 to IE)
*4: Only operation unit: signed integers (SS, SL) can be specified.
*5: Only operation unit: unsigned integers (US, UL) can be specified.
*6: Only operation unit: integers (US, SS, UL, SL) can be specified.
*7: Only operation unit: single-precision, floating-point real numbers (SF) can be specified.
*8: Only operation unit: double-precision, floating-point real numbers (DF) can be specified.

■ Outline of operation

- Subtract the value [S2] from [S1] according to the operation unit [i].
- The calculation result is stored in the address starting with [D].

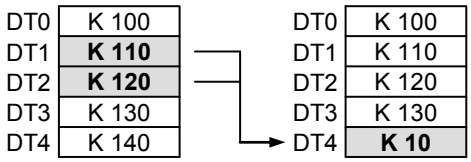
[S1] - [S2] → [D]

■ Process details

Example 1) Operation unit: 16 bits (US, SS)

[i]…US,SS

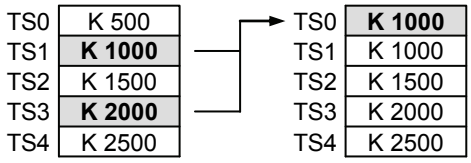
[S1]…DT2 [S2]…DT1 [D]…DT4



Example 2) Operation unit: 32 bits (UL, SL, SF)

[i]…UL,SL,SF

[S1]…TS3 [S2]…TS1 [D]…TS0

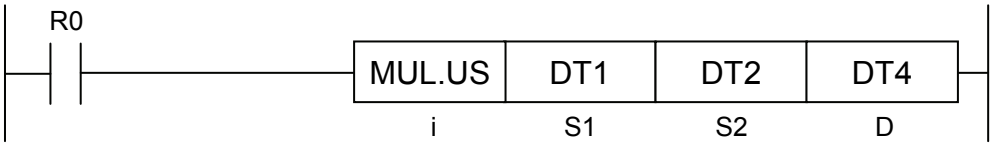


■ Flag operations

Name	Explanation
SR7	To be set in the case of out-of-range in indirect access (index modification).
SR8 (ER)	To be set when a non-real number is specified for [S1] or [S2], and the operation unit is a real number (SF).

MUL (Multiplication)

■ Ladder diagram



■ Available operation units (●: Available)

Operation unit	bit	US	SS	UL	SL	SF	DF
i		●	●	●	●	●	●

■ List of operands

Operand	Explanation
S1	Calculation target data 1 (device address or constant)
S2	Calculation target data 2 (device address or constant)
D	Calculation result data (device address)

■ Available devices (●: Available)

Operand	16-bit device											32-bit device *1			Integers			Real numbers		String	Index modifier *2
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX *3	K *4	U *5	H *6	SF *7	DF *8	" "	
S1	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●		●
S2	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●		●
D	●	●	●	●			●	●	●		●	●	●	●							●

*1: Operation unit: 16-bit integers (SS, US) cannot be specified.
*2: Only 16-bit devices, 32-bit devices, and integer constants can be modified (real number constants, and character constants cannot be specified).
*3: Index register (I0 to IE)
*4: Only operation unit: signed integers (SS, SL) can be specified.
*5: Only operation unit: unsigned integers (US, UL) can be specified.
*6: Only operation unit: integers (US, SS, UL, SL) can be specified.
*7: Only operation unit: single-precision, floating-point real numbers (SF) can be specified.
*8: Only operation unit: double-precision, floating-point real numbers (DF) can be specified.

■ Outline of operation

- Multiply the values [S1] and [S2] according to the operation unit [i].
- The calculation result is stored in the address starting with [D].

[S1] × [S2] → ([D], [D]+1)

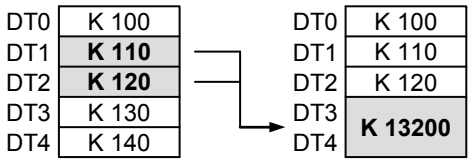
- The size of [D] should be twice the operation unit in the case of integer.

■ Process details

Example 1) Operation unit: 16 bits (US, SS)

[i]…US,SS

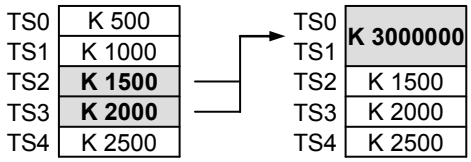
[S1]…DT1 [S2]…DT2 [D]…DT3



Example 2) Operation unit: 32 bits (UL, SL, SF)

[i]…UL,SL,SF

[S1]…TS2 [S2]…TS3 [D]…TS0



■ Precautions during programming

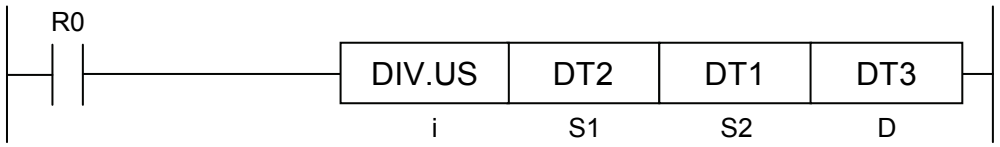
- The area where the calculation result should be stored has twice the size of the operation unit. Allocate the memory area so that other areas will not be overwritten.

■ Flag operations

Name	Explanation
SR7	To be set in the case of out-of-range in indirect access (index modification).
SR8 (ER)	To be set when a non-real number is specified for [S1] or [S2], and the operation unit is a real number (SF).

DIV (Division)

■ Ladder diagram



■ Available operation units (●: Available)

Operation unit	bit	US	SS	UL	SL	SF	DF
i		●	●	●	●	●	●

■ List of operands

Operand	Explanation
S1	Calculation target data 1 (device address or constant)
S2	Calculation target data 2 (device address or constant)
D	Calculation result data (device address)

■ Available devices (●: Available)

Operand	16-bit device											32-bit device *1			Integers			Real numbers		String	Index modifier *2
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX *3	K *4	U *5	H *6	SF *7	DF *8	" "	
S1	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●		●
S2	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●		●
D	●	●	●	●			●	●	●		●	●	●	●							●

*1: Operation unit: 16-bit integers (SS, US) cannot be specified.
*2: Only 16-bit devices, 32-bit devices, and integer constants can be modified (real number constants, and character constants cannot be specified).
*3: Index register (I0 to IE)
*4: Only operation unit: signed integers (SS, SL) can be specified.
*5: Only operation unit: unsigned integers (US, UL) can be specified.
*6: Only operation unit: integers (US, SS, UL, SL) can be specified.
*7: Only operation unit: single-precision, floating-point real numbers (SF) can be specified.
*8: Only operation unit: double-precision, floating-point real numbers (DF) can be specified.

■ Outline of operation

- Divide the value [S1] with [S2] according to the operation unit [i].
- The calculation result is stored in the address starting with [D].

[S1] / [S2] → Quotient ([D])

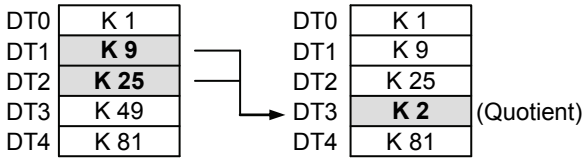
- The residue is not output. If the residue is required, use the DIVMOD instruction.

■ Process details

Example 1) Operation unit: 16 bits (US, SS)

[i]…US,SS

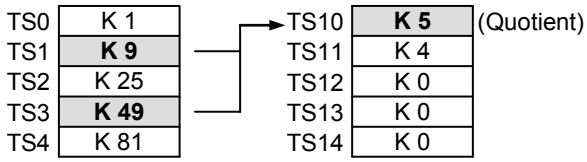
[S1]…DT2 [S2]…DT1 [D]…DT3



Example 2) Operation unit: 32 bits (UL, SL, SF)

[i]…UL,SL,SF

[S1]…TS3 [S2]…TS1 [D]…TS10



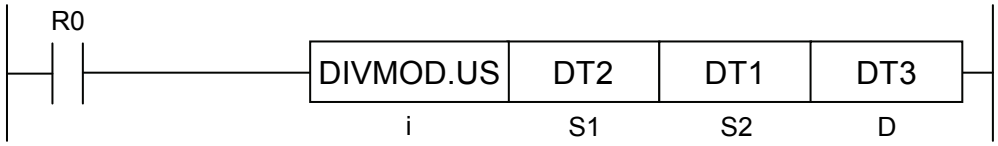
* Setting of a residue is not necessary for a real number.

■ Flag operations

Name	Explanation
SR7	To be set in the case of out-of-range in indirect access (index modification).
SR8	To be set when a non-real number is specified for [S1] or [S2], and the operation unit is a real number (SF).
(ER)	To be set when '0' is specified for [S2].

DIVMOD (Division)

■ Ladder diagram



■ Available operation units (●: Available)

Operation unit	bit	US	SS	UL	SL	SF	DF
i		●	●	●	●		

■ List of operands

Operand	Explanation
S1	Calculation target data 1 (device address or constant)
S2	Calculation target data 2 (device address or constant)
D	Calculation result data (device address)

■ Available devices (●: Available)

Operand	16-bit device											32-bit device *1			Integers			Real numbers		String	Index modifier *2
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX *3	K *4	U *5	H *6	SF	DF	" "	
S1	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●				●
S2	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●				●
D	●	●	●	●			●	●	●		●	●	●	●							●

*1: Operation unit: 16-bit integers (SS, US) cannot be specified.
*2: Only 16-bit devices, 32-bit devices, and integer constants can be modified (real number constants, and character constants cannot be specified).
*3: Index register (I0 to IE)
*4: Only operation unit: signed integers (SS, SL) can be specified.
*5: Only operation unit: unsigned integers (US, UL) can be specified.
*6: Only operation unit: integers (US, SS, UL, SL) can be specified.

■ Outline of operation

- Divide the value [S1] with [S2] according to the operation unit [i].
- The calculation result is stored in the address starting with [D].

[S1] / [S2] → Quotient ([D]), Residue ([D]+1)

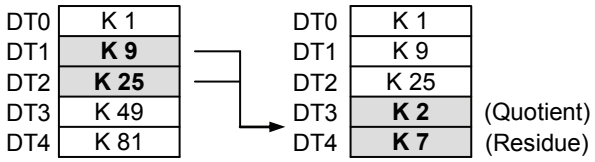
- In the case of integer, the residue should be specified as [D+1].

■ Process details

Example 1) Operation unit: 16 bits (US, SS)

[i]…US,SS

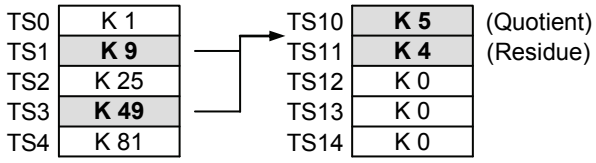
[S1]…DT2 [S2]…DT1 [D]…DT3



Example 2) Operation unit: 32 bits (UL, SL, SF)

[i]…UL,SL,SF

[S1]…TS3 [S2]…TS1 [D]…TS10



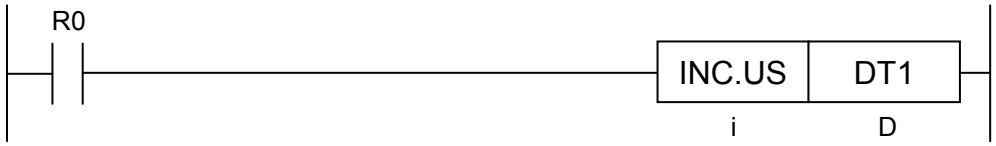
* Setting of a residue is not necessary for a real number.

■ Flag operations

Name	Explanation
SR7	To be set in the case of out-of-range in indirect access (index modification).
SR8	To be set when a non-real number is specified for [S1] or [S2], and the operation unit is a real number (SF).
(ER)	To be set when '0' is specified for [S2].

INC (Increment)

■ Ladder diagram



■ Available operation units (●: Available)

Operation unit	bit	US	SS	UL	SL	SF	DF
i		●	●	●	●	●	●

■ List of operands

Operand	Explanation
D	Calculation result data (device address)

■ Available devices (●: Available)

Operand	16-bit device											32-bit device *1			Integers			Real numbers		String	Index modifier *2
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX *3	K	U	H	SF	DF	" "	
D	●	●	●	●			●	●	●		●	●	●	●							●

*1: Operation unit: 16-bit integers (SS, US) cannot be specified.
*2: Only 16-bit devices, and 32-bit devices can be modified (integer constants, real number constants, and character constants cannot be specified).
*3: Index register (I0 to IE)

■ Outline of operation

- Add 1 to the value of [D] according to the operation unit [i].
- The calculation result is stored in the address starting with [D].

[D] + 1 → [D]

■ Process details

Example 1) Operation unit: 16 bits (US, SS)
[i]…US,SS
[D]…DT1

DT0	K 100	→	DT0	K 100
DT1	**K 110**		DT1	**K 111**
DT2	K 120		DT2	K 120
DT3	K 130		DT3	K 130

Example 2) Operation unit: 32 bits (UL, SL, SF)
[i]…UL,SL,SF
[D]…TS1

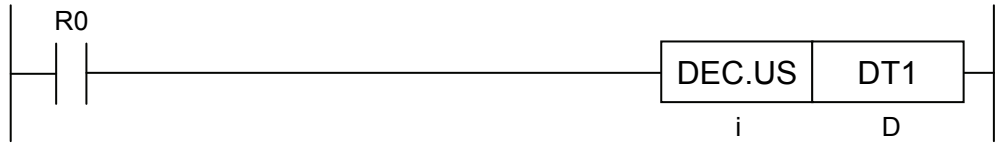
TS0	K 500	→	TS0	K 500
TS1	**K 1000**		TS1	**K 1001**
TS2	K 1500		TS2	K 1500
TS3	K 2000		TS3	K 2000

■ Flag operations

Name	Explanation
SR7	To be set in the case of out-of-range in indirect access (index modification).
SR8 (ER)	To be set when a non-real number is specified for [D], and the operation unit is a real number (SF).

DEC (Decrement)

■ Ladder diagram



■ Available operation units (●: Available)

Operation unit	bit	US	SS	UL	SL	SF	DF
i		●	●	●	●	●	●

■ List of operands

Operand	Explanation
D	Calculation target data (device address)

■ Available devices (●: Available)

Operand	16-bit device											32-bit device			Integers			Real numbers		String	Index modifier *1
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX	K	U	H	SF	DF	" "	
D	●	●	●	●			●	●	●		●	●	●	●							●

*1: Operation unit: 16-bit integers (SS, US) cannot be specified.
*2: Only 16-bit devices, and 32-bit devices can be modified (integer constants, real number constants, and character constants cannot be specified).
*3: Index register (I0 to IE)

■ Outline of operation

- Subtract 1 from the value of [D] according to the operation unit [i].
- The calculation result is stored in the address starting with [D].

[D] - 1 → [D]

■ Process details

Example 1) Operation unit: 16 bits (US, SS)
[i]…US,SS
[D]…DT1

DT0	K 100	→	DT0	K 100
DT1	**K 110**		DT1	**K 109**
DT2	K 120		DT2	K 120
DT3	K 130		DT3	K 130

Example 2) Operation unit: 32 bits (UL, SL, SF)
[i]…UL,SL,SF
[D]…TS1

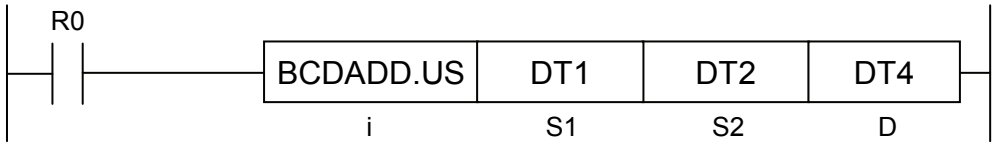
TS0	K 500	→	TS0	K 500
TS1	**K 1000**		TS1	**K 999**
TS2	K 1500		TS2	K 1500
TS3	K 2000		TS3	K 2000

■ Flag operations

Name	Explanation
SR7	To be set in the case of out-of-range in indirect access (index modification).
SR8 (ER)	To be set when a non-real number is specified for [D], and the operation unit is a real number (SF).

BCDADD (BCD Data Addition)

■ Ladder diagram



■ Available operation units (●: Available)

Operation unit	bit	US	SS	UL	SL	SF	DF
i		●		●			

■ List of operands

Operand	Explanation
S1	Calculation target data 1 (device address or constant) (settable data: H0 to H9 for each digit)
S2	Calculation target data 2 (device address or constant) (settable data: H0 to H9 for each digit)
D	Calculation result data (device address)

■ Available devices (●: Available)

Operand	16-bit device											32-bit device *1			Integers			Real numbers		String	Index modifier *2
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX *3	K	U	H	SF	DF	" "	
S1	●	●	●	●	●	●	●	●	●	●	●	●	●	●			●				●
S2	●	●	●	●	●	●	●	●	●	●	●	●	●	●			●				●
D	●	●	●	●			●	●	●		●	●	●	●							●

*1: Operation unit: 16-bit integers (SS, US) cannot be specified.
*2: Only 16-bit devices, and 32-bit devices can be modified (integer constants, real number constants, and character constants cannot be specified).
*3: Index register (I0 to IE)

■ Outline of operation

- Add the BCD data for [S1] and [S2] according to the operation unit [i].
- The calculation result is stored in the address starting with [D].

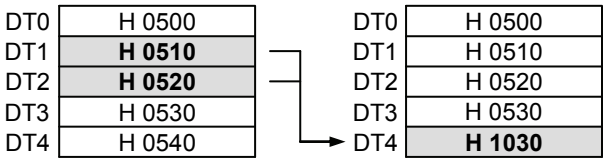
[S1] + [S2] → [D]

■ Process details

Example 1) Operation unit: 16 bits (US)

[i]…US

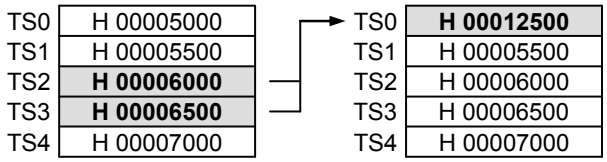
[S1]…DT1 [S2]…DT2 [D]…DT4



Example 2) Operation unit: 32 bits (UL)

[i]…UL

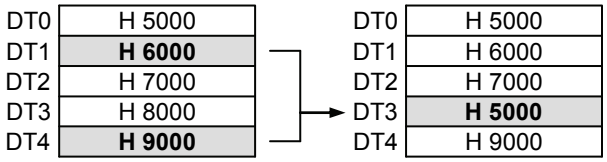
[D1]…TS2 [D2]…TS3 [D]…TS0



Example 3) Operation unit: 16 bits (US)

[i]…US

[S1]…DT1 [S2]…DT4 [D]…DT3

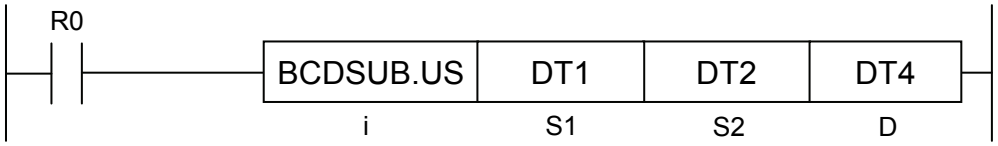


■ Flag operations

Name	Explanation
SR7	To be set in the case of out-of-range in indirect access (index modification).
SR8 (ER)	To be set when data other than BCD are specified for the targeted data [S1] or [S2].

BCDSUB (BCD Data Subtraction)

■ Ladder diagram



■ Available operation units (●: Available)

Operation unit	bit	US	SS	UL	SL	SF	DF
i		●		●			

■ List of operands

Operand	Explanation
S1	Calculation target data 1 (device address or constant) (settable data: H0 to H9 for each digit)
S2	Calculation target data 2 (device address or constant) (settable data: H0 to H9 for each digit)
D	Calculation result data (device address)

■ Available devices (●: Available)

Operand	16-bit device											32-bit device *1			Integers			Real numbers		String	Index modifier *2
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX *3	K	U	H	SF	DF	" "	
S1	●	●	●	●	●	●	●	●	●	●	●	●	●	●			●				●
S2	●	●	●	●	●	●	●	●	●	●	●	●	●	●			●				●
D	●	●	●	●			●	●	●		●	●	●	●							●

*1: Operation unit: 16-bit integers (SS, US) cannot be specified.
*2: Only 16-bit devices, and 32-bit devices can be modified (integer constants, real number constants, and character constants cannot be specified).
*3: Index register (I0 to IE)

■ Outline of operation

- Subtract the BCD data for [S1] from [S2] according to the operation unit [i].
- The calculation result is stored in the address starting with [D].

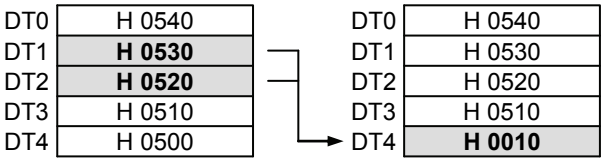
[S1] - [S2] → [D]

■ Process details

Example 1) Operation unit: 16 bits (US)

[i]…US

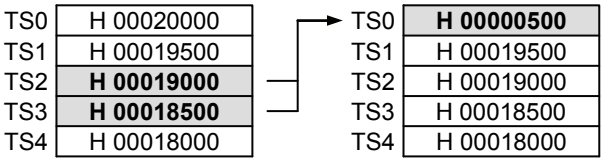
[S1]…DT1 [S2]…DT2 [D]…DT4



Example 2) Operation unit: 32 bits (UL)

[i]…UL

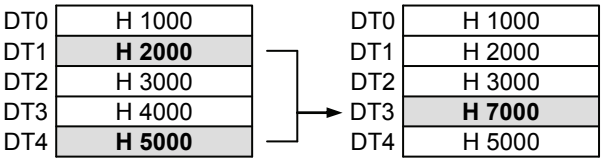
[D1]…TS2 [D2]…TS3 [D]…TS0



Example 3) Operation unit: 16 bits (US)

[i]…US

[S1]…DT1 [S2]…DT4 [D]…DT3

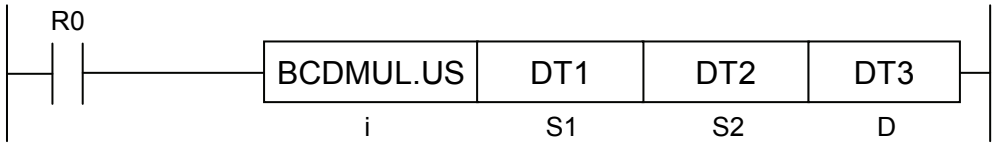


■ Flag operations

Name	Explanation
SR7	To be set in the case of out-of-range in indirect access (index modification).
SR8 (ER)	To be set when data other than BCD are specified for the targeted data [S1] or [S2].

BCDMUL (BCD Data Multiplication)

■ Ladder diagram



■ Available operation units (●: Available)

Operation unit	bit	US	SS	UL	SL	SF	DF
i		●		●			

■ List of operands

Operand	Explanation
S1	Calculation target data 1 (device address or constant) (settable data: H0 to H9 for each digit)
S2	Calculation target data 2 (device address or constant) (settable data: H0 to H9 for each digit)
D	Calculation result data (device address)

■ Available devices (●: Available)

Operand	16-bit device											32-bit device *1			Integers			Real numbers		String	Index modifier *2
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX *3	K	U	H	SF	DF	" "	
S1	●	●	●	●	●	●	●	●	●	●	●	●	●	●			●				●
S2	●	●	●	●	●	●	●	●	●	●	●	●	●	●			●				●
D	●	●	●	●			●	●	●		●	●	●	●							●

*1: Operation unit: 16-bit integers (SS, US) cannot be specified.
*2: Only 16-bit devices, and 32-bit devices can be modified (integer constants, real number constants, and character constants cannot be specified).
*3: Index register (I0 to IE)

■ Outline of operation

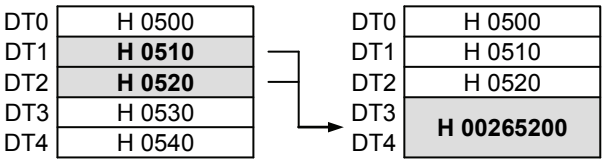
- Multiply the BCD data for [S1] and [S2] according to the operation unit [i].
- The calculation result is stored in the address starting with [D].

[S1] × [S2] → ([D], [D]+1)

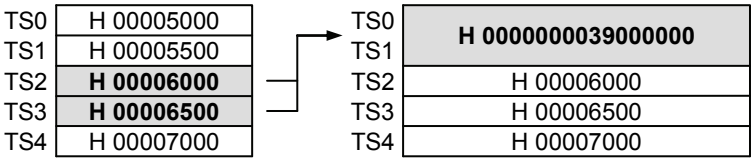
- The output of calculation result for [D] should be twice the operation unit.

■ Process details

Example 1) Operation unit: 16 bits (US)
[i]…US
[S1]…DT1 [S2]…DT2 [D]…DT3



Example 2) Operation unit: 32 bits (UL)
[i]…UL
[S1]…TS2 [S2]…TS3 [D]…TS0



■ Precautions during programming

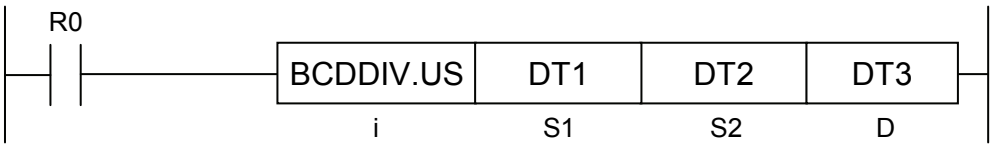
- When data around the end of the operation device is specified for [D], the memory for the subsequent device may be damaged over an area twice the size of the operation unit.

■ Flag operations

Name	Explanation
SR7	To be set in the case of out-of-range in indirect access (index modification).
SR8 (ER)	To be set when data other than BCD are specified for the targeted data [S1] or [S2].

BCDDIV (BCD Data Division)

■ Ladder diagram



■ Available operation units (●: Available)

Operation unit	bit	US	SS	UL	SL	SF	DF
i		●		●			

■ List of operands

Operand	Explanation
S1	Calculation target data 1 (device address or constant) (settable data: H0 to H9 for each digit)
S2	Calculation target data 2 (device address or constant) (settable data: H0 to H9 for each digit)
D	Calculation result data (device address)

■ Available devices (●: Available)

Operand	16-bit device											32-bit device *1			Integers			Real numbers		String	Index modifier *2
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX *3	K	U	H	SF	DF	" "	
S1	●	●	●	●	●	●	●	●	●	●	●	●	●	●			●				●
S2	●	●	●	●	●	●	●	●	●	●	●	●	●	●			●				●
D	●	●	●	●			●	●	●		●	●	●	●							●

*1: Operation unit: 16-bit integers (SS, US) cannot be specified.
*2: Only 16-bit devices, and 32-bit devices can be modified (integer constants, real number constants, and character constants cannot be specified).
*3: Index register (I0 to IE)

■ Outline of operation

- Divide [S1] with the BCD data for [S2] according to the operation unit [i].
- The calculation result is stored in the address starting with [D].

[S1] / [S2] → Quotient ([D]), Residue ([D]+1)

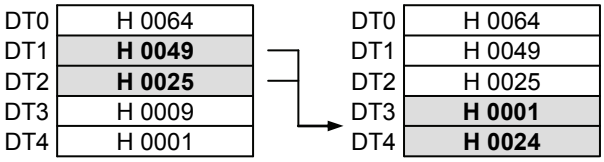
- The residue is stored in ([D]+1).

■ Process details

Example 1) Operation unit: 16 bits (US)

[i]…US

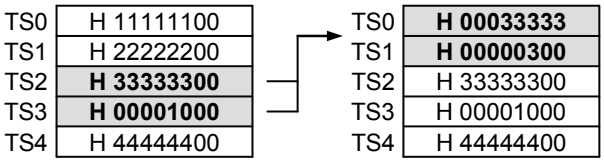
[S1]…DT1 [S2]…DT2 [D]…DT3



Example 2) Operation unit: 32 bits (UL)

[i]…UL

[S1]…TS2 [S2]…TS3 [D]…TS0

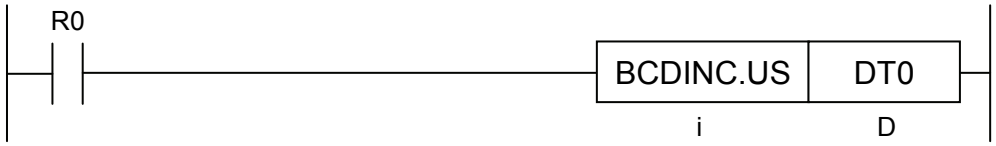


■ Flag operations

Name	Explanation
SR7	To be set in the case of out-of-range in indirect access (index modification).
SR8	To be set when '0' is specified for [S2].
(ER)	To be set when data other than BCD are specified for the targeted data [S1] or [S2].

BCDINC (BCD Data Increment)

■ Ladder diagram



■ Available operation units (●: Available)

Operation unit	bit	US	SS	UL	SL	SF	DF
i		●		●			

■ List of operands

Operand	Explanation
D	Calculation target data (device address)

■ Available devices (●: Available)

Operand	16-bit device											32-bit device *1			Integers			Real numbers		String	Index modifier *2
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX *3	K	U	H	SF	DF	" "	
D	●	●	●	●			●	●	●		●	●	●	●							●

*1: Operation unit: 16-bit integers (SS, US) cannot be specified.
*2: Only 16-bit devices, and 32-bit devices can be modified (integer constants, real number constants, and character constants cannot be specified).
*3: Index register (I0 to IE)

■ Outline of operation

- Add 1 to the BCD data of [D] according to the operation unit [i].
- The calculation result is stored in the address starting with [D].

[D] + 1 → [D]

■ Process details

Example 1) Operation unit: 16 bits (US)
[i]…US
[D]…DT0

DT0	**H 0100**	→	DT0	**H 0101**
DT1	H 0200		DT1	H 0200
DT2	H 0300		DT2	H 0300
DT3	H 0400		DT3	H 0400

Example 2) Operation unit: 32 bits (UL)

[i]…UL

[D]…TS1

TS0	H 01000000	→	TS0	H 01000000
TS1	**H 01999999**		TS1	**H 02000000**
TS2	H 03000000		TS2	H 03000000
TS3	H 04000000		TS3	H 04000000

Example 3) Operation unit: 16 bits (US)

[i]…US

[D]…DT0

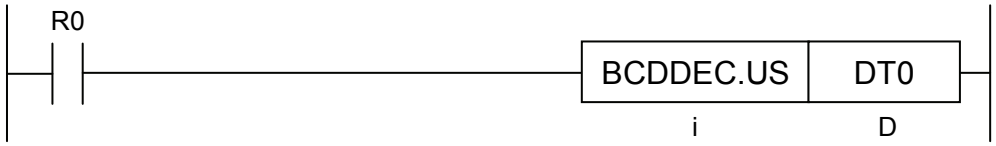
DT0	**H 9999**	→	DT0	**H 0000**
DT1	H 0200		DT1	H 0200
DT2	H 0300		DT2	H 0300
DT3	H 0400		DT3	H 0400

■ Flag operations

Name	Explanation
SR7	To be set in the case of out-of-range in indirect access (index modification).
SR8 (ER)	To be set when data other than BCD are specified for the targeted data [D].

BCDDEC (BCD Data Decrement)

■ Ladder diagram



■ Available operation units (●: Available)

Operation unit	bit	US	SS	UL	SL	SF	DF
i		●		●			

■ List of operands

Operand	Explanation
D	Calculation target data (device address)

■ Available devices (●: Available)

Operand	16-bit device											32-bit device *1			Integers			Real numbers		String	Index modifier *2
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX *3	K	U	H	SF	DF	" "	
D	●	●	●	●			●	●	●		●	●	●	●							●

*1: Operation unit: 16-bit integers (SS, US) cannot be specified.
*2: Only 16-bit devices, and 32-bit devices can be modified (integer constants, real number constants, and character constants cannot be specified).
*3: Index register (I0 to IE)

■ Outline of operation

- Subtract 1 from the BCD data of [D] according to the operation unit [i].
- The calculation result is stored in the area starting with [D].

[D] - 1 → [D]

■ Process details

Example 1) Operation unit: 16 bits (US)
[i]…US
[D]…DT0

DT0	**H 0100**	→	DT0	**H 0099**
DT1	H 0200		DT1	H 0200
DT2	H 0300		DT2	H 0300
DT3	H 0400		DT3	H 0400

Example 2) Operation unit: 32 bits (UL)

[i]…UL

[D]…TS1

TS0	H 01000000	→	TS0	H 01000000
TS1	**H 02000000**		TS1	**H 01999999**
TS2	H 03000000		TS2	H 03000000
TS3	H 04000000		TS3	H 04000000

Example 3) Operation unit: 16 bits (US)

[i]…US

[D]…DT0

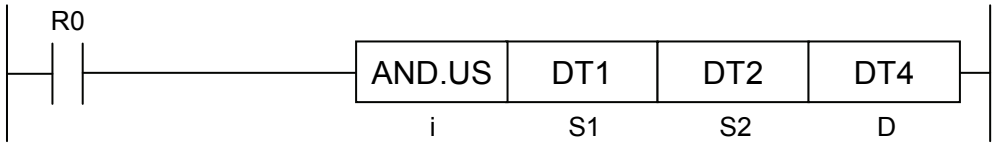
DT0	**H 0000**	→	DT0	**H 9999**
DT1	H 0200		DT1	H 0200
DT2	H 0300		DT2	H 0300
DT3	H 0400		DT3	H 0400

■ Flag operations

Name	Explanation
SR7	To be set in the case of out-of-range in indirect access (index modification).
SR8 (ER)	To be set when data other than BCD are specified for the targeted data [D].

AND (Logical Conjunction)

■ Ladder diagram



■ Available operation units (●: Available)

Operation unit	bit	US	SS	UL	SL	SF	DF
i		●	●	●	●		

■ List of operands

Operand	Explanation
S1	Calculation target data 1 (device address or constant)
S2	Calculation target data 2 (device address or constant)
D	Calculation result data (device address)

■ Available devices (●: Available)

Operand	16-bit device											32-bit device *1			Integers			Real numbers		String	Index modifier *2
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX *3	K *4	U *5	H *6	SF	DF	" "	
S1	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●				●
S2	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●				●
D	●	●	●	●			●	●	●		●	●	●	●							●

*1: Operation unit: 16-bit integers (SS, US) cannot be specified.
*2: Only 16-bit devices, 32-bit devices, and integer constants can be modified (real number constants, and character constants cannot be specified).
*3: Index register (I0 to IE)
*4: Only operation unit: signed integers (SS, SL) can be specified.
*5: Only operation unit: unsigned integers (US, UL) can be specified.
*6: Only operation unit: integers (US, SS, UL, SL) can be specified.

■ Outline of operation

- Calculate the logical conjunction for [S1] and [S2] according to the operation unit [i].
- The calculation result is stored in the area starting with [D].

[S1] ∧ [S2] → [D]

■ Process details

■ Logical conjunction (AND)

[S1] bit	[S2] bit	Logical conjunction
0	0	0
0	1	0
1	0	0
1	1	1

Example 1) Operation unit: 16 bits (US, SS)
[i]…US,SS
[S1]…DT1 [S2]…DT2 [D]…DT4

DT0	H 1234	DT0	H 1234
DT1	**H FF00**	DT1	H FF00
DT2	**H 5678**	DT2	H 5678
DT3	H 00FF	DT3	H 00FF
DT4	H 90AB	DT4	**H 5600**

Example 2) Operation unit: 32 bits (UL, SL)
[i]…UL,SL
[S1]…TS1 [S2]…TS3 [D]…TS0

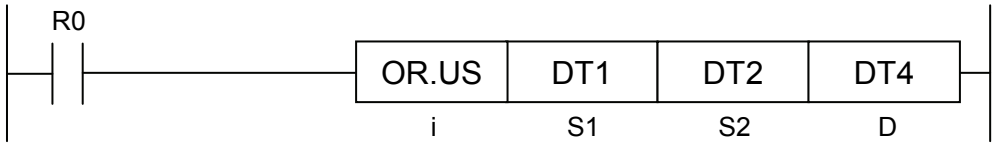
TS0	H 12345678	TS0	**H A0A0A0A0**
TS1	**H F0F0F0F0**	TS1	H F0F0F0F0
TS2	H 0F0F0F0F	TS2	H 0F0F0F0F
TS3	**H AAAAAAAA**	TS3	H AAAAAAAA
TS4	H 5AA5A55A	TS4	H 5AA5A55A

■ Flag operations

Name	Explanation
SR7 SR8 (ER)	To be set in the case of out-of-range in indirect access (index modification).

OR (Logical Disjunction)

■ Ladder diagram



■ Available operation units (●: Available)

Operation unit	bit	US	SS	UL	SL	SF	DF
i		●	●	●	●		

■ List of operands

Operand	Explanation
S1	Calculation target data 1 (device address or constant)
S2	Calculation target data 2 (device address or constant)
D	Calculation result data (device address)

■ Available devices (●: Available)

Operand	16-bit device											32-bit device *1			Integers			Real numbers		String	Index modifier *2
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX *3	K *4	U *5	H *6	SF	DF	" "	
S1	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●				●
S2	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●				●
D	●	●	●	●			●	●	●		●	●	●	●							●

*1: Operation unit: 16-bit integers (SS, US) cannot be specified.
*2: Only 16-bit devices, 32-bit devices, and integer constants can be modified (real number constants, and character constants cannot be specified).
*3: Index register (I0 to IE)
*4: Only operation unit: signed integers (SS, SL) can be specified.
*5: Only operation unit: unsigned integers (US, UL) can be specified.
*6: Only operation unit: integers (US, SS, UL, SL) can be specified.

■ Outline of operation

- Calculate the logical disjunction for [S1] and [S2] according to the operation unit [i].
- The calculation result is stored in the area starting with [D].

[S1] ∨ [S2] → [D]

■ Process details

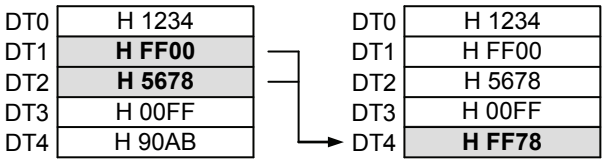
■ OR

[S1] bit	[S2] bit	OR
0	0	0
0	1	1
1	0	1
1	1	1

Example 1) Operation unit: 16 bits (US, SS)

[i]…US,SS

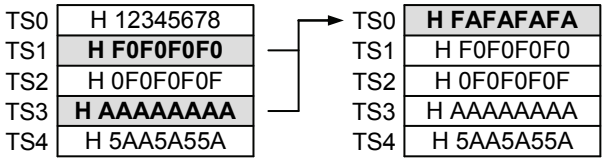
[S1]…DT1 [S2]…DT2 [D]…DT4



Example 2) Operation unit: 32 bits (UL, SL)

[i]…UL,SL

[S1]…TS1 [S2]…TS3 [D]…TS0

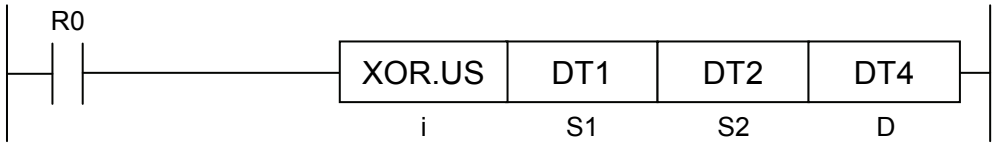


■ Flag operations

Name	Explanation
SR7 SR8 (ER)	To be set in the case of out-of-range in indirect access (index modification).

XOR (Exclusive OR)

■ Ladder diagram



■ Available operation units (●: Available)

Operation unit	bit	US	SS	UL	SL	SF	DF
i		●	●	●	●		

■ List of operands

Operand	Explanation
S1	Calculation target data 1 (device address or constant)
S2	Calculation target data 2 (device address or constant)
D	Calculation result data (device address)

■ Available devices (●: Available)

Operand	16-bit device											32-bit device *1			Integers			Real numbers		String	Index modifier *2
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX *3	K *4	U *5	H *6	SF	DF	" "	
S1	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●				●
S2	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●				●
D	●	●	●	●			●	●	●		●	●	●	●							●

*1: Operation unit: 16-bit integers (SS, US) cannot be specified.
*2: Only 16-bit devices, 32-bit devices, and integer constants can be modified (real number constants, and character constants cannot be specified).
*3: Index register (I0 to IE)
*4: Only operation unit: signed integers (SS, SL) can be specified.
*5: Only operation unit: unsigned integers (US, UL) can be specified.
*6: Only operation unit: integers (US, SS, UL, SL) can be specified.

■ Outline of operation

- Calculate the exclusive OR for [S1] and [S2] according to the operation unit [i].
- The calculation result is stored in the area starting with [D].

{ [S1] ∧ /[S2] } ∨ { /[S1] ∧ [S2] } → [D]

■ Process details

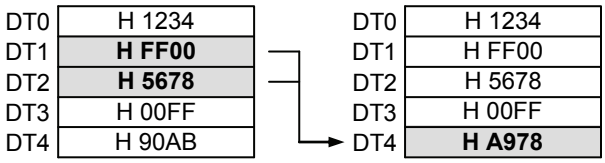
■ Exclusive OR (XOR)

[S1] bit	[S2] bit	Exclusive OR
0	0	0
0	1	1
1	0	1
1	1	0

Example 1) Operation unit: 16 bits (US, SS)

[i]…US,SS

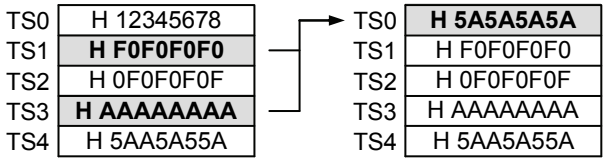
[S1]…DT1 [S2]…DT2 [D]…DT4



Example 2) Operation unit: 32 bits (UL, SL)

[i]…UL,SL

[S1]…TS1 [S2]…TS3 [D]…TS0

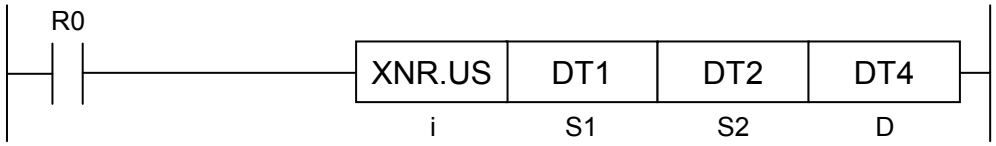


■ Flag operations

Name	Explanation
SR7 SR8 (ER)	To be set in the case of out-of-range in indirect access (index modification).

XNR (Exclusive NOR)

■ Ladder diagram



■ Available operation units (●: Available)

Operation unit	bit	US	SS	UL	SL	SF	DF
i		●	●	●	●		

■ List of operands

Operand	Explanation
S1	Calculation target data 1 (device address or constant)
S2	Calculation target data 2 (device address or constant)
D	Calculation result data (device address)

■ Available devices (●: Available)

Operand	16-bit device											32-bit device *1			Integers			Real numbers		String	Index modifier *2
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX *3	K *4	U *5	H *6	SF	DF	" "	
S1	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●				●
S2	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●				●
D	●	●	●	●			●	●	●		●	●	●	●							●

*1: Operation unit: 16-bit integers (SS, US) cannot be specified.
*2: Only 16-bit devices, 32-bit devices, and integer constants can be modified (real number constants, and character constants cannot be specified).
*3: Index register (I0 to IE)
*4: Only operation unit: signed integers (SS, SL) can be specified.
*5: Only operation unit: unsigned integers (US, UL) can be specified.
*6: Only operation unit: integers (US, SS, UL, SL) can be specified

■ Outline of operation

- Calculate the exclusive NOR for [S1] and [S2] according to the operation unit [i].
- The calculation result is stored in the area starting with [D].
- { [S1] ∧ [S2] } ∨ { /[S1] ∧ /[S2] } → [D]

■ Process details

■ Exclusive NOR (XNR)

[S1] bit	[S2] bit	Exclusive NOR
0	0	1
0	1	0
1	0	0
1	1	1

Example 1) Operation unit: 16 bits (US, SS)

[i]…US,SS

[S1]…DT1 [S2]…DT2 [D]…DT4

DT0	H 1234	DT0	H 1234
DT1	**H FF00**	DT1	H FF00
DT2	**H 5678**	DT2	H 5678
DT3	H 00FF	DT3	H 00FF
DT4	H 90AB	DT4	**H 5687**

Example 2) Operation unit: 32 bits (UL, SL)

[i]…UL,SL

[S1]…TS1 [S2]…TS3 [D]…TS0

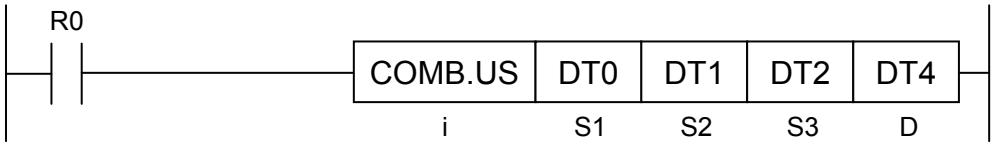
TS0	H 12345678	TS0	**H A5A5A5A5**
TS1	**H F0F0F0F0**	TS1	H F0F0F0F0
TS2	H 0F0F0F0F	TS2	H 0F0F0F0F
TS3	**H AAAAAAAA**	TS3	H AAAAAAAA
TS4	H 5AA5A55A	TS4	H 5AA5A55A

■ Flag operations

Name	Explanation
SR7 SR8 (ER)	To be set in the case of out-of-range in indirect access (index modification).

COMB (Combination)

■ Ladder diagram



■ Available operation units (●: Available)

Operation unit	bit	US	SS	UL	SL	SF	DF
i		●	●	●	●		

■ List of operands

Operand	Explanation
S1	Combined data 1 (device address or constant)
S2	Combined data 2 (device address or constant)
S3	Combination masked data (device address or constant)
D	Combination result data (device address)

■ Available devices (●: Available)

Operand	16-bit device											32-bit device *1			Integers			Real numbers		String	Index modifier *2
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX *3	K *4	U *5	H *6	SF	DF	" "	
S1	●	●	●	●			●	●	●	●	●	●	●	●	●	●	●				●
S2	●	●	●	●			●	●	●	●	●	●	●	●	●	●	●				●
S3	●	●	●	●			●	●	●	●	●	●	●	●	●	●	●				●
D	●	●	●	●			●	●	●		●	●	●	●							●

*1: Operation unit: 16-bit integers (SS, US) cannot be specified.
*2: Only 16-bit devices, 32-bit devices, and integer constants can be modified (real number constants, and character constants cannot be specified).
*3: Index register (I0 to IE)
*4: Only operation unit: signed integers (SS, SL) can be specified.
*5: Only operation unit: unsigned integers (US, UL) can be specified.
*6: Only operation unit: integers (US, SS, UL, SL) can be specified.

■ Outline of operation

- Combine the data for [S1] and [S2] according to the operation unit [i] and the masked data stored in [S3].
- The calculation result is stored in the area starting with [D].
- The combination target should start with [S1] for bits where the masked data specified by [S3] are ON, and [S2] where the bits are OFF.

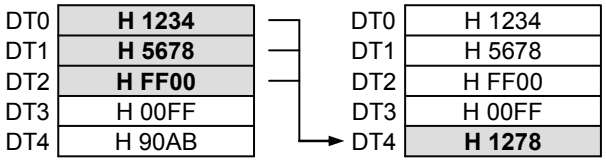
{ [S1] ∧ [S3] } ∨ { [S2] ∧ /[S3] } → [D]

■ Process details

Example 1) Operation unit: 16 bits (US, SS)

[i]…US,SS

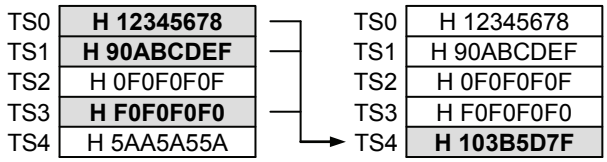
[S1]…DT0 [S2]…DT1 [S3]…DT2 [D]…DT4



Example 2) Operation unit: 32 bits (UL, SL)

[i]…UL,SL

[S1]…TS0 [S2]…TS1 [S3]…TS3 [D]…TS4

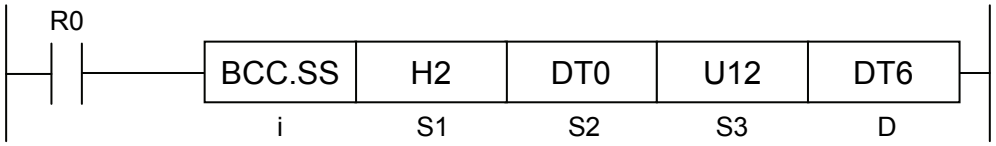


■ Flag operations

Name	Explanation
SR7 SR8 (ER)	To be set in the case of out-of-range in indirect access (index modification).

BCC (Block Check Code Calculation)

■ Ladder diagram



■ Available operation units (●: Available)

Operation unit	bit	US	SS	UL	SL	SF	DF
i		●	●				

■ List of operands

Operand	Explanation
S1	Area that stores the data specifying a calculation method, or constant data
S2	Starting address of the area that stores the targeted data
S3	Area that stores the length (no. of bytes) of the targeted data, or constant data
D	Area that stores the calculation result

■ Available devices (●: Available)

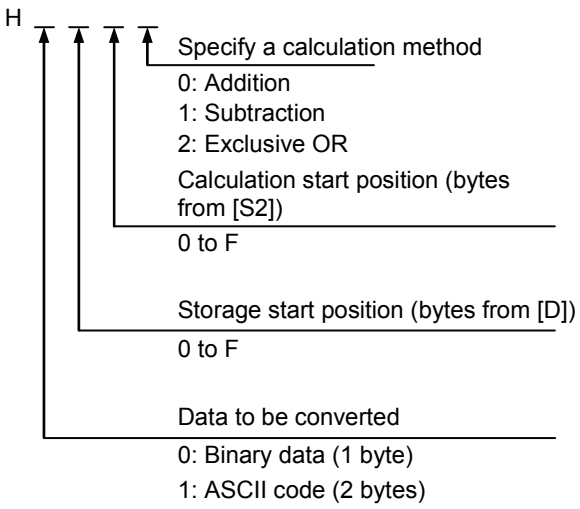
Operand	16-bit device											32-bit device *1			Integers			Real numbers		String	Index modifier *2
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX *3	K *4	U *5	H *6	SF	DF	" "	
S1	●	●	●	●			●	●	●	●	●				●	●	●				●
S2	●	●	●	●			●	●	●	●	●										●
S3	●	●	●	●			●	●	●	●	●					●	●				●
D	●	●	●	●			●	●	●		●										●

*1: Only 16-bit devices, 32-bit devices, and integer constants can be modified (real number constants, and character constants cannot be specified).
*2: Only operation unit: signed integers (SS) can be specified.
*3: Only operation unit: unsigned integers (US) can be specified.
*4: Only operation unit: integers (US, SS) can be specified.

■ Outline of operation

- Calculate the block check code (BCC).
- Calculate the block check code (BCC) for the targeted data that correspond to the no. of bytes specified by [S3], starting from the calculation start position specified by [S2], in the calculation method specified by [S1].
- The calculation result is stored according to the conversion method specified by [S1], starting from the storage position specified by [D] and [S1].

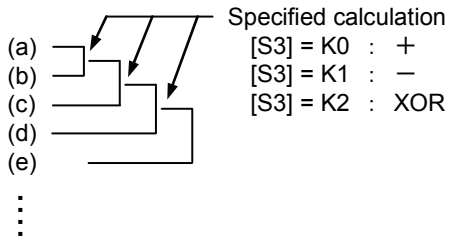
■ Specification of control data [S1]



As indicated below, calculation should be carried out as specified, for every eight bits.

%	→	H 25	→	00100101	(a)
0	→	H 30	→	00110000	(b)
1	→	H 31	→	00110001	(c)
#	→	H 23	→	00100011	(d)
R	→	H 52	→	01010010	(e)

⋮

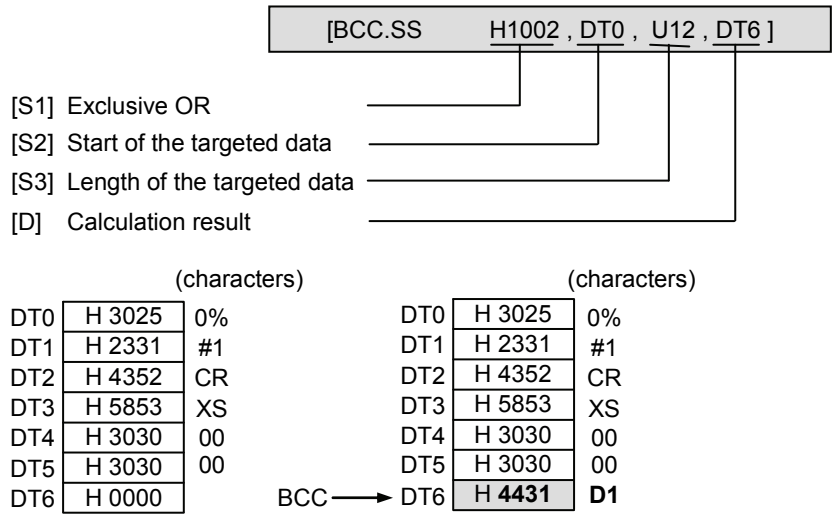


⋮

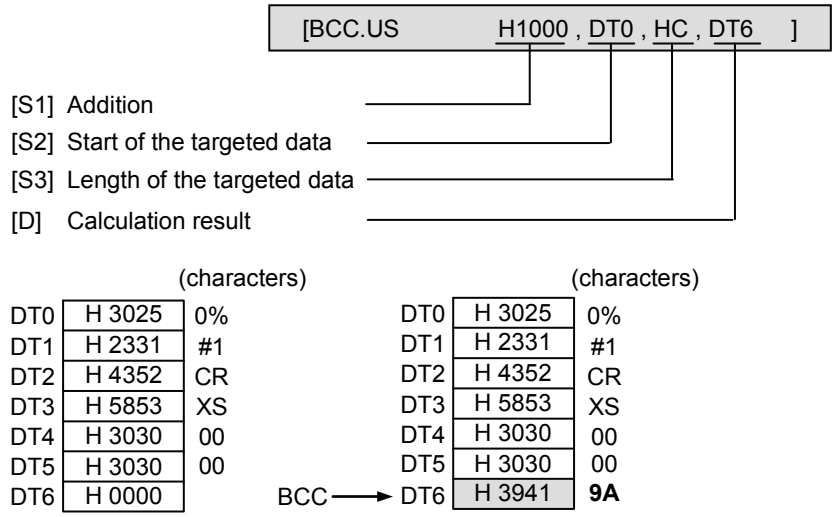
■ Example of conversion

- Calculate the block check code (BCC) for a message to be sent "%01#RCSX0000", and append the result to the message.

Example 1) Operation unit: 16 bits (SS) / Calculation method: Exclusive OR; Data to be converted: ASCII code



Example 2) Operation unit: 16 bits (US) / Calculation method: Addition; Data to be converted: ASCII code



■ Flag operations

Name	Explanation
SR7 SR8 (ER)	To be set in the case of out-of-range in indirect access (index modification).

CRC (CRC Code Calculation)

■ Ladder diagram

■ Available operation units (●: Available)

Operation unit	bit	US	SS	UL	SL	SF	DF
i		●	●				

■ List of operands

Operand	Explanation
S1	Area that stores the data specifying a calculation method, or constant data
S2	Starting address of the area that stores the targeted data
S3	Area that stores the length (no. of bytes) of the targeted data, or constant data
D	Area that stores the calculation result

■ Available devices (●: Available)

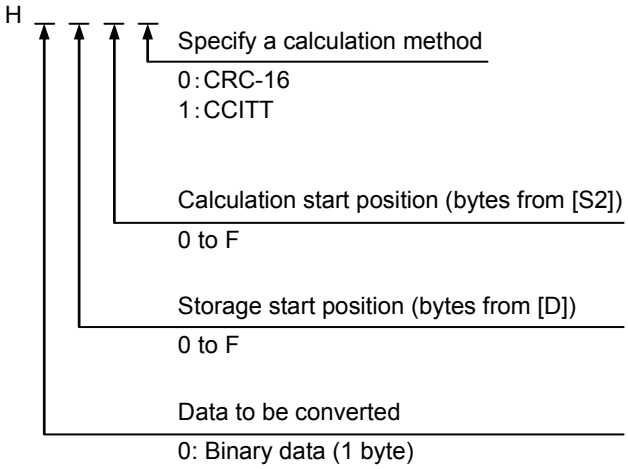
Operand	16-bit device											32-bit device			Integers			Real numbers		String	Index modifier *1
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX	K *2	U *3	H *4	SF	DF	" "	
S1	●	●	●	●			●	●	●	●	●				●	●	●				●
S2	●	●	●	●			●	●	●	●	●										●
S3(*5)	●	●	●	●			●	●	●	●	●					●	●				●
D	●	●	●	●			●	●	●		●										●

*1: Only 16-bit devices, 32-bit devices, and integer constants can be modified (real number constants, and character constants cannot be specified).
*2: Only operation unit: signed integers (SS) can be specified.
*3: Only operation unit: unsigned integers (US) can be specified.
*4: Only operation unit: integers (US, SS) can be specified.
*5: To be handled as a 16-bit unsigned integer (US), regardless of operation unit.

■ Outline of operation

- Calculate the block check code (BCC) in a CRC (16-bit CRC code) calculation form.
- Calculate the block check code (BCC), in a CRC calculation form, for the targeted data that correspond to the no. of bytes specified by [S3], starting from the calculation start position specified by [S2], in the calculation method specified by [S1].
- The calculation result is stored according to the conversion method specified by [S1], starting from the storage position specified by [D] and [S1].

■ Specification of control data [S1]



- Apply a specified calculation method to every 16 bits as follows.
- When the calculation form is specified as CRC, calculation should be carried out using the following generating polynomial.

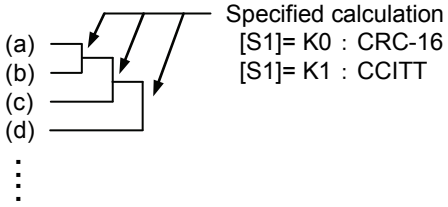
(This is the same calculation method as MODBUS-RTU.)

・Generating polynomial: (CRC-16) $X^{16} + X^{15} + X^2 + 1$

・Generating polynomial: (CRCITT) $X^{16} + X^{12} + X^5 + 1$

%	→	H 25	→	00100101	(a)
0	→	H 30	→	00110000	(b)
1	→	H 31	→	00110001	(c)
#	→	H 23	→	00100011	(d)
R	→	H 52	→	01010010	(e)

⋮

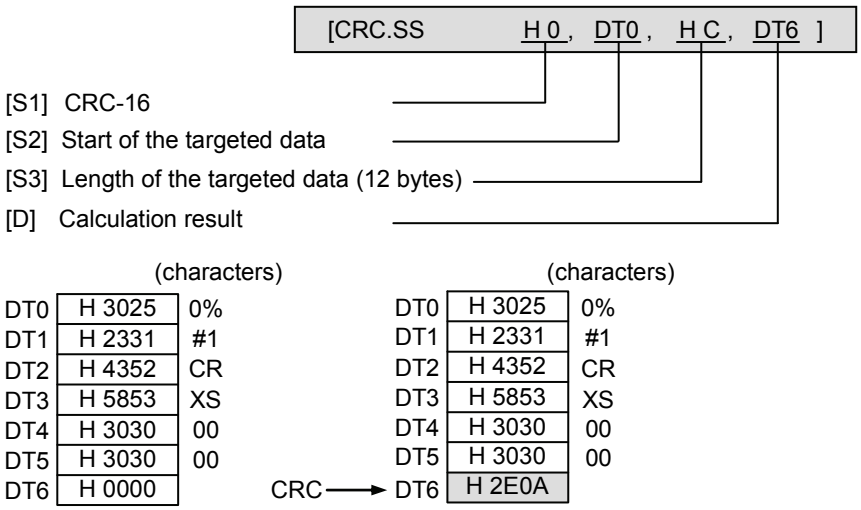


⋮

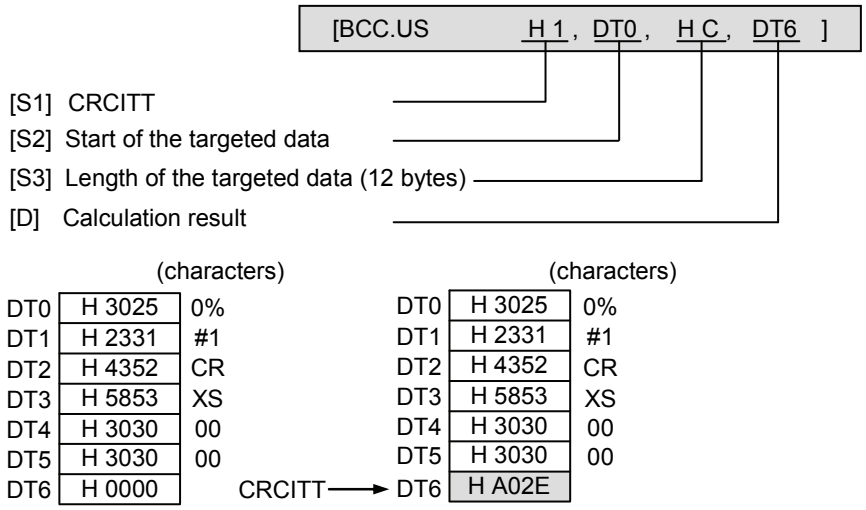
■ Example of conversion

- Calculate the block check code for a message to be sent "%01#RCSX0000", and append the result to the message.

Example 1) Operation unit: 16 bits (SS) / Calculation method: CRC-16; Data to be converted: Binary data



Example 2) Operation unit: 16 bits (US) / Calculation method: CRCITT; Data to be converted: Binary data

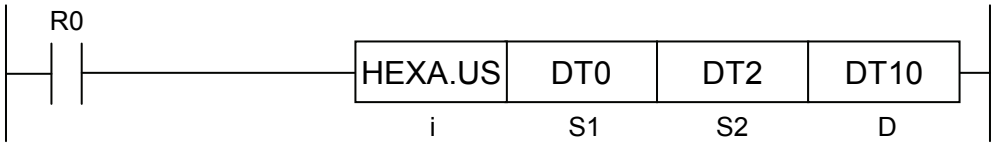


■ Flag operations

Name	Explanation
SR7	To be set in the case of out-of-range in indirect access (index modification).
SR8	Calculation method specified by [S1] is out of the specified range.
(ER)	Conversion data specified by [S1] is out of the specified range.

HEXA (Conversion: HEX → Hexadecimal ASCII)

■ Ladder diagram



■ Available operation units (●: Available)

Operation unit	bit	US	SS	UL	SL	SF	DF
i		●	●	●	●		

■ List of operands

Operand	Explanation
S1	Starting number of the area that stores a hexadecimal figure
S2	Area that stores the length (no. of bytes) to be converted, or constant
D	Starting number of the area that stores the ASCII code as the conversion result

■ Available devices (●: Available)

Operand	16-bit device											32-bit device			Integers			Real numbers		String	Index modifier
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX	K *2	U *3	H *4	SF	DF	" "	*1
S1	●	●	●	●	●	●	●	●													●
S2	●	●	●	●			●	●							●	●	●				●
D	●	●	●	●			●	●													●

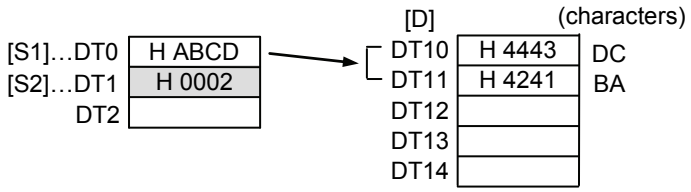
*1: Only 16-bit devices, 32-bit devices, and integer constants can be modified (real number constants, and character constants cannot be specified).
*2: Only operation unit: signed integers (SS, SL) can be specified.
*3: Only operation unit: unsigned integers (US, UL) can be specified.
*4: Only operation unit: integers (US, SS, UL, SL) can be specified.

■ Outline of operation

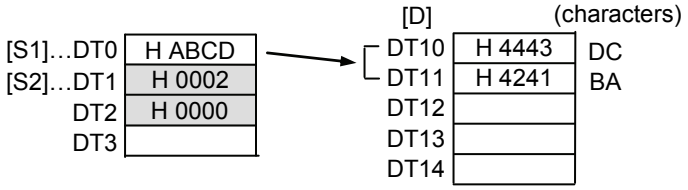
- Convert a hexadecimal figure into an ASCII code.
- Convert the hexadecimal numerical data stored in the area specified by [S1] into an ASCII code, and store the ASCII code in the area specified by [D].
- The number of bytes specified by [S2] are to be converted.
- The resulting ASCII code should have twice the size of the source data.

■ Example of conversion

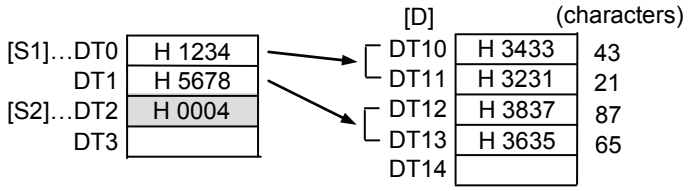
Example 1) Operation unit: 16 bits (US, SS)
[i]…US,SS
[S1]…DT0 [S2]…DT1
[D]…DT10



Example 2) Operation unit: 32 bits (UL, SL)
[i]…UL,SL
[S1]…DT0 [S2]…DT1
[D]…DT10

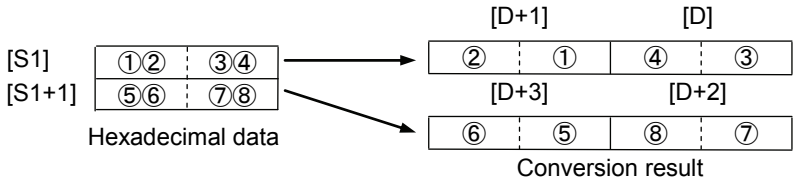


Example 3) Operation unit: 16 bits (US, SS)
[i]…US,SS
[S1]…DT0 [S2]…DT2
[D]…DT10



■ Precautions during programming

- The two characters that comprise one byte are stored in the opposite order following conversion.
- Conversion is carried out with two bytes as one unit.

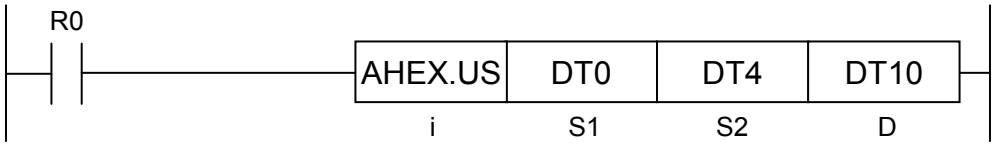


■ Flag operations

Name	Explanation
SR7 SR8 (ER)	To be set in the case of out-of-range in indirect access (index modification).
	To be set when the no. of bytes for [S2] is specified, and the conversion range exceeds the area.
	To be set when the conversion result exceeds the area.
	To be set when [S2] is specified as '0' or a negative value.

AHEX (Conversion: Hexadecimal ASCII → HEX)

■ Ladder diagram



■ Available operation units (●: Available)

Operation unit	bit	US	SS	UL	SL	SF	DF
i		●	●	●	●		

■ List of operands

Operand	Explanation
S1	Starting number of the area that stores an ASCII code
S2	Area that stores the no. of ASCII codes (no. of characters) to be converted, or constant
D	Starting number of the area that stores the hexadecimal figure as the conversion result

■ Available devices (●: Available)

Operand	16-bit device											32-bit device			Integers			Real numbers		String	Index modifier *1
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX	K *2	U *3	H *4	SF	DF	" "	
S1	●	●	●	●	●	●	●	●													●
S2	●	●	●	●			●	●							●	●	●				●
D	●	●	●	●			●	●													●

*1: Only 16-bit devices, 32-bit devices, and integer constants can be modified (real number constants, and character constants cannot be specified).
*2: Only operation unit: signed integers (SS, SL) can be specified.
*3: Only operation unit: unsigned integers (US, UL) can be specified.
*4: Only operation unit: integers (US, SS, UL, SL) can be specified.

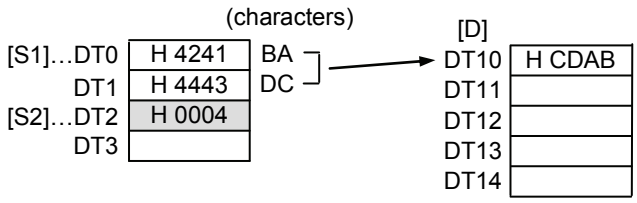
■ Outline of operation

- Convert the ASCII code string into a hexadecimal figure.
- Convert the ASCII code stored in the area specified by [S1] into a hexadecimal figure, and store the converted figure in the area specified by [D].
- The no. of ASCII codes (no. of characters) to be converted is specified by [S2].
- The conversion result is stored in bytes.
- If the no. of characters to be converted is an odd number, Bits 0 to 3 of the final data (bytes) of the conversion result are padded with '0'.

■ Example of conversion

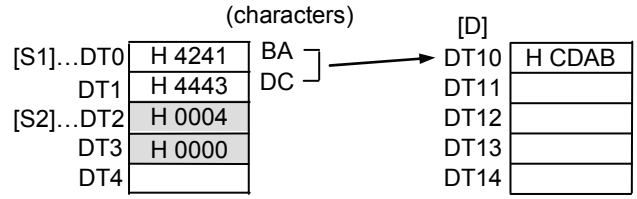
Example 1) Operation unit: 16 bits (US, SS)

[i]…US,SS
[S1]…DT0 [S2]…DT2
[D]…DT10



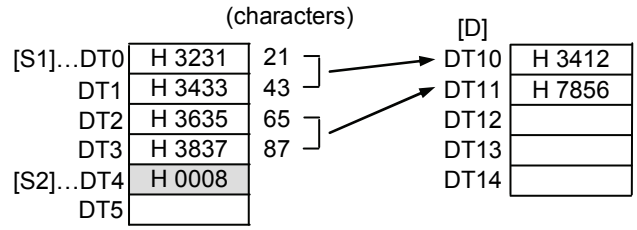
Example 2) Operation unit: 32 bits (UL, SL)

[i]…UL,SL
[S1]…DT0 [S2]…DT2
[D]…DT10



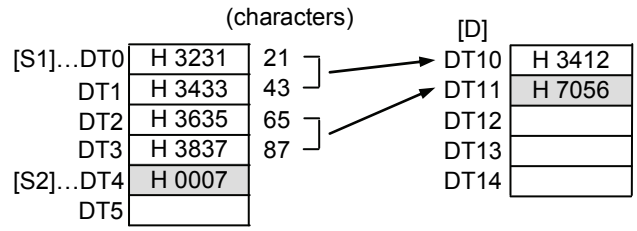
Example 3) Operation unit: 16 bits (US, SS)

[i]…US,SS
[S1]…DT0 [S2]…DT4
[D]…DT10



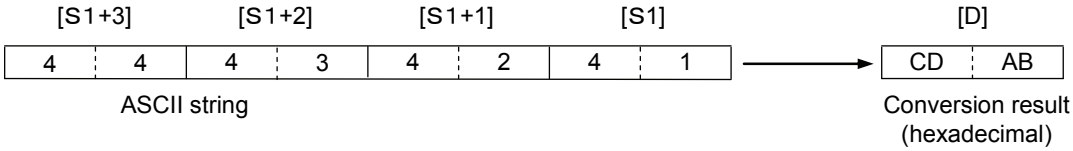
Example 4) Operation unit: 16 bits (US, SS); No. of characters to be converted is an odd number

[i]…US,SS
[S1]…DT0 [S2]…DT4
[D]…DT10



■ Precautions during programming

- Two characters of ASCII code are converted into two one-byte numerical digits. In this process, higher-level characters and lower-level characters are exchanged.
- Conversion is carried out with four characters as one unit.

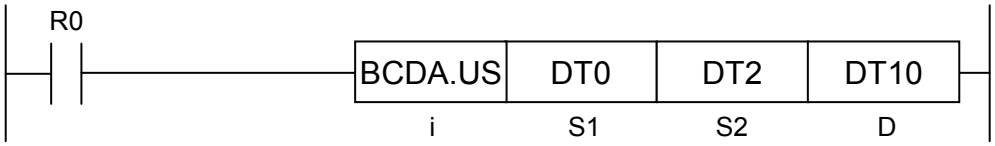


■ Flag operations

Name	Explanation
SR7 SR8 (ER)	To be set in the case of out-of-range in indirect access (index modification).
	To be set when the no. of characters for [S2] is specified, and the conversion range exceeds the area.
	To be set when the conversion result exceeds the area.
	To be set when [S2] is specified as '0' or a negative value.
	To be set when the ASCII codes specified by [S1] contains character codes other than 0 to F.

BCDA (Conversion: BCD → Decimal ASCII)

■ Ladder diagram



■ Available operation units (●: Available)

Operation unit	bit	US	SS	UL	SL	SF	DF
i		●		●			

■ List of operands

Operand	Explanation
S1	Starting number of the area that stores a BCD figure
S2	Area that stores the data that express data size to be converted or conversion direction, or constant
D	Starting number of the area that stores the ASCII code as the conversion result

■ Available devices (●: Available)

Operand	16-bit device											32-bit device			Integers			Real numbers		String	Index modifier
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX	K	U	H	SF	DF	" "	*1
S1	●	●	●	●	●	●	●	●													●
S2	●	●	●	●			●	●								●	●				●
D	●	●	●	●			●	●													●

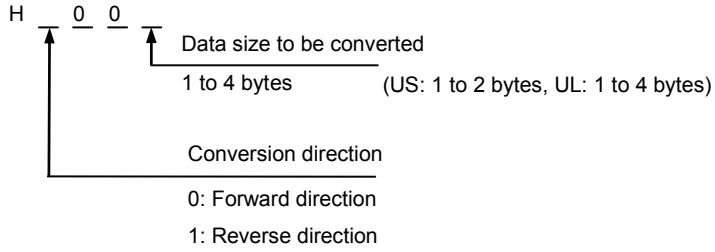
*1: Only 16-bit devices, 32-bit devices, and integer constants can be modified (real number constants, and character constants cannot be specified).

■ Outline of operation

- Up to eight-digit BCD data are converted into an ASCII code string.
- Convert the BCD figure stored in the area specified by [S1] into ASCII codes.
- The conversion result is stored in the area starting with [D].
- The BCD data size (no. of bytes) to be converted, and the conversion direction, are specified by [S2].
- The resulting ASCII code should have twice the size of the source data.
- The max. value of data size to be converted varies by operation unit. (US: 2 bytes, UL: 4 bytes)
- Because data size to be converted is specified by bytes, it is possible to convert only lower bytes of one-word data.

■ Setting conversion data size and conversion direction [S2]

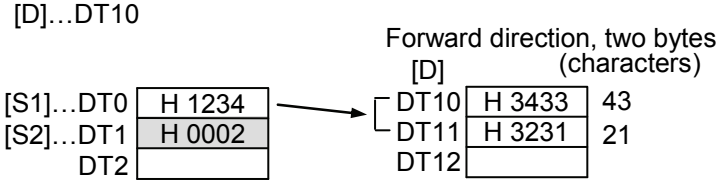
Specify the value with a four-digit H constant.



■ Example of conversion

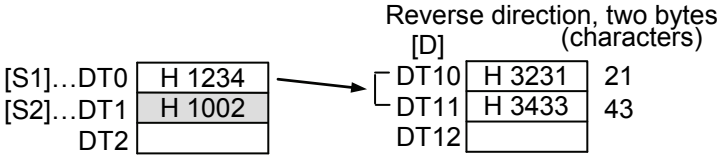
Example 1) Operation unit: 16 bits (US)

[i]…US
[S1]…DT0 [S2]…DT1
[D]…DT10



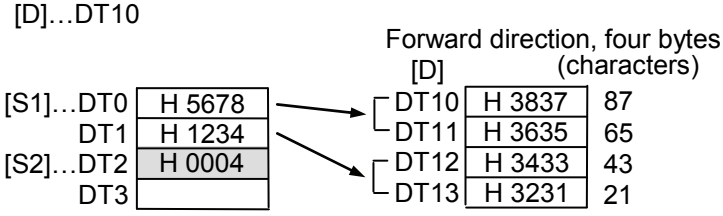
Example 2) Operation unit: 16 bits (US)

[i]…US
[S1]…DT0 [S2]…DT1
[D]…DT10



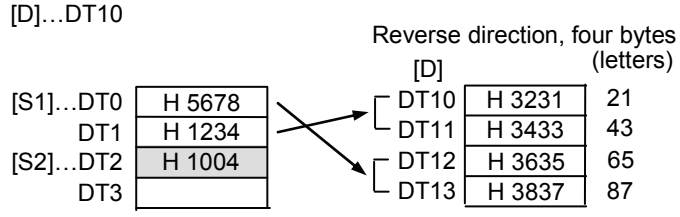
Example 3) Operation unit: 32 bits (UL)

[i]…UL
[S1]…DT0 [S2]…DT2
[D]…DT10



Example 4) Operation unit: 32 bits (UL)

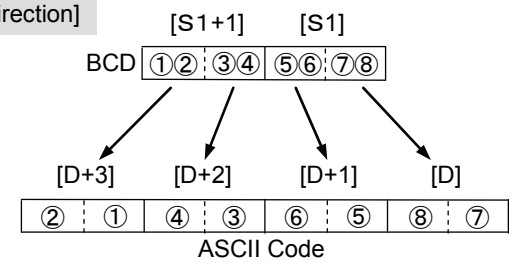
[i]…UL
[S1]…DT0 [S2]…DT2
[D]…DT10



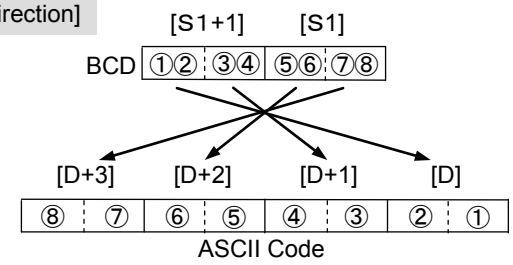
■ Precautions during programming

- The two characters that comprise one byte are stored in the opposite order following conversion.
- Conversion is carried out with two bytes as one unit.

[Forward direction]



[Reverse direction]

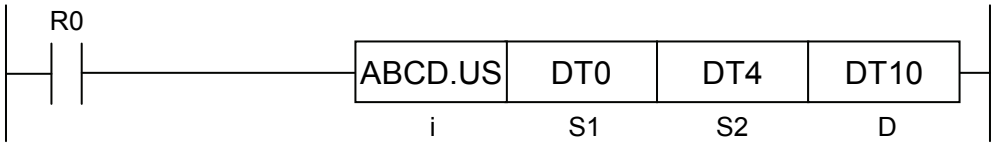


■ Flag operations

Name	Explanation
SR7 SR8 (ER)	To be set in the case of out-of-range in indirect access (index modification).
	To be set when the data specified by [S1] contain non-BCD data.
	To be set when the no. of bytes specified by [S2] exceeds the [S1] area.
	To be set when the conversion result exceeds the area.
	To be set when [S2] is specified as '0'.
	To be set when the conversion direction of [S2] is out of the range.
	To be set when the conversion data size of [S2] is out of the range.

ABCD (Conversion: Decimal ASCII → BCD)

■ Ladder diagram



■ Available operation units (●: Available)

Operation unit	bit	US	SS	UL	SL	SF	DF
i		●		●			

■ List of operands

Operand	Explanation
S1	Starting number of the area that stores an ASCII code
S2	Area that stores the data that express the no. of ASCII codes to be converted and conversion direction, or constant
D	Starting number of the area that stores the BCD figure as the conversion result

■ Available devices (●: Available)

Operand	16-bit device											32-bit device			Integers			Real numbers		String	Index modifier
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX	K	U	H	SF	DF	" "	*1
S1	●	●	●	●	●	●	●	●													●
S2	●	●	●	●			●	●								●	●				●
D	●	●	●	●			●	●													●

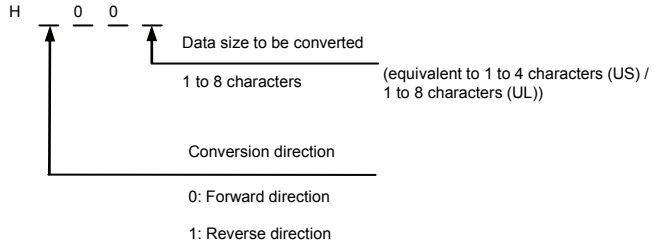
*1: Only 16-bit devices, 32-bit devices, and integer constants can be modified (real number constants, and character constants cannot be specified).

■ Outline of operation

- Up to eight-character ASCII code string is converted into BCD data.
- Convert the ASCII code stored in the area specified by [S1] into BCD data, and store the converted data in the area specified by [D].
- The no. of ASCII codes (no. of characters) to be converted and conversion direction are specified by [S2].
- The resulting BCD data should have half the size of the source ASCII code string.
- The max. value of data size to be converted varies by operation unit. (US: 4 characters, UL: 8 characters)
- Because data size to be converted is specified by bytes, it is possible to convert only lower bytes of one-word data.
- If the data size to be converted is an odd number, bits of the final data of the conversion result are padded with '0'.
  i) Bits 0 to 3 are padded with '0'. (Forward direction)
  ii) Bits 4 to 7 are padded with '0'. (Reverse direction)

■ Setting conversion data size and conversion direction [S2]

According to the following format, specify the setting in a four-digit BCD (H constant).



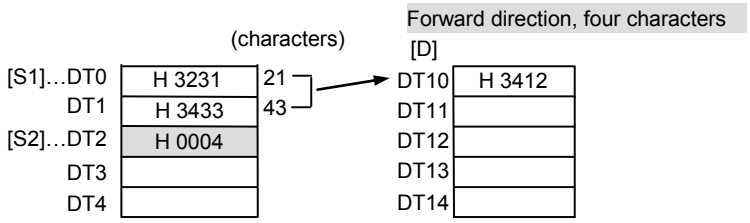
■ Example of conversion

Example 1) Operation unit: 16 bits (US)

[i]…US

[S1]…DT0 [S2]…DT2

[D]…DT10

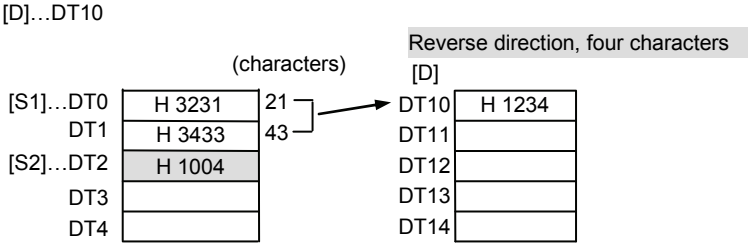


Example 2) Operation unit: 16 bits (US)

[i]…US

[S1]…DT0 [S2]…DT2

[D]…DT10



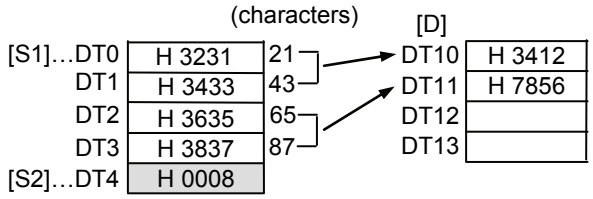
Example 3) Operation unit: 32 bits (UL)

[i]…UL

[S1]…DT0 [S2]…DT4

[D]…DT10

Forward direction, eight characters



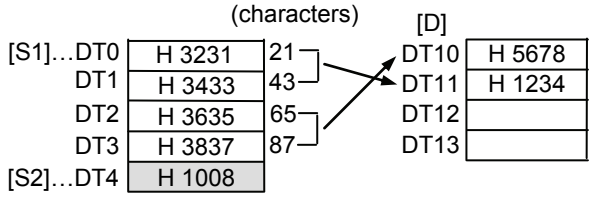
Example 4) Operation unit: 32 bits (UL)

[i]…UL

[S1]…DT0 [S2]…DT4

[D]…DT10

Reverse direction, eight characters



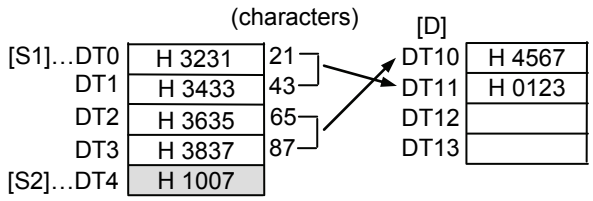
Example 5) Operation unit: 32 bits (UL)

[i]…UL

[S1]…DT0 [S2]…DT4

[D]…DT10

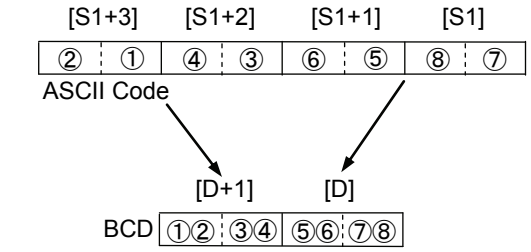
Reverse direction, seven characters



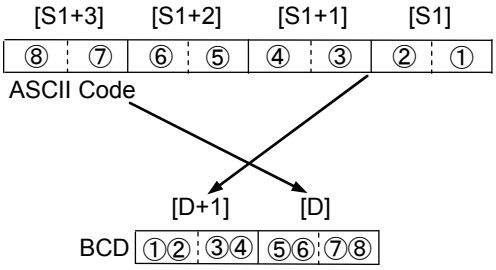
■ Precautions during programming

- The two characters that comprise one byte are stored in the opposite order following conversion.
- Conversion is carried out with two bytes as one unit.

[Forward direction]



[Reverse direction]

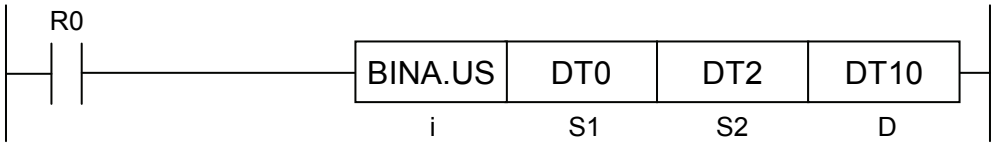


■ Flag operations

Name	Explanation
SR7 SR8 (ER)	To be set in the case of out-of-range in indirect access (index modification).
	To be set when the ASCII codes specified by [S1] contain data other than 0 to 9.
	To be set when the no. of bytes specified by [S2] exceeds the [S1] area.
	To be set when the conversion result exceeds the area.
	To be set when [S2] is specified as '0'.
	To be set when the conversion direction of [S2] is out of the range.
	To be set when the conversion data size of [S2] is out of the range.

BINA (Conversion: BIN → Decimal ASCII)

■ Ladder diagram



■ Available operation units (●: Available)

Operation unit	bit	US	SS	UL	SL	SF	DF
i		●	●	●	●		

■ List of operands

Operand	Explanation
S1	Area that stores the BIN data that express a decimal figure, or constant data
S2	Area that stores the no. of bytes of the area to store the conversion result, or constant data
D	Starting number of the area that stores the ASCII code as the conversion result

■ Available devices (●: Available)

Operand	16-bit device											32-bit device			Integers			Real numbers		String	Index modifier *1
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX	K *2	U *3	H *4	SF	DF	" "	
S1	●	●	●	●	●	●	●	●							●	●	●				●
S2	●	●	●	●			●	●							●	●	●				●
D	●	●	●	●			●	●													●

*1: Only 16-bit devices, 32-bit devices, and integer constants can be modified (real number constants, and character constants cannot be specified).
*2: Only operation unit: signed integers (SS, SL) can be specified.
*3: Only operation unit: unsigned integers (US, UL) can be specified.
*4: Only operation unit: integers (US, SS, UL, SL) can be specified.

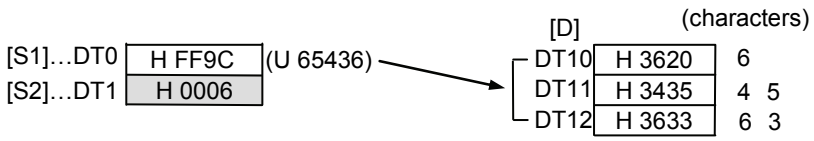
■ Outline of operation

- BIN data that express a decimal figure are converted into an ASCII code string.
- Convert the BIN data expressed as a decimal figure specified by [S1] into an ASCII code, and store the ASCII code in the area specified by [D].
- The starting position of the storage area is specified by [D], and the size by [S2].
- If the no. of bytes of the resulting ASCII code (including the negative sign) is larger than the no. of bytes specified by [S2], an operation error occurs.
- Specify the no. of digits of the conversion target, including the sign, by [S2].

■ Example of conversion

Example 1) Operation unit: 16 bits (US)

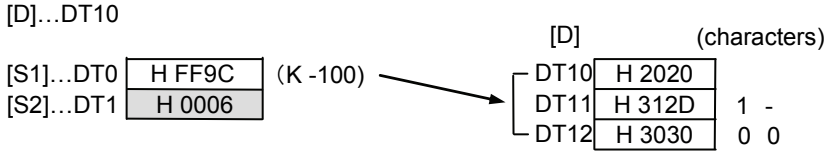
[i]…US
[S1]…DT0 [S2]…DT1
[D]…DT10



DT0: H FF9C (U 65436) → DT10 to DT12: ” 65436”

Example 2) Operation unit: 16 bits (SS)

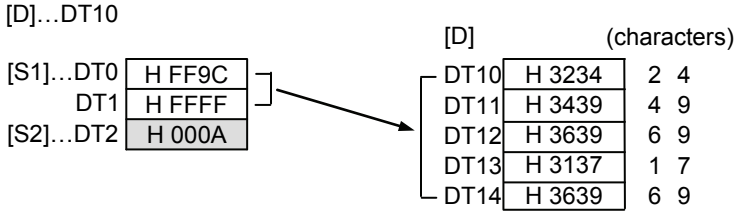
[i]…SS
[S1]…DT0 [S2]…DT1
[D]…DT10



DT0: H FF9C (K -100) → DT10 to DT12: ” -100”

Example 3) Operation unit: 32 bits (UL)

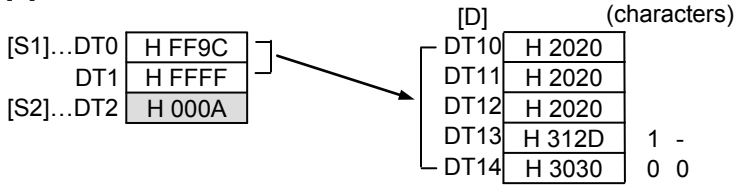
[i]…UL
[S1]…DT0 [S2]…DT2
[D]…DT10



DT0: H FFFF FF9C (U 4294967196) → DT10 to DT14: ” 4294967196”

Example 4) Operation unit: 32 bits (SL)

[i]…SL
[S1]…DT0 [S2]…DT2
[D]…DT10



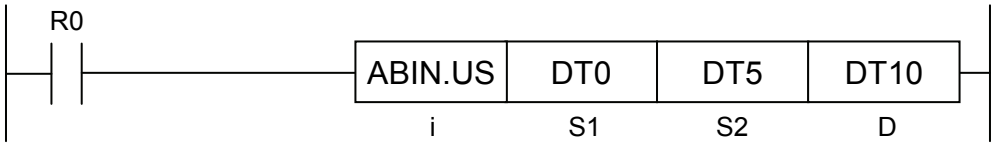
DT0: H FFFF FF9C (K -100) → DT10 to DT14: ” -100”

■ Flag operations

Name	Explanation
SR7 SR8 (ER)	To be set in the case of out-of-range in indirect access (index modification).
	To be set when the no. of bytes specified by [S2] exceeds the [D] area.
	To be set when the conversion result exceeds the area.
	To be set when the resulting no. of bytes exceeds the no. of bytes specified by [S2].

ABIN (Conversion: Decimal ASCII → BIN)

■ Ladder diagram



■ Available operation units (●: Available)

Operation unit	bit	US	SS	UL	SL	SF	DF
i		●	●	●	●		

■ List of operands

Operand	Explanation
S1	Starting number of the area that stores the ASCII code to be converted
S2	Area that stores the no. of bytes to be converted, or constant data
D	Area that stores the conversion result

■ Available devices (●: Available)

Operand	16-bit device											32-bit device			Integers			Real numbers		String	Index modifier *1
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX	K *2	U *3	H *4	SF	DF	" "	
S1	●	●	●	●	●	●	●	●													●
S2(*5)	●	●	●	●			●	●							●	●	●				●
D	●	●	●	●			●	●													●

*1: Only 16-bit devices, 32-bit devices, and integer constants can be modified (real number constants, and character constants cannot be specified).
*2: Only operation unit: signed integers (SS, SL) can be specified.
*3: Only operation unit: unsigned integers (US, UL) can be specified.
*4: Can be specified only when the operation unit is integer (US, SS, UL, SL).
*5: To be handled as a 16-bit integer (US, SS), regardless of operation unit.

■ Outline of operation

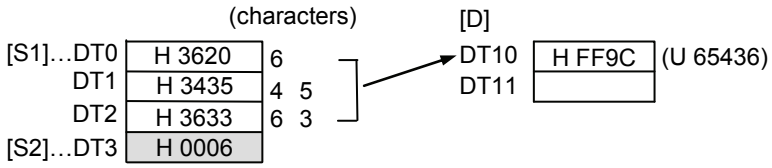
- Convert the ASCII code string into BIN data that express a decimal figure.
- Convert the ASCII code that express a decimal figure equivalent to the no. of bytes (i.e. no. of characters) specified by [S2], starting from the area specified by [S1], into a decimal figure, and store the figure in the area specified by [D].
- If a negative sign (-) is contained in the ASCII code specified by [S1], specify SS or SL for the operation unit. An operation error occurs in the case of operation unit: US or UL.

■ Example of conversion

Example 1) Operation unit: 16 bits (US)

[i]…US

[S1]…DT0 [S2]…DT3 [D]…DT10

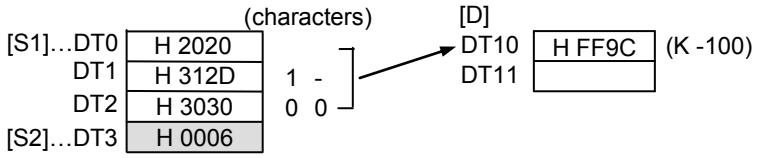


DT0 to DT2: "65436" → DT10: H FF9C (U 65436)

Example 2) Operation unit: 16 bits (SS)

[i]…SS

[S1]…DT0 [S2]…DT3 [D]…DT10

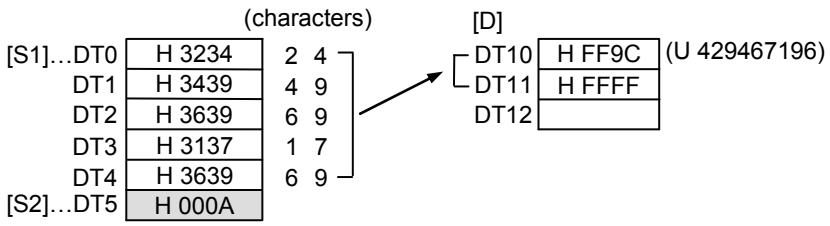


DT0 to DT2: "-100" → DT10: H FF9C (K -100)

Example 3) Operation unit: 32 bits (UL)

[i]…UL

[S1]…DT0 [S2]…DT5 [D]…DT10

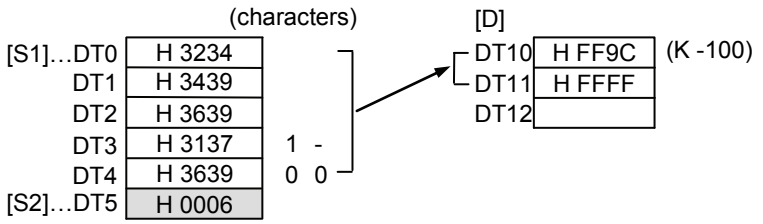


DT0 to DT4: "4294967196" → DT10 to DT11: H FFFF FF9C (U 4294967196)

Example 4) Operation unit: 32 bits (SL)

[i]…SL

[S1]…DT0 [S2]…DT5 [D]…DT10



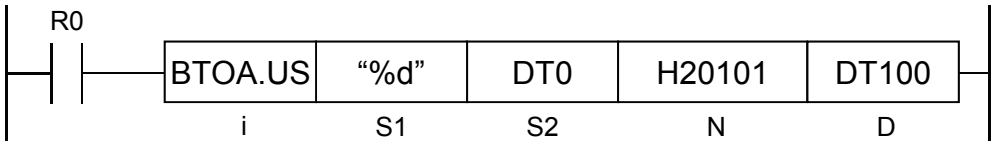
DT0 to DT4: "-100" → DT10 to DT11: H FFFF FF9C (K -100)

■ Flag operations

Name	Explanation
SR7 SR8 (ER)	To be set in the case of out-of-range in indirect access (index modification).
	To be set when the no. of bytes specified by [S2] exceeds the [D] area.
	To be set when the conversion result exceeds the area.
	To be set when [S1] contains character ASCII codes other than 0 to 9, sign code, or space.

BTOA (Conversion: BIN → ASCII)

■ Ladder diagram



■ Available operation units (●: Available)

Operation unit	bit	US	SS	UL	SL	SF	DF
i		●	●	●	●	●	●

■ List of operands

Operand	Explanation
S1	Control string (2 to 16 characters)
S2	Starting number of the area that stores binary data
N	Conversion method
D	Starting number of the area that stores the ASCII code as the conversion result

■ Available devices (●: Available)

Operand	16-bit device											32-bit device			Integers			Real numbers		String	Index modifier *1
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX	K	U	H	SF	DF	" "	
S1	●	●	●	●			●	●												●	●
S2	●	●	●	●	●	●	●	●													●
n(*2)	●	●	●	●			●	●								●	●				●
D	●	●	●	●			●	●													●

*1: Only 16-bit devices, 32-bit devices, and integer constants can be modified (real number constants, and character constants cannot be specified).

*2: To be handled as a 32-bit integer (UL), regardless of operation unit.

■ Outline of operation

- Convert the binary data stored in the area starting with [S2] into ASCII codes, in a conversion method specified by [S1] and [N].
- The conversion result is stored in the area starting with [D].

■ Setting of control strings [S1]

Control string	Format of data to be converted	Available operation unit	Example
"%d" or "%i"	Convert 16-bit data into decimal ASCII data (Signed integer)	SS	"%d" , "%5d" , "%+5d" , "%-5d" "%05d" , "%10.5d" , "%d," "% d"
	Convert 32-bit data into decimal ASCII data (Signed integer)	SL	
"%u"	Convert 16-bit data into decimal ASCII data (Unsigned integer)	US	"%u" , "%5u" , "%+5u" , "%-5u" "%05u" , "%10.5u" , "%u,"
	Convert 32-bit data into decimal ASCII data (Unsigned integer)	UL	
"%x"	Convert 16-bit data into hexadecimal ASCII data (Forward/reverse direction)	US	"%x" , "%5x" , "%-5x" , "%05x" "%10.5x" , "%x," , "%#x" , "%X"
	Convert 32-bit data into hexadecimal ASCII data (Forward/reverse direction)	UL	
"%b"	Convert 16-bit BCD data into decimal ASCII data (Forward/reverse direction)	US	"%b" , "%5b" , "%-5b" , "%05b" "%10.5b" , "%b,"
	Convert 32-bit BCD data into decimal ASCII data (Forward/reverse direction)	UL	
"%f"	Convert 32-bit single precision real number data into floating point ASCII data	SF	"%f" , "%5.2f" , "%+5.2f" "%-5.2f" , "%05.2f" , "%f," "%#f" , "% f"
"%e"	Convert 32-bit single precision real number data into exponential ASCII data	SF	"%e" , "%5.2e" , "%+5.2e" "%-5.2e" , "%05.2e" , "%e," "%#5.2e" , "% e" , "%E"
"%s"	Convert 32-bit single precision real number data into exponential ASCII data, or floating point ASCII data (whichever is shorter in the relevant notation)	SF	"%g" , "%5.2g" , "%+5.2g" "%-5.2g" , "%05.2g" , "%g," "%#5.2g" , "%G"

■ Format of control strings [S1]

1) Setting of alphabetical upper case or lower case

Specify whether alphabetical upper case or lower case should be used for the hexadecimal or exponential ASCII data.

Control string	Binary data	ASCII data
%x	H ABCD	"abcd"
%X	H ABCD	"ABCD"
%e	1234.5678	"1.2345678e+3"
%E	1234.5678	"1.2345678E+3"

2) Setting of no. of notation digits

No. of notation digits should be specified by the total no. of characters and the no. of precision characters.

Unless specified, the default values are used.

'n.m', 'n', '.m', etc. are used for specification.

n: total no. of characters; m: no. of precision characters

< No. of precision characters >

[d , i , u , x , X , b] expresses the no. of characters in the numerical string.

[f , e , E] expresses the no. of characters after point.

[g , G] expresses the no. of significant digits.

Control string	Binary data	ASCII data
%d	K 100	"100"
%5d	K 100	"@@100"
%10.5d	K 100	"@@@@@00100"
%x	H 12A	"12a"
%5x	H 12A	"@@12a"
%10.5x	H 12A	"@@@@@0012a"
%b	H 123	"123"
%5b	H 123	"@@123"
%f	123.45678	"123.45678"
%8.3f	123.45678	"@123.456"
%e	1234.5678	"1.2345678e+3"
%9.3e	1234.5678	"@1.234e+3"
%g	1234.5678	"1234.5678"
%8.6g	1234.5678	"@1234.56"

(Note) '@' expresses a space.

3) Specification of zero padding

When the notation number is specified, zero padding can be used. Attach 0's before the notation digits.

Control string	Binary data	ASCII data
%05d	K 100	"00100"
%05x	H 12A	"0012a"
%05b	H 123	"00123"
%08.3f	123.45678	"0123.456"
%09.3e	1234.5678	"01.234e+3"

4) Padding-after or padding-before

Default is set to padding-after. To switch to padding-before, attach a negative sign (-) before the specified no. of digits.

Control string	Binary data	ASCII data
%-5d	K 100	"100@@"
%-5x	H 12A	"12a@@"
%-5b	H 123	"123@@"
%-8.3f	123.45678	"123.456@"
%-9.3e	1234.5678	"1.234e+3@"

(Note) '@' expresses a space.

5) Specification of sign

By default, a positive sign (+) is not added. To add a positive sign (+), add (+).

Control string	Binary data	ASCII data
%+d	K 100	"+100"
%+d	K -100	"-100"
%+5d	K 100	"@+100"
%+8.3f	123.45678	"+123.456"
%+9.3e	1234.5678	"+1.234e+3"

(Note) '@' expresses a space.

6) Specification of a figure position

Insert a space before a positive number, so that its figure position matches with that of a negative number. In order to match the positions, add (space).

Control string	Binary data	ASCII data
%_d	K 100	"@100"
%_d	K -100	"-100"
%_8.3f	123.45678	"@123.456"
%_8.3f	-123.45678	"-123.456"
%_9.3e	1234.5678	"@1.234e+3"
%_9.3e	-1234.5678	"-1.234e+3"

(Note) '@' expresses a space.

7) Specification of a different output format for numerical data

By adding '#', a different output format is automatically added.

Control string	Binary data	ASCII data	Remark
%#x	H 12A	"0x12a"	Add "0x".
%#X	H 12A	"0X12A"	Add "0X".
%#8.0f	123.45678	"@@@@123."	Always add "."
%#9.0e	1234.5678	"@@@@1.e+3"	
%#9.3E	1234.5678	"@@@@1.E+3"	
%#9.0g	1234	"@@@1234.0"	Always add "." and do not omit 0's after the point.
%#.9G	1234	"1234.56780"	

(Note) '@' expresses a space.

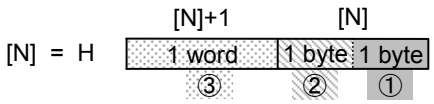
8) Appending characters to the figure

Append conversion specification characters (e.g. d, x, b, f, e, g) in lower digits to the numerical ASCII data.

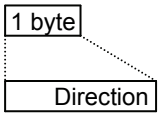
Control string	Binary data	ASCII data
%d,	K 100	"100,"
%x,	H 100	"100,"
%xH	H 100	"100H"
%bBCD	H 100	"100BCD"

■ Specification of conversion method [N]

- Conversion method [N] should be specified in a Hex format in the 32-bit area.



①: Direction of ASCII data



Direction: '0' = Forward direction, '1' = Reverse direction

Direction constraints are as follows:

Control string	Direction
"%d"	Reverse direction only
"%u"	Reverse direction only
"%x"	Forward and reverse directions
"%b"	Forward and reverse directions
"%f"	Reverse direction only
"%e"	Reverse direction only
"%g"	Reverse direction only

②: Storage start position

Specify lower bytes of [D] with '0'
Range to be specified: '0' to the max. value of ASCII data digits

Example) Conversion start position (1)

[D]	0 byte	**
	1 byte	31 ← Storage starts with a high address of [D]
[D]+1	2 bytes	32
	3 bytes	33
[D]+2	4 bytes	
	.	
	.	

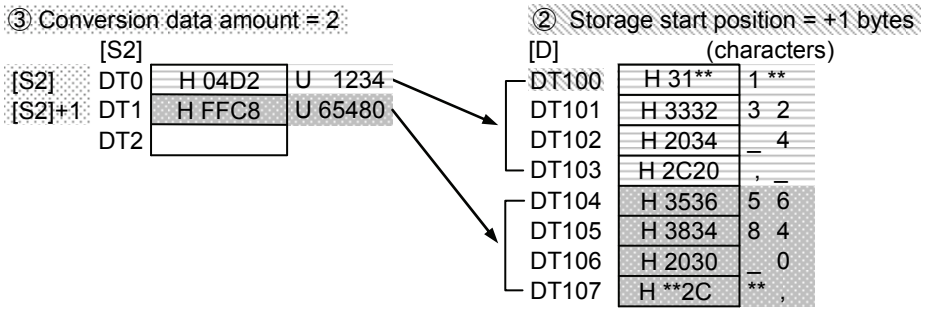
③: Conversion data amount

Specify conversion data amount
Range to be specified: 1 to 65535

■ Example of conversion

Example 1) Operation unit: 16 bits (US)

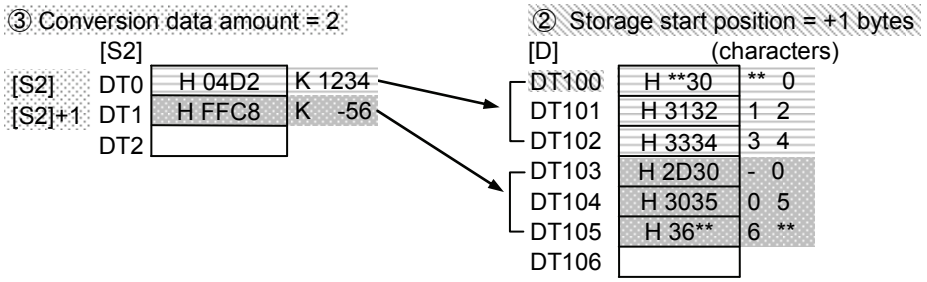
[i]…US
[S1]…"%-6u," : Convert 16-bit data into decimal ASCII data (6 digits)
[S2]…DT0
[N]…H 000201 01 : ① ASCII data (reverse direction)
[D]…DT100 ② Storage start position (1) → [D]+1 bytes
③ Conversion data amount (2) → Convert [S2] and [S2]+1



DT0: U 1234 (H 04D2) → DT100 to DT103: "1234 ,"
DT1: U 65480 (H FFC8) → DT104 to DT107: "65480 ,"

Example 2) Operation unit: 16 bits (SS)

[i]…SS
[S1]…"%05d" : Convert 16-bit data into decimal ASCII data (5 digits) (0 padding)
[S2]…DT0
[N]…H 000201 01 : ① ASCII data (reverse direction)
[D]…DT100 ② Storage start position (1) → [D]+1 bytes
③ Conversion data amount (2) → Convert [S2] and [S2]+1



DT0: K 1234 (H 04D2) → DT100 to DT102: "01234"
DT1: K -56 (H FFC8) → DT103 to DT105: "-0056"

Example 3) Operation unit: 32 bits (UL)

[i]…UL

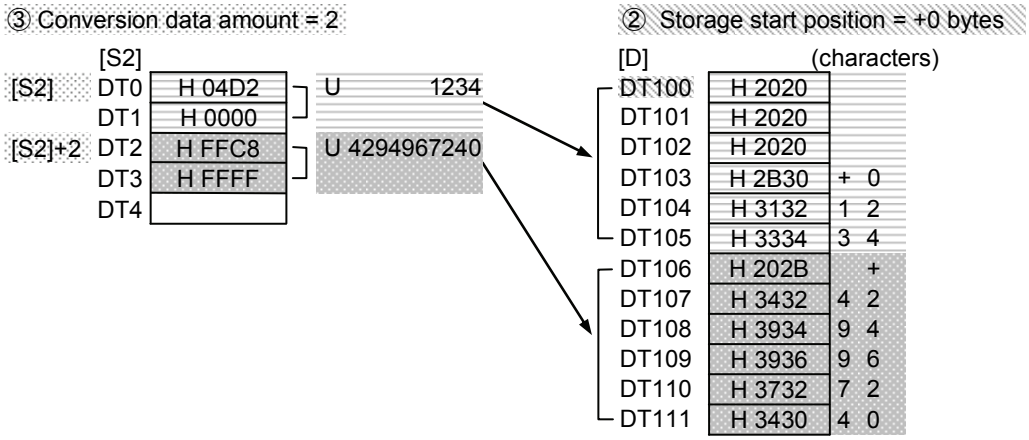
[S1]……"%+12.5d" : Convert 32-bit data into decimal ASCII data (12 digits) (sign specification)

[S2]…DT0

[N]…H 0002 00 01 : ① ASCII data (reverse direction)

[D]…DT100 ② Storage start position (0) → [D]+0 bytes

③ Conversion data amount (2) → Convert [S2] and [S2]+2



DT0 to DT1: U 1234 (H 0000 04D2) → DT100 to DT105: " +01234"
DT2 to DT3: U 4294967240 (H FFFF FFC8) → DT106 to DT111: " +4294967240"

Example 4) Operation unit: 32 bits (SL)

[i]…SL

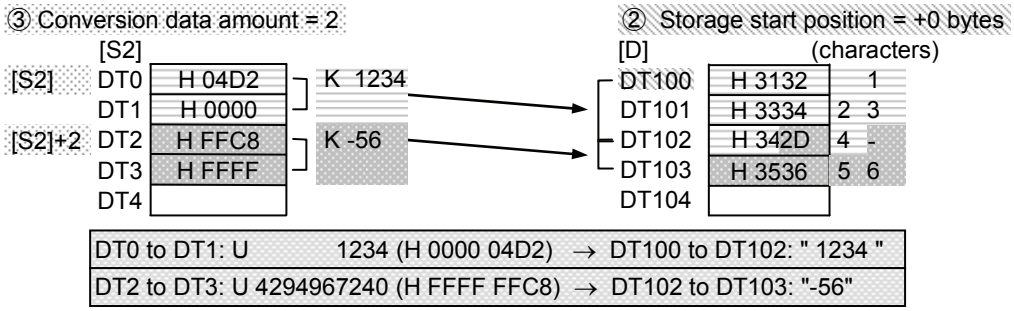
[S1]…"% d," : Convert 32-bit data into decimal ASCII data (figure position specification)

[S2]…DT0

[N]…H 0002 00 01 : ① ASCII data (reverse direction)

[D]…DT100 ② Storage start position (0) → [D]+0 bytes

③ Conversion data amount (2) → Convert [S2] and [S2]+2



DT0 to DT1: U 1234 (H 0000 04D2) → DT100 to DT102: " 1234 "
DT2 to DT3: U 4294967240 (H FFFF FFC8) → DT102 to DT103: "-56"

Example 5) Operation unit: 16 bits (US)

[i]…US

[S1]…"%+10.5u" : Convert 16-bit data into decimal ASCII data (10 digits) (sign specification)

[S2]…DT0

[N]…H 0002 00 01 : ① ASCII data (reverse direction)

[D]…DT100 ② Storage start position (0) → [D]+0 bytes

③ Conversion data amount (2) → Convert [S2] and [S2]+1

DT0: U 1234 (H 04D2) → DT100 to DT104: " +01234"
DT1: U 65480 (H FC8) → DT105 to DT109: " +65480"

Example 6) Operation unit: 32 bits (UL)

[i]…UL

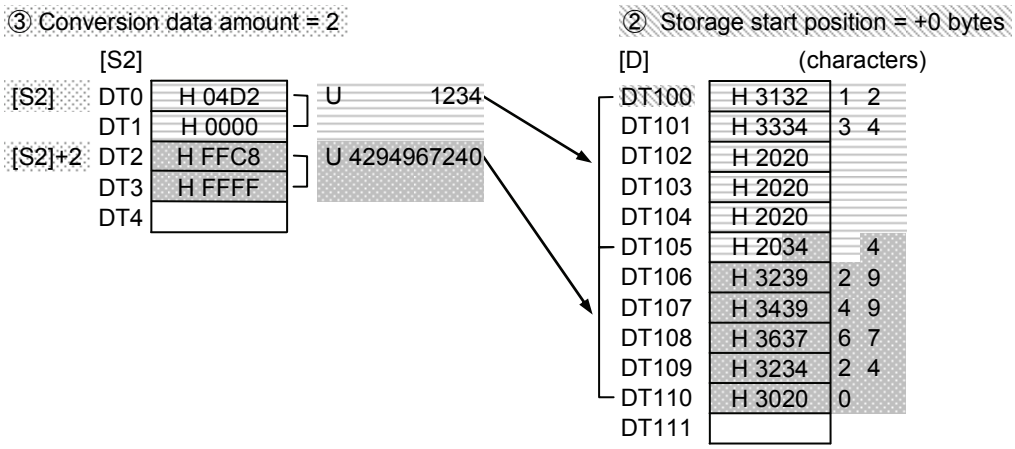
[S1]…"%-11u" : Convert 32-bit data into decimal ASCII data (11 digits) (sign specification)

[S2]…DT0

[N]…H 0002 00 01 : ① ASCII data (reverse direction)

[D]…DT100 ② Storage start position (0) → [D]+0 bytes

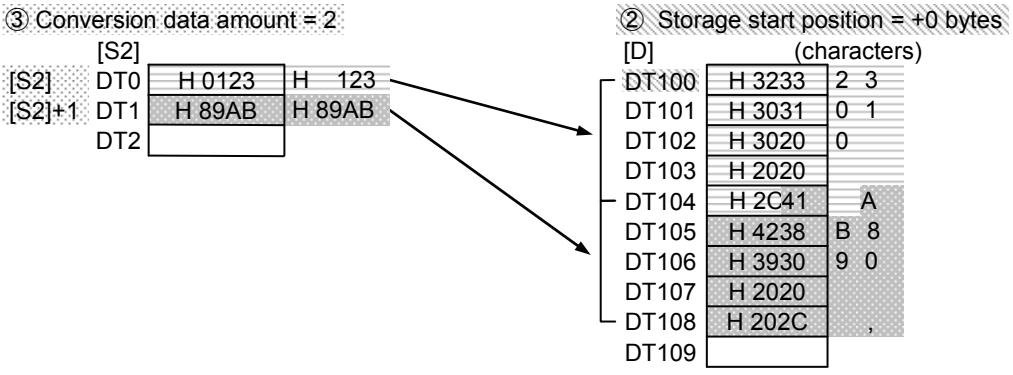
③ Conversion data amount (2) → Convert [S2] and [S2]+2



DT0 to DT1: U 1234 (H 0000 04D2) → DT100 to DT105: "1234 "
DT2 to DT3: U 4294967240 (H FFFF FFC8) → DT105 to DT110: "4294967240 "

Example 7) Operation unit: 16 bits (US)

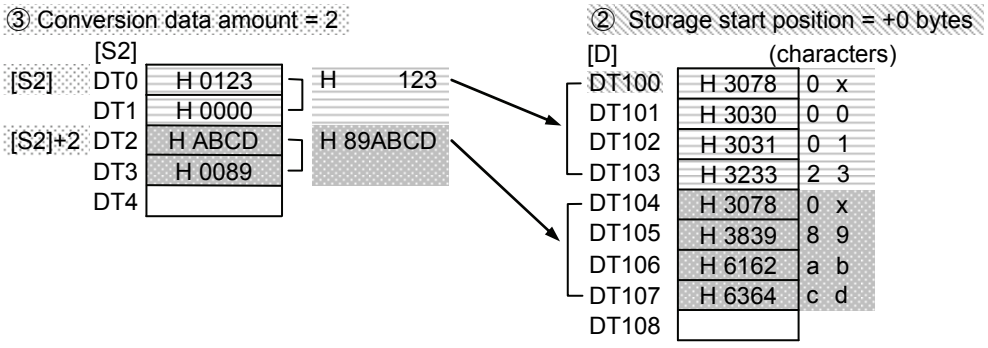
[i]…US
[S1]…"%8.5X," : Convert 16-bit data into hexadecimal ASCII data (8 digits)
[S2]…DT0
[N]…H 0002 00 00 : ① ASCII data (forward direction)
[D]…DT100 ② Storage start position (0) → [D]+0 bytes
③ Conversion data amount (2) → Convert [S2] and [S2]+1



DT0: H 0123 → DT100 to DT104: "23010 ,"
DT1: H 89AB → DT104 to DT108: "AB890 ,"

Example 8) Operation unit: 32 bits (UL)

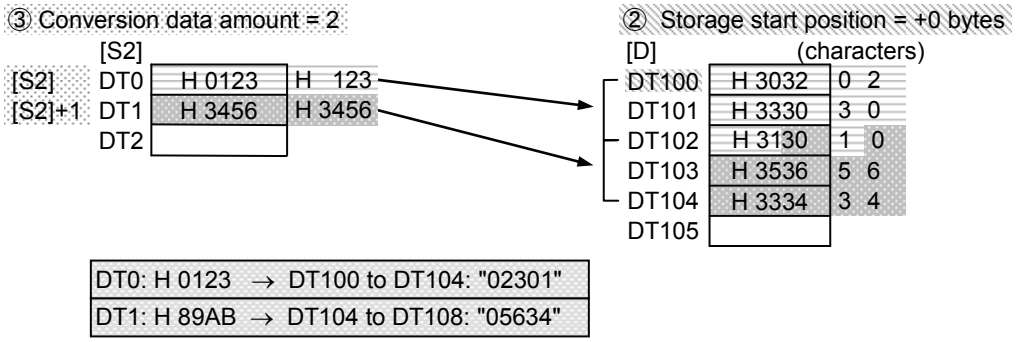
[i]…UL
[S1]…"%#8.6x" : Convert 32-bit data into hexadecimal ASCII data (8 digits)
[S2]…DT0
[N]…H 0002 00 01 : ① ASCII data (reverse direction)
[D]…DT100 ② Storage start position (0) → [D]+0 bytes
③ Conversion data amount (2) → Convert [S2] and [S2]+2



DT0 to DT1: H 0123 → DT100 to DT104: "0x000123"
DT2 to DT3: H 89AB → DT104 to DT108: "0x89abcd"

Example 9) Operation unit: 32 bits (UL)

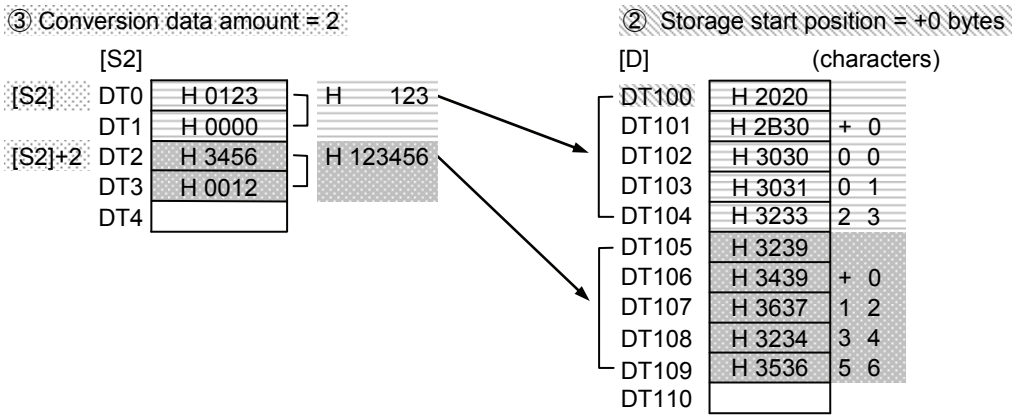
[i]…UL
[S1]…"%05b" : Convert 16-bit data into decimal ASCII data (5 digits)
[S2]…DT0
[N]…H 0002 00 00 : ① ASCII data (forward direction)
[D]…DT100 ② Storage start position (0) → [D]+0 bytes
③ Conversion data amount (2) → Convert [S2] and [S2]+1



DT0: H 0123 → DT100 to DT104: "02301"
DT1: H 89AB → DT104 to DT108: "05634"

Example 10) Operation unit: 32 bits (UL)

[i]…UL
[S1]…"%+10.7" : Convert 32-bit BCD data into decimal ASCII data (10 digits)
[S2]…DT0
[N]…H 0002 00 01 : ① ASCII data (reverse direction)
[D]…DT100 ② Storage start position (0) → [D]+0 bytes
③ Conversion data amount (2) → Convert [S2] and [S2]+2



DT0 to DT1: U 1234 (H 0000 04D2) → DT100 to DT104: " +0000123"
DT2 to DT3: U 4294967240 (H FFFF FFC8) → DT105 to DT109: " +0123456"

Example 11) Operation unit: 32 bits (SF)

[i]…SF

[S1]…"%#8.0f" : Convert 32-bit single precision real number data into floating point ASCII data (8 digits)

[S2]…DT0

[N]…H 0002 01 01 : ① ASCII data (reverse direction)

[D]…DT100 ② Storage start position (1) → [D]+1 bytes

③ Conversion data amount (2) → Convert [S2] and [S2]+4

DT0 to DT3: 123.45678 → DT100 to DT104: " 123."
DT4 to DT7: -12.3456789 → DT104 to DT108: "-12"

Example 12) Operation unit: 32 bits (SF)

[i]…SF

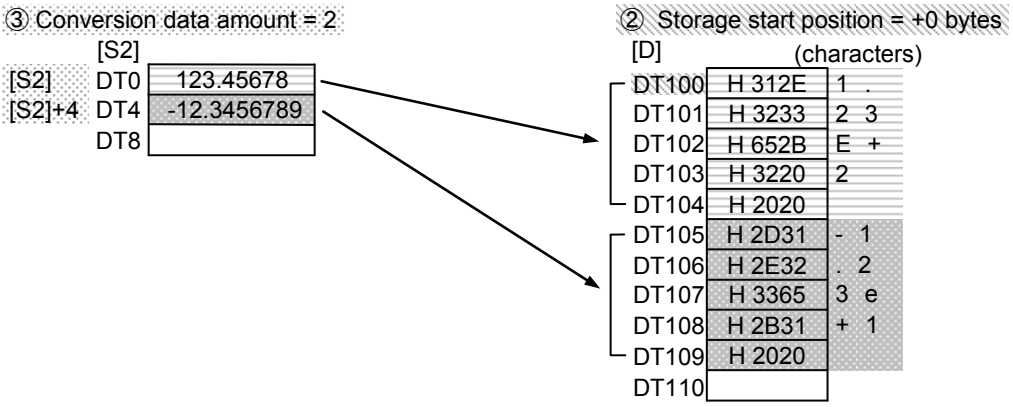
[S1]…"%-10.2e" : Convert 32-bit single precision real number data into exponential ASCII data (10 digits)

[S2]…DT0

[N]…H 0002 00 01 : ① ASCII data (reverse direction)

[D]…DT100 ② Storage start position (0) → [D]+0 bytes

③ Conversion data amount (2) → Convert [S2] and [S2]+4



DT0 to DT3: 123.45678 → DT100 to DT104: "1.23e+2 "
DT4 to DT7: -12.3456789 → DT105 to DT109: "-1.23e+1 "

Example 13) Operation unit: 32 bits (SF)

[i]…SF

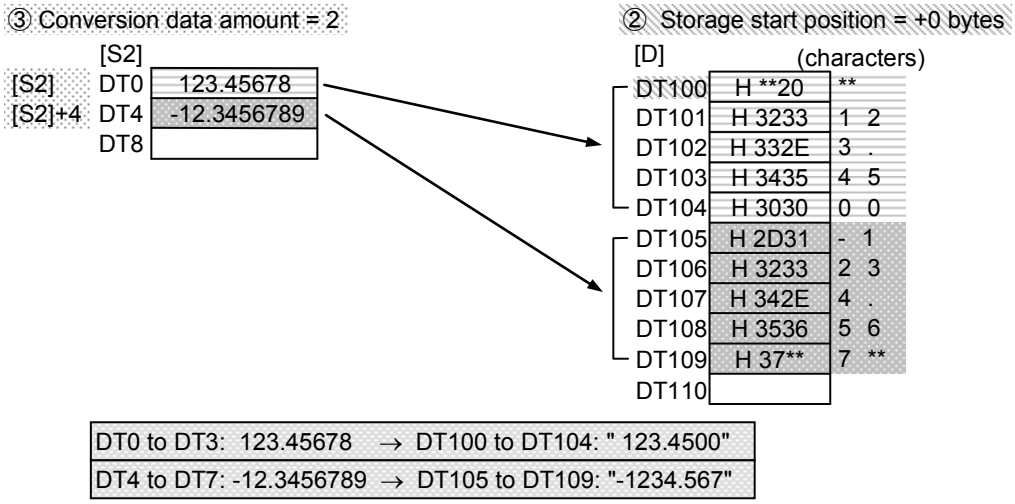
[S1]…"%#9.7g" : Convert 32-bit single precision real number data into exponential ASCII data (9 digits) or floating point ASCII data, whichever is shorter in the relevant notation

[S2]…DT0

[N]…H 00020001 : ① ASCII data (reverse direction)

[D]…DT100 ② Storage start position (0) → [D]+0 bytes

③ Conversion data amount (2) → Convert [S2] and [S2]+4



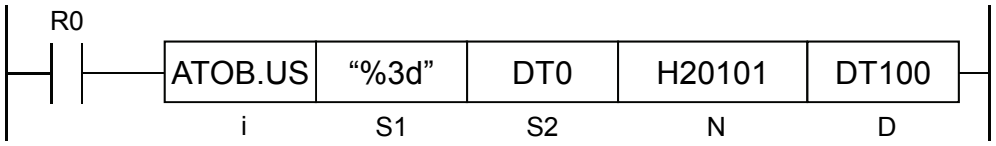
DT0 to DT3: 123.45678 → DT100 to DT104: " 123.4500"
DT4 to DT7: -12.3456789 → DT105 to DT109: "-1234.567"

■ Flag operations

Name	Explanation
SR7 SR8 (ER)	To be set in the case of out-of-range in indirect access (index modification).
	To be set when the conversion format specified by [S1] is not a settable operation unit.
	To be set when the conversion format specified by [S1] is not a control string.
	To be set when the no. of ASCII code digits specified by [N] exceeds the max. no. of digits for the control string specified by [S1].
	To be set when the storage start position specified by [N] is out of the range.
	To be set when the conversion data amount specified by [N] exceeds the [S2] area.
	To be set when the conversion data amount specified by [N] is out of the range.
	To be set when the conversion result exceeds the ASCII code storage area specified by [N].
	To be set when the conversion result exceeds the area specified by [D].

ATOB (Conversion: ASCII → BIN)

■ Ladder diagram



■ Available operation units (●: Available)

Operation unit	bit	US	SS	UL	SL	SF	DF
i		●	●	●	●	●	●

■ List of operands

Operand	Explanation
S1	Control string (2 to 16 characters)
S2	Starting number of the area that stores an ASCII code
N	Conversion method
D	Starting number of the area that stores the binary data as the conversion result

■ Available devices (●: Available)

Operand	16-bit device											32-bit device			Integers			Real numbers		String	Index modifier *1
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX	K	U	H	SF	DF	" "	
S1	●	●	●	●			●	●												●	
S2	●	●	●	●	●	●	●	●													●
N(*2)	●	●	●	●			●	●								●	●				●
D	●	●	●	●			●	●													●

*1: Only 16-bit devices, 32-bit devices, and integer constants can be modified (real number constants, and character constants cannot be specified).

*2: To be handled as a 32-bit integer (UL), regardless of operation unit.

■ Outline of operation

- Convert the specified ASCII code string into a specified data code.
- According to the control string specified by [S1], convert the ASCII code stored in the area specified by [S2] into binary data in a conversion method [N].
- The conversion result is stored in the area starting with [D].

■ Setting of control strings [S1]

Control string	Format of data to be converted	Available operation unit	Data range
"%nd"	Convert decimal ASCII data into 16-bit data	US	U0 to U65,535
	Convert decimal ASCII data into 32-bit data	SS	K-32,768 to K32,767
	Convert hexadecimal ASCII data into 16-bit data	UL	U0 to U4,294,967,000
	Convert hexadecimal ASCII data into 32-bit data	SL	K-2,147483,000 to K2,147,483,000
"%nx"	Convert hexadecimal ASCII data into 16-bit data (Forward/reverse direction)	US	H0 to HFFFF
	Convert hexadecimal ASCII data into 32-bit data (Forward/reverse direction)	UL	H0 to HFFFFFFFF
"%nb"	Convert decimal ASCII data into BCD data (Forward/reverse direction)	US	H0 to H9999
	Convert decimal ASCII data into BCD data (Forward/reverse direction)	UL	H0 to H99999999
"%nf"	Convert floating point ASCII data into 32-bit single precision real number data	SF	3.402823E+38 to 2.802597E-45
			-2.802597E-45 to -3.402823E+38
	Convert floating point ASCII data into 64-bit double precision real number data	DF	1.797693134862231E+308 to 4.940656458412465E-324
			1.797693134862231E+308 to 4.940656458412465E-324
"%nf"	Convert exponential ASCII data into 32-bit single precision real number data	SF	3.402823E+38 to 2.802597E-45
			-2.802597E-45 to -3.402823E+38
	Convert exponential ASCII data into 64-bit double precision real number data	DF	1.797693134862231E+308 to 4.940656458412465E-324
			1.797693134862231E+308 to 4.940656458412465E-324

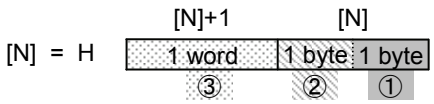
■ Format of ASCII data [S2]

Format	Example
Decimal integer	"1234567890",
	" 1234567", "1234567 ",
Hexadecimal integer	"123456789ABCDEF", "123456789abcdef",
	" 123456789", "123456789 ",
	"0000000123456789",
Floating point	"1234.56789",
	" 1234.567", "1234.567 ",
Exponential	"1.234E+12"
	" 1.23E+12", "1.23E+12 ",
	"1.234e+12",

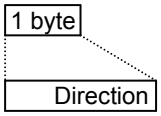
(Note 1) When the ASCII data are delimited with a comma (","), divide the data at the comma.
(Note 2) When the ASCII data end with a comma (","), finish reading at the comma.

■ Specification of conversion method [N]

- Conversion method [N] should be specified in a Hex format in the 32-bit area.



①: Direction of ASCII data



Direction: '0' = Forward direction, '1' = Reverse direction

Direction constraints are as follows:

Control string	Direction
"%nd"	Reverse direction only
"%nx"	Forward and reverse directions
"%nb"	Forward and reverse directions
"%nf"	Reverse direction only
"%ne"	Reverse direction only

②: Conversion start position

Specify lower bytes of [S2] with '0'
Range to be specified: '0' to the max. value of ASCII data digits

Example) Conversion start position (1)

[S2]	0 byte	**
	1 byte	H 31
[S2] +1	2 bytes	H 32
	3 bytes	H 33
[S2] +2	4 bytes	
	▪	

← Storage starts with a high address of [S2]

③: Conversion data amount

Specify conversion data amount
Range to be specified: 1 to 65535

■ Format of ASCII data

Format	Example of data
Decimal integer	"1234567890",
	" 1234567", "1234567 ",
Hexadecimal integer	"123456789ABCDEF", "123456789abcdef",
	" 123456789", "123456789 ",
	"0000000123456789",
Floating point	"1234.56789",
	" 1234.567", "1234.567 ",
Exponential	"1.234E+12"
	" 1.23E+12", "1.23E+12 ",
	"1.234e+12",

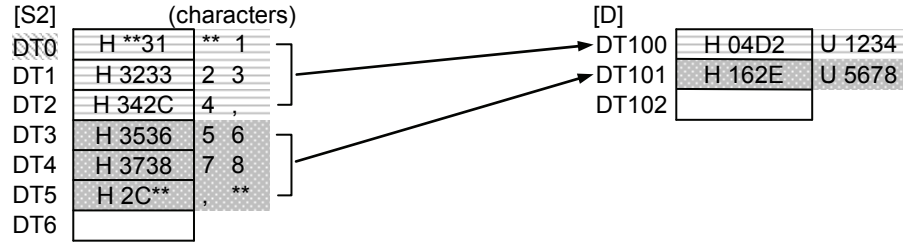
(Note 1) When the ASCII data are delimited with a comma (","), divide the data at the comma.
(Note 2) When the ASCII data end with a comma (","), finish reading at the comma.

■ Example of conversion

Example 1) Convert decimal ASCII data → Binary data

[i]…US
[S1]…"%d," : Convert hexadecimal ASCII data into 10-bit data (Data end: ',')
[S2]…DT0
[N]…H 00020101 : ① ASCII data (reverse direction)
[D]…DT100 ② Storage start position (1) → [S2]+1 bytes
③ Conversion data amount (2) → Read down to two ','s

② Storage start position = +1 bytes
③ Conversion data amount = 2

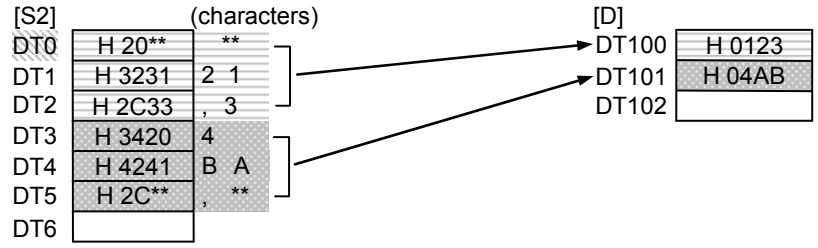


DT0 to DT2: "1234," → DT100: U 1234 (H 04D2)
DT3 to DT5: "5678," → DT101: U 5678 (H 162E)

Example 2) Convert hexadecimal ASCII data → Binary data

[i]…US
[S1]…"%x" : Convert hexadecimal ASCII data into 16-bit data (Data end: ',')
[S2]…DT0
[N]…H 00020101 : ① ASCII data (reverse direction)
[D]…DT100 ② Storage start position (1) → [S2]+1 bytes
③ Conversion data amount (2) → Read down to two ',' s

② Storage start position = +1 bytes
③ Conversion data amount = 2

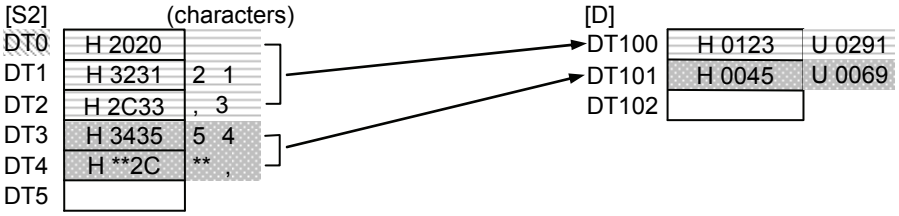


DT0 to DT2: "123," → DT100: H 0123
DT3 to DT5: "4AB," → DT101: H 04AB

Example 3) Convert decimal ASCII data → BCD data

[i]…US
[S1]…"%b" : Convert decimal ASCII data into BCD data (Data end: ',')
[S2]…DT0
[N]…H 00020001 : ① ASCII data (reverse direction)
[D]…DT100 ② Storage start position (0) → [S2]+0 bytes
③ Conversion data amount (2) → Read down to two ','s

② Storage start position = +0 bytes
③ Conversion data amount = 2

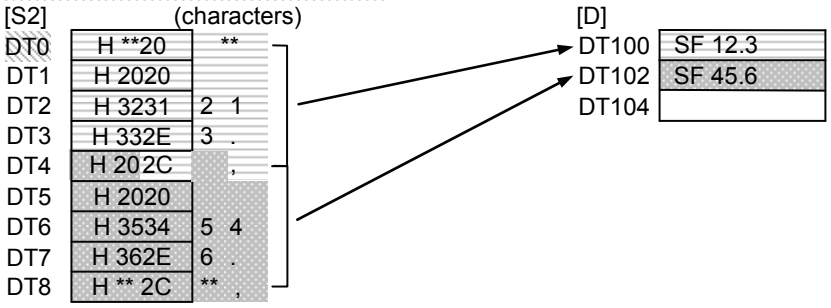


DT0 to DT2: " 123," → DT100: U 0291 (H 0123)
DT3 to DT5: "45," → DT101: U 0069 (H 0045)

Example 4) Convert floating point ASCII data → Real number data

[i]…SF
[S1]…"%f" : Convert floating point ASCII data into 32-bit real number data (Data end: ',')
[S2]…DT0
[N]…H 00020101 : ① ASCII data (reverse direction)
[D]…DT100 ② Storage start position (1) → [S2]+1 bytes
③ Conversion data amount (2) → Read down to two ','s

② Storage start position = +1 bytes
③ Conversion data amount = 2



DT0 to DT4: " 12.3," → DT100 to DT101: SF 12.3
DT4 to DT8: " 45.6," → DT102 to DT103: SF 45.6

Example 5) Operation unit: 32 bits (SF)

[i]…SF

[S1]…"%e" : Convert exponential ASCII data into 32-bit single precision real number data (Data end: ',')

[S2]…DT0

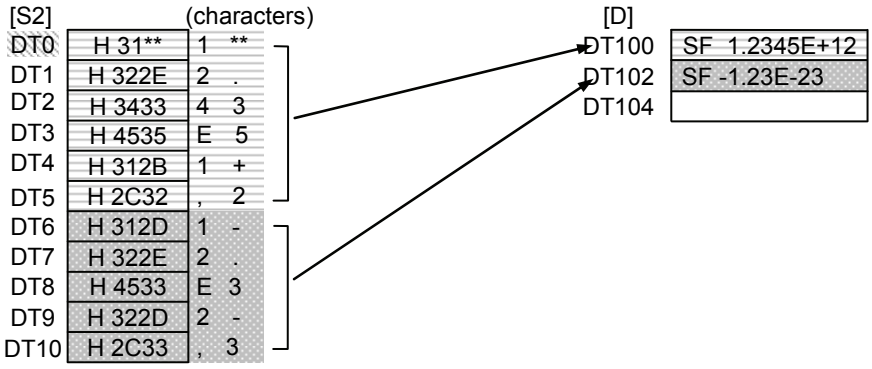
[N]…H 000201 01 : ① ASCII data (reverse direction)

[D]…DT100 ② Storage start position (1) → [S2]+1 bytes

③ Conversion data amount (2) → Read down to two ','s

② Storage start position = +1 bytes

③ Conversion data amount = 2



DT0 to DT5: "1.2345E+12," → DT100 to DT101: SF 1.2345E+12
DT6 to DT10: "-1.23E-23," → DT102 to DT103: SF -1.23E-23

Example 6) Operation unit: 16 bits (US)

[i]…US

[S1]…"%5d" : Convert decimal ASCII data (5 digits) into 16-bit data

[S2]…DT0

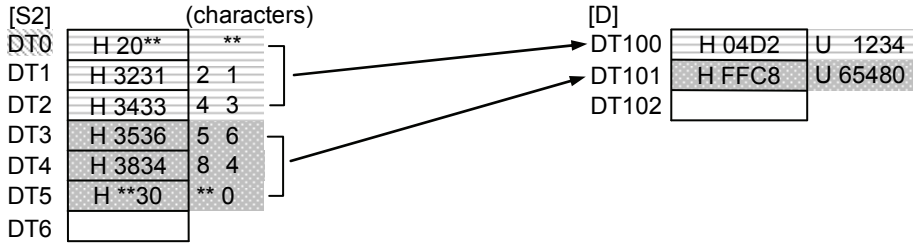
[N]…H 000201 01 : ① ASCII data (reverse direction)

[D]…DT100 ② Storage start position (1) → [S2]+1 bytes

③ Conversion data amount (2) → Convert DT0.H to DT2.H and DT3.L to DT5.L

② Storage start position = +1 bytes

③ Conversion data amount = 2



DT0 to DT2: "1234" → DT100: U 1234 (H 04D2)
DT3 to DT5: "65480 " → DT101: U 65480 (H FFC8)

Example 7) Operation unit: 16 bits (SS)

[i]…SS

[S1]…"%5d" : Convert decimal ASCII data (5 digits) into 16-bit data

[S2]…DT0

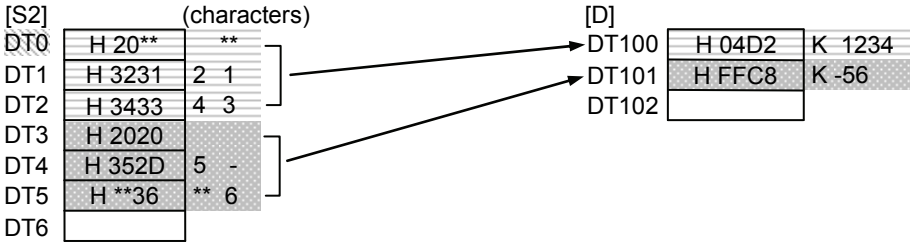
[N]…H 0002 01 01 : ① ASCII data (reverse direction)

[D]…DT100 ② Storage start position (1) → [S2]+1 bytes

③ Conversion data amount (2) → Convert DT0.H to DT2.H and DT3.L to DT5.L

② Storage start position = +1 bytes

③ Conversion data amount = 2



DT0 to DT2: " 1234" → DT100: K 1234 (H 04D2)
DT3 to DT5: " -56" → DT101: K -56 (H FFC8)

Example 8) Operation unit: 32 bits (UL)

[i]…UL

[S1]…"%10d" : Convert decimal ASCII data (10 digits) into 32-bit data

[S2]…DT0

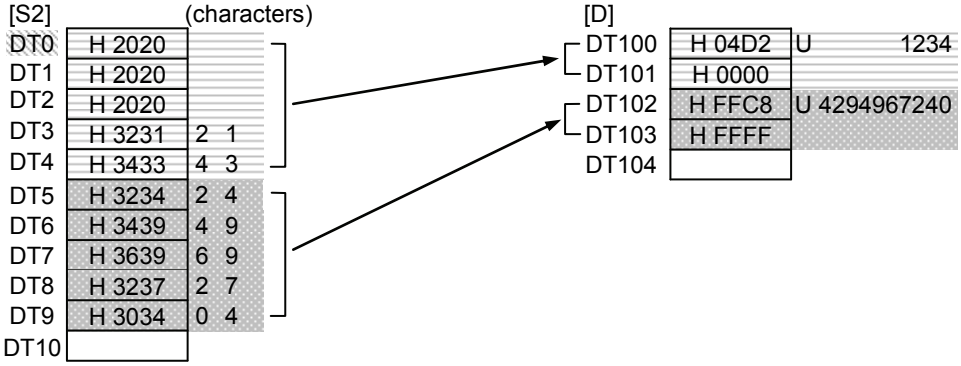
[N]…H 0002 00 01 : ① ASCII data (reverse direction)

[D]…DT100 ② Storage start position (0) → [S2]+0 bytes

③ Conversion data amount (2) → Convert DT0.L to DT4.H and DT5.L to DT9.H

② Storage start position = +0 bytes

③ Conversion data amount = 2

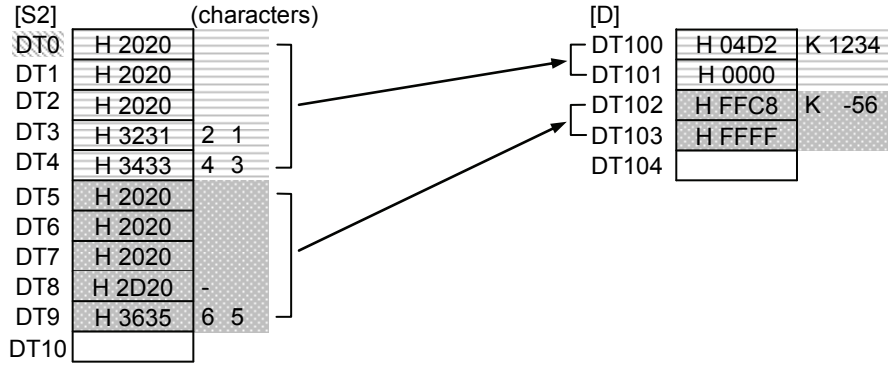


DT0 to DT4: " 1234" → DT100 to DT101: U 1234 (H 0000 04D2)
DT5 to DT9: "4294967240" → DT102 to DT103: U 4294967240 (H FFFF FFC8)

Example 9) Operation unit: 32 bits (SL)

[i]…SL
[S1]…"%10d" : Convert decimal ASCII data (10 digits) into 32-bit data
[S2]…DT0
[N]…H 0002 00 01 : ① ASCII data (reverse direction)
[D]…DT100 ② Storage start position (0) → [S2]+0 bytes
③ Conversion data amount (2) → Convert DT0.L to DT4.H and DT5.L to DT9.H

② Storage start position = +0 bytes
③ Conversion data amount = 2

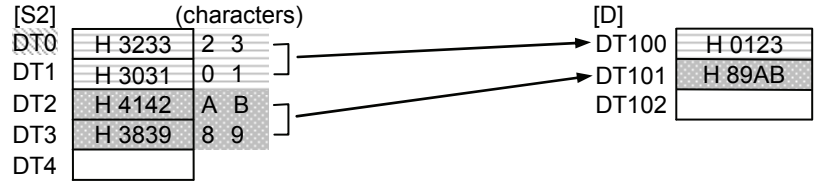


DT0 to DT4: " 1234" → DT100 to DT101: K 1234
DT5 to DT9: " -56" → DT102 to DT103: K -56

Example 10) Operation unit: 16 bits (US)

[i]…US
[S1]…"%4x" : Convert hexadecimal ASCII data (4 digits) into 16-bit data
[S2]…DT0
[N]…H 0002 00 00 : ① ASCII data (forward direction)
[D]…DT100 ② Storage start position (0) → [S2]+0 bytes
③ Conversion data amount (2) → Convert DT0.L to DT1.H and DT2.L to DT3.H

② Storage start position = +0 bytes
③ Conversion data amount = 2

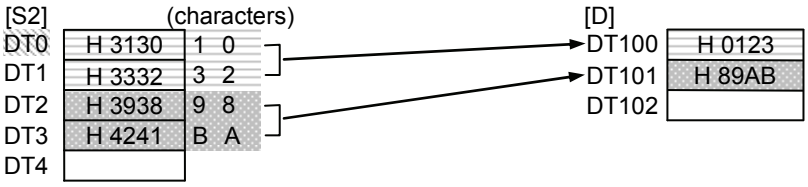


DT0 to DT1: "0123" → DT100: H 0123
DT2 to DT3: "89AB" → DT101: H 89AB

Example 11) Operation unit: 16 bits (US)

[i]…US
[S1]…"%4x" : Convert hexadecimal ASCII data (4 digits) into 16-bit data
[S2]…DT0
[N]…H 0002 00 01 : ① ASCII data (reverse direction)
[D]…DT100 ② Storage start position (0) → [S2]+0 bytes
③ Conversion data amount (2) → Convert DT0.L to DT1.H and DT2.L to DT3.H

② Storage start position = +0 bytes
③ Conversion data amount = 2

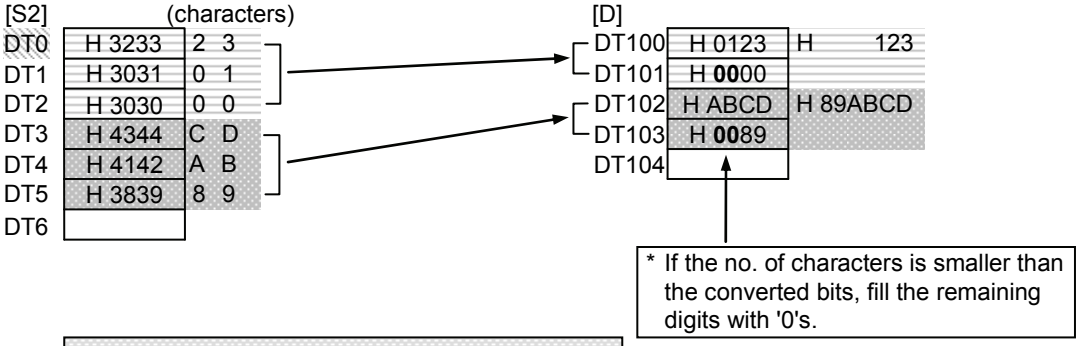


DT0 to DT1: "0123" → DT100: H 0123
DT2 to DT3: "89AB" → DT101: H 89AB

Example 12) Operation unit: 32 bits (UL)

[i]…UL
[S1]…"%6x" : Convert hexadecimal ASCII data (6 digits) into 32-bit data
[S2]…DT0
[N]…H 0002 00 00 : ① ASCII data (forward direction)
[D]…DT100 ② Storage start position (0) → [S2]+0 bytes
③ Conversion data amount → Read down to two ','s

② Storage start position = +0 bytes
③ Conversion data amount = 2



DT0 to DT2: "000123" → DT100: H 0000 0123
DT3 to DT5: "89ABCD" → DT101: H 0089 ABCD

Example 13) Operation unit: 32 bits (UL)

[i]…UL

[S1]…"%6x" : Convert hexadecimal ASCII data (6 digits) into 32-bit data

[S2]…DT0

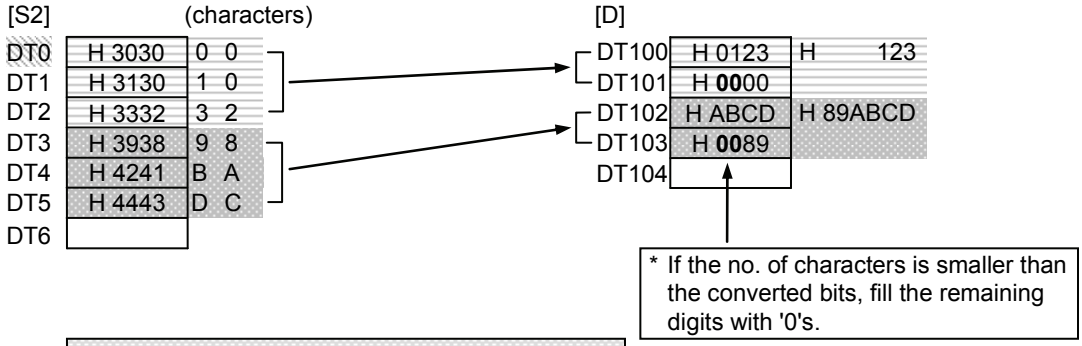
[N]…H 0002 00 01 : ① ASCII data (reverse direction)

[D]…DT100 ② Storage start position (0) → [S2]+0 bytes

③ Conversion data amount → Read down to two ','s

② Storage start position = +0 bytes

③ Conversion data amount = 2



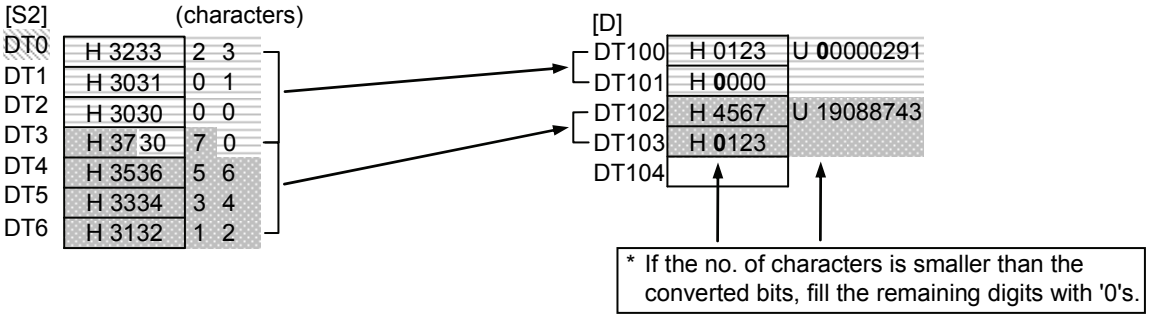
DT0 to DT2: "000123" → DT100: H 0000 0123

DT3 to DT5: "89ABCD" → DT101: H 0089 ABCD

Example 14) Operation unit: 32 bits (UL)

[i]…UL
[S1]…"%7b" : Convert decimal ASCII data (7 digits) into 32-bit BCD data
[S2]…DT0
[N]…H 0002 00 00 : ① ASCII data (forward direction)
[D]…DT100 ② Storage start position (0) → [S2]+0 bytes
③ Conversion data amount (2) → Convert DT0.L to DT3.L and DT3.H to DT6.H

② Storage start position = +0 bytes
③ Conversion data amount = 2

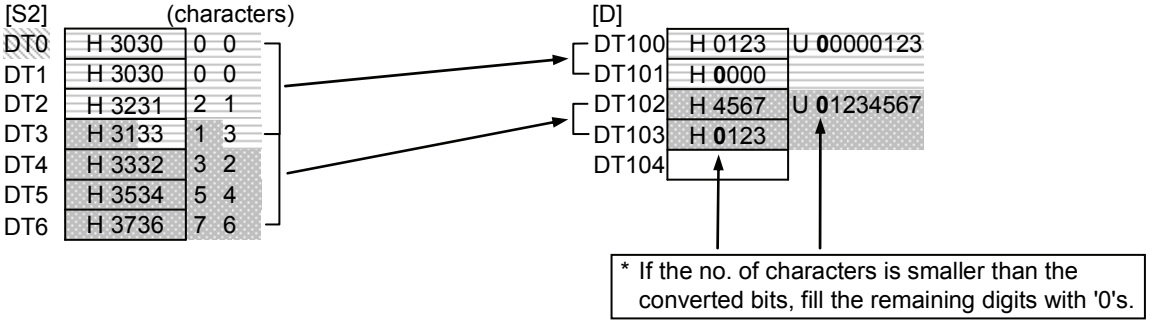


DT0 to DT3: "0000123" → DT100 to DT101: H 0000 0123
DT3 to DT6: "1234567" → DT102 to DT103: H 0123 4567

Example 15) Operation unit: 32 bits (UL)

[i]…UL
[S1]…"%7b" : Convert decimal ASCII data (7 digits) into 32-bit BCD data
[S2]…DT0
[N]…H 0002 00 01 : ① ASCII data (reverse direction)
[D]…DT100 ② Storage start position (0) → [S2]+0 bytes
③ Conversion data amount (2) → Convert DT0.L to DT3.L and DT3.H to DT6.H

② Storage start position = +0 bytes
③ Conversion data amount = 2



DT0 to DT3: "0000123" → DT100 to DT101: H 0000 0123
DT3 to DT6: "1234567" → DT102 to DT103: H 0123 4567

Example 16) Operation unit: 32 bits (SF)

[i]…SF

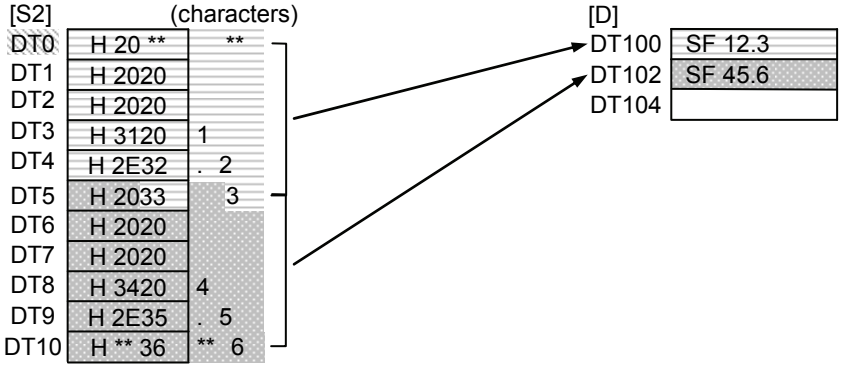
[S1]…"%10f" : Convert floating point ASCII data (10 characters) into 32-bit real number data

[S2]…DT0

[N]…H 0002 01 01 : ① ASCII data (reverse direction)
② Storage start position (1) → [S2]+1 bytes
③ Conversion data amount (2) → Convert DT0.H to DT5.L and DT5.H to DT10.L

[D]…DT100

② Storage start position = +1 bytes
③ Conversion data amount = 2



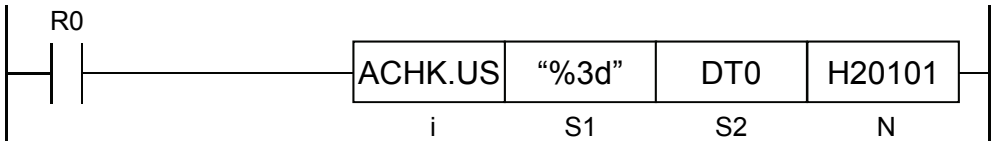
DT0 to DT5: " 12.3" → DT100 to DT101: SF 12.3
DT5 to DT10: " 45.6" → DT102 to DT103: SF 45.6

■ Flag operations

Name	Explanation
SR7 SR8 (ER)	To be set in the case of out-of-range in indirect access (index modification).
	To be set when the conversion format specified by [S1] is not a settable operation unit.
	To be set when the conversion format specified by [S1] is not a control string.
	To be set when the no. of ASCII code digits specified by [N] exceeds the max. no. of digits for the control string specified by [S1].
	To be set when the storage start position specified by [N] is out of the range.
	To be set when the conversion data amount specified by [N] exceeds the [S2] area.
	To be set when the conversion data amount specified by [N] is out of the range.
	To be set when the conversion result exceeds the area specified by [D].

ACHK (ASCII Data Check)

■ Ladder diagram



■ Available operation units (●: Available)

Operation unit	bit	US	SS	UL	SL	SF	DF
i		●	●	●	●	●	●

■ List of operands

Operand	Explanation
S1	Control string
S2	Starting number of the area that stores an ASCII code
N	Conversion method

■ Available devices (●: Available)

Operand	16-bit device											32-bit device			Integers			Real numbers		String	Index modifier *1
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX	K	U	H	SF	DF	" "	
S1	●	●	●	●			●	●												●	
S2	●	●	●	●	●	●	●	●													●
N(*2)	●	●	●	●			●	●								●	●				●

*1: Only 16-bit devices, 32-bit devices, and integer constants can be modified (real number constants, and character constants cannot be specified).

*2: To be handled as a 32-bit integer (UL), regardless of operation unit.

■ Outline of operation

- Check whether the specified ASCII code string can be converted using the ATOB instruction.
- According to the control string specified by [S1], check whether the ASCII code stored in the area specified by [S2] can be normally converted using the conversion method specified by [N].
- If the check result is normal, the system relay SRB turns on (1), and if abnormal, SRB turns off (0).
- Specify the same value as the ATOB instruction for the control string in [S1], start of the source data in [S2], and conversion method in [N].
- Operations of the ASC max. string length, valid range of conversion data value, etc. are the same as ATOB.

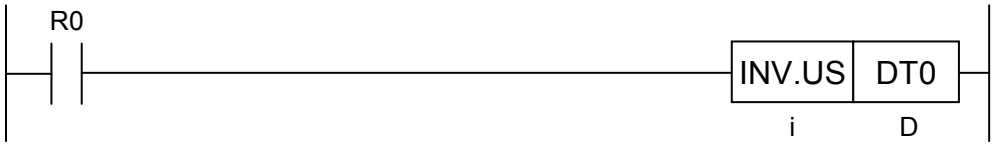
■ Flag operations

Name	Explanation
SR7 SR8 (ER)	To be set in the case of out-of-range in indirect access (index modification).
	To be set when the conversion format specified by [S1] is not a settable operation unit.
	To be set when the conversion format specified by [S1] is not a control string.
	To be set when the no. of ASCII code digits specified by [N] exceeds the max. no. of digits for the control string specified by [S1].
	To be set when the storage start position specified by [N] is out of the range.
	To be set when the conversion data amount specified by [N] exceeds the [S2] area.
	To be set when the conversion data amount specified by [N] is out of the range.

■ **MEMO**

INV (Data Inversion)

■ Ladder diagram



■ Available operation units (●: Available)

Operation unit	bit	US	SS	UL	SL	SF	DF
i		●	●	●	●		

■ List of operands

Operand	Explanation
D	Device address where the data to be inverted are stored

■ Available devices (●: Available)

Operand	16-bit device											32-bit device *1			Integers			Real numbers		String	Index modifier *2
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX *3	K	U	H	SF	DF	" "	
D	●	●	●	●			●	●	●		●	●	●	●							●

*1: Operation unit: 16-bit integers (SS, US) cannot be specified.
*2: Only 16-bit devices, 32-bit devices, and integer constants can be modified (real number constants, and character constants cannot be specified).
*3: Index register (I0 to IE)

■ Outline of operation

- Logically invert the device address value specified by [D].

/[D] → [D]

■ Process details

Example 1) Operation unit: 16 bits (US, SS)
[i]…US,SS
[D]…DT2

DT0	H 0011	→	DT0	H 0011
DT1	H 2233		DT1	H 2233
DT2	**H 4455**		DT2	**H BBAA**
DT3	H 6677		DT3	H 6677
DT4	H 8899		DT4	H 8899

Example 2) Operation unit: 32 bits (UL, SL)
[i]…UL,SL
[D]…TS1

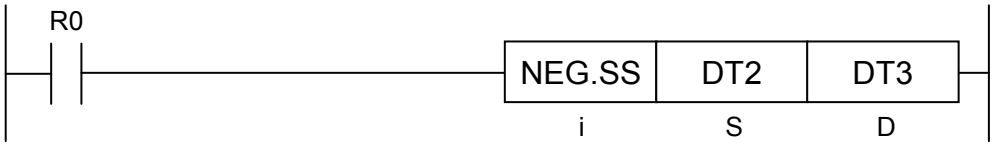
TS0	H 11223344	→	TS0	H 11223344
TS1	**H 55667788**		TS1	**H AA998877**
TS2	H 9900AABB		TS2	H 9900AABB
TS3	H CCDDEEFF		TS3	H CCDDEEFF
TS4	H 12345678		TS4	H 12345678

■ Flag operations

Name	Explanation
SR7 SR8 (ER)	To be set in the case of out-of-range in indirect access (index modification).

NEG (Sign Inversion)

■ Ladder diagram



■ Available operation units (●: Available)

Operation unit	bit	US	SS	UL	SL	SF	DF
i			●		●		

■ List of operands

Operand	Explanation
S	Device address where the data whose sign is to be inverted are stored, or constant
D	Storage device address

■ Available devices (●: Available)

Operand	16-bit device											32-bit device *1			Integers			Real numbers		String	Index modifier *2
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX *3	K	U	H	SF	DF	" "	
S	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●		●				●
D	●	●	●	●			●	●	●		●	●	●	●							●

*1: Operation unit: 16-bit integers (SS) cannot be specified.
*2: Only 16-bit devices, 32-bit devices, and integer constants can be modified (real number constants, and character constants cannot be specified).
*3: Index register (I0 to IE)

■ Outline of operation

- Calculate the complement for 2 with respect to the device address value specified by [S] or constant, and invert the sign of the data.
- The calculation result is stored in the device address specified by [D].

■ Process details

Example 1) Operation unit: 16 bits (SS)
[i]…SS
[S]…DT2 [D]…DT3

DT0	K 100	DT0	K 100
DT1	K 110	DT1	K 110
DT2	**K 120**	DT2	K 120
DT3	K 130	DT3	**K -120**
DT4	K 140	DT4	K 140

Example 2) Operation unit: 32 bits (SL)
[i]…SL
[S]…TS1 [D]…TS4

TS0	K 500	TS0	K 500
TS1	**K 1000**	TS1	K 1000
TS2	K 1500	TS2	K 1500
TS3	K 2000	TS3	K 2000
TS4	K 2500	TS4	**K -1000**

■ Precautions during programming

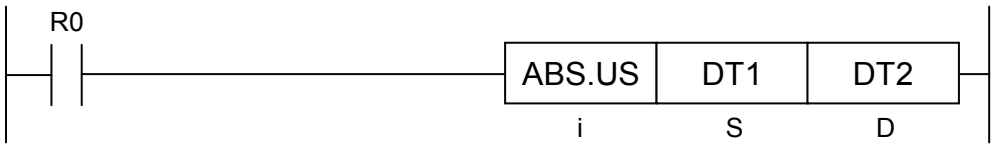
- When the min. negative value is specified, the result should be the min. negative value.

■ Flag operations

Name	Explanation
SR7 SR8 (ER)	To be set in the case of out-of-range in indirect access (index modification).

ABS (Absolute Value)

■ Ladder diagram



■ Available operation units (●: Available)

Operation unit	bit	US	SS	UL	SL	SF	DF
i		●	●	●	●		

■ List of operands

Operand	Explanation
S	Device address where the data for taking an absolute value are stored, or constant
D	Storage device address

■ Available devices (●: Available)

Operand	16-bit device											32-bit device *1			Integers			Real numbers		String	Index modifier *2
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX *3	K *4	U *5	H	SF	DF	" "	
S	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●				●
D	●	●	●	●			●	●	●		●	●	●	●							●

*1: Operation unit: 16-bit integers (SS, US) cannot be specified.
*2: Only 16-bit devices, 32-bit devices, and integer constants can be modified (real number constants, and character constants cannot be specified).
*3: Index register (I0 to IE)
*4: Only operation unit: signed integers (SS, SL) can be specified.
*5: Only operation unit: unsigned integers (US, UL) can be specified.

■ Outline of operation

- Take an absolute value for the device address specified by [S] or constant, and store the value in the device address specified by [D].

■ Process details

Example 1) Operation unit: Signed 16 bits (SS)
[i]…SS
[S]…DT1 [D]…DT2

DT0	K-100		DT0	K-100	
DT1	**K-110**	→	DT1	K-110	
DT2	K-120		DT2	**K110**	
DT3	K-130		DT3	K-130	
DT4	K-140		DT4	K-140	

Example 2) Operation unit: Signed 32 bits (SL)
[i]…SL
[S]…TS1 [D]…TS2

TS0	K-500		TS0	K-500	
TS1	**K-1000**	→	TS1	K-1000	
TS2	K-1500		TS2	**K1000**	
TS3	K-2000		TS3	K-2000	
TS4	K-2500		TS4	K-2500	

■ Precautions during programming

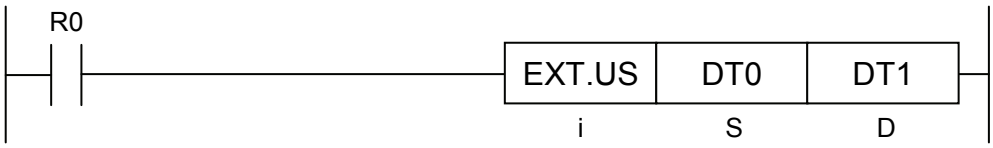
- If an unsigned integer (US, UL) is specified, the same value is stored in the storage location.

■ Flag operations

Name	Explanation
SR7	To be set in the case of out-of-range in indirect access (index modification).
SR8 (ER)	To be set when the min. negative value is specified for [S].

EXT (Sign Extension)

■ Ladder diagram



■ Available operation units (●: Available)

Operation unit	bit	US	SS	UL	SL	SF	DF
i		●	●				

■ List of operands

Operand	Explanation
S	Device address where the data for sign extension are stored, or constant
D	Storage device address

■ Available devices (●: Available)

Operand	16-bit device											32-bit device *1			Integers			Real numbers		String	Index modifier *2
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX *3	K *4	U *5	H	SF	DF	" "	
S	●	●	●	●	●	●	●	●	●	●	●				●	●	●				●
D	●	●	●	●			●	●	●		●	●	●	●							●

*1: Only 16-bit devices, 32-bit devices, and integer constants can be modified (real number constants, and character constants cannot be specified).
*2: Index register (I0 to IE)
*3: Only operation unit: signed integers (SS) can be specified.
*4: Only operation unit: unsigned integers (US) can be specified.

■ Outline of operation

- Extend the sign for the device address value specified by [S] or constant, and store the result in the device address specified by [D].

■ Process details

Example 1) Operation unit: Signed 16 bits (SS)

[i]…SS

[S]…DT0 [D]…DT0

DT0 K -2(H FFFE) → DT0・DT1 K-2(H **FFFF**FFFE)
DT1 **K 0(H 0000)**

Example 2) Operation unit: Unsigned 16 bits (US)

[i]…US

[S]…DT0 [D]…DT0

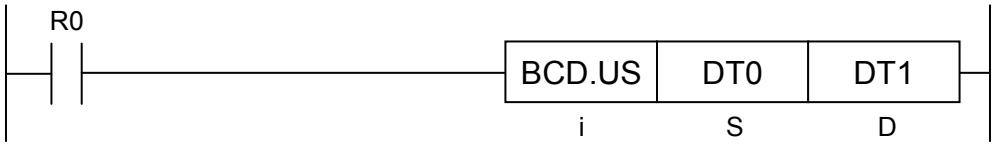
DT0 H FFFE → DT0・DT1 **H 0000**FFFFE
DT1 **H 1234**

■ Flag operations

Name	Explanation
SR7 SR8 (ER)	To be set in the case of out-of-range in indirect access (index modification).

BCD (Conversion: BCD Data)

■ Ladder diagram



■ Available operation units (●: Available)

Operation unit	bit	US	SS	UL	SL	SF	DF
i		●		●			

■ List of operands

Operand	Explanation
S	Device address that store the binary data to be converted are stored, or constant
D	Device address to store the conversion result

■ Available devices (●: Available)

Operand	16-bit device											32-bit device			Integers			Real numbers		String	Index modifier
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX	K	U	H	SF	DF	" "	*1
S	●	●	●	●	●	●	●	●	●	●	●					●	●				●
D	●	●	●	●			●	●	●		●										●

*1: Only 16-bit devices, 32-bit devices, and integer constants can be modified (real number constants, and character constants cannot be specified).

■ Outline of operation

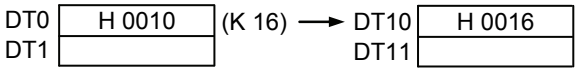
- Convert the device address value specified by [S] or constant, from binary data to BCD data, and store the result in the device address specified by [D].

■ Process details

Example 1) Operation unit: 16 bits (US)

[i]…US

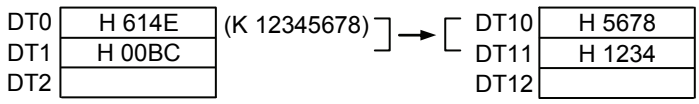
[S]…DT0 [D]…DT10



Example 2) Operation unit: 32 bits (UL)

[i]…UL

[S]…DT0 [D]…DT10



■ Precautions during programming

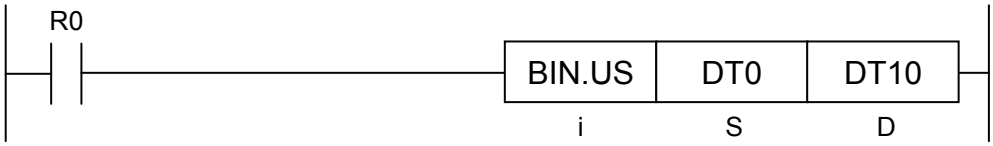
- If an unsigned integer (US, UL) is specified, the same value is stored in the storage location.

■ Flag operations

Name	Explanation
SR7 SR8 (ER)	To be set in the case of out-of-range in indirect access (index modification).
	To be set when the binary data exceed the BCD-convertible range. (E.g. Cases exceeding US: K 9999, UL: K 99999999)

BIN (Conversion: BCD → BIN)

■ Ladder diagram



■ Available operation units (●: Available)

Operation unit	bit	US	SS	UL	SL	SF	DF
i		●		●			

■ List of operands

Operand	Explanation
S	Device address that store the BCD data to be converted, or constant
D	Device address to store the conversion result

■ Available devices (●: Available)

Operand	16-bit device											32-bit device			Integers			Real numbers		String	Index modifier
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX	K	U	H	SF	DF	" "	*1
S	●	●	●	●	●	●	●	●	●	●	●					●	●				●
D	●	●	●	●			●	●	●		●										●

*1: Only 16-bit devices, 32-bit devices, and integer constants can be modified (real number constants, and character constants cannot be specified).

■ Outline of operation

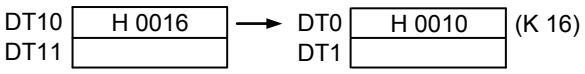
- Convert the device address value specified by [S] or constant, from BCD data to binary data, and store the result in the device address specified by [D].

■ Process details

Example 1) Operation unit: 16 bits (US)

[i]…US

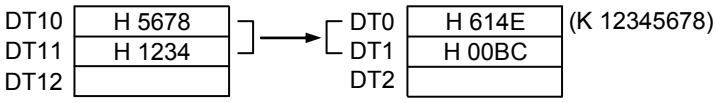
[S]…DT0 [D]…DT10



Example 2) Operation unit: 32 bits (UL)

[i]…UL

[S]…DT0 [D]…DT10



■ Precautions during programming

- If an unsigned integer (US, UL) is specified, the same value is stored in the storage location.

■ Flag operations

Name	Explanation
SR7	To be set in the case of out-of-range in indirect access (index modification).
SR8 (ER)	To be set when [S] is not BCD data.

DECO (Decoding)

■ Ladder diagram

■ Available operation units (●: Available)

- No operation units

■ List of operands

Operand	Explanation
S	Device address where the data for decoding are stored, or constant
N	Device address where the control data (specification of the conversion start bit, specification of the conversion-enabled bit length) are stored, or constant
D	Storage device address

■ Available devices (●: Available)

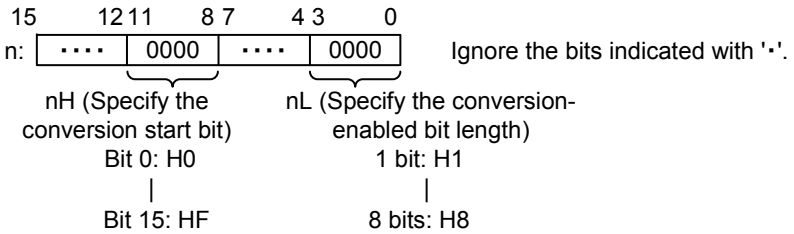
Operand	16-bit device											32-bit device			Integers			Real numbers		String	Index modifier *1
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX	K	U	H	SF	DF	" "	
S	●	●	●	●	●	●	●	●	●	●	●				●	●	●				●
n	●	●	●	●	●	●	●	●	●	●	●						●				●
D	●	●	●	●			●	●	●		●										●

*1: Only 16-bit devices, and integer constants can be modified (real number constants, and character constants cannot be specified).

■ Outline of operation

- Decode the device address value specified by [S] or constant, and store the decoding result in the device address specified by [D].
- The portion targeted by decoding is specified by the control data [n].
- Length of the device address required for storing the decoding result varies by the length of the portion to be decoded.

■ Specification of control data [N]



■ Example of conversion

Data to be converted	Decoded result (16 bits)
0000	0000 0000 0000 0001
0001	0000 0000 0000 0010
0010	0000 0000 0000 0100
0011	0000 0000 0000 1000
0100	0000 0000 0001 0000
0101	0000 0000 0010 0000
0110	0000 0000 0100 0000
0111	0000 0000 1000 0000
1000	0000 0001 0000 0000
1001	0000 0010 0000 0000
1010	0000 0100 0000 0000
1011	0000 1000 0000 0000
1100	0001 0000 0000 0000
1101	0010 0000 0000 0000
1110	0100 0000 0000 0000
1111	1000 0000 0000 0000

■ Specification of nL and length of calculation result

nL value Conversion-enabled bit length	Occupancy length of the decoded result	Enabled bit length of the decoded result	Value other than the enabled bit length in [D]
1	1 word	2	0
2	1 word	4	0
3	1 word	8 (1 byte)	0
4	1 word	16 (1 byte)	-
5	2 words	32 (2 bytes)	-
6	4 words	64 (4 bytes)	-
7	8 words	128 (8 bytes)	-
8	16 words	256 (16 bytes)	-

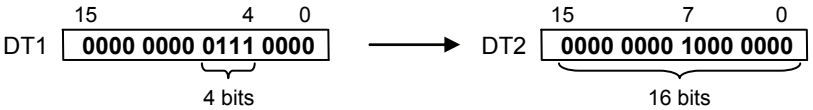
■ Process details

Example) Decode 4 bits from Bit 4

[S]…DT1 [n]…H 0404 [D]…DT2

* Store the result of decoding the specified portion (“0111”=7) into the 16-bit (2^4-bit) device address starting with DT2.

* The 16-bit area starting with DT2 turns on, while the other bits become '0'.

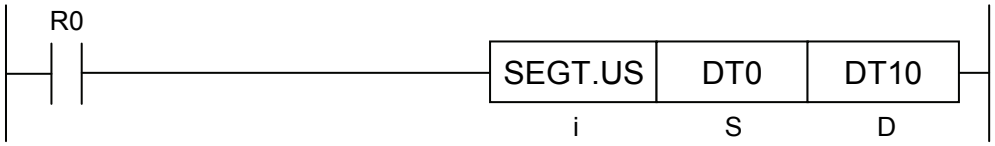


■ Flag operations

Name	Explanation
SR7 SR8 (ER)	To be set in the case of out-of-range in indirect access (index modification).
	To be set unless 1 ≤ nL ≤ 8, where nL is the conversion-enabled bit length.
	To be set unless 1 ≤ nH + nL ≤ 16, where nH is the conversion start bit and nL is the conversion-enabled bit length.
	To be set if, when the decoded result is stored in the device address specified by [D], it exceeds the area.

SEGT (7-Segment Decoding)

■ Ladder diagram



■ Available operation units (●: Available)

Operation unit	bit	US	SS	UL	SL	SF	DF
i		●					

■ List of operands

Operand	Explanation
S	Device address where the data for decoding are stored, or constant
D	Storage device address

■ Available devices (●: Available)

Operand	16-bit device											32-bit device			Integers			Real numbers		String	Index modifier
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX	K	U	H	SF	DF	" "	*1
S	●	●	●	●	●	●	●	●	●	●	●					●	●				●
D	●	●	●	●			●	●	●		●										●

*1: Only 16-bit devices, and integer constants can be modified (real number constants, and character constants cannot be specified).

■ Outline of operation

- Convert the device address value specified by [S] or constant into 4-digit data for 7-segment notation, and store the resulting data in the 2-word device address starting with [D].

■ Process details

Example) H ABCD is stored in [S]

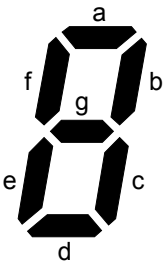
[S]…DT0 [D]…DT10

DT0	H ABCD	→ DT10	0011	1001	0101	1110
		DT11	0111	0111	0111	1100

■ Precautions during programming

- If an unsigned constant U is specified for [S], it should be converted as HEX data.

■ Notation and corresponding data

Figure	1-digit data to be converted [S]				1-digit data for 7-segment notation [D]								7-segment notation	
						g	f	e	d	c	b	a		
0	0	0	0	0	0	0	1	1	1	1	1	1	0	
1	0	0	0	1	0	0	0	0	0	1	1	0	1	
2	0	0	1	0	0	1	0	1	1	0	1	1	2	
3	0	0	1	1	0	1	0	0	1	1	1	1	3	
4	0	1	0	0	0	1	1	0	0	1	1	0	4	
5	0	1	0	1	0	1	1	0	1	1	0	1	5	
6	0	1	1	0	0	1	1	1	1	1	0	1	6	
7	0	1	1	1	0	0	1	0	0	1	1	1	7	
8	1	0	0	0	0	1	1	1	1	1	1	1	8	
9	1	0	0	1	0	1	1	0	1	1	1	1	9	
A	1	0	1	0	0	1	1	1	0	1	1	1	A	
B	1	0	1	1	0	1	1	1	1	1	0	0	b	
C	1	1	0	0	0	0	1	1	1	0	0	1	C	
D	1	1	0	1	0	1	0	1	1	1	1	0	d	
E	1	1	1	0	0	1	1	1	1	0	0	1	E	
F	1	1	1	1	0	1	1	1	0	0	0	1	F	

■ Flag operations

Name	Explanation
SR7	To be set in the case of out-of-range in indirect access (index modification).
SR8 (ER)	To be set if, when the conversion result is stored in the device address specified by [D], it exceeds the area.

ENCO (Encoding)

■ Ladder diagram

■ Available operation units (●: Available)

- No operation units

■ List of operands

Operand	Explanation
S	Device address where the data for encoding are stored, or constant
n	Device address where control data (specification of the result output start bit, specification of the conversion-enabled bit length) are stored, or constant
D	Storage device address

■ Available devices (●: Available)

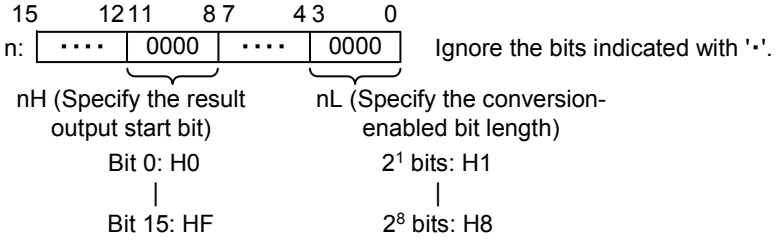
Operand	16-bit device											32-bit device			Integers			Real numbers		String	Index modifier *1
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX	K	U	H	SF	DF	" "	
S	●	●	●	●	●	●	●	●	●	●	●										●
n	●	●	●	●	●	●	●	●	●	●	●						●				●
D	●	●	●	●			●	●	●		●										●

*1: Only 16-bit devices, and integer constants can be modified (real number constants, and character constants cannot be specified).

■ Outline of operation

- Encode the device address value specified by [S] or constant, and store the encoding result in the device address specified by [D].
- The portion targeted by encoding is specified by the control data [n].
- When several bits targeted by encoding are ON, the uppermost bit is enabled.
- Encode the content for 2nL bits, starting with the device address specified by [S]. The encoded result is stored as a decimal number within 8 bits from the bit specified by nH.
- In the device address specified by [D], portions other than the one storing conversion result are set to 0.

■ Specification of control data [N]



■ Example of conversion

Data to be converted (16 bits)	Encoded result
0000 0000 0000 0001	0000
0000 0000 0000 0010	0001
0000 0000 0000 0100	0010
0000 0000 0000 1000	0011
0000 0000 0001 0000	0100
0000 0000 0010 0000	0101
0000 0000 0100 0000	0110
0000 0000 1000 0000	0111
0000 0001 0000 0000	1000
0000 0010 0000 0000	1001
0000 0100 0000 0000	1010
0000 1000 0000 0000	1011
0001 0000 0000 0000	1100
0010 0000 0000 0000	1101
0100 0000 0000 0000	1110
1000 0000 0000 0000	1111

■ Specification of nL and length of result

nL value	Conversion-enabled bit length	nL value	Conversion-enabled bit length
1	2	5	32 (2 bytes)
2	4	6	64 (4 bytes)
3	8 (1 byte)	7	128 (8 bytes)
4	16 (1 byte)	8	256 (16 bytes)

■ Process details

Example) [S]…DT10 [n]…H 0005 [D]…DT20

DGT S , S1 , n , D , D1

Transfer [n] digits from the [S1]th digit of the area specified by [S], to the [D1] digit of the 16-bit data specified by [D]. Transfer starts with the 0th digit, 1st digit, 2nd digit, and 3rd digit by every four bits from the lower level.

Conversion-enabled bits are DT10 to DT11 (32 bits from DT10).
The bit numbers that are ON in these two-word area are stored in a decimal form from Bit 0 of DT20.

DT10	0000	0001	0000	0000
DT11	0000	0000	0000	0000

Bit 8, counted from the lowest bit of DT10, is ON.

↓

DT20	0000	0000	0000	1000

Store H8 in DT20.

■ Precautions during programming

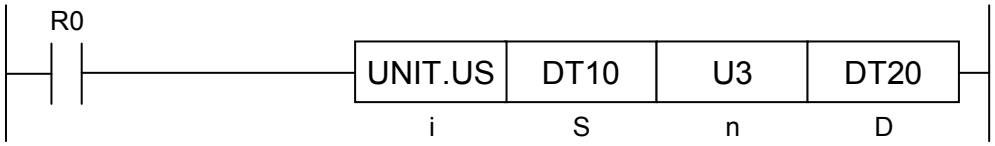
- If an unsigned constant U is specified for [S], it should be converted as Hex data.

■ Flag operations

Name	Explanation
SR7 SR8 (ER)	To be set in the case of out-of-range in indirect access (index modification).
	To be set unless 1 ≤ nL ≤ 8, where nL is the conversion-enabled bit length.
	To be set (consistency) unless the sum of nH (result output start bit no.) and nL (conversion-enabled bit length) is 1 ≤ nH + nL ≤ 16.
	To be set when the data to be encoded is all "0".

UNIT (Digit Unification)

■ Ladder diagram



■ Available operation units (●: Available)

Operation unit	bit	US	SS	UL	SL	SF	DF
i		●					

■ List of operands

Operand	Explanation
S	Device address where the data to be unified are stored, or constant (data format: unsigned 16-bit integer)
n	Device address where the no. of data to be unified is stored, or constant (data format: unsigned 16-bit integer)
D	Storage device address (data format: according to the operation unit)

■ Available devices (●: Available)

Operand	16-bit device											32-bit device			Integers			Real numbers		String	Index modifier *1
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX	K	U	H	SF	DF	" "	
S	●	●	●	●	●	●	●	●	●	●	●										●
n	●	●	●	●	●	●	●	●	●	●	●					●	●				●
D	●	●	●	●			●	●	●		●										●

*1: Only 16-bit devices, and integer constants can be modified (32-bit devices, real number constants, and character constants cannot be specified).

■ Outline of operation

- The lower 4 bits of the 16-bit data for [n] words starting with [S] are combined into 16-bit data.
- The available range for the no. of data to be unified [n] is 0 to 4. When [n] = 0, this instruction is not executed.
- The other portions of [D] are padded with "0".

■ Process details

[i]…US
[S]…DT10 [n]…U3 [D]…DT20

	15･･･12	11･･･8	7･･･4	3･･･0
DT10	0000	0000	0000	**0001**
DT11	0000	0000	0000	**0010**
DT12	0000	0000	0000	**0100**

DT20			
15･･･12	11･･･8	7･･･4	3･･･0
0000	**0100**	**0010**	**0001**

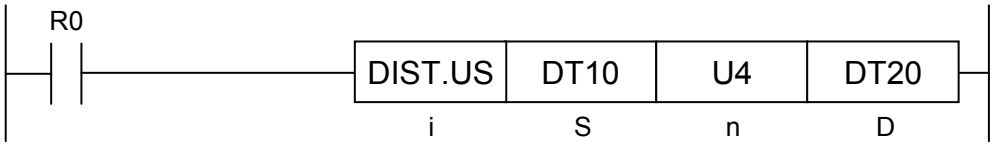
Padded with '0'.

■ Flag operations

Name	Explanation
SR7 SR8 (ER)	To be set in the case of out-of-range in indirect access (index modification).
	To be set when the no. of data to be unified [n] is out of the specified range.

DIST (Digit Disintegration)

■ Ladder diagram



■ Available operation units (●: Available)

Operation unit	bit	US	SS	UL	SL	SF	DF
i		●					

■ List of operands

Operand	Explanation
S	Device address where the data to be disintegrated are stored, or constant (data format: according to the operation unit)
n	Device address where the no. of points to be disintegrated are stored, or constant (data format: unsigned 16-bit integer)
D	Storage device address (data format: unsigned 16-bit integer)

■ Available devices (●: Available)

Operand	16-bit device											32-bit device			Integers			Real numbers		String	Index modifier *1
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX	K	U	H	SF	DF	" "	
S	●	●	●	●	●	●	●	●	●	●	●					●	●				●
n	●	●	●	●	●	●	●	●	●	●	●					●	●				●
D	●	●	●	●			●	●	●		●										●

*1: Only 16-bit devices, and integer constants can be modified (32-bit devices, real number constants, and character constants cannot be specified).

■ Outline of operation

- 16-bit device specified by [S] is disintegrated into 16-bit data by 4 bits. (The available range for the no. of data to be disintegrated [n] is 0 to 4.)
- When [n] = 0, this instruction is not executed.

■ Process details

Example) Operation unit: 16 bits (US)

[i]…US

[S]…DT10 [n]…U4 [D]…DT20

DT10			
15･･･12	11･･･8	7･･･4	3･･･0
0100	**0011**	**0001**	**0000**

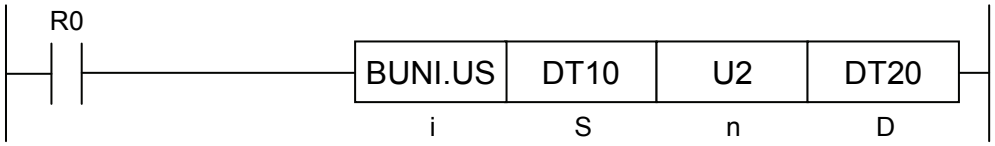
DT20	0000	0000	0000	**0000**
DT21	0000	0000	0000	**0001**
DT22	0000	0000	0000	**0011**
DT23	0000	0000	0000	**0100**

■ Flag operations

Name	Explanation
SR7	To be set in the case of out-of-range in indirect access (index modification).
SR8	To be set when the no. of data points to be disintegrated [n] is out of the specified range.
(ER)	To be set when, if data equivalent to [n] are transferred from the address specified by [D], it exceeds the device address.

BUNI (Byte Data Unification)

■ Ladder diagram



■ Available operation units (●: Available)

Operation unit	bit	US	SS	UL	SL	SF	DF
i		●		●			

■ List of operands

Operand	Explanation
S	Device address where the data to be unified are stored, or constant (data format: unsigned 16-bit integer)
n	Device address where the no. of data to be unified is stored, or constant (data format: unsigned 16-bit integer)
D	Storage device address (data format: according to the operation unit)

■ Available devices (●: Available)

Operand	16-bit device											32-bit device			Integers			Real numbers		String	Index modifier *1
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX *2	K	U	H	SF	DF	" "	
S	●	●	●	●	●	●	●	●	●	●	●										●
n	●	●	●	●	●	●	●	●	●	●	●					●	●				●
D	●	●	●	●			●	●	●		●										●

*1: Only 16-bit devices, and integer constants can be modified (32-bit devices, real number constants, and character constants cannot be specified).
*2: Index register (I0 to IE)

■ Outline of operation

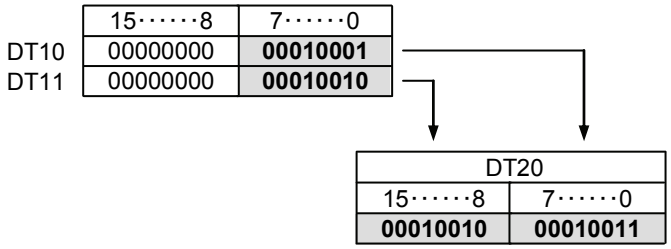
- When the operation unit is 16 bits (US), the lower 1 bit of the 16-bit data for [n] words starting with [S] are unified into 16-bit data. (The available range for the no. of data to be unified [n] is 0 to 2.)
- When the operation unit is 32 bits (UL), the lower 1 bit of the 16-bit data for [n] words starting with [S] are unified into 32-bit data. (The available range for the no. of data to be unified [n] is 0 to 4.)
- When [n] = 0, this instruction is not executed.
- The other portions of [D] are padded with "0".

■ Process details

Example 1) Operation unit: 16 bits (US)

[i]…US

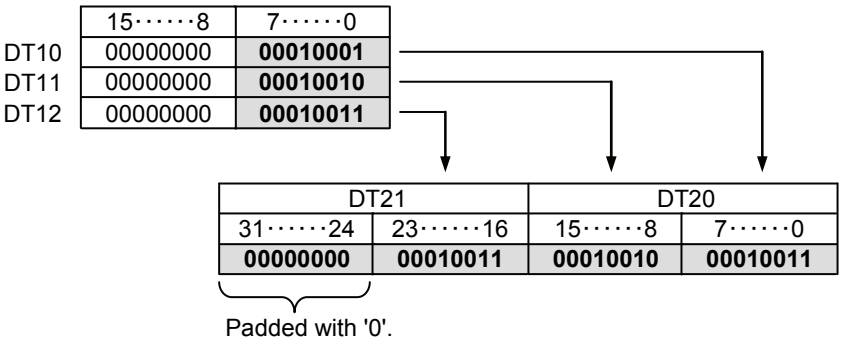
[S]…DT10 [n]…U2 [D]…DT20



Example 2) Operation unit: 32 bits (UL)

[i]…UL

[S]…DT10 [n]…U3 [D]…DT20

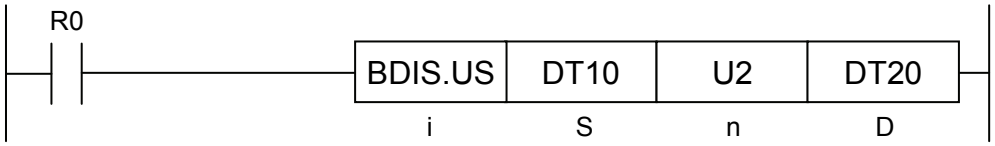


■ Flag operations

Name	Explanation
SR7 SR8 (ER)	To be set in the case of out-of-range in indirect access (index modification).
	To be set when the no. of data to be unified [n] is out of the specified range.

BDIS (Byte Data Disintegration)

■ Ladder diagram



■ Available operation units (●: Available)

Operation unit	bit	US	SS	UL	SL	SF	DF
i		●		●			

■ List of operands

Operand	Explanation
S	Device address where the data to be disintegrated are stored, or constant
n	Device address where the no. of points to be disintegrated are stored, or constant
D	Storage device address

■ Available devices (●: Available)

Operand	16-bit device											32-bit device			Integers			Real numbers		String	Index modifier *1
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX *2	K	U	H	SF	DF	" "	
S	●	●	●	●	●	●	●	●	●	●	●					●	●				●
n	●	●	●	●	●	●	●	●	●	●	●					●	●				●
D	●	●	●	●			●	●	●		●										●

*1: Only 16-bit devices, and integer constants can be modified (32-bit devices, real number constants, and character constants cannot be specified).
*2: Index register (I0 to IE)

■ Outline of operation

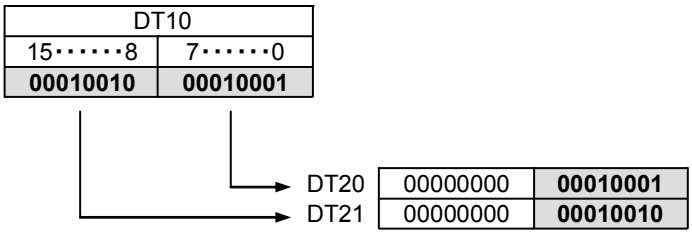
- When the operation unit is 16 bits (US), the 16-bit device specified by [S] is disintegrated into 16-bit data by 1 byte. (The available range for the no. of data to be disintegrated [n] is 0 to 2.)
- When the operation unit is 32 bits (UL), the 32-bit device specified by [S] is disintegrated into 16-bit data by 1 byte. (The available range for the no. of data to be disintegrated [n] is 0 to 4.)
- When [n] = 0, this instruction is not executed.

■ Process details

Example 1) Operation unit: 16 bits (US)

[i]…US

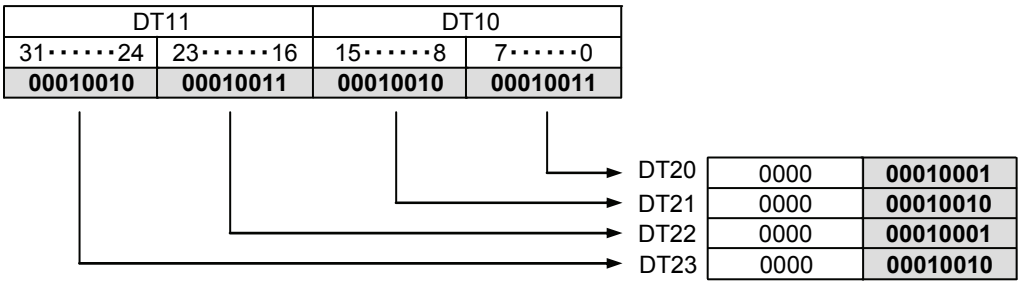
[S]…DT10 [n]…U2 [D]…DT20



Example 2) Operation unit: 32 bits (UL)

[i]…UL

[S]…DT10 [n]…U4 [D]…DT20

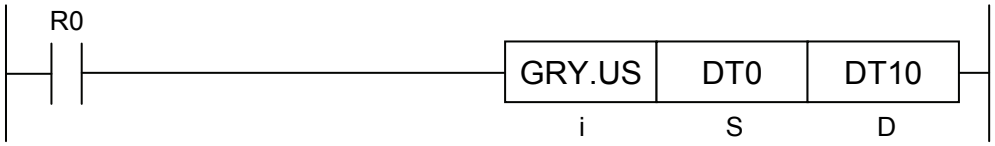


■ Flag operations

Name	Explanation
SR7 SR8 (ER)	To be set in the case of out-of-range in indirect access (index modification).
	To be set when the no. of data points to be disintegrated [n] is out of the specified range.
	To be set when, if data equivalent to [n] are transferred from the address specified by [D], it exceeds the device address.

GRY (Conversion: Binary → Gray Code)

■ Ladder diagram



■ Available operation units (●: Available)

Operation unit	bit	US	SS	UL	SL	SF	DF
i		●		●			

■ List of operands

Operand	Explanation
S	Device address where the data for conversion are stored, or constant
D	Storage device address

(Note) For gray codes, refer to the Correspondence Table: BIN / Gray Code.

■ Available devices (●: Available)

Operand	16-bit device											32-bit device *1			Integers			Real numbers		String	Index modifier *2
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX *3	K	U	H	SF	DF	" "	
S	●	●	●	●	●	●	●	●	●	●	●					●	●				●
D	●	●	●	●			●	●	●		●										●

*1: Operation unit: 16-bit integers (US) cannot be specified.
*2: Only 16-bit devices, and integer constants can be modified (32-bit devices, real number constants, and character constants cannot be specified).
*3: Index register (I0 to IE)

■ Outline of operation

- Convert the device address value specified by [S] or constant into a gray code, and store the result in the device address specified by [D].

■ Process details

Example 1) Operation unit: 16 bits (US)

[i]…US

[S]…DT0 [D]…DT10

DT0 0000000000011000 ⟶ DT10 0000000000010100

Example 2) Operation unit: 32 bits (UL)

[i]…UL

[S]…DT0 [D]…DT10

DT0, DT1 00000000000000000000000000011111

↓

DT10, DT11 00000000000000000000000000010000

■ Flag operations

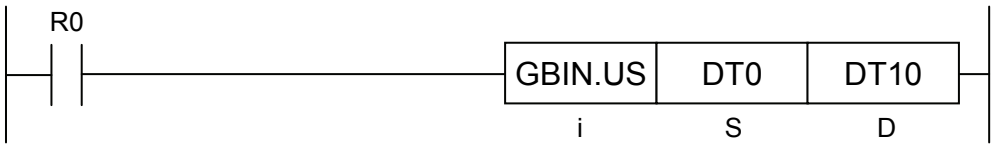
Name	Explanation
SR7 SR8 (ER)	To be set in the case of out-of-range in indirect access (index modification).

■ Correspondence Table: DEC / BIN / Gray Code

Decimal number	Binary number	Gray code
0	0000 0000 0000 0000	0000 0000 0000 0000
1	0000 0000 0000 0001	0000 0000 0000 0001
2	0000 0000 0000 0010	0000 0000 0000 0011
3	0000 0000 0000 0011	0000 0000 0000 0010
4	0000 0000 0000 0100	0000 0000 0000 0110
5	0000 0000 0000 0101	0000 0000 0000 0111
6	0000 0000 0000 0110	0000 0000 0000 0101
7	0000 0000 0000 0111	0000 0000 0000 0100
8	0000 0000 0000 1000	0000 0000 0000 1100
9	0000 0000 0000 1001	0000 0000 0000 1101
10	0000 0000 0000 1010	0000 0000 0000 1111
11	0000 0000 0000 1011	0000 0000 0000 1110
12	0000 0000 0000 1100	0000 0000 0000 1010
13	0000 0000 0000 1101	0000 0000 0000 1011
14	0000 0000 0000 1110	0000 0000 0000 1001
15	0000 0000 0000 1111	0000 0000 0000 1000
16	0000 0000 0001 0000	0000 0000 0001 1000
17	0000 0000 0001 0001	0000 0000 0001 1001
18	0000 0000 0001 0010	0000 0000 0001 1011
19	0000 0000 0001 0011	0000 0000 0001 1010
20	0000 0000 0001 0100	0000 0000 0001 1110
21	0000 0000 0001 0101	0000 0000 0001 1111
22	0000 0000 0001 0110	0000 0000 0001 1101
23	0000 0000 0001 0111	0000 0000 0001 1100
24	0000 0000 0001 1000	0000 0000 0001 0100
25	0000 0000 0001 1001	0000 0000 0001 0101
26	0000 0000 0001 1010	0000 0000 0001 0111
27	0000 0000 0001 1011	0000 0000 0001 0110
28	0000 0000 0001 1100	0000 0000 0001 0010
29	0000 0000 0001 1101	0000 0000 0001 0011
30	0000 0000 0001 1110	0000 0000 0001 0001
31	0000 0000 0001 1111	0000 0000 0001 0000
32	0000 0000 0010 0000	0000 0000 0011 0000
-	-	-
63	0000 0000 0010 1111	0000 0000 0010 0000
64	0000 0000 0100 1111	0000 0000 0110 0000
-	-	-
255	0000 00001111 1111	0000 0000 1000 0000

GBIN (Conversion: Gray Code → BIN)

■ Ladder diagram



■ Available operation units (●: Available)

Operation unit	bit	US	SS	UL	SL	SF	DF
i		●		●			

■ List of operands

Operand	Explanation
S	Device address where the data for conversion are stored, or constant
D	Storage device address

(Note) For gray codes, refer to the Correspondence Table: BIN / Gray Code.

■ Available devices (●: Available)

Operand	16-bit device											32-bit device *1			Integers			Real numbers		String	Index modifier *2
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX *3	K	U	H	SF	DF	" "	
S	●	●	●	●	●	●	●	●	●	●	●	●	●	●		●	●				●
D	●	●	●	●			●	●	●		●	●	●	●							●

*1: Operation unit: 16-bit integers (US) cannot be specified.
*2: Only 16-bit devices, and integer constants can be modified (real number constants, and character constants cannot be specified).
*3: Index register (I0 to IE)

■ Outline of operation

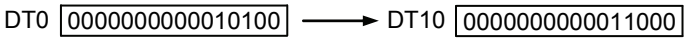
- Convert the device address value specified by [S] or constant, from gray code to binary data, and store the result in the device address specified by [D].

■ Process details

Example 1) Operation unit: 16 bits (US)

[i]…US

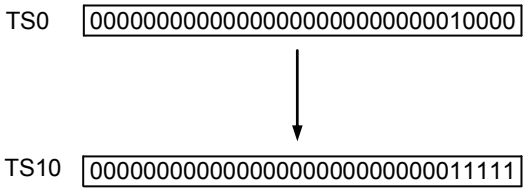
[S]…DT0 [D]…DT10



Example 2) Operation unit: 32 bits (UL)

[i]…UL

[S]…TS0 [D]…TS10

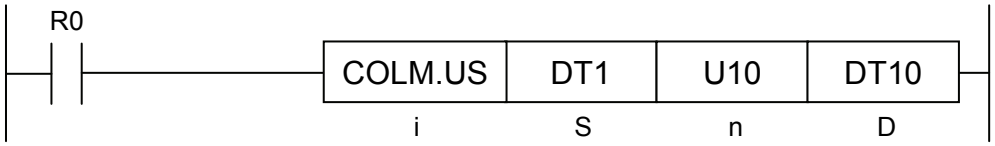


■ Flag operations

Name	Explanation
SR7 SR8 (ER)	To be set in the case of out-of-range in indirect access (index modification).

COLM (Conversion: Bit Line → Bit Column)

■ Ladder diagram



■ Available operation units (●: Available)

Operation unit	bit	US	SS	UL	SL	SF	DF
i		●					

■ List of operands

Operand	Explanation
S	Device address where the data for conversion are stored, or constant
n	Device address that stores the bit position specification, or constant (data available range: 0 to 15)
D	Starting address of the device to rewrite the bit column

■ Available devices (●: Available)

Operand	16-bit device											32-bit device *1			Integers			Real numbers		String	Index modifier *1
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX *3	K	U	H	SF	DF	" "	
S	●	●	●	●	●	●	●	●	●	●	●	●	●	●		●	●				●
n	●	●	●	●	●	●	●	●	●	●	●	●	●	●		●	●				●
D	●	●	●	●	●			●	●		●	●	●	●							●

*1: Only 16-bit devices, and integer constants can be modified (real number constants, and character constants cannot be specified).

■ Outline of operation

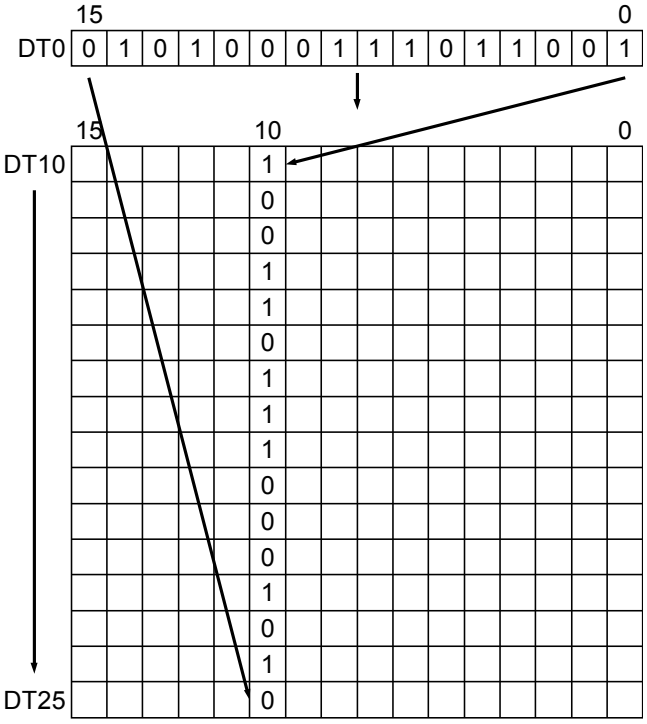
- The 16-bit line data specified by [S] are transferred to the [n] bit column in the 16-word device area specified by [D].
- Portions other than the specified bit column are not changed.

■ Process details

Example) Operation unit: 16 bits (US)

[i]…US

[S]…DT1 [n]…U10 [D]…DT10

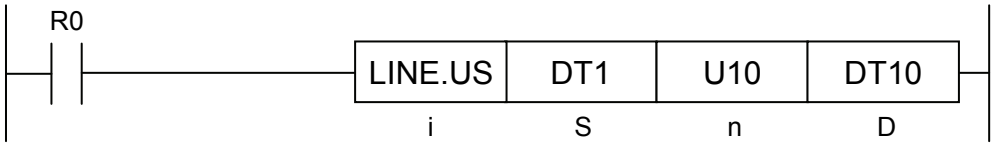


■ Flag operations

Name	Explanation
SR7 SR8 (ER)	To be set in the case of out-of-range in indirect access (index modification).
	To be set unless 0 ≤ n ≤ 15, where [n] is the bit position specification.
	To be set if, when the conversion result is stored in the device address specified by [D], it exceeds the area.

LINE (Conversion: Bit Column → Bit Line)

■ Ladder diagram



■ Available operation units (●: Available)

Operation unit	bit	US	SS	UL	SL	SF	DF
i		●					

■ List of operands

Operand	Explanation
S	Starting address of the device address to read the bit column
n	Device address that stores the bit position specification, or constant (data available range: 0 to 15)
D	Storage device address

■ Available devices (●: Available)

Operand	16-bit device											32-bit device			Integers			Real numbers		String	Index modifier *1
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX	K	U	H	SF	DF	" "	
S	●	●	●	●	●	●	●	●	●	●	●										●
n	●	●	●	●	●	●	●	●	●	●	●					●	●				●
D	●	●	●	●			●	●	●		●										●

*1: Only 16-bit devices, and integer constants can be modified (real number constants, and character constants cannot be specified).

■ Outline of operation

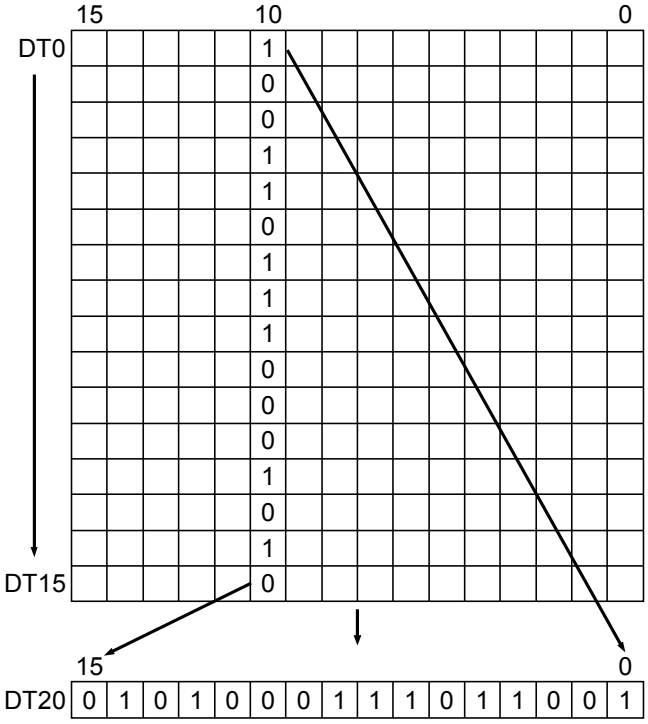
- The [n]-bit column data, in the 16-word device area specified by [S], are transferred to the 16-bit data specified by [D].

■ Process details

Example) Operation unit: 16 bits (US)

[i]…US

[S]…DT1 [n]…U10 [D]…DT20

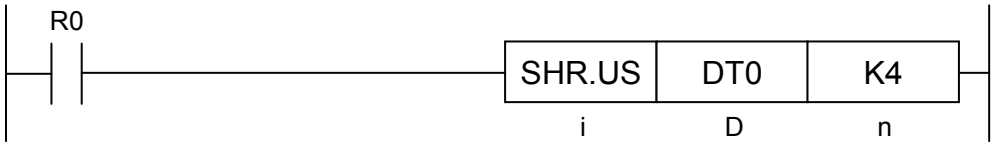


■ Flag operations

Name	Explanation
SR7 SR8 (ER)	To be set in the case of out-of-range in indirect access (index modification).
	To be set unless 0 ≤ n ≤ 15, where [n] is the bit position specification.
	To be set when the conversion range specified by [S] exceeds the device address.

SHR (n-bit Right Shift)

■ Ladder diagram



■ Available operation units (●: Available)

Operation unit	bit	US	SS	UL	SL	SF	DF
i		●	●	●	●		

■ List of operands

Operand	Explanation
D	Device address that stores the data to be shifted
n	Device address where the no. of shift bits is stored, or constant

■ Available devices (●: Available)

Operand	16-bit device											32-bit device *1			Integers			Real numbers		String	Index modifier *2
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX *3	K *4	U *5	H	SF	DF	" "	
D	●	●	●	●			●	●	●		●	●	●	●							●
n	●	●	●	●			●	●	●	●	●	●	●	●	●	●	●				●

*1: Operation unit: 16-bit integers (SS, US) cannot be specified.
*2: Only 16-bit devices, 32-bit devices, and integer constants can be modified (real number constants, and character constants cannot be specified).
*3: Index register (I0 to IE)
*4: Only operation unit: signed integers (SS, SL) can be specified.
*5: Only operation unit: unsigned integers (US, UL) can be specified.

■ Outline of operation

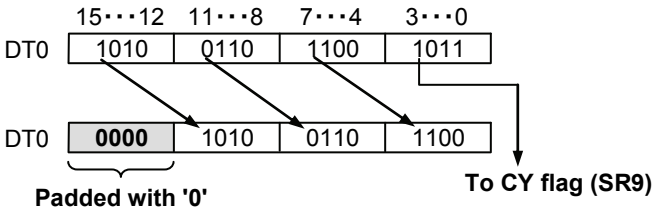
- Shift the data specified by [D] to the right (to the lower bit position), by the no. of bits specified by [n] (decimal specification).
- Following the shift, [n] bits from the highest bit are padded with 0. The data in the [n]th bit from the lowest bit are stored in SR9 (CY).
- Only the lower 8 bits in the [n] data are valid. The amount of shift is specified between 0 and 255 bits.

■ Process details

Example 1) Operation unit: 16 bits (US, SS)

[i]…US,SS

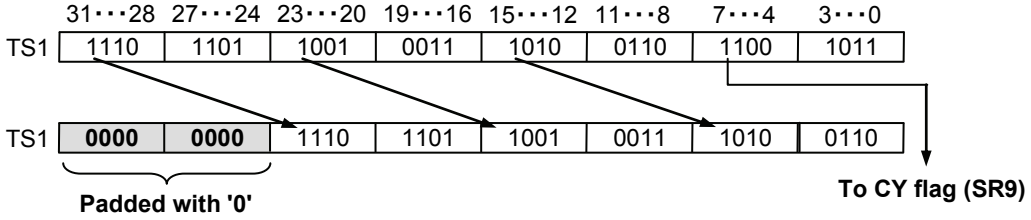
[D]…DT0 [n]…K4



Example 2) Operation unit: 32 bits (UL, SL)

[i]…UL,SL

[D]…TS1 [n]…K8

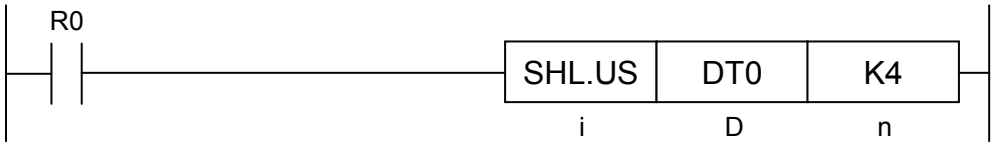


■ Flag operations

Name	Explanation
SR7 SR8 (ER)	To be set in the case of out-of-range in indirect access (index modification).
SR9(CY)	To be reset if the [n] (no. of shift bits) is larger than the operation unit. In other cases, data in the [n]th bit from the lowest bit are to be set.

SHL (n-bit Left Shift)

■ Ladder diagram



■ Available operation units (●: Available)

Operation unit	bit	US	SS	UL	SL	SF	DF
i		●	●	●	●		

■ List of operands

Operand	Explanation
D	Device address that stores the data to be shifted
n	Device address where the no. of shift bits is stored, or constant

■ Available devices (●: Available)

Operand	16-bit device											32-bit device *1			Integers			Real numbers		String	Index modifier *2
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX *3	K *4	U *5	H	SF	DF	" "	
D	●	●	●	●			●	●	●		●	●	●	●							●
n	●	●	●	●			●	●	●	●	●	●	●	●	●	●	●				●

*1: Operation unit: 16-bit integers (SS, US) cannot be specified.
*2: Only 16-bit devices, 32-bit devices, and integer constants can be modified (real number constants, and character constants cannot be specified).
*3: Index register (I0 to IE)
*4: Only operation unit: signed integers (SS, SL) can be specified.
*5: Only operation unit: unsigned integers (US, UL) can be specified.

■ Outline of operation

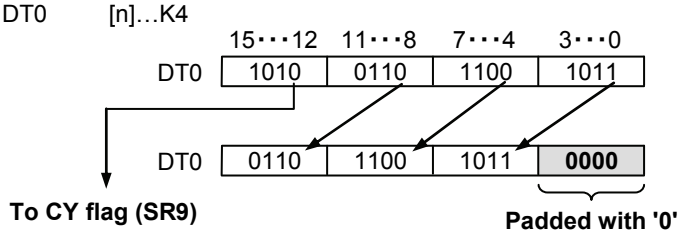
- Shift the data specified by [D] to the left (to the higher bit position), by the no. of bits specified by [n] (decimal specification).
- Following the shift, [n] bits from the lowest bit are padded with 0. The data in the [n]th bit from the highest bit are stored in SR9 (CY).
- Only the lower 8 bits in the [n] data are valid. The amount of shift is specified between 0 and 255 bits.

■ Process details

Example 1) Operation unit: 16 bits (US, SS)

[i]…US,SS

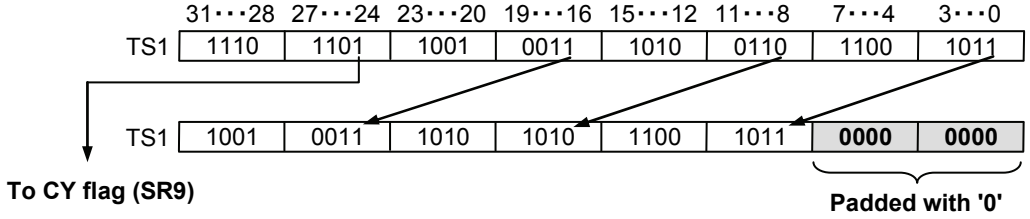
[D]…DT0 [n]…K4



Example 2) Operation unit: 32 bits (UL, SL)

[i]…UL,SL

[D]…SV1 [n]…K8

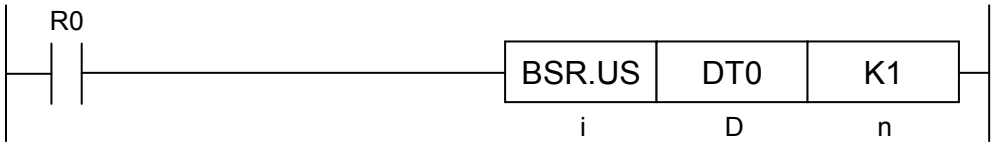


■ Flag operations

Name	Explanation
SR7 SR8 (ER)	To be set in the case of out-of-range in indirect access (index modification).
SR9(CY)	To be reset if the [n] (no. of shift bits) is larger than the operation unit. In other cases, data in the [n]th bit from the highest bit are to be set.

BSR (n-digit Right Shift)

■ Ladder diagram



■ Available operation units (●: Available)

Operation unit	bit	US	SS	UL	SL	SF	DF
i		●	●	●	●		

■ List of operands

Operand	Explanation
D	Device address that stores the data to be shifted
n	Device address where the no. of shift digits is stored, or constant

■ Available devices (●: Available)

Operand	16-bit device											32-bit device *1			Integers			Real numbers		String	Index modifier *2
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX *3	K *4	U *5	H	SF	DF	" "	
D	●	●	●	●			●	●	●		●	●	●	●							●
n	●	●	●	●			●	●	●	●	●	●	●	●	●	●	●				●

*1: Operation unit: 16-bit integers (SS, US) cannot be specified.
*2: Only 16-bit devices, 32-bit devices, and integer constants can be modified (real number constants, and character constants cannot be specified).
*3: Index register (I0 to IE)
*4: Only operation unit: signed integers (SS, SL) can be specified.
*5: Only operation unit: unsigned integers (US, UL) can be specified.

■ Outline of operation

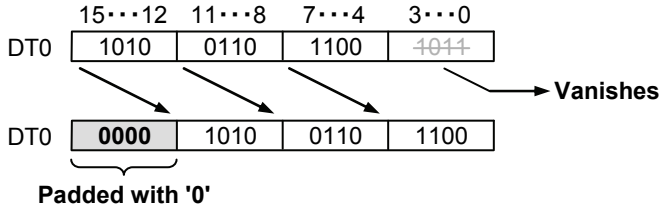
- Shift the data specified by [D] to the right (to the lower bit position), by the no. of digits specified by [n] (decimal specification) (4 bits).
- Following the shift, [n] digits from the highest pre-shift digit are padded with 0.
- Only the lower 8 bits in the [n] data are valid.
- If the operation unit is 16 bits (US, SS), the amount of shift is specified between 1 to 4 digits.
- If the operation unit is 32 bits (UL, SL), the amount of shift is specified between 1 to 8 digits.

■ Process details

Example 1) Operation unit: 16 bits (US, SS)

[i]…US,SS

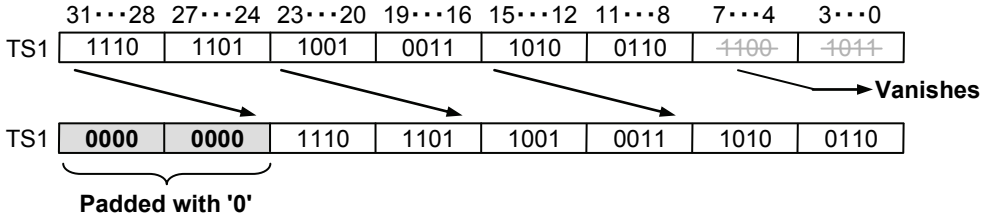
[D]…DT0 [n]…K1



Example 2) Operation unit: 32 bits (UL, SL, SF)

[i]…UL,SL,SF

[D]…TS1 [n]…K2



■ Precautions during programming

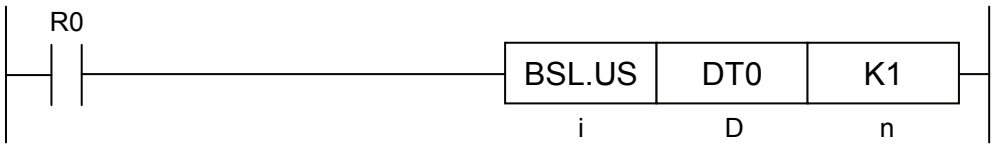
- The digit data that have been shifted out are discarded. There is no sub register for shift operation.

■ Flag operations

Name	Explanation
SR7 SR8 (ER)	To be set in the case of out-of-range in indirect access (index modification).

BSL (n-digit Left Shift)

■ Ladder diagram



■ Available operation units (●: Available)

Operation unit	bit	US	SS	UL	SL	SF	DF
i		●	●	●	●		

■ List of operands

Operand	Explanation
D	Device address that stores the data to be shifted
n	Device address where the no. of shift digits is stored, or constant

■ Available devices (●: Available)

Operand	16-bit device											32-bit device *1			Integers			Real numbers		String	Index modifier *2
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX *3	K *4	U *5	H	SF	DF	" "	
D	●	●	●	●			●	●	●		●	●	●	●							●
n	●	●	●	●			●	●	●	●	●	●	●	●	●	●	●				●

*1: Operation unit: 16-bit integers (SS, US) cannot be specified.
*2: Only 16-bit devices, 32-bit devices, and integer constants can be modified (real number constants, and character constants cannot be specified).
*3: Index register (I0 to IE)
*4: Only operation unit: signed integers (SS, SL) can be specified.
*5: Only operation unit: unsigned integers (US, UL) can be specified.

■ Outline of operation

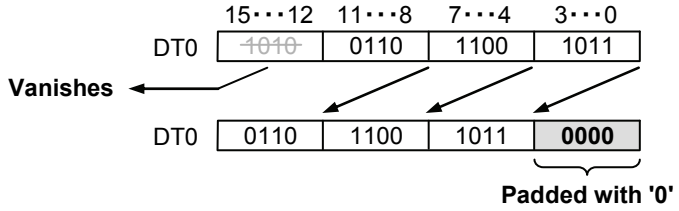
- Shift the data specified by [D] to the right (to the lower bit position), by the no. of digits specified by [n] (decimal specification) (4 bits).
- Following the shift, [n] digits from the highest pre-shift digit are padded with 0.
- Only the lower 8 bits in the n data are valid.
- If the operation unit is 16 bits (US, SS), the amount of shift is specified between 1 to 4 digits.
- If the operation unit is 32 bits (UL, SL), the amount of shift is specified between 1 to 8 digits.

■ Process details

Example 1) Operation unit: 16 bits (US, SS)

[i]…US,SS

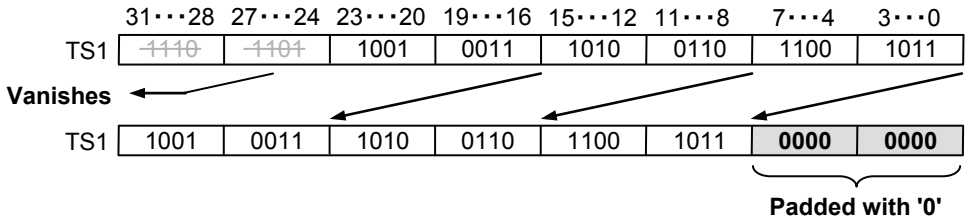
[D]…DT0 [n]…K1



Example 2) Operation unit: 32 bits (UL, SL)

[i]…US,SL

[D]…TS1 [n]…K2



■ Precautions during programming

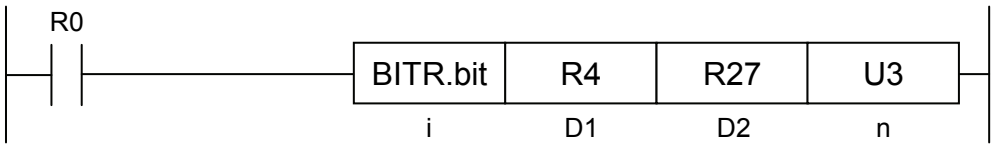
- The digit data that have been shifted out are discarded. There is no sub register for shift operation.

■ Flag operations

Name	Explanation
SR7 SR8 (ER)	To be set in the case of out-of-range in indirect access (index modification).

BITR (n-bit Right Shift of Multiple Devices)

■ Ladder diagram



■ Available operation units (●: Available)

Operation unit	bit	US	SS	UL	SL	SF	DF
i	●						

■ List of operands

Operand	Explanation
D1	Starting address of the devices to be shifted (data format: according to the operation unit)
D2	End address of the devices to be shifted (data format: according to the operation unit)
n	Device address where the no. of shift bits is stored, or constant

■ Available bit devices (●: Available)

Operand	Bit device											Bit specification of the word device			Index modifier
	X	Y	R	L	T	C	P	E	S	IN	OT	DT.b	LD.b	SD.b	
D1	○	○	○	○	-	-	-	-	-	-	○	○	○	-	○
D2	○	○	○	○	-	-	-	-	-	-	○	○	○	-	○

■ Available word devices (●: Available)

Operand	16-bit device											32-bit device *1			Integers			Real numbers		String	Index modifier *1
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX *3	K	U	H	SF	DF	" "	
n	●	●	●	●		●	●	●	●	●	●					●	●				●

*1: Only 16-bit devices, 32-bit devices, and integer constants can be modified (real number constants, and character constants cannot be specified)

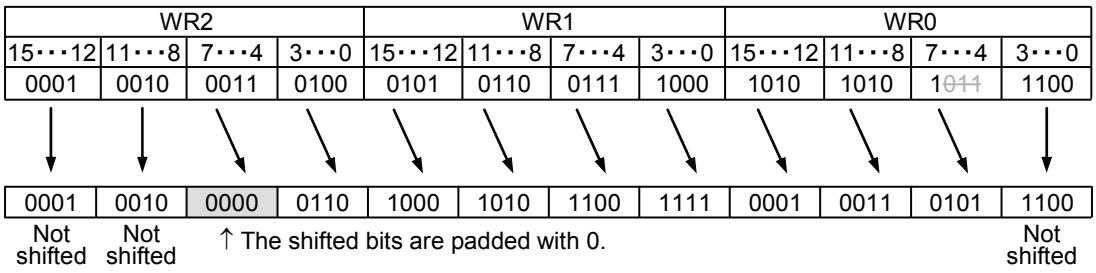
■ Outline of operation

- The range from [D1] to [D2] is shifted to the right by [n] bits.
- The starting address of the bit is specified by [D1], and the end address by [D2].
- Following the shift, the pre-shift lower [n] bits of [D1] vanish. The post-shift higher [n] bits of [D2] are padded with 0.
- The setting range of [n] is from 0 to 15. When [n] = 0, no shift takes place.

■ Process details

Example) Shift R4 through R27 by 3 bits

[D1]…R4 [D2]…R27 [n]…U3



■ Precautions during programming

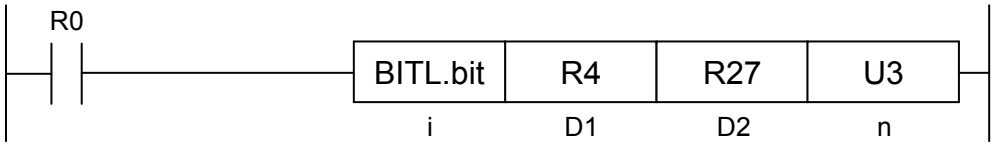
- In the case of a direct address and an index modification address, specify the same type of device for [D1] and [D2]. At the same time, specify [D2] ≥ [D1].

■ Flag operations

Name	Explanation
SR7 SR8 (ER)	To be set in the case of out-of-range in indirect access (index modification).
	To be set when [D1] > [D2].
	To be set when [n] ≥ 16.

BITL (n-bit Left Shift of Multiple Devices)

■ Ladder diagram



■ Available operation units (●: Available)

Operation unit	bit	US	SS	UL	SL	SF	DF
i	●						

■ List of operands

Operand	Explanation
D1	Starting address of the devices to be shifted (data format: according to the operation unit)
D2	End address of the devices to be shifted (data format: according to the operation unit)
n	Device address where the no. of shift bits is stored, or constant

■ Available bit devices (●: Available)

Operand	Bit device											Bit specification of the word device			Index modifier
	X	Y	R	L	T	C	P	E	S	IN	OT	DT.b	LD.b	SD.b	
D1	●	●	●	●							●	●	●		●
D2	●	●	●	●							●	●	●		●

■ Available word devices (●: Available)

Operand	16-bit device											32-bit device			Integers			Real numbers		String	Index modifier *1
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX	K	U	H	SF	DF	" "	
n	●	●	●	●			●	●	●	●	●					●	●				●

*1: Only 16-bit devices, 32-bit devices, and integer constants can be modified (real number constants, and character constants cannot be specified).

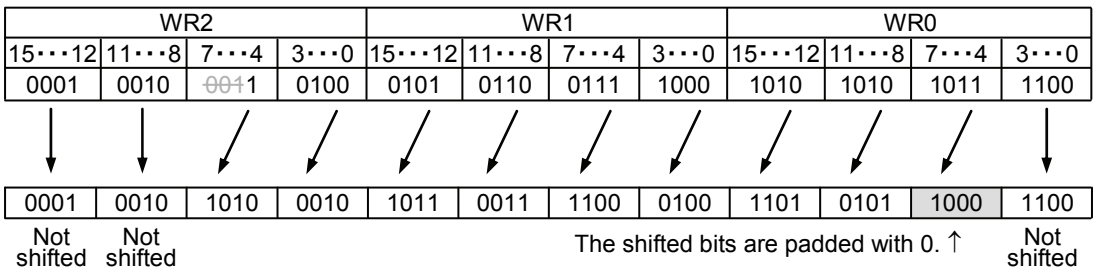
■ Outline of operation

- The range from [D1] to [D2] is shifted to the left by [n] bits.
- The starting address of the bit is specified by [D1], and the end address by [D2].
- Following the shift, the pre-shift higher [n] bits of [D1] vanish. The post-shift lower [n] bits of [D2] are padded with 0.
- The setting range of [n] is from 0 to 15. When [n] = 0, no shift takes place.

■ Process details

Example) Shift R4 through R27 by 3 bits

[D1]…R4 [D2]…R27 [n]…U3



■ Precautions during programming

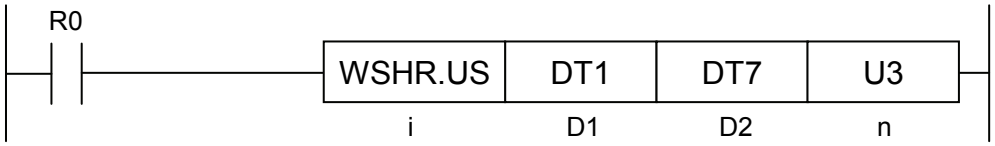
- In the case of a direct address and an index modification address, specify the same type of device for [D1] and [D2]. At the same time, specify [D2] ≥ [D1].

■ Flag operations

Name	Explanation
SR7 SR8 (ER)	To be set in the case of out-of-range in indirect access (index modification).
	To be set when [D1] > [D2].
	To be set when [n] ≥ 16.

WSHR (n-word Right Shift of a Block Area)

■ Ladder diagram



■ Available operation units (●: Available)

Operation unit	bit	US	SS	UL	SL	SF	DF
i		●	●				

■ List of operands

Operand	Explanation
D1	Starting address of the shift target
D2	End address of the shift target
n	No. of words to be right shifted (Data available range: 0 - 255 words)

■ Available devices (●: Available)

Operand	16-bit device											32-bit device			Integers			Real numbers		String	Index modifier
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX	K	U	H	SF	DF	" "	*1
D1	●	●	●	●			●	●	●		●										●
D2	●	●	●	●			●	●	●		●										●
n	●	●	●	●			●	●	●	●	●				●	●	●				●

*1: Only 16-bit devices, 32-bit devices, and integer constants can be modified (real number constants, and character constants cannot be specified).
*2: Only operation unit: signed integers (SS) can be specified.
*3: Only operation unit: unsigned integers (US) can be specified.

■ Outline of operation

- From the area specified by [D1] to the area specified by [D2] is right shifted by [n] words.
- From the starting address to the end address of the shift target is right shifted by the specified shift words.
- The specified shift words vanish from the starting address. The specified shift words in the end address are padded with H0.
- If the specified shift words are larger than the shift target range, the entire shift target range is padded with H0000.

■ Process details

Example 1) Operation unit: 16 bits (US)

[i]…US

[D1]…DT1 [D2]…DT7 [n]…U3

DT9	DT8	DT7	DT6	DT5	DT4	DT3	DT2	DT1	DT0	
5678	1234	**EEFF**	**CCDD**	**AABB**	**8899**	~~6677~~	~~4455~~	~~2233~~	0011	(HEX)

DT9	DT8	DT7	DT6	DT5	DT4	DT3	DT2	DT1	DT0	
5678	1234	**0000**	**0000**	**0000**	**EEFF**	**CCDD**	**AABB**	**8899**	0011	(HEX)

↑ The shifted bits are padded with H 0000.

Example 2) Operation unit: 16 bits (SS)

[i]…SS

[D1]…DT1 [D2]…DT7 [n]…K2

DT9	DT8	DT7	DT6	DT5	DT4	DT3	DT2	DT1	DT0	
5678	1234	**EEFF**	**CCDD**	**AABB**	**8899**	**6677**	~~4455~~	~~2233~~	0011	(HEX)

DT9	DT8	DT7	DT6	DT5	DT4	DT3	DT2	DT1	DT0	
5678	1234	**0000**	**0000**	**EEFF**	**CCDD**	**AABB**	**8899**	**6677**	0011	(HEX)

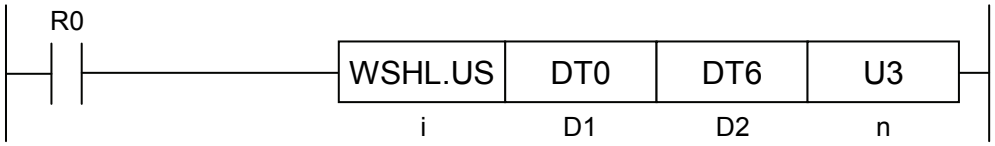
↑ The shifted bits are padded with H 0000.

■ Flag operations

Name	Explanation
SR7	To be set in the case of out-of-range in indirect access (index modification).
SR8	To be set when [D1] > [D2].
(ER)	To be set when [n] (specified shift words) is out of the available range.

WSHL (n-word Left Shift of a Block Area)

■ Ladder diagram



■ Available operation units (●: Available)

Operation unit	bit	US	SS	UL	SL	SF	DF
i		●	●				

■ List of operands

Operand	Explanation
D1	Starting address of the shift target
D2	End address of the shift target
n	No. of words to be right shifted (Data available range: 0 - 255 words)

■ Available devices (●: Available)

Operand	16-bit device											32-bit device			Integers			Real numbers		String	Index modifier *1
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX	K *2	U *3	H	SF	DF	" "	
D1	●	●	●	●			●	●	●		●										●
D2	●	●	●	●			●	●	●		●										●
n	●	●	●	●			●	●	●	●	●				●	●	●				●

*1: Only 16-bit devices, 32-bit devices, and integer constants can be modified (real number constants, and character constants cannot be specified).
*2: Only operation unit: signed integers (SS) can be specified.
*3: Only operation unit: unsigned integers (US) can be specified.

■ Outline of operation

- From the area specified by [D1] to the area specified by [D2] is left shifted by [n] words.
- From the starting address to the end address of the shift target is left shifted by the specified shift words.
- The specified shift words vanish from the end address. The specified shift words in the starting address are padded with H0.
- If the specified shift words are larger than the shift target range, the entire shift target range is padded with H0000.

■ Process details

Example 1) Operation unit: 16 bits (US)

[i]…US

[D1]…DT0 [D2]…DT6 [n]…U3

DT9	DT8	DT7	DT6	DT5	DT4	DT3	DT2	DT1	DT0	
5678	1234	EEFF	~~CCDD~~	~~AABB~~	~~8899~~	**6677**	**4455**	**2233**	**0011**	(HEX)
DT9	DT8	DT7	DT6	DT5	DT4	DT3	DT2	DT1	DT0	
5678	1234	EEFF	**6677**	**4455**	**2233**	**0011**	**0000**	**0000**	**0000**	(HEX)

↑ The shifted bits are padded with H 0000.

Example 2) Operation unit: 16 bits (SS)

[i]…SS

[D1]…DT1 [D2]…DT6 [n]…K2

DT9	DT8	DT7	DT6	DT5	DT4	DT3	DT2	DT1	DT0	
5678	1234	EEFF	~~CCDD~~	~~AABB~~	**8899**	**6677**	**4455**	**2233**	0011	(HEX)
DT9	DT8	DT7	DT6	DT5	DT4	DT3	DT2	DT1	DT0	
5678	1234	**EEFF**	**8899**	**6677**	**4455**	**2233**	**0000**	**0000**	0011	(HEX)

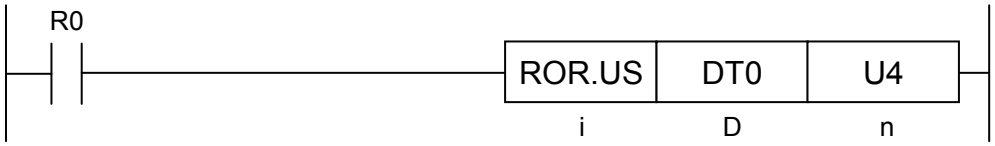
↑ The shifted bits are padded with H 0000.

■ Flag operations

Name	Explanation
SR7	To be set in the case of out-of-range in indirect access (index modification).
SR8	To be set when [D1] > [D2].
(ER)	To be set when [n] (specified shift words) is out of the available range.

ROR (Right Rotation of Data)

■ Ladder diagram



■ Available operation units (●: Available)

Operation unit	bit	US	SS	UL	SL	SF	DF
i		●		●			

■ List of operands

Operand	Explanation
D	Device address that stores the data to be rotated
n	Device address that stores the no. of rotation bits, or constant (data available range: 0 to 255)

■ Available devices (●: Available)

Operand	16-bit device											32-bit device *1			Integers			Real numbers		String	Index modifier *2
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX	K	U	H	SF	DF	" "	
D	●	●	●	●			●	●	●		●	●	●	●							●
n	●	●	●	●			●	●	●	●	●	●	●	●		●	●				●

*1 Operation unit: 16-bit integers (US) cannot be specified.
*2 Only 16-bit devices, 32-bit devices, and integer constants can be modified (real number constants, and character constants cannot be specified).

■ Outline of operation

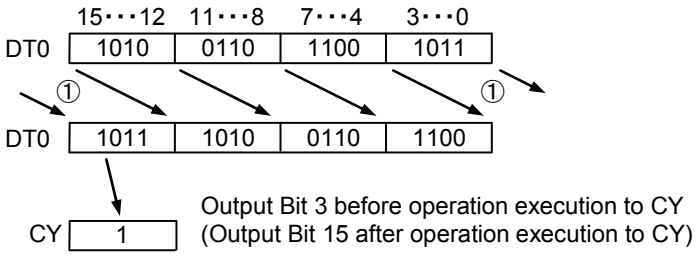
- Rotate the data specified by [D] to the right (to the lower bit position), by the no. of bits specified by [n] (decimal specification).
- Only the lower 8 bits in the [n] data are valid. The rotation amount is specified between 0 and 255 bits.
- (Rotation amount - 1) bits are output to SR9 (CY).
- When the operation unit is 16 bits (US), if [n] is either 0 or a multiple of 16, the rotation amount is regarded as 0, and this instruction is not executed.
- When the operation unit is 32 bits (UL), if [n] is either 0 or a multiple of 32, the rotation amount is regarded as 0, and this instruction is not executed.

■ Process details

Example 1) Operation unit: 16 bits (US)

[i]…US

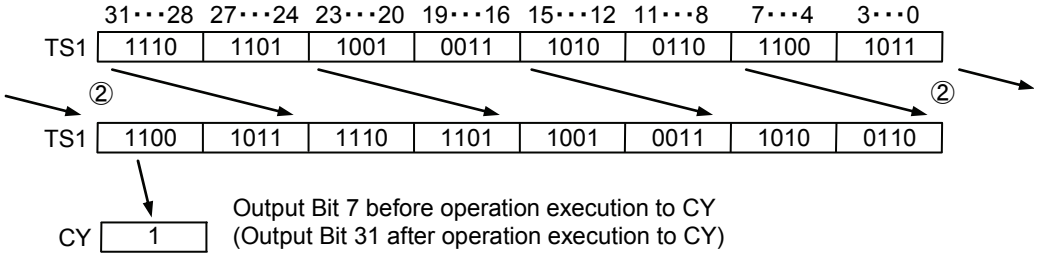
[D]…DT0 [n]…U4



Example 2) Operation unit: 32 bits (UL)

[i]…UL

[D]…TS1 [n]…U8

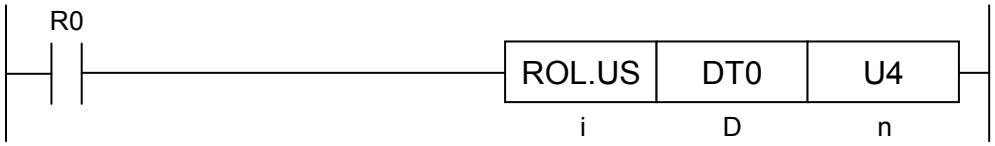


■ Flag operations

Name	Explanation
SR7 SR8 (ER)	To be set in the case of out-of-range in indirect access (index modification).
SR9(CY)	(Rotation amount - 1) bits of the pre-operation data are output. When the operation unit is 16 bits (US), if [n] is either 0 or a multiple of 16, the rotation amount is regarded as 0, and no change occurs. When the operation unit is 32 bits (UL), if [n] is either 0 or a multiple of 32, the rotation amount is regarded as 0, and no change occurs.

ROL (Left Rotation of Data)

■ Ladder diagram



■ Available operation units (●: Available)

Operation unit	bit	US	SS	UL	SL	SF	DF
i		●		●			

■ List of operands

Operand	Explanation
D	Device address that stores the data to be rotated
n	Device address that stores the no. of rotation bits, or constant (data available range: 0 to 255)

■ Available devices (●: Available)

Operand	16-bit device											32-bit device *1			Integers			Real numbers		String	Index modifier *2
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX	K	U	H	SF	DF	" "	
D	●	●	●	●			●	●	●		●	●	●	●							●
n	●	●	●	●			●	●	●	●	●	●	●	●		●	●				●

*1: Operation unit: 16-bit integers (US) cannot be specified.
*2: Only 16-bit devices, 32-bit devices, and integer constants can be modified (real number constants, and character constants cannot be specified).

■ Outline of operation

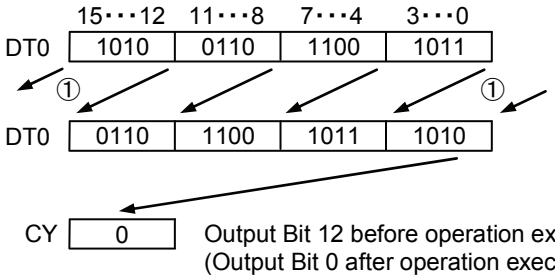
- Rotate the data specified by [D] to the left (to the higher bit position), by the no. of bits specified by [n] (decimal specification).
- Only the lower 8 bits in the [n] data are valid. The rotation amount is specified between 0 and 255 bits.
- (Bit length of the operation unit - rotation amount) bits are output to SR9 (CY).
- When the operation unit is 16 bits (US), if [n] is either 0 or a multiple of 16, the rotation amount is regarded as 0, and this instruction is not executed.
- When the operation unit is 32 bits (UL), if [n] is either 0 or a multiple of 32, the rotation amount is regarded as 0, and this instruction is not executed.

■ Process details

Example 1) Operation unit: 16 bits (US)

[i]…US

[D]…DT0 [n]…U4

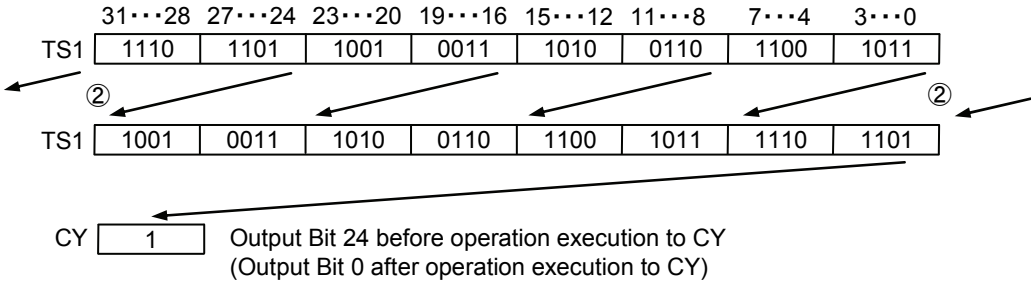


Output Bit 12 before operation execution to CY
(Output Bit 0 after operation execution to CY)

Example 2) Operation unit: 32 bits (UL)

[i]…UL

[D]…TS1 [n]…U8



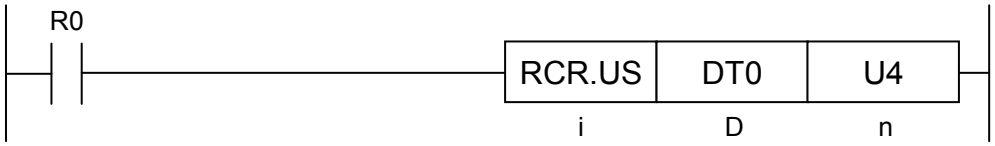
Output Bit 24 before operation execution to CY
(Output Bit 0 after operation execution to CY)

■ Flag operations

Name	Explanation
SR7 SR8 (ER)	To be set in the case of out-of-range in indirect access (index modification).
SR9(CY)	(Bit length of the operation unit - rotation amount) bits of the pre-operation data are output. When the operation unit is 16 bits (US), if [n] is either 0 or a multiple of 16, the rotation amount is regarded as 0, and no change occurs. When the operation unit is 32 bits (UL), if [n] is either 0 or a multiple of 32, the rotation amount is regarded as 0, and no change occurs.

RCR (Right Rotation of Data with Carry-Flag Data)

■ Ladder diagram



■ Available operation units (●: Available)

Operation unit	bit	US	SS	UL	SL	SF	DF
i		●		●			

■ List of operands

Operand	Explanation
D	Device address that stores the data to be rotated
n	Device address that stores the no. of rotation bits, or constant (data available range: 0 to 255)

■ Available devices (●: Available)

Operand	16-bit device											32-bit device *1			Integers			Real numbers		String	Index modifier *2
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX	K	U	H	SF	DF	" "	
D	●	●	●	●			●	●	●		●	●	●	●							●
n	●	●	●	●			●	●	●	●	●	●	●	●		●	●				●

*1: Operation unit: 16-bit integers (US) cannot be specified.
*2: Only 16-bit devices, 32-bit devices, and integer constants can be modified (real number constants, and character constants cannot be specified).

■ Outline of operation

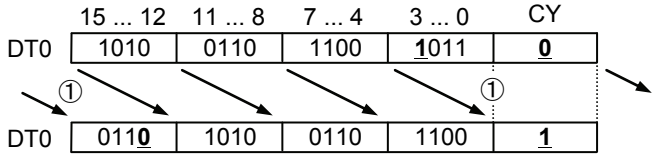
- Rotate the data specified by [D] to the right (to the lower bit position), by the no. of bits specified by [n] (decimal specification), with SR9 (CY).
- Only the lower 8 bits in the [n] data are valid. The rotation amount is specified between 0 and 255 bits.
- (Rotation amount - 1) bits are output to SR9 (CY).
- When the operation unit is 16 bits (US), if [n] is either 0 or a multiple of 17, the rotation amount is regarded as 0, and this instruction is not executed.
- When the operation unit is 32 bits (UL), if [n] is either 0 or a multiple of 33, the rotation amount is regarded as 0, and this instruction is not executed.

■ Process details

Example 1) Operation unit: 16 bits (US)

[i]…US

[D]…DT0 [n]…U4 CY…0

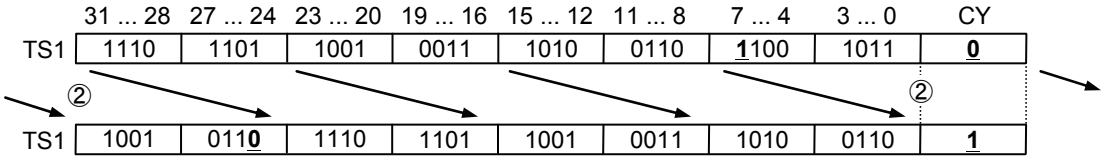


Output Bit 3 before operation execution to CY
Output CY before operation execution to Bit 12

Example 2) Operation unit: 32 bits (UL)

[i]…UL

[D]…TS1 [n]…U8 CY…0



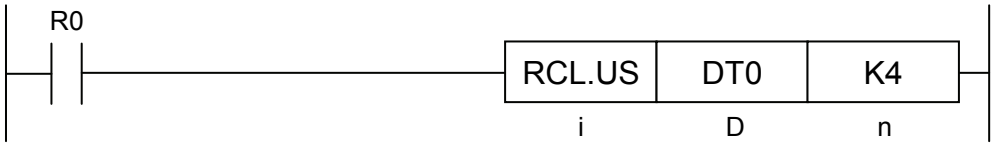
Output Bit 7 before operation execution to CY
Output CY before operation execution to Bit 24

■ Flag operations

Name	Explanation
SR7 SR8 (ER)	To be set in the case of out-of-range in indirect access (index modification).
SR9(CY)	(Rotation amount - 1) bits of the pre-operation data are output. When the operation unit is 16 bits (US), if [n] is either 0 or a multiple of 17, the rotation amount is regarded as 0, and no change occurs. When the operation unit is 32 bits (UL), if [n] is either 0 or a multiple of 33, the rotation amount is regarded as 0, and no change occurs.

RCL (Left Rotation of Data with Carry-Flag Data)

■ Ladder diagram



■ Available operation units (●: Available)

Operation unit	bit	US	SS	UL	SL	SF	DF
i		●		●			

■ List of operands

Operand	Explanation
D	Device address that stores the data to be rotated
n	Device address that stores the no. of rotation bits, or constant (data available range: 0 to 255)

■ Available devices (●: Available)

Operand	16-bit device											32-bit device *1			Integers			Real numbers		String	Index modifier *2
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX	K	U	H	SF	DF	" "	
D	●	●	●	●			●	●	●		●	●	●	●							●
n	●	●	●	●			●	●	●	●	●	●	●	●		●	●				●

*1: Operation unit: 16-bit integers (US) cannot be specified.
*2: Only 16-bit devices, 32-bit devices, and integer constants can be modified (real number constants, and character constants cannot be specified).

■ Outline of operation

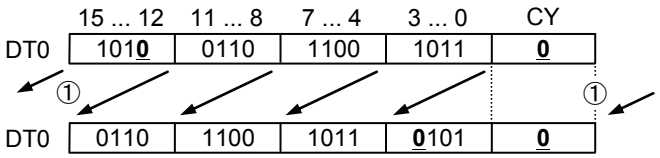
- Rotate the data specified by [D] to the left (to the higher bit position), by the no. of bits specified by [n] (decimal specification), with SR9 (CY).
- Only the lower 8 bits in the [n] data are valid. The rotation amount is specified between 0 and 255 bits.
- (Bit length of the operation unit - rotation amount) bits of are output to SR9 (CY).
- When the operation unit is 16 bits (US), if [n] is either 0 or a multiple of 17, the rotation amount is regarded as 0, and this instruction is not executed.
- When the operation unit is 32 bits (UL), if [n] is either 0 or a multiple of 33, the rotation amount is regarded as 0, and this instruction is not executed.

■ Process details

Example 1) Operation unit: 16 bits (US)

[i]…US

[D]…DT0 [n]…K4 CY…0

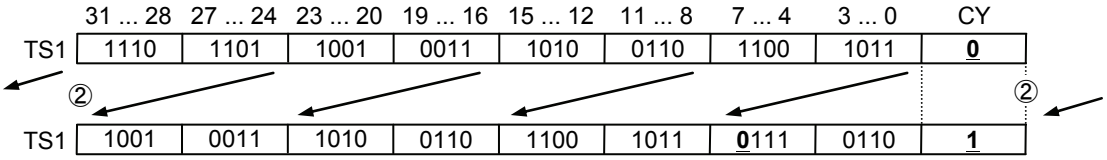


Output Bit 12 before operation execution to CY
Output CY before operation execution to Bit 3

Example 2) Operation unit: 32 bits (UL)

[i]…UL

[D]…TS1 [n]…K8 CY…0



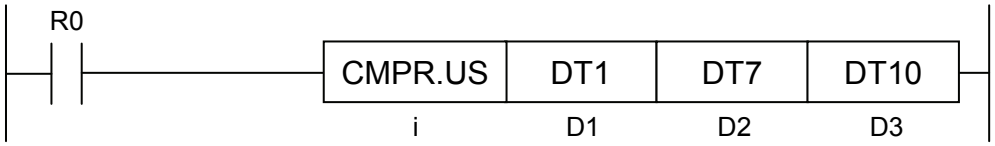
Output Bit 24 before operation execution to CY
Output CY before operation execution to Bit 7

■ Flag operations

Name	Explanation
SR7 SR8 (ER)	To be set in the case of out-of-range in indirect access (index modification).
SR9(CY)	(Bit length of the operation unit - rotation amount) bits of the pre-operation data are output. When the operation unit is 16 bits (US), if [n] is either 0 or a multiple of 17, the rotation amount is regarded as 0, and no change occurs. When the operation unit is 32 bits (UL), if [n] is either 0 or a multiple of 33, the rotation amount is regarded as 0, and no change occurs.

CMPR (Data Table Shift-out and Compress)

■ Ladder diagram



■ Available operation units (●: Available)

Operation unit	bit	US	SS	UL	SL	SF	DF
i		●	●				

■ List of operands

Operand	Explanation
D1	Starting address of the buffer
D2	End address of the buffer
D3	Device address to store the read data

■ Available devices (●: Available)

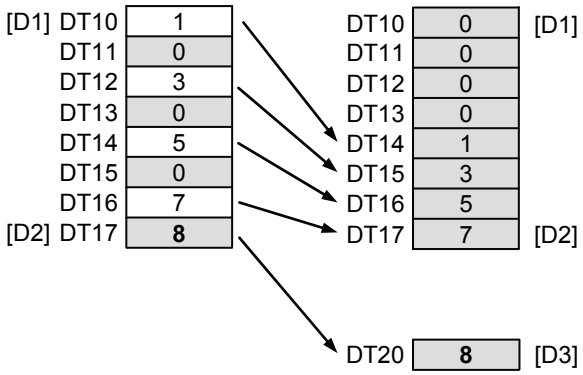
Operand	16-bit device											32-bit device			Integers			Real numbers		String	Index modifier
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX *1	K	U	H	SF	DF	" "	
D1	●	●	●	●			●	●	●		●										●
D2	●	●	●	●			●	●	●		●										●
D3	●	●	●	●			●	●	●		●										●

*1: Index register (I0 to IE)

■ Outline of operation

- According to the operation unit [i], [D2] is transferred to [D3], and the area specified by [D1] to [D2] is compressed.
  (Excluding the data transferred to [D3] during compression)
- The data in the specified area, excluding 0, are allocated in descending order from the higher address of the specified area, and the remaining area is cleared to zero.

Example of data table shift-out and compress when DT10, DT17 and DT20 are respectively specified for [D1], [D2] and [D3].



■ Process details

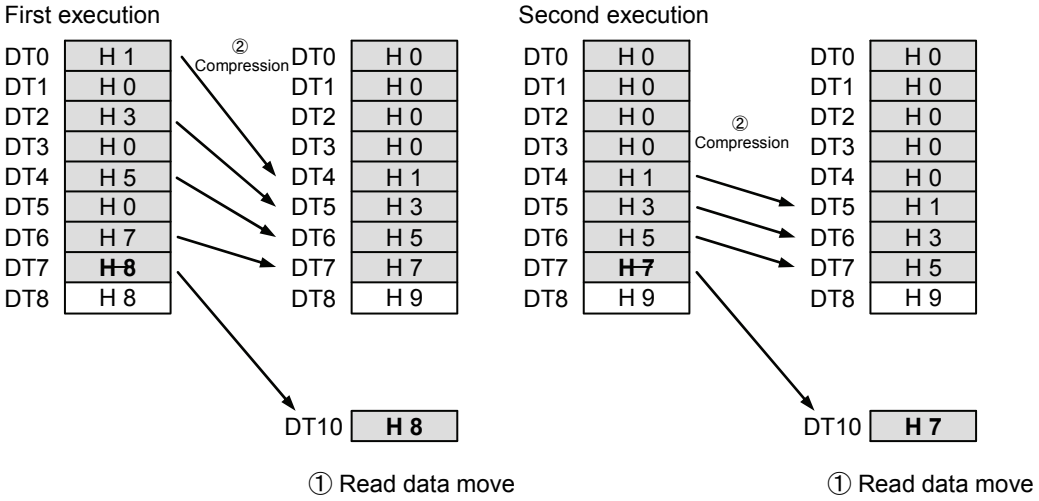
1) The buffer end is transferred to read data.

2) The data are compressed, excluding the data containing buffer end.

Example) Operation unit: 16 bits (US, SS) (executed twice)

[i]…US,SS

[D1]…DT1 [D2]…DT7 [D3]…DT10



■ Programming cautions

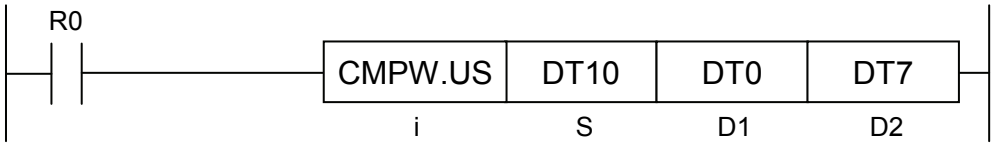
- In the case of a direct address and an index modification address, specify the same type of device for [D1] and [D2]. At the same time, specify [D2] ≥ [D1].

■ Flag operations

Name	Explanation
SR7	To be set in the case of out-of-range in indirect access (index modification).
SR8 (ER)	To be set when [D1] > [D2].

CMPW (Data Table Shift-In and Compress)

■ Ladder diagram



■ Available operation units (●: Available)

Operation unit	bit	US	SS	UL	SL	SF	DF
i		●	●				

■ List of operands

Operand	Explanation
S	Write data
D1	Starting address of the buffer
D2	End address of the buffer

■ Available devices (●: Available)

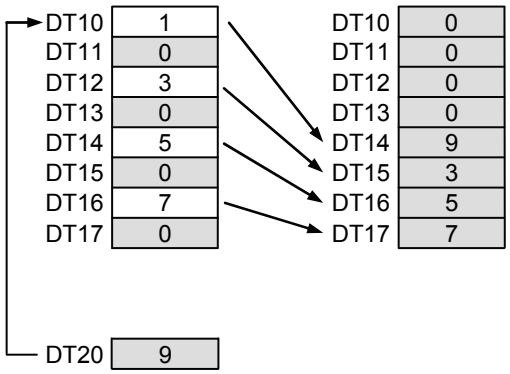
Operand	16-bit device											32-bit device			Integers			Real numbers		String	Index modifier
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX *2	K *3	U *4	H	SF	DF	" "	*1
S	●	●	●	●	●	●	●	●	●	●	●				●	●	●				●
D1	●	●	●	●			●	●													●
D2	●	●	●	●			●	●													●

*1: Only 16-bit devices, and integer constants can be modified.
*2: Index register (I0 to IE)
*3: Only operation unit: signed integers (SS) can be specified.
*4: Only operation unit: unsigned integers (US) can be specified.

■ Outline of operation

- According to the operation unit [n], [S] is transferred to [D1], and the area specified by [D1] to [D2] is compressed.
- The data in the specified area, excluding 0, are allocated in descending order from the higher address of the specified area, and the remaining area is cleared to zero.

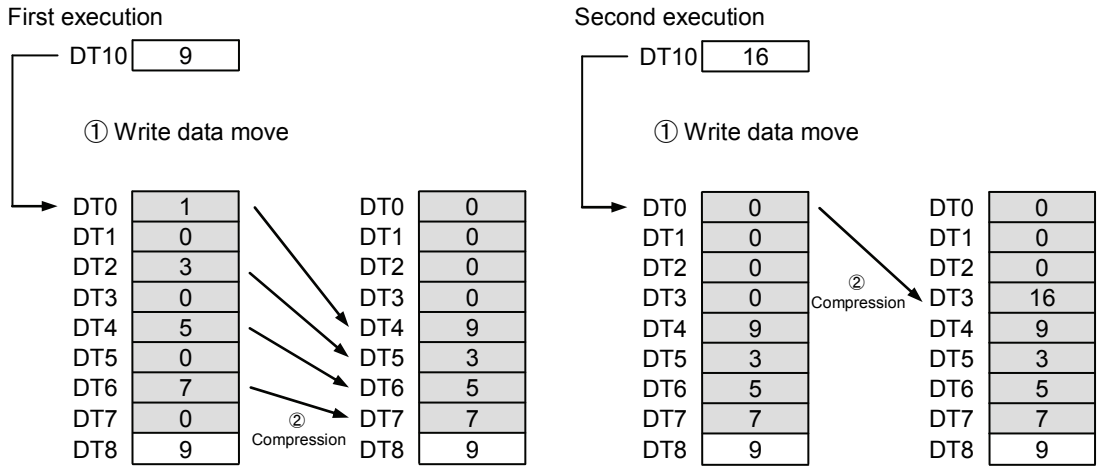
Example of data table shift-out and compress when DT10, DT17 and DT20 are respectively specified for [D1], [D2] and [D3].



■ Process details

1) The write data are transferred to the buffer start. (The starting data are overwritten.)
2) The data are compressed in the range from buffer start to buffer end.

Example) Operation unit: 16 bits (US, SS) (executed twice)
[i]…US,SS
[S]…DT10 [D1]…DT0 [D2]…DT7



■ Programming cautions

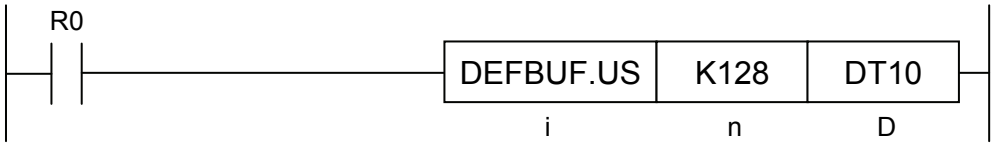
- In the case of a direct address and an index modification address, specify the same type of device for [D1] and [D2]. At the same time, specify [D2] ≥ [D1].

■ Flag operations

Name	Explanation
SR7	To be set in the case of out-of-range in indirect access (index modification).
SR8 (ER)	To be set when [D1] > [D2].

DEFBUF (Buffer Definition)

■ Ladder diagram



■ Available operation units (●: Available)

Operation unit	bit	US	SS	UL	SL	SF	DF
i		●	●				

■ List of operands

Operand	Explanation
n	Device address that specifies the buffer size, or constant (data available range: 1 to 4096)
D	Starting device address of the data buffer

■ Available devices (●: Available)

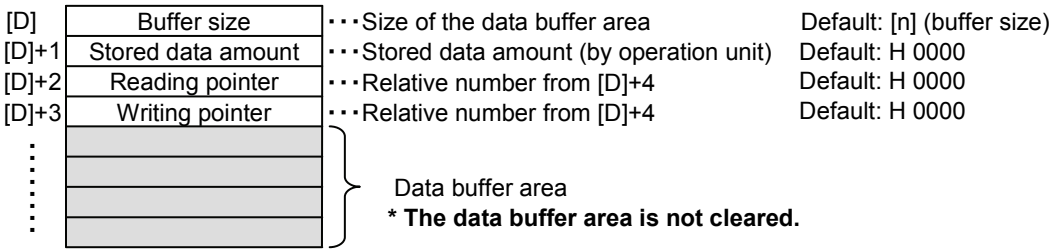
Operand	16-bit device											32-bit device			Integers			Real numbers		String	Index modifier
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX *2	K *3	U *4	H	SF	DF	" "	*1
n	●	●	●	●			●	●	●		●				●	●	●				●
D							●	●													●

*1: Only 16-bit devices, and integer constants can be modified.
*2: Index register (I0 to IE)
*3: Only operation unit: signed integers (SS, SL) can be specified.
*4: Only operation unit: unsigned integers (US, UL) can be specified.

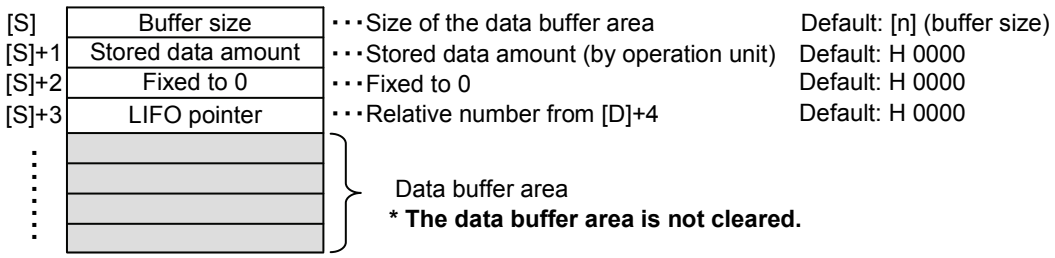
■ Outline of operation

- According to the operation unit [i], data buffer for [n] data are defined, starting with the [D] area.
- From ([D]+1) (usable size) to ([D]+3) (writing pointer) are initialized (cleared to zero).

■ Format of data buffer (FIFO buffer)



■ Format of data buffer (LIFO buffer)



■ Process details

1) [n] (buffer size) is specified in [D] (buffer start).

2) From ([D]+1) (stored data amount) to ([D]+3) (writing pointer) are cleared to zero.

Example) Operation unit: 16 bits (US, SS)

[i]…US,SS

[n]…K 128(U 128) [D]…DT0

DT0	**K 0**		DT0	**K 128**	(Buffer size)
DT1	**K 1**		DT1	**K 0**	(Stored data amount)
DT2	**K 2**		DT2	**K 0**	(Reading pointer)
DT3	**K 3**		DT3	**K 0**	(Writing pointer)
DT4	K 4		DT4	K 4	
DT5	K 5		DT5	K 5	
DT6	K 6		DT6	K 6	
DT7	K 7		DT7	K 7	
DT8	K 8		DT8	K 8	

■ Related instructions

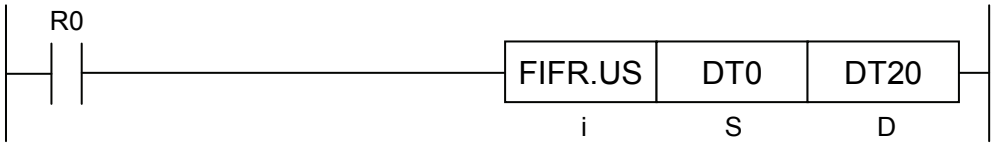
- FIFR (Read data from the 16- or 32-bit data buffer (First-In-First-Out))
- BUFW (Write data in the 16- or 32-bit data buffer)
- LIFR (Read data from the 16- or 32-bit data buffer (Last-In-First-Out))

■ Flag operations

Name	Explanation
SR7	To be set in the case of out-of-range in indirect access (index modification).
SR8	To be set when [n] (buffer size) is out of the available range.
(ER)	To be set when the range [D] (buffer start) + [n] (buffer size) is out of the available range.

FIFR (Data Read (First-In-First-Out))

■ Ladder diagram



■ Available operation units (●: Available)

Operation unit	bit	US	SS	UL	SL	SF	DF
i		●	●				

■ List of operands

Operand	Explanation
S	Starting device address of the data buffer
D	Device address of the read data

■ Available devices (●: Available)

Operand	16-bit device											32-bit device			Integers			Real numbers		String	Index modifier
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX *2	K	U	H	SF	DF	" "	*1
S							●	●													●
D	●	●	●	●			●	●	●		●										●

*1: Only 16-bit devices, and integer constants can be modified.
*2: Index register (I0 to IE)

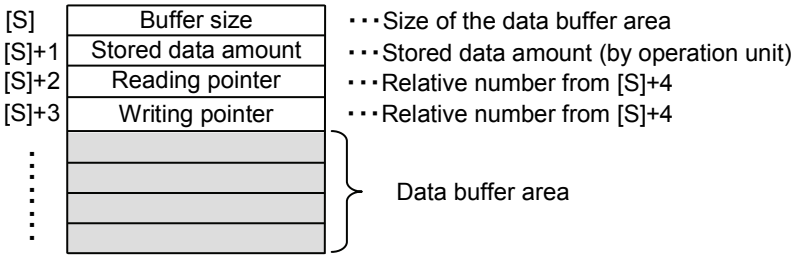
■ Outline of operation

- Data are read from the FIFO buffer specified by [S], and set to [D].
  (In the [S] buffer area, it is necessary to define buffer first using the DEFBUF instruction.)
- Pre-execution buffer consistency check (An operation error occurs in the following cases.)

1) [S] (buffer size) > 4096, or [S] (buffer size) = 0
2) [S]+1 (stored data amount) = 0
3) [S]+1 (stored data amount) > [S] (buffer size)
4) [S]+2 (reading pointer) > [S] (buffer size)
5) Buffer area exceeds the upper limit of the specified device.

- According to the operation unit [i], the data of the area specified by [S]+2 (reading pointer) are set to [D].
- Increment (+1) [S]+2 (reading pointer).
- Following the increment (+1), if [S]+2 (reading pointer) = [S] (buffer size), 0 is set to [S]+2 (reading pointer).
- Decrement (-1) [S]+1 (stored data amount).

■ Format of data buffer (FIFO buffer)



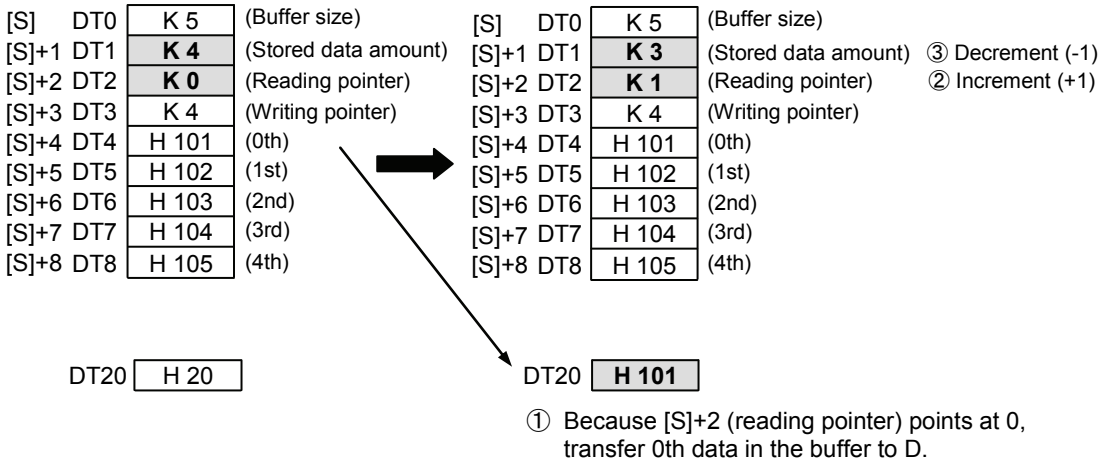
■ Process details

1) Set the area specified by ([S]+2) (reading pointer) to [D] (read data).

2) Increment (+1) [S]+2 (reading pointer).

3) Decrement (-1) [S]+1 (stored data amount).

Example) Operation unit: 16 bits (US, SS)
[i]…US,SS
[S]…DT0 [D]…DT20

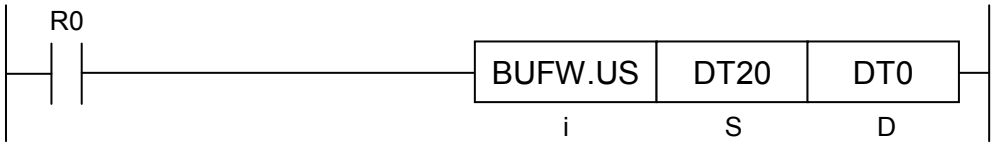


■ Flag operations

Name	Explanation
SR7 SR8 (ER)	To be set in the case of out-of-range in indirect access (index modification).
	To be set when [S] (buffer size) > 4096, or [S] (buffer size) = 0.
	To be set when [S]+1 (stored data amount) = 0.
	To be set when [S]+1 (stored data amount) > [S] (buffer size).
	To be set when [S] +2 (reading pointer) > [S] (buffer size).
	To be set when the buffer area exceeds the upper limit of the specified device.

BUFW (Data Write)

■ Ladder diagram



■ Available operation units (●: Available)

Operation unit	bit	US	SS	UL	SL	SF	DF
i		●	●				

■ List of operands

Operand	Explanation
S	Device address of the write data, or constant
D	Starting device address of the data buffer

■ Available devices (●: Available)

Operand	16-bit device											32-bit device			Integers			Real numbers		String	Index modifier
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX *2	K *3	U *4	H	SF	DF	" "	*1
S	●	●	●	●	●	●	●	●	●	●	●				●	●	●				●
D							●	●													●

*1: Only 16-bit devices, and integer constants can be modified.
*2: Index register (I0 to IE)
*3: Only operation unit: signed integers (SS, SL) can be specified.
*4: Only operation unit: unsigned integers (US, UL) can be specified.

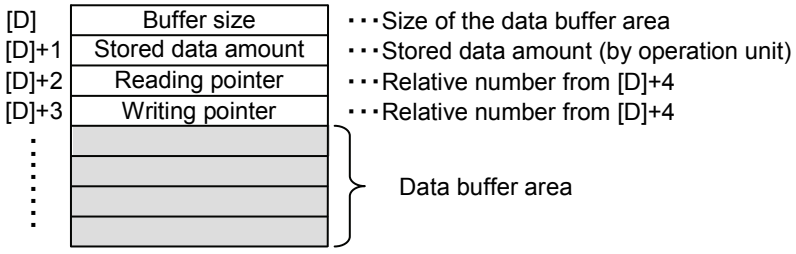
■ Outline of operation

- Set the data specified by [S] to the buffer specified by [D].
  (In the [D] buffer area, it is necessary to define buffer first using the DEFBUF instruction.)
- Pre-execution buffer consistency check (An operation error occurs in the following cases.)

1) [D] (buffer size) > 4096, or [D] (buffer size) = 0
2) [D]+1 (stored data amount) ≥ [D] (buffer size)
3) [D]+3 (writing pointer) ≥ [D] (buffer size)
4) Buffer area exceeds the upper limit of the specified device.

- According to the operation unit [i], [S] is set to the area specified by [D]+3 (writing pointer).
- Increment (+1) [D]+3 (writing pointer).
- Following the increment (+1), if [D]+3 (writing pointer) = [S] (buffer size),
- 0 is set to [D]+3 (writing pointer).
- Increment (+1) [S]+1 (stored data amount).

■ Format of data buffer (FIFO)



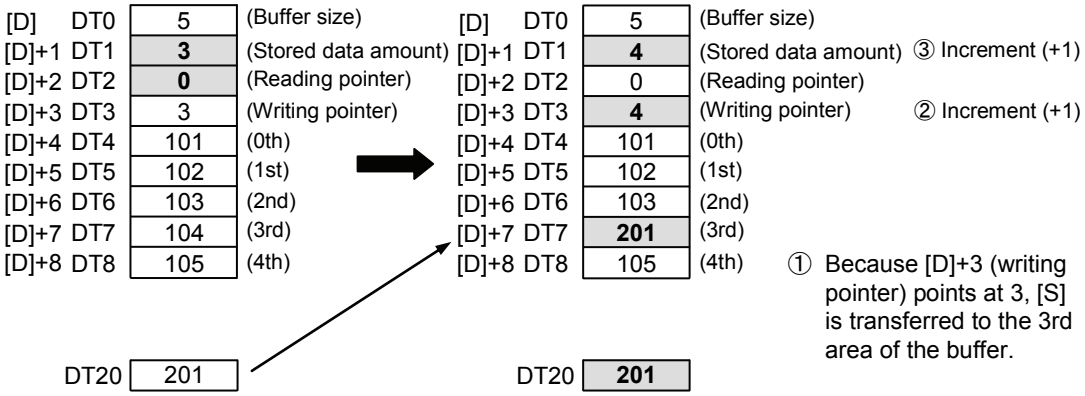
■ Process details

1) Set [S] (write data) to the area specified by [D]+3 (writing pointer).

2) Increment (+1) [D]+3 (writing pointer).

3) Increment (+1) [D]+1 (stored data amount).

Example) 16 bits (US, SS)
[i]…US,SS
[S]…DT20 [D]…DT0

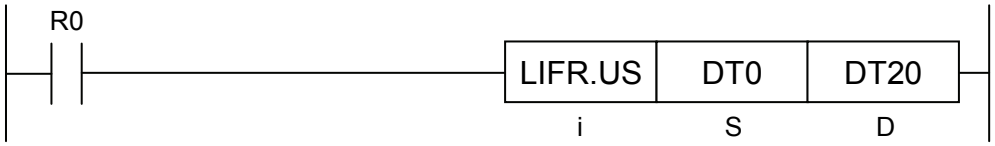


■ Flag operations

Name	Explanation
SR7 SR8 (ER)	To be set in the case of out-of-range in indirect access (index modification).
	To be set when [D] (buffer size) > 4096, or [D] (buffer size) = 0.
	To be set when [D] +1 (stored data amount) ≥ [D] (buffer size).
	To be set when [D] +3 (writing pointer) ≥ [D] (buffer size).
	To be set when the buffer area exceeds the upper limit of the specified device.

LIFR (Data Read (Last-In-First-Out))

■ Ladder diagram



■ Available operation units (●: Available)

Operation unit	bit	US	SS	UL	SL	SF	DF
i		●	●				

■ List of operands

Operand	Explanation
S	Starting device address of the data buffer
D	Device address of the read data

■ Available devices (●: Available)

Operand	16-bit device											32-bit device			Integers			Real numbers		String	Index modifier
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX *2	K	U	H	SF	DF	" "	*1
S							●	●													●
D	●	●	●	●			●	●	●		●										●

*1: Only 16-bit devices, and integer constants can be modified.
*2: Index register (I0 to IE)

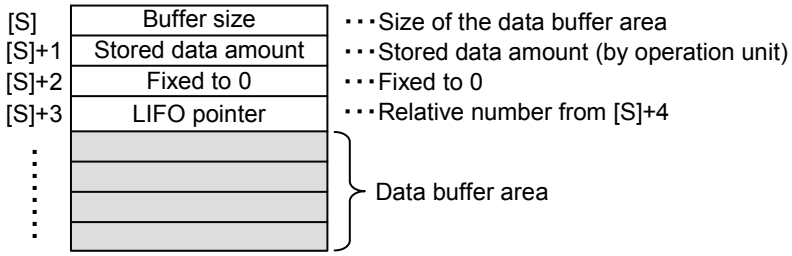
■ Outline of operation

- Data are read from the LIFO buffer specified by [S], and set to [D].
  (In the [S] buffer area, it is necessary to define buffer first using the DEFBUF instruction.)
- Pre-execution buffer consistency check (An operation error occurs in the following cases.)

1) [S] (buffer size) > 4096, or [S] (buffer size) = 0
2) [S]+1 (stored data amount) = 0
3) [S]+2 ≠ 0
4) [S]+1 (stored data amount) > [S] (buffer size)
5) [S] + 3 (LIFO pointer) ≥ [S] (buffer size)
6) Buffer area exceeds the upper limit of the specified device.

- If [S]+3 (LIFO pointer) is 0, set [S] (buffer size) to [S]+3 (LIFO pointer).
- Decrement (-1) [S]+3 (LIFO pointer).
- According to the operation unit [i], the data of the area specified by [S]+3 (LIFO pointer) are set to [D].
- Decrement (-1) [S]+1 (stored data amount).

■ Format of data buffer (LIFO)



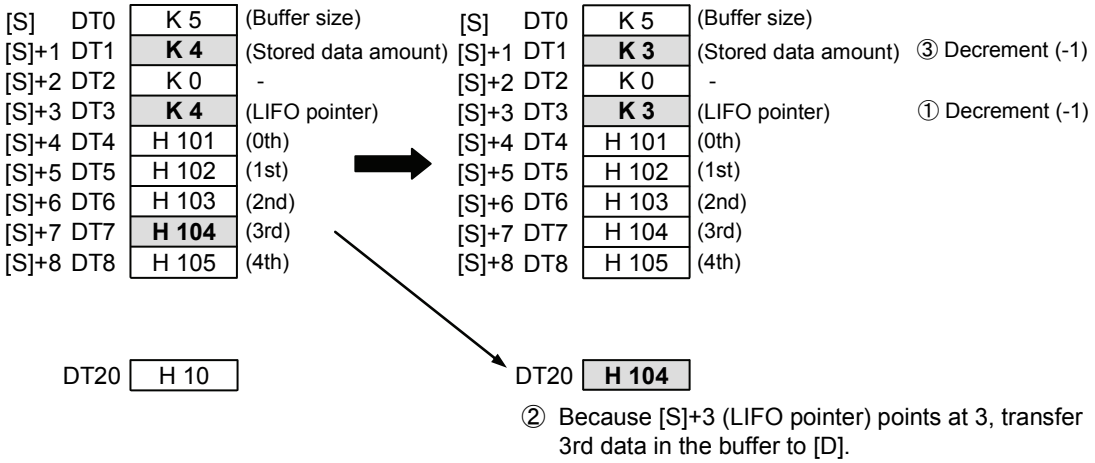
■ Process details

1) Decrement (-1) [S]+3 (LIFO pointer).

2) Set the data of the area specified by [S]+3 (LIFO pointer) to [D] (read data).

3) Decrement (-1) [S]+1 (stored data amount).

Example) 16 bits (US, SS)
[i]…US,SS
[S]…DT0 [D]…DT20



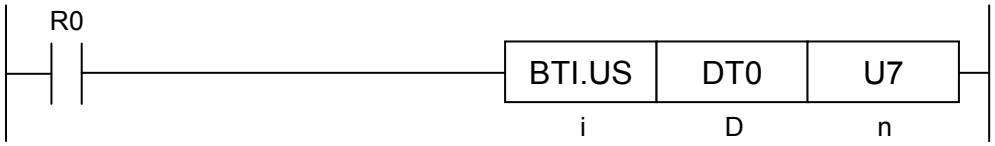
② Because [S]+3 (LIFO pointer) points at 3, transfer 3rd data in the buffer to [D].

■ Flag operations

Name	Explanation
SR7 SR8 (ER)	To be set in the case of out-of-range in indirect access (index modification).
	To be set when [S] (buffer size) > 4096, or [S] (buffer size) = 0.
	To be set when [S]+1 (stored data amount) = 0.
	To be set when [S]+2 is other than 0.
	To be set when [S]+1 (stored data amount) > [S] (buffer size).
	To be set when [S]+3 (LIFO pointer) ≥ [S] (buffer size).
	To be set when the buffer area exceeds the upper limit of the specified device.

BTI (Bit Inversion)

■ Ladder diagram



■ Available operation units (●: Available)

Operation unit	bit	US	SS	UL	SL	SF	DF
i		●					

■ List of operands

Operand	Explanation
D	Inversion target data (device address)
n	Bit number (device address or constant) (data available range: 0 to 15)

■ Available devices (●: Available)

Operand	16-bit device											32-bit device			Integers			Real numbers		String	Index modifier
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX *2	K	U	H	SF	DF	" "	*1
S							●	●													●
D	●	●	●	●			●	●	●		●										●

*1: Only 16-bit devices, and integer constants can be modified.
*2: Index register (I0 to IE)

■ Outline of operation

- According to the operation unit [i], invert the [n]th bit in the area specified by [D].

■ Process details

Example) Operation unit: 16 bits (US)
[i]…US
[D]…DT0 [n]…U7 <Invert Bit 7>

	DT0															
Bit	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0
BIN	0	1	0	1	1	0	0	0	**0**	1	1	1	0	0	0	0

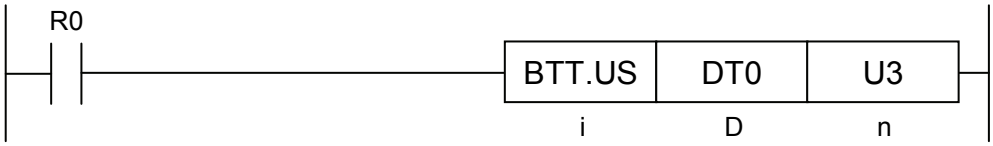
	DT0															
Bit	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0
BIN	0	1	0	1	1	0	0	0	**1**	1	1	1	0	0	0	0

■ Flag operations

Name	Explanation
SR7	To be set in the case of out-of-range in indirect access (index modification).
SR8 (ER)	To be set when [n] is out of the range.

BTT (Bit Test)

■ Ladder diagram



■ Available operation units (●: Available)

Operation unit	bit	US	SS	UL	SL	SF	DF
i		●					

■ List of operands

Operand	Explanation
D	Test target data (device address)
n	Bit number (device address or constant) (data available range: 0 to 15)

■ Available devices (●: Available)

Operand	16-bit device											32-bit device			Integers			Real numbers		String	Index modifier *1
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX	K	U	H	SF	DF	" "	
D	●	●	●	●			●	●	●		●										●
n	●	●	●	●			●	●	●	●	●					●	●				●

*1: Only 16-bit devices, and 32-bit devices can be modified (integer constants, real number constants, and character constants cannot be specified).

■ Outline of operation

- According to the operation unit [i], test (assess ON or OFF of) the [n]th bit in the area specified by [D], and output the result to SRB (=).

■ Process details

Depending on the state of the specified bit, the results of SRB (=) become as follows.

State of the specified bit	SRB(=)
ON(1)	OFF(0)
OFF(0)	ON(1)

Example 1) Operation unit: 16 bits (US) (SRB is OFF)

[i]…US
[D]…DT0 [n]…U3

	DT0															
Bit	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0
BIN	0	1	0	1	1	0	0	0	0	1	1	1	**1**	0	0	0

Flag operations during execution

State of the specified bit	SRB(=)
ON(1)	OFF(0)

Example 2) Operation unit: 16 bits (US) (SRB is ON)

[i]…US
[D]…DT0 [n]…U3

	DT0															
Bit	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0
BIN	0	1	0	1	1	0	0	0	0	1	1	1	**0**	0	0	0

Flag operations during execution

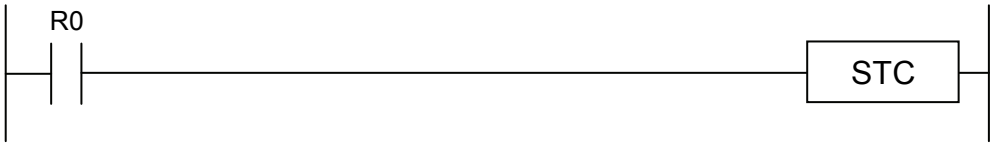
State of the specified bit	SRB(=)
OFF(0)	ON(1)

■ Flag operations

Name	Explanation
SR7 SR8 (ER)	To be set in the case of out-of-range in indirect access (index modification).
	To be set when [n] is out of the range.
SRB(＝)	To be set when the test bit (Bit [n]) is '0'.
	To be reset when the test bit (Bit [n]) is '1'.

STC (Carry-Flag Set)

■ Ladder diagram



■ Available operation units (●: Available)

- No operation units

■ List of operands

- None

■ Available devices (●: Available)

- None

■ Outline of operation

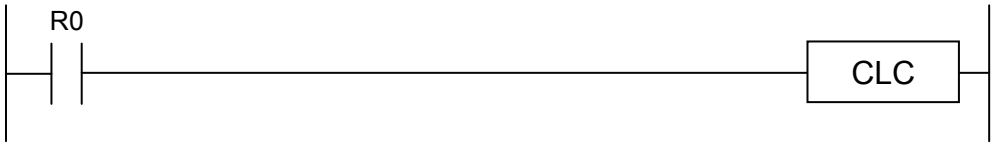
- Switch SR9 (CY) on.

■ Flag operations

Name	Explanation
SR9 (CY)	To be set after this instruction is executed.

CLC (Carry-Flag Reset)

■ Ladder diagram



■ Available operation units (●: Available)

- No operation units

■ List of operands

- None

■ Available devices (●: Available)

- None

■ Outline of operation

- Switch SR9 (CY) off.

■ Flag operations

Name	Explanation
SR9 (CY)	To be reset after this instruction is executed.

SSET (Conversion: Character Constant → ASCII Code)

■ Ladder diagram

■ Available operation units (●: Available)

- No operation units

■ List of operands

Operand	Explanation
S	Source string
D	Destination starting device address

■ Available devices (●: Available)

Operand	16-bit device											32-bit device			Integers			Real numbers		String	Index modifier *1
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX	K	U	H	SF	DF	" "	
S																				●	
D	●	●	●	●			●	●	●												●

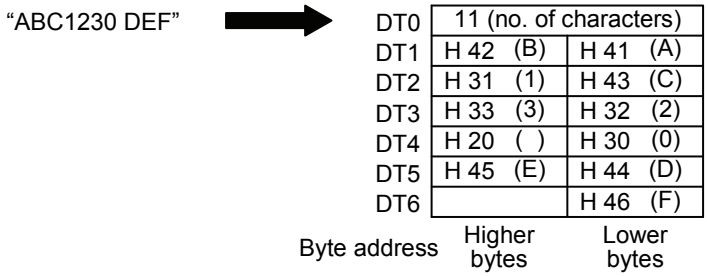
*1: Index register (I0 to IE)

■ Outline of operation

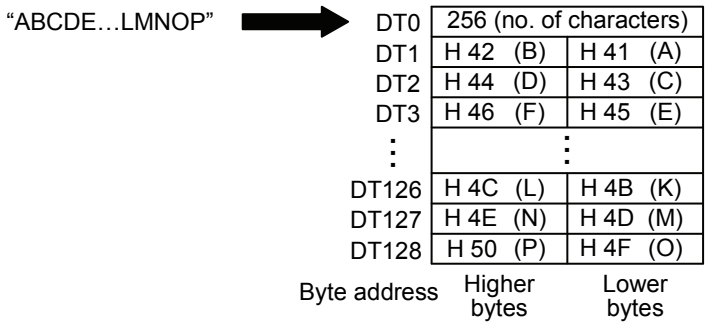
- Convert the character constant specified by [S] to an ASCII code. The result is stored in the area starting with [D].
- After setting the no. of characters in 1 word at the start of the area, and store the character data in the subsequent area.
- Put a character constant between "" (double quotations) for specification.
- From 1 to 256 characters can be specified for a character constant.
- No NULL (00), etc. should be appended to the specified characters.

■ Process details

Example 1) Set the 11 characters of the string "ABC1230 DEF" in DT0



Example 2) Set 256 characters to DT0, repeating the set of the 16 characters from A to P



■ Precautions during programming

- Character data in the pre-operation [D] area are overwritten.

■ Flag operations

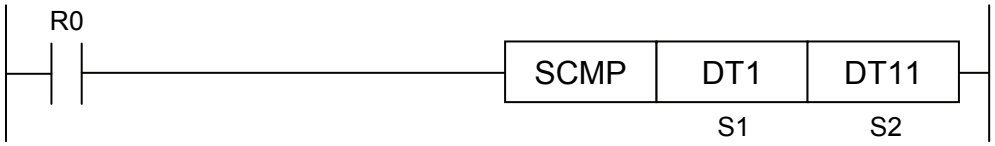
Name	Explanation
SR7 SR8 (ER)	To be set in the case of out-of-range in indirect access (index modification).
	To be set when the destination range is out of the available range.

■ Reference Table: ASCII Codes

b7	b6	b5	b4	b3	b2	b1	b0	R \ C	0	1	2	3	4	5	6	7
				0	0	0	0	0	NUL	DEL	SPACE	0	@	P	`	p
				0	0	0	1	1	SOH	DC_1	!	1	A	Q	a	q
				0	0	1	0	2	STX	DC_2	"	2	B	R	b	r
				0	0	1	1	3	ETX	DC_3	#	3	C	S	c	s
				0	1	0	0	4	EOT	DC_4	$	4	D	T	d	t
				0	1	0	1	5	ENQ	NAK	%	5	E	U	e	u
				0	1	1	0	6	ACK	SYN	&	6	F	V	f	v
				0	1	1	1	7	BEL	ETB	'	7	G	W	g	w
				1	0	0	0	8	BS	CAN	(	8	H	X	h	x
				1	0	0	1	9	HT	EM	)	9	I	Y	i	y
				1	0	1	0	A	LF	SUB	*	:	J	Z	j	z
				1	0	1	1	B	VT	ESC	+	;	K	[	k	{
				1	1	0	0	C	FF	FS	,	<	L	¥	l	|
				1	1	0	1	D	CR	GS	-	=	M	]	m	}
				1	1	1	0	E	SO	RS	.	>	N	^	n	~
				1	1	1	1	F	SI	US	/	?	O	_	o	DEL

SCMP (Comparison of Strings)

■ Ladder diagram



■ Available operation units (●: Available)

- No operation units

■ List of operands

Operand	Explanation
S1	String 1 to be compared
S2	String 2 to be compared

■ Available devices (●: Available)

Operand	16-bit device											32-bit device			Integers			Real numbers		String	Index modifier
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX *1	K	U	H	SF	DF	" "	
S1	●	●	●	●			●	●	●												●
S2	●	●	●	●			●	●	●												●

*1: Index register (I0 to IE)

■ Outline of operation

- Compare the string specified by [S1] and the string specified by [S2]. The comparison result is output to the system relays SRA to SRC (assessment flags for the comparison instruction).

● System relays SRA to SRC are as follows, based on their relative sizes.

Relationship between [S1] and [S2]	Assessment flag for the comparison instruction		
	SRA	SRB	SRC
	>	=	<
[S1] < [S2]	OFF	OFF	ON
[S1] = [S2]	OFF	ON	OFF
[S1] > [S2]	ON	OFF	OFF

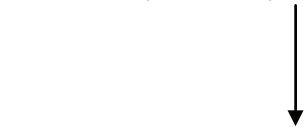
● Relative sizes with differing nos. of characters are as follows.

[S1]	Relative size	[S2]
“ABCDE”	=	“ABCDE”
“ABCD”	<	“ABCDE”
“B”	>	“ABCDE”

■ Process details

Example) Compare DT1 and DT11

DT0	4 (no. of characters)		DT10	5 (no. of characters)	
DT1	“B”	“A”	DT11	“B”	“A”
DT2	“D”	“C”	DT12	“D”	“C”
DT3			DT13		“E”
DT4			DT14		
DT5			DT15		
DT6					
Byte address	Higher bytes	Lower bytes	Byte address	Higher bytes	Lower bytes



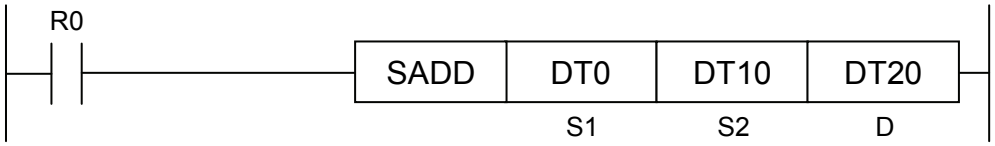
Relationship between [S1] and [S2]	Assessment flag for the comparison instruction		
	SRA	SRB	SRC
	>	=	<
[S1] < [S2]	OFF	OFF	ON

■ Flag operations

Name	Explanation
SR7 SR8 (ER)	To be set in the case of out-of-range in indirect access (index modification).
	To be set when the string range specified by [S1] or [S2] is out of the available range.
SRA (>)	Varies depending on the comparison result.
SRB(＝)	
SRC (<)	

SADD (String Addition)

■ Ladder diagram



■ Available operation units (●: Available)

- No operation units

■ List of operands

Operand	Explanation
S1	Starting device address of String 1 to be connected
S2	Starting device address of String 2 to be connected
D	Starting device address to store the connected string

■ Available devices (●: Available)

Operand	16-bit device											32-bit device			Integers			Real numbers		String	Index modifier
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX *1	K	U	H	SF	DF	" "	
S1	●	●	●	●			●	●	●												●
S2	●	●	●	●			●	●	●												●
D	●	●	●	●			●	●	●												●

*1: Index register (I0 to IE)

■ Outline of operation

- Connect the string specified by [S1] and the string specified by [S2], and set the connected string to the device address specified by [D].
- The post-connection max. no. of characters is 4096.

■ Process details

Example) Combine the strings of DT0 and DT10, and set the result to DT20.

	DT0	5 (no. of characters)	
	DT1	"B"	"A"
	DT2	"D"	"C"
	DT3		"E"
+	DT4		
	DT5		
	DT6		
Byte address		Higher bytes	Lower bytes

	DT10	3 (no. of characters)	
	DT11	"2"	"1"
	DT12		"3"
=	DT13		
	DT14		
	DT15		
	DT16		
Byte address		Higher bytes	Lower bytes

	DT20	8 (no. of characters)	
	DT21	"B"	"A"
	DT22	"D"	"C"
	DT23	"1"	"E"
	DT24	"3"	"2"
	DT25		
	DT26		
Byte address		Higher bytes	Lower bytes

■ Precautions during programming

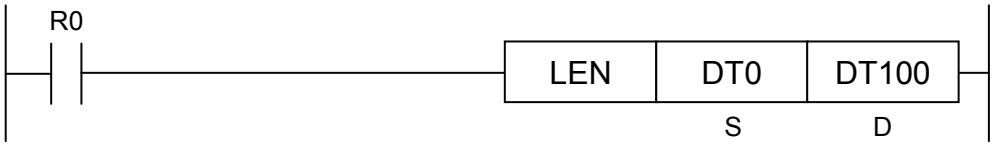
- Character data in the pre-operation [D] area are overwritten.

■ Flag operations

Name	Explanation
SR7 SR8 (ER)	To be set in the case of out-of-range in indirect access (index modification).
	To be set when the string range specified by [S1] and [S2] is out of the available range.
	To be set when the destination range is out of the available range.
	To be set when the connected string exceeds the max. no. of characters.

LEN (Obtainment of String Length)

■ Ladder diagram



■ Available operation units (●: Available)

- No operation units

■ List of operands

Operand	Explanation
S	Starting device address of the string
D	Starting device address to store the string length

■ Available devices (●: Available)

Operand	16-bit device											32-bit device			Integers			Real numbers		String	Index modifier
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX *1	K	U	H	SF	DF	" "	
S	●	●	●	●			●	●	●												●
D	●	●	●	●			●	●	●		●										●

■ Outline of operation

- The no. of characters stored at the start of the string specified by [S] is set to the device address specified by [D].

■ Process details

Example) Set the no. of characters of DT0 in DT100

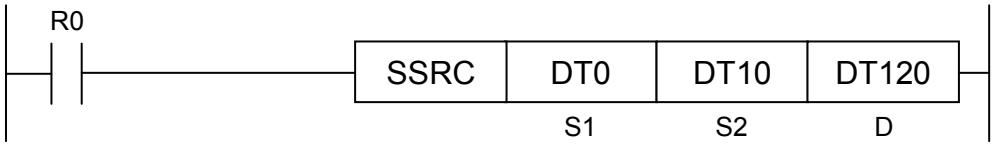
DT0	8 (no. of characters)	→ DT100: 8
DT1	**"B"**	**"A"**
DT2	**"D"**	**"C"**
DT3	**"1"**	**"E"**
DT4	**"3"**	**"2"**
DT5		
Byte address	Higher bytes	Lower bytes

■ Flag operations

Name	Explanation
SR7	To be set in the case of out-of-range in indirect access (index modification).
SR8	To be set when the obtained no. of characters exceeds 4096.
(ER)	To be set when the string range specified by [S] is out of the available range.

SSRC (String Search)

■ Ladder diagram



■ Available operation units (●: Available)

- No operation units

■ List of operands

Operand	Explanation
S1	Starting device address of the string data to be searched for
S2	Starting device address of the string to be searched
D	Starting device address to store the search result

■ Available devices (●: Available)

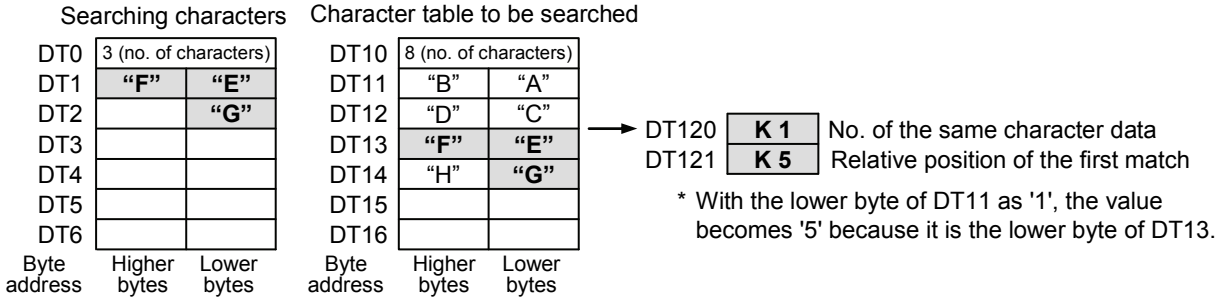
Operand	16-bit device											32-bit device			Integers			Real numbers		String	Index modifier
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX *1	K	U	H	SF	DF	" "	
S1	●	●	●	●			●	●	●												●
S2	●	●	●	●			●	●	●												●
D	●	●	●	●			●	●	●		●										●

■ Outline of operation

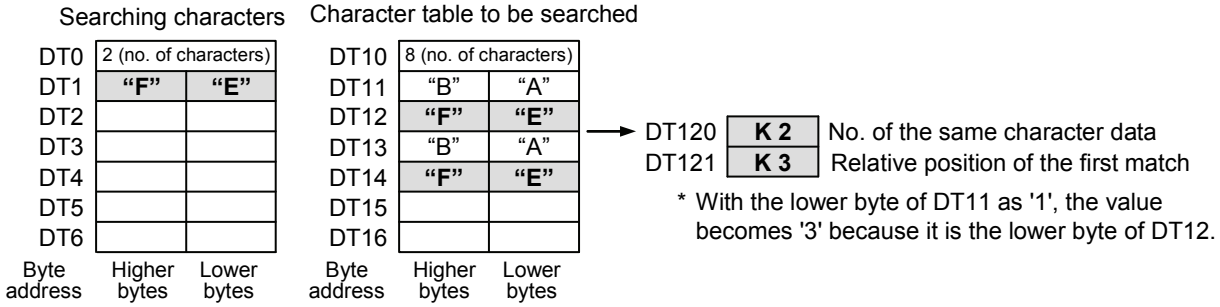
- Search the string specified by [S2] for the character data specified by [S1].
- Following the search, the no. of the same character data is stored in the device address specified by [D], and the relative position of the first match (in bytes) is stored in [D]+1.

■ Process details

Example 1) Combine the strings of DT0 and DT10, and set the result to DT20.



Example 2) Two searching character data are searched for



■ Precautions during programming

- Specify the no. of characters to be searched for in [S1] (no. of characters of the string to be searched for).

Example)

DT0	2 (no. of characters)	
DT1	"B"	"A"
DT2	"D"	"C"

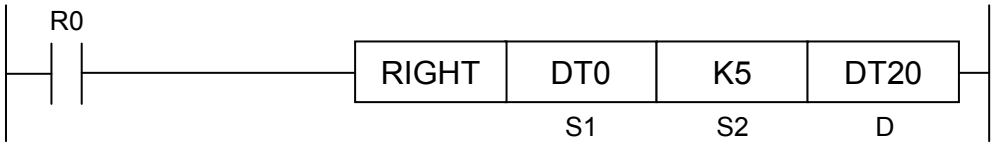
A letter 'A' is searched for when the no. of characters is specified as '1'.
A unit of letters 'AB' is searched for when the no. of characters is specified as '2'.

■ Flag operations

Name	Explanation
SR7 SR8 (ER)	To be set in the case of out-of-range in indirect access (index modification).
	To be set when the no. of characters in [S1] > no. of characters in [S2].
	To be set when the string range specified by [S1] or [S2] is out of the available range.

RIGHT (Takeout of the Right Side of a String)

■ Ladder diagram



■ Available operation units (●: Available)

- No operation units

■ List of operands

Operand	Explanation
S1	Starting device address of the source data
S2	No. of characters to be taken out (data available range: 1 to 4096)
D	Starting device address to store the takeout result

■ Available devices (●: Available)

Operand	16-bit device											32-bit device			Integers			Real numbers		String	Index modifier
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX *1	K	U	H	SF	DF	" "	
S1	●	●	●	●			●	●	●												●
S2	●	●	●	●	●	●	●	●	●	●	●				●	●	●				●
D	●	●	●	●			●	●	●												●

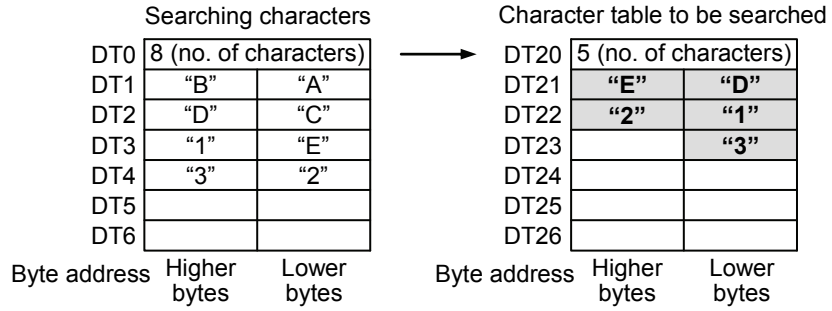
■ Outline of operation

- Take out characters as specified by [S2] from the right side (end of the character data) of the string specified by [S1], and store them in the device address specified by [D].

■ Process details

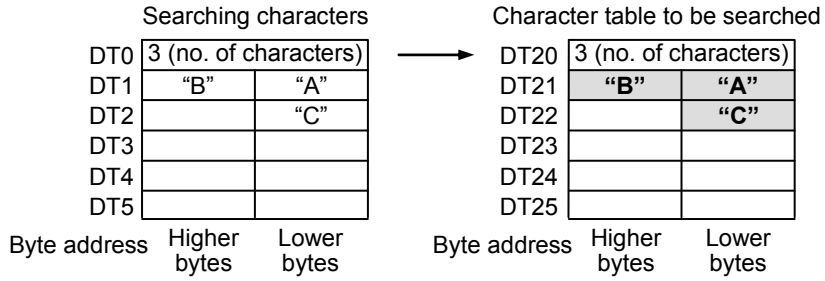
Example 1) Take out the last five characters from the the DT0 string, and transfer them to DT20

[S1]…DT0 [S2]…K5 [D]…DT20



Example 2) No. of characters in [S2] > No. of characters in the [S1] string

[S1]…DT0 [S2]…K7 [D]…DT20



■ Precautions during programming

- Character data in the pre-operation [D] area are overwritten.
- When the no. of characters in [S2] > no. of characters in the string of [S1], the no. of characters to be transferred is limited to that of [S1].

■ Flag operations

Name	Explanation
SR7	To be set in the case of out-of-range in indirect access (index modification).
SR8	To be set when the destination range is out of the available range.
(ER)	To be set when [S2] (no. of characters) is out of the range.

LEFT (Takeout of the Left Side of a String)

■ Ladder diagram

■ Available operation units (●: Available)

- No operation units

■ List of operands

	Explanation
S1	Starting device address of the source data
S2	No. of characters to be taken out (data available range: 1 to 4096)
D	Starting device address to store the takeout result

■ Available devices (●: Available)

Operand	16-bit device											32-bit device			Integers			Real numbers		String	Index modifier
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX *1	K	U	H	SF	DF	" "	
S1	●	●	●	●			●	●	●												●
S1	●	●	●	●	●	●	●	●	●	●	●				●	●	●				●
D	●	●	●	●			●	●	●												●

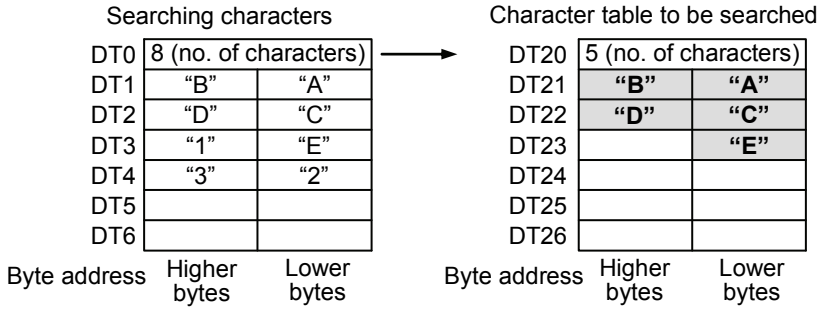
■ Outline of operation

- Take out characters as specified by [S2] from the left side (start of the character data) of the string specified by [S1], and store them in the device address specified by [D].

■ Process details

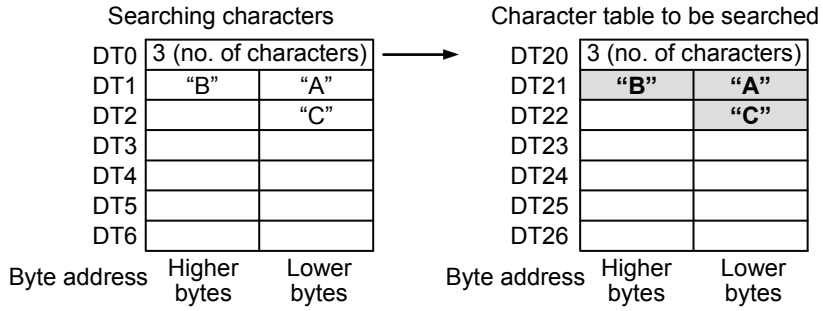
Example 1) Take out the first five characters from the the DT0 string, and transfer them to DT20

[S1]…DT0 [S2]…K5 [D]…DT20



Example 2) No. of characters in [S2] > No. of characters in the [S1] string

[S1]…DT0 [S2]…K7 [D]…DT20



■ Precautions during programming

- Character data in the pre-operation [D] area are overwritten.
- When the no. of characters in [S2] > no. of characters in the string of [S1], the no. of characters to be transferred is limited to that of [S1].

■ Flag operations

Name	Explanation
SR7	To be set in the case of out-of-range in indirect access (index modification).
SR8	To be set when the destination range is out of the available range.
(ER)	To be set when [S2] (no. of characters) is out of the range.

MIDR (Data Read from a Given Position in the String)

■ Ladder diagram

■ Available operation units (●: Available)

- No operation units

■ List of operands

Operand	Explanation
S1	Starting device address of the source data
S2	Start position (data available range: 0 to 4095)
S3	No. of characters to be taken out (data available range: 1 to 4096)
D	Starting device address to store the takeout result

■ Available devices (●: Available)

Operand	16-bit device											32-bit device			Integers			Real numbers		String	Index modifier
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX *1	K	U	H	SF	DF	" "	
S1	●	●	●	●			●	●	●												●
S2	●	●	●	●	●	●	●	●	●	●	●				●	●	●				●
S3	●	●	●	●	●	●	●	●	●	●	●				●	●	●				●
D	●	●	●	●			●	●	●												●

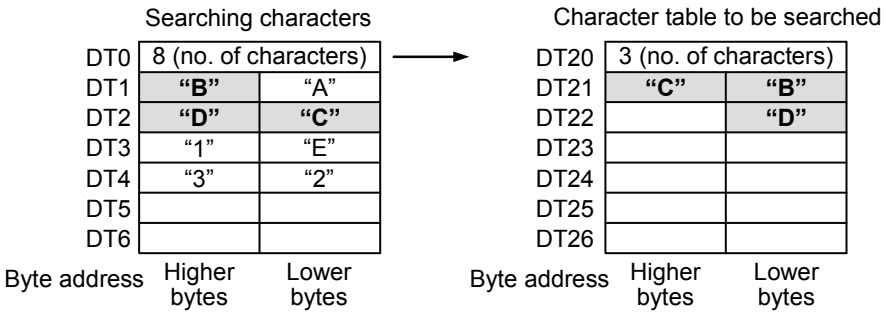
■ Outline of operation

- Take out characters as specified by [S3] from the position specified by [S2] in the string specified by [S1], and store them in the device address specified by [D].

■ Process details

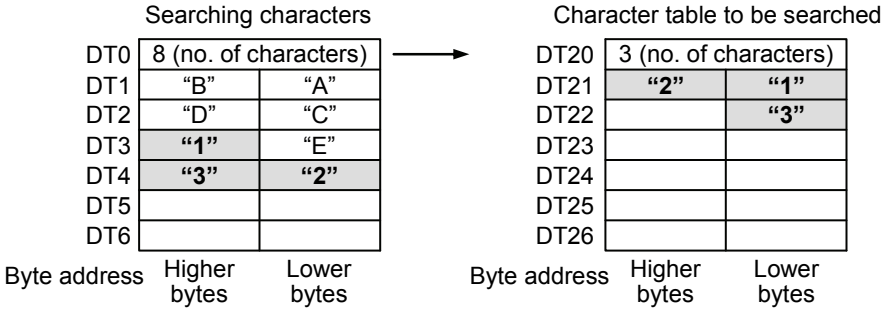
Example 1) Take out three characters from the 1st byte (2nd character) of the DT0 string, and transfer them to DT20

[S1]…DT0 [S2]…K1 [S3]…K3 [D]…DT20



Example 2) No. of characters in [S3] > No. of characters in the [S1] string from the [S2] position

[S1]…DT0 [S2]…K5 [S3]…K5 [D]…DT20



■ Precautions during programming

- Character data in the pre-operation [D] area are overwritten.
- When the no. of characters in [S3] > no. of characters of the string [S1] from the position [S2], the no. of characters to be transferred is limited to that of [S1].
- The [S2] position should be specified counting 0, 1, 2 and so on in ascending order from lower bytes, with the lowest bytes as K0 (Byte 0).

■ Flag operations

Name	Explanation
SR7 SR8 (ER)	To be set in the case of out-of-range in indirect access (index modification).
	To be set when the destination range is out of the available range.
	To be set when [S3] (no. of characters) is out of the range.
	To be set when the no. of characters in [S1] < no. of characters in [S2].

MIDW (Rewrite from a Given Position in the String)

■ Ladder diagram

■ Available operation units (●: Available)

- No operation units

■ List of operands

Operand	Explanation
S1	Starting device address of the source data
S2	No. of characters (data available range: 1 to 4096)
D	Destination starting device address
n	Start position of the destination string (data available range: 0 to 4095)

■ Available devices (●: Available)

Operand	16-bit device											32-bit device			Integers			Real numbers		String	Index modifier
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX *1	K	U	H	SF	DF	" "	
S1	●	●	●	●			●	●	●												●
S2	●	●	●	●	●	●	●	●	●	●	●				●	●	●				●
D	●	●	●	●			●	●	●												●
n	●	●	●	●	●	●	●	●	●	●	●				●	●	●				●

■ Outline of operation

- Take out characters as specified by [S2] from the string specified by [S1], and transfer them to the [n] position in the string specified by [D].

■ Process details

Example 1) Take out three characters from the DT0 string, and transfer them to the 1st byte (2nd character) of the DT20 string

[S1]…DT0 [S2]…K3 [D]…DT20 [n]…K1

String

DT0	8 (no. of characters)	
DT1	**"B"**	**"A"**
DT2	"D"	**"C"**
DT3	"F"	"E"
DT4	"H"	"G"
DT5		
DT6		
Byte address	Higher bytes	Lower bytes

Destination

DT20	8 (no. of characters)	
DT21	"2"	"1"
DT22	"4"	"3"
DT23	"6"	"5"
DT24	"8"	"7"
DT25		
DT26		
Byte address	Higher bytes	Lower bytes

Destination following execution

DT20	8 (no. of characters)	
DT21	**"A"**	"1"
DT22	**"C"**	**"B"**
DT23	"6"	"5"
DT24	"8"	"7"
DT25		
DT26		
Byte address	Higher bytes	Lower bytes

Example 2) No. of characters in [S2] > No. of characters in the [S1] string

[S1]…DT0 [S2]…K5 [D]…DT20 [n]…K1

String

DT0	8 (no. of characters)	
DT1	**"B"**	**"A"**
DT2	**"D"**	**"C"**
DT3		
DT4		
DT5		
DT6		
Byte address	Higher bytes	Lower bytes

Destination

DT20	8 (no. of characters)	
DT21	"2"	"1"
DT22	"4"	"3"
DT23	"6"	"5"
DT24	"8"	"7"
DT25		
DT26		
Byte address	Higher bytes	Lower bytes

Destination following execution

DT20	8 (no. of characters)	
DT21	**"A"**	"1"
DT22	**"C"**	**"B"**
DT23	"6"	**"D"**
DT24	"8"	"7"
DT25		
DT26		
Byte address	Higher bytes	Lower bytes

■ Precautions during programming

- Character data in the pre-operation [D] area are overwritten.
- When the no. of characters in [S2] > no. of characters in the string of [S1], the no. of characters to be transferred is limited to that of [S1].
- The [n] position should be specified counting 0, 1, 2 and so on in ascending order from lower bytes, with the lowest bytes as K0 (Byte 0).

■ Flag operations

Name	Explanation
SR7 SR8 (ER)	To be set in the case of out-of-range in indirect access (index modification).
	To be set when [S2] (no. of characters) is out of the range.
	To be set when the no. of characters in [D] < [n].

SREP (Replacement of a String)

■ Ladder diagram

■ Available operation units (●: Available)

- No operation units

■ List of operands

Operand	Explanation
S	Starting device address of the source string
D	Starting device address of the destination string
p	Replacement start position of the destination string (data available range: 0 to 4095)
n	No. of characters to be replaced (data available range: 1 to 4096)

■ Available devices (●: Available)

Operand	16-bit device											32-bit device			Integers			Real numbers		String	Index modifier
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX *1	K	U	H	SF	DF	" "	
S	●	●	●	●			●	●	●												●
D	●	●	●	●			●	●	●												●
p	●	●	●	●	●	●	●	●	●	●	●				●	●	●				●
n	●	●	●	●	●	●	●	●	●	●	●				●	●	●				●

■ Outline of operation

- The string specified by [S] is replaced with characters as specified by [n] from the position [p] in the string specified by [D].

■ Process details

Example 1) Replace the DT0 string with three characters from the 1st byte (2nd character) of DT20

[S1]…DT0 [D]…DT20 [p]…K1 [n]…K3

String

DT0	5 (no. of characters)	
DT1	**“B”**	**“A”**
DT2	**“D”**	**“C”**
DT3		**“E”**
DT4		
DT5		
DT6		
Byte address	Higher bytes	Lower bytes

Destination

DT20	8 (no. of characters)	
DT21	**“2”**	“1”
DT22	**“4”**	**“3”**
DT23	“6”	“5”
DT24	“8”	“7”
DT25		
DT26		
Byte address	Higher bytes	Lower bytes

Destination following execution

DT20	**10 (no. of characters)**	
DT21	**“A”**	“1”
DT22	**“C”**	**“B”**
DT23	**“E”**	**“D”**
DT24	“6”	“5”
DT25	“8”	“7”
DT26		
Byte address	Higher bytes	Lower bytes

* In this process, the data from the original data in the 1st byte to the 4th byte are deleted from DT20. Consequently, the no. of characters increases from 8 to 10.

Example 2) The no. of characters of [n] is larger than the no. of characters of the [S1] string from the position specified by [p].

[S1]…DT0 [D]…DT20 [p]…K4 [n]…K8

String

DT0	5 (no. of characters)	
DT1	**“B”**	**“A”**
DT2	**“D”**	**“C”**
DT3		**“E”**
DT4		
DT5		
DT6		
Byte address	Higher bytes	Lower bytes

Destination

DT20	10 (no. of characters)	
DT21	“2”	“1”
DT22	“4”	“3”
DT23	**“6”**	**“5”**
DT24	**“8”**	**“7”**
DT25	**“0”**	**“9”**
DT26		
Byte address	Higher bytes	Lower bytes

Destination following execution

DT20	9 (no. of characters)	
DT21	“2”	“1”
DT22	“4”	“3”
DT23	**“B”**	**“A”**
DT24	**“D”**	**“C”**
DT25		**“E”**
DT26		
Byte address	Higher bytes	Lower bytes

■ Precautions during programming

- Character data in the pre-operation [D] area are not cleared. (They are overwritten.)
- If the no. of characters [n] is larger than the no. of characters in the string of [S1] following the position specified by [p], then characters to be replaced are limited to the no. of characters following the position specified by [p] in the string of [S1].
- The [p] position should be specified counting 0, 1, 2 and so on in ascending order from lower bytes, with the lowest bytes as K0 (Byte 0).

■ Flag operations

Name	Explanation
SR7 SR8 (ER)	To be set in the case of out-of-range in indirect access (index modification).
	To be set when the no. of characters > string size.
	To be set when the no. of characters in [D] < [n].
	To be set when [S2] (no. of characters) is out of the range.

SRC (Data Search)

■ Ladder diagram

■ Available operation units (●: Available)

Operation unit	bit	US	SS	UL	SL	SF	DF
i		●	●	●	●	●	●

■ List of operands

Operand	Explanation
S1	Device address where the data to be searched are stored, or constant (data format: according to the operation unit)
S2	Start position of the search range (data format: according to the operation unit)
S3	End position of the search range (data format: according to the operation unit)
D	Device address to store the search result (data format: unsigned 32-bit integer)

■ Available devices (●: Available)

Operand	16-bit device											32-bit device *1			Integers			Real numbers		String	Index modifier *2
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX *3	K *4	U *5	H *6	SF *7	DF *8	" "	
S1	●	●	●	●			●	●	●	●	●	●	●	●	●	●	●	●	●		●
S2	●	●	●	●			●	●	●	●	●	●	●	●							●
S3	●	●	●	●			●	●	●	●	●	●	●	●							●
D	●	●	●	●			●	●	●		●	●	●	●							●

*1: Operation unit: 16-bit integers (SS, US) cannot be specified.
*2: Only 16-bit devices, 32-bit devices, and integer constants can be modified (real number constants, and character constants cannot be specified).
*3: Index register (I0 to IE)
*4: Only operation unit: signed integers (SS, SL) can be specified.
*5: Only operation unit: unsigned integers (US, UL) can be specified.
*6: Only operation unit: integers (US, SS, UL, SL) can be specified.
*7: Only operation unit: single-precision, floating-point real numbers (SF) can be specified.
*8: Only operation unit: double-precision, floating-point real numbers (DF) can be specified.

■ Outline of operation

- Search the range specified by [S2] and [S3] for the search data specified by [S1].
- The search result is given in the data format of unsigned 32-bit integer, and stored as follows.

16-bit device	32-bit device	Description of output
[D], [D]+1	[D]	Store the number of data with the same value in a decimal form
[D]+2、[D]+3	[D]+1	Store the position of the first matching data (relative position with the first data as '0')

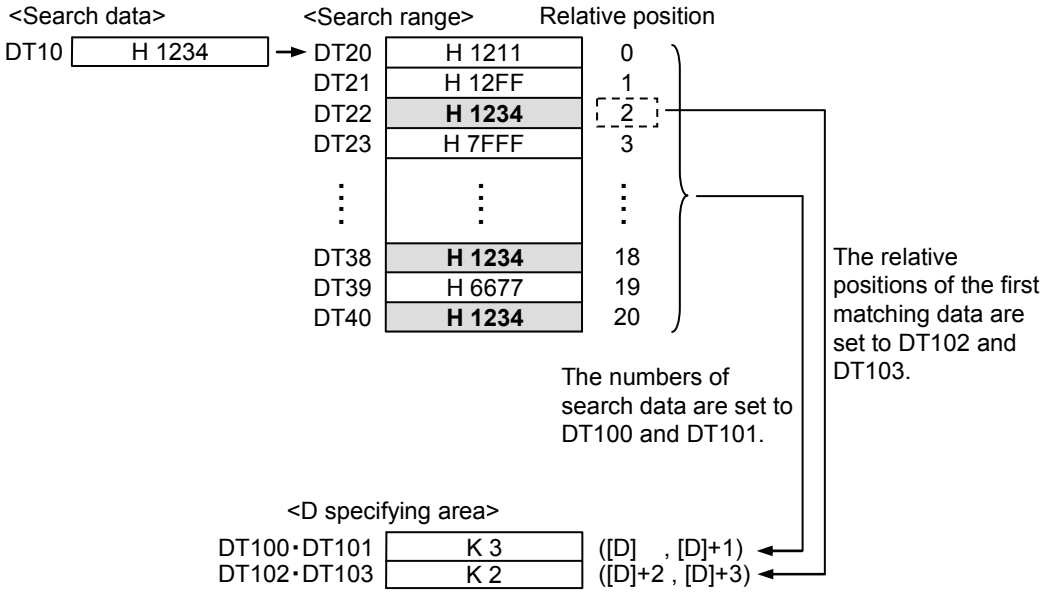
- The max. no. of data that can be specified is 30000.
- Search is executed from [S2] to [S3].

■ Process details

Example 1) Operation unit: 16 bits (US, SS)

[i]…US,SS

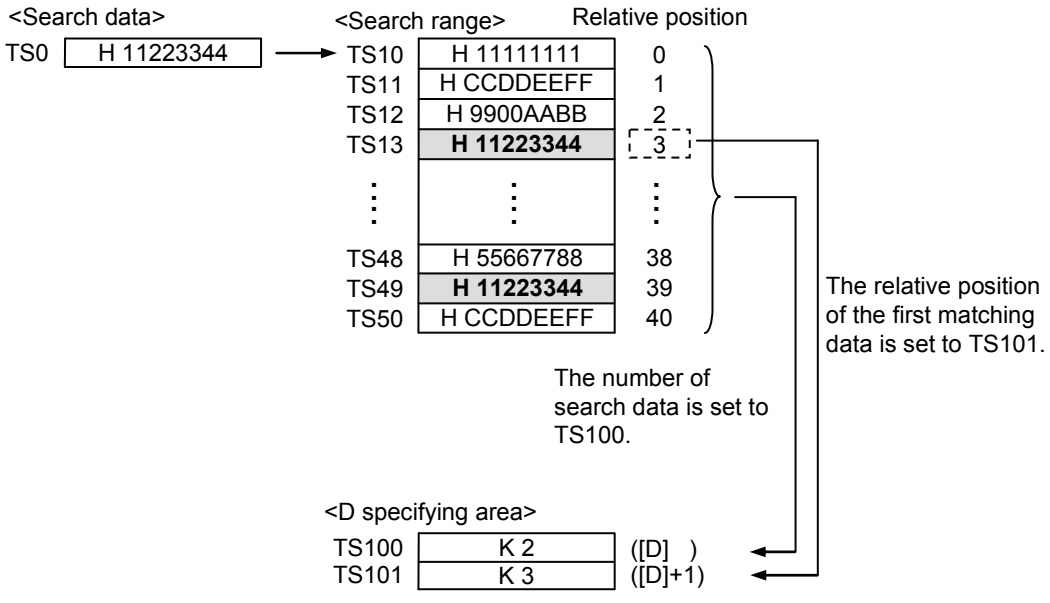
[S1]…DT10 [S2]…DT20 [S3]…DT40 [D]…DT100



Example 2) Operation unit: 32 bits (UL, SL, SF)

[i]…UL,SL,SF

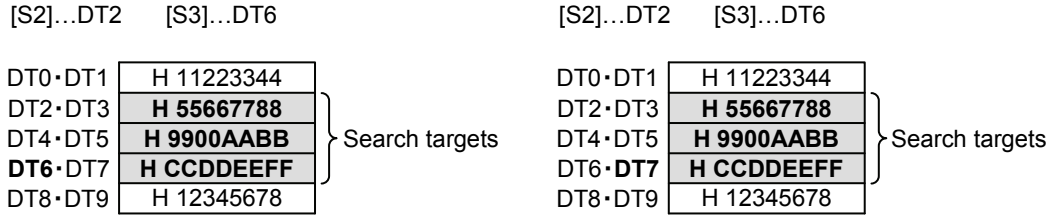
[S1]…TS0 [S2]…TS10 [S3]…TS50 [D]…TS100



■ Precautions during programming

- The end position of the search range includes down to the device containing [S3].

Example) When the operation unit is specified as 32 bits, the search range becomes the same whether a higher or lower address is specified for the [S3] device address.

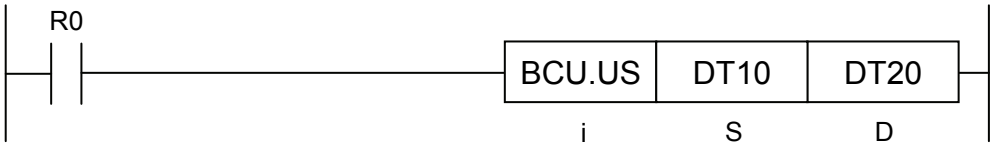


■ Flag operations

Name	Explanation
SR7 SR8 (ER)	To be set in the case of out-of-range in indirect access (index modification, pointer access).
	To be set when [S2] > [S3].
	To be set when the [S2] area and the [S3] area differ.

BCU (ON Bits Count)

■ Ladder diagram



■ Available operation units (●: Available)

Operation unit	bit	US	SS	UL	SL	SF	DF
i		●		●			

■ List of operands

Operand	Explanation
S	Target device address or constant (data format: according to the operation unit)
D	Device address to store the result (data format: unsigned 16-bit integer)

■ Available devices (●: Available)

Operand	16-bit device											32-bit device			Integers			Real numbers		String	Index modifier
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX *1	K	U	H	SF	DF	" "	
S	●	●	●	●	●	●	●	●	●	●	●	●	●								●
D	●	●	●	●			●	●	●		●										●

*1: Operation unit: 16-bit integers (SS, US) cannot be specified.
*2: Only 16-bit devices, 32-bit devices, and integer constants can be modified (real number constants, and character constants cannot be specified).

■ Outline of operation

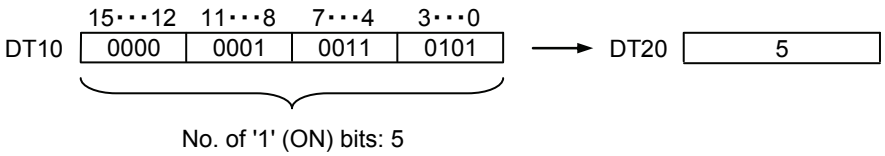
- Count the number of bits in the ON state (with the value '1') in the data specified by [S]. The count result is stored in the device address specified by [D].
- The result is stored as an unsigned 16-bit integer.

■ Process details

Example 1) Operation unit: 16 bits (US)

[i]…US

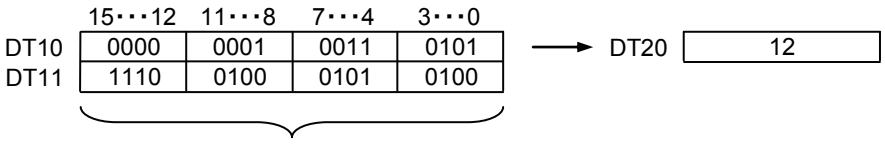
[S1]…DT10 [D]…DT20



Example 2) Operation unit: 32 bits (UL) (specify a 16-bit device for the device)

[i]…UL

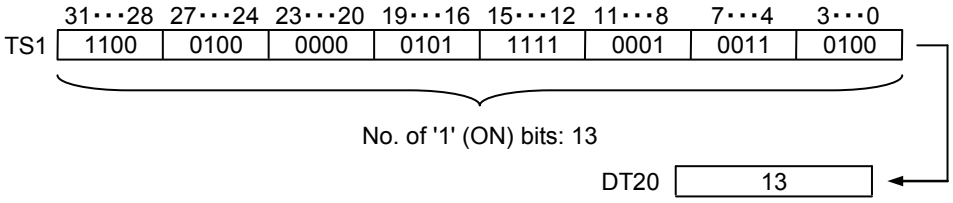
[S1]…DT10 [D]…DT20



Example 3) Operation unit: 32 bits (UL) (specify a 32-bit device for the device)

[i]…UL

[S1]…TS1 [D]…DT20

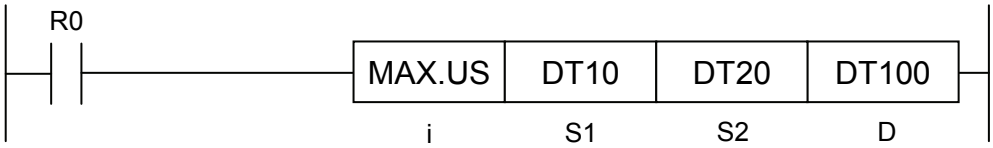


■ Flag operations

Name	Explanation
SR7 SR8 (ER)	To be set in the case of out-of-range in indirect access (index modification, pointer access).

MAX (Obtainment of the Max. Value)

■ Ladder diagram



■ Available operation units (●: Available)

Operation unit	bit	US	SS	UL	SL	SF	DF
i		●	●	●	●	●	●

■ List of operands

Operand	Explanation
S1	Start position of the search range for the max. value (data format: according to the operation unit)
S2	End position of the search range for the max. value (data format: according to the operation unit)
D	Device address to store the result of the search for the max. value (data format: by the operation unit)

■ Available devices (●: Available)

Operand	16-bit device											32-bit device *1			Integers			Real numbers		String	Index modifier *2
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX *3	K	U	H	SF	DF	" "	
S1	●	●	●	●			●	●	●	●	●	●	●	●							●
S2	●	●	●	●			●	●	●	●	●	●	●	●							●
D	●	●	●	●			●	●	●		●	●	●	●							●

*1: Operation unit: 16-bit integers (SS, US) cannot be specified.
*2: Only 16-bit devices, 32-bit devices, and integer constants can be modified (real number constants, and character constants cannot be specified).
*3: Index register (I0 to IE)

■ Outline of operation

- The range of the device area specified by [S1] and [S2] is searched for the max. value.
- The resulting value is stored in the device area specified by [D], and the relative address value from [S1] is stored in [n].
- The relative address storage position ([D]+[n]) varies by operation unit.
- The max. no. of data that can be specified is 30000.
- [D] is in the following format by operation unit.

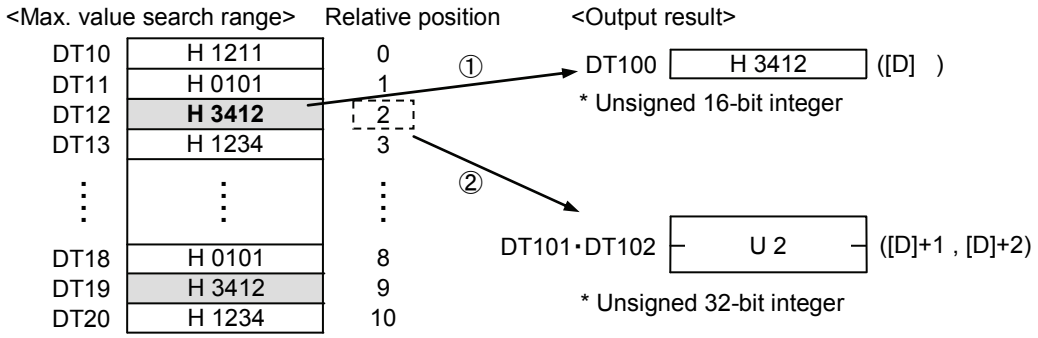
Operation unit	16 bits (US, SS)		32 bits (UL, SL, SF)		64 bits (DF)		Description of output
Device size	16 bits	32 bits	16 bits	32 bits	16 bits	32 bits	
Result storage area	[D]	Not available	[D] to [D+1]	[D]	[D] to [D+3]	[D] to [D+1]	Store the max. value
	[D+1] to [D+2]	Not available	[D+2] to [D+3]	[D+1]	[D+4] to [D+5]	[D+2]	Position of the first detected max. value; Relative position with the start of [S1] as 0.

■ Process details

Example 1) Operation unit: 16 bits (US)

[i]…US

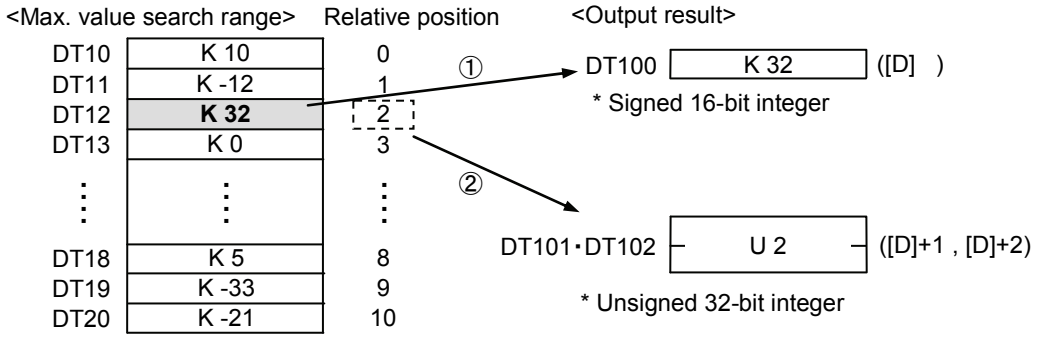
[S1]…DT10 [S2]…DT20 [D]…DT100



Example 2) Operation unit: 16 bits (SS)

[i]…SS

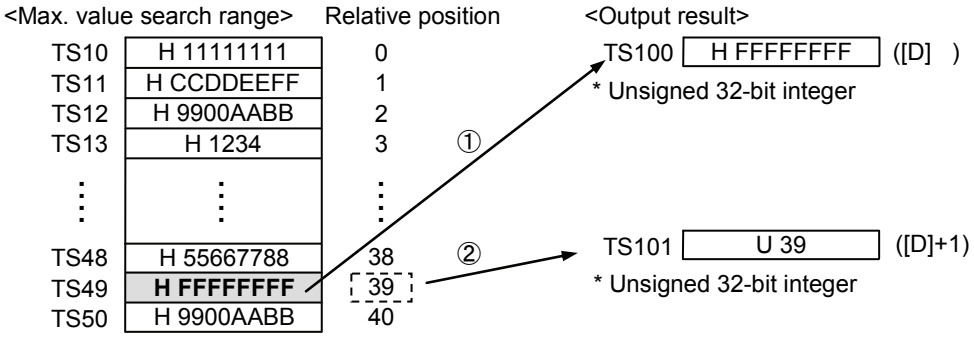
[S1]…DT10 [S2]…DT20 [D]…DT100



Example 3) Operation unit: 32 bits (UL) (specify a 32-bit device)

[i]…UL

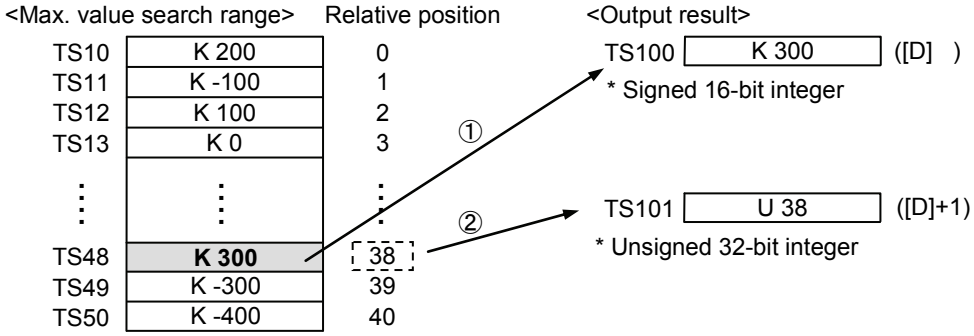
[S1]…TS10 [S2]…TS50 [D]…TS100



Example 4) Operation unit: 32 bits (SL) (specify a 32-bit device)

[i]…SL

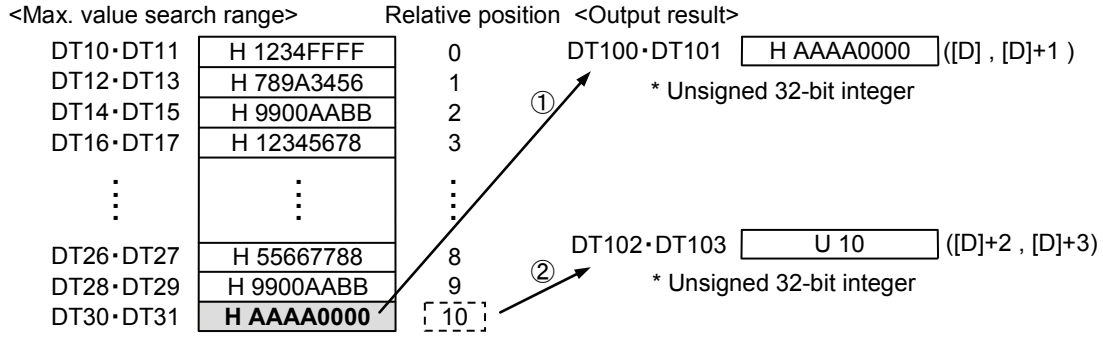
[S1]…TS10 [S2]…TS50 [D]…TS100



Example 5) Operation unit: 32 bits (UL) (specify a 16-bit device)

[i]…UL

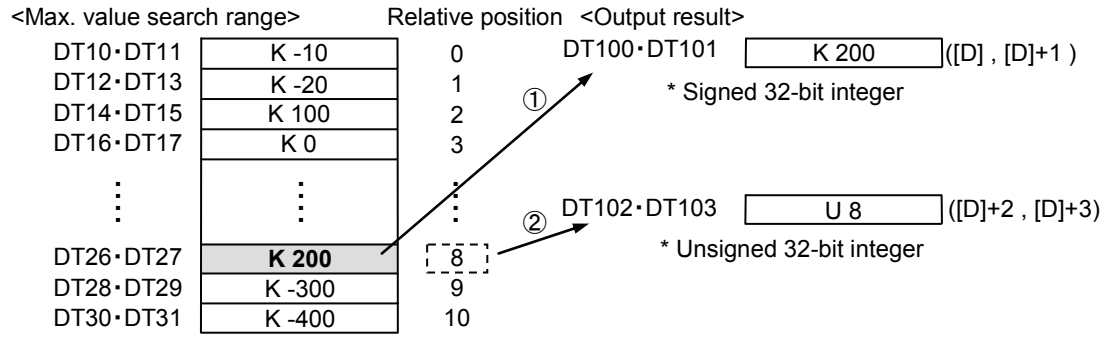
[S1]…DT10 [S2]…DT30 [D]…DT100



Example 6) Operation unit: 32 bits (SL) (specify a 16-bit device)

[i]…SL

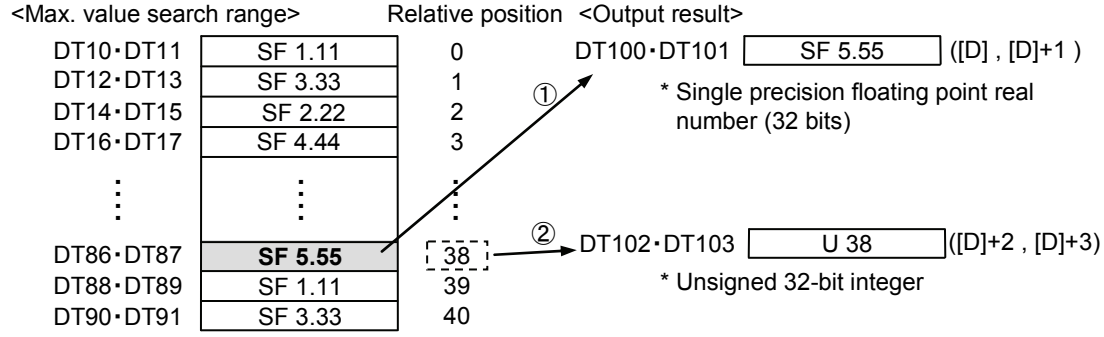
[S1]…DT10 [S2]…DT30 [D]…DT100



Example 7) Operation unit: Single-precision, floating-point real number (SF)

[i]…SF

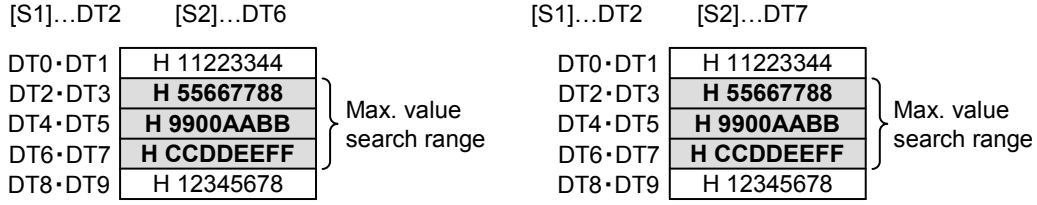
[S1]…DT10 [S2]…DT90 [D]…DT100



■ Precautions during programming

- The end position of the search range for the max. value includes down to the device containing [S2].

Example) When the operation unit is specified as 32 bits, the max. value search range becomes the same whether a higher or lower address is specified for the [S2] device address.



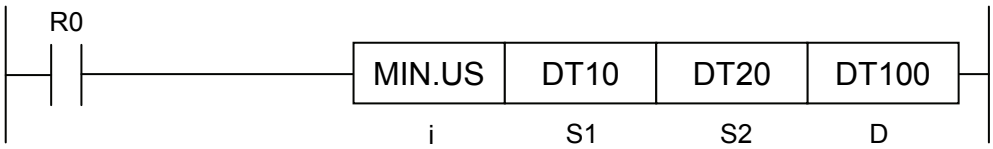
- Data are overwritten if [D] (max. value search result) is specified within the search range for the max. value.

■ Flag operations

Name	Explanation
SR7	To be set in the case of out-of-range in indirect access (index modification, pointer access).
SR8	To be set when [S1] > [S2].
(ER)	To be set when the [S1] device and the [S2] device differ.

MIN (Obtainment of the Min. Value)

■ Ladder diagram



■ Available operation units (●: Available)

Operation unit	bit	US	SS	UL	SL	SF	DF
i		●	●	●	●	●	●

■ List of operands

Operand	Explanation
S1	Start position of the search range for the min. value (data format: according to the operation unit)
S2	End position of the search range for the min. value (data format: according to the operation unit)
D	Device address to store the result of the search for the min. value (data format: by the operation unit)

■ Available devices (●: Available)

Operand	16-bit device											32-bit device *1			Integers			Real numbers		String	Index modifier *2
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX *3	K	U	H	SF	DF	" "	
S1	●	●	●	●			●	●	●	●	●	●	●	●							●
S2	●	●	●	●			●	●	●	●	●	●	●	●							●
D	●	●	●	●			●	●	●		●	●	●	●							●

*1: Operation unit: 16-bit integers (SS, US) cannot be specified.
*2: Only 16-bit devices, 32-bit devices, and integer constants can be modified (real number constants, and character constants cannot be specified).
*3: Index register (I0 to IE)

■ Outline of operation

- The range of the device area specified by [S1] and [S2] is searched for the min. value. The resulting value is stored in the device area specified by [D], and the relative address value from [S1] is stored in [D]+[n].
- The relative address storage position ([D]+[n]) varies by operation unit.
- The max. no. of data that can be specified is 30,000.
- [D] is in the following format by operation unit.

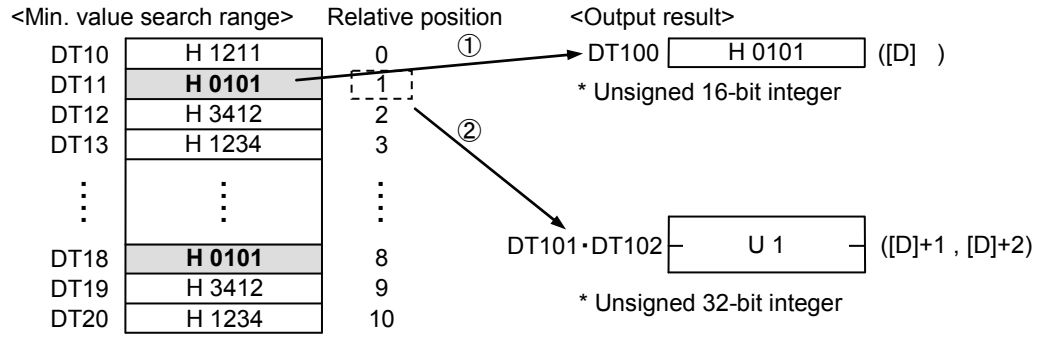
Operation unit	16 bits (US, SS)		32 bits (UL, SL, SF)		64 bits (DF)		Description of output
Device size	16 bits	32 bits	16 bits	32 bits	16 bits	32 bits	
Result storage area	[D]	Not available	[D] to [D+1]	[D]	[D] to [D+3]	[D] to [D+1]	Store the min. value
	[D+1] to [D+2]	Not available	[D+2] to [D+3]	[D+1]	[D+4] to [D+5]	[D+2]	Position of the first detected max. value; Relative position with the start of [S1] as 0.

■ Process details

Example 1) Operation unit: 16 bits (US)

[i]…US

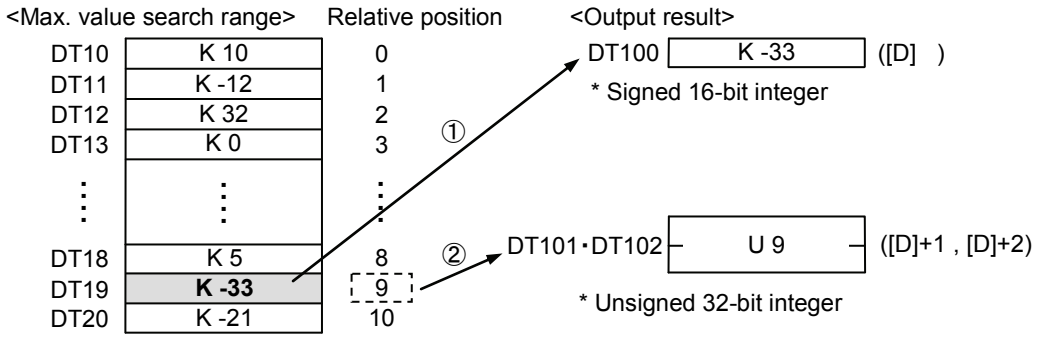
[S1]…DT10 [S2]…DT20 [D]…DT100



Example 2) Operation unit: 16 bits (SS)

[i]…SS

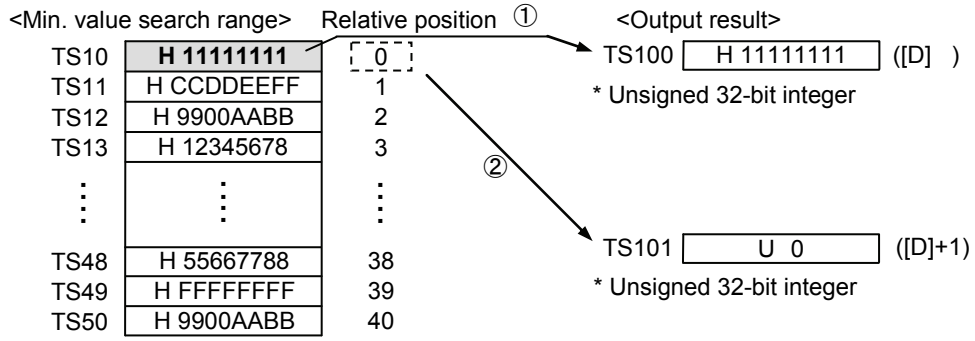
[S1]…DT10 [S2]…DT20 [D]…DT100



Example 3) Operation unit: 32 bits (UL) (specify a 32-bit device)

[i]…UL

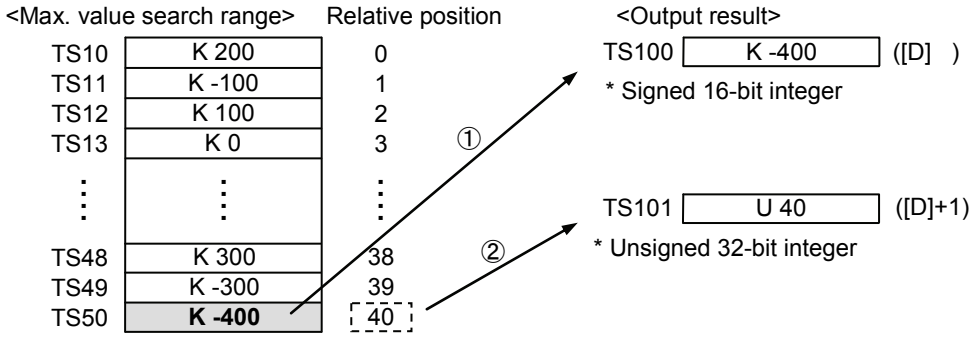
[S1]…TS10 [S2]…TS50 [D]…TS100



Example 4) Operation unit: 32 bits (SL) (specify a 32-bit device)

[i]…SL

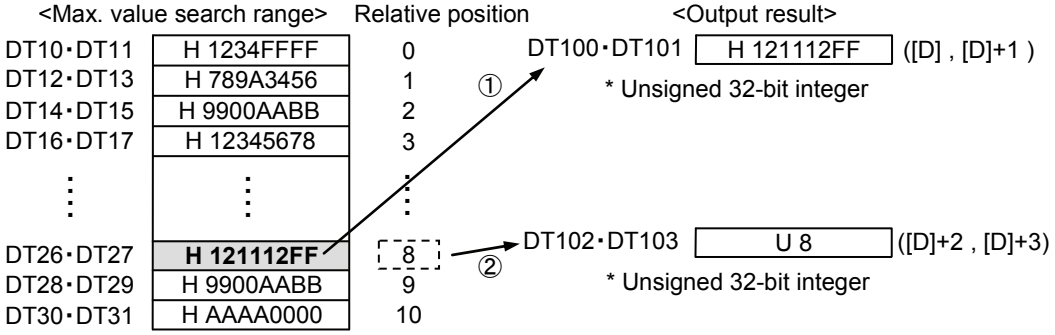
[S1]…TS10 [S2]…TS50 [D]…TS100



Example 5) Operation unit: 32 bits (UL) (specify a 16-bit device)

[i]…UL

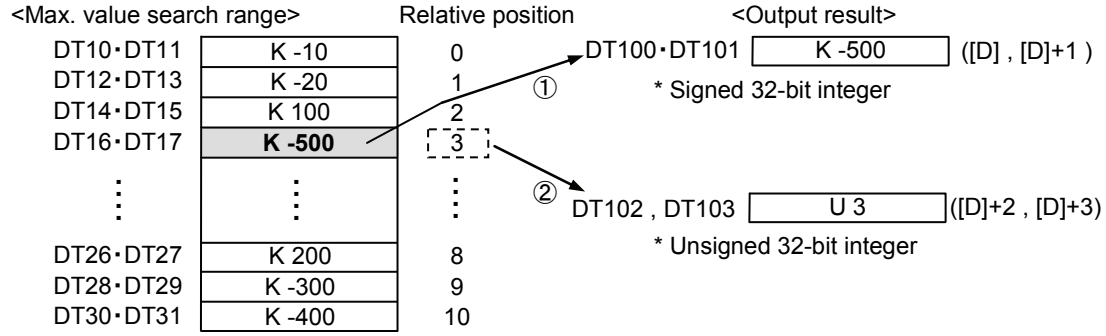
[S1]…DT10 [S2]…DT30 [D]…DT100



Example 6) Operation unit: 32 bits (SL) (specify a 16-bit device)

[i]…SL

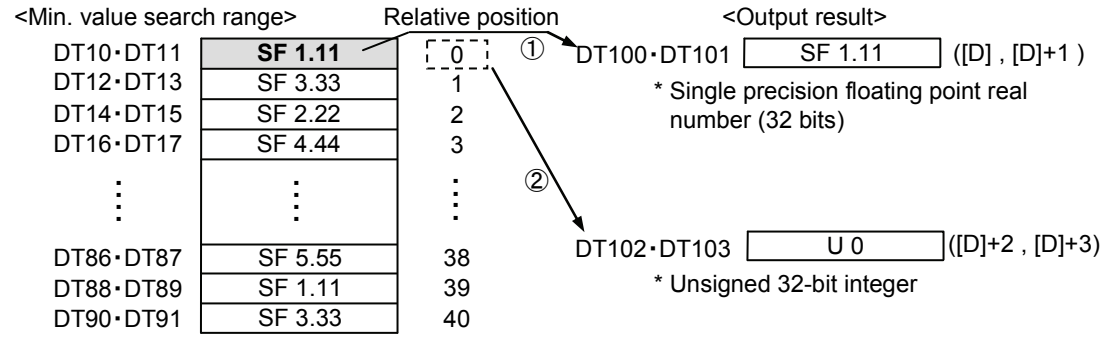
[S1]…DT10 [S2]…DT30 [D]…DT100



Example 7) Operation unit: Single-precision, floating-point real number (SF)

[i]…SF

[S1]…DT10 [S2]…DT90 [D]…DT100



■ Precautions during programming

- The end position of the search range for the min. value includes down to the device containing [S2].

Example) When the operation unit is specified as 32 bits, the min. value search range becomes the same whether a higher or lower address is specified for the [S2] device address.

[S1]…DT2 [S2]…DT6

DT0・DT1	H 11223344
DT2・DT3	**H 55667788**
DT4・DT5	**H 9900AABB**
DT6・DT7	**H CCDDEEFF**
DT8・DT9	H 12345678

Min. value search range

[S1]…DT2 [S2]…DT7

DT0・DT1	H 11223344
DT2・DT3	**H 55667788**
DT4・DT5	**H 9900AABB**
DT6・DT7	**H CCDDEEFF**
DT8・DT9	H 12345678

Min. value search range

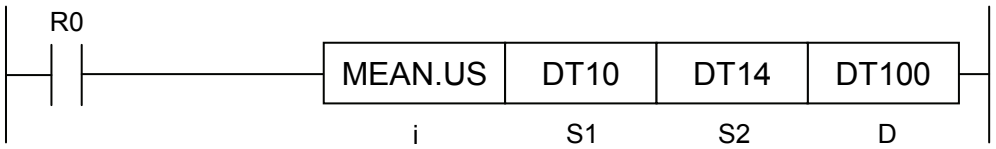
- Data are overwritten if [D] (min. value search result) is specified within the search range for the max. value.

■ Flag operations

Name	Explanation
SR7 SR8 (ER)	To be set in the case of out-of-range in indirect access (index modification, pointer access).
	To be set when [S1] > [S2].
	To be set when the [S1] device and the [S2] device differ.

MEAN (Obtainment of the Total and the Mean Value)

■ Ladder diagram



■ Available operation units (●: Available)

Operation unit	bit	US	SS	UL	SL	SF	DF
i		●	●	●	●	●	●

■ List of operands

Operand	Explanation
S1	Start position of the target area (data format: according to the operation unit)
S2	End position of the target area (data format: according to the operation unit)
D	Device address to store the result (data format: by the operation unit)

■ Available devices (●: Available)

Operand	16-bit device											32-bit device *1			Integers			Real numbers		String	Index modifier *2
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX *3	K	U	H	SF	DF	" "	
S1	●	●	●	●			●	●	●	●	●	●	●	●							●
S2	●	●	●	●			●	●	●	●	●	●	●	●							●
D	●	●	●	●			●	●	●		●	●	●	●							●

*1: Operation unit: 16-bit integers (SS, US) cannot be specified.
*2: Only 16-bit devices, 32-bit devices, and integer constants can be modified (real number constants, and character constants cannot be specified).
*3: Index register (I0 to IE)

■ Outline of operation

- The total and the mean in the range of the device area specified by [S1] and [S2] are stored in the device area specified by [D].
- The storage position of the total and the mean, with [D] as the starting address, varies by operation unit.
- The total value takes up twice the area size of the operation unit, and the mean value takes up the same area size as the operation unit. Note that, in the case of floating point real number (SF), the total value also takes up the same area size as the operation unit.
- The max. no. of data that can be specified is 30,000.
- When the operation unit is US, SS, UL or SL, the mean value is given in an integer rounding the first decimal point down.
- [D] is in the following format by operation unit.

Operation unit	16 bits (US, SS)		32 bits (UL, SL)		32 bits (SF)		64 bits (DF)		Description of output
Device size	16 bits	32 bits	16 bits	32 bits	16 bits	32 bits	16 bits	32 bits	
Result storage Area	[D] to [D+1]	Not available	[D] to [D+3]	[D] to [D+1]	[D] to [D+1]	[D]	[D] to [D+3]	[D] to [D+1]	Total value
	[D+2]	Not available	[D+4] to [D+5]	[D+2]	[D+2] to [D+3]	[D+1]	[D+4] to [D+7]	[D+2] to [D+3]	Mean Value

■ Process details

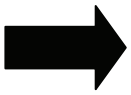
Example 1) Operation unit: 16 bits (US)

Total value is given in 32-bit data, and the mean value is given in 16-bit data.

[i]…US
[S1]…DT10 [S2]…DT14 [D]…DT100

<Total/mean calculation range>

	Value
DT10	H 1111
DT11	H 5555
DT12	H 7777
DT13	H AAAA
DT14	H FFFF



<Output result>

	Value	
DT100・DT101	**H 00028886**	Total value ([D] and [D]+1)

* Unsigned 32-bit integer

	Value	
DT102	**H 81B4**	Mean value ([D]+2)

* Unsigned 16-bit integer

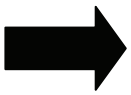
Example 2) Operation unit: 16 bits (SS)

Total value is given in 32-bit data, and the mean value is given in 16-bit data.

[i]…SS
[S1]…DT10 [S2]…DT14 [D]…DT100

<Total/mean calculation range>

	Value
DT10	K 11
DT11	K -33
DT12	K 44
DT13	K 55
DT14	K -22



<Output result>

	Value	
DT100・DT101	**K -55**	Total value ([D] and [D]+1)

* Signed 32-bit integer

	Value	
DT102	**K 11**	Mean value ([D]+2)

* Signed 16-bit integer

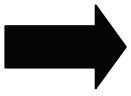
Example 3) Operation unit: 32 bits (UL) (specify a 32-bit device)

Total value is given in 64-bit data, and the mean value is given in 32-bit data.

[i]…UL
[S1]…TS10 [S2]…TS14 [D]…TS100

<Total/mean calculation range>

	Value
TS10	H 00000000
TS11	H 22222222
TS12	H 33333333
TS13	H 44444444
TS14	H 55555555



<Output result>

	Value	
TS100・TS101	**H EEEEEEEE**	Total value ([D] and [D]+1)

* Unsigned 64-bit integer

	Value	
TS102	**H 2FC962FC**	Mean value ([D]+2)

* Unsigned 32-bit integer

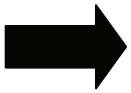
Example 4) Operation unit: 32 bits (SL) (specify a 32-bit device)

Total value is given in 64-bit data, and the mean value is given in 32-bit data.

[i]…SL
[S1]…TS10 [S2]…TS14 [D]…TS100

<Total/mean calculation range>

	Value
TS10	K 1000
TS11	K 2000
TS12	K -3000
TS13	K -4000
TS14	K -5000



<Output result>

	Value	
TS100・TS101	**K -9000**	Total value ([D] and [D]+1)

* Signed 64-bit integer

	Value	
TS102	**K -1800**	Mean value ([D]+2)

* Signed 32-bit integer

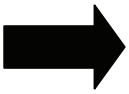
Example 5) Operation unit: 32 bits (UL) (specify a 16-bit device)

Total value is given in 64-bit data, and the mean value is given in 32-bit data.

[i]…UL
[S1]…DT10 [S2]…DT14 [D]…DT100

<Total/mean calculation range>

	Value
DT10・DT11	H 11110000
DT12・DT13	H 33332222
DT14・DT15	H 55554444
DT16・DT17	H 77776666



<Output result>

	Value	
DT100-DT103	**H 11110CCCC**	Total value ([D] to [D]+3)

* Unsigned 64-bit integer

	Value	
DT104・DT105	**H 44443333**	Mean value ([D]+4, [D]+5)

* Unsigned 32-bit integer

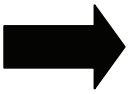
Example 6) Operation unit: 32 bits (SL) (specify a 16-bit device)

Total value is given in 64-bit data, and the mean value is given in 32-bit data.

[i]…SL
[S1]…DT10 [S2]…DT16 [D]…DT100

<Total/mean calculation range>

	Value
DT10・DT11	K -100
DT12・DT13	K 600
DT14・DT15	K 500
DT16・DT17	K -200



<Output result>

	Value	
DT100・DT103	**K 800**	Total value ([D] to [D]+3)

* Signed 64-bit integer

	Value	
DT104・DT105	**K 200**	Mean value ([D]+4, [D]+5)

* Signed 32-bit integer

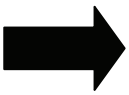
Example 7) Operation unit: Single-precision, floating-point real number (SF) (specify a 16-bit device)

Total value is given in 32-bit data, and the mean value is given in 32-bit data.

[i]…SF
[S1]…DT10 [S2]…DT24 [D]…DT100

<Total/mean calculation range>

	Value
DT10・DT11	SF 3.33E+00
DT12・DT13	SF 1.11E+00
DT14・DT15	SF 4.44E+00
DT16・DT17	SF 5.55E+00
DT18・DT19	SF 2.22E+00
DT20・DT21	SF 1.11E+00
DT22・DT23	SF 3.33E+00
DT24・DT25	SF 5.55E+00



<Output result>

	Value	
DT100・DT101	**SF 2.66E+01**	Total value ([D] and [D]+1)

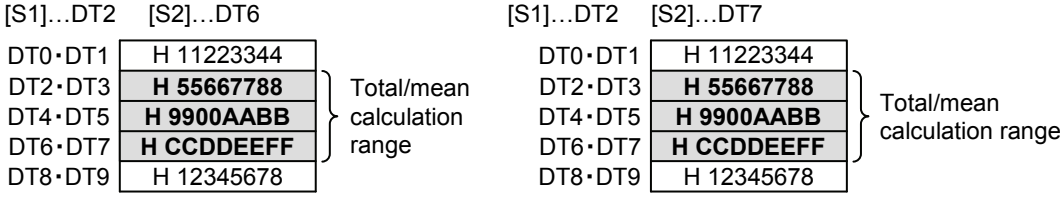
* Single precision floating point real number (32 bits)

	Value	
DT102・DT103	**SF 3.33E+00**	Mean value ([D]+2, [D]+3)

* Single precision floating point real number (32 bits)

■ Precautions during programming

- The end position of the calculation range for the total and the mean includes down to the device containing [S2].
  Example) When the operation unit is 32 bits, the calculation range is the same whether a device number of higher level or lower level is specified.



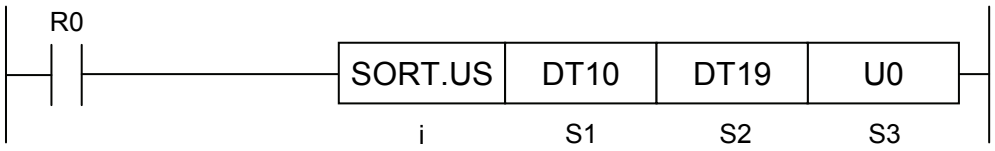
- Data are overwritten if [D] (total and mean calculation results) is specified within the calculation range for the total and the mean.

■ Flag operations

Name	Explanation
SR7 SR8 (ER)	To be set in the case of out-of-range in indirect access (index modification, pointer access).
	To be set when [S1] > [S2].
	To be set when the [S1] device and the [S2] device differ.

SORT (Sort)

■ Ladder diagram



■ Available operation units (●: Available)

Operation unit	bit	US	SS	UL	SL	SF	DF
i		●	●	●	●	●	●

■ List of operands

Operand	Explanation
S1	Start position of the target area (data format: according to the operation unit)
S2	End position of the target area (data format: according to the operation unit)
S3	Sort conditions (data format: unsigned 16-bit integer)

■ Available devices (●: Available)

Operand	16-bit device											32-bit device			Integers			Real numbers		String	Index modifier *1
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX *2	K	U	H	SF	DF	" "	
S1							●	●													●
S2							●	●													●
S3	●	●	●	●			●	●								●	●				●

*1: Only 16-bit devices, 32-bit devices, and integer constants can be modified (real number constants, and character constants cannot be specified).
*2: Index register (I0 to IE)

■ Outline of operation

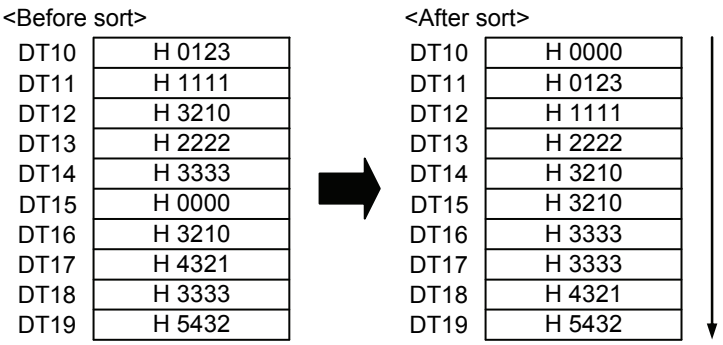
- Data in the range from the area specified by [S1] to the area specified by [S2] are sorted in ascending order or in descending order, according to the sort conditions specified by [S3].
- Available sort conditions for [S3] are as follows.
  U0: Ascending order
  U1: Descending order

■ Process details

Example 1) Operation unit: 16 bits (US)

[i]…US

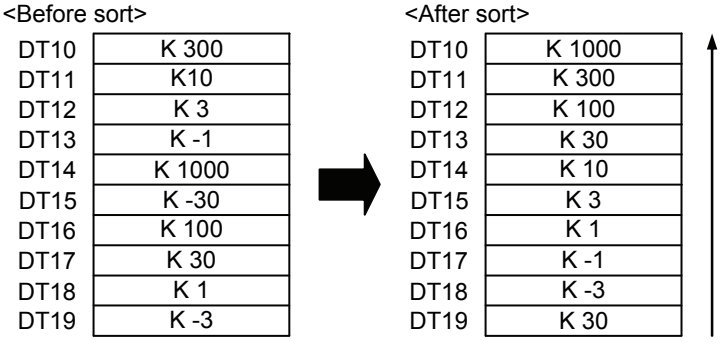
[S1]…DT10 [S2]…DT19 [S3]…U0 (Ascending order)



Example 2) Operation unit: 16 bits (SS)

[i]…SS

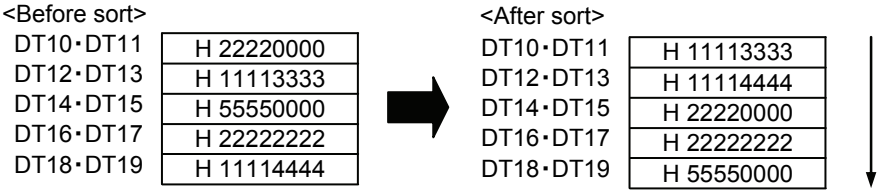
[S1]…DT10 [S2]…DT19 [S3]…U1 (Descending order)



Example 3) Operation unit: 32 bits (UL)

[i]…UL

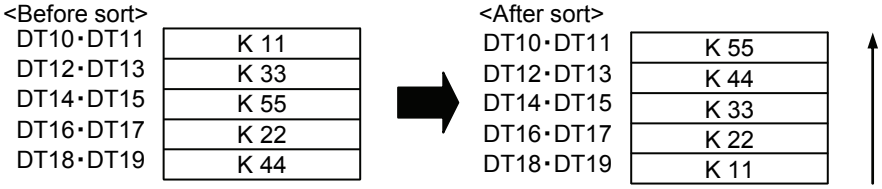
[S1]…DT10 [S2]…DT19 [S3]…U0 (Ascending order)



Example 4) Operation unit: 32 bits (SL)

[i]…SL

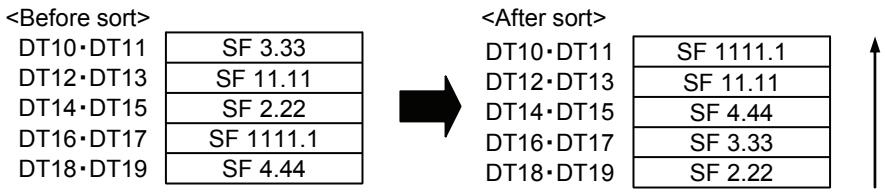
[S1]…DT10 [S2]…DT19 [S3]…U1 (Descending order)



Example 5) Operation unit: Single-precision, floating-point real number (SF)

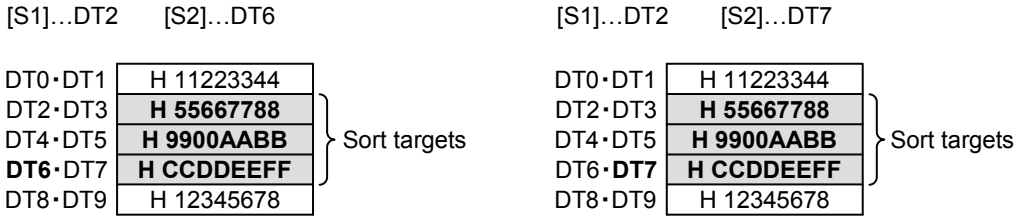
[i]…SF

[S1]…DT10 [S2]…DT19 [S3]…U1 (Descending order)



■ Precautions during programming

- It must be noted that, since the time for data comparison increases in proportion to the square of the number of data, the sorting process can take long time when there are a large number of data to be sorted.
- During sort execution, data in [S1] to [S2] are sorted in sequence according to the sort conditions.
- The end position of the sort range includes down to the device containing [S2].
  Example) When the operation unit is 32 bits, the sort range is the same whether a device number of higher level or lower level is specified.

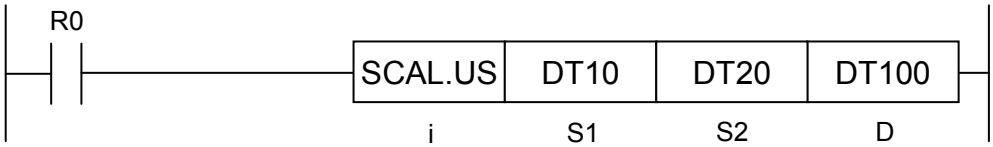


■ Flag operations

Name	Explanation
SR7	To be set in the case of out-of-range in indirect access (index modification, pointer access).
SR8	To be set when [S1] > [S2].
(ER)	To be set when the [S1] device and the [S2] device differ.

SCAL (Linearization)

■ Ladder diagram



■ Available operation units (●: Available)

Operation unit	bit	US	SS	UL	SL	SF	DF
i		●	●	●	●	●	●

■ List of operands

Operand	Explanation
S1	Data equivalent to the input value X, or the area storing it (data format: according to the operation unit)
S2	Starting address of the data table used for scaling (linearization) (data format: by the operation unit)
D	Area to store the output result Y (data format: according to the operation unit)

■ Available devices (●: Available)

Operand	16-bit device											32-bit device *1			Integers			Real numbers		String	Index modifier *2
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX *3	K *4	U *5	H *6	SF *7	DF *8	" "	
S1	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●		●
S2	●	●	●	●	●	●	●	●	●	●	●										●
D	●	●	●	●			●	●	●		●	●	●	●							●

*1: Operation unit: 16-bit integers (SS, US) cannot be specified.
*2: Only 16-bit devices, 32-bit devices, and integer constants can be modified (real number constants, and character constants cannot be specified).
*3: Index register (I0 to IE)
*4: Only operation unit: signed integers (SS, SL) can be specified.
*5: Only operation unit: unsigned integers (US, UL) can be specified.
*6: Only operation unit: integers (US, SS, UL, SL) can be specified.
*7: Only operation unit: single-precision, floating-point real numbers (SF) can be specified.
*8: Only operation unit: double-precision, floating-point real numbers (DF) can be specified.

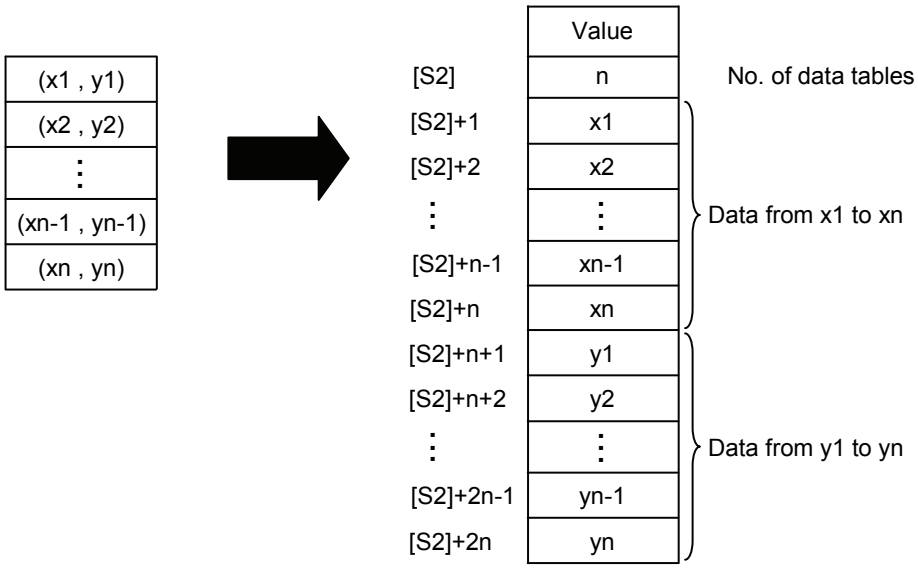
■ Outline of operation

- The data specified by [S1] are scaled according to the data table specified by [S2]. The result is stored in the device area specified by [D].
- [S2] is in the following format by operation unit.

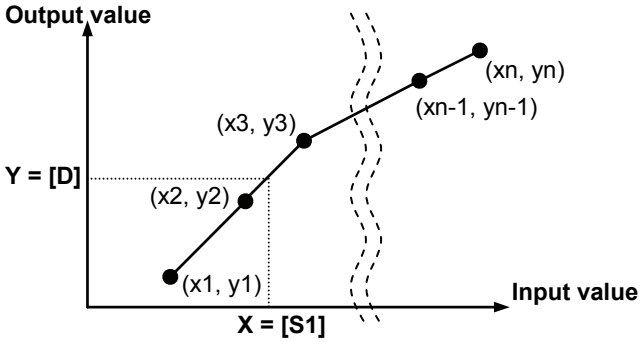
Operation unit	16 bits (US, SS)		32 bits (UL, SL, SF)		64 bits (DF)		Description of output
Device size	16 bits	32 bits	16 bits	32 bits	16 bits	32 bits	
Result storage Area	[S2]	Not available	[S2]	Not available	[S2]	Not available	No. of data tables Available range: 2 to 256
	[S2+1]	Not available	[S2+1] to [S2+2]	Not available	[S2+1] to [S2+4]	Not available	Data from X1 to Xn
	[S2+n+1]	Not available	[S2+n+1] to [S2+n+2]	Not available	[S2+n+1] to [S2+n+4]	Not available	Data from y1 to yn

■ Process details

- Regardless of the operation unit, the output value Y for [D], corresponding to the input value X for [S1], is calculated.



In response to the information of input value X as specified by [S1], referring to the [S2] data table, calculate the information of output value Y and store it into the [D] area.



<< Device reference by operation unit (n) >>

Example) Operation unit: 32 bits (UL, SL, SF) (specify a 16-bit device)

	Value
DT10・DT11	H 22220000
DT12・DT13	H 11113333
DT14・DT15	H55550000
DT16・DT17	H 22222222
DT18・DT19	H 11114444

・ The lower word and the higher word are referenced as a single set of 32-bit data.
・ Because of handling as a 32-bit data set, if the device specification forms a word border, up to [S2]+1 is included in the range.

■ Precautions during programming

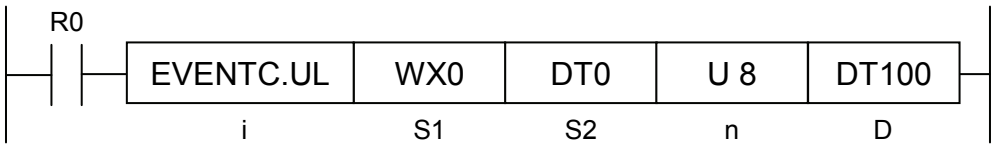
- Ensure [Xn-1 < Xn] for data in [S2]: data table.
- When [X([S1]) < x1], [Y([D]) = y1].
- When [X([S1]) > xn], [Y([D]) = yn].
- Max. value for [n], representing the no. of data in [S2]: data table, is 256.

■ Flag operations

Name	Explanation
SR7 SR8 (ER)	To be set in the case of out-of-range in indirect access (index modification, pointer access).
	To be set when [n < 2] or [n > 256], where [n] represents the no. of data in [S2]: data table.
	To be set when data in [S2]: data table exceeds the area.
	To be set when Xn is not in ascending order.

EVENTC (Instruction to Count the No. of Events)

■ Ladder diagram



■ Available operation units (●: Available)

Operation unit	bit	US	SS	UL	SL	SF	DF
i				●			

■ List of operands

Operand	Explanation
S1	Starting address of the counting start position
S2	Starting address of the working area for counting
n	No. of bits to be counted
D	Starting device address to store the count result

■ Available devices (●: Available)

Operand	16-bit device											32-bit device			Integers			Real numbers		String	Index modifier *1
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX	K	U	H	SF	DF	" "	
S1	●	●	●	●	●	●	●	●	●	●	●										●
S2	●	●	●	●			●	●													●
n	●	●	●	●			●	●	●	●	●					●	●				●
D	●	●	●	●			●	●	●		●										●

*1: Only 16-bit devices, 32-bit devices, and integer constants can be modified (real number constants, and character constants cannot be specified).

■ Outline of operation

- Count the number of ONs for [n] bits from the device specified by [S1].
- The count result is stored in the area of [n]*2 words starting with [D].

■ Process details

Example 1) Carry out counting for 8 points from WX0

[i]…UL [S1]…WX0 [S2]…DT0 [n]…U 8 [D]…DT10

[S1]

WX0	Targeted bits: 8 bits from X0 to X7

[S2]

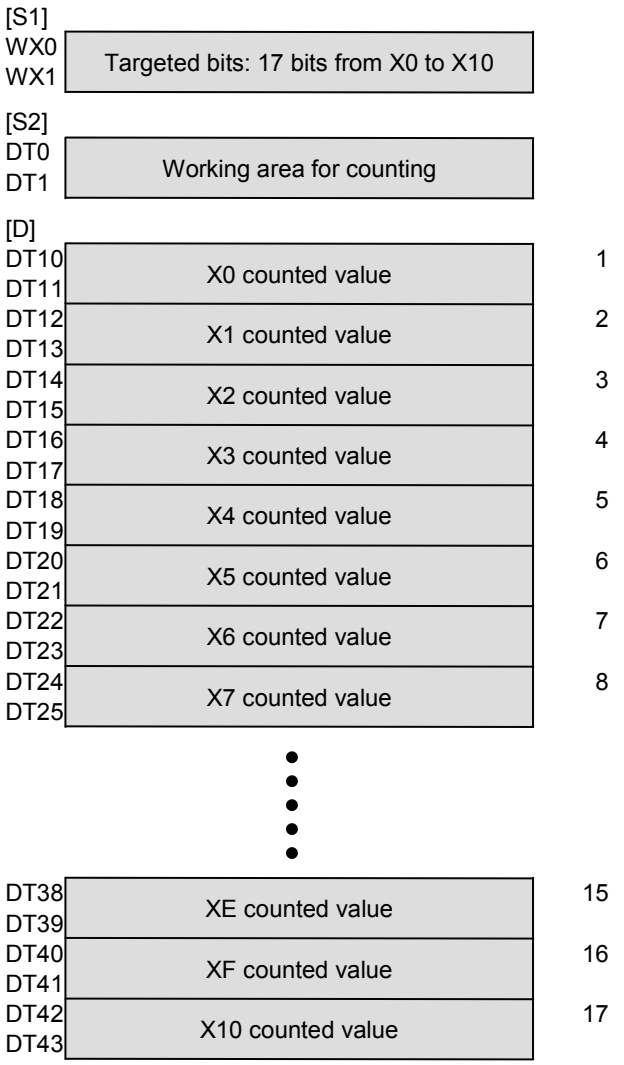
DT0	Working area for counting

[D]

DT10 DT11	X0 counted value	1
DT12 DT13	X1 counted value	2
DT14 DT15	X2 counted value	3
DT16 DT17	X3 counted value	4
DT18 DT19	X4 counted value	5
DT20 DT21	X5 counted value	6
DT22 DT23	X6 counted value	7
DT24 DT25	X7 counted value	8

Example 2) Carry out counting for 17 points from WX0

[i]…UL [S1]…WX0 [S2]…DT0 [n]…U 17 [D]…DT10

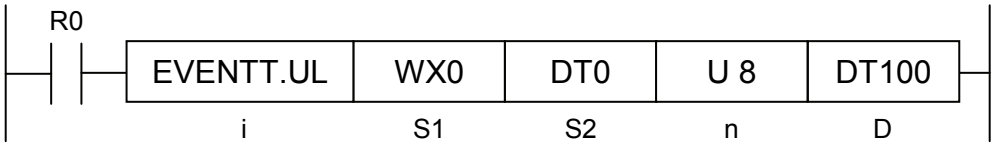


■ Flag operations

Name	Explanation
SR7	To be set in the case of out-of-range in indirect access (index modification, pointer access).
SR8 (ER)	When one of the count target area, the count working area or the count result storage area exceeds the device area.

EVENTT (Instruction to Count the Time of Events)

■ Ladder diagram



■ Available operation units (●: Available)

Operation unit	bit	US	SS	UL	SL	SF	DF
i				●			

■ List of operands

Operand	Explanation
S1	Starting address of the counting start position
S2	Starting address of the working area for counting
n	No. of bits to be counted
D	Starting device address to store the count result

■ Available devices (●: Available)

Operand	16-bit device											32-bit device			Integers			Real numbers		String	Index modifier *1
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX	K	U	H	SF	DF	" "	
S1	●	●	●	●	●	●	●	●	●	●	●										●
S2	●	●	●	●			●	●													●
n	●	●	●	●			●	●	●	●	●					●	●				●
D	●	●	●	●			●	●	●	●	●										●

*1: Only 16-bit devices, 32-bit devices, and integer constants can be modified (real number constants, and character constants cannot be specified).

■ Outline of operation

- Count the time of ONs for [n] bits from the device specified by [S1].
- The count result is stored in the area of [n]*2 words starting with [D].

■ Process details

Example 1) Carry out counting for 8 points from WX0

[i]…UL [S1]…WX0 [S2]…DT0 [n]…U 8 [D]…DT10

[S1]

WX0	Targeted bits: 8 bits from X0 to X7

[S2]

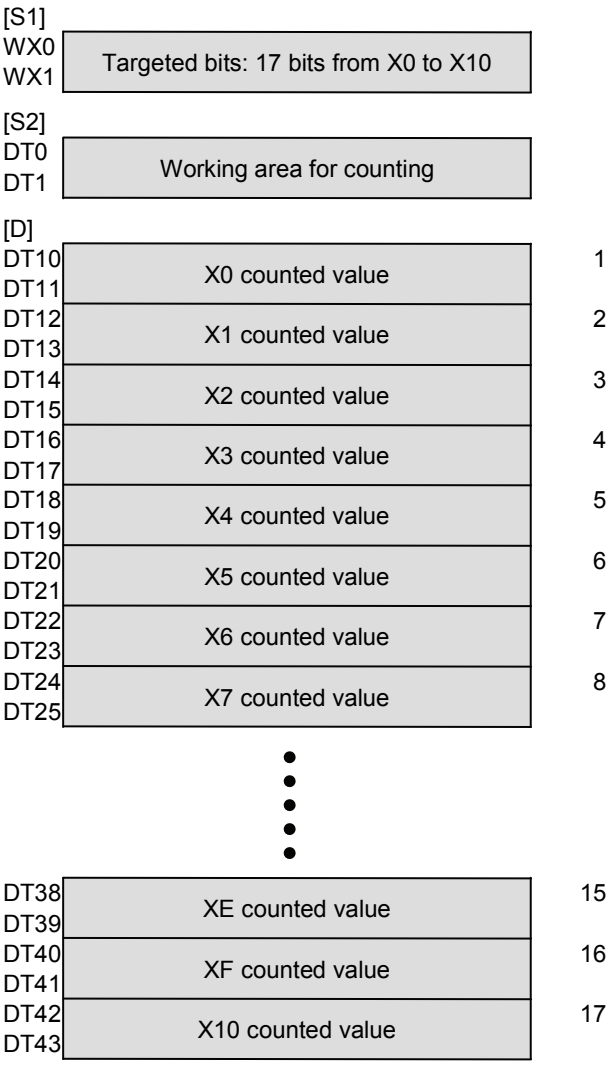
DT0	Working area for counting

[D]

DT10 DT11	X0 counted value	1
DT12 DT13	X1 counted value	2
DT14 DT15	X2 counted value	3
DT16 DT17	X3 counted value	4
DT18 DT19	X4 counted value	5
DT20 DT21	X5 counted value	6
DT22 DT23	X6 counted value	7
DT24 DT25	X7 counted value	8

Example 2) Carry out counting for 17 points from WX0

[i]…UL [S1]…WX0 [S2]…DT0 [n]…U 17 [D]…DT10

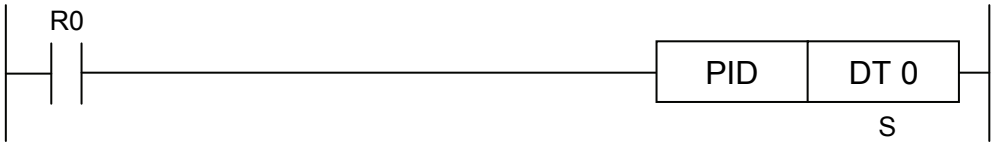


■ Flag operations

Name	Explanation
SR7	To be set in the case of out-of-range in indirect access (index modification, pointer access).
SR8 (ER)	When one of the count target area, the count working area or the count result storage area exceeds the device area.

PID (PID Operation)

■ Ladder diagram



■ Available operation units (●: Available)

- No operation units

■ List of operands

Operand	Explanation
S	Starting number of the PID operation parameter area (30 words)

■ Available device (●: Available)

Operand	16-bit device											32-bit device			Integers			Real numbers		String	Index modifier
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX	K	U	H	SF	DF	" "	
S							●														

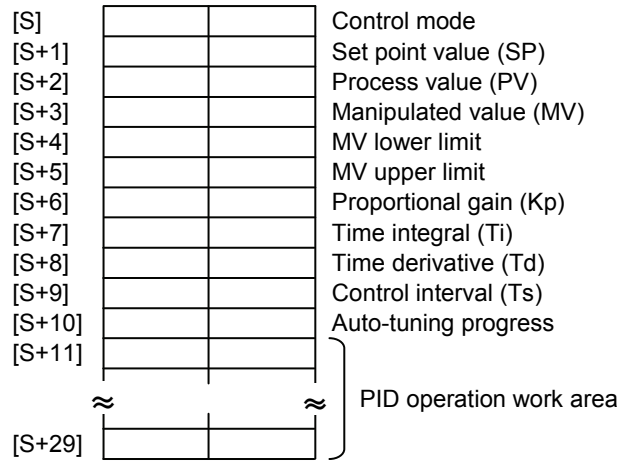
■ Outline of operation

- PID operation is carried out to retain the process value PV stored in [S+2], in consistency with the set point value SP specified by [S+1].
- The operation result is stored, as a manipulated value [MV], in the area specified by [S+3].
- Methods for PID operation (derivative-first / proportional-plus-derivative-first, reverse operation / forward operation) and coefficients used for PID operation (proportional gain, time integral, time derivative), as well as the types and interval of operation, are set to the parameter table [S] to [S+29].

■ Types of PID operation

Items	Explanation
Reverse operation / forward operation	Select the upward/downward direction of output in the case of change to the process. Specify "reverse operation" if output is increased when the process value decreases (e.g. heating). Specify "forward operation" if output is increased when the process value increases (e.g. cooling).
Derivative-first PID / proportional-plus-derivative-first PID	Derivative-first PID: Usually, the output variation becomes larger when the set point value is changed, but it converges faster.
	Proportional-plus-derivative-first PID: Usually, the output variation becomes smaller when the set point value is changed, but it converges slower.
Auto-tuning	By measuring process response, the respective optimal values for Kp, Ti and Td as PID parameters are measured. After auto-tuning is completed, the estimated results are reflected into the parameter area.

■ Setting of the parameter table



■ Setting of parameters: [S] to [S+23]

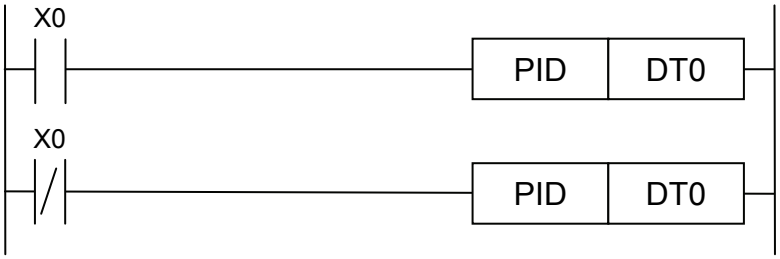
Operand	Parameter name	Setting range	Setting method
[S]	Control mode	H0 to H3	Auto-tuning is off H0: Derivative-first, reverse operation H1: Derivative-first, forward operation H2: Proportional-plus-derivative-first, reverse operation H3: Proportional-plus-derivative-first, forward operation
		H8000 to H8003	Auto-tuning is on H8000: Derivative-first, reverse operation H8001: Derivative-first, forward operation H8002: Proportional-plus-derivative-first, reverse operation H8003: Proportional-plus-derivative-first, forward operation
[S+1]	Set point value (SP)	K0 to K10000	Specify the target value for process control amount in the following range.
[S+2]	Process value (PV)	K0 to K10000	Input the current value for process control amount using an A/D converter, etc.
[S+3]	Manipulated value (MV)	K0 to K10000	PID operation result is stored. Output the value using a D/A converter, etc.
[S+4]	MV lower limit	K0 to K9999	Specify the range of manipulated value (MV). The value of the specified range is output.
[S+5]	MV upper limit	K1 to K10000	
[S+6]	Proportional gain (Kp)	K1 to K9999 (0.1 to 999.9)	Specify the coefficient to be used for PID operation. Actual proportional gain is set value x 0.1. (Note1)
[S+7]	Time integral (Ti)	K1 to K30000 (0.1 to 3000 s)	Specify the coefficient to be used for PID operation. Actual time integral is set point value x 0.1. The integral operation is not executed if 0 is specified. (Note1)
[S+8]	Time derivative (Td)	K0 to K10000 (0 to 1000 s)	Specify the coefficient to be used for PID operation. Actual derivative time is set point value x 0.1. The integral operation is not executed if 0 is specified. (Note1)
[S+9]	Control time (Ts)	K1 to 6000 (0.01 to 60.0 s)	Specify the interval to execute PID operation. Actual control interval is set point value x 0.01.
[S+10]	Auto-tuning progress	K0 to K5	When auto-tuning is specified in the control mode, this indicates the progress of auto-tuning. Starting with the default [0], a value between K1 and K5 is stored in accordance with the progress. After auto-tuning is completed, the value returns to the default.
[S+11] to [S+29]	PID operation work area	-	This work area is used by the system program for operation.

(Note 1) If auto-tuning is specified in the control mode, automatic adjustment is carried out, and the set point value is rewritten.

■ Precautions during programming

- The parameter table requires an area of 30 words, including the work area for operation. Ensure that this area value is not rewritten by other instructions.
- Error is not detected even if the parameter table exceeds the area. Ensure to specify a number at least 30 words before the final number for [S].
- Ensure that the area is not exceeded by index modification. Error is not detected even if the area is exceeded.
- As indicated below, the system does not operate normally if two or more PID instructions specifying the same table are described in your program. Even if execution conditions are not met, PID instructions operate internally using the specified table. In such cases, set the table to differing addresses.

Example:



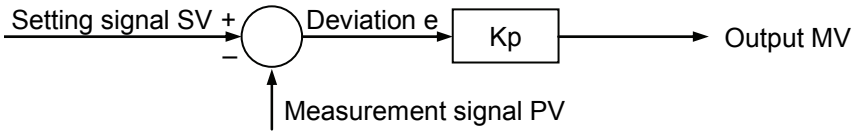
■ Precautions during auto-tuning

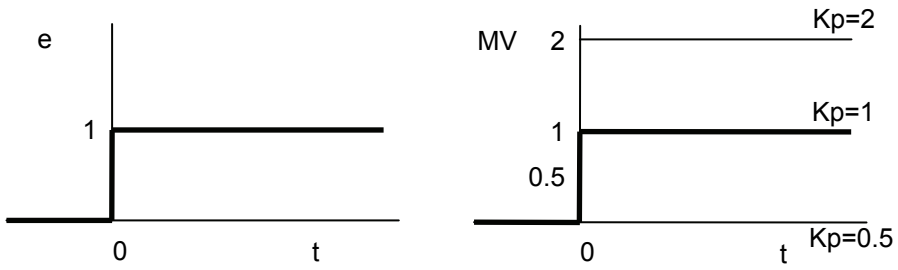
- Auto-tuning may not be executable depending on the process. In that case, the system returns to the original parameter operation.
- After auto-tuning is completed, the control mode [S] area is automatically rewritten from H 8000 through H 8003 to H 0 through H 3. Ensure that this is not rewritten again by your program, etc.
- After auto-tuning is completed, the respective optimal values are stored for proportional gain [Kp], time integral [Ti], and time derivative [Td]. Before execution, it is necessary to specify appropriate values (e.g. lower limits) within the respective setting ranges.
- After auto-tuning is completed, optimal values are stored for proportional gain [Kp], time derivative [Td], and time derivative [Td]. Ensure that the stored values are not rewritten.
- During auto-tuning, the output values for Kp, Ti and Td are calculated by measuring changes to process values (PVs) when manipulated values (MVs) are set to the upper limits, as well as changes to process values (PVs) when manipulated values (MVs) are set to the lower limits, so that process values (PVs) will be increased or decreased in accordance with the respective set point values (SPs).
- Changes to manipulated values (MVs) are completed following, at least three sessions (upper limit MV - lower limit MV - upper limit MV). If the auto-tuning progress does not change from 0 following changes of multiple sessions, shorten the control synchronization Ts and retry auto-tuning.

■ Overview of PID control

(1) Proportional operation

A proportional part generates an output that is proportional to the input.

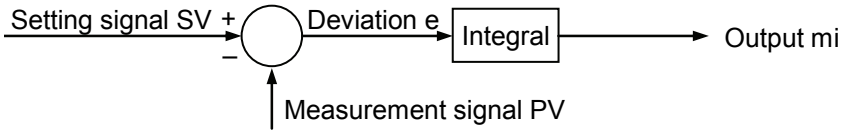




- Control amount is retained at a specified level.
- Offset (steady-state error) remains.
- The larger the Kp is, the stronger proportional operation occurs.

(2) Integral operation

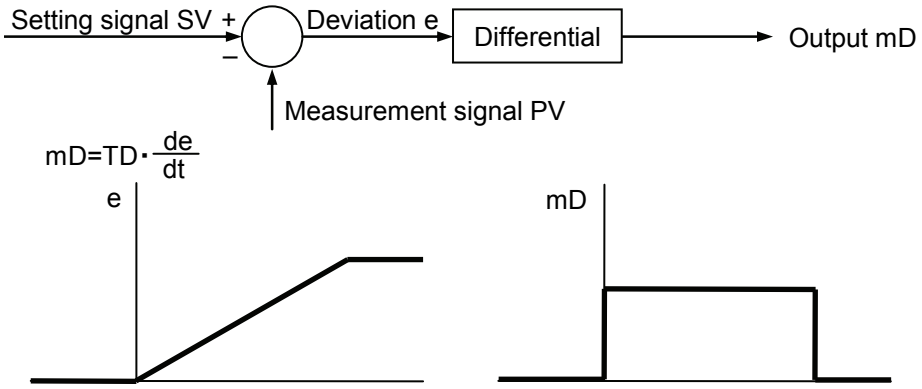
An integral part produces an output quantity that corresponds to the time integral and input quantity.



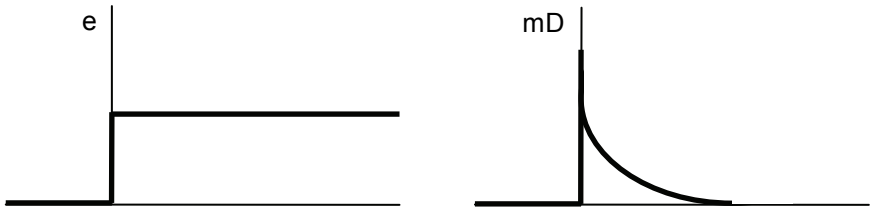
$mi = 1/Ti \int edt$

- In combination with proportional operation or proportional plus derivative operation, this removes the generated offset.
- The smaller the Ti is, the stronger integration occurs.

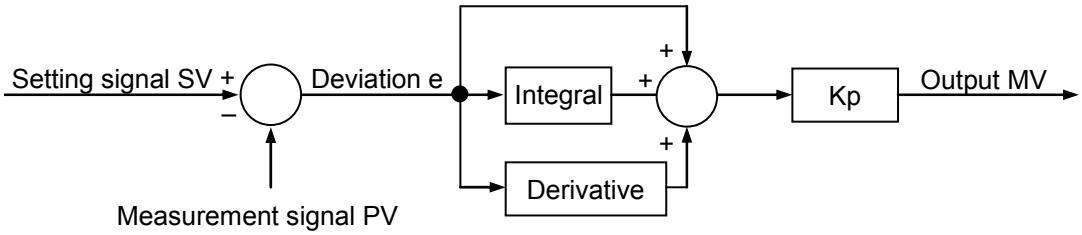
(3) Derivative operation
The derivative part produces an output quantity that corresponds to the time derivation of the input quantity.



- In accordance with the proceeding characteristics of derivation, the process delaying characteristics reduce negative influence on controls.
- The larger the Td is, the stronger derivation occurs.
- Because pure derivation may become temporarily disabled due to noise input, etc., causing negative influence on the operating terminal, inexact derivation is executed.



(4) PID operation
A PID operation is a combination of proportional, integral, and derivative operations.



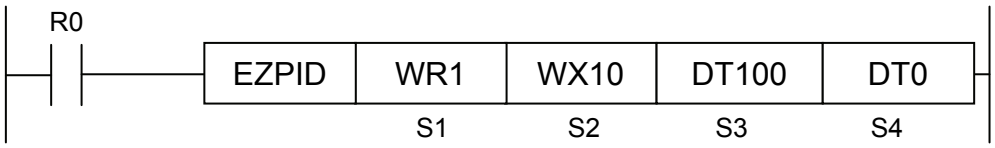
- When the parameters are optimally adjusted, a PID controller can quickly control and maintain a quantity at a predetermined set value.

■ Flag operations

Name	Explanation
SR7	To be set in the case of out-of-range in indirect access (index modification).
SR8 (ER)	To be set when a parameter is out of the setting range.

EZPID (PID Operation: PWM Output Available)

■ Ladder diagram



■ Available operation units (●: Available)

- No operation units

■ List of operands

Operand	Explanation
S1	1-word area for setting control data that determine methods for auto-tuning and manipulated value (MV)
S2	1-word area for inputting the process value (PV)
S3	4-word area for setting the set point value (SV), proportional gain (Kp), time integral (Ti), and time derivative (Td).
S4	30-word area comprising the following: storage area for the manipulated value (MV), control interval (Ts), control mode, setting area for parameters related to auto-tuning, and operation work area

■ Available devices (●: Available)

Operand	16-bit device											32-bit device			Integers			Real numbers		String	Index modifier
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX	K	U	H	SF	DF	" "	
S1	●	●	●	●			●	●													
S2	●	●	●	●	●	●	●	●													
S3	●	●	●	●			●	●													
S4	●	●	●	●			●	●													

■ Outline of operation

- PID operation is carried out to retain the process value (PV) stored in [S+2], in consistency with the set point value (SP) specified by [S3].
- The operation result is stored, as a manipulated value (MV), in the area specified by [S4]. If an OUT instruction is described immediately after this instruction, PMW output (ON-OFF output) can be gained, proportionate to the manipulated value.
- Parameters used for PID operation (proportional gain, time integral and time derivative) are set to [S3+1] to [S3+3]. Auto-tuning function for automatically calculating these values is also available.
- Methods for PID operation (derivative-first / proportional-plus-derivative-first, reverse operation / forward operation), as well as changes to the control interval Ts, etc., are set to the areas [S4] to [S4+9].

■ Types of PID operation

Items	Explanation
Reverse operation / forward operation	Select the upward/downward direction of output in the case of change to the process. Specify "reverse operation" if output is increased when the process value decreases (e.g. heating). Specify "forward operation" if output is increased when the process value increases (e.g. cooling).
Derivative-first PID / proportional-plus-derivative-first PID	Derivative-first PID: Usually, the output variation becomes larger when the set point value is changed, but it converges faster.
	Proportional-plus-derivative-first PID: Usually, the output variation becomes smaller when the set point value is changed, but it converges slower.
Auto-tuning	By measuring process response, the respective optimal values for Kp, Ti and Td as PID parameters are measured. After auto-tuning is completed, the estimated results are reflected into the parameter area.

(Note 1) By default, controls are carried out in reverse operation, derivative-first. To change operation methods, change values in the [S4+5] area before the second execution of EZPID instruction.
(Note 2) To execute auto-tuning, set so in the area of control data [S1].

■ Setting of parameters [S1] to [S3]

Operand	Parameter name	Setting range	Setting method
[S1]	Control data	Bit 0	Bit 0 = 1 (ON): Auto-tuning request is issued. After auto-tuning is completed, this is reset when the EZPID instruction is executed. Also reset this bit to cancel auto-tuning. Bit 0 = 0 (OFF): PID control is executed.
		Bit 1	When auto-tuning is completed normally, Bit 1 = 1 (ON) is set. This is reset to Bit 1 = 0 (OFF) at the start of instruction execution.
		Bit 2	Specify whether the [S4] manipulated value (MV) should be cleared when the execution condition for the instruction is OFF > ON. Bit 2 = 0 (OFF): Clear the manipulated value (MV) at the time of execution of the previous instruction. Bit 2 = 1 (ON): Retain the manipulated value (MV) at the time of execution of the previous instruction.
		Bit 3	Select a method for outputting the result of PID operation. Bit 3 = 0 (OFF): PWM output Bit 3 = 1 (ON): Analog output
		Bit 4	Specify the range (max. value and min. value) for internal calculation of the manipulated value (MV). Bit 4 = 0 (OFF): Respectively +20% and -20% of difference between the output upper limit and the output lower limit Bit 4 = 1 (ON): Use the output upper limit and the output lower limit
		Bit F to Bit 5	Bits 5 to F are reserved bits. Use them as 0.
[S2]	Process value (PV)	K -30000 to K 30000	Input the current value for process control amount using an analog input unit, etc. It is also possible to directly specify the input data area WXn of the analog input unit.
[S3]	Set point value (SP)	K -30000 to K 30000	Set a target value for process control amount. Set the value using an instruction or an external device (e.g. display).
[S3+1]	Proportional gain (Kp)	U1 to U9999 (0.1 to 999.9)	Specify parameters used for PID operation. The respective parameter value is set point value x 0.1. When auto-tuning is executed, the relevant data are stored upon its completion.
[S3+2]	Time integral (Ti)	U0 to U0000 (0 to 3000 s)	
[S3+3]	Time derivative (Td)	U0 to U10000 (0 to 1000 s)	

(Note 1) It is recommended to allocate the [S1] area to a non-hold-type operation memory area (WR), capable of bit operation, for use.
(Note 2) It is recommended to allocate the [S2] area to a non-hold-type operation memory area, for use.
(Note 3) It is recommended to allocate the [S3] through [S3+3] areas to a hold-type operation memory area, for use.
(Note 4) If the [S3+1] through [S3+3] areas are all "0" when the EZPID instruction is started up, operation is continued with proportional gain (Kp) = 1, time integral (Ti) = 0, and time derivative (Td) = 0.

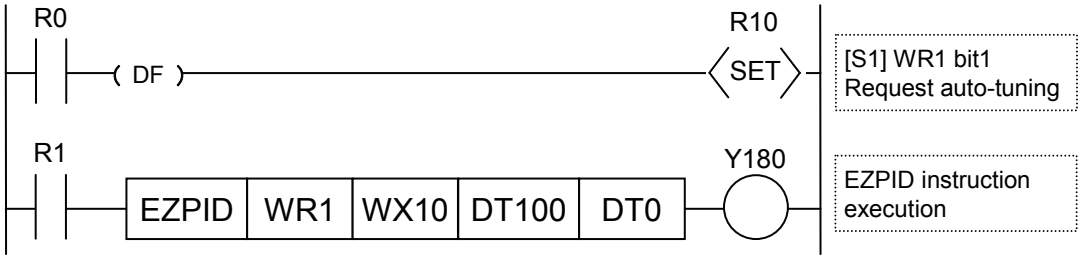
■ Setting of parameters [S4] to [S4+29]

Operand	Parameter name	Default	Setting range	Setting method
[S4]	Manipulated value (MV)	K0	K -10000 to K 10000	• The result of PID operation is stored. • If PWM output is selected in Bit 3 of [S1], calculation is carried out in the range 0 to 10000, as the duty ratio (0 to 100%) of PWM output. • If analog output is selected in Bit 3 of [S1], the value given by the following formula is stored. The stored value is converted into the range of the analog output unit, and is output. (Output upper limit - Output lower limit) x Internal calculated value / 10000 + Lower limit
[S4+1]	MV lower limit	K0	K-10000 ≤ lower limit < upper limit ≤ K 10000	Specify the range of manipulated value (MV). The value of the specified range is output.
[S4+2]	MV upper limit	K10000		
[S4+3]	100% output zone	U0 (0%)	U0 to U80 (0% to 80%)	• Specify the level of process value (PV), as percentage to set point value, above which PID control should be initiated. Until the specified process value (PV) is reached, the manipulated value (MV) is retained at 100%. If the process value (PV) is smaller than the set point value (SP), this shortens lead-time until reaching the set point value (SP). • When the set point value is K80, the output level is retained at 100% until the process value (PV) reaches 80% of the set point value (SP). Once the 80% is exceeded, PID control is initiated. If the set point value is K0 (default), PID control is executed from the beginning.
[S4+4]	Control interval (Ts)	U100 (1s)	U1 to 6000 (0.01 to 60.0 s)	Specify the interval to execute PID operation. Actual control interval is set point value x 0.01.
[S4+5]	Control mode	U0 (derivative-first, reverse operation)	U0 to U3	U0: Derivative-first, reverse operation U1: Derivative-first, forward operation U2: Proportional-plus-derivative-first, reverse operation U3: Proportional-plus-derivative-first, forward operation
[S4+6]	Auto-tuning bias value	U0	U0	In order to control overheating during auto-tuning, specify difference between the set point value during auto-tuning and the actual set point value. (Note 3)
[S4+7]	Auto-tuning proportional gain (Kp) correction coefficient	U125 (125%)	U50 to U500 (50 to 500%)	By setting coefficients listed on the left, and executing auto-tuning, parameter values obtained through auto-tuning are multiplied by the respective coefficients, and stored in the parameter setting areas [S3+1] through [S3+3]. Example) If U200 is set to [S4+7], the value given as 200% of the proportional gain Kp, obtained through auto-tuning, is set to [S3+1].
[S4+8]	Auto-tuning time integral (Ti) correction coefficient	U200 (200%)	U50 to U500 (50 to 500%)	
[S4+9]	Auto-tuning time derivative (Td) correction coefficient	U100 (100%)	U50 to U500 (50 to 500%)	
[S4+10]	Auto-tuning progress	U0	U0 to U5	This indicates the progress of auto-tuning. Starting with the default [0], a value between U1 and U5 is stored in accordance with the progress. After auto-tuning is completed, the value returns to U0.
[S4+11] to [S4+29]	PID operation work area	K0	-	This work area is used by the system program for operation. Ensure that this is not overwritten by your program.

(Note 1) By default, the [S4] manipulated value (MV) is cleared when the EZPID instruction is started up. If it is necessary to retain the value from the previous execution, set 1 to Bit 2 of the operand [S1].
(Note 2) The areas [S4+1] through [S4+29] should be preset to default when the EZPID instruction is started up.
(Note 3) For details of the auto-tuning bias value, see Setting of auto-tuning bias value [S4+6].

■ Sample program (PWM output)

- In this sample, PWM output by PID operation is carried out, after setting proportional gain (Kp), time integral (Ti), and time derivative (Td) through auto-tuning.



- Set the set point value (SP) to the operand [S3] area, using an instruction or a display.
- Using an instruction or a display, set Bit 0 (auto-tuning request flag) of [S1]. Subsequently, switch the execution condition for the EZPID instruction to ON, to initiate auto-tuning operation.
- Once auto-tuning is completed normally, parameters Kp, Ti and Td are set to the operand [S3+1] through [S3+3] areas. At this time, Bit 0 (auto-tuning request flag) of the operand [S1] is reset to OFF, and Bit 1 (auto-tuning completion flag) of the operand [S1] is switched ON.
- If the execution condition for the EZPID instruction is ON, PID control and PWM output are initiated in the following scan.
- The value (K0 to K10000) of the operand [S4] manipulated value (MV) is converted into a duty ratio from 0 to 100%, and PWM output is carried out.
- The operand [S4] through [S4+29] areas are preset to default when the execution condition for the EZPID instruction is switched on. If the [S4] manipulated value (MV) should not be cleared, set 1 to Bit 2 of [S1] before executing the EZPID instruction.
- If the execution condition of the EZPID instruction is switched OFF during PID control, PWM output Y180 is also turned OFF. At this time, the value of the operand [S4] manipulated value (MV) is retained.

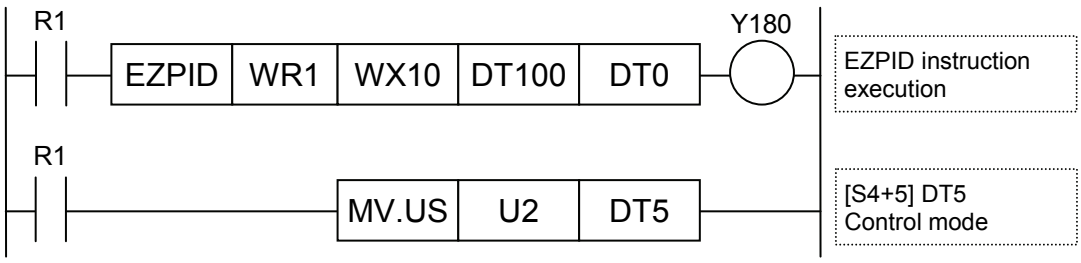
■ Interval and duty ratio of PWM output

- The duty of PWM is determined by the ratio of the manipulated value (MV), stored in [S4], to K0 through K10000.
- If the manipulated value (MV) is K0, PWM output is always OFF. If the MV is K10000, PWM output is always ON.
- The PWM output interval is determined by the control interval value specified by the operand [S4+4]. By default, the interval is set at 1 second.

■ Sample program (change of control conditions)

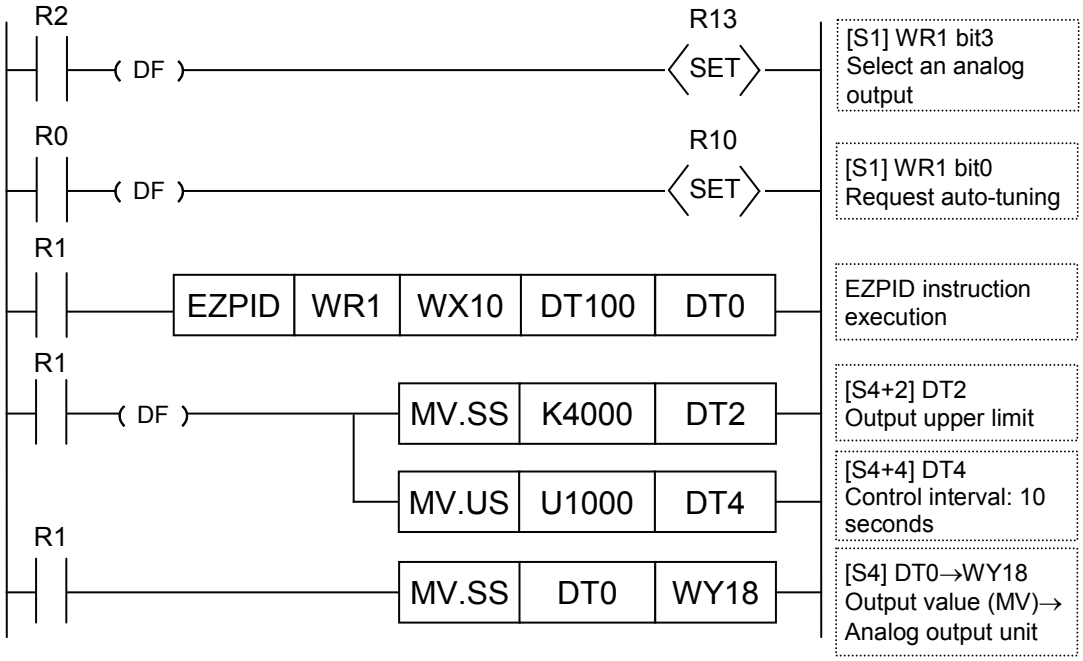
- By default, the following control conditions are used for operation: 1) Operation interval Ts: 1 second; 2) Control method: derivative-first, reverse operation (heating)
- In order to modify the control conditions, change values of the operands [S4+1] through [S4+9] using the MV instruction, etc.
- Carry out modifications after execution of the EZPID instruction, and before its second execution.

[Example] Switch the control mode to proportional-plus-derivative-first, and carry out PWM output.



■ Sample program (analog output)

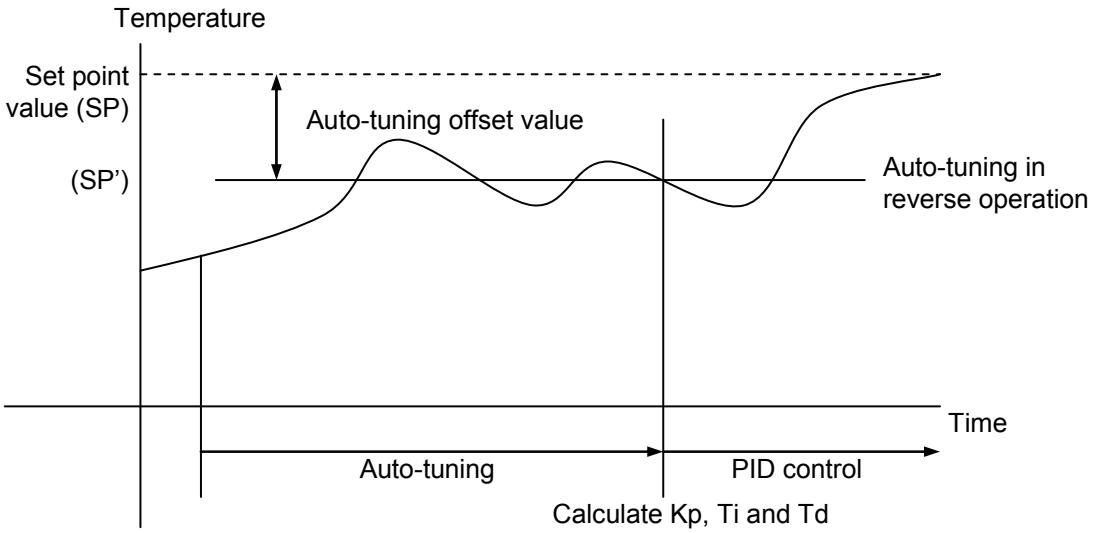
[Example] In analog output, change the output upper limit [S4+2] to K4000, and the control interval [S4+4] to K1000 (10 seconds).

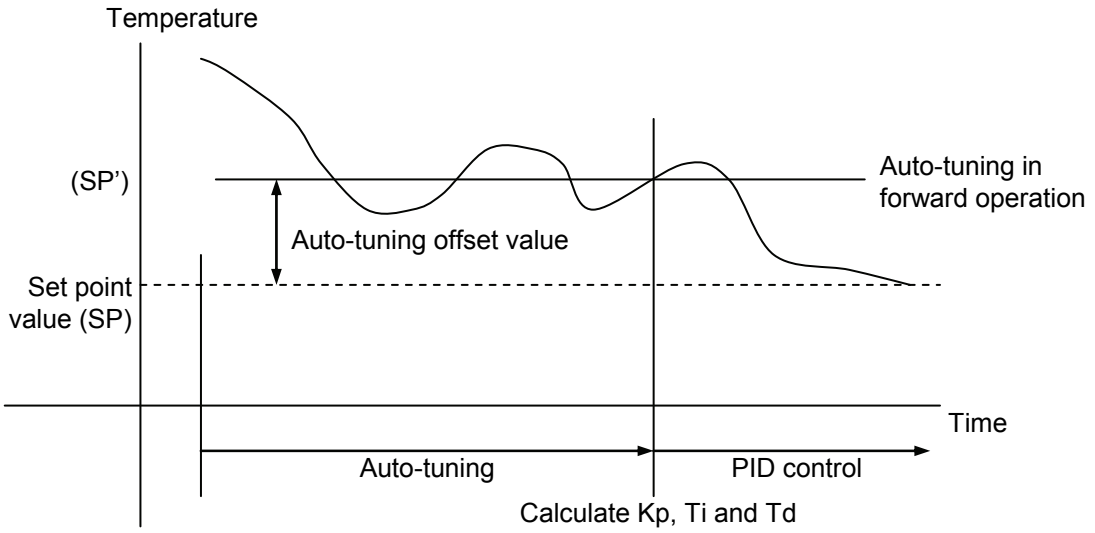


- In order to use analog output, set 1 to Bit 3 of the operand [S1].
- The output lower limit [S4+1] and the output upper limit [S4+2] should be set according to the output range of the analog output unit.
- The value of control interval (Ts): [S4+4] should be modified according to the input updating interval (normally ≥ 0.1 second) of the analog input unit.
  Example) If the [S4+4] value is K10, Ts = 100 ms.
- Control mode and other parameters should be modified as necessary.
- The manipulated value (MV) stored in the operand [S4] is transferred to the digital value output area WY that corresponds to MV of the analog output unit.
- In the [S4] manipulated value (MV), the output internal calculated value (K0 to K10000) is converted by the following formula, and stored.
  Conversion formula: (Output upper limit - Output lower limit) x (Internal calculated value) / 10000 + (Output lower limit)
- If [S4] is allocated to a hold-type area, the manipulated value (MV) is retained even if the execution condition of the EZPID instruction is switched OFF.
- If analog output is to be used, it is not necessary to describe the OUT instruction following the EZPID instruction. When analog output is to be used, PWM output is turned OFF.

■ Setting of auto-tuning bias value [S4+6]

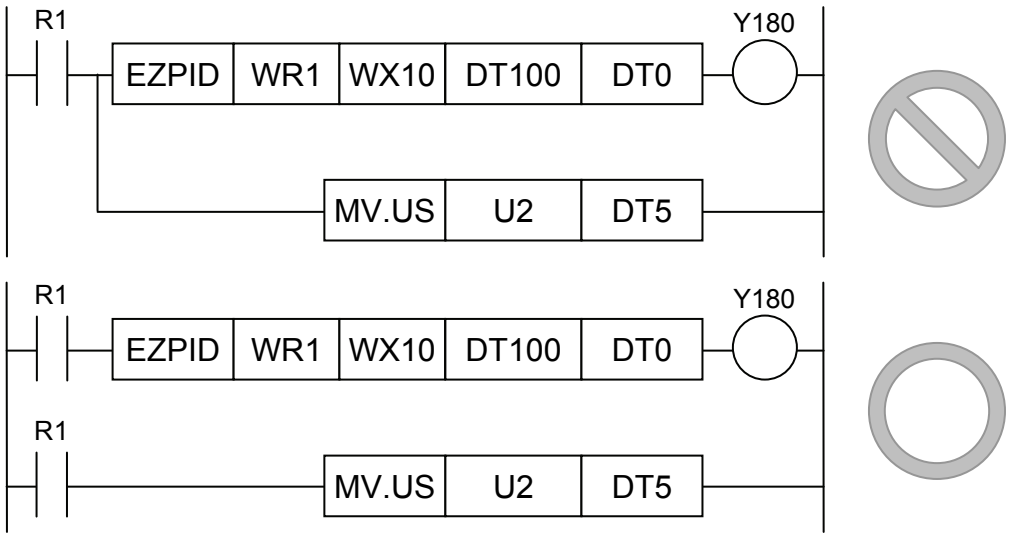
- If it is necessary to control output during auto-tuning (e.g. prevent overheating), set, as an bias value, the difference between the actual set point value (SP) during PID control operation and the set point value (SP') during auto-tuning.
- In the case of reverse operation (heating), (SP) ≥ (SP') and the difference thereof is given as the bias value.
- In the case of forward operation (cooling), (SP) ≤ (SP') and the difference thereof is given as the bias value.
- Even if auto-tuning is started up, with the process value (PV) located near the set point value (SP), auto-tuning is executed using the set point value (SP') adjusted with the bias value.





■ Precautions during programming

- The operand [S4] through [S4+29] areas are initialized, when the execution condition is switched on. If values other than default are to be used, set them using the MV instruction, etc.
- PID operation instructions are always internally calculating operation intervals, PWM output timings, etc. Ensure that the operation instructions are carried out only once in every scan. Ensure that they are not executed in a sub routine or an interrupt program. At the same time, it is not possible to describe multiple EZPID instructions specifying the same operand.
- Do not switch off the execution conditions during PID operation. It will disable PID operation.
- When multiple targets are to be controlled, and if the PWM output interval is not to be synchronized, then use different startup timings by adjusting the launch times of startup conditions, etc.
- The system does not operate normally if two or more PID instructions specifying the same table are described in your program. Even if execution conditions are not met, PID instructions operate internally using the specified table. In such cases, set the table to differing addresses.
- Ensure that the work area for PID operation is not rewritten by other instructions.
- Error is not detected even if the parameter table exceeds the area. Ensure to specify a number at least 30 words before the final number, when specifying the [S4] area. Also ensure that the area is not exceeded by index modification. Error is not detected even if the area is exceeded.
- Execution conditions vary immediately after the EZPID instruction, just like PWM output. Therefore, the system does not operate normally if the subsequent instructions are described.



■ Precautions during auto-tuning

- Auto-tuning may not be executable depending on the process. In that case, the system returns to the original parameter operation.
- After auto-tuning is completed, the respective optimal values are stored for proportional gain [Kp], time integral [Ti], and time derivative [Td]. Before execution, it is necessary to specify appropriate values (e.g. lower limits) within the respective setting ranges.
- After auto-tuning is completed, optimal values are stored for proportional gain [Kp], time derivative [Td], and time derivative [Td]. Ensure that the stored value is not rewritten.
- During auto-tuning, the output values for Kp, Ti and Td are calculated by measuring changes to process values (PVs) when manipulated values (MVs) are set to the upper limits, as well as changes to process values (PVs) when manipulated values (MVs) are set to the lower limits, so that process values (PVs) will be increased or decreased in accordance with the respective set point values (SPs).
- Changes to manipulated values (MVs) are completed following, at least three sessions (upper limit MV - lower limit MV - upper limit MV). If the auto-tuning progress does not change from 0 following changes of multiple sessions, shorten the control synchronization Ts and retry auto-tuning.

■ Operation actions

- [S4] through [S4+29] are initialized, when the execution condition is switched on.
- If the parameters Kp, Ti and Td are all 0 at the start of PID operation, they are respectively initialized to 1, 0 and 0, and operation is continued.
- When the auto-tuning request signal is switched on, Bit 2 of [S1] (auto-tuning completion flag) and [S4+10] (auto-tuning completion code) are cleared.
- Set point values for auto-tuning are operated with [Set point value (SP) - Bias value] as the target value.
- After auto-tuning is normally completed, values given by multiplying the calculated Kp, Ti and Td, with the correction coefficients specified by [S4+7] through [S4+9], are stored.
- After auto-tuning is normally completed, Bit 2 of [S1] (auto-tuning completion flag) and the auto-tuning completion code are stored in [S4+10].
- When [S1] Bit 3 = 0: PWM output, output is carried out in the range from 0 to 10000,
  with the duty given by the conversion formula: (Upper limit - Lower limit) x Internal calculated value / 10000.
- When [S1] Bit 3 = 1: analog output, the internal calculated value is output in the range from 0 to 10000, converted by the following formula, and set to the operand [S4] as the manipulated value (MV).
  Conversion formula: (Upper limit - Lower limit) x Internal calculated value / 10000 + Lower limit

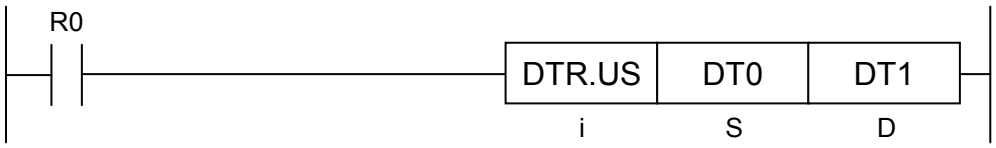
■ Flag operations

Name	Explanation
SR7 SR8 (ER)	To be set when the following parameters are out of the setting range: [S2]: process value (PV); [S3]: set point value (SP); [S3]+1: KP; [S3]+2: TI; [S3]+3: TD; and [S4]+4 through [S4]+9.
	To be set when an area specified by [S3] or [S4] exceeds the upper limit of the specified operation device.
	To be set in the case of out-of-range in indirect access (index modification).

■ **MEMO**

DTR (Data Revision Detection)

■ Ladder diagram



■ Available operation units (●: Available)

Operation unit	bit	US	SS	UL	SL	SF	DF
i		●	●	●	●	●	●

■ List of operands

Operand	Explanation
S	Device address to detect revision of a data value
D	Device address to store the data value from the previous execution

■ Available devices (●: Available)

Operand	16-bit device											32-bit device *1			Integers			Real numbers		String	Index modifier
	WX	WY	WR	WL	WS	SD	DT"	LD	UM	WI	WO	TS CS	TE CE	IX	K	U	H	SF	DF	"	*2
S	●	●	●	●	●	●	●	●	●	●	●	●	●	●							●
D	●	●	●	●			●	●	●		●	●	●	●							●

*1: Operation unit: 16-bit integers (SS, US) cannot be specified.
*2: Only 16-bit devices, 32-bit devices, and integer constants can be modified (real number constants, and character constants cannot be specified).

■ Outline of operation

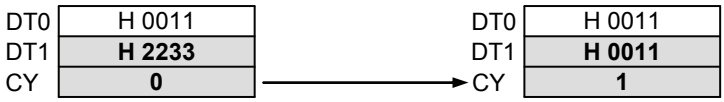
- If data in the device address specified by [S] have been changed from the values in the previous execution, SR9 (CY) is switched on.

■ Process details

Example 1) Operation unit: 16 bits (US, SS)

[i]…US,SS

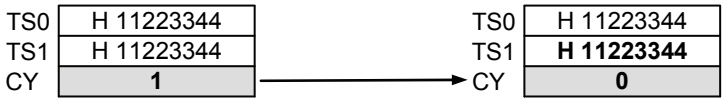
[S]…DT0 [D]…DT1



Example 2) Operation unit: 32 bits (UL, SL, SF)

[i]…UL,SL,SF

[S]…TS0 [D]…TS1



■ Precautions during programming

- Even when the operation unit is a real number, only data changes are checked. Type check for non-real numbers, etc. is not executed.

■ Flag operations

Name	Explanation
SR7 SR8 (ER)	To be set in the case of out-of-range in indirect access (index modification, pointer access).
SR9(CY)	To be set if the [S] value differs from the [D] value.
	To be reset if the [S] value equals the [D] value.

RAMP (Ramp Output)

■ Ladder diagram

■ Available operation units (●: Available)

Operation unit	bit	US	SS	UL	SL	SF	DF
i		●	●	●	●	●	●

■ List of operands

Operand	Explanation
S1	Device address where the default is stored, or constant
S2	Device address where the target value is stored, or constant
S3	Device address that stores the time width, or constant (data available range: 1 to 30000)
D	Output storage device address

■ Available devices (●: Available)

Operand	16-bit device											32-bit device *1			Integers			Real numbers		String	Index modifier
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX *3	K *4	U *5	H *6	SF *7	DF *8	" "	*2
S1	●	●	●	●			●	●				●	●	●	●	●	●	●	●		●
S2	●	●	●	●			●	●				●	●	●	●	●	●	●	●		●
S3	●	●	●	●			●	●				●	●	●	●	●	●	●	●		●
D	●	●	●	●			●	●				●	●	●							●

*1: Operation unit: 16-bit integers (SS, US) cannot be specified.
*2: Only 16-bit devices, 32-bit devices, and integer constants can be modified (real number constants, and character constants cannot be specified).
*3: Index register (I0 to IE)
*4: Only operation unit: signed integers (SS, SL) can be specified.
*5: Only operation unit: unsigned integers (US, UL) can be specified.
*6: Only operation unit: integers (US, SS, UL, SL) can be specified.
*7: Only operation unit: single-precision, floating-point real numbers (SF) can be specified.
*8: Only operation unit: double-precision, floating-point real numbers (DF) can be specified.

■ Outline of operation

- Scaling is carried out from the output default value, output target value, and output time (in ms), and linear output is executed in accordance with the time elapsed from the execution start.

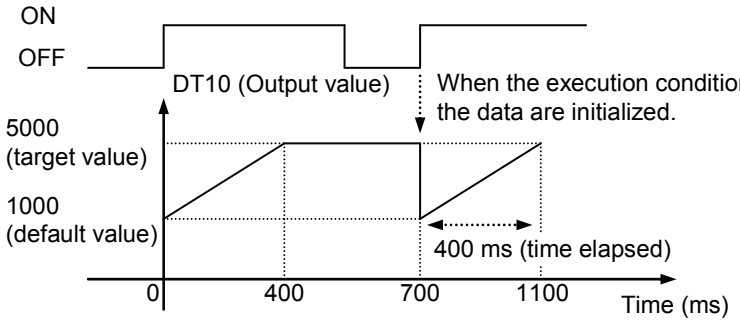
■ Process details

Example 1) Operation unit: 16 bits (US, SS)

[i]…US,SS

[S1]…DT0：K1000　[S2]…DT1：K5000　[S3]…DT2：K400　[D]…DT10

Execution condition

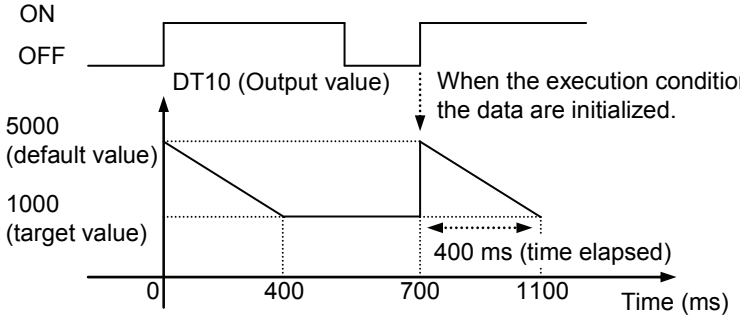


Example 2) Operation unit: 32 bits (UL, SL)

[i]…UL,SL

[S1]…DT0：K5000　[S2]…DT1：K1000　[S3]…DT2：K400　[D]…DT10

Execution condition

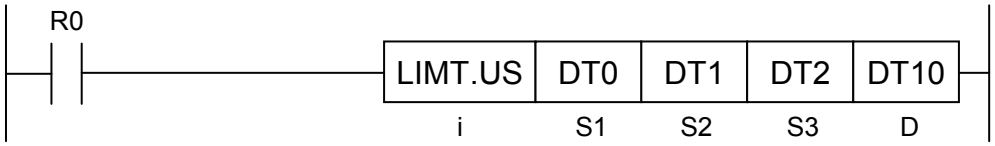


■ Flag operations

Name	Explanation
SR7	To be set in the case of out-of-range in indirect access (index modification, pointer access).
SR8 (ER)	To be set when the output time width specified by [S3] is out of the accessible range.

LIMT (Upper and Lower Limit Control)

■ Ladder diagram



■ Available operation units (●: Available)

Operation unit	bit	US	SS	UL	SL	SF	DF
i		●	●	●	●	●	●

■ List of operands

Operand	Explanation
S1	Device address where the lower limit is stored, or lower limit data
S2	Device address where the upper limit is stored, or upper limit data
S3	Device address where the input value is stored, or input value data
D	Output storage device address

■ Available devices (●: Available)

Operand	16-bit device											32-bit device *1			Integers			Real numbers		String	Index modifier
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX *3	K *4	U *5	H *6	SF *7	DF *8	" "	*2
S1	●	●	●	●			●	●				●	●	●	●	●	●	●	●		●
S2	●	●	●	●			●	●				●	●	●	●	●	●	●	●		●
S3	●	●	●	●			●	●				●	●	●	●	●	●	●	●		●
D	●	●	●	●			●	●				●	●	●							●

*1: Operation unit: 16-bit integers (SS, US) cannot be specified.
*2: Only 16-bit devices, 32-bit devices, and integer constants can be modified (real number constants, and character constants cannot be specified).
*3: Index register (I0 to IE)
*4: Only operation unit: signed integers (SS, SL) can be specified.
*5: Only operation unit: unsigned integers (US, UL) can be specified.
*6: Only operation unit: integers (US, SS, UL, SL) can be specified.
*7: Only operation unit: single-precision, floating-point real numbers (SF) can be specified.
*8: Only operation unit: double-precision, floating-point real numbers (DF) can be specified.

■ Outline of operation

- The output value, to be stored in the device address specified by [D], is controlled based on whether or not the input value specified by [S3] falls within the range bounded by the upper and lower limits set in [S1] and [S2].
- Output values are defined as follows:

When Lower limit [S1] > Input value [S3],	Lower limit [S1] → Output value [D]
When Upper limit [S2] < Input value [S3],	Upper limit [S2] → Output value [D]
When Lower limit [S1] ≤ Input value [S3] ≤ Upper limit [S2],	Input value [S3] → Output value [D]

- For control by the upper limit only, specify the min. value for the relevant operation unit as the lower limit [S1].
- For control by the lower limit only, specify the max. value for the relevant operation unit as the upper limit [S1].

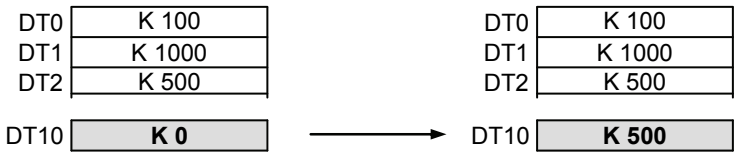
	US	SS	UL	SL	SF
Min. value	0	**-32768**	0	**-2147483648**	**Negative infinite**
Max. value	65535	32767	4294967295	2147483647	Positive infinite

■ Process details

Example 1) Operation unit: 16 bits (US, SS)

[i]…US,SS

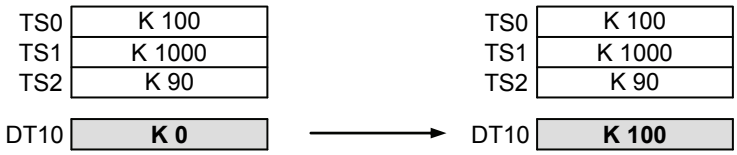
[S1]…DT0 [S2]…DT1 [S3]…DT2 [D]…DT10



Example 2) Operation unit: 32 bits (UL, SL)

[i]…UL,SL

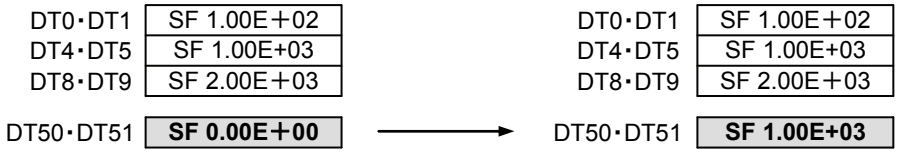
[S1]…TS0 [S2]…TS1 [S3]…TS2 [D]…DT10



Example 3) Operation unit: Single-precision, floating-point real number (SF)

[i]…SF

[S1]…DT0 [S2]…DT4 [S3]…DT8 [D]…DT50



■ Flag operations

Name	Explanation
SR7	To be set in the case of out-of-range in indirect access (index modification, pointer access).
SR8 (ER)	To be set when [S1] > [S2].

BAND (Deadband Control)

■ Ladder diagram

■ Available operation units (●: Available)

Operation unit	bit	US	SS	UL	SL	SF	DF
i			●		●	●	●

■ List of operands

Operand	Explanation
S1	Device address where the lower limit is stored, or lower limit data
S2	Device address where the upper limit is stored, or upper limit data
S3	Device address where the input value is stored, or input value data
D	Output storage device address

■ Available devices (●: Available)

Operand	16-bit device											32-bit device *1			Integers			Real numbers		String	Index modifier *2
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX *3	K *4	U	H *5	SF *6	DF *7	" "	
S1	●	●	●	●			●	●				●	●	●	●		●	●	●		●
S2	●	●	●	●			●	●				●	●	●	●		●	●	●		●
S3	●	●	●	●			●	●				●	●	●	●		●	●	●		●
D	●	●	●	●			●	●				●	●	●							●

*1: Operation unit: 16-bit integers (SS, US) cannot be specified.
*2: Only 16-bit devices, 32-bit devices, and integer constants can be modified (real number constants, and character constants cannot be specified).
*3: Index register (I0 to IE)
*4: Only operation unit: signed integers (SS, SL) can be specified.
*5: Only operation unit: integers (US, SS, UL, SL) can be specified.
*6: Only operation unit: single-precision, floating-point real numbers (SF) can be specified.
*7: Only operation unit: double-precision, floating-point real numbers (DF) can be specified.

■ Outline of operation

- The output value, to be stored in the device address specified by [D], is controlled based on whether or not the input value specified by [S3] falls within the range bounded by the upper and lower limits set in [S1] and [S2].
- The output value [D] stores the following data.

When Lower limit [S1] > Input value [S3],	Input value [S3] - Lower limit [S1] → Output value [D]
When Upper limit [S2] < Input value [S3],	Upper limit [S3] - Lower limit [S2] → Output value [D]
When Lower limit [S1] ≤ Input value [S3] ≤ Upper limit [S2],	0 → Output value [D]

■ Process details

Example 1) Operation unit: 16 bits (SS)

[i]…SS

[S1]…DT0 [S2]…DT1 [S3]…DT2 [D]…DT10

DT0	K 100		DT0	K 100	
DT1	K 1000		DT1	K 1000	
DT2	K 500		DT2	K 500	
DT10	**K 100**	→	DT10	**K 0**	

Example 2) Operation unit: 32 bits (SL)

[i]…SL

[S1]…TS0 [S2]…TS1 [S3]…TS2 [D]…DT10

TS0	K 100		TS0	K 100	
TS1	K 1000		TS1	K 1000	
TS2	K 90		TS2	K 90	
DT10	**K 100**	→	DT10	**K -10**	

Example 3) Operation unit: Single-precision, floating-point real number (SF)

[i]…SF

[S1]…DT0 [S2]…DT4 [S3]…DT8 [D]…DT50

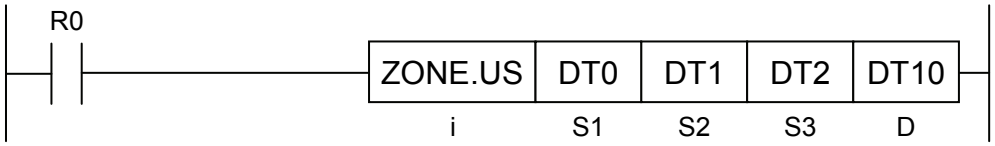
DT0・DT1	SF 1.00E＋02		DT0・DT1	SF 1.00E＋02	
DT4・DT5	SF 1.00E+03		DT4・DT5	SF 1.00E+03	
DT8・DT9	SF 3.00E＋03		DT8・DT9	SF 3.00E＋03	
DT50・DT51	**SF 1.00E＋02**	→	DT50・DT51	**SF 2.00E＋03**	

■ Flag operations

Name	Explanation
SR7	To be set in the case of out-of-range in indirect access (index modification, pointer access).
SR8 (ER)	To be set when [S1] > [S2].

ZONE (Zone Control)

■ Ladder diagram



■ Available operation units (●: Available)

Operation unit	bit	US	SS	UL	SL	SF	DF
i		●	●	●	●	●	●

■ List of operands

Operand	Explanation
S1	Device address that stores the negative bias value at input, or bias value data
S2	Device address that stores the positive bias value at input, or bias value data
S3	Device address where the input value is stored, or input value data
D	Output storage device address

■ Available devices (●: Available)

Operand	16-bit device											32-bit device *1			Integers			Real numbers		String	Index modifier *2
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX *3	K *4	U *5	H *6	SF *7	DF *8	" "	
S1	●	●	●	●			●	●				●	●	●	●	●	●	●	●		●
S2	●	●	●	●			●	●				●	●	●	●	●	●	●	●		●
S3	●	●	●	●			●	●				●	●	●	●	●	●	●	●		●
D	●	●	●	●			●	●				●	●	●							●

*1: Operation unit: 16-bit integers (SS, US) cannot be specified.
*2: Only 16-bit devices, 32-bit devices, and integer constants can be modified (real number constants, and character constants cannot be specified).
*3: Index register (I0 to IE)
*4: Only operation unit: signed integers (SS, SL) can be specified.
*5: Only operation unit: unsigned integers (US, UL) can be specified.
*6: Only operation unit: integers (US, SS, UL, SL) can be specified.
*7: Only operation unit: single-precision, floating-point real numbers (SF) can be specified.
*8: Only operation unit: double-precision, floating-point real numbers (DF) can be specified.

■ Outline of operation

- Add an bias value specified by [S1] or [S2] to the input value specified by [S3]. The resulting value is stored in the device address specified by [D].
- Output values are defined as follows:

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]

■ Process details

Example 1) Operation unit: 16 bits (US, SS)

[i]…US,SS

[S1]…DT0 [S2]…DT1 [S3]…DT2 [D]…DT10

DT0	K -100		DT0	K -100
DT1	K 1000		DT1	K 1000
DT2	K 500		DT2	K 500
DT10	**K 100**	→	DT10	**K 600**

Example 2) Operation unit: 32 bits (UL, SL)

[i]…UL,SL

[S1]…TS0 [S2]…TS1 [S3]…TS2 [D]…DT10

TS0	K -100		TS0	K -100
TS1	K 1000		TS1	K 1000
TS2	K -300		TS2	K -300
DT10	**K 100**	→	DT10	**K -400**

Example 3) Operation unit: Single-precision, floating-point real number (SF)

[i]…SF

[S1]…DT0 [S2]…DT4 [S3]…DT8 [D]…DT50

DT0・DT1	SF -1.00E+02		DT0・DT1	SF -1.00E+02
DT4・DT5	SF 1.00E+02		DT4・DT5	SF 1.00E+02
DT8・DT9	SF 0.00E+00		DT8・DT9	SF 0.00E+00
DT50・DT51	**SF 1.00E+02**	→	DT50・DT51	**SF 0.00E+00**

■ Flag operations

Name	Explanation
SR7 SR8 (ER)	To be set in the case of out-of-range in indirect access (index modification, pointer access).

FILTR (Time Constant Processing)

■ Ladder diagram

■ Available operation units (●: Available)

- No operation units

■ List of operands

Operand	Explanation
S1	Filtering target data (device address)
S2	Filtering targeted bit (device address or constant) (data available range: H0000 to HFFFF)
S3	Filtering time (device address or constant) (data available range: 0 to 30000, in ms)
D	Filtering result (device address)

■ Available devices (●: Available)

Operand	16-bit device											32-bit device			Integers			Real numbers		String	Index modifier *1
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX	K	U	H	SF	DF	" "	
S1	●	●	●	●			●	●													●
S2	●	●	●	●			●	●								●	●				●
S3	●	●	●	●			●	●								●	●				●
D	●	●	●	●			●	●													●

*1: Only 16-bit devices, 32-bit devices, and integer constants can be modified (real number constants, and character constants cannot be specified).

■ Outline of operation

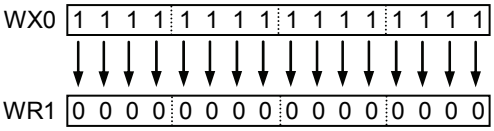
- Among data specified by [S1], bits specified by [S2] with the value 0 are directly output, and those with the value 1 are filtered and output.
- Filtering is carried out for the targeted bits within the time specified by [S3] (in 0 to 30000 ms). The result is output to the area specified by [D].
- When the execution condition is switched on, all input bits specified by [S1] are directly output without conditions.
- It is possible that a delay of up to one scan may be caused in the filtering time.

■ Process details

Among targeted data, specified bits with the value 0 are directly output, and those with the value 1 are filtered and output. Filtering is carried out within the specified time and the result is output.

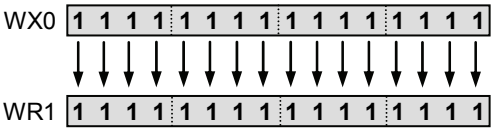
① Default conditions

Operand	Description	Device	Setting value
[S1]	Targeted data	WX0	H FFFF
[S2]	Targeted bit	DT0	H FF00
[S3]	Filtering time	DT1	K500
[D]	Processing result	WR1	H 0000



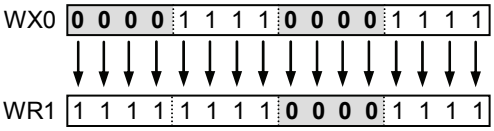
② Execution condition switched on (all input bits are directly output unconditionally)

Operand	Description	Device	Setting value
[S1]	Targeted data	WX0	H **FFFF**
[S2]	Targeted bit	DT0	H FF00
[S3]	Filtering time	DT1	K500
[D]	Processing result	WR1	H **FFFF**



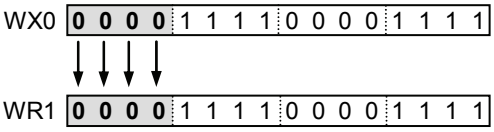
③ Filtering targeted data change (only untargeted bits are output)

Operand	Description	Device	Setting value
[S1]	Targeted data	WX0	H **0**F**0**F
[S2]	Targeted bit	DT0	H FF00
[S3]	Filtering time	DT1	K500
[D]	Processing result	WR1	H FF**0**F



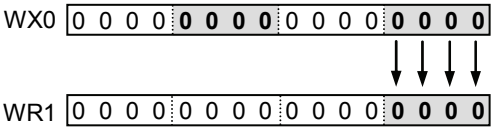
④ Filter processing time elapse (targeted bits are output)

Operand	Description	Device	Setting value
[S1]	Targeted data	WX0	H **0**F0F
[S2]	Targeted bit	DT0	H FF00
[S3]	Filtering time	DT1	K500
[D]	Processing result	WR1	H **0**F0F



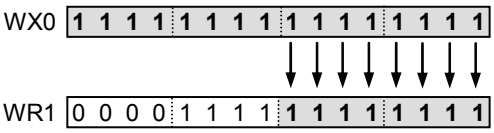
⑤: Filtering targeted data change (only untargeted bits are output)

Operand	Description	Device	Setting value
[S1]	Targeted data	WX0	H 0**00**0
[S2]	Targeted bit	DT0	H FF00
[S3]	Filtering time	DT1	K500
[D]	Processing result	WR1	H 0F0**0**



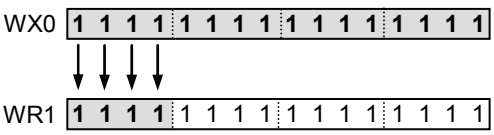
⑥ Filtering targeted data change before filter processing time elapse (only untargeted bits are output)

Operand	Description	Device	Setting value
[S1]	Targeted data	WX0	H **FFFF**
[S2]	Targeted bit	DT0	H FF00
[S3]	Filtering time	DT1	K500
[D]	Processing result	WR1	H 0F**FF**



⑦: Filter processing time elapse (targeted bits are output)

Operand	Description	Device	Setting value
[S1]	Targeted data	WX0	H **FFFF**
[S2]	Targeted bit	DT0	H FF00
[S3]	Filtering time	DT1	K500
[D]	Processing result	WR1	H **F**FFF

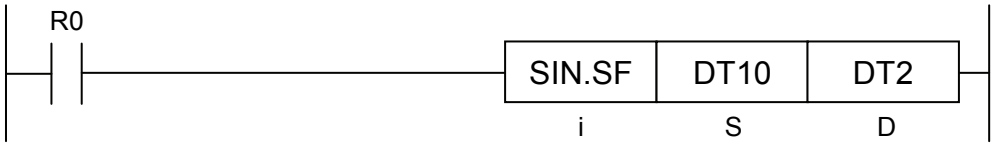


■ Flag operations

Name	Explanation
SR7	To be set in the case of out-of-range in indirect access (index modification).
SR8 (ER)	To be set when the filtering time [S3] is out of the range.

SIN (Sine Operation)

■ Ladder diagram



■ Available operation units (●: Available)

Operation unit	bit	US	SS	UL	SL	SF	DF
i						●	●

■ List of operands

Operand	Explanation
S	Angle data (device address or constant) (unit: radian)
D	Calculation result (device address)

■ Available devices (●: Available)

Operand	16-bit device											32-bit device			Integers			Real numbers		String	Index modifier
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX *2	K	U	H	SF *3	DF *4	" "	*1
S	●	●	●	●			●	●	●	●	●	●	●	●				●	●		●
D	●	●	●	●			●	●	●		●	●	●	●							●

*1: Only 16-bit devices, and 32-bit devices can be modified (real number constants cannot be specified).
*2: Index register (I0 to IE)
*3: Only operation unit: single-precision, floating-point real numbers (SF) can be specified.
*4: Only operation unit: double-precision, floating-point real numbers (DF) can be specified.

■ Outline of operation

- According to the operation unit [i], SIN for the angle data (unit: radian) stored in [S] is calculated.
- The calculation result is stored in the area starting with [D].
  SIN([S]) → [D]

■ Process details

Example) Operation unit: Single-precision, floating-point real number (SF) (Designate 30° radian for [S])

[i]…SF
[S]…DT10 [D]…DT2

	Angle	Value (radians)
DT10・DT11	30°	**SF 5.235988E−01**
DT12・DT13	60°	SF 1.047198E+00
DT14・DT15	90°	SF 1.570796E+00

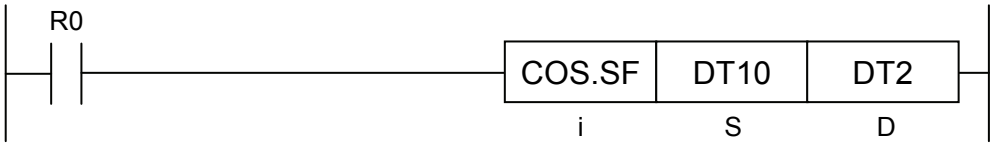
	Value
DT0・DT1	SF 0.000000E+00
DT2・DT3	**SF 5.000000E−01**
DT4・DT5	SF 0.000000E+00

■ Flag operations

Name	Explanation
SR7	To be set in the case of out-of-range in indirect access (index modification).
SR8 (ER)	To be set when a non-real number is specified for [S] (angle data).

COS (Cosine Operation)

■ Ladder diagram



■ Available operation units (●: Available)

Operation unit	bit	US	SS	UL	SL	SF	DF
i						●	●

■ List of operands

Operand	Explanation
S	Angle data (device address or constant) (unit: radian)
D	Calculation result (device address)

■ Available devices (●: Available)

Operand	16-bit device											32-bit device			Integers			Real numbers		String	Index modifier
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX *2	K	U	H	SF *3	DF *4	" "	*1
S	●	●	●	●			●	●	●	●	●	●	●	●				●	●		●
D	●	●	●	●			●	●	●		●	●	●	●							●

*1: Only 16-bit devices, and 32-bit devices can be modified (real number constants cannot be specified).
*2: Index register (I0 to IE)
*3: Only operation unit: single-precision, floating-point real numbers (SF) can be specified.
*4: Only operation unit: double-precision, floating-point real numbers (DF) can be specified.

■ Outline of operation

- According to the operation unit [i], COS for the angle data (unit: radian) stored in [S] is calculated.
- The calculation result is stored in the area starting with [D].
  COS([S]) → (D)

■ Process details

Example) Operation unit: Single-precision, floating-point real number (SF) (Designate 30° radian for [S])

[i]…SF
[S]…DT10 [D]…DT2

	Angle	Value (radians)
DT10・DT11	30°	**SF 5.235988E−01**
DT12・DT13	60°	SF 1.047198E+00
DT14・DT15	90°	SF 1.570796E+00

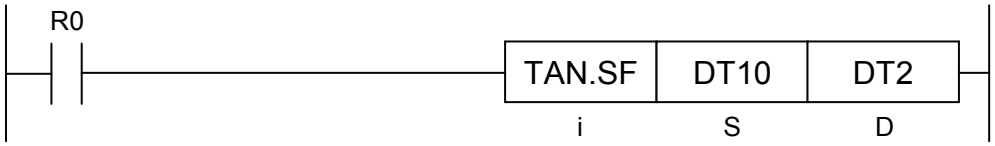
	Value
DT0・DT1	SF 0.000000E+00
DT2・DT3	**SF 8.660254E−01**
DT4・DT5	SF 0.000000E+00

■ Flag operations

Name	Explanation
SR7	To be set in the case of out-of-range in indirect access (index modification).
SR8 (ER)	To be set when a non-real number is specified for [S] (angle data).

TAN (Tangent Operation)

■ Ladder diagram



■ Available operation units (●: Available)

Operation unit	bit	US	SS	UL	SL	SF	DF
i						●	●

■ List of operands

Operand	Explanation
S	Angle data (device address or constant) (unit: radian)
D	Calculation result (device address)

■ Available devices (●: Available)

Operand	16-bit device											32-bit device			Integers			Real numbers		String	Index modifier
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX *2	K	U	H	SF *3	DF *4	" "	*1
S	●	●	●	●			●	●	●	●	●	●	●	●				●	●		●
D	●	●	●	●			●	●	●		●	●	●	●							●

*1: Only 16-bit devices, and 32-bit devices can be modified (real number constants cannot be specified).
*2: Index register (I0 to IE)
*3: Only operation unit: single-precision, floating-point real numbers (SF) can be specified.
*4: Only operation unit: double-precision, floating-point real numbers (DF) can be specified.

■ Outline of operation

- According to the operation unit [i], TAN for the angle data (unit: radian) stored in [S] is calculated.
- The calculation result is stored in the area starting with [D].
  TAN([S]) → (D)

■ Process details

Example) Operation unit: Single-precision, floating-point real number (SF) (Designate 30° radian for [S])

[i]…SF
[S]…DT10 [D]…DT2

	Angle	Value (radians)
DT10・DT11	30°	**SF 5.235988E−01**
DT12・DT13	60°	SF 1.047198E+00
DT14・DT15	90°	SF 1.570796E+00

→

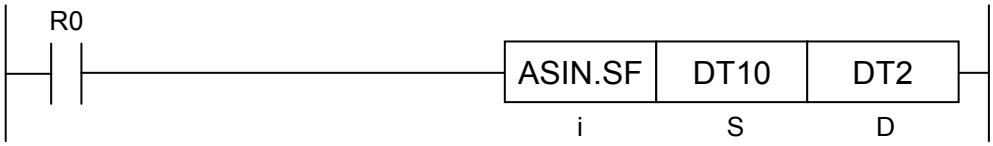
	Value
DT0・DT1	SF 0.000000E+00
DT2・DT3	**SF 5.773503E−01**
DT4・DT5	SF 0.000000E+00

■ Flag operations

Name	Explanation
SR7	To be set in the case of out-of-range in indirect access (index modification).
SR8 (ER)	To be set when a non-real number is specified for [S] (angle data).

ASIN (Arcsine Operation)

■ Ladder diagram



■ Available operation units (●: Available)

Operation unit	bit	US	SS	UL	SL	SF	DF
i						●	●

■ List of operands

Operand	Explanation
S	Angle data (device address or constant) (SIN value) (data available range: -1.0 to +1.0)
D	Calculation result (device address) (unit: radian)

■ Available devices (●: Available)

Operand	16-bit device											32-bit device			Integers			Real numbers		String	Index modifier
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX *2	K	U	H	SF *3	DF *4	" "	*1
S	●	●	●	●			●	●	●	●	●	●	●	●				●	●		●
D	●	●	●	●			●	●	●		●	●	●	●							●

*1: Only 16-bit devices, and 32-bit devices can be modified (real number constants cannot be specified).
*2: Index register (I0 to IE)
*3: Only operation unit: single-precision, floating-point real numbers (SF) can be specified.
*4: Only operation unit: double-precision, floating-point real numbers (DF) can be specified.

■ Outline of operation

- According to the operation unit [i], ASIN (arcsine) for the SIN value stored in [S] is calculated.
- The calculation result is stored in the area starting with [D].
  ASIN([S]) → (D)

■ Process details

Example) Operation unit: Single-precision, floating-point real number (SF) (Designate 15° SIN value for [S])

[i]…SF

[S]…DT10 [D]…DT2

	Angle	Value (radians)
DT10・DT11	15°	**SF2.588190E−01**
DT12・DT13	30°	SF1.047198E+00
DT14・DT15	45°	SF1.570796E+00

→

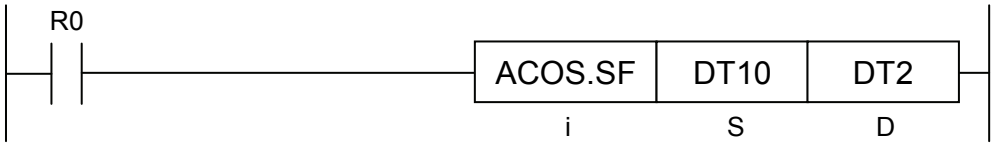
	Value
DT0・DT1	SF0.000000E+00
DT2・DT3	**SF2.617994E−01**
DT4・DT5	SF0.000000E+00

■ Flag operations

Name	Explanation
SR7	To be set in the case of out-of-range in indirect access (index modification).
SR8	To be set when a non-real number is specified for [S] (angle data).
(ER)	To be set when [S] (angle data) is out of the accessible range.

ACOS (Arccosine Operation)

■ Ladder diagram



■ Available operation units (●: Available)

Operation unit	bit	US	SS	UL	SL	SF	DF
i						●	●

■ List of operands

Operand	Explanation
S	Angle data (device address or constant) (COS value) (data available range: -1.0 to +1.0)
D	Calculation result (device address) (unit: radian)

■ Available devices (●: Available)

Operand	16-bit device											32-bit device			Integers			Real numbers		String	Index modifier
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX *2	K	U	H	SF *3	DF *4	" "	*1
S	●	●	●	●			●	●	●	●	●	●	●	●				●	●		●
D	●	●	●	●			●	●	●		●	●	●	●							●

*1: Only 16-bit devices, and 32-bit devices can be modified (real number constants cannot be specified).
*2: Index register (I0 to IE)
*3: Only operation unit: single-precision, floating-point real numbers (SF) can be specified.
*4: Only operation unit: double-precision, floating-point real numbers (DF) can be specified.

■ Outline of operation

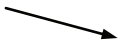
- According to the operation unit [i], ACOS (arccosine) for the COS value stored in [S] is calculated.
  ACOS([S]) → (D)

■ Process details

Example) Operation unit: Single-precision, floating-point real number (SF) (Designate 15° COS value for [S])

[i]…SF
[S]…DT10 [D]…DT2

	Angle	Value (radians)
DT10・DT11	15–	**SF9.659258E−01**
DT12・DT13	30–	SF1.047198E+00
DT14・DT15	45–	SF1.570796E+00



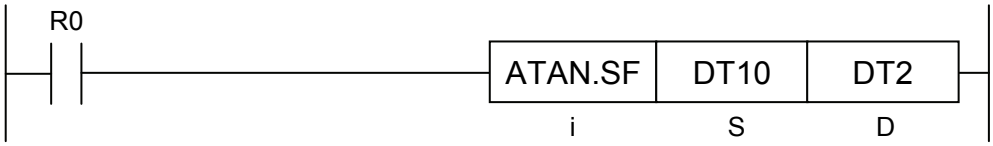
	Value
DT0・DT1	SF0.000000E+00
DT2・DT3	**SF2.617994E−01**
DT4・DT5	SF0.000000E+00

■ Flag operations

Name	Explanation
SR7	To be set in the case of out-of-range in indirect access (index modification).
SR8	To be set when a non-real number is specified for [S] (angle data).
(ER)	To be set when [S] (angle data) is out of the accessible range.

ATAN (Arctangent Operation)

■ Ladder diagram



■ Available operation units (●: Available)

Operation unit	bit	US	SS	UL	SL	SF	DF
i						●	●

■ List of operands

Operand	Explanation
S	Angle data (device address or constant) (TAN value)
D	Calculation result (device address) (unit: radian)

■ Available devices (●: Available)

Operand	16-bit device											32-bit device			Integers			Real numbers		String	Index modifier *1
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX *2	K	U	H	SF *3	DF *4	" "	
S	●	●	●	●			●	●	●	●	●	●	●	●				●	●		●
D	●	●	●	●			●	●	●		●	●	●	●							●

*1: Only 16-bit devices, and 32-bit devices can be modified (real number constants cannot be specified).
*2: Index register (I0 to IE)
*3: Only operation unit: single-precision, floating-point real numbers (SF) can be specified.
*4: Only operation unit: double-precision, floating-point real numbers (DF) can be specified.

■ Outline of operation

- According to the operation unit [i], ATAN (arctangent) for the TAN value stored in [S] is calculated.
- The calculation result is stored in the area starting with [D].
  ATAN([S]) → (D)

■ Process details

Example) Operation unit: Single-precision, floating-point real number (SF) (Designate 15° TAN value for [S])

[i]…SF
[S]…DT10 [D]…DT2

	Angle	Value (radians)
DT10・DT11	15°	**SF 2.679392E−01**
DT12・DT13	30°	SF 1.047198E+00
DT14・DT15	45°	SF 1.570796E+00

→

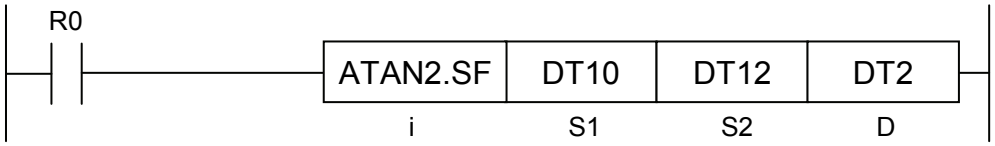
	Value
DT0・DT1	SF 0.000000E+00
DT2・DT3	**SF 2.617994E−01**
DT4・DT5	SF 0.000000E+00

■ Flag operations

Name	Explanation
SR7	To be set in the case of out-of-range in indirect access (index modification).
SR8 (ER)	To be set when a non-real number is specified for [S] (angle data).

ATAN2 (Conversion: Coordinate Data → Angle Radian)

■ Ladder diagram



■ Available operation units (●: Available)

Operation unit	bit	US	SS	UL	SL	SF	DF
i						●	●

■ List of operands

Operand	Explanation
S1	Dividend of angle data (device address or constant) (Y coordinate)
S2	Angle data divisor (device address or constant) (X coordinate)
D	Calculation result (device address) (unit: radian)

■ Available devices (●: Available)

Operand	16-bit device											32-bit device			Integers			Real numbers		String	Index modifier *1
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX *2	K	U	H	SF *3	DF *4	" "	
S1	●	●	●	●			●	●	●	●	●	●	●	●				●	●		●
S2	●	●	●	●			●	●	●	●	●	●	●	●				●	●		●
D	●	●	●	●			●	●	●		●	●	●	●							●

*1: Only 16-bit devices, and 32-bit devices can be modified (real number constants cannot be specified).
*2: Index register (I0 to IE)
*3: Only operation unit: single-precision, floating-point real numbers (SF) can be specified.
*4: Only operation unit: double-precision, floating-point real numbers (DF) can be specified.

■ Outline of operation

- According to the operation unit [i], ATAN (unit: radian) is calculated from the Y coordinate specified by [S1] and the X coordinate specified by [S2].
- The calculation result is stored in the area starting with [D].
  ATAN2([S1], [S2]) → [D]

■ Process details

Example) Operation unit: Single-precision, floating-point real number (SF) (Designate 1.0 for [S1] (Y coordinate) and [S2] (X coordinate))

[i]…SF
[S1]…DT10 [S2]…DT12 [D]…DT2

	Value
DT10・DT11	**SF 1.000000E+00**
DT12・DT13	**SF 1.000000E+00**
DT14・DT15	SF 0.000000E+00

→

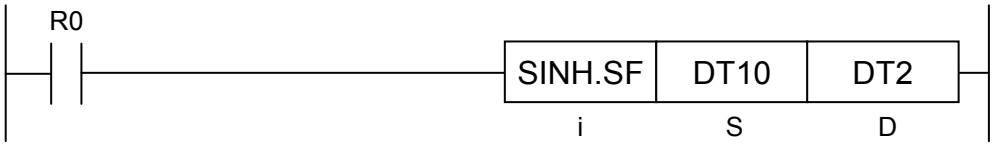
	Value
DT0・DT1	SF 0.000000E+00
DT2・DT3	**SF 7.853982E−01**
DT4・DT5	SF 0.000000E+00

■ Flag operations

Name	Explanation
SR7	To be set in the case of out-of-range in indirect access (index modification).
SR8	To be set when a non-real number is specified for [S1] (Y coordinate) or [S2] (X coordinate).
(ER)	To be set when 0.0 is specified for [S1] (Y coordinate) and 0.0 for [S2] (X coordinate).

SINH (Hyperbolic Sine Operation)

■ Ladder diagram



■ Available operation units (●: Available)

Operation unit	bit	US	SS	UL	SL	SF	DF
i						●	●

■ List of operands

Operand	Explanation
S	Angle data (device address or constant) (unit: radian)
D	Calculation result (device address)

■ Available devices (●: Available)

Operand	16-bit device											32-bit device			Integers			Real numbers		String	Index modifier
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX *2	K	U	H	SF *3	DF *4	" "	*1
S	●	●	●	●			●	●	●	●	●	●	●	●				●	●		●
D	●	●	●	●			●	●	●		●	●	●	●							●

*1: Only 16-bit devices, and 32-bit devices can be modified (real number constants cannot be specified).
*2: Index register (I0 to IE)
*3: Only operation unit: single-precision, floating-point real numbers (SF) can be specified.
*4: Only operation unit: double-precision, floating-point real numbers (DF) can be specified.

■ Outline of operation

- According to the operation unit [i], SINH (hyperbolic sine) for the angle data (unit: radian) stored in [S] is calculated.
- The calculation result is stored in the area starting with [D].
  SINH([S]) → [D]

■ Process details

Example) Operation unit: Single-precision, floating-point real number (SF) (Designate 30° radian for [S])

[i]…SF
[S]…DT10 [D]…DT2

	Angle	Value (radians)
DT10・DT11	30°	**SF 5.235988E−01**
DT12・DT13	60°	SF 1.047198E+00
DT14・DT15	90°	SF 1.570796E+00

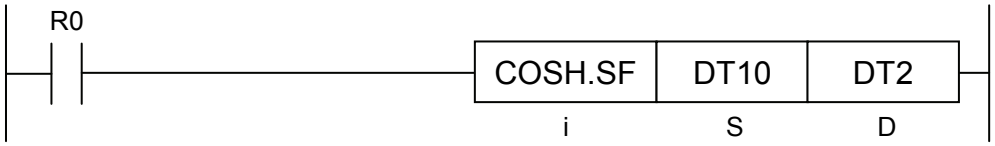
	Value
DT0・DT1	SF 0.000000E+00
DT2・DT3	**SF 5.478535E−01**
DT4・DT5	SF 0.000000E+00

■ Flag operations

Name	Explanation
SR7	To be set in the case of out-of-range in indirect access (index modification).
SR8 (ER)	To be set when a non-real number is specified for [S] (angle data).

COSH (Hyperbolic Cosine Operation)

■ Ladder diagram



■ Available operation units (●: Available)

Operation unit	bit	US	SS	UL	SL	SF	DF
i						●	●

■ List of operands

Operand	Explanation
S	Angle data (device address or constant) (unit: radian)
D	Calculation result (device address)

■ Available devices (●: Available)

Operand	16-bit device											32-bit device			Integers			Real numbers		String	Index modifier
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX *2	K	U	H	SF *3	DF *4	" "	*1
S	●	●	●	●			●	●	●	●	●	●	●	●				●	●		●
D	●	●	●	●			●	●	●		●	●	●	●							●

*1: Only 16-bit devices, and 32-bit devices can be modified (real number constants cannot be specified).
*2: Index register (I0 to IE)
*3: Only operation unit: single-precision, floating-point real numbers (SF) can be specified.
*4: Only operation unit: double-precision, floating-point real numbers (DF) can be specified.

■ Outline of operation

- According to the operation unit [i], COSH (hyperbolic cosine) for the angle data (unit: radian) stored in [S] is calculated.
- The calculation result is stored in the area starting with [D].
  COSH([S]) → [D]

■ Process details

Example) Operation unit: Single-precision, floating-point real number (SF) (Designate 30° radian for [S])

[i]…SF
[S]…DT10 [D]…DT2

	Angle	Value (radians)
DT10・DT11	30°	**SF 5.235988E−01**
DT12・DT13	60°	SF 1.047198E+00
DT14・DT15	90°	SF 1.570796E+00

→

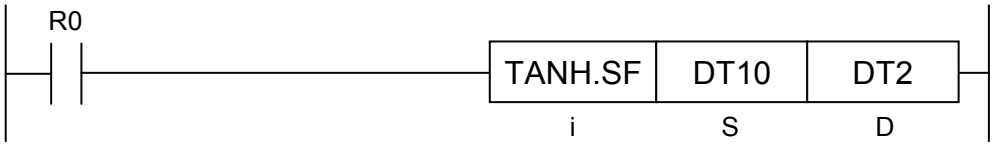
	Value
DT0・DT1	SF 0.000000E+00
DT2・DT3	**SF 1.140238E+00**
DT4・DT5	SF 0.000000E+00

■ Flag operations

Name	Explanation
SR7	To be set in the case of out-of-range in indirect access (index modification).
SR8 (ER)	To be set when a non-real number is specified for [S] (angle data).

TANH (Hyperbolic Tangent Operation)

■ Ladder diagram



■ Available operation units (●: Available)

Operation unit	bit	US	SS	UL	SL	SF	DF
i						●	●

■ List of operands

Operand	Explanation
S	Angle data (device address or constant) (unit: radian)
D	Calculation result (device address)

■ Available devices (●: Available)

Operand	16-bit device											32-bit device			Integers			Real numbers		String	Index modifier
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX *2	K	U	H	SF *3	DF *4	" "	*1
S	●	●	●	●			●	●	●	●	●	●	●	●				●	●		●
D	●	●	●	●			●	●	●		●	●	●	●							●

*1: Only 16-bit devices, and 32-bit devices can be modified (real number constants cannot be specified).
*2: Index register (I0 to IE)
*3: Only operation unit: single-precision, floating-point real numbers (SF) can be specified.
*4: Only operation unit: double-precision, floating-point real numbers (DF) can be specified.

■ Outline of operation

- According to the operation unit [i], TANH (hyperbolic tangent) for the angle data (unit: radian) stored in [S] is calculated.
- The calculation result is stored in the area starting with [D].
  TANH([S]) → [D]

■ Process details

Example) Operation unit: Single-precision, floating-point real number (SF) (Designate 30° radian for [S])
[i]…SF
[S]…DT10 [D]…DT2

	Angle	Value (radians)
DT10・DT11	30°	**SF 5.235988E−01**
DT12・DT13	60°	SF 1.047198E+00
DT14・DT15	90°	SF 1.570796E+00

→

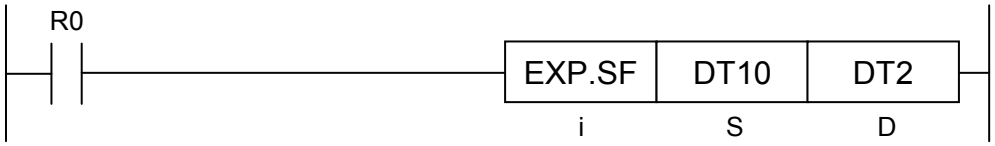
	Value
DT0・DT1	SF 0.000000E+00
DT2・DT3	**SF 4.804728E−01**
DT4・DT5	SF 0.000000E+00

■ Flag operations

Name	Explanation
SR7	To be set in the case of out-of-range in indirect access (index modification).
SR8 (ER)	To be set when a non-real number is specified for [S] (angle data).

EXP (Exponential Operation)

■ Ladder diagram



■ Available operation units (●: Available)

Operation unit	bit	US	SS	UL	SL	SF	DF
i						●	●

■ List of operands

Operand	Explanation
S	Calculation target data (device address or constant) (real number value)
D	Calculation result (device address)

■ Available devices (●: Available)

Operand	16-bit device											32-bit device			Integers			Real numbers		String	Index modifier *1
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX *2	K	U	H	SF *3	DF *4	" "	
S	●	●	●	●			●	●	●	●	●	●	●	●				●	●		●
D	●	●	●	●			●	●	●		●	●	●	●							●

*1: Only 16-bit devices, and 32-bit devices can be modified (real number constants cannot be specified).
*2: Index register (I0 to IE)
*3: Only operation unit: single-precision, floating-point real numbers (SF) can be specified.
*4: Only operation unit: double-precision, floating-point real numbers (DF) can be specified.

■ Outline of operation

- According to the operation unit [i], EXP (tangent) for the real number value stored in the area starting with [S] is calculated.
- The calculation result is stored in the area starting with [D].
  EXP([S]) → [D]

■ Process details

Example) Operation unit: Single-precision, floating-point real number (SF)

[i]…SF
[S]…DT10 [D]…DT2

	Value (radians)
DT10・DT11	**SF 3.000000E+00**
DT12・DT13	SF 4.000000E+00
DT14・DT15	SF 5.000000E+00

→

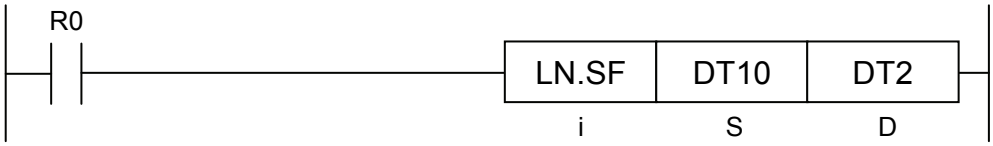
	Value
DT0・DT1	SF 0.000000E+00
DT2・DT3	**SF 2.008554E+01**
DT4・DT5	SF 0.000000E+00

■ Flag operations

Name	Explanation
SR7	To be set in the case of out-of-range in indirect access (index modification).
SR8 (ER)	To be set when a non-real number is specified for [S] (calculation target data).

LN (Natural Logarithmic Operation)

■ Ladder diagram



■ Available operation units (●: Available)

Operation unit	bit	US	SS	UL	SL	SF	DF
i						●	●

■ List of operands

Operand	Explanation
S	Calculation target data (device address or constant) (real number value)
D	Calculation result (device address)

■ Available devices (●: Available)

Operand	16-bit device											32-bit device			Integers			Real numbers		String	Index modifier
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX *2	K	U	H	SF *3	DF *4	" "	*1
S	●	●	●	●			●	●	●	●	●	●	●	●				●	●		●
D	●	●	●	●			●	●	●		●	●	●	●							●

*1: Only 16-bit devices, and 32-bit devices can be modified (real number constants cannot be specified).
*2: Index register (I0 to IE)
*3: Only operation unit: single-precision, floating-point real numbers (SF) can be specified.
*4: Only operation unit: double-precision, floating-point real numbers (DF) can be specified.

■ Outline of operation

- According to the operation unit [i], LN (natural logarithm) for the real number value stored in the area starting with [S] is calculated.
- The calculation result is stored in the area starting with [D].
  LN([S]) → [D]

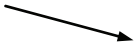
■ Process details

- LN (calculation target data) is calculated, and set for the calculation result.

Example) Operation unit: Single-precision, floating-point real number (SF)

[i]…SF
[S]…DT10 [D]…DT2

	Value (radians)
DT10・DT11	**SF 3.000000E+00**
DT12・DT13	SF 4.000000E+00
DT14・DT15	SF 5.000000E+00



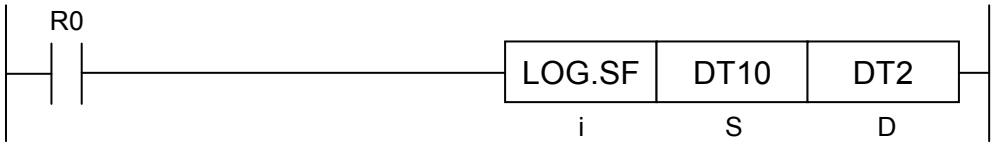
	Value
DT0・DT1	SF 0.000000E+00
DT2・DT3	**SF 1.098612E+00**
DT4・DT5	SF 0.000000E+00

■ Flag operations

Name	Explanation
SR7	To be set in the case of out-of-range in indirect access (index modification).
SR8	To be set when a non-real number is specified for [S] (calculation target data).
(ER)	To be set when a value ≤ 0.0 is specified for [S] (calculation target data).

LOG (Common Logarithmic Operation)

■ Ladder diagram



■ Available operation units (●: Available)

Operation unit	bit	US	SS	UL	SL	SF	DF
i						●	●

■ List of operands

Operand	Explanation
S	Calculation target data (device address or constant) (real number value)
D	Calculation result (device address)

■ Available devices (●: Available)

Operand	16-bit device											32-bit device			Integers			Real numbers		String	Index modifier
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX *2	K	U	H	SF *3	DF *4	" "	*1
S	●	●	●	●			●	●	●	●	●	●	●	●				●	●		●
D	●	●	●	●			●	●	●		●	●	●	●							●

*1: Only 16-bit devices, and 32-bit devices can be modified (real number constants cannot be specified).
*2: Index register (I0 to IE)
*3: Only operation unit: single-precision, floating-point real numbers (SF) can be specified.
*4: Only operation unit: double-precision, floating-point real numbers (DF) can be specified.

■ Outline of operation

- According to the operation unit [i], LOG (common logarithm) for the real number value stored in the area starting with [S] is calculated.
- The calculation result is stored in the area starting with [D].
  LOG([S]) → [D]

■ Process details

Example) Operation unit: Single-precision, floating-point real number (SF)

[i]…SF
[S]…DT10 [D]…DT2

	Value (radians)
DT10・DT11	**SF 3.000000E+00**
DT12・DT13	SF 4.000000E+00
DT14・DT15	SF 5.000000E+00

→

	Value
DT0・DT1	SF 0.000000E+00
DT2・DT3	**SF 4.771213E−01**
DT4・DT5	SF 0.000000E+00

■ Flag operations

Name	Explanation
SR7	To be set in the case of out-of-range in indirect access (index modification).
SR8	To be set when a non-real number is specified for [S] (calculation target data).
(ER)	To be set when a value ≤ 0.0 is specified for [S] (calculation target data).

PWR (Power Operation)

■ Ladder diagram

■ Available operation units (●: Available)

Operation unit	bit	US	SS	UL	SL	SF	DF
i						●	●

■ List of operands

Operand	Explanation
S1	Data to be powered (device address or constant) (real number value)
S2	Powering data (device address or constant) (real number value)
D	Calculation result (device address)

■ Available devices (●: Available)

Operand	16-bit device											32-bit device			Integers			Real numbers		String	Index modifier
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX *2	K	U	H	SF *3	DF *4	" "	*1
S1	●	●	●	●			●	●	●	●	●	●	●	●				●	●		●
S2	●	●	●	●			●	●	●	●	●	●	●	●				●	●		●
D	●	●	●	●			●	●	●		●	●	●	●							●

*1: Only 16-bit devices, and 32-bit devices can be modified (real number constants cannot be specified).
*2: Index register (I0 to IE)
*3: Only operation unit: single-precision, floating-point real numbers (SF) can be specified.
*4: Only operation unit: double-precision, floating-point real numbers (DF) can be specified.

■ Outline of operation

- According to the operation unit [i], the real number value stored in the area starting with [S1] is powered by the real number value stored in the area starting with [S2].
- The calculation result is stored in the area starting with [D].
  [S1] ^ [S2] → [D]

■ Process details

Example) Operation unit: Single-precision, floating-point real number (SF)

[i]…SF
[S1]…DT10 [S2]…DT12 [D]…DT2

	Value (radians)
DT10・DT11	**SF 3.000000E+00**
DT12・DT13	**SF 4.000000E+00**
DT14・DT15	SF 5.000000E+00

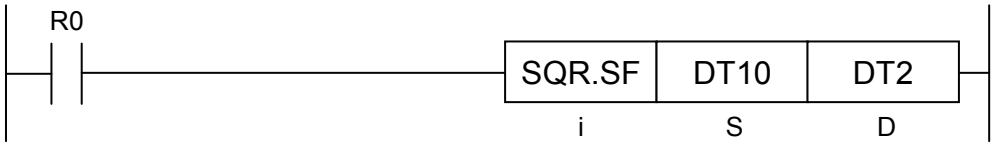
	Value
DT0・DT1	SF 0.000000E+00
DT2・DT3	**SF 8.100000E+01**
DT4・DT5	SF 0.000000E+00

■ Flag operations

Name	Explanation
SR7 SR8 (ER)	To be set in the case of out-of-range in indirect access (index modification).
	To be set when a non-real number is specified for [S1] (data to be powered) or [S2] (powering data).
	To be set when 0.0 is specified for [S1] (data to be powered) and a value other than 0.0 for [S2] (powering data).
	To be set when a negative value is specified for [S1] (data to be powered) and a non-integer value for [S2] (powering data).

SQR (Square Root Operation)

■ Ladder diagram



■ Available operation units (●: Available)

Operation unit	bit	US	SS	UL	SL	SF	DF
i						●	●

■ List of operands

Operand	Explanation
S	Calculation target data (device address or constant) (real number value)
D	Calculation result (device address)

■ Available devices (●: Available)

Operand	16-bit device											32-bit device			Integers			Real numbers		String	Index modifier
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX *2	K	U	H	SF *3	DF *4	" "	*1
S	●	●	●	●			●	●	●	●	●	●	●	●				●	●		●
D	●	●	●	●			●	●	●		●	●	●	●							●

*1: Only 16-bit devices, and 32-bit devices can be modified (real number constants cannot be specified).
*2: Index register (I0 to IE)
*3: Only operation unit: single-precision, floating-point real numbers (SF) can be specified.
*4: Only operation unit: double-precision, floating-point real numbers (DF) can be specified.

■ Outline of operation

- According to the operation unit [i], square root for the real number value stored in the area starting with [S] is calculated.
- The calculation result is stored in the area starting with [D].
  SQR([S]) → [D]

■ Process details

Example) Operation unit: Single-precision, floating-point real number (SF)

[i]…SF
[S]…DT10 [D]…DT2

	Value (radians)
DT10・DT11	**SF 3.000000E+00**
DT12・DT13	SF 4.000000E+00
DT14・DT15	SF 5.000000E+00

→

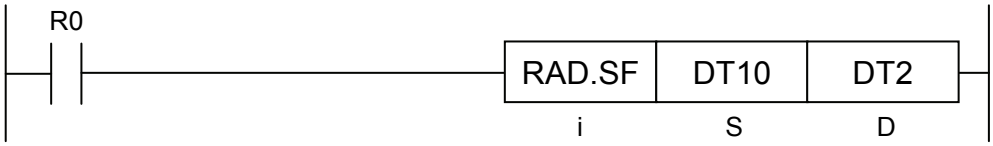
	Value
DT0・DT1	SF 0.000000E+00
DT2・DT3	**SF 1.732051E+00**
DT4・DT5	SF 0.000000E+00

■ Flag operations

Name	Explanation
SR7	To be set in the case of out-of-range in indirect access (index modification).
SR8	To be set when a non-real number is specified for [S] (calculation target data).
(ER)	To be set when a negative value is specified for [S] (calculation target data).

RAD (Conversion: Degrees → Radian)

■ Ladder diagram



■ Available operation units (●: Available)

Operation unit	bit	US	SS	UL	SL	SF	DF
i						●	●

■ List of operands

Operand	Explanation
S	Angle data (device address or constant) (unit: degrees)
D	Angle data (device address) (unit: radian)

■ Available devices (●: Available)

Operand	16-bit device											32-bit device			Integers			Real numbers		String	Index modifier
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX *2	K	U	H	SF *3	DF *4	" "	*1
S	●	●	●	●			●	●	●	●	●	●	●	●				●	●		●
D	●	●	●	●			●	●	●		●	●	●	●							●

*1: Only 16-bit devices, and 32-bit devices can be modified (real number constants cannot be specified).
*2: Index register (I0 to IE)
*3: Only operation unit: single-precision, floating-point real numbers (SF) can be specified.
*4: Only operation unit: double-precision, floating-point real numbers (DF) can be specified.

■ Outline of operation

- According to the operation unit [i], angle data (unit: degrees), stored in the area starting with [S], are converted into angle data (unit: radian).
- The calculation result is stored in the area starting with [D].
  [S] × (π / 180) → [D]

■ Process details

Example) Operation unit: Single-precision, floating-point real number (SF) (Designate 30 (degrees) for [S])
[i]…SF
[S]…DT10 [D]…DT2

	Value (degrees)
DT10・DT11	**SF 3.000000E+01**
DT12・DT13	SF 6.000000E＋01
DT14・DT15	SF 9.000000E＋01

→

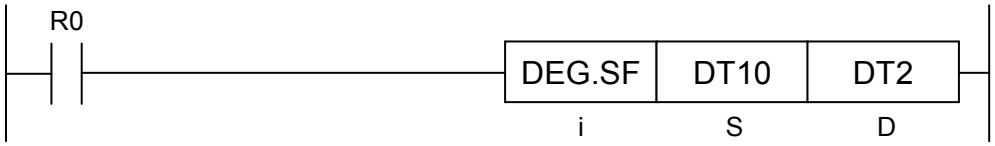
	Value (radians)
DT0・DT1	SF 0.000000E＋00
DT2・DT3	**SF 5.235988E－01**
DT4・DT5	SF 0.000000E＋00

■ Flag operations

Name	Explanation
SR7	To be set in the case of out-of-range in indirect access (index modification).
SR8 (ER)	To be set when a non-real number is specified for [S] (angle data).

DEG (Conversion: Radian → Degrees)

■ Ladder diagram



■ Available operation units (●: Available)

Operation unit	bit	US	SS	UL	SL	SF	DF
i						●	●

■ List of operands

Operand	Explanation
S	Angle data (device address or constant) (unit: radian)
D	Angle data (device address) (unit: degrees)

■ Available devices (●: Available)

Operand	16-bit device											32-bit device			Integers			Real numbers		String	Index modifier
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX *2	K	U	H	SF *3	DF *4	" "	*1
S	●	●	●	●			●	●	●	●	●	●	●	●				●	●		●
D	●	●	●	●			●	●	●		●	●	●	●							●

*1: Only 16-bit devices, and 32-bit devices can be modified (real number constants cannot be specified).
*2: Index register (I0 to IE)
*3: Only operation unit: single-precision, floating-point real numbers (SF) can be specified.
*4: Only operation unit: double-precision, floating-point real numbers (DF) can be specified.

■ Outline of operation

- According to the operation unit [i], angle data (unit: radian), stored in the area starting with [S], are converted into angle data (unit: degrees).
- The calculation result is stored in the area starting with [D].
  [S] × (180 / π) → [D]

■ Process details

Example) Operation unit: Single-precision, floating-point real number (SF) (Designate 30° radian for [S])

[i]…SF
[S]…DT10 [D]…DT2

	Angle	Value (radians)
DT10・DT11	30°	**SF 5.235988E−01**
DT12・DT13	60°	SF 1.047198E+00
DT14・DT15	90°	SF 1.570796E+00

→

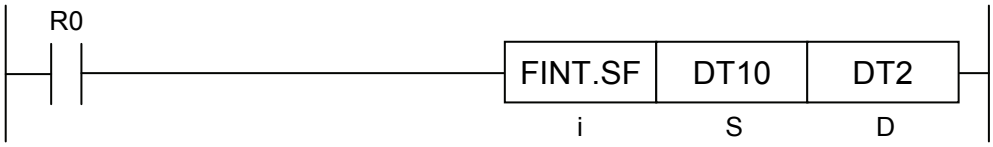
	Value (degrees)
DT0・DT1	SF 0.000000E+00
DT2・DT3	**SF 3.000000E+01**
DT4・DT5	SF 0.000000E+00

■ Flag operations

Name	Explanation
SR7	To be set in the case of out-of-range in indirect access (index modification).
SR8 (ER)	To be set when a non-real number is specified for [S] (angle data).

FINT (Floating Point Real Number Data - Rounding the First Decimal Point Down)

■ Ladder diagram



■ Available operation units (●: Available)

Operation unit	bit	US	SS	UL	SL	SF	DF
i						●	●

■ List of operands

Operand	Explanation
S	Calculation target data (device address or constant)
D	Calculation result (device address)

■ Available devices (●: Available)

Operand	16-bit device											32-bit device			Integers			Real numbers		String	Index modifier *1
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX *2	K	U	H	SF *3	DF *4	" "	
S	●	●	●	●			●	●	●	●	●	●	●	●				●	●		●
D	●	●	●	●			●	●	●		●	●	●	●							●

*1: Only 16-bit devices, and 32-bit devices can be modified (real number constants cannot be specified).
*2: Index register (I0 to IE)
*3: Only operation unit: single-precision, floating-point real numbers (SF) can be specified.
*4: Only operation unit: double-precision, floating-point real numbers (DF) can be specified.

■ Outline of operation

- According to the operation unit [i], the real number value stored in the area starting with [S] is processed, rounding the first decimal point down.
- The calculation result is stored in the area starting with [D].

■ Process details

Example 1) Operation unit: Single-precision, floating-point real number (SF) (positive real number)

[i]…SF
[S]…DT10 [D]…DT2

DT10・DT11	**SF 1.234560E+02**	DT0・DT1	SF 0.000000E+00
DT12・DT13	SF 3.456780E+02	DT2・DT3	**SF 1.230000E+02**
DT14・DT15	SF 5.678900E+02	DT4・DT5	SF 0.000000E+00

Example 2) Operation unit: Single-precision, floating-point real number (SF) (negative value)

[i]…SF
[S]…DT10 [D]…DT2

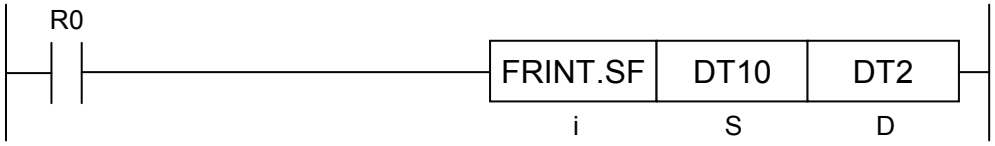
DT10・DT11	**SF -1.234560E+02**	DT0・DT1	SF 0.000000E+00
DT12・DT13	SF -3.456780E+02	DT2・DT3	**SF -1.230000E+02**
DT14・DT15	SF -5.678900E+02	DT4・DT5	SF 0.000000E+00

■ Flag operations

Name	Explanation
SR7	To be set in the case of out-of-range in indirect access (index modification).
SR8 (ER)	To be set when a non-real number is specified for [S].

FRINT (Floating Point Real Number Data - Rounding the First Decimal Point Off)

■ Ladder diagram



■ Available operation units (●: Available)

Operation unit	bit	US	SS	UL	SL	SF	DF
i						●	●

■ List of operands

Operand	Explanation
S	Calculation target data (device address or constant)
D	Calculation result (device address)

■ Available devices (●: Available)

Operand	16-bit device											32-bit device			Integers			Real numbers		String	Index modifier *1
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX *2	K	U	H	SF *3	DF *4	" "	
S	●	●	●	●			●	●	●	●	●	●	●	●				●	●		●
D	●	●	●	●			●	●	●		●	●	●	●							●

*1: Only 16-bit devices, and 32-bit devices can be modified (real number constants cannot be specified).
*2: Index register (I0 to IE)
*3: Only operation unit: single-precision, floating-point real numbers (SF) can be specified.
*4: Only operation unit: double-precision, floating-point real numbers (DF) can be specified.

■ Outline of operation

- According to the operation unit [i], the real number value stored in the area starting with [S] is processed, rounding the first decimal point off.
- The calculation result is stored in the area starting with [D].

■ Process details

Example 1) Operation unit: Single-precision, floating-point real number (SF) (positive real number)

[i]…SF
[S]…DT10 [D]…DT2

DT10・DT11	**SF 1.234560E+02**	DT0・DT1	SF 0.000000E+00
DT12・DT13	SF 3.456780E+02	DT2・DT3	**SF 1.230000E+02**
DT14・DT15	SF 5.678900E+02	DT4・DT5	SF 0.000000E+00

Example 2) Operation unit: Single-precision, floating-point real number (SF) (negative value)

[i]…SF
[S]…DT10 [D]…DT2

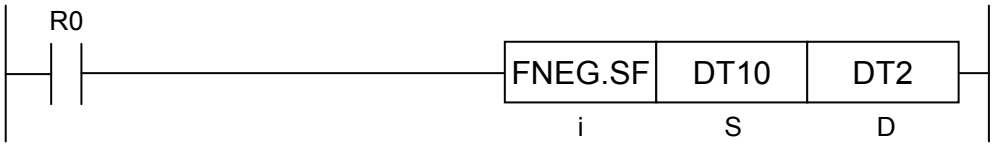
DT10・DT11	**SF -1.234560E+02**	DT0・DT1	SF 0.000000E+00
DT12・DT13	SF -3.456780E+02	DT2・DT3	**SF -1.230000E+02**
DT14・DT15	SF -5.678900E+02	DT4・DT5	SF 0.000000E+00

■ Flag operations

Name	Explanation
SR7	To be set in the case of out-of-range in indirect access (index modification).
SR8 (ER)	To be set when a non-real number is specified for [S].

FNEG (Floating Point Real Number Data - Sign Changes (Negative/Positive Conversion)

■ Ladder diagram



■ Available operation units (●: Available)

Operation unit	bit	US	SS	UL	SL	SF	DF
i						●	●

■ List of operands

Operand	Explanation
S	Calculation target data (device address or constant)
D	Calculation result (device address)

■ Available devices (●: Available)

Operand	16-bit device											32-bit device			Integers			Real numbers		String	Index modifier *1
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX *2	K	U	H	SF *3	DF *4	" "	
S	●	●	●	●			●	●	●	●	●	●	●	●				●	●		●
D	●	●	●	●			●	●	●		●	●	●	●							●

*1: Only 16-bit devices, and 32-bit devices can be modified (real number constants cannot be specified).
*2: Index register (I0 to IE)
*3: Only operation unit: single-precision, floating-point real numbers (SF) can be specified.
*4: Only operation unit: double-precision, floating-point real numbers (DF) can be specified.

■ Outline of operation

- According to the operation unit [i], the sign (negative or positive) of the real number value stored in the area starting with [S] is inverted.
- The calculation result is stored in the area starting with [D].

■ Process details

Example 1) Operation unit: Single-precision, floating-point real number (SF) (positive real number)

[i]…SF
[S]…DT10 [D]…DT2

DT10・DT11	**SF 1.234560E+02**	DT0・DT1	SF 0.000000E+00
DT12・DT13	SF 3.456780E+02	DT2・DT3	**SF -1.234560E+02**
DT14・DT15	SF 5.678900E+02	DT4・DT5	SF 0.000000E+00

Example 2) Operation unit: Single-precision, floating-point real number (SF) (negative value)

[i]…SF
[S]…DT10 [D]…DT2

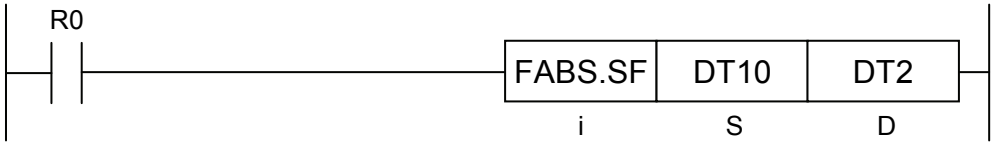
DT10・DT11	**SF -1.234560E+02**	DT0・DT1	SF 0.000000E+00
DT12・DT13	SF -3.456780E+02	DT2・DT3	**SF 1.234560E+02**
DT14・DT15	SF -5.678900E+02	DT4・DT5	SF 0.000000E+00

■ Flag operations

Name	Explanation
SR7	To be set in the case of out-of-range in indirect access (index modification).
SR8 (ER)	To be set when a non-real number is specified for [S].

FABS (Floating Point Real Number Data - Absolute Value)

■ Ladder diagram



■ Available operation units (●: Available)

Operation unit	bit	US	SS	UL	SL	SF	DF
i						●	●

■ List of operands

Operand	Explanation
S	Calculation target data (device address or constant)
D	Calculation result (device address)

■ Available devices (●: Available)

Operand	16-bit device											32-bit device			Integers			Real numbers		String	Index modifier *1
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX *2	K	U	H	SF *3	DF *4	" "	
S	●	●	●	●			●	●	●	●	●	●	●	●				●	●		●
D	●	●	●	●			●	●	●		●	●	●	●							●

*1: Only 16-bit devices, and 32-bit devices can be modified (real number constants cannot be specified).
*2: Index register (I0 to IE)
*3: Only operation unit: single-precision, floating-point real numbers (SF) can be specified.
*4: Only operation unit: double-precision, floating-point real numbers (DF) can be specified.

■ Outline of operation

- According to the operation unit [i], the absolute value of the real number value stored in the area starting with [S] is calculated.
- The calculation result is stored in the area starting with [D].

■ Process details

Example 1) Operation unit: Single-precision, floating-point real number (SF) (positive real number)

[i]…SF
[S]…DT10 [D]…DT2

DT10・DT11	**SF 1.234560E+02**	DT0・DT1	SF 0.000000E+00
DT12・DT13	SF 3.456780E+02	DT2・DT3	**SF 1.234560E+02**
DT14・DT15	SF 5.678900E+02	DT4・DT5	SF 0.000000E+00

Example 2) Operation unit: Single-precision, floating-point real number (SF) (negative value)

[i]…SF
[S]…DT10 [D]…DT2

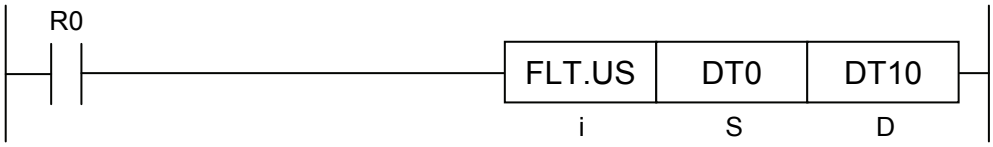
DT10・DT11	**SF -1.234560E+02**	DT0・DT1	SF 0.000000E+00
DT12・DT13	SF -3.456780E+02	DT2・DT3	**SF 1.234560E+02**
DT14・DT15	SF -5.678900E+02	DT4・DT5	SF 0.000000E+00

■ Flag operations

Name	Explanation
SR7	To be set in the case of out-of-range in indirect access (index modification).
SR8 (ER)	To be set when a non-real number is specified for [S].

FLT (Conversion: Integer → Floating Point Real Number Data)

■ Ladder diagram



■ Available operation units (●: Available)

Operation unit	bit	US	SS	UL	SL	SF	DF
i		●	●	●	●		

■ List of operands

Operand	Explanation
S	Conversion target data (device address or constant (data format: according to the operation unit))
D	Conversion result (device address (data format: single precision floating point real number data)

■ Available devices (●: Available)

Operand	16-bit device											32-bit device *1			Integers			Real numbers		String	Index modifier
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX *2	K *3	U *4	H	SF	DF	" "	
S	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●				●
D	●	●	●	●			●	●	●		●	●	●	●							●

*1: Operation unit: 16-bit integers (SS, US) cannot be specified.
*2: Index register (I0 to IE)
*3: Only operation unit: signed integers (SS, SL) can be specified.
*4: Only operation unit: unsigned integers (US, UL) can be specified.

■ Outline of operation

- According to the operation unit [i], the integer value stored in the area starting with [S] is converted into a single precision floating point real number value.
- The calculation result is stored in the area starting with [D].

■ Process details

Example 1) Unsigned 16 bits (US)

[i]…US
[S]…DT0 [D]…DT10

DT0	**U 123**	→	DT10・DT11	**SF 1.230000E＋02**
DT1	U 456		DT12・DT13	SF 0.000000E＋00
DT2	U 789		DT14・DT15	SF 0.000000E＋00

Example 2) Signed 16 bits (SS) (positive value)

[i]…SS
[S]…DT20 [D]…DT10

DT20	**K 123**	→	DT10・DT11	**SF 1.230000E＋02**
DT21	K 456		DT12・DT13	SF 0.000000E＋00
DT22	K 789		DT14・DT15	SF 0.000000E＋00

Example 3) Signed 16 bits (SS) (negative value)

[i]…SS
[S]…DT20 [D]…DT10

DT20	**K -123**	→	DT10・DT11	**SF -1.230000E＋02**
DT21	K -456		DT12・DT13	SF 3.456780E＋02
DT22	K -789		DT14・DT15	SF 5.678900E＋02

Example 4) Unsigned 32 bits (UL)

[i]…UL
[S]…DT0 [D]…DT10

DT0・DT1	**U 12345**	→	DT10・DT11	**SF 1.234500E＋04**
DT2・DT3	U 67890		DT12・DT13	SF 0.000000E＋00
DT4・DT5	U 13579		DT14・DT15	SF 0.000000E＋00

Example 5) Signed 32 bits (SL) (positive value)

[i]…SL
[S]…DT20 [D]…DT10

DT20・DT21	**K 12345**	→	DT10・DT11	**SF 1.234500E＋04**
DT22・DT23	K 67890		DT12・DT13	SF 0.000000E＋00
DT24・DT25	K 13579		DT14・DT15	SF 0.000000E＋00

Example 6) Signed 32 bits (SL) (negative value)

[i]…SL
[S]…DT20 [D]…DT10

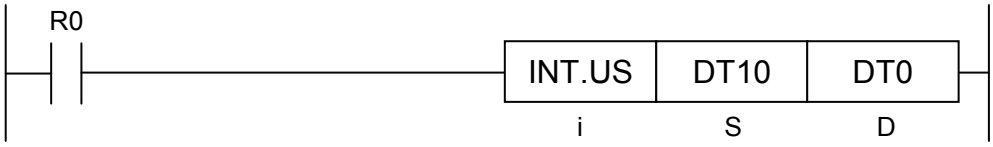
DT20・DT21	**K -12345**	→	DT10・DT11	**SF -1.234500E＋04**
DT22・DT23	K -67890		DT12・DT13	SF 0.000000E＋00
DT24・DT25	K -13579		DT14・DT15	SF 0.000000E＋00

■ Flag operations

Name	Explanation
SR7 SR8 (ER)	To be set in the case of out-of-range in indirect access (index modification).

INT (Conversion: Floating Point Real Number Data → Integer) (Max. Value Not Exceeding the Data)

■ Ladder diagram



■ Available operation units (●: Available)

Operation unit	bit	US	SS	UL	SL	SF	DF
i		●	●	●	●		

■ List of operands

Operand	Explanation
S	Conversion target data (device address or constant (data format: according to the operation unit))
D	Conversion target data (device address or constant (data format: according to the operation unit))

■ Available devices (●: Available)

Operand	16-bit device											32-bit device *1			Integers			Real numbers		String	Index modifier *2
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX *3	K	U	H	SF	DF	" "	
S	●	●	●	●			●	●	●	●	●	●	●	●	●	●	●	●			●
D	●	●	●	●			●	●	●		●	●	●	●							●

*1: Operation unit: 16-bit integers (SS, US) cannot be specified.
*2: Only 16-bit devices, and 32-bit devices can be modified (real number constants cannot be specified).
*3: Index register (I0 to IE)

■ Outline of operation

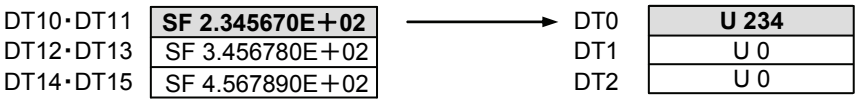
- According to the operation unit [i], the single precision floating point real number value, stored in the area starting with [S], is converted into an integer value (max. value not exceeding the data).
- The calculation result is stored in the area starting with [D].
- The data setting ranges for conversion target data [S] by operation unit are indicated below.

	Min. value	Max. value
US	0	65,530
SS	**-32,760**	32,760
UL	0	4,294,967,000
SL	**-2,147,483,000**	2,147,483,000

■ Process details

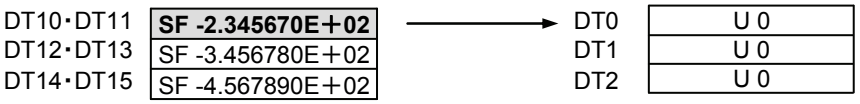
Example 1) Unsigned 16 bits (US) (positive value)

[i]…US
[S]…DT10 [D]…DT0



Example 2) Unsigned 16 bits (US) (negative value)

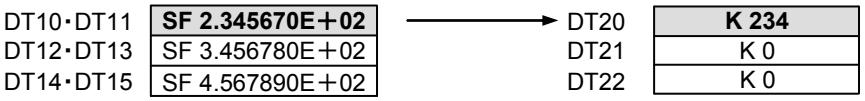
[i]…US
[S]…DT10 [D]…DT0



* Operation error occurs if an unsigned integer is specified for the operation unit and a negative value is converted.

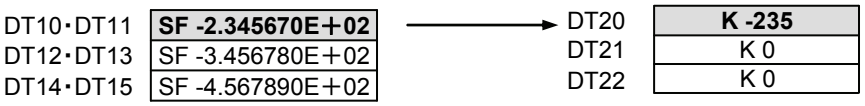
Example 3) Signed 16 bits (SS) (positive value)

[i]…SS
[S]…DT10 [D]…DT20



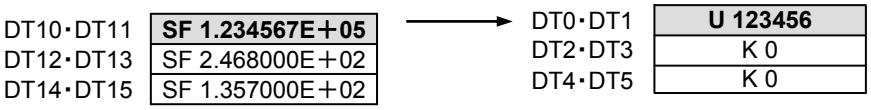
Example 4) Signed 16 bits (SS) (negative value)

[i]…SS
[S]…DT10 [D]…DT20



Example 5) Unsigned 32 bits (UL) (positive value)

[i]…UL
[S]…DT10 [D]…DT0



Example 6) Unsigned 32 bits (UL) (negative value)

[i]…UL
[S]…DT10 [D]…DT0

DT10・DT11	**SF -1.234567E＋05**	→	DT0・DT1	U 0
DT12・DT13	SF -2.468000E＋02		DT2・DT3	U 0
DT14・DT15	SF -1.357000E＋02		DT4・DT5	U 0

* Operation error occurs if an unsigned integer is specified for the operation unit and a negative value is converted.

Example 7) Signed 32 bits (SL) (positive value)

[i]…SL
[S]…DT10 [D]…DT20

DT10・DT11	**SF 1.234567E＋05**	→	DT20・DT21	**K 123456**
DT12・DT13	SF 2.468000E＋02		DT22・DT23	K 0
DT14・DT15	SF 1.357000E＋02		DT24・DT25	K 0

Example 8) Signed 32 bits (SL) (negative value)

[i]…SL
[S]…DT10 [D]…DT20

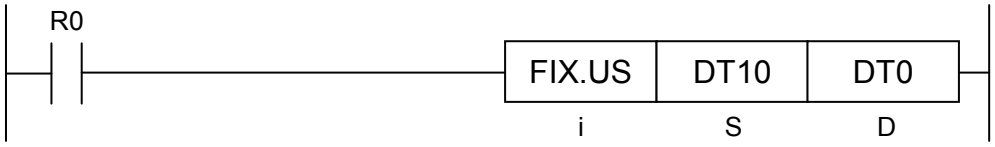
DT10・DT11	**SF -1.234567E＋05**	→	DT20・DT21	**K -12346**
DT12・DT13	SF -2.468000E＋02		DT22・DT23	K 0
DT14・DT15	SF -1.357000E＋02		DT24・DT25	K 0

■ Flag operations

Name	Explanation
SR7 SR8 (ER)	To be set in the case of out-of-range in indirect access (index modification).
	To be set when a non-real number is specified for [S].
	To be set when an out-of-range value is specified for [S] (conversion target data).

FIX (Conversion: Floating Point Real Number Data → Integer (Rounding the First Decimal Point Down)

■ Ladder diagram



■ Available operation units (●: Available)

Operation unit	bit	US	SS	UL	SL	SF	DF
i		●	●	●	●		

■ List of operands

Operand	Explanation
S	Conversion target data (device address or constant (data format: according to the operation unit))
D	Conversion target data (device address or constant (data format: according to the operation unit))

■ Available devices (●: Available)

Operand	16-bit device											32-bit device *1			Integers			Real numbers		String	Index modifier *2
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX *3	K	U	H	SF	DF	" "	
S	●	●	●	●			●	●	●	●	●	●	●	●				●			●
D	●	●	●	●			●	●	●		●	●	●	●							●

*1: Operation unit: 16-bit integers (SS, US) cannot be specified.
*2: Only 16-bit devices, and 32-bit devices can be modified (real number constants cannot be specified).
*3: Index register (I0 to IE)

■ Outline of operation

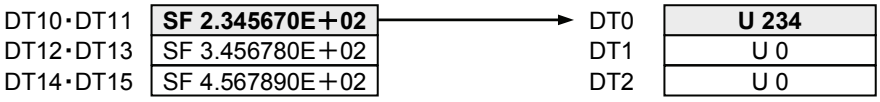
- According to the operation unit [i], the single precision floating point real number value, stored in the area starting with [S], is converted into an integer value (rounding the first decimal point down).
- The calculation result is stored in the area starting with [D].
- The data setting ranges for conversion target data [S] by operation unit are indicated below.

	Min. value	Max. value
US	0	65,530
SS	**-32,760**	32,760
UL	0	4,294,967,000
SL	**-2,147,483,000**	2,147,483,000

■ Process details

Example 1) Unsigned 16 bits (US) (positive value)

[i]…US
[S]…DT10 [D]…DT0



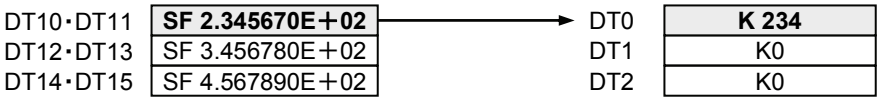
Example 2) Unsigned 16 bits (US) (negative value)

[i]…US
[S]…DT10 [D]…DT0

* Operation error occurs if an unsigned integer is specified for the operation unit and a negative value is converted.

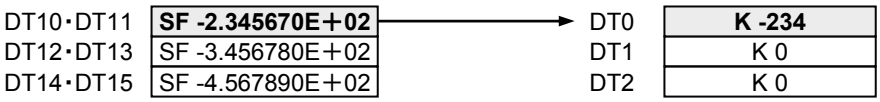
Example 3) Signed 16 bits (SS) (positive value)

[i]…SS
[S]…DT10 [D]…DT0



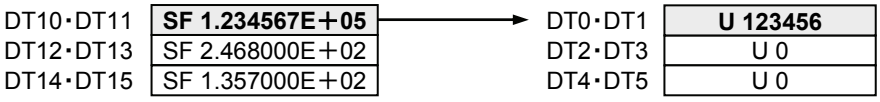
Example 4) Signed 16 bits (SS) (negative value)

[i]…SS
[S]…DT10 [D]…DT0



Example 5) Unsigned 32 bits (UL) (positive value)

[i]…UL
[S]…DT10 [D]…DT0



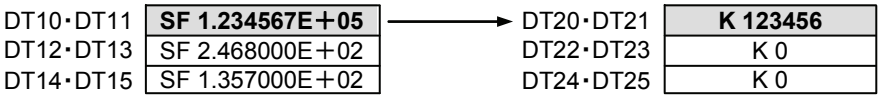
Example 6) Unsigned 32 bits (UL) (negative value)

[i]…UL
[S]…DT10 [D]…DT0

* Operation error occurs if an unsigned integer is specified for the operation unit and a negative value is converted.

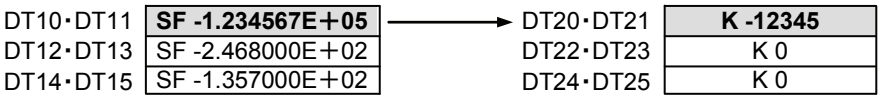
Example 7) Signed 32 bits (SL) (positive value)

[i]…SL
[S]…DT10 [D]…DT20



Example 8) Signed 32 bits (SL) (negative value)

[i]…SL
[S]…DT10 [D]…DT20

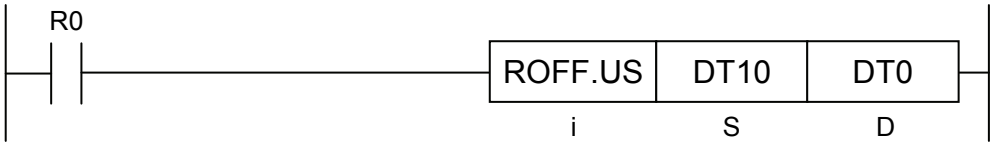


■ Flag operations

Name	Explanation
SR7 SR8 (ER)	To be set in the case of out-of-range in indirect access (index modification).
	To be set when a non-real number is specified for [S].
	To be set when an out-of-range value is specified for [S] (conversion target data).

ROFF (Conversion: Floating Point Real Number Data → Integer) (Rounding the First Decimal Point Off)

■ Ladder diagram



■ Available operation units (●: Available)

Operation unit	bit	US	SS	UL	SL	SF	DF
i		●	●	●	●		

■ List of operands

Operand	Explanation
S	Conversion target data (device address or constant (data format: according to the operation unit))
D	Conversion target data (device address or constant (data format: according to the operation unit))

■ Available devices (●: Available)

Operand	16-bit device											32-bit device *1			Integers			Real numbers		String	Index modifier *2
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX *3	K	U	H	SF	DF	" "	
S	●	●	●	●			●	●	●	●	●	●	●	●				●			●
D	●	●	●	●			●	●	●		●	●	●	●							●

*1: Operation unit: 16-bit integers (SS, US) cannot be specified.
*2: Only 16-bit devices, and 32-bit devices can be modified (real number constants, and character constants cannot be specified).
*3: Index register (I0 to IE)

■ Outline of operation

- According to the operation unit [i], the single precision floating point real number value, stored in the area starting with [S], is converted into an integer value (rounding the first decimal point off).
- The calculation result is stored in the area starting with [D].
- The data setting ranges for conversion target data [S] by operation unit are indicated below.

	Min. value	Max. value
US	0	65,530
SS	**-32,760**	32,760
UL	0	4,294,967,000
SL	**-2,147,483,000**	2,147,483,000

■ Process details

Example 1) Unsigned 16 bits (US) (positive value)

[i]…US
[S]…DT10 [D]…DT0

DT10・DT11	**SF 2.345670E+02**	→	DT0	**U 235**
DT12・DT13	SF 3.456780E+02		DT1	U 0
DT14・DT15	SF 4.567890E+02		DT2	U 0

Example 2) Unsigned 16 bits (US) (negative value)

[i]…US
[S]…DT10 [D]…DT0

DT10・DT11	**SF -2.345670E+02**	→	DT0	U 0
DT12・DT13	SF -3.456780E+02		DT1	U 0
DT14・DT15	SF -4.567890E+02		DT2	U 0

* Operation error occurs if an unsigned integer is specified for the operation unit and a negative value is converted.

Example 3) Coded 16 bits (SS) (positive value)

[i]…SS
[S]…DT10 [D]…DT0

DT10・DT11	**SF 2.345670E+02**	→	DT0	**K 235**
DT12・DT13	SF 3.456780E+02		DT1	K 0
DT14・DT15	SF 4.567890E+02		DT2	K 0

Example 4) Coded 16 bits (SS) (negative value)

[i]…SS
[S]…DT10 [D]…DT0

DT10・DT11	**SF -2.345670E+02**	→	DT0	**K -235**
DT12・DT13	SF -3.456780E+02		DT1	K 0
DT14・DT15	SF -4.567890E+02		DT2	K 0

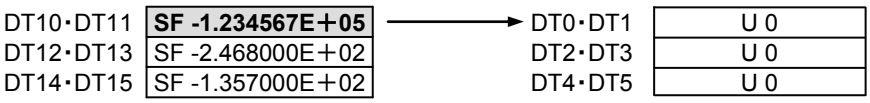
Example 5) Unsigned 32 bits (UL) (positive value)

[i]…UL
[S]…DT10 [D]…DT0

DT10・DT11	**SF 1.234567E+05**	→	DT0・DT1	**U 123457**
DT12・DT13	SF 2.468000E+02		DT2・DT3	U 0
DT14・DT15	SF 1.357000E+02		DT4・DT5	U 0

Example 6) Unsigned 32 bits (UL) (negative value)

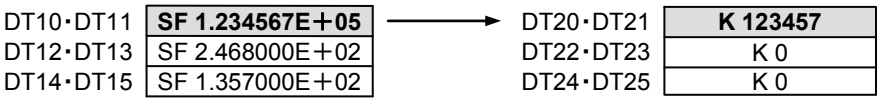
[i]…UL
[S]…DT10 [D]…DT0



* Operation error occurs if an unsigned integer is specified for the operation unit and a negative value is converted.

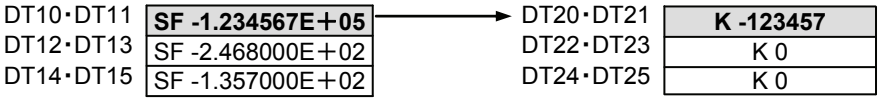
Example 7) Coded 32 bits (SL) (positive value)

[i]…SL
[S]…DT10 [D]…DT20



Example 8) Coded 32 bits (SL) (negative value)

[i]…SL
[S]…DT10 [D]…DT20

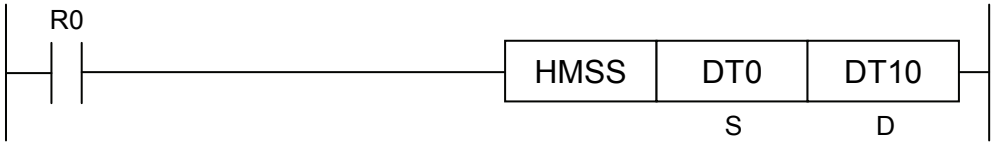


■ Flag operations

Name	Explanation
SR7 SR8 (ER)	To be set in the case of out-of-range in indirect access (index modification).
	To be set when a non-real number is specified for [S].
	To be set when an out-of-range value is specified for [S] (conversion target data).

HMSS (Conversion: Time Data (Hours, Minutes and Seconds) → Seconds Data)

■ Ladder diagram



■ Available operation units (●: Available)

- No operation units

■ List of operands

Operand	Explanation
S	Starting device address of time data
D	Device address of seconds data

■ Available devices (●: Available)

Operand	16-bit device											32-bit device			Integers			Real numbers		String	Index modifier
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX	K	U	H	SF	DF	" "	
S	●	●	●	●	●	●	●	●	●	●	●										●
D	●	●	●	●			●	●	●		●										●

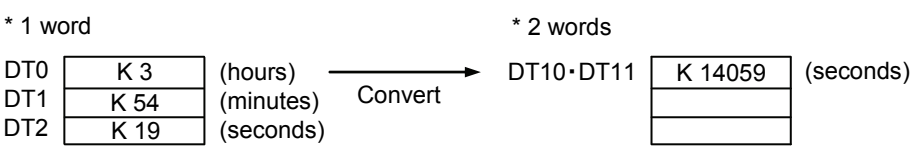
■ Outline of operation

- Time data stored in the area starting with [S], comprising respectively 1 word for hours, minutes and seconds, are converted into a 2-word integer representing seconds data.
- The calculation result is stored in the area starting with [D].

■ Process details

Example) Convert 3 hours, 54 minutes and 19 seconds

[S]…DT0 [D]…DT10

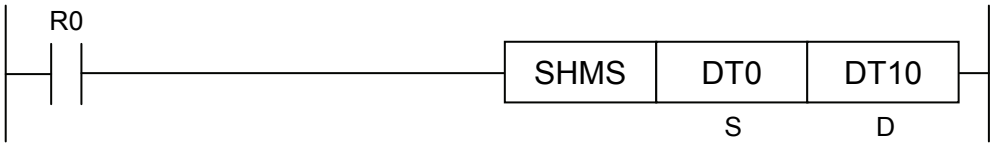


■ Flag operations

Name	Explanation
SR7 SR8 (ER)	To be set in the case of out-of-range in indirect access (index modification).
	To be set when the time data range is exceeded.

SHMS (Conversion: Seconds Data → Time Data (Hours, Minutes and Seconds)

■ Ladder diagram



■ Available operation units (●: Available)

- No operation units

■ List of operands

Operand	Explanation
S	Device address of seconds data (available data range: 0 to 35,999,999)
D	Starting device address of time data

■ Available devices (●: Available)

Operand	16-bit device											32-bit device			Integers			Real numbers		String	Index modifier
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX	K	U	H	SF	DF	" "	
S	●	●	●	●	●	●	●	●	●	●	●										●
D	●	●	●	●			●	●	●		●										●

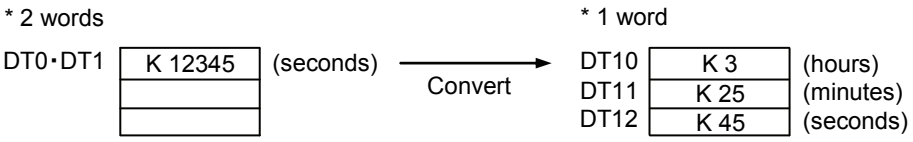
■ Outline of operation

- Two-word integer representing seconds data, stored in the area starting with [S], are converted into data comprising respectively 1 word for hours, minutes and seconds.
- The calculation result is stored in the area starting with [D].

■ Process details

Example) Convert 12,345 seconds

[S]…DT0 [D]…DT10

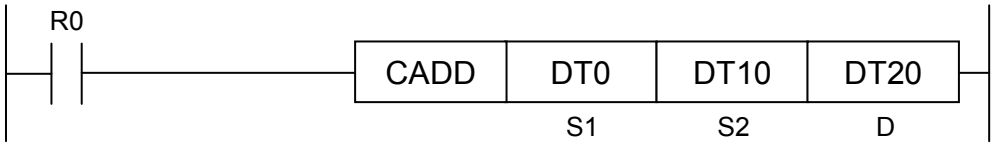


■ Flag operations

Name	Explanation
SR7 SR8 (ER)	To be set in the case of out-of-range in indirect access (index modification).
	To be set when the seconds data range is exceeded.

CADD (Clock Addition)

■ Ladder diagram



■ Available operation units (●: Available)

- No operation units

■ List of operands

Operand	Explanation
S1	Starting device address of clock data
S2	Starting device address of time data
D	Starting device address of addition result

■ Available devices (●: Available)

Operand	16-bit device											32-bit device			Integers			Real numbers		String	Index modifier
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX	K	U	H	SF	DF	" "	
S1	●	●	●	●			●	●													●
S2	●	●	●	●			●	●													●
D	●	●	●	●			●	●													●

■ Outline of operation

- Time data (hours, minutes and seconds), stored in the area starting with [S2], are added to the clock data (Year, month, day, hours, minutes and seconds), stored in the area starting with [S1].
- The calculation result is stored in the area starting with [D].
- This operation takes leap years into account.

■ Process details

Example) Add 20 hours, 23 minutes and 45 seconds to 08:54:19, January 1, 2012
[S1]...DT0 [S2]...DT10 [D]...DT20

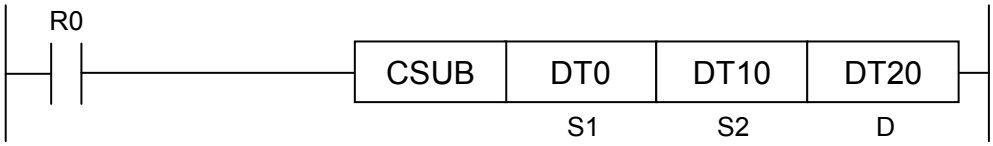
DT0	K 12	(year)				DT20	K 12	(year)
DT1	K 1	(month)				DT21	K 1	(month)
DT2	K 1	(day)				DT22	K 2	(day)
DT3	K 8	(hours)	+ DT10	K 20	(hours)	DT23	K 5	(hours)
DT4	K 54	(minutes)	DT11	K 23	(minutes)	DT24	K 18	(minutes)
DT5	K 19	(seconds)	DT12	K 45	(seconds)	DT25	K 4	(seconds)

■ Flag operations

Name	Explanation
SR7	To be set in the case of out-of-range in indirect access (index modification).
SR8	To be set when the clock data range and/or the time data range is exceeded.
(ER)	To be set when the addition result is out of the accessible range.

CSUB (Clock Subtraction)

■ Ladder diagram



■ Available operation units (●: Available)

- No operation units

■ List of operands

Operand	Explanation
S1	Starting device address of clock data
S2	Starting device address of time data
D	Starting device address of subtraction result

■ Available devices (●: Available)

Operand	16-bit device											32-bit device			Integers			Real numbers		String	Index modifier
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX	K	U	H	SF	DF	" "	
S1	●	●	●	●			●	●													●
S2	●	●	●	●			●	●													●
D	●	●	●	●			●	●													●

■ Outline of operation

- Time data (hours, minutes and seconds), stored in the area starting with [S2], are subtracted from the clock data (Year, month, day, hours, minutes and seconds), stored in the area starting with [S1].
- The calculation result is stored in the area starting with [D].
- This operation takes leap years into account.

■ Process details

Example) Subtract 20 hours, 23 minutes and 45 seconds from 08:54:19, January 1, 2012
[S1]…DT0 [S2]…DT10 [D]…DT20

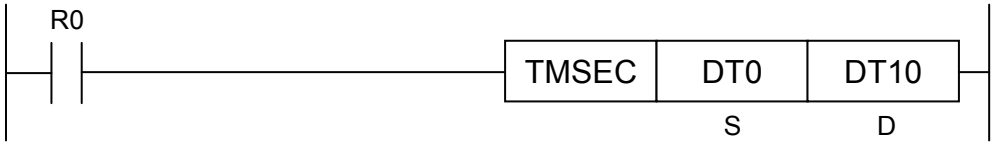
DT0	K 12	(year)				DT20	K 11	(year)
DT1	K 1	(month)				DT21	K 12	(month)
DT2	K 1	(day)				DT22	K 31	(day)
DT3	K 8	(hours)	DT10	K 20	(hours)	DT23	K 12	(hours)
DT4	K 54	(minutes)	DT11	K 23	(minutes)	DT24	K 30	(minutes)
DT5	K 19	(seconds)	DT12	K 45	(seconds)	DT25	K 34	(seconds)

■ Flag operations

Name	Explanation
SR7	To be set in the case of out-of-range in indirect access (index modification).
SR8	To be set when the clock data range and/or the time data range is exceeded.
(ER)	To be set when the subtraction result is out of the accessible range.

TMSEC (Calculation: Clock Data → Seconds Data from the Base Time)

■ Ladder diagram



■ Available operation units (●: Available)

- No operation units

■ List of operands

Operand	Explanation
S	Device address of clock data (available data range: 1970/1/1 00:00:00 to 2106/2/6 23:59:59)
D	Device address of seconds data from the base time

■ Available devices (●: Available)

Operand	16-bit device											32-bit device			Integers			Real numbers		String	Index modifier
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX	K	U	H	SF	DF	" "	
S	●	●	●	●	●	●	●	●	●	●	●										●
D	●	●	●	●			●	●	●		●										●

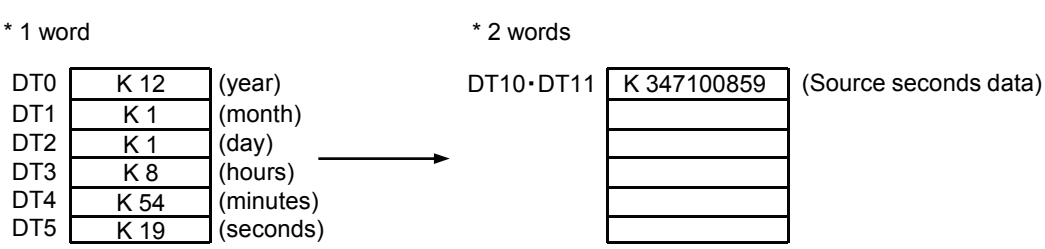
■ Outline of operation

- From the clock data (year, month, day, hours, minutes and seconds) stored in the area starting with [S], time elapsed from the base time is calculated.
- The calculation result is stored in the area starting with [D].
- The base time is 2001/1/1 00:00:00.

■ Process details

Example) Calculate seconds data against the base time, from 08:54:19, January 1, 2012

[S]…DT0 [D]…DT10

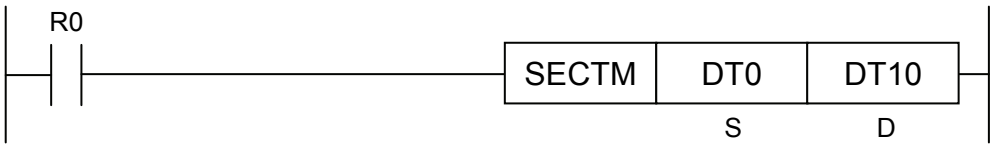


■ Flag operations

Name	Explanation
SR7	To be set in the case of out-of-range in indirect access (index modification).
SR8 (ER)	To be set when the clock data range is exceeded.

SECTM (Calculation: Seconds Data from the Base Time → Clock Data)

■ Ladder diagram



■ Available operation units (●: Available)

- No operation units

■ List of operands

Operand	Explanation
S	Device address of seconds data from the base time
D	Starting device address of clock data (available data range: 1970/1/1 00:00:00 to 2106/2/6 23:59:59)

■ Available devices (●: Available)

Operand	16-bit device											32-bit device			Integers			Real numbers		String	Index modifier
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX *1	K	U	H	SF	DF	" "	
S	●	●	●	●	●	●	●	●	●	●	●										●
D	●	●	●	●			●	●	●		●										●

■ Outline of operation

- From the time elapsed from the base time, stored in the area starting with [S], clock data (year, month, day, hours, minutes and seconds) is calculated.
- The calculation result is stored in the area starting with [D].
- The base time is 2001/1/1 00:00:00.

■ Process details

Example) Calculate data from 1,325,408,059 seconds

[S]…DT0 [D]…DT10

* 2 words

DT0・DT1	K 1325408059	(Source seconds data)

* 1 word

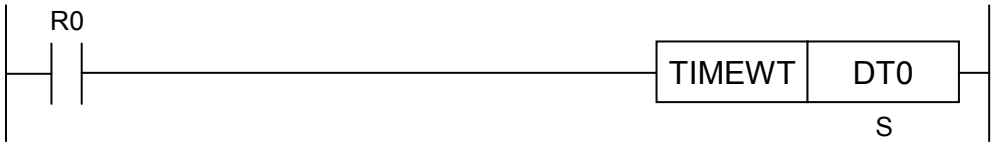
DT10	K 12	(year)
DT11	K 1	(month)
DT12	K 1	(day)
DT13	K 8	(hours)
DT14	K 54	(minutes)
DT15	K 19	(seconds)

■ Flag operations

Name	Explanation
SR7	To be set in the case of out-of-range in indirect access (index modification).
SR8 (ER)	To be set when the clock data range is exceeded.

TIMEWT (Setting of Clock/Calendar)

■ Ladder diagram



■ Available operation units (●: Available)

- No operation units

■ List of operands

Operand	Explanation
S	Starting device address of clock data

(Note) Only this instruction comprises 7 words in total, including day of week.

■ Available devices (●: Available)

Operand	16-bit device											32-bit device			Integers			Real numbers		String	Index modifier
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX	K	U	H	SF	DF	" "	
S	●	●	●	●	●	●	●	●	●												●

■ Outline of operation

- The clock data (year, month, day, hours, minutes and seconds), stored in the area starting with [S], is set as RTC data for the CPU unit.
- The range of clock data that can be set for clock/calender of the FP7 CPU unit is as follows.
  2000/1/1 00:00:00 to 2099/12/31 23:59:59

■ Process details

Example) Specify 08:54:19, January 1, 2012
[S]…DT0

* 1 word

DT0	K 12	(year)		K 12	(year)
DT1	K 1	(month)		K 1	(month)
DT2	K 1	(day)		K 1	(day)
DT3	K 8	(hours)	Update →	K 8	(hours)
DT4	K 54	(minutes)		K 54	(minutes)
DT5	K 19	(seconds)		K 19	(seconds)
DT6	K 0	(day of week)		K 0	(day of week)

● Day-of-week data

0	Sun
1	Mon
2	Tue
3	Wed
4	Thu
5	Fri
6	Sat

■ Precautions during programming

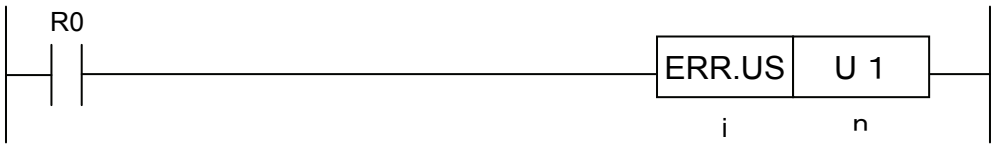
- Consistency of the day of week data with the date is not checked.

■ Flag operations

Name	Explanation
SR7	To be set in the case of out-of-range in indirect access (index modification).
SR8	To be set when the clock data range is exceeded.
(ER)	To be set when the day of week range is exceeded.

ERR (Self-Diagnostic Error Code Set)

■ Ladder diagram



■ Available operation units (●: Available)

Operation unit	bit	US	SS	UL	SL	SF	DF
i		●					

■ List of operands

Operand	Explanation
n	Specify a self-diagnostic error code.

■ Available devices (●: Available)

Operand	16-bit device											32-bit device			Integers			Real numbers		String	Index modifier
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX *1	K	U	H	SF	DF	" "	
n																●	●				

■ Outline of operation

- In a user program, set an optional error code.
- Store a self-diagnostic error code, specified by [n], in the self-diagnostic abnormality code register (SD0), and set the self-diagnostic abnormality flag (S0).
- It is also possible to describe an ERR instruction to set the same error code.

■ Setting of a self-diagnostic error code [n]

- n (self-diagnostic error code) can be set within the range from U1000 to U2999.

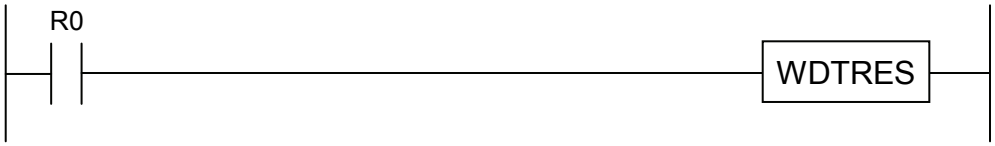
Setting of [n] (self-diagnostic error code)	Operations when error occurs
U 1000 to 1999	Operation stop
U 2000 to 2999	Continue operation

■ Flag operations

Name	Explanation
SR7 SR8 (ER)	To be set when the error code [n] becomes out of the range.

WDTRES (Watchdog Timer Reset)

■ Ladder diagram



■ Available operation units (●: Available)

- No operation units

■ List of operands

- No operands

■ Available devices (●: Available)

- No available devices

■ Outline of operation

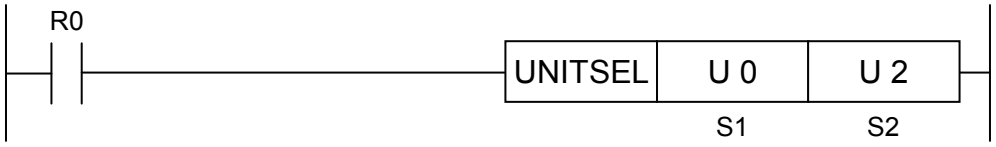
- The watchdog timer is reset.

■ Flag operations

- No change occurs.

UNITSEL (Specification of a Communication Unit Slot Port)

■ Ladder diagram



■ Available operation units (●: Available)

- No operation units

■ List of operands

Operand	Explanation
S1	Slot no. of the unit
S2	COM port no. or connection no.

■ Available devices (●: Available)

Operand	16-bit device											32-bit device			Integers			Real numbers		String	Index modifier
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX *1	K	U	H	SF	DF	" "	
S1	●	●	●	●			●	●								●					●
S2	●	●	●	●			●	●								●					●

■ Outline of operation

- To be described immediately before the following communication instructions, to specify the targets of execution.
  GPSEND, GPRECV, SEND, RECV, RDET, PMSET, PMGET
- In the case of CPU with built-in SCU, specify a slot no. (U0) and a COM port no.
- In the case of CPU with built-in ET-LAN, specify a slot no. (U100) and a connection no.
- In the case of serial communication unit, specify a slot no. (U1 to 16) and a COM port no.
- Obtain the type of slot specified by [S1], and check that the communication port no. specified by [S2] falls within the available range. If the number is out of the range, an error will occur.
- In the case of SCU, check that the specified communication port (COM port no.) is equipped with a communication cassette. If the specified COM port is not equipped with a communication cassette, an error will occur.
- If no error occurs, the values of [S1] and [S2] should be set to system data (SD40, SD41) of the CPU unit.

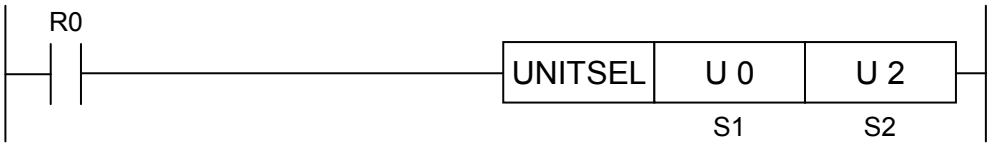
■ Specification of [S1] and [S2]

- Specify a slot no. of the communication unit in [S1].
- The set point value for [S1] should be set to system data SD40.
- Specify a communication port in [S2]. (In the case of SCU: COM port no.; In the case of CPU with built-in ET-LAN: connection no.)
- The set point value for [S2] should be set to system data SD41.

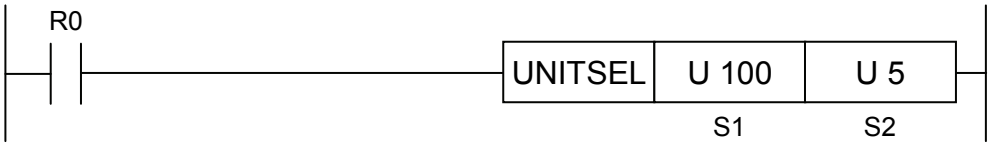
Types of unit	[S1] Slot No.	[S2] COM Port No. Connection No.
CPU with built-in SCU	U0	U0 to U2
CPU with built-in ET-LAN	U100	U1 to U16
Serial Communication Unit	U1 to U16	U1 to U4

■ Program example

Example 1) Specify COM2 for SCU with built-in CPU in Slot 0



Example 2) Specify User Connection 5 for ET-LAN with built-in CPU in Slot 100

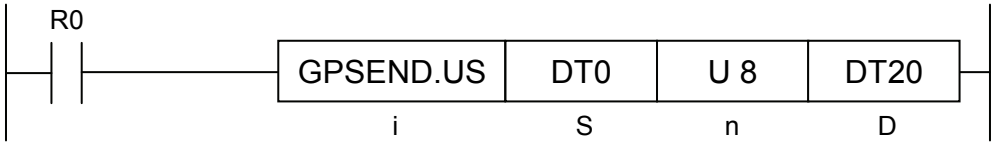


■ Flag operations

Name	Explanation
SR7 SR8 (ER)	To be set in the case of out-of-range in indirect access (index modification).
	The COM port specified by [S2] does not exist (no cassette, not a communication cassette).
	The connection specified by [S2] does not exist (out of the connection no. range)

GPSEND (General-Purpose Communication Send Instruction)

■ Ladder diagram



■ Available operation units (●: Available)

Operation unit	bit	US	SS	UL	SL	SF	DF
i		●	●				

■ List of operands

Operand	Explanation
S	Starting no. of the source data area
n	No. of bytes of the sent data (always 16-bit data, regardless of the operation unit [i])
D	Starting no. of the device area in the master unit that stores the processing result (1 word) (Always 1-word (16-bit) data, regardless of the operation unit [i])

(Note) When a negative integer value is specified for [n], it is necessary to specify an SS (signed) operation unit.

■ Available devices (●: Available)

Operand	16-bit device											32-bit device			Integers			Real numbers		String	Index modifier *1
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX *1	K	U	H	SF	DF	" "	
S1	●	●	●	●			●	●													●
n(*2)	●	●	●	●			●	●							●	●	●				●
D(*2)	●	●	●	●			●	●													●

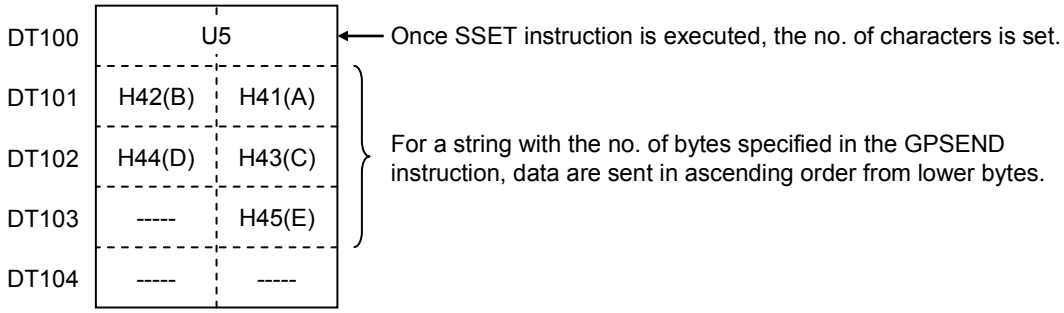
*1: Index modification is not available for the UM/WI/WO area.
*2: Always 16-bit data/device, regardless of the specification of operation unit [i].

■ Outline of operation

- Data are sent from the communication port of the unit to external devices.
- Data of [n] bytes are sent from the communication unit / communication port set by the UNITSEL instruction, starting with the starting address (word address) of the sent data area specified by [S].
- Data to be sent are set by the user program, in the area starting with [S].
- The processing result is stored in the area specified by [D].

■ Formulation of a sent data table [S]

- Data to be sent is stored in a given data register by the user program.



■ Specification of the no. of bytes in sent data [n]

Types of unit	Setting of sent data amount	Explanation
SCU *1	1 to 4094	In the case of a positive value, a start code and an end code set by the configuration menu are automatically added. Do not include a start code or an end code in the sent data.
	-1 to -4096	End code is not added in the case of a negative value.
CPU with built-in ET-LAN	1 to 16372	When "Add a special header" is on (default of connection settings): Sent data and end code are not distinguished. (Not automatically added)
	1 to 16384 *2	When "Add no special header" is on: Sent data and end code are not distinguished. (Not automatically added)

*1: In the SCU settings, if the automatic addition of start code is enabled, the max. sent data amount is reduced by 1. Up to 4096 bytes can be sent, including a start code and an end code.

*2: When sending to FP2 ET-LAN, set "Add no special header" on in the user connection settings, and set the max. sent bytes within 8192 bytes in the user specification.

■ Content of the processing result [D]

Status	Value to be set
Sending done	No. of sent bytes
When an error occurred:	FFFFH

■ Precautions during programming

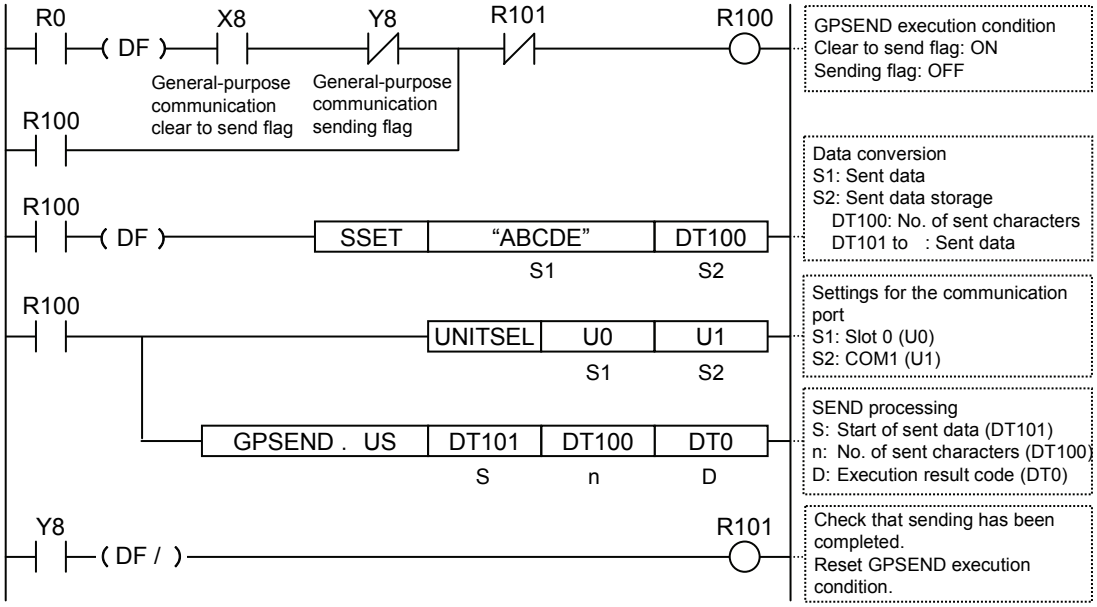
- In order to enable communication, settings should be made in the configuration menu of the tool software FPWIN GR7. In the communication mode section, select "general-purpose communication".
- As for execution condition of the GPSEND instruction, retain the ON condition until sending is done and the general-purpose sending flag turns off.
- Check that the general-purpose communication clear to send flag is on for the targeted COM port, and execute the GPSEND instruction.
- If the GPSEND instruction is executed for the communication port where sending is in progress, the sending flag and the execution result are updated.
- The GPSEND instruction cannot be used in an interrupt program.

◆ KEY POINTS

- **The case of SCU shows the case that it is used in the following combination.**
  - **- COM0 port equipped in the CPU unit**
  - **- Communication cassettes attached to the CPU unit (COM.1 to COM.2 ports)**
  - **- Communication cassettes attached to the serial communication unit (COM.1 to COM.4 ports)**
- **As the communication cassette (Ethernet type) has an Ethernet-serial conversion function, the internal interface operates with similar programs as the case of SCU. The setting method and programming method are different from those for the CPU with built-in ET-LAN.**

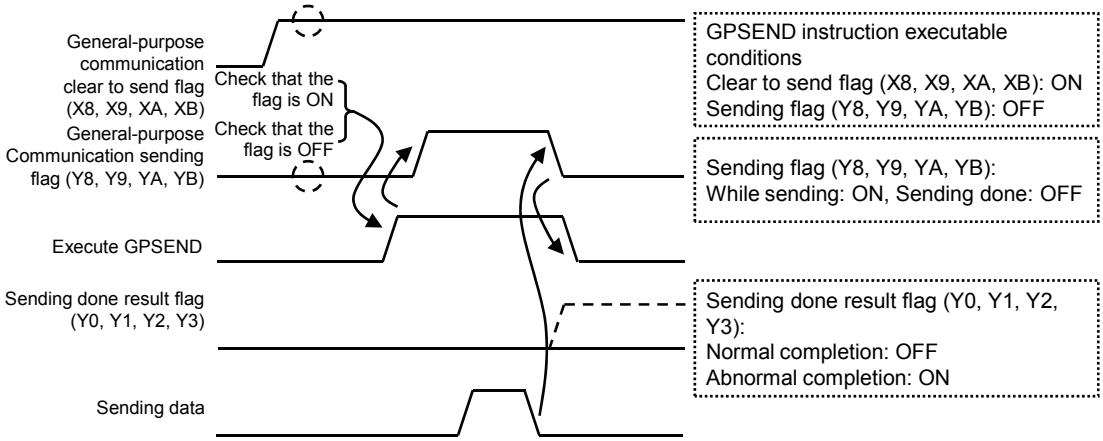
■ Sample program (in the case of SCU)

- Check that the general-purpose communication mode is on (X8), and that general-purpose sending is not in progress in the same port (Y8), and start up the sending program.
- Using the SSET instruction, convert a given message into an ASCII string. Set the no. of sent strings to the data register DT100, and the sent message to the data register DT101.
- Using the UNITSEL instruction, specify the slot no. (U0) and the COM. port no. (U1).
- In the GPSEND instruction, specify and execute the start of the table that stores the message to be sent (DT101) and the no. of characters in the data (DT100).



■ Time chart (in the case of SCU)

- Data are sent in ascending order from lower bytes of [S] in the table specified by the GPSEND instruction.
- During sending, the general-purpose communication sending flags (Y8, Y9, YA, YB) are turned on. They turn off when sending is completed. (The flags do not turn off immediately after the execution of the instruction but at the start of the second scan.)
- The sending result (0: normal completion; 1: abnormal completion) is stored in the general-purpose communication sending done result flags (Y0, Y1, Y2, Y3).



■ I/O allocation (In the case of CPU with built-in SCU)

COM Port No.			Name	Name
1	2	0		
X8	X9	XA	General-purpose communication clear to send flag	Turns on when the general-purpose communication mode is set.
Y8	Y9	YA	General-purpose communication sending flag	Turns on during sending by the general-purpose communication GPSEND. Turns off when sending is completed.
Y0	Y1	Y2	Sending done result flag	Completion result of sending by general-purpose communication / master communication is notified. (Normal completion: 0; Abnormal completion: 1)

(Note 1) Each contact is used for reading the operation status. Avoid writing by a user program.
(Note 2) If sending time is shorter than scan time, the general-purpose communication sending flags (Y8, Y9, YA, YB) are turned off when the GPSEND instruction is executed in the subsequent scan following completion of data sending. The flags remain ON for at least one scan time.

■ I/O allocation (In the case of Serial Communication Unit)

COM Port No.				Name	Name
1	2	3	4		
X8	X9	XA	XB	General-purpose communication clear to send flag	Turns on when the general-purpose communication mode is set.
Y8	Y9	YA	YB	General-purpose communication sending flag	Turns on during sending by the general-purpose communication GPSEND. Turns off when sending is completed.
Y0	Y1	Y2	Y3	Sending done result flag	Completion result of sending by general-purpose communication / master communication is notified. (Normal completion: 0; Abnormal completion: 1)

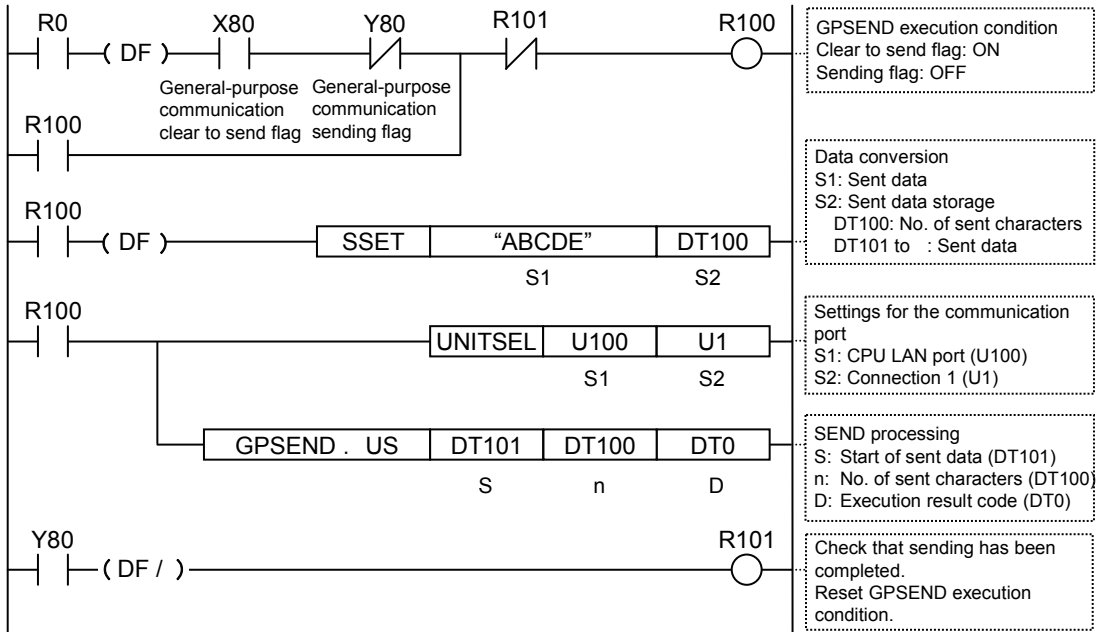
(Note 1) Each contact is used for reading the operation status. Avoid writing by a user program.
(Note 2) If sending time is shorter than scan time, the general-purpose communication sending flags (Y8, Y9, YA, YB) are turned off when the GPSEND instruction is executed in the subsequent scan following completion of data sending. The flags remain ON for at least one scan time.

■ Precautions during programming (in the case of SCU)

- Using the UNITSEL instruction immediately before the GPSEND instruction, specify a port targeted in communication.
- If no end code is to be added, specify a negative value for the sent data amount of the GPSEND instruction. Also, select "SS" for the operation unit.
- A start code and an end code specified by the configuration menu are automatically added to the data to be sent. Do not include a start code or an end code in the sent data.
- The maximum volume of data that can be sent is 4,096 bytes. If a start code is enabled, the max. volume is 4,096 bytes including the start code and the end code.
- It is also possible to send binary data.

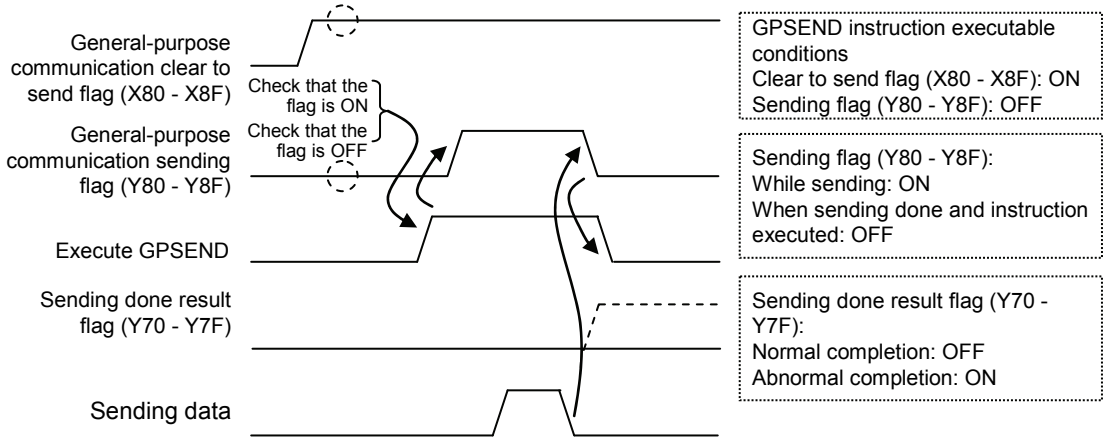
■ Sample program (in the case of CPU with built-in ET-LAN)

- Check that Connection 1 is established in the general-purpose communication mode (X80), and that general-purpose sending is not in progress in the same port (Y80), and start up the sending program.
- Using the SSET instruction, convert a given message into an ASCII string. Set the no. of sent strings to the data register DT100, and the sent message to the data register DT101.
- Using the UNITSEL instruction, specify the slot no. (LAN port: U100) and the connection no. (U1).
- In the GPSEND instruction, specify and execute the start of the table that stores the message to be sent (DT101) and the no. of characters in the data (DT100).



■ Time chart (in the case of CPU with built-in ET-LAN)

- Data are sent in ascending order from lower bytes of the table specified by the GPSEND instruction.
- During sending, the general-purpose communication sending flags that correspond to the connection (Y80 to Y8F) are turned on. Turns off when sending is completed.
- The sending result (0: normal completion; 1: abnormal completion) is stored in the general-purpose communication sending done result flags (Y70 to Y7F).



■ I/O allocation (in the case of CPU with built-in ET-LAN)

I/O number	Name	Explanation
X80 to X8F	General-purpose communication clear to send flag	Turns on when general-purpose communication is in a connected status.
Y80 to Y8F	General-purpose communication sending flag	Turns on during sending by the GPSEND instruction. Turns off when the GPSEND instruction is executed in the subsequent session, following completion of sending.
Y70 to Y7F	Sending done result flag	Completion result of sending by general-purpose communication / master communication is notified. (Normal completion: 0; Abnormal completion: 1)

(Note 1) Each contact is used for reading the operation status. Avoid writing by a user program.

■ Precautions during programming (in the case of CPU with built-in ET-LAN)

- Using the UNITSEL instruction immediately before the GPSEND instruction, specify a connection targeted in communication.
- No start code or an end code is added to data sent from the FP7 CPU unit. If it is necessary to send a start code and an end code, in accordance with protocol of an external device, store them as part of the data to be sent.
- The max. volume of data that can be sent in one session using the GPSEND instruction, from the LAN port of the FP7 CPU unit, is 16,384 bytes.

◆ KEY POINTS

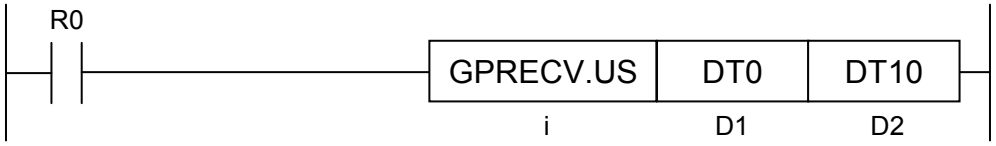
- **As the communication cassette (Ethernet type) has an Ethernet-serial conversion function, the internal interface operates with similar programs as the case of SCU. The setting method and programming method are different from those for the CPU with built-in ET-LAN. Refer to the section describing the case of SCU.**

■ Flag operations

Name	Explanation
SR7 SR8 (ER)	To be set in the case of out-of-range in indirect access (index modification).
	Communication is not possible in the connection specified by UNITSEL.
	The communication mode in the communication port specified by UNITSEL is not "general-purpose communication".
	Data device specified by [S] exceeds the area.
	Sent data amount specified by [n] is 0. The volume including a start code and an end code exceeds the specified max. value.
	The sent data amount specified by [n] exceeds the data area.
	Either 0 or a negative value is set for [N] in the settings of sending to ET-LAN.

GPRECV (General-Purpose Communication Receive Instruction)

■ Ladder diagram



■ Available operation units (●: Available)

Operation unit	bit	US	SS	UL	SL	SF	DF
i		●	●				

■ List of operands

Operand	Explanation
D1	Starting address of the received data storage data area (Always 16-bit device of word data, regardless of the operation unit [i])
D2	Ending address of the received data storage data area (Always 16-bit device of word data, regardless of the operation unit [i])

■ Available devices (●: Available)

Operand	16-bit device											32-bit device			Integers			Real numbers		String	Index modifier *1
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX *1	K	U	H	SF	DF	" "	
D1(*1)	●	●	●	●			●	●													●
D2(*1)	●	●	●	●			●	●													●

*1: Always 16-bit data/device, regardless of the specification of operation unit [i].

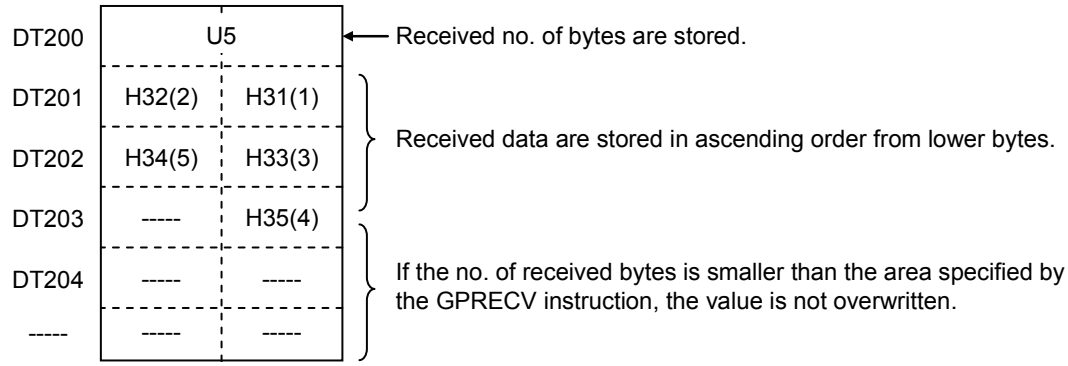
■ Outline of operation

- Read received data from an external device to the communication port of the unit.
- Read received data from the communication unit and the communication port set by the UNITSEL instruction, and store the no. of received bytes in the area specified by [D1], and the received data in the areas [D1+1] to [D2].
- In the case of SCU, data received from the partner are stored in 8 receive buffers for each COM port. By executing the GPRECV instruction, data in the receive buffer can be copied to a given operation memory.
- In the case of ET-LAN, data received from the partner are stored in 1 receive buffer for each connection. By executing the GPRECV instruction, data in the receive buffer can be copied to a given operation memory.

- No. of received data and end code

Items	In the case of SCU	In the case of CPU with built-in ET-LAN
No. of received data	0 to 4096 (with no end code)	0 to 16384
	0 to 4094, 4095 (with an end code, depending on the end settings) Note that the end code is excluded from the receive buffer and the received data amount.	
End code identification	Yes (according to the SCU communication settings (end settings))	No discrimination

■ Storage method for received data

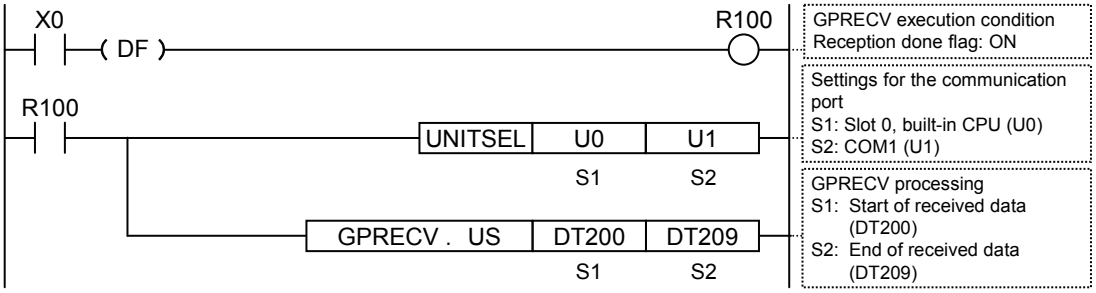


◆ KEY POINTS

- **The case of SCU shows the case that it is used in the following combination.**
  - **- COM0 port equipped in the CPU unit**
  - **- Communication cassettes attached to the CPU unit (COM.1 to COM.2 ports)**
  - **- Communication cassettes attached to the serial communication unit (COM.1 to COM.4 ports)**
- **As the communication cassette (Ethernet type) has an Ethernet-serial conversion function, the internal interface operates with similar programs as the case of SCU. The setting method and programming method are different from those for the CPU with built-in ET-LAN.**

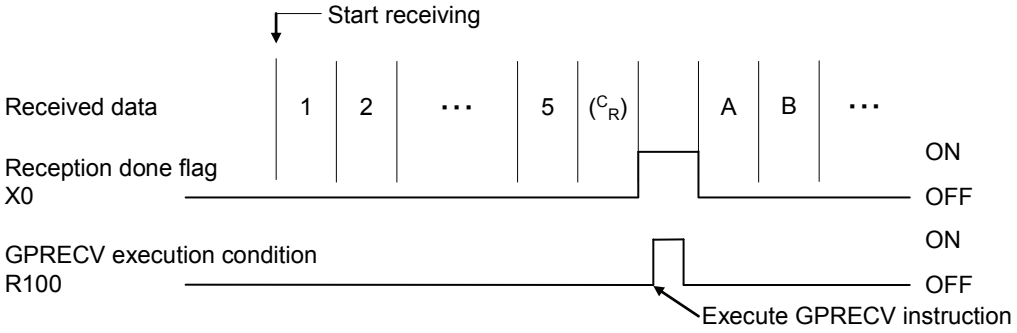
■ Sample program (in the case of SCU)

- When the reception done flag (X0) turns on, the reception program is started up by the GPRECV instruction.
- Using the UNITSEL instruction, specify the slot no. (U0) and the COM. port no. (U1).
- In the GPRECV instruction, specify and execute the start of the data table that stores the received message (DT200) and the final address (DT209).

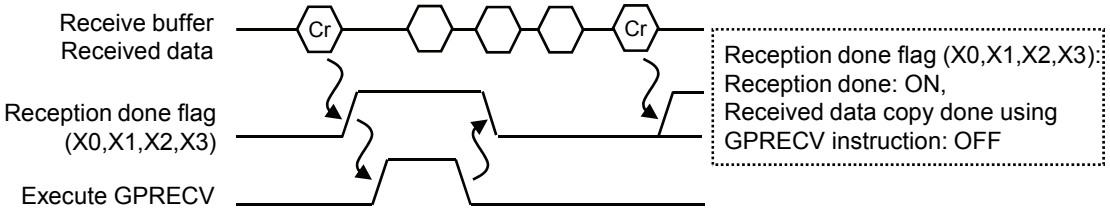


■ Time chart (in the case of SCU)

- Data received from an external device are stored in the receive buffer.
- When the end code is received, the reception done flag (X0, X1, X2, X3) turns on. Subsequently, the following data are stored in the buffers upon reception. Eight data can be received consecutively.



- When the GPRECV instruction is executed, data are copied to the specified area, and the reception done flags (X0, X1, X2, X3) are turned off. The reception done flags (X0, X1, X2, X3) are turned off when I/O refresh is executed at the start of the following scans.



■ I/O allocation (in the case of CPU unit)

COM Port No.			Name	Explanation
1	2	0		
X0	X1	X2	General-purpose communication reception done flag	Turns on when receiving is completed in the general-purpose communication mode.
X4	X5	X6	General-purpose communication reception copy done flag	When the GPRECV instruction is executed, and the received data have been copied to the specified operation memory, this flag turns on. Turns off when there are no data.

■ I/O allocation (in the case of Serial Communication Unit)

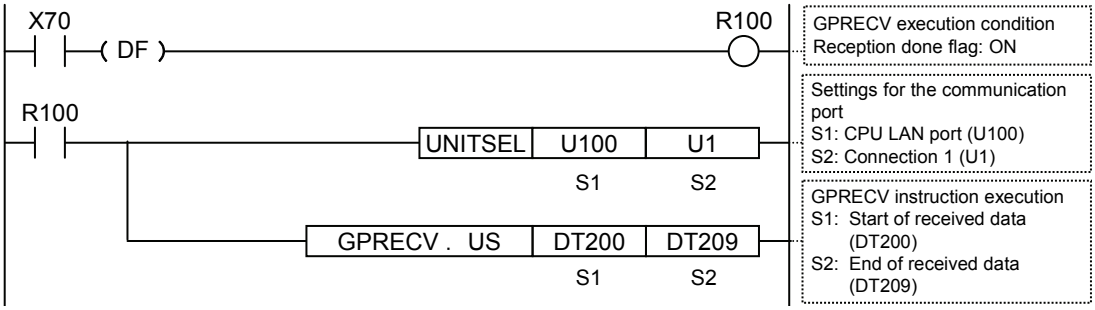
COM Port No.				Name	Explanation
1	2	3	4		
X0	X1	X2	X3	General-purpose communication reception done flag	Turns on when receiving is completed in the general-purpose communication mode.
X4	X5	X6	X7	General-purpose communication reception copy done flag	When the GPRECV instruction is executed, and the received data have been copied to the specified operation memory, this flag turns on. Turns off when there are no data.

■ Precautions during programming (in the case of SCU)

- Using the UNITSEL instruction immediately before the GPRECV instruction, specify a port targeted in communication.
- When the general-purpose communication reception done flag is on for the targeted COM port, execute GPRECV.
- When multiplex reception is in progress, the reception done flag remains on after the received data have been copied using the GPRECV instruction. Therefore, the received data cannot be copied at the start of the reception done signal.
- The received data copied by the GPRECV instruction do not include a start code or an end code.
- It is also possible to receive binary data using the GPRECV instruction. In this case, "time" should be used for the end setting.
- The received data or the received data amount do not include the end code. (It is stripped off.)
- In the case of SCU, eight data can be received consecutively, because it has eight 4096-byte buffers inside.
- If the ninth datum is received by SCU before this GPRECV instruction is executed to take out data from SCU's receive buffer, a buffer FULL error occurs in SCU, and the ninth datum is discarded.
- If the GPRECV instruction is executed when the receive buffer FULL error is on, the oldest received datum is taken out, and the receive buffer FULL error is canceled.
- When no data have been received, the general-purpose communication control flag (reception done copy flag) turns off.
- After data have been received, and copy to the operation memory of the CPU unit has been completed, the general-purpose communication control flag (reception done copy flag) turns on.
- In the case of a direct address and an index modification address, specify the same type of device for D1 and D2. At the same time, specify D2 ≥ D1.

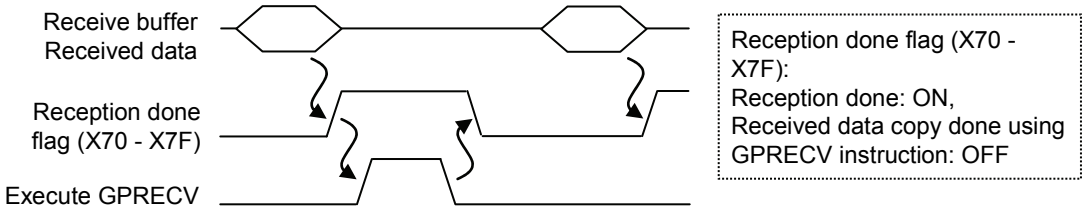
■ Sample program (in the case of CPU with built-in ET-LAN)

- When the reception done flag (X70) of Connection 1 turns on, the reception program is started up by the GPRECV instruction.
- Using the UNITSEL instruction, specify the slot no. (LAN port: U100) and the connection no. (U1).
- In the GPRECV instruction, specify and execute the start of the data table that stores the received message (DT200) and the final address (DT209).



■ Time chart (in the case of CPU with built-in ET-LAN)

- Data received from an external device are stored in the receive buffer for each connection.
- When data are received, the reception done flag (X70 to X7F) turns on.
- When the GPRECV instruction is executed, data are copied to the specified area, and the reception done flags (X70 to X7F) are turned off. The reception done flags (X70 to X7F) are turned off when I/O refresh is executed at the start of the following scans.



■ I/O allocation (in the case of CPU with built-in ET-LAN)

I/O allocation	Name	Name
X70 to X7F	General-purpose communication reception done flag	Turns on when receiving is completed in the general-purpose communication mode.

■ Precautions during programming (in the case of CPU with built-in ET-LAN)

- Using the UNITSEL instruction immediately before the GPRECV instruction, specify a connection targeted in communication.
- When the general-purpose communication reception done flag is on for the targeted connection, execute GPRECV.
- The max. volume of data that can be received in one session using the GPRECV instruction, from the LAN port of the FP7 CPU unit, is 16,384 bytes.
- If a start code and an end code are included in the communication format of the external device, they are stored in the operation memory as part of the received data. If necessary, insert a program for data extraction.
- In the case of a direct address and an index modification address, specify the same type of device for D1 and D2. At the same time, specify D2 ≥ D1.

◆ KEY POINTS

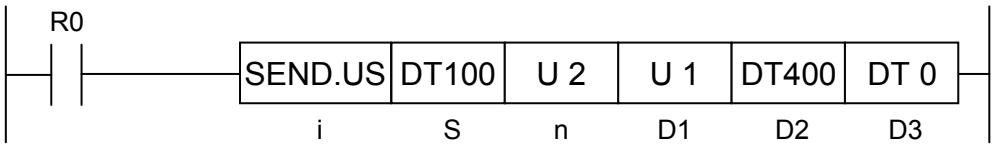
- **As the communication cassette (Ethernet type) has an Ethernet-serial conversion function, the internal interface operates with similar programs as the case of SCU. The setting method and programming method are different from those for the CPU with built-in ET-LAN. Refer to the section describing the case of SCU.**

■ Flag operations

Name	Explanation
SR7 SR8 (ER)	To be set in the case of out-of-range in indirect access (index modification).
	To be set when the destination range is out of the accessible range.
	SCU or ET-LAN unit does not exist in the slot specified by UNITSEL.
	The communication mode in the communication port specified by UNITSEL is not "general-purpose communication".
	COM port specified by UNITSEL does not exist.
	Connection specified by UNITSEL is in a "reception done OFF" status, but not in a "connected" status.
	Data device specified by [D1] and/or [D2] exceeds the area.
	When [D1] ≥ [D2] is specified.
	When the [D1] device and the [D2] device differ.

SEND (MEWTOCOL Master / MODBUS Master)

■ Ladder diagram



■ Available operation units (●: Available)

Operation unit	bit	US	SS	UL	SL	SF	DF
i		●	●				

■ List of operands

Operand	Explanation
S	Starting address of the source data area
n	Sent data amount
D1	Partner station no.
D2	Starting address of the receiver area in the partner unit
D3	Starting address of the device area in the master unit that stores the execution result code (1 word)

■ Available word devices (●: Available)

Operand	16-bit device											32-bit device			Integers			Real numbers		String	Index modifier
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX	K	U	H	SF	DF	" "	
S	●	●	●	●			●	●													●
N	●	●	●	●			●	●								●	●				●
D1	●	●	●	●			●	●								●	●				●
D2(*1)	*2	●	●	*2			●	*2													●
D3	●	●	●	●			●	●													●

*1: When the receiver is FP7, only a global device can be specified. (A local device cannot be specified.)
*2: In the MODBUS mode, this cannot be specified as the receiver. (Word device or bit device cannot be specified.)

■ Available bit devices (●: Available)

Operand	Bit device											Bit specification of the word		Index modifier
	X	Y	R	L	T	C	P	E	S	IN	OT	DT.b	LD.b	
S	●	●	●	●								●	●	●
n														
D1														
D2*1	*2	●	●	*2										●
D3														

*1 When the receiver is FP7, only a global device can be specified.
(A local device cannot be specified.)
*2 In the case of MODBUS mode, a bit device cannot be specified.

■ Outline of operation

- Commands are sent from the communication port of the unit to perform the data transmission with external devices.
  Message in accordance with the protocol is automatically formulated by PLC. The user program only has to specify the station no. and the memory address, and execute the SEND/RECV instruction, to carry out reading and writing.
- Communication mode should be selected in the configuration menu of the tool software FPWIN GR7.
- When the SEND instruction is executed, data are read from the device in the master unit, starting with [S], and stored in the address starting with [D2] of the partner unit.

- Depending on the type of device specified by [S] and [D2], the transfer method (register transfer / bit transfer) varies.
- Sent data amount [n] is provided in words when register transfer is selected, and in the no. of words or bits when bit transfer is selectede.
- The execution result code is stored in 1 word of area within the master unit specified by [D3].

■ Setting of sent data amount [n]

Types of sending	Communication mode	Sent data amount n	Note
Register sending	MEWTOCOL-COM	1 to 507 words	
	MEWTOCOL-DAT	1 to 1020 words	Connection setting: Setting of the MEWTOCOL communication type: Connect with FP2 ET-LAN
		1 to 2038 words	Connection setting: Setting of the MEWTOCOL communication type: Do not connect with FP2 ET-LAN
	MODBUS	1 to 127 words	MODBUS Command 15 (WY and WR writing) and Command 16 (DT multiple word writing) are used.
Bit sending	MEWTOCOL-COM	Fixed to 1 bit	During MEWTOCOL-COM, WCS command is used.
	MEWTOCOL-DAT	Fixed to 1 bit	During MEWTOCOL-DAT, contact information write 52H is used.
	MODBUS	1 to 2040	Use the force multiple coils command 15.

(Note 1) Transmission methods vary by the type of device to be specified for the operands [S] and [D2]. Register transfer is selected for a 16-bit device, and bit transfer for a 1-bit device.
(Note 2) Sent data amount is provided in words when register transfer is selected, and in bits when bit transfer is selected.

■ Specification of partner station no. [D1]

Communication mode	When SCU is used	When ET-LAN is used
MEWTOCOL-COM	0 to 99	1 to 64
MEWTOCOL-DAT	Non-SCU-compliant	
MODBUS	0 to 247	1 to 247

(Note 1) When "0" is specified for the partner station no., global transfer is selected. At this time, there is no response message from the partner.
(Note 2) For connection between FP7 and FP7, specify "1". Destination is determined by the IP address.

■ Specification of receiver address [D2]

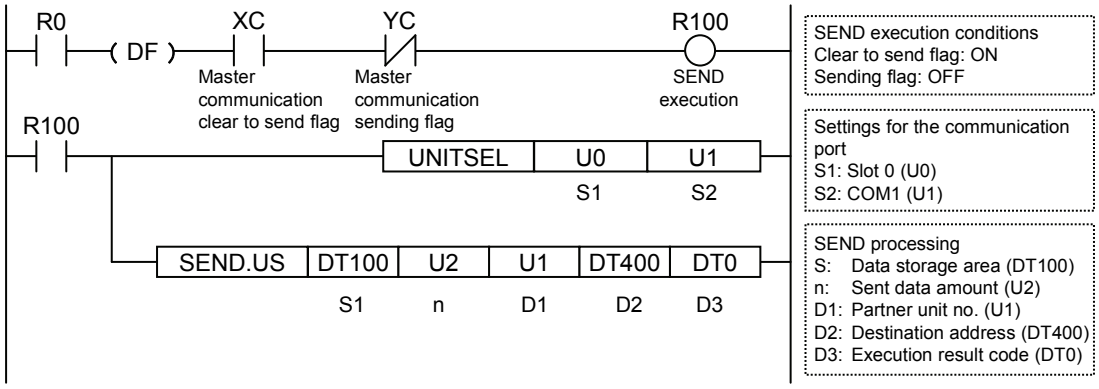
Communication mode	Address range
MEWTOCOL-COM	0 to 9999
MEWTOCOL-DAT	0 to 65535 (H FFFF)
MODBUS	0 to 65535 (H FFFF)

■ Execution result code [D3]

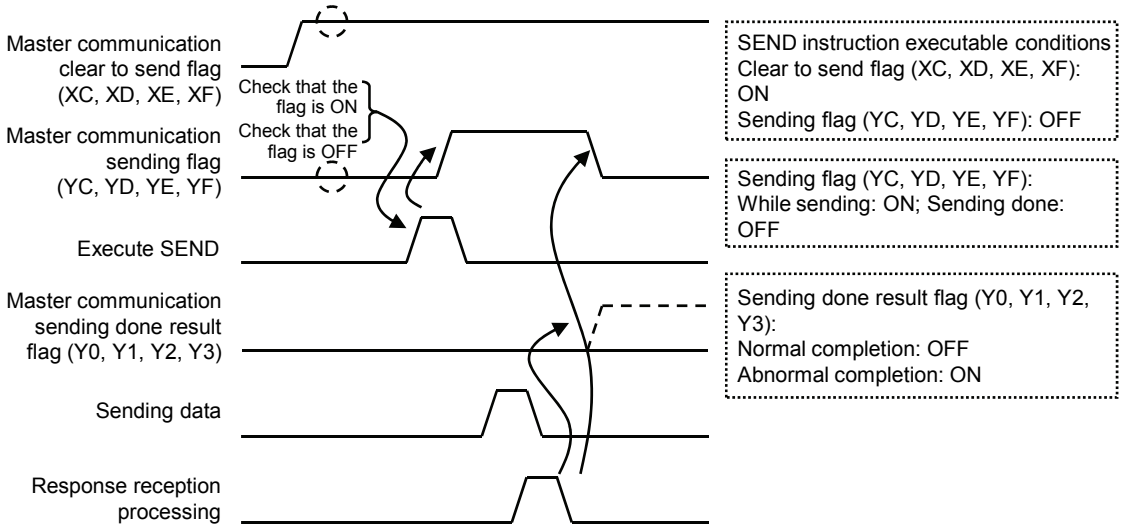
Execution result	Execution result code
Normal completion	0
Communication port is being used for master communication.	1
Communication port is being used for slave communication.	2
No. of master communication instructions that can be used at the same time is exceeded.	3
Sending timeout	4
Response reception timeout	5
Received data error	6

■ Sample program (in the case of SCU)

- Send the command from the COM1 port of the CPU unit, and write the content, of PLC's data registers DT100 to DT101, into the data areas DT400 to DT401 of the external device (Station No. 1).
- Check that the master mode is on (XC), and that sending is not in progress in the same port (YC), and start up the SEND instruction.
- Using the UNITSEL instruction, specify the slot no. (U0) and the COM. port no. (U1).
- In the SEND instruction, specify and execute the destination's starting address (DT100) and data amount (U2), the destination's station no. (U1) and starting address (DT400).



■ Time chart (in the case of SCU)



◆ KEY POINTS

- **The case of SCU shows the case that it is used in the following combination.**
  - **- COM0 port equipped in the CPU unit**
  - **- Communication cassettes attached to the CPU unit (COM.1 to COM.2 ports)**
  - **- Communication cassettes attached to the serial communication unit (COM.1 to COM.4 ports)**
- **As the communication cassette (Ethernet type) has an Ethernet-serial conversion function, the internal interface operates with similar programs as the case of SCU. The setting method and programming method are different from those for the CPU with built-in ET-LAN. The communication cassette (Ethernet type) does not support MODBUS.**

■ I/O allocation (in the case of CPU with built-in SCU)

COM Port No.			Name	Explanation
1	2	0		
XC	XD	XE	Master communication clear to send flag	Turns on when MEWTOCOL-COM, MEWTOCOL7, and MODBUS-RTU are set in the communication mode, and the RUN mode is on.
YC	YD	YE	Master communication sending flag	Turns on during sending by the SEND and RECV instructions. Turns off when sending is completed.
Y0	Y1	Y2	Sending done result flag	Completion result of sending by general-purpose communication / master communication is notified. (Normal completion: 0; Abnormal completion: 1)

(Note 1) Each contact is used for reading the operation status. Avoid writing by a user program.

■ I/O allocation (in the case of Serial Communication Unit)

COM Port No.				Name	Explanation
1	2	3	4		
XC	XD	XE	XF	Master communication clear to send flag	Turns on when MEWTOCOL-COM, MEWTOCOL7, and MODBUS-RTU are set in the communication mode, and the RUN mode is on.
YC	YD	YE	YF	Master communication sending flag	Turns on during sending by the SEND and RECV instructions. Turns off when sending is completed.
Y0	Y1	Y2	Y3	Sending done result flag	Completion result of sending by general-purpose communication / master communication is notified. (Normal completion: 0; Abnormal completion: 1)

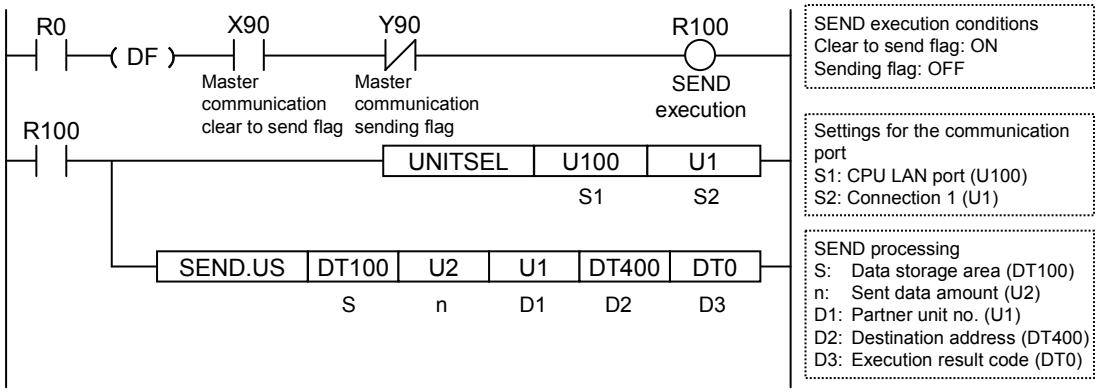
(Note 1) Each contact is used for reading the operation status. Avoid writing by a user program.

■ Precautions during programming (in the case of SCU)

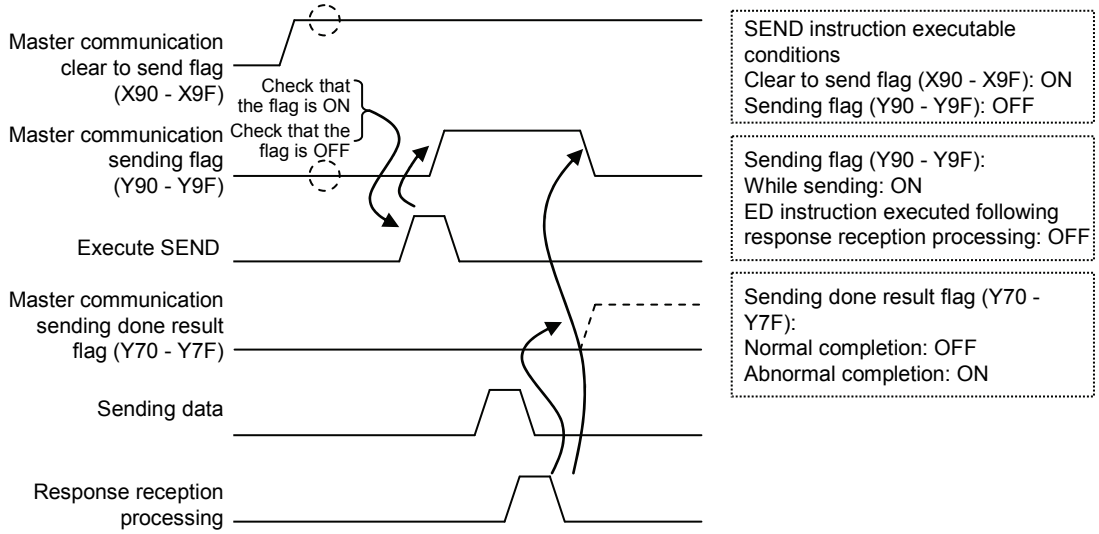
- Using the UNITSEL instruction immediately before the SEND/RECV instruction, specify a port targeted in communication.
- Master communication is only enabled when MEWTOCOL or MODBUS is selected. Check that the master communication clear to send flags (XC to XF) are on for the corresponding channel, and execute the SEND/RECV instruction.
- Another SEND/RECV instruction cannot be executed for a communication port where master communication is in progress. Check that the master communication sending flags (YC to YF) are off, and execute the instruction.
- A SEND/RECV instruction cannot be executed for a port where slave communication is in progress.
- If there is no response, the master communication sending flags (YC to YF) remain ON during the time-out period set in the CPU configuration.
- Up to 16 SEND/RECV instructions can be executed simultaneously for differing COM. ports.

■ Sample program (in the case of CPU with built-in ET-LAN)

- Send the command from the LAN port of the CPU unit, and write the content, of PLC's data registers DT100 to DT101, into the data areas DT400 to DT401 of the external device.
- Check that Connection 1 is established in the master mode (X90), and that sending is not in progress in the same port (Y90), and start up the SEND instruction.
- Using the UNITSEL instruction, specify the slot no. (LAN port: U100) and the connection no. (U1).
- In the SEND instruction, specify and execute the destination's starting address (DT100) and data amount (U2), the destination's station no. (U1) and starting address (DT400).



■ Time chart (in the case of CPU with built-in ET-LAN)



■ I/O allocation (in the case of CPU with built-in ET-LAN)

I/O number	Name	Explanation
X90 to X9F	Master communication clear to send flag	Turns on when master communication is in a connected status.
Y90 to Y9F	Master communication sending flag	Turns on during sending by the SEND and RECV instructions. Turns off when the ED instruction is executed following completion of response reception processing.
Y70 to Y7F	Sending done result flag	Completion result of sending by general-purpose communication / master communication is notified. (Normal completion: 0; Abnormal completion: 1)

(Note 1) Each contact is used for reading the operation status. Avoid writing by a user program.

■ Precautions during programming (in the case of CPU with built-in ET-LAN)

- Using the UNITSEL instruction immediately before the SEND/RECV instruction, specify a connection no. targeted in communication.
- Master communication is only enabled when MEWTOCOL or MODBUS is selected. Check that the master communication clear to send flags (X90 to X9F) are on for the corresponding connection, and execute the SEND/RECV instruction.
- Another SEND/RECV instruction cannot be executed for a connection where master communication is in progress. Check that the master communication sending flags (Y90 to Y9F) are off, and execute the instruction.
- A SEND/RECV instruction cannot be executed for a connection where slave communication is in progress.
- Up to 16 SEND/RECV instructions can be executed simultaneously for differing connections.
- For communication between LAN ports of FP7, specify "U1" for the partner station no. Receiver is determined by the IP address.

◆ KEY POINTS

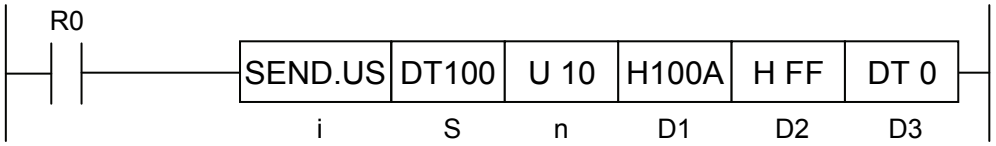
- **As the communication cassette (Ethernet type) has an Ethernet-serial conversion function, the internal interface operates with similar programs as the case of SCU. The setting method and programming method are different from those for the CPU with built-in ET-LAN. Refer to the section describing the case of SCU.**

■ Flag operations

Name	Explanation
SR7 SR8 (ER)	To be set in the case of out-of-range in indirect access (index modification).
	To be set when the sender range is out of the accessible range.
	COM port or connection specified by UNITSEL does not exist, or communication is not possible in the specified connection.
	Data device specified by [S] is invalid, or exceeds the area.
	Sent data amount specified by [n] is invalid.
	Station no. specified by [D1] is out of the range.
	Data device specified by [D2] is invalid, or exceeds the area.
	Result storage device specified by [D3] is invalid.
	Integer specification for [D2] is only available for the MODBUS direct address specification type, and invalid for other types.
	Specified bit devices for [S] and [D2], and/or specified16-bit device, differ.

SEND (MODBUS Master: Function Code Specification)

■ Ladder diagram



■ Available operation units (●: Available)

Operation unit	bit	US	SS	UL	SL	SF	DF
i		●	●				

■ List of operands

Items	Settings			Setting range
i	Specify the operation unit.			US / SS
S	Specify the start position of the sender data area.			-
n	Specify the sent data amount.			1 to 127 words 1 to 2040 bits
D1	Specify the MODBUS command to be used, and the partner station no.			
		Higher byte	Two hexadecimal digits that indicate the MODBUS function code.	H5, H6, HF, H10
		Lower byte	Two hexadecimal digits that indicate the station no.	H0 to HF7 (0 to 247)
D2	Specify the start position of the MODBUS address for the receiver data area in the partner unit.			H0~HFFFF (0 to 65535)
D3	Specify the device area in the master unit that stores the execution result code (1 word).			-

■ Available word devices (●: Available)

Operand	16-bit device											32-bit device			Integers			Real numbers		String	Index modifier
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX	K	U	H	SF	DF	" "	
S	●	●	●	●			●	●													●
n	●	●	●	●			●	●								●	●				●
D1	●	●	●	●			●	●								●	●				●
D2	●	●	●	●			●	●								●	*1				●
D3	●	●	●	●			●	●													●

*1: When the receiver is FP7, only a global device can be specified. (A local device cannot be specified.)

■ Available bit devices (●: Available)

Operand	Bit device											Bit specification of the word		Index modifier
	X	Y	R	L	T	C	P	E	S	IN	OT	DT.b	LD.b	
S	●	●	●	●								●	●	●
n														
D1														
D2 *1														
D3														

■ Outline of operation

- The MODBUS command is sent from the communication port of the unit to perform the data transmission with external devices.
- Message in accordance with the protocol is automatically formulated by PLC. The user program only has to specify the station no. and the memory address, and execute the SEND/RECV instruction, to carry out reading and writing.
- Communication mode should be selected in the configuration menu of the tool software FPWIN GR7.
- Specify the MODBUS command to be used, and the partner MODBUS station no., in a Hex format in [D1].
- When the SEND instruction is executed, data are read from the device in the master unit, starting with [S], and stored in the address starting with [D2] of the partner unit.
- Depending on the type of device specified by [S], and data amount specified by [n], the transfer method (register transfer / bit transfer), and the type of MODBUS command that can be used, vary.
- Sent data amount [n] is provided in words when register transfer is selected, and in the no. of words or bits when bit transfer is selected.
- The execution result code is stored in 1 word of area within the master unit specified by [D3].

■ Specification of [S], [n] and [D1]

- Depending on the type of device specified by the operand [S], and sent data amount specified by [n], the transfer method, and the MODBUS function code that can be used, vary.

Types of device to be specified for [S]	Transfer method	Sent data amount [n]	Value that can be specified for higher bytes of [D1]
16-bit device WX, WY, WR, WL, DT, LD	Register sending	1	H6: Preset single register (06) HF: Force multiple coils (15) H10: Preset multiple registers (16)
		2 to 127	HF: Force multiple coils (15) H10: Preset multiple registers (16)
1-bit device X, Y, R, L, DT, n, LD, n	Bit sending	1	H5: Force single coil (05) HF: Force multiple coils (15)
		2 to 2040	HF: Force multiple coils (15)

- Sent data amount [n] is provided in words when register transfer is selected, and in bits when bit transfer is selected.
- Operand [D1] is specified as a combination of a two-digit hexadecimal MODBUS function code and a two-digit hexadecimal partner station no.
  Example: Specify "H100A" in the case of MODBUS function code 16 (preset multiple registers) and station No. 10.
- In the case of SCU, when "0" is specified for the partner station no., global transfer is selected. At this time, there is no response message from the partner.

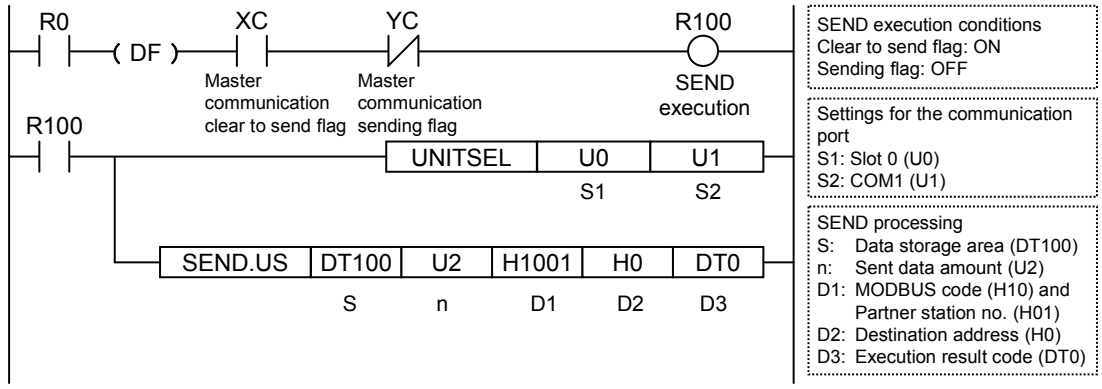
■ Execution result code [D3]

Execution result	Execution result code
Normal completion	0
Communication port is being used for master communication.	1
Communication port is being used for slave communication.	2
No. of master communication instructions that can be used at the same time is exceeded.	3
Sending timeout	4
Response reception timeout	5
Received data error	6

(Note) Stored data amount: 1 word

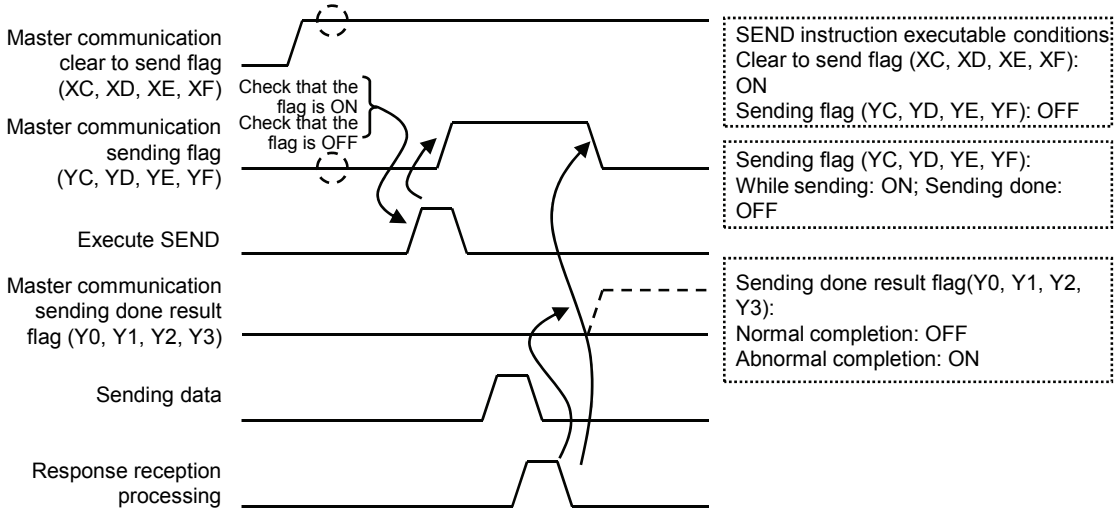
■ Sample program (in the case of SCU)

- Send the command from the COM1 port of the CPU unit, and write the content, of PLC's data registers DT100 to DT101, into the data areas 40001 to 40002 of the external device (Station No. 1).
- Check that the master mode is on (XC), and that sending is not in progress in the same port (YC), and start up the SEND instruction.
- Using the UNITSEL instruction, specify the slot no. (U0) and the COM. port no. (U1).
- In the SEND instruction, specify and execute PLC's starting address (DT100) and data amount (U2), MODBUS function code to be used (16 : H10), and partner station no. (H01) and starting address (H0). See operating instructions of partner devices for their address setting.



(Note 1) Operand [D1] of SEND instruction is specified by combining two hexadecimal digits of MODBUS function code with two hexadecimal digits of partner device station no. When the MODBUS function code is 16, [D1] H10 should be specified.
(Note 2) When the partner device is FP series PLC, Operand [D2] of SEND instruction can be specified using the Device No.

■ Time chart (in the case of SCU)



◆ KEY POINTS

- **The case of SCU shows the case that it is used in the following combination.**
  - **- COM0 port equipped in the CPU unit**
  - **- Communication cassettes attached to the CPU unit (COM.1 to COM.2 ports)**
  - **- Communication cassettes attached to the serial communication unit (COM.1 to COM.4 ports)**
- **The communication cassette (Ethernet type) does not support MODBUS.**

■ I/O allocation (in the case of CPU with built-in SCU)

COM Port No.			Name	Explanation
1	2	0		
XC	XD	XE	Master communication clear to send flag	Turns on when MEWTOCOL-COM, MEWTOCOL7, and MODBUS-RTU are set in the communication mode, and the RUN mode is on.
YC	YD	YE	Master communication sending flag	Turns on during sending by the SEND and RECV instructions. Turns off when sending is completed.
Y0	Y1	Y2	Sending done result flag	Completion result of sending by general-purpose communication / master communication is notified. (Normal completion: 0; Abnormal completion: 1)

(Note 1) Each contact is used for reading the operation status. Avoid writing by a user program.

■ I/O allocation (in the case of Serial Communication Unit)

COM Port No.				Name	Explanation
1	2	3	4		
XC	XD	XE	XF	Master communication clear to send flag	Turns on when MEWTOCOL-COM, MEWTOCOL7, and MODBUS-RTU are set in the communication mode, and the RUN mode is on.
YC	YD	YE	YF	Master communication sending flag	Turns on during sending by the SEND and RECV instructions. Turns off when sending is completed.
Y0	Y1	Y2	Y3	Sending done result flag	Completion result of sending by general-purpose communication / master communication is notified. (Normal completion: 0; Abnormal completion: 1)

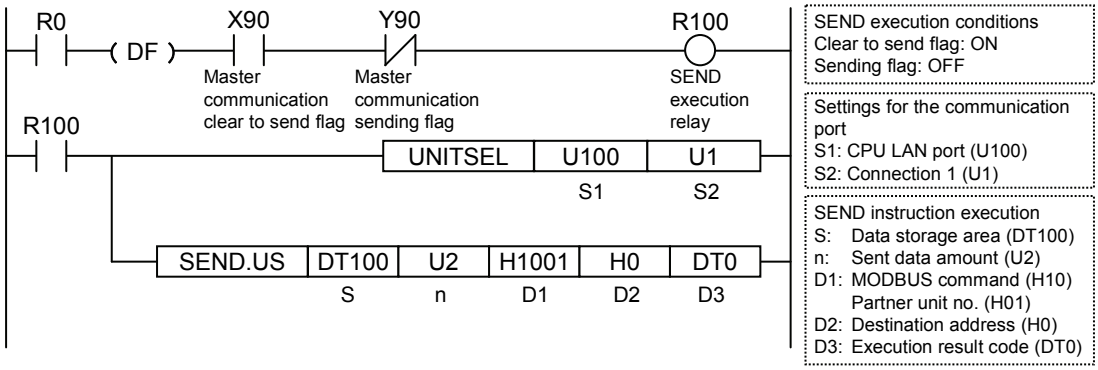
(Note 1) Each contact is used for reading the operation status. Avoid writing by a user program.

■ Precautions during programming (in the case of SCU)

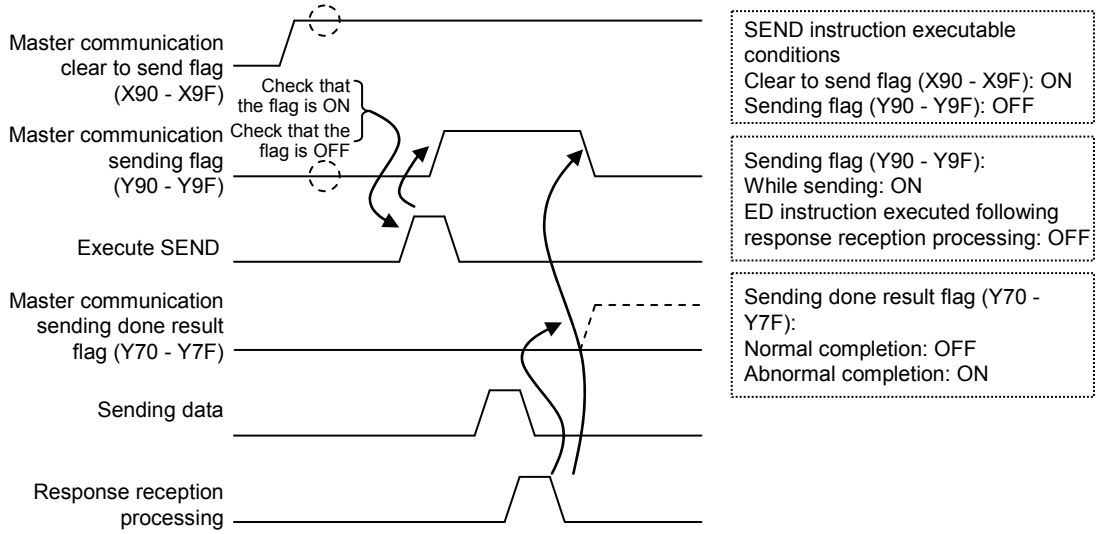
- Using the UNITSEL instruction immediately before the SEND/RECV instruction, specify a port targeted in communication.
- Master communication is only enabled when MEWTOCOL or MODBUS is selected. Check that the master communication clear to send flags (XC to XF) are on for the corresponding channel, and execute the SEND/RECV instruction.
- Another SEND/RECV instruction cannot be executed for a communication port where master communication is in progress. Check that the master communication sending flags (YC to YF) are off, and execute the instruction.
- A SEND/RECV instruction cannot be executed for a port where slave communication is in progress.
- If there is no response, the master communication sending flags (YC to YF) remain ON during the time-out period set in the CPU configuration.
- Up to 16 SEND/RECV instructions can be executed simultaneously for differing COM. ports.

■ Sample program (in the case of CPU with built-in ET-LAN)

- Send the MODBUS command (16) from the LAN port of the CPU unit, and write the content, of PLC's data registers DT100 to DT101, into the data areas 40001 to 40002 of the external device (MODBUS addresses 0000H to 0001H).
- Check that Connection 1 is established in the master mode (X90), and that sending is not in progress in the same port (Y90), and start up the SEND instruction.
- Using the UNITSEL instruction, specify the slot no. (LAN port: U100) and the connection no. (U1).
- In the SEND instruction, specify and execute PLC's starting address (DT100) and data amount (U2), MODBUS command (16 = H10), and partner station no. (H01) and starting address (H0). See operating instructions of partner devices for their address setting.



■ Time chart



■ I/O allocation

I/O number	Name	Explanation
X90 to X9F	Master communication clear to send flag	Turns on when master communication is in a connected status.
Y90 to Y9F	Master communication sending flag	Turns on during sending by the SEND and RECV instructions. Turns off when the ED instruction is executed following completion of response reception processing.
Y70 to Y7F	Sending done result flag	Completion result of sending by general-purpose communication / master communication is notified. (Normal completion: 0; Abnormal completion: 1)

(Note 1) Each contact is used for reading the operation status. Avoid writing by a user program.

■ Precautions during programming (in the case of CPU with built-in ET-LAN)

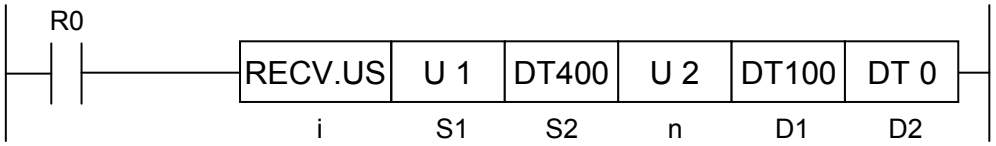
- Using the UNITSEL instruction immediately before the SEND/RECV instruction, specify a connection no. targeted in communication.
- Master communication is only enabled when MEWTOCOL or MODBUS is selected. Check that the master communication clear to send flags (X90 to X9F) are on for the corresponding connection, and execute the SEND/RECV instruction.
- Another SEND/RECV instruction cannot be executed for a connection where master communication is in progress. Check that the master communication sending flags (Y90 to Y9F) are off, and execute the instruction.
- A SEND/RECV instruction cannot be executed for a connection where slave communication is in progress.
- Up to 16 SEND/RECV instructions can be executed simultaneously for differing connections.
- In the MODBUS-TCP mode, specify the partner station no. as operand for the SEND/RECV instruction.

■ Flag operations

Name	Explanation
SR7 SR8 (ER)	To be set in the case of out-of-range in indirect access (index modification).
	To be set when the sender range is out of the accessible range.
	COM port or connection specified by UNITSEL does not exist, or communication is not possible in the specified connection.
	Data device specified by [S] is invalid, or exceeds the area.
	Sent data amount specified by [n] is invalid.
	MODBUS command and/or station no. specified by [D1] is invalid.
	Data device specified by [D2] is invalid, or exceeds the area.
	Result storage device specified by [D3] is invalid.
	Integer specification for [D2] is only available for the MODBUS address direct specification type, and invalid for other types.
	Result storage device specified by [D3] is invalid.

RECV (MEWTOCOL Master / MODBUS Master)

■ Ladder diagram



■ Available operation units (●: Available)

Operation unit	bit	US	SS	UL	SL	SF	DF
i		●	●				

■ List of operands

Operand	Explanation
S1	Partner station no.
S2	Device starting address of the source data area in the partner unit
n	No. of received data
D1	Starting address of the receiver area in the partner unit
D2	Starting address of the device area in the master unit that stores the execution result code (1 word)

■ Available word devices (●: Available)

Operand	16-bit device											32-bit device			Integers			Real numbers		String	Index modifier
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX	K	U	H	SF	DF	" "	
S1	●	●	●	●			●	●								●	●				●
S2(*1)	●	●	●	●			●	●													●
n	●	●	●	●			●	●								●	●				●
D1	●	●	●	●			●	●													●
D2	●	●	●	●			●	●													●

*1: When the sender is FP7, only a global device can be specified. (A local device cannot be specified.)

■ Available bit devices (●: Available)

Operand	Bit device											Bit specification of the word		Index modifier
	X	Y	R	L	T	C	P	E	S	IN	OT	DT.b	LD.b	
S1														
S2 *1	●	●	●	*2								*3	*3	●
n														
D1	●	●	●	●								●	●	●
D2														

*1 When the sender is FP7, only a global device can be specified. (A local device cannot be specified.)
*2 In the case of MODBUS mode, a bit device cannot be specified.
*3 In the case of MEWTOCOL-COM mode or MODBUS mode, a bit device cannot be specified.

■ Outline of operation

- Commands are sent from the communication port of the unit to perform the data transmission with external devices.
- Message in accordance with the protocol is automatically formulated by PLC. The user program only has to specify the station no. and the memory address, and execute the SEND/RECV instruction, to carry out reading and writing.
- Communication mode should be selected in the configuration menu of the tool software FPWIN GR7.
- When the RECV instruction is executed, data are read from the address starting with [S2] in the partner station no. [S1], and stored in the area starting with [D1] in the master unit.
- Depending on the type of device specified by [S2] and [D1], the transfer method (register transfer / bit transfer) varies.
- Sent data amount [n] is provided in words when register transfer is selected, and in the no. of words or bits when bit transfer is selected.
- The execution result code is stored in 1 word of area within the master unit specified by [D2].

■ Specification of partner station no. [S1]

Communication mode	When SCU is used	When ET-LAN is used
MEWTOCOL-COM	1 to 99	1 to 64 (Note)
MEWTOCOL-DAT (Note)	Non-SCU-compliant	
MODBUS	1 to 247	1 to 247

(Note) For connection between FP7 and FP7, specify "1". Destination is determined by the IP address.

■ Specification of starting address [S2] of the sender data area

Communication mode	Address range
MEWTOCOL-COM	0 to 9999
MEWTOCOL-DAT	0 to 65535 (H FFFF)
MODBUS	0 to 65535 (H FFFF)

■ Specification of received data amount [n]

Types of sending	Communication mode	Types of communication port	Setting range
Register sending *1	MEWTOCOL-COM	1 to 509 words	RCC command and RD command are used.
	MEWTOCOL-DAT	1 to 1020 words	Connection setting: Setting of the MEWTOCOL communication type Connect with FP2 ET-LAN
		1 to 2038 words	Connection setting: Setting of the MEWTOCOL communication type Do not connect with FP2 ET-LAN
	MODBUS	1 to 127 words	For reading WY and WR, use Command 1. For reading WX, use Command 2. For reading DT, use Command 3. For reading WL and LD, use Command 4.
Bit sending *2	MEWTOCOL-COM	Fixed to 1 bit	During MEWTOCOL-COM, RCS command is used.
	MEWTOCOL-DAT	Fixed to 1 bit	During MEWTOCOL-DAT, read contact information 53H is used.
	MODBUS	1 to 2040 bits	Command 1 is used for reading Y and R. Command 2 is used for X.

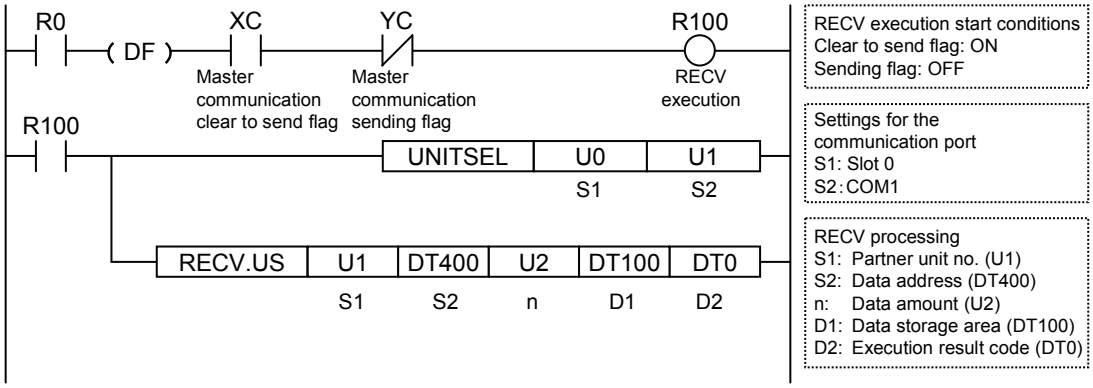
*1 When 16-bit devices are specified for sender [S] and receiver [D2].
*2 When bit devices are specified for sender [S] and receiver [D2].

■ Execution result code [D2]

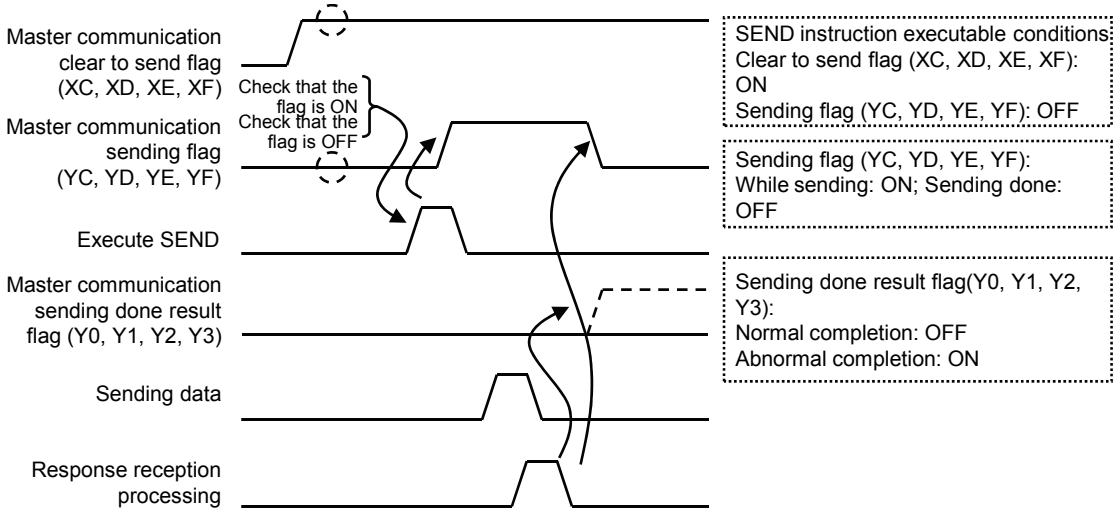
Execution result	Execution result code
Normal completion	0
Communication port is being used for master communication.	1
Communication port is being used for slave communication.	2
No. of master communication instructions that can be used at the same time is exceeded.	3
Sending timeout	4
Response reception timeout	5
Received data error	6

■ Sample program (in the case of SCU)

- Send the command from the COM1 port of the CPU unit, read data from the data areas DT400 to DT401 of the external device (Station No. 1), and write the data into the data registers DT100 to DT101 of the PLC.
- Check that the master mode is on (XC), and that sending is not in progress in the same port (YC), and start up the SEND instruction.
- Using the UNITSEL instruction, specify the slot no. (U0) and the COM. port no. (U1).
- In the RECV instruction, specify and execute the partner station no. (U1), starting address (DT400), data amount (U2), and PLC's starting address to store the data (DT100).



■ Time chart (in the case of SCU)



◆ **KEY POINTS**

- **The case of SCU shows the case that it is used in the following combination.**
  - **- COM0 port equipped in the CPU unit**
  - **- Communication cassettes attached to the CPU unit (COM.1 to COM.2 ports)**
  - **- Communication cassettes attached to the serial communication unit (COM.1 to COM.4 ports)**
- **As the communication cassette (Ethernet type) has an Ethernet-serial conversion function, the internal interface operates with similar programs as the case of SCU. The setting method and programming method are different from those for the CPU with built-in ET-LAN. The communication cassette (Ethernet type) does not support MODBUS.**

■ I/O allocation (in the case of CPU with built-in SCU)

COM Port No.			Name	Explanation
1	2	0		
XC	XD	XE	Master communication clear to send flag	Turns on when MEWTOCOL-COM, MEWTOCOL7, and MODBUS-RTU are set in the communication mode, and the RUN mode is on.
YC	YD	YE	Master communication sending flag	Turns on during sending by the SEND and RECV instructions. Turns off when sending is completed.
Y0	Y1	Y2	Sending done result flag	Completion result of sending by general-purpose communication / master communication is notified. (Normal completion: 0; Abnormal completion: 1)

(Note 1) Each contact is used for reading the operation status. Avoid writing by a user program.

■ I/O allocation (in the case of Serial Communication Unit)

COM Port No.				Name	Explanation
1	2	3	4		
XC	XD	XE	XF	Master communication clear to send flag	Turns on when MEWTOCOL-COM, MEWTOCOL7, and MODBUS-RTU are set in the communication mode, and the RUN mode is on.
YC	YD	YE	YF	Master communication sending flag	Turns on during sending by the SEND and RECV instructions. Turns off when sending is completed.
Y0	Y1	Y2	Y3	Sending done result flag	Completion result of sending by general-purpose communication / master communication is notified. (Normal completion: 0; Abnormal completion: 1)

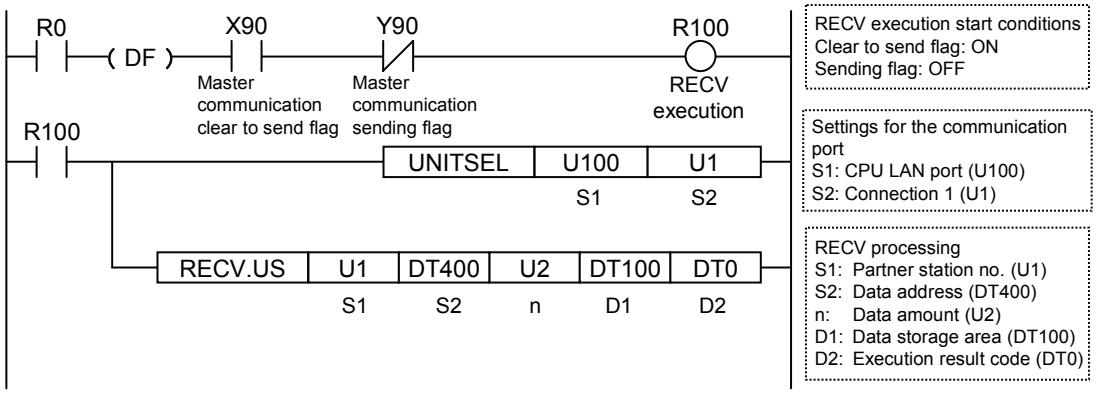
(Note 1) Each contact is used for reading the operation status. Avoid writing by a user program.

■ Precautions during programming (in the case of SCU)

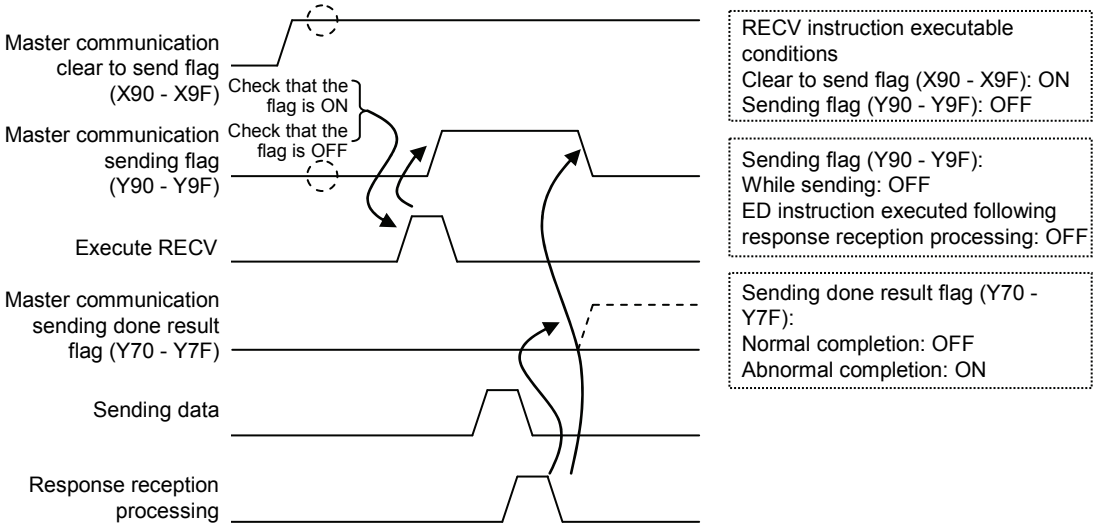
- Using the UNITSEL instruction immediately before the SEND/RECV instruction, specify a port targeted in communication.
- Master communication is only enabled when MEWTOCOL or MODBUS is selected. Check that the master communication clear to send flags (XC to XF) are on for the corresponding channel, and execute the SEND/RECV instruction.
- Another SEND/RECV instruction cannot be executed for a communication port where master communication is in progress. Check that the master communication sending flags (YC to YF) are off, and execute the instruction.
- A SEND/RECV instruction cannot be executed for a port where slave communication is in progress.
- If there is no response, the master communication sending flags (YC to YF) remain ON during the time-out period set in the CPU configuration.
- Up to 16 SEND/RECV instructions can be executed simultaneously for differing COM. ports.

■ Sample program (in the case of CPU with built-in ET-LAN)

- Send the command from the LAN port of the CPU unit, read data from the data areas DT400 to DT401 of the external device, and write into PLC's data registers DT100 to DT101.
- Check that Connection 1 is established in the master mode (X90), and that sending is not in progress in the same port (Y90), and start up the SEND instruction.
- Using the UNITSEL instruction, specify the slot no. (LAN port: U100) and the connection no. (U1).
- In the RECV instruction, specify and execute the partner station no. (U1), starting address (DT400), data amount (U2), and PLC's starting address to store the data (DT100).



■ Time chart (in the case of CPU with built-in ET-LAN)



■ I/O allocation (in the case of CPU with built-in ET-LAN)

I/O number	Name	Explanation
X90 to X9F	Master communication clear to send flag	Turns on when master communication is in a connected status.
Y90 to Y9F	Master communication sending flag	Turns on during sending by the SEND and RECV instructions. Turns off when the ED instruction is executed following completion of response reception processing.
Y70 to Y7F	Sending done result flag	Completion result of sending by general-purpose communication / master communication is notified. (Normal completion: 0; Abnormal completion: 1)

(Note 1) Each contact is used for reading the operation status. Avoid writing by a user program.

■ Precautions during programming (in the case of CPU with built-in ET-LAN)

- Using the UNITSEL instruction immediately before the SEND/RECV instruction, specify a connection no. targeted in communication.
- Master communication is only enabled when MEWTOCOL or MODBUS is selected. Check that the master communication clear to send flags (X90 to X9F) are on for the corresponding connection, and execute the SEND/RECV instruction.
- Another SEND/RECV instruction cannot be executed for a connection where master communication is in progress. Check that the master communication sending flags (Y90 to Y9F) are off, and execute the instruction.
- A SEND/RECV instruction cannot be executed for a connection where slave communication is in progress.
- Up to 16 SEND/RECV instructions can be executed simultaneously for differing connections.
- For communication between LAN ports of FP7, specify "U1" for the partner station no. Receiver is determined by the IP address.

◆ KEY POINTS

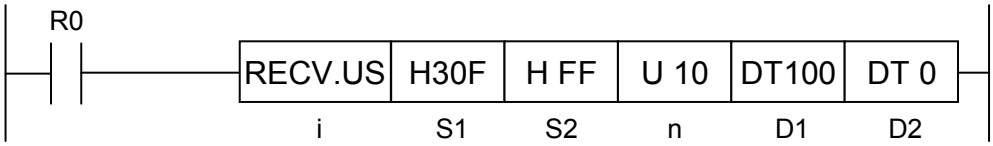
- **As the communication cassette (Ethernet type) has an Ethernet-serial conversion function, the internal interface operates with similar programs as the case of SCU. The setting method and programming method are different from those for the CPU with built-in ET-LAN. Refer to the section describing the case of SCU.**

■ Flag operations

Name	Explanation
SR7 SR8 (ER)	To be set in the case of out-of-range in indirect access (index modification).
	To be set when the destination range is out of the accessible range.
	COM port or connection specified by UNITSEL does not exist, or communication is not possible in the specified connection.
	Partner station no. specified by [S1] is out of the range.
	Partner unit sender data device specified by [S2] is invalid.
	Sent data amount specified by [n] is invalid.
	Data device of the receiver data area in the master unit specified by [D1] is invalid, or exceeds the area.
	Result storage device specified by [D2] is invalid.
	Specified bit devices for [S2] and [D1], and/or specified16-bit device, differ.
	Integer specification for [S2] is only available for the MODBUS address direct specification type, and invalid for other types.

RECV (MODBUS Master: Function Code Specification)

■ Ladder diagram



■ Available operation units (●: Available)

Operation unit	bit	US	SS	UL	SL	SF	DF
i		●	●				

■ List of operands

Items	Settings			Setting range
i	Specify the operation unit.			US / SS
S1	Specify the MODBUS function code to be used, and the partner station no.			
		Higher byte	Two hexadecimal digits that indicate the MODBUS function code.	H1 to H4 (1 to 4)
		Lower byte	Two hexadecimal digits that indicate the station no.	H1 to HF7 (1 to 247)
S2	Specify the sender MODBUS address of the partner unit.			H0 to HFFFF (0 to 65535)
n	Specify the sent data amount.			1 to 127 words 1 to 2040 bits
D1	Specify the device starting address of the receiver data area in the master unit.			-
D2	Specify the device area in the master unit that stores the execution result code (1 word).			

■ Available word devices (●: Available)

Operand	16-bit device											32-bit device			Integers			Real numbers		String	Index modifier
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX	K	U	H	SF	DF	" "	
S1	●	●	●	●			●	●								●	●				●
S2	●	●	●	●			●	●								●	*1				●
n	●	●	●	●			●	●								●	●				●
D1	●	●	●	●			●	●													●
D2	●	●	●	●			●	●													●

*1: Only in the case of "direct address specification" (main instruction) in the MODBUS mode, an integer can be specified for the sender address.

■ Available bit devices (●: Available)

Operand	Bit device											Bit specification of the word device		Index modifier
	X	Y	R	L	T	C	P	E	S	IN	OT	DT.b	LD.b	
S1														
S2	●	●	●	●								●	●	●
n														
D1	●	●	●	●								●	●	●
D2														

■ Outline of operation

- The MODBUS command is sent from the communication port of the unit to perform the data transmission with external devices.
- Message in accordance with the protocol is automatically formulated by PLC. The user program only has to specify the station no. and the memory address, and execute the SEND/RECV instruction, to carry out reading and writing.
- Specify the MODBUS command, and the partner MODBUS address, in a Hex format in [S1].
- When the RECV instruction is executed, data are read from the address starting with [S2] in the partner unit, and stored in the area starting with [D1] in the master unit.
- Depending on the type of device specified by [D1], the transfer method (register transfer / bit transfer), and the MODBUS that can be used, vary.
- Sent data amount [n] is provided in words when register transfer is selected, and in the no. of words or bits when bit transfer is selected.
- The execution result code is stored in 1 word of area within the master unit specified by [D2].

■ Specification of [S1] and [n]

- Operand [S1] is specified as a combination of a two-digit hexadecimal MODBUS function code and a two-digit hexadecimal partner station no.
  Example: Specify "H030F" in the case of MODBUS function code 03 (read holding registers) and station no.. 15.
- Depending on the type of device specified by the operand [D1], the transfer method, and the MODBUS function code that can be used, vary.

Device to be specified for [D1]	Transfer method	Value that can be specified for higher bytes of [S1]
16-bit device WX, WY, WR, WL, DT, LD	Register sending	H1: Read coil status (01) H2: Read input status (02) H3: Read holding registers (03) H4: Read input registers (04)
1-bit device X, Y, R, L, DT, n, LD, n	Bit sending	H1: Read coil status (01) H2: Read input status (02)

- Sent data amount is provided in words when register transfer is selected, and in bits when bit transfer is selected.

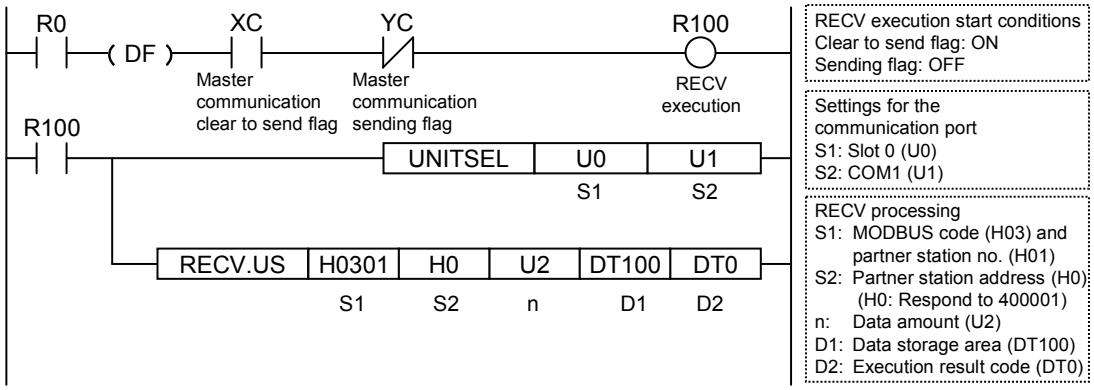
■ Execution result code [D2]

Execution result	Execution result code
Normal completion	0
Communication port is being used for master communication.	1
Communication port is being used for slave communication.	2
No. of master communication instructions that can be used at the same time is exceeded.	3
Sending timeout	4
Response reception timeout	5
Received data error	6

(Note 1) Operand [S1] of the RECV instruction is specified as a combination of a two-digit hexadecimal MODBUS function code and a two-digit hexadecimal partner station no.
(Note 2) If the partner device is an FP series PLC, Operand [S2] of the RECV instruction can be specified with the device no.

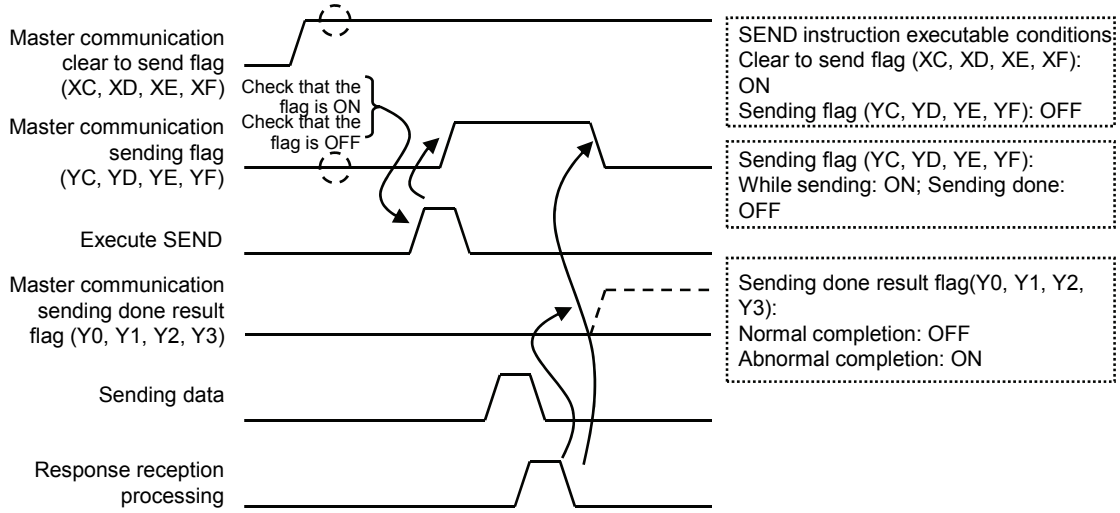
■ Sample program (in the case of SCU)

- Send the command from the COM1 port of the CPU unit, read data from the data areas 400001 to 400002 of the external device (station no. 1), and write the data into the data registers DT100 to DT101 of the PLC.
- Check that the master mode is on (XC), and that sending is not in progress in the same port (YC), and start up the SEND instruction.
- Using the UNITSEL instruction, specify the slot no. (U0) and the COM. port no. (U1).
- In the RECV instruction, specify and execute the partner station no. (U1), MODBUS command to be used and partner station no. (H0301), starting address (40001), data amount (U2), and PLC's starting address to store the data (DT100). See operating instructions of partner devices for their address setting.



(Note 1) Operand [S1] of the RECV instruction is specified as a combination of a two-digit hexadecimal MODBUS function code and a two-digit hexadecimal partner station no.
(Note 2) If the partner device is an FP series PLC, Operand [S2] of the RECV instruction can be specified with the device no.

■ Time chart (in the case of SCU)



◆ KEY POINTS

- **The case of SCU shows the case that it is used in the following combination.**
- **- COM0 port equipped in the CPU unit**
  - Communication cassettes attached to the CPU unit (COM.1 to COM.2 ports)
  - Communication cassettes attached to the serial communication unit (COM.1 to COM.4 ports)
- **The communication cassette (Ethernet type) does not support MODBUS.**

■ I/O allocation (in the case of CPU with built-in SCU)

COM Port No.			Name	Explanation
1	2	0		
XC	XD	XE	Master communication clear to send flag	Turns on when MEWTOCOL-COM, MEWTOCOL7, and MODBUS-RTU are set in the communication mode, and the RUN mode is on.
YC	YD	YE	Master communication sending flag	Turns on during sending by the SEND and RECV instructions. Turns off when sending is completed.
Y0	Y1	Y2	Sending done result flag	Completion result of sending by general-purpose communication / master communication is notified. (Normal completion: 0; Abnormal completion: 1)

(Note 1) Each contact is used for reading the operation status. Avoid writing by a user program.

■ I/O allocation (in the case of Serial Communication Unit)

COM Port No.				Name	Explanation
1	2	3	4		
XC	XD	XE	XF	Master communication clear to send flag	Turns on when MEWTOCOL-COM, MEWTOCOL7, and MODBUS-RTU are set in the communication mode, and the RUN mode is on.
YC	YD	YE	YF	Master communication sending flag	Turns on during sending by the SEND and RECV instructions. Turns off when sending is completed.
Y0	Y1	Y2	Y3	Sending done result flag	Completion result of sending by general-purpose communication / master communication is notified. (Normal completion: 0; Abnormal completion: 1)

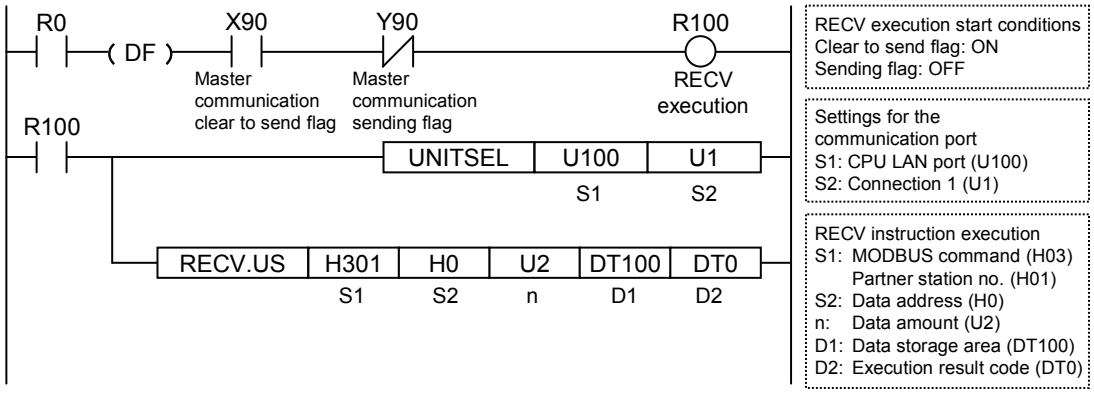
(Note 1) Each contact is used for reading the operation status. Avoid writing by a user program.

■ Precautions during programming (in the case of SCU)

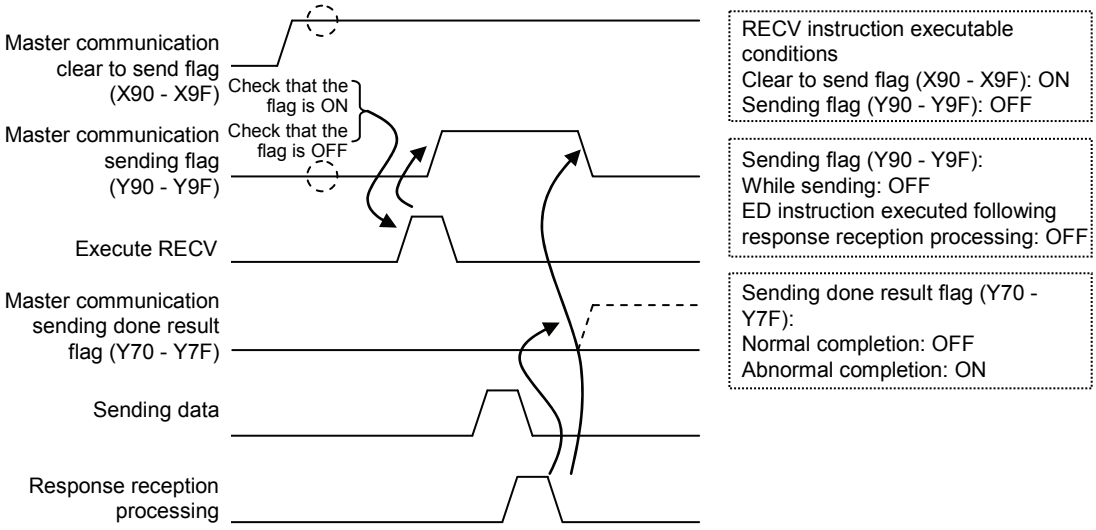
- Using the UNITSEL instruction immediately before the SEND/RECV instruction, specify a port targeted in communication.
- Master communication is only enabled when MEWTOCOL or MODBUS is selected. Check that the master communication clear to send flags (XC to XF) are on for the corresponding channel, and execute the SEND/RECV instruction.
- Another SEND/RECV instruction cannot be executed for a communication port where master communication is in progress. Check that the master communication sending flags (YC to YF) are off, and execute the instruction.
- A SEND/RECV instruction cannot be executed for a port where slave communication is in progress. If there is no response, the master communication sending flags (YC to YF) remain ON during the time-out period set in the CPU configuration.
- Up to 16 SEND/RECV instructions can be executed simultaneously for differing COM. ports.

■ Sample program (in the case of CPU with built-in ET-LAN)

- Send MODBUS commands (03) from the LAN port of the CPU unit, read data from the data area of an external device 40001 to 40002 (MODBUS address 0000H to 0001H), and write the content into PLC's data register DT100 to DT101.
- Check that Connection 1 is established in the master mode (X90), and that sending is not in progress in the same port (Y90), and start up the SEND instruction.
- Using the UNITSEL instruction, specify the slot no. (LAN port: U100) and the connection no. (U1).
- In the RECV instruction, specify and execute the type of MODBUS command and partner station no. (H0301), starting address (H0), data amount (U2), and PLC's starting address to store the data (DT100). See operating instructions of partner devices for their address setting.



■ Time chart (in the case of CPU with built-in ET-LAN)



■ I/O allocation (in the case of CPU with built-in ET-LAN)

I/O number	Name	Explanation
X90 to X9F	Master communication clear to send flag	Turns on when master communication is in a connected status.
Y90 to Y9F	Master communication sending flag	Turns on during sending by the SEND and RECV instructions. Turns off when the ED instruction is executed following completion of response reception processing.
Y70 to Y7F	Sending done result flag	Completion result of sending by general-purpose communication / master communication is notified. (Normal completion: 0; Abnormal completion: 1)

(Note 1) Each contact is used for reading the operation status. Avoid writing by a user program.

■ Precautions during programming (in the case of CPU with built-in ET-LAN)

- Using the UNITSEL instruction immediately before the SEND/RECV instruction, specify a connection no. targeted in communication.
- Master communication is only enabled when MEWTOCOL or MODBUS is selected. Check that the master communication clear to send flags (X90 to X9F) are on for the corresponding connection, and execute the SEND/RECV instruction.
- Another SEND/RECV instruction cannot be executed for a connection where master communication is in progress. Check that the master communication sending flags (Y90 to Y9F) are off, and execute the instruction.
- A SEND/RECV instruction cannot be executed for a connection where slave communication is in progress.
- Up to 16 SEND/RECV instructions can be executed simultaneously for differing connections.
- In the MODBUS-TCP mode, specify the partner station no. as operand for the SEND/RECV instruction.

■ Flag operations

Name	Explanation
SR7 SR8 (ER)	To be set in the case of out-of-range in indirect access (index modification).
	To be set when the destination range is out of the accessible range.
	COM port or connection specified by UNITSEL does not exist, or communication is not possible in the specified connection.
	Partner station no. specified by [S1] is out of the range.
	Partner unit sender data device specified by [S2] is invalid.
	Sent data amount specified by [n] is invalid.
	Data device of the receiver data area in the master unit specified by [D1] is invalid, or exceeds the area.
	Result storage device specified by [D2] is invalid.
	Integer specification for [S2] is only available for the MODBUS address direct specification type, and invalid for other types.

PMSET (Change of SCU Parameters)

■ Ladder diagram

■ List of operands

Operand	Explanation
S	Start of the area that stores data to be set as communication parameters
n	Specified no. of words (setting range: 1 to 12)
D	Starting address of the device area in the master unit that stores the processing result (1 word)

■ Available devices (●: Available)

Operand	16-bit device											32-bit device			Integers			Real numbers		String	Index modifier
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX	K	U	H	SF	DF	" "	
S	●	●	●	●			●	●													●
n	●	●	●	●			●	●								●					●
D	●	●	●	●			●	●													●

■ Outline of operation

- Communication parameters of the COM port of the unit is changed with a user program.
- Describe UNITSEL instruction immediately before the PMGET instruction, and specify the slot and COM port numbers of the unit the parameters of which are changed.
- Set communication parameters to be changed within [n] words from the area starting with [S], and execute the PMSET instruction, to issue the setting change request to the unit.
- While the requested change is being processed, Bit 15 of the processing result storage area [D] turns on. When the process is completed, it turns off.
- The processing result is stored in the area specified by [D]. If there is any abnormality, Bit 14 of [D] turns on. The error code is stored in lower bytes of [D].
- By reading setting parameters using the PMGET instruction, and setting parameters to be changed using the PMSET instruction, the settings can be simplified.

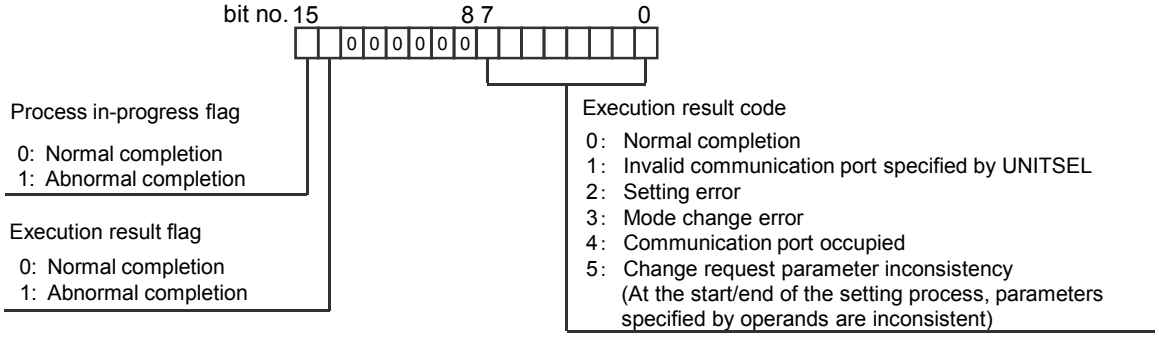
■ Parameter settings

Operand	Parameter	Range	Settings
[S]	Communication mode	U0 U1 U2 U8 U9	U0: MEWTOCOL-COM U1: MEWTOCOL7-COM U2: MODBUS-RTU U8: General-purpose communication U9: PLC link (only station no. can be changed)
[S+1]	Station no. setting	U1 to 999	Station no. U1 to U999 MEWTOCOL-COM: U1 to U99 MEWTOCOL7-COM: U1 to U999 MODBUS-RTU: U1 to U247 PLC link: U1 to U16 (Default: 0)
[S+2]	Baud rate setting	U0 to 10	U0: 300, U1: 600, U2: 1200, U3: 2400, U4: 4800, U5: 9600, U6: 19200, U7: 38400, U8: 57600., U9: 115200, U10: 230400 bps
[S+3]	Data length setting	U0, U1	U0: 7-bit length, U1: 8-bit length
[S+4]	Parity setting	U0 to 2	U0: No parity; U1: Odd parity; U2: Even parity
[S+5]	Stop bit length setting	U0, U1	U0: 1-bit; U1: 2-bit
[S+6]	RS/CS enabled or disabled (Note 1)	U0, U1	U0: Disabled; U1: Enabled
[S+7]	Send Waiting	U0 to 10000	U0: Immediate effective time = Un × 0.01ms (0 to 100 ms)
[S+8]	Start code STX	U0, U1	U0: Disabled; U1: Enabled
[S+9]	Terminator setting	U0 to 3	U0: cR; U1: cR＋Lf; U2: Time; U3: ETX
[S+10]	Terminator judgment time	U0 to 10000	U0: 32 bits Effective time = Un × 0.01 ms (Only enabled when the end setting is "Time".)
[S+11]	Modem initialization	U0 to 2	U0: Modem is not initialized. U1: Initialization is executed only in the first session. (Note 2) U2: Re-execute modem initialization during setting.

(Note 1) RS/CS can be selected only when a 1-channel, 5-wire switch cassette is used for RS232C.
(Note 2) The modem is initialized during setting (when the power is on, PMSET instruction is executed, and the RUN mode is turned on). Initialization is executed only in the first session. (Excluding when the power supply is turned off and then on again)

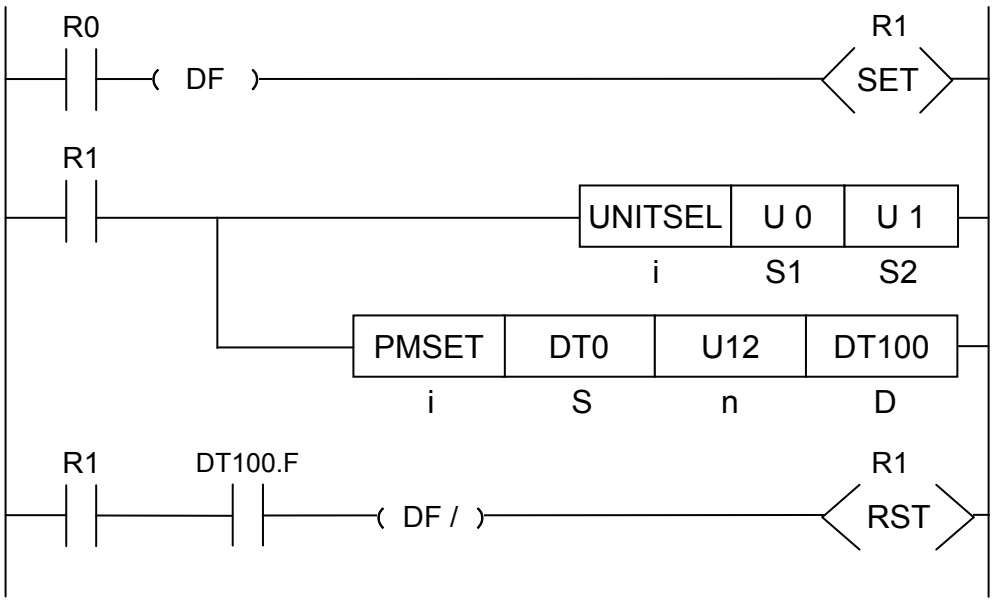
■ Content of the processing result [D]

- Execution results are stored in the area of one word.
- The execution result code in the lower byte is valid when the process in-progress flag of bit 15 is zero.



■ Program example

- Setting 12 words of communication parameters in COM1 port of CPU with built-in SCU, which are stored in the area starting with Data Register DT0.



DT0	U 0	Communication mode
DT1	U 1	Unit no. setting
DT2	U 5	Baud rate setting
DT3	U 1	Data length setting
DT4	U 1	Parity setting
DT5	U 0	Stop bit length setting
DT6	U 0	RS/CS valid or invalid
DT7	U 0	Send waiting time
DT8	U 0	Header STX
DT9	U 0	Terminator setting
DT10	U 0	Terminator judgment time
DT11	U 0	Modem initialization

	Higher byte	Lower byte	
DT100	H 0	H 0	Processing result ('0' for normal completion)

■ Precautions during programming

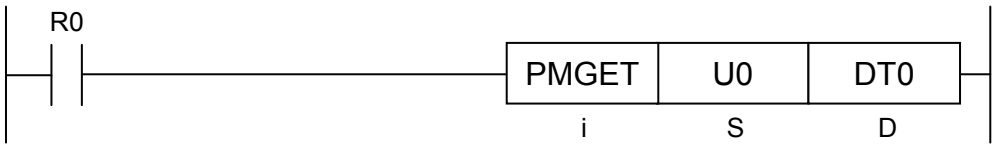
- Checking of the processing result should be carried out when Bit 15 (process in-progress flag) of the area specified by [D] is switched from 1 to 0.
- If parameter change is carried out for a COM port where sending/receiving is in progress, the sending/receiving process is canceled and parameters are changed. At this time, received data are lost. The sending process is suspended.

■ Flag operations

Name	Explanation
SR7 SR8 (ER)	To be set in the case of out-of-range in indirect access (index modification).
	To be set when the COM port specified by UNITSEL does not exist.
	To be set when the setting parameter device specified by [S] is invalid.
	To be set when the no. of words specified by [n] is out of the available range.
	To be set when the device specification for processing result specified by [D] is invalid.

PMGET (Obtainment of SCU Parameters)

■ Ladder diagram



■ List of operands

Operand	Explanation
S	Types of data to be obtained: 0: Communication parameters; 1: COM port operation status monitor
D	Starting address of the area to store the obtained communication parameters (monitoring information)

■ Available devices (●: Available)

Operand	16-bit device											32-bit device			Integers			Real numbers		String	Index modifier
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX	K	U	H	SF	DF	" "	
S	●	●	●	●			●	●								●					●
D	●	●	●	●			●	●													●

■ Outline of operation

- Read the parameters of the COM port of the unit.
- Describe UNITSEL instruction immediately before the PMGET instruction, and specify the slot and COM port numbers of the unit the parameters of which are obtained.
- If the content of [S] is "0", communication parameters are obtained, and stored in the 26-word area starting with [D].
- If the content of [S] is "1", communication monitoring area status is obtained, and stored in the 7-word area starting with [D].

■ Obtainment of SCU communication parameters

Operand	Parameter	Range	Settings
[D]	Communication mode	U0 U1 U2 U8 U9	U0: MEWTOCOL-COM U1: MEWTOCOL7-COM U2: MODBUS-RTU U8: General-purpose communication U9: PLC link
[D+1]	Station no. setting	U1 to 999	Station no. U1 to U999 MEWTOCOL-COM: U1 to U99 MEWTOCOL7-COM: U1 to U999 MODBUS-RTU: U1 to U247 PLC link: U1 to U16 (Default: 0)
[D+2]	Baud rate setting	U0 to 10	U0: 300, U1: 600, U2: 1200, U3: 2400, U4: 4800, U5: 9600, U6: 19200, U7: 38400, U8: 57600., U9: 115200, U10: 230400 bps
[D+3]	Data length setting	U0, U1	U0: 7-bit length, U1: 8-bit length
[D+4]	Parity setting	U0 to U2	U0: No parity; U1: Odd parity; U2: Even parity
[D+5]	Stop bit length setting	U0, U1	U0: 1-bit; U1: 2-bit
[D+6]	RS/CS enabled or disabled (Note 1) Note 1)	U0, U1	U0: Disabled; U1: Enabled
[D+7]	Send Waiting	U0 to 10000	U0: Immediate effective time = Un × 0.01ms (0 to 100 ms)
[D+8]	Start code STX	U0, U1	U0: Disabled; U1: Enabled
[D+9]	End setting	U0 to U3	U0: cR; U1: cR + Lf; U2: Time; U3: ETX
[D+10]	End judgment time	U0 to 10000	U0: 32 bits (Note 3) Effective time = Un × 0.01 ms (Only enabled when the end setting is "Time".)
[D+11]	Modem initialization	U0 to U2	U0: Modem is not initialized. U1: Initialization is executed only in the first session. (Note 2) U2: Re-execute modem initialization during setting.
[D+12]	Reserved area	U0	Reserved area
[D+13]	Reserved area	U0	Reserved area
[D+14]	Linked area block no.	U0, U1	Block no. of the link relay / link register area
[D+15]	PLC link W0 max. station no.	U2 to 16	Values out of the range are handled as "16".
[D+16]	Link relay range	U0 to 64	Specify the range of link relays used for communication (relative values within the specified block)
[D+17]	Link register range	U0 to 128	Specify the range of link registers used for communication (relative values within the specified block)
[D+18]	Link relay sending start no.	U0 to 63	Link relay sending start no. (specification by no. of words, relative value within the specified block)
[D+19]	Size of link relay send area	U0 to 64	Size of link relay send area (specification by no. of words)
[D+20]	Link register sending start no.	U0 to 127	Link register sending start no. (specification by no. of words, relative value within the specified block)
[D+21]	Size of link register send area	U0 to 127	Size of link register send area (specification by no. of words)
[D+22]	Reserved area	U0	Reserved area
[D+23]	Reserved area	U0	Reserved area
[D+24]	Reserved area	U0	Reserved area
[D+25]	Reserved area	U0	Reserved area

(Note 1) RS/CS can be selected only when a 1-channel, 5-wire switch cassette is used for RS232C.
(Note 2) The modem is initialized during setting (when the power is on, PMGET instruction is executed, and the RUN mode is turned on). Initialization is executed only in the first session. (Excluding when the power supply is turned off and then on again)
(Note 3) Settings of [D+14] to [D+21] are only enabled when the communication mode in the COM1 port is PLC link.

■ SCU COM port operation status monitoring information

Operand	Types of monitoring information	Range	Settings
[D]	Operation Mode	U 0 U 1 U 2 U 8 U 9	U0: MEWTOCOL-COM U1: MEWTOCOL7-COM U2: MODBUS-RTU U8: General-purpose communication U9: PLC link
[D+1]	Identification of communication cassettes	U0 U232 U422 U485	U 0: No communication cassette U 232: RS232C U 422: RS422 U 485: RS485
[D+2]	Reception error code	Bit 9: Receive buffer FULL Bit 8: Receive buffer overflow Bit 2: Parity inconsistency Bit 1: Stop bit undetected (frame error) Bit 0: Receive buffer overrun	
[D+3]	No. of occurrences of reception error	Times of detection of reception errors to be stored in lower bytes of the reception error code (unsigned 16-bit circulation)	
[D+4]	Setting error code	Bit 9: Sent data amount abnormality Bit 8: Communication parameters setting abnormality Bit 0: Mode setting/change abnormality (unavailable mode no. is specified)	
[D+5]	Error parameter no.	U 1 to 12	Parameter numbers that specify out-of-range data (Enabled only when setting abnormality of communication parameters has occurred)
[D+6]	Modem initialization status	U 0000 U 0100 U 0200 U 02FF	No processing Initialization in progress Initialization succeeded Initialization failed

■ Precautions during programming

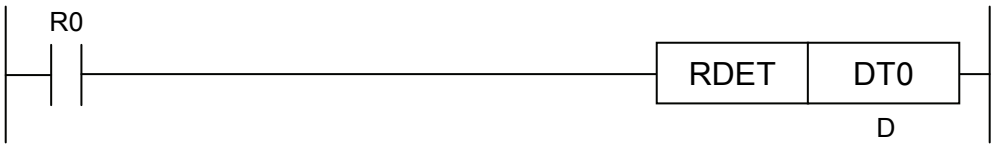
- Using the UNITSEL instruction immediately before the PMGET instruction, specify a port targeted in settings.

■ Flag operations

Name	Explanation
SR7 SR8 (ER)	Out-of-range in indirect access (index modification)
	Destination range is out of the accessible range.
	SCU unit does not exist in the slot specified by UNITSEL.
	COM port specified by UNITSEL does not exist.
	Parameter storage device specified by [D] is invalid.

RDET (Read the ET-LAN Status)

■ Ladder diagram



■ Available operation units (●: Available)

- No operation units

■ List of operands

Operand	Explanation
D	Stored in the 7-word area [D] to [D+6], at the start of the status storage area.

■ Available devices (●: Available)

Operand	16-bit device											32-bit device			Integers			Real numbers		String	Index modifier
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX	K	U	H	SF	DF	" "	
D	●	●	●	●			●	●													●

■ Outline of operation

- Obtain a status summary that indicates statuses of all connections of ET-LAN.
- Describe the UNITSEL instruction immediately before the RDET instruction, and specify the targeted ET-LAN port. Set given values as connection numbers, within the range 1 to 16.
- The obtained information is converted into integer values in the Hex format, in accordance with the relevant allocation, and stored in the 7-word area starting with [D].

■ ET-LAN status information

- Connection status of all connections
- OPEN status
- OPEN abnormality status
- No. of connections in-progress in the FTP server

■ ET-LAN status information

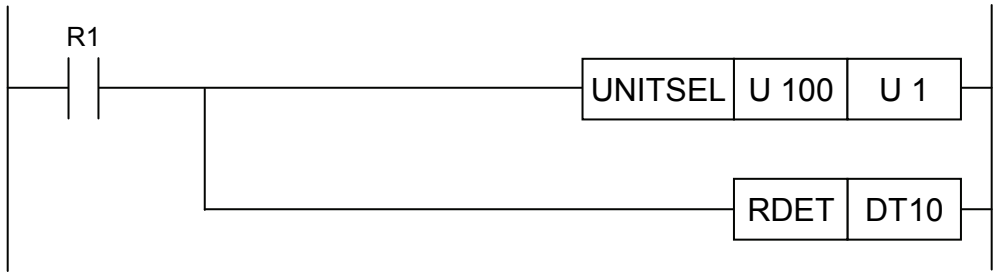
Operand	Data name		Data to be stored	
[D]	Connection status summary	Lower words	0: Other than connected status 1: Connected status	They are stored in the corresponding bits as allocated in the following table. S1 to S4: System connection U1 to U16: User connection FTP-S: FTP server
[D+1]		Higher words		
[D+2]	OPEN status summary	Lower words	0: Close 1: Open	
[D+3]		Higher words		
[D+4]	OPEN abnormality status summary	Lower words	0: No abnormality 1: Abnormality	
[D+5]		Higher words		
[D+6]	No. of connections in-progress in the FTP server			

(Note) Correspondence between connections and bits for the connection status summary, OPEN status summary, and OPEN abnormal status summary

Higher words							Lower words															
b15 - b9	b8	b7 - b4	b3	b2	b1	b0	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	FTP-S	-	U16	U15	U14	U13	U12	U11	U10	U9	U8	U7	U6	U5	U4	U3	U2	U1	S4	S3	S2	S1

■ Program example

- Obtain all the connection summaries for CPU with built-in ET-LAN, and store them in the 7-word area starting with DT10.



■ Precautions during programming

- It is necessary to set the slot no. and connection no. of ET-LAN targeted in communication, using the UNITSEL instruction.

■ Flag operations

Name	Explanation
SR7 SR8 (ER)	To be set in the case of out-of-range in indirect access (index modification).
	To be set when the destination range is out of the accessible range.
	Connection specified by UNITSEL does not exist, or the value is out of the range.
	Parameter storage device specified by [D] is invalid.

POSSET (Setting of Positioning Starting Table)

■ Ladder diagram

■ Available operation units (●: Available)

- No operation units

■ List of operands

Operand	Explanation
S1	Slot no. where the positioning unit is attached (unsigned 16-bit integer)
S2	Axis no. to start up the positioning table (unsigned 16-bit integer); 1 to 4: Axis 1 to 4; 8: Virtual axis
S3	Table no. to start up the position control (unsigned 16-bit integer); 1 to 600, 10001 to 10025

(Note) If a value out of the available range is set as the starting table no., the positioning unit notifies an error.

■ Available devices (●: Available)

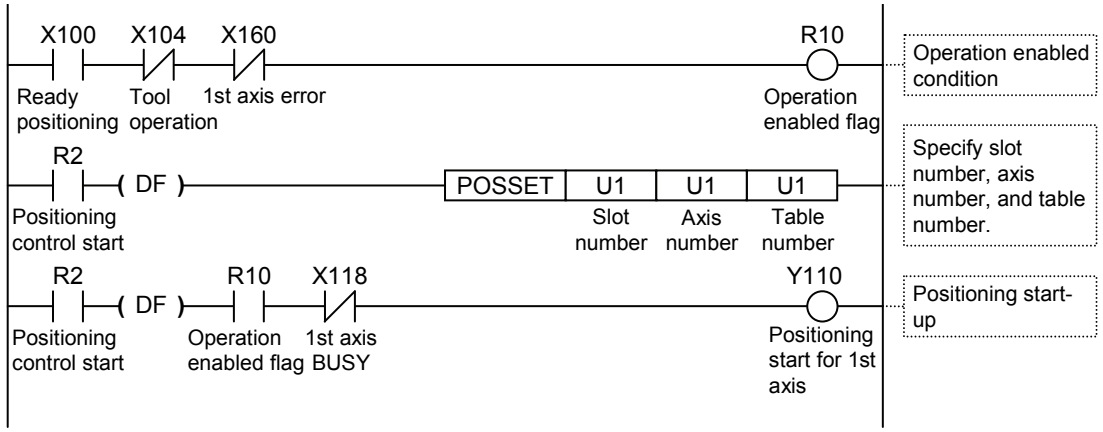
Operand	16-bit device											32-bit device			Integers			Real numbers		String	Index modifier *1
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX	K	U	H	SF	DF	" "	
S1	●	●	●	●			●	●	●	●	●					●	●				●
S2	●	●	●	●			●	●	●	●	●					●	●				●
S3	●	●	●	●			●	●	●	●	●					●	●				●

*1: Only 16-bit devices, and integer constants can be modified (real number constants, and character constants cannot be specified).

■ Outline of operation

- Describe it immediately before the program to start up positioning, and set the positioning data table to be started up.
- For the axis no. of the positioning unit specified by [S1] and [S2], set the data table specified by [S3].
- Data for the positioning data table should be set using Configurator PM7 in the tool software FPWIN GR7, or a user program.
- Positioning parameters should be set in the configuration menu of the tool software FPWIN GR7.

■ Program example



■ Flag operations

Name	Explanation
SR7 SR8 (ER)	To be set in the case of out-of-range in indirect access (index modification).
	To be set when the slot no. and/or the axis no. is out of the available range.

PSTRD (Obtainment of Axis Status)

■ Ladder diagram

■ Available operation units (●: Available)

- No operation units

■ List of operands

Operand	Explanation
S1	Slot no. where the positioning unit is attached (unsigned 16-bit integer)
S2	Axis no. to read the axis status information (unsigned 16-bit integer); 1 to 4: Axis 1 to 4; 8: Virtual axis
D	Device address to store the axis status information (unsigned 16-bit integer)

■ Available devices (●: Available)

Operand	16-bit device											32-bit device			Integers			Real numbers		String	Index modifier *1
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX	K	U	H	SF	DF	" "	
S1	●	●	●	●			●	●	●	●	●					●	●				●
S2	●	●	●	●			●	●	●	●	●					●	●				●
D	●	●	●	●			●	●	●		●										●

*1: Only 16-bit devices, and integer constants can be modified (real number constants, and character constants cannot be specified).

■ Outline of operation

- For the axis no. of the positioning unit specified by [S1] and [S2], obtain main flag statuses as axis statuses.
- The obtained information is converted into integer values in the Hex format, in accordance with the relevant allocation, and stored in the area specified by [D].

■ Types of axis status information

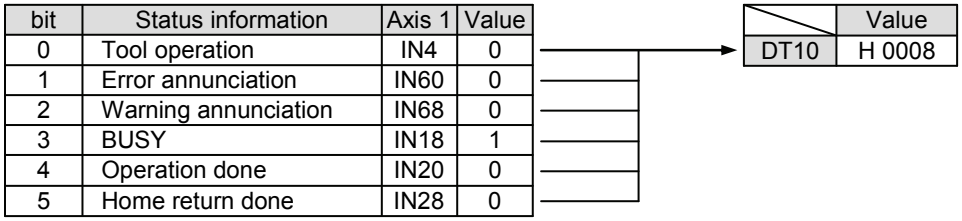
Status information	Description
Tool operation	Turned on in the case of tool operation using Configurator PM, regardless of the specified axis.
Error annunciation	Turned on when an error occurs in the specified axis.
Warning annunciation	Turned on when a warning occurs in the specified axis.
BUSY	Turned on when the specified axis is operating.
Operation done	Turned on when the specified axis has completed operation.
Home return done	Turned on when the specified axis has completed home return.

■ Allocation of axis status information to be stored in [D]

bit	Status information	Axis 1	Axis 2	Axis 3	Axis 4	Virtual axis
0	Tool operation	X4	X4	X4	X4	X4
1	Error annunciation	X60	X61	X62	X63	X67
2	Warning annunciation	X68	X69	X6A	X6B	X6F
3	BUSY	X18	X19	X1A	X1B	X1F
4	Operation done	X20	X21	X22	X23	X27
5	Home return done	X28	X29	X2A	X2B	X2F

■ Example of processing

- Read axis status information for Axis 1 of the positioning unit attached to Slot No. 3.



bit	Status information	Axis 1	Value
0	Tool operation	IN4	0
1	Error annunciation	IN60	0
2	Warning annunciation	IN68	0
3	BUSY	IN18	1
4	Operation done	IN20	0
5	Home return done	IN28	0

	Value
DT10	H 0008

■ Flag operations

Name	Explanation
SR7	To be set in the case of out-of-range in indirect access (index modification).
SR8 (ER)	To be set when the slot no. and/or the axis no. is out of the available range.

PERRD (Obtainment of Error/Warning in the Positioning Unit)

■ Ladder diagram

■ Available operation units (●: Available)

- No operation units

■ List of operands

Operand	Explanation
S1	Slot no. where the positioning unit is attached (unsigned 16-bit integer)
S2	Axis no. to read the error/warning information (unsigned 16-bit integer); 1 to 4: Axis 1 to 4; 8: Virtual axis
D	Device address to store the error/warning code (unsigned 16-bit integer)

■ Available devices (●: Available)

Operand	16-bit device											32-bit device			Integers			Real numbers		String	Index modifier *1
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX	K	U	H	SF	DF	" "	
S1	●	●	●	●			●	●	●	●	●					●	●				●
S2	●	●	●	●			●	●	●	●	●					●	●				●
D	●	●	●	●			●	●	●		●										●

*1: Only 16-bit devices, and integer constants can be modified (real number constants, and character constants cannot be specified).

■ Outline of operation

- For axis no. of the positioning unit specified by [S1] and [S2], obtain error codes and warning codes stored in the respective notification buffer 1.
- The error codes are stored in the area specified by [D], and warning codes in the area specified by [D+1].

■ Example of processing

- Read error codes and warning codes for Axis 3 of the positioning unit attached to Slot No. 1.

Classification	Name	UM (Hex)	Value
Error information	Three-axis error code Alarm buffer 1	UM 0015A	H 4022
Warning information	Three-axis warning code Alarm buffer 1	UM 001E2	H B010

・Storage location
[D]……DT0 H 4022
[D+1]…DT1 H B010

■ Flag operations

Name	Explanation
SR7	To be set in the case of out-of-range in indirect access (index modification).
SR8	To be set when the slot no. and/or the axis no. is out of the available range.
(ER)	To be set when the destination range is out of the accessible range.

UCLR (Error /Warning Clear)

■ Ladder diagram

■ Available operation units (●: Available)

- No operation units

■ List of operands

Operand	Explanation
S	Specify the slot no. (unsigned 16-bit integer)

■ Available devices (●: Available)

Operand	16-bit device											32-bit device			Integers			Real numbers		String	Index modifier *1
	WX	WY	WR	WL	WS	SD	DT	LD	UM	WI	WO	TS CS	TE CE	IX	K	U	H	SF	DF	" "	
S	●	●	●	●			●	●		●						●	●				●

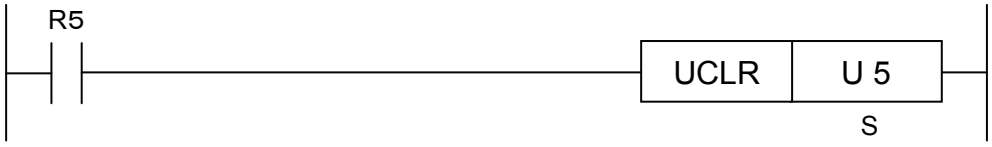
*1: Only 16-bit devices, and integer constants can be modified (real number constants, and character constants cannot be specified).

■ Outline of operation

- Clear an error/warning in the unit attached to the slot no. specified by [S].

■ Program example

Example) Clear an error/warning in the unit attached to Slot No.5
[S]…U5



■ Flag operations

Name	Explanation
SR7	To be set in the case of out-of-range in indirect access (index modification).
SR8 (ER)	To be set when the slot no. and/or the axis no. is out of the available range.

4

Precautions During Programming

4-1 Common Precautions

The FP7 series differs from the existing FP series in the following points:

■ Operations of the carry flag and comparison flags (●: Change; -: Do not change)

Type of instruction	Mnemonic	Name of instruction	System relay no.			
			Carry flag	Comparison flags		
			SR9 (CY)	SRA (>)	SRB (=)	SRC (<)
Data comparison instructions	CMP	Data compare	–	●	●	●
	WIN	Band compare	–	●	●	●
	BCMP	Block data compare	–	–	●	–
Bit comparison instructions	BTT	16-bit data specified bit test	–	–	●	–
	STC	Carry-flag set	●	–	–	–
	CLC	Carry-flag reset	●	–	–	–
String operation instructions	SCMP	String compare	–	●	●	●

- When operation is executed, the carry flag SR9 (CY) and comparison flags SRA (>), SRB (=), and SRC (<) are respectively activated by the instructions listed above.
- The flags do not change even if the operation result is "0" in the case of overflow or underflow with instructions other than listed above.

■ Operation at the time of overflow and underflow

- The carry flag SR9 (CY) does not change even in the case of overflow or underflow.
- Note that overflow or underflow may result if an incorrect operation unit is specified.

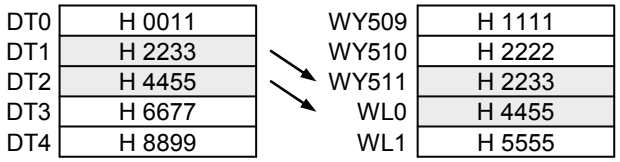
■ Error flags

- Operation error flags are not cleared even in the case of normal operation.
- In order to clear an error flag, use an ERR instruction.

■ Operation device areas

- When the operation unit is 32 bits (UL, SL, SF) or 64 bits (DF), if an area near the end of an operation device is specified, the device memory with a consecutive address within PLC is overwritten. Specify an operand so that areas are not incorrectly read or written.

Example) When a transfer instruction is used, and the operation unit is specified as 32 bits, if WY511 is selected for the transfer instruction [D], the starting area of WL0 is overwritten.



4-2 Clock and Time Data

■ Built-in calender timer of the CPU unit

- The calender timer should be adjusted in the "Set PLC Date and Time" menu of the tool software FPWINGR7, or using the TIMEWT instruction.
- The set values are stored in the system data registers SD50 through 56, as listed below.

Device number	Name of instruction	Data range	Remark
SD50	Calender timer (year)	00 to 99	The two lower digits of year in A.D. can be adjusted (up to 2099). Leap years can be set in this operation. Take leap years into account during setting. A year that can be divided by 4 is a leap year, and the days of February can be set to 29.
SD51	Calender timer (month)	01 to 12	
SD52	Calender timer (day)	01 to 31	
SD53	Calender timer (hours)	0 to 23	
SD54	Calender timer (minutes)	0 to 59	
SD55	Calender timer (seconds)	0 to 59	
SD56	Calender timer (day-of-the-week)	0 to 6	0: Sun., 1: Mon., 2: Tue., 3: Wed., 4: Thu., 5: Fri., 6: Sat.

■ Instructions that handle clock or time data

Mnemonic	Operand(s)	Functions
HMSS	S, D	(Time data) → (Seconds data)
SHMS	S, D	(Seconds data) → (Time data)
CADD	S1, S2, D	(Clock data) + (Time data) → (Clock data)
CSUB	S1, S2, D	(Clock data) - (Time data) → (Clock data)
TMSEC	S, D	(Clock data) → (Seconds data from the base time)
SECTM	S, D	(Seconds data from the base time) → (Clock data)
TIMEWT	S	(Clock data) → (SD50 to SD56)

■ Format of clock data

- One-word 16-bit binary data are allocated to each unit (year, month, day, hours, minutes and seconds), and the overall clock data are handled in the unit of 6 words.

Example) DT0 is specified for operand

	Word	Range
DT0	Year	00 to 99
DT1	Month	01 to 12
DT2	Day	01 to 31
DT3	Hours	0 to 23
DT4	Minutes	0 to 59
DT5	Seconds	0 to 59

■ Format of time data

- One-word 16-bit binary data are allocated to each unit (hours, minutes and seconds), and the overall time data are handled in the unit of 3 words.
- Hours data can be specified in the range from 0 to 9999.

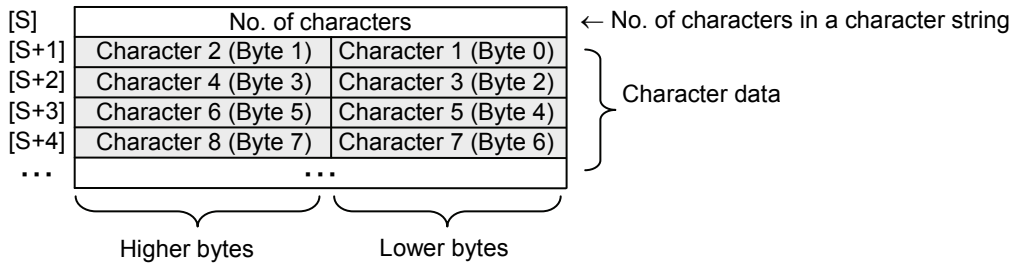
Example) DT0 is specified for operand

	Word	Range
DT0	Hours	0 to 9999
DT1	Minutes	0 to 59
DT2	Seconds	0 to 59

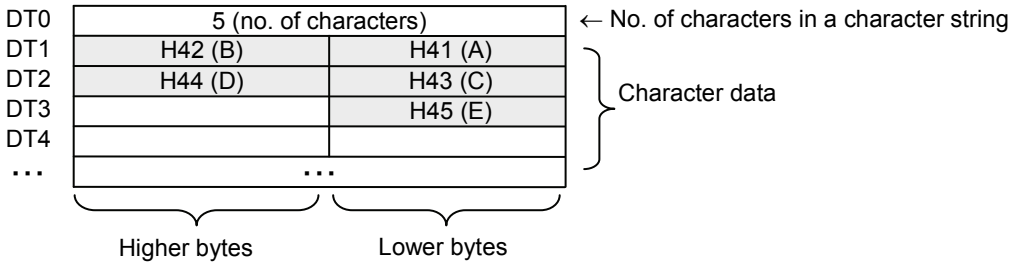
4-3 Data Table for String Instructions

■ Structure of data table

- Data handled by a string instruction store the no. of characters in the initial word, and character data in the subsequent words.
- The max. available size for string data is 4,096 bytes.



Example) A string data table is specified with the no. of characters: 5, and character data: "ABCDE"

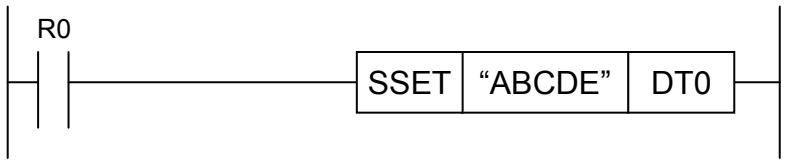


■ Conversion of string data using an SSET instruction

- Using an SSET instruction, a given string can be easily converted into a string data table.

Example) Convert string data "ABCDE"

The no. of characters is stored in DT0, and ASCII-converted character data are stored in DT1 and subsequent data registers.



4-4 Floating Point Real Number Operation

In floating point real number operation, operations in the case of infinitesimal or infinite operation result or input value are as follows.

■ Infinitesimal operation result

Instruction	SIN	COS	TAN	ASIN	ACOS	ATAN	ATAN2	SINH	COSH	TANH	EXP	LN	LOG	PWR	SQR
Input value	–	–	-π/2	●	●	–	●	–	–	–	–	●	●	●	●

(Note 1) Instructions indicated with "-" do not result in an infinitesimal value.
(Note 2) Instructions indicated with ● result in an operation error.
(Note 3) An operation error is output if the input value is out of the available range.

■ Infinite operation result

Instruction	SIN	COS	TAN	ASIN	ACOS	ATAN	ATAN2	SINH	COSH	TANH	EXP	LN	LOG	PWR	SQR
Input value	–	–	-π/2	●	●	–	●	–	–	–	–	●	●	●	●

(Note 1) Instructions indicated with "-" do not result in an infinite value.
(Note 2) Instructions indicated with ● result in an operation error.
(Note 3) An operation error is output if the input value is out of the available range.

■ Infinitesimal (-∞) input value

Instruction	SIN	COS	TAN	ASIN	ACOS	ATAN	ATAN2	SINH	COSH	TANH	EXP	LN	LOG	PWR	SQR
Input value	nan	nan	nan	●	●	-π/2	●	-∞	+∞	-1.0	+0.0	●	●	●	●

(Note 1) Instructions indicated with "nan" result in an unstable value.
(Note 2) Instructions indicated with ● result in an operation error.
(Note 3) An operation error is output if the input value is out of the available range.

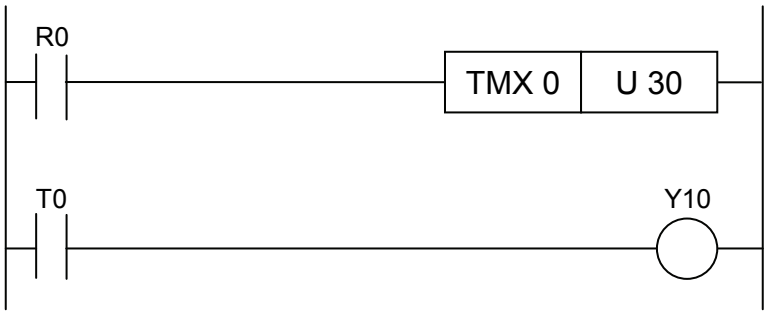
■ Infinite (+∞) input value

Instruction	SIN	COS	TAN	ASIN	ACOS	ATAN	ATAN2	SINH	COSH	TANH	EXP	LN	LOG	PWR	SQR
Input value	nan	nan	nan	●	●	-π/2	●	+∞	+∞	+1.0	+∞	●	●	●	●

(Note 1) Instructions indicated with "nan" result in an unstable value.
(Note 2) Instructions indicated with ● result in an operation error.
(Note 3) An operation error is output if the input value is out of the available range.

4-5 Changing Timer/Counter Set Value in the RUN Mode

(1) Method to rewrite a constant in a program



■ Set value (constant) in a program

A constant in a program can be rewritten under the following conditions.

・Operation mode: RAM/ROM operation only

・Rewriting method: (1) Use the tool software

■ Use the tool software FPWIN GR7

<Procedure>

In this example, a set value of Timer 0 is changed from U30 to U50.

1) Place the cursor over the set value for Timer 0 ("U30").

2) Enter a new constant "U50" and press the return key.

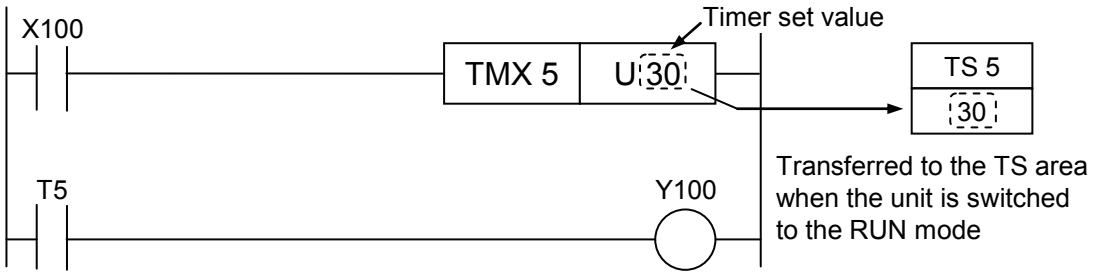
3) Press <Ctrl> + <F1> keys to execute [Convert PBs].

4) Press [Yes (Y)] after a confirmation message is displayed.

■ Post-change operation and precautions

- Timer/counter in operation continues the pre-change operation. The timer/counter starts operation based on the changed set value after the execution condition is turned off and then on again in the next time.
- When a constant in a program is rewritten, the program itself is overwritten. Therefore, when the unit is switched to another mode and then returned to RUN, or when it is powered off and then on again, the program is preset with the changed set value.

(2) Method to rewrite a value in the set value area



■ Changing a value in the set value area TS/CS

The set value area TS/CS can be rewritten under the following conditions.

・Operation mode: RAM/ROM operation

・Rewriting method: (1) Use the tool software; (2) Use a programmed high-level instruction

■ Post-change operation and precautions

- Timer/counter in operation continues the pre-change operation. The timer/counter starts operation based on the changed set value after the execution condition is turned off and then on again in the next time.
- In this method, the program itself is not overwritten even if a value in the set value area TS/CS is changed. Therefore, operations as described below follow when the unit is switched to another mode and then returned to RUN, or when it is powered off and then on again.

1) When a set value is specified by a U constant:

The constant is preset in the set value area TS/CS. The changed value is disabled.

2) When a set value is specified by a set value area number:

In the case of the timer, "0" is preset in the set value area TS.

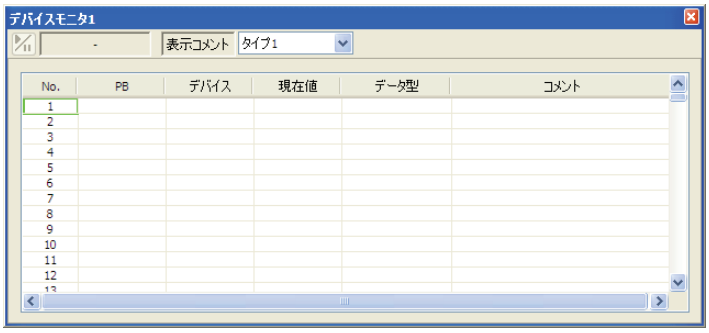
In the case of the counter, a value changed in a method as described in the next page is preset in the set value area CS.

■ Method (1)　Use the tool software FPWIN GR7

<Procedure>

1) From the menu bar, select "Online" → "Device monitor".

The "Device monitor" dialog box is displayed.



2) Double click on the "Device" column.

The "Register monitor device" dialog box is displayed.

3) Under the device type, select TS (timer set value) or CS (counter set value) and enter a given device no.

The entered no. is displayed in the "Device monitor" dialog box.

4) Enter a given value under "Current value" and press the return key.

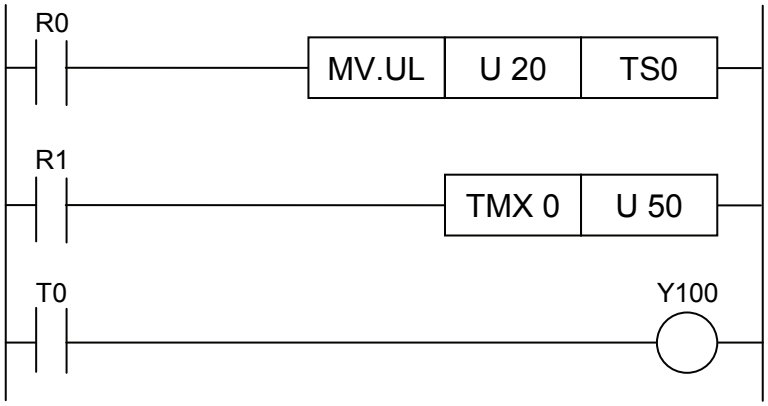
The set value area is updated.

■ Method (2)　Use a programmed high-level instruction

- When a set value for the timer/counter is to be changed due to input conditions or for other reasons, rewrite the value of the set value area TS of the timer/counter to be changed, using a high-level instruction as described below.

[Example] When the input R0 is ON, change the set value to U20

When R0 is ON, the timer set value is changed from 5 to 2 seconds.



- The set value can be changed by specifying the data register DT as the set value area, and modifying the value to be transferred using an MV instruction, etc. It is also possible to specify the number of the set value area (TS/CS) as an operand for the set value area.

4-6 Use of Duplicate Outputs

■ What is "duplicate outputs"?

- "Duplicate outputs" refers to cases where the same output is duplicatedly specified in a single sequence program.
- If the same output is specified for the “OT” and “KP” instructions, it is considered to be duplicate outputs.
- Even if the same output is used for other multiple instructions, such as SET, RST, or high-level instructions (e.g. data transfer), it is not regarded as duplicate outputs.
- Usually, an error occurs if the unit is switched to the RUN mode with duplicate outputs. The ERROR LED goes on and the self-diagnosis flag SR0 turns on.

■ How to check for duplicate outputs

- You can check for duplicate outputs in a program using the tool software FPWIN GR7 in the following method.

1) From the menu bar, select "Debug" → "Project total check".

The "Project total check" dialog box is displayed.

2) Press the [Execute (E)] button.

If duplicate outputs are identified, the PB name, address, and error details (duplicate outputs definition error) are displayed.

3) Select a given line and press the [Jump] button.

The cursor moves to the instruction where duplicate outputs are involved.

■ Enabling duplicate outputs

- If you need to duplicatedly use the same output due to the content of the program, duplicate outputs can be enabled.
- In such cases, switch "Duplicate output authorization" to ON under "CPU configuration" → "Select operation".
- When this is done, an error will not occur when the program is executed.

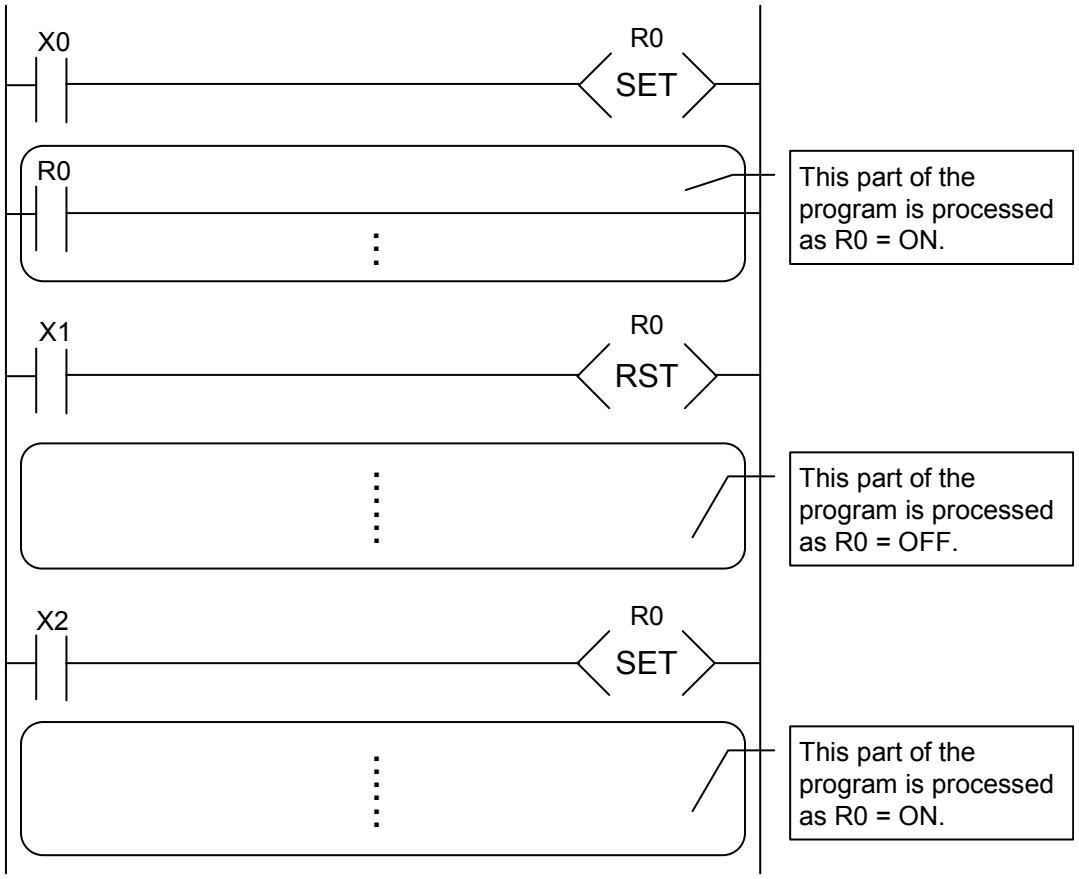
◆ **NOTE**

- **Even when a project total check is conducted using the tool software FPWINGR7, the instruction used at the start is not indicated. Instead, the second and later outputs that are regarded as duplicate use are indicated.**

■ Processing of duplicate uses of the same output

- If the same coil is specified in instructions ouput to internal relays or output relays (e.g. OT, KP, SET, RST and/or transfer instructions), contents thereof are rewritten at each step during operation.

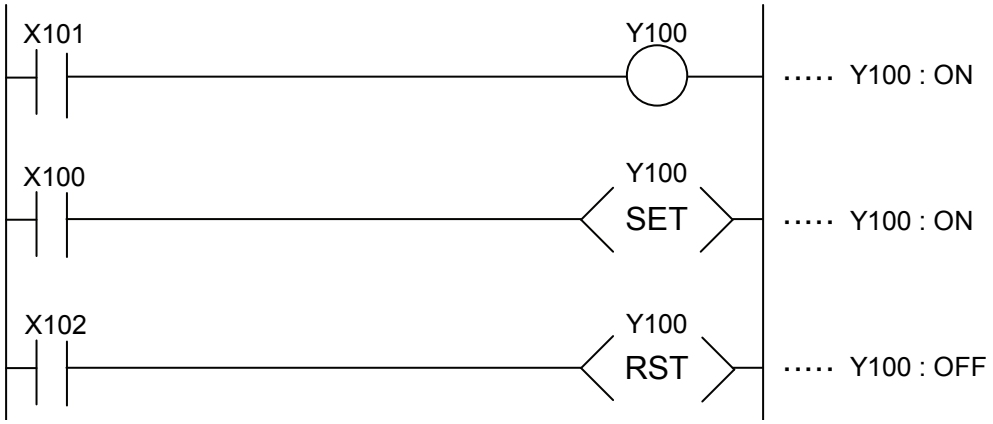
[Example] Processing when SET, RST and OT instructions are used (with X100 to X102 all on)



■ Determination of operation result

- If the same output is duplicatedly used by several instructions such as the OT, KP, SET, RST and/or transfer functions, the output obtained at the I/O refresh is determined by the final operation result.

[Example] The same output relay Y100 is used in OT, SET and RST instructions



4-7 Leading Edge Detection Method

■ Instructions using the leading edge detection operation

(1) DF (leading edge differential) instruction

(2) Count input for CT (counter) instruction

(3) Count input for UDC (up-down counter) instruction

(4) Shift input for SR (shift register) instruction

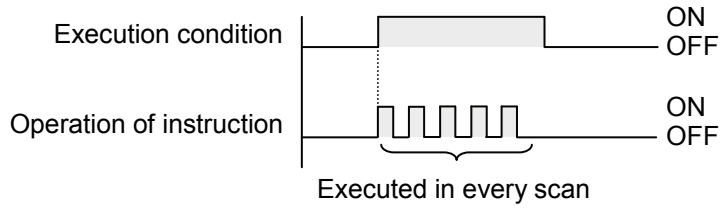
(5) Shift input for LRSR (left-right shift register) instruction

(6) High-level instructions executed only at the leading edge (instructions specified with P and a number)

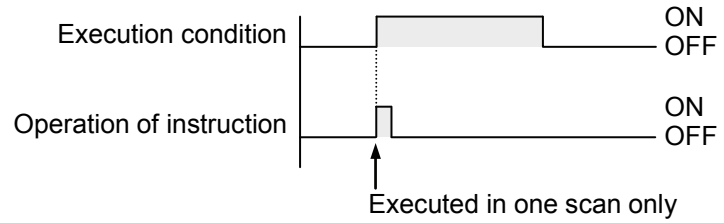
■ What is "leading edge detection method"?

- An instruction with a leading edge detection method operates only in the scan where its trigger (execution condition) is detected switching from off to on.

1) Standard operation



2) Leading edge detection operation



■ How to perform leading edge detection

- The condition of the previous execution and the condition of the current execution are compared, and the instruction is executed only if the previous condition was off and the current condition is on.
- In any other case, the instruction is not executed.

■ Precautions when using an instruction which performs leading edge detection

- This type of instruction is not executed after powering on or at the start of the RUN mode, because the off to on change of the execution condition is not detected.
- Take care that, when such instructions are used in combination with any of the instructions listed in (1) to (6) below, which change the order of execution of instructions, operations may change depending on input timing.

[Be careful when leading edge detection type instructions are used in combination with:]

(1) MC - MCE instructions

(2) JP - LBL instructions

(3) LOOP - LBL instructions

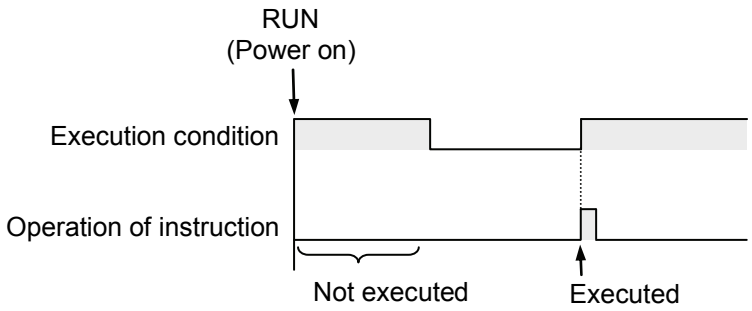
(4) CNDE instructions

(5) Stepladder instructions

(6) Subroutine instructions

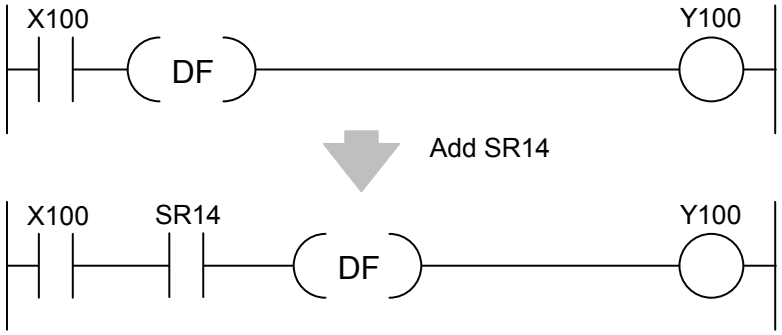
■ Operation and precautions when RUN starts

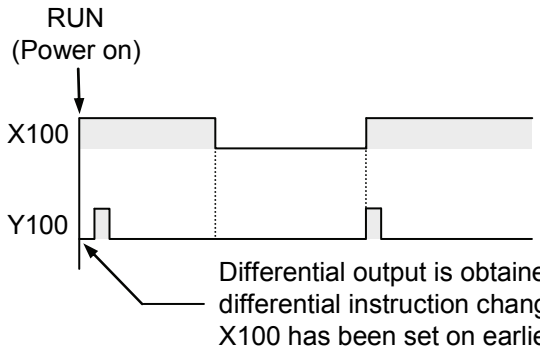
- The leading edge detection instruction is not executed when the mode has been switched to the RUN mode, or when the power supply is booted in the RUN mode, and if the trigger (execution condition) is already on.



- If you need to execute an instruction when the trigger (execution condition) is on prior to switching to the RUN mode, make a program as below using SR14 (special internal relay). (SR14 is a special internal relay which is off during the first scan and turns on at the second scan onwards.)

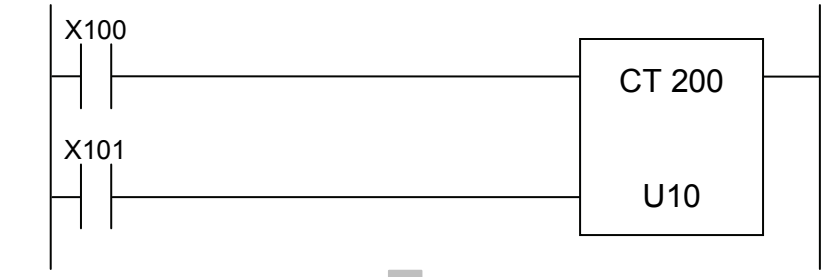
[Example 1] DF (leading edge differential) instruction



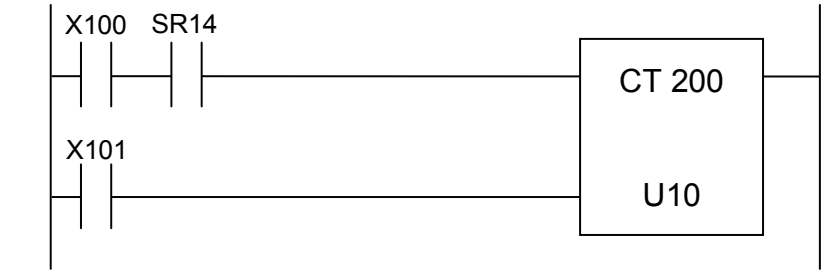


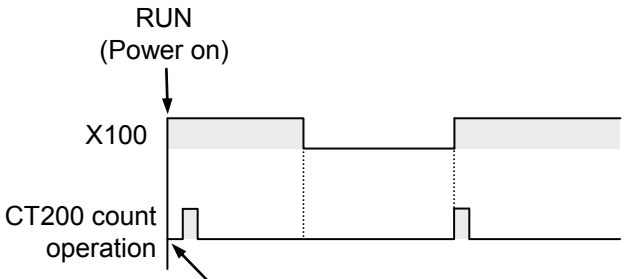
Differential output is obtained because the input condition of the differential instruction changes from off to on in the second scan, even if X100 has been set on earlier.

[Example 2] CT (counter) instruction



Add SR14



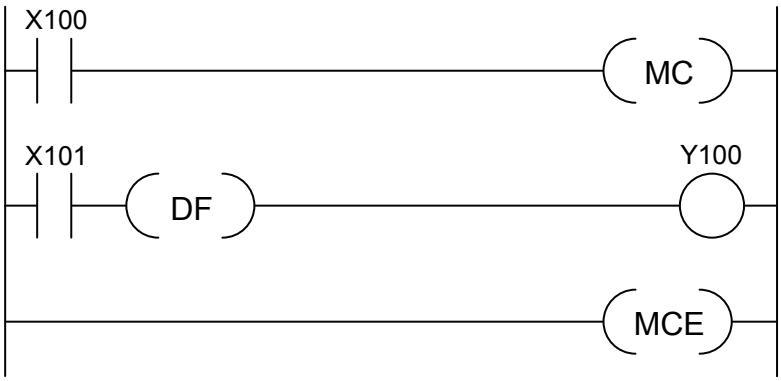


This is counted because the counter input condition changes from off to on in the second scan, even if X100 has been set on earlier.

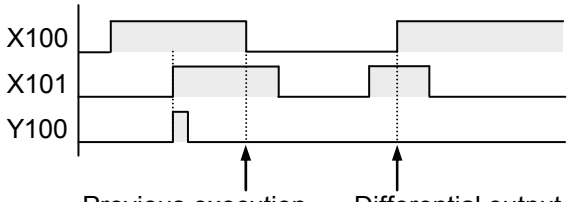
■ Precautions when using a control instruction

- The condition of the previous execution and the condition of the current execution are compared, and the leading edge detection instruction is executed only if the previous condition was off and the current condition is on. In any other case, the instruction is not executed.
- Therefore, take care that, when a leading edge detection instruction is used in combination with an instruction which changes the order of instruction execution, such as MC - MCE or JP - LBL, operations may change as follows depending on input timing.

[Example 1] DF (leading edge differential) instruction is used between MC - MCE



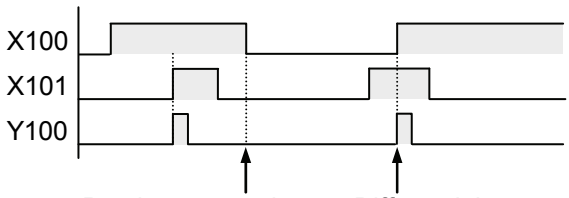
Timing chart 1



Previous execution of differential instruction

Differential output is not obtained because X101 as the input condition for the differential instruction has not been changed from off to on after the previous execution.

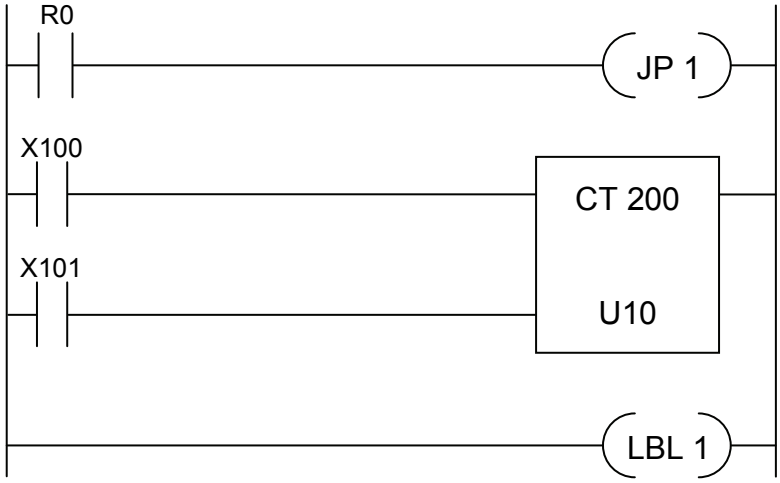
Timing chart 2



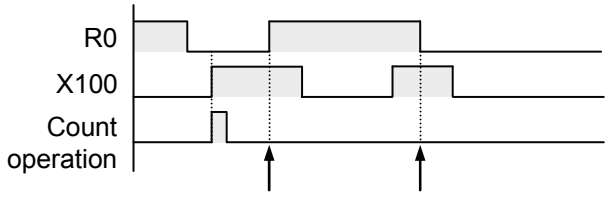
Previous execution of differential instruction

Differential output is obtained because X101 as the input condition for the differential instruction has been changed from off to on after the previous execution.

[Example 2] CT (counter) instruction is used between JP - LBL



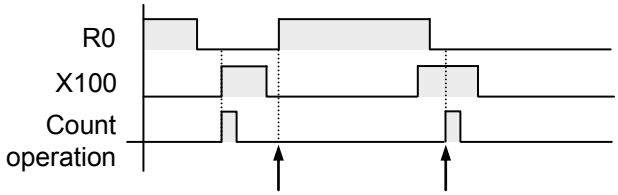
Timing chart 1



Final timing of the previous non-execution of JP instruction

This is not counted because X0 as the trigger (execution condition) for counter input has not been changed after the final timing of the previous non-execution of JP instruction.

Timing chart 2



Final timing of the previous non-execution of JP instruction

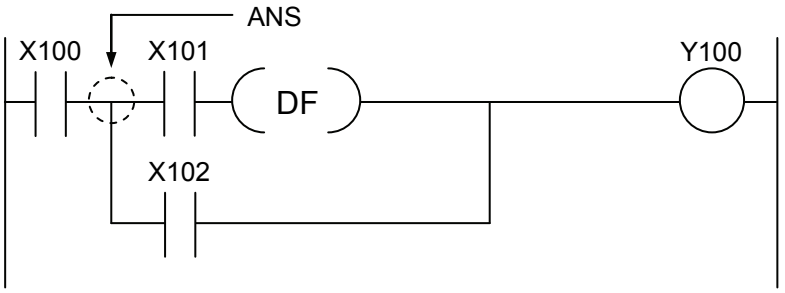
This is counted because counter input has been changed from off to on after the final timing of the previous non-execution of JP instruction

4-8 Precautions for Programming

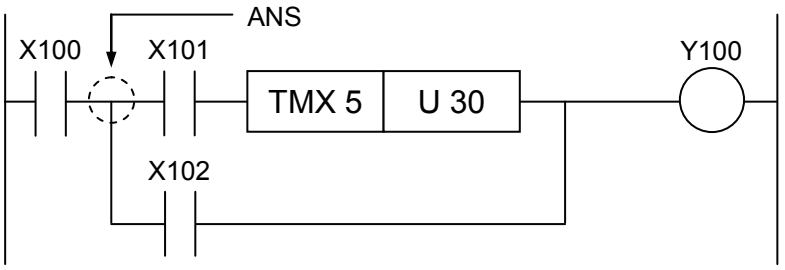
■ Programs which are not executed correctly

- Do not write the following programs as they will not be executed correctly.

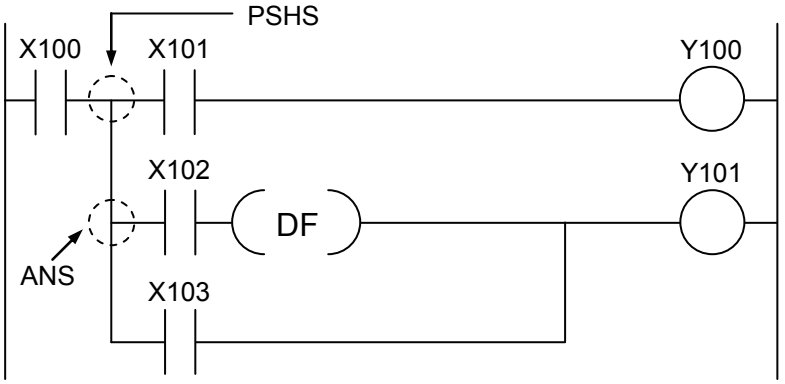
[Example 1] If X101 is set on earlier, Y100 does not go on even when X100 is turned on.



[Example 2] Regardless of the on or off state of X100, TMX5 is activated if X101 is turned on.



[Example 3] If X102 is set on earlier, Y101 does not go on even when X100 is turned on.

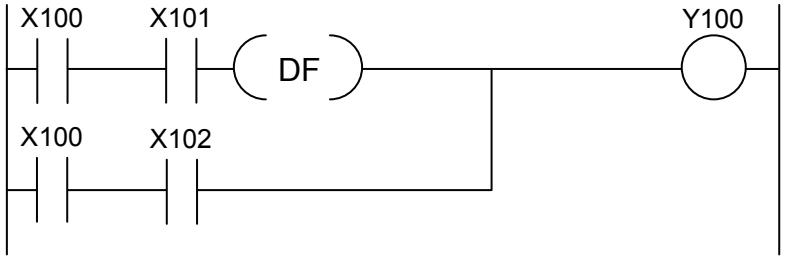


- When a combination of contacts are set as the trigger (execution condition) of a differential instruction or timer instruction, do not use an ANS instruction, RDS instruction, or POPS instruction.

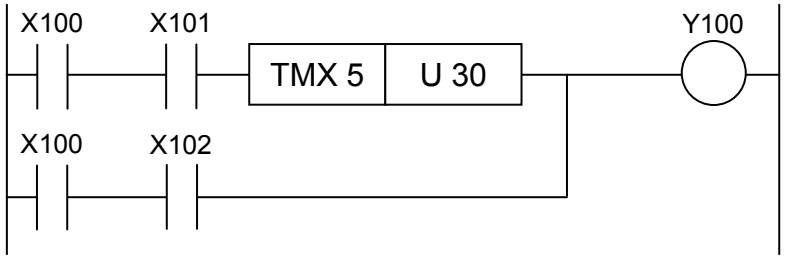
■ Examples for correcting invalid programs

- The invalid programs above can be respectively corrected in the following manner.

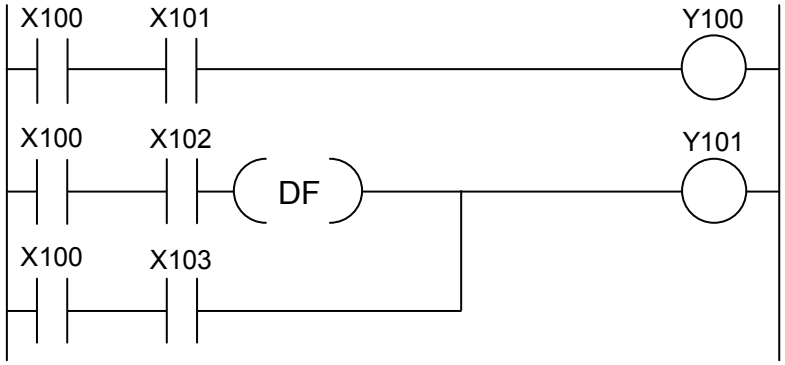
<Program for correcting Example 1>



<Program for correcting Example 2>

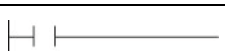
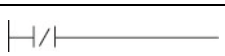
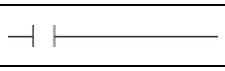
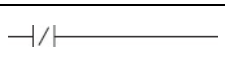
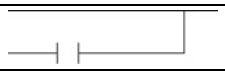
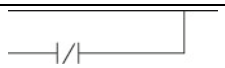
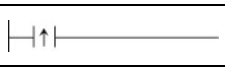
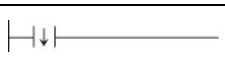
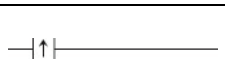
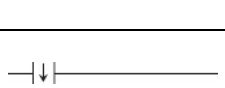
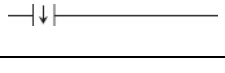
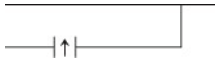
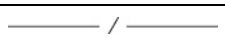
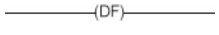
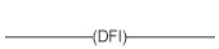
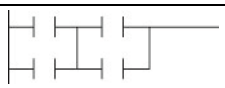
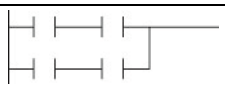
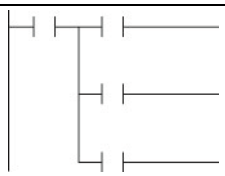
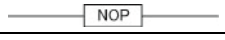


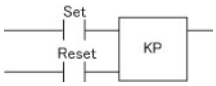
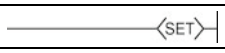
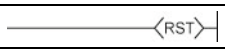
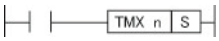
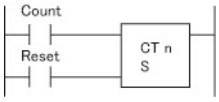
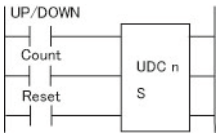
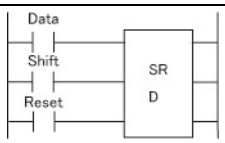
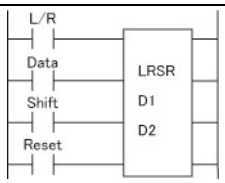
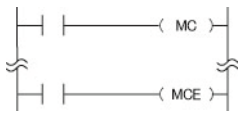
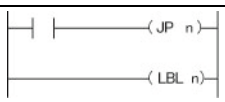
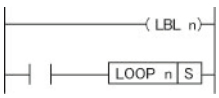
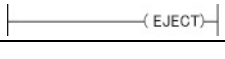
<Program for correcting Example 3>

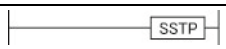
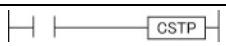
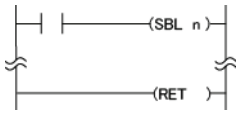
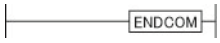


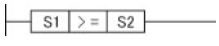
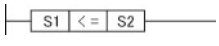
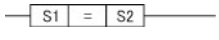
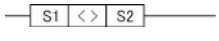
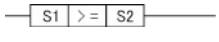
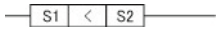
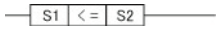
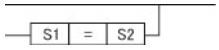
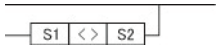
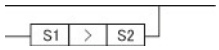
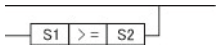
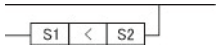
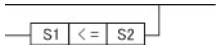
5
List of Instructions

5-1 List of Basic Instructions

Name	Mnemonic	Symbol	Overview of Functions	Page
Sequence Basic Instructions				
Start	ST		Begins a logic operation with a Form A (normally open) contact.	2-2
Start Not	ST/		Begins a logic operation with a Form B (normally closed) contact.	2-2
Out	OT		Outputs the operated result to the specified output.	2-2
AND	AN		Connects a Form A (normally open) contact serially.	2-2
AND Not	AN/		Connects a Form B (normally closed) contact serially.	2-2
OR	OR		Connects a Form A (normally open) contact in parallel.	2-2
OR Not	OR/		Connects a Form B (normally closed) in parallel.	2-2
Leading Contact Instruction	ST↑		Begins a logic operation only for one scan when the leading edge of the trigger is detected.	2-5
Trailing Contact Instruction	ST↓		Begins a logic operation only for one scan when the trailing edge of the trigger is detected.	2-5
Leading Contact Instruction	AN↑		Connects a Form A (normally open) contact serially only for one scan when the leading edge of the trigger is detected.	2-5
Trailing Contact Instruction	AN↓		Connects a Form A (normally open) contact serially only for one scan when the trailing edge of the trigger is detected.	2-5
Leading Contact Instruction	OR↑		Connects a Form A (normally open) contact in parallel only for one scan when the leading edge of the trigger is detected.	2-5
Trailing Contact Instruction	OR↓		Connects a Form A (normally open) contact in parallel only for one scan when the trailing edge of the trigger is detected.	2-5
Not	/		Inverts the operated result up to this instruction.	2-7
Leading Edge Differential	DF		Turns on the contact only for one scan when the leading edge of the trigger is detected.	2-8
Trailing Edge Differential	DF/		Turns on the contact only for one scan when the trailing edge of the trigger is detected.	2-8
Leading Edge Differential (Initial Execution Type)	DFI		Turns on the contact only for one scan when the leading edge of the trigger is detected. The leading edge detection is possible on the first scan.	2-8
AND Stack	ANS		Connects the multiple instruction blocks serially.	2-14
OR Stack	ORS		Connects the multiple instruction blocks in parallel.	2-15
Push Stack	PUSHS		Stores the operated result up to this instruction.	2-16
Read Stack	RDS		Reads the operated result stored by the PSHS instruction.	2-16
Pop Stack	POPS		Reads and clears the operated result stored by the PSHS instruction.	2-16
Nop	NOP		No operation.	2-19
Leading Edge Out	↑OT		Outputs the operated result to the specified output only for one scan when the leading edge of the trigger is detected (for pulse relay P).	2-20
Trailing Edge Out	↓OT		Outputs the operated result to the specified output only for one scan when the trailing edge of the trigger is detected (for pulse relay P).	2-20

Name	Mnemonic	Symbol	Overview of Functions	Page
Keep	KP		Turns on at set trigger and holds until reset trigger turns on.	2-21
Set	SET		Output is set to and held at on.	2-22
Reset	RST		Output is set to and held at off.	2-22
Alternative Out	ALT		Inverts the output condition (on/off) each time the leading edge of the trigger is detected.	2-24
Basic function instructions				
Timer	TM		On-delay timer; Decrements at the specified time S, and turns on the timer contact when the elapsed value reaches zero. S is specified by 32-bit data (U1 to U4294967295). TMS: in 0.1 ms unit TML: in 1 ms unit TMR: in 0.01 s unit TMX: in 0.1 s unit TMY: in 1 s unit	2-25
Unsigned 32-bit Incremental Auxiliary Timer	SPTM		Operates as an on-delay timer in 0.01 s unit.	2-33
Down Counter	CT		Decrements from the preset value S. Turns on the counter contact when the elapsed value reaches zero. S is specified by 32-bit data (U1 to U4294967295).	2-37
Up/Down Counter	UDC		Increments or decrements from the preset value S based on up/down input. To be used with a comparison instruction described immediately after.	2-44
Shift Register	SR		The content of the word device specified by D is shifted to the left by 1 bit.	2-47
Left/Right Shift Register	LRSR		The area specified by D1 to D2 is shifted either to the left or to the right by 1 bit.	2-51
Master Control Relay	MC		When execution conditions are off, output between MC and MCE are turned off. When execution conditions are on, program between MC and MCE is executed.	2-54
Master Control Relay End	MCE			2-54
Jump	JP		When execution conditions are on, the program jumps to the LBL instruction of the same number, and continues from there.	2-59
Label	LBL			2-59
Loop	LOOP		When execution conditions are on, the program jumps to the LBL instruction of the same number and continues from there. The number of repetitive operations is specified by S.	2-64
Label	LBL			2-64
End	ED		The operation of program is ended. Indicates the end of a main program.	2-69
End Program Block	EDPB		Ends a program block (PB).	2-70
Conditional End	CNDE		The operation of program is ended when execution conditions are on.	2-71
Eject	EJECT		Adds page break for use when printing.	2-72

Name	Mnemonic	Symbol	Overview of Functions	Page
Step Ladder Instructions				
Start Step	SSTP		The start of program “n” for process control.	2-73
Next Step	NSTL		Starts the specified process “n” and clears the process currently being started. (Executed in every scan)	2-73
Clear Step	CSTP		Clears the process "n" currently being started.	2-73
Step End	STPE		End of step ladder area	2-73
Block Clear	ZRST		Clears the processes "n1" to "n2" currently being started.	2-82
Subroutine Instructions				
Subroutine Call	CALL		When execution conditions are on: Subroutine program is executed. When execution conditions are off: Subroutine program is not executed. Output within the subroutine is held.	2-84
Subroutine Label	SBL		When the CALL instruction is executed, the program jumps to the subroutine label that has the label of the same number, and executes the subroutine. At the RET instruction, the program returns to the address of the main program and continues.	2-84
Subroutine Return	RET			2-84
Local Subroutine Call (Output Off Type)	FCAL		When execution conditions are on: Subroutine program is executed. When execution conditions are off: Subroutine program is not executed. Output within the subroutine is cleared.	2-88
Subroutine Call (with PB No. Specification)	ECAL		With respect to a subroutine within the program block PBn, When execution conditions are on: Program jumps to the subroutine. When execution conditions are off: Subroutine program is not executed.	2-90
Subroutine Call (Output Off Type) (with PB No. Specification)	EFCAL		With respect to a subroutine within the program block PBn, When execution conditions are on: Program jumps to the subroutine. When execution conditions are off: Subroutine program is not executed. Output within the subroutine is cleared.	2-92
Commenting Instructions				
Comment Out	COMOUT		Comments out instructions between the "COMOUT" instruction and the "ENDCOM" instruction.	2-94
Comment Out End	ENDCOM			2-94

Name	Operation unit	Mnemonic	Symbol	Overview of Functions	Page
Data comparison instructions					
Data Comparison (Start)	US, SS, UL, SL, SF, DF	ST=		Begins a logic operation at a conductive contact in the comparative condition “S1 = S2”.	2-95
		ST<>		Begins a logic operation at a conductive contact in the comparative condition “S1 ≠ S2”.	2-95
		ST>		Begins a logic operation at a conductive contact in the comparative condition “S1 > S2”.	2-95
		ST>=		Begins a logic operation at a conductive contact in the comparative condition “S1 ≧ S2”.	2-95
		ST<		Begins a logic operation at a conductive contact in the comparative condition “S1 < S2”.	2-95
		ST<=		Begins a logic operation at a conductive contact in the comparative condition “S1 ≦ S2”.	2-95
Data Comparison (AND)	US, SS, UL, SL, SF, DF	AN=		Connects serially a conductive contact in the comparative condition “S1 = S2”.	2-95
		AN<>		Connects serially a conductive contact in the comparative condition “S1 ≠ S2”.	2-95
		AN>		Connects serially a conductive contact in the comparative condition “S1 > S2”.	2-95
		AN>=		Connects serially a conductive contact in the comparative condition “S1 ≧ S2”.	2-95
		AN<		Connects serially a conductive contact in the comparative condition “S1 < S2”.	2-95
		AN<=		Connects serially a conductive contact in the comparative condition “S1 ≦ S2”.	2-95
Data Comparison (OR)	US, SS, UL, SL, SF, DF	OR=		Connects in parallel a conductive contact in the comparative condition “S1 = S2”.	2-95
		OR<>		Connects in parallel a conductive contact in the comparative condition “S1 ≠ S2”.	2-95
		OR>		Connects in parallel a conductive contact in the comparative condition “S1 > S2”.	2-95
		OR>=		Connects in parallel a conductive contact in the comparative condition “S1 ≧ S2”.	2-95
		OR<		Connects in parallel a conductive contact in the comparative condition “S1 < S2”.	2-95
		OR<=		Connects in parallel a conductive contact in the comparative condition “S1 ≦ S2”.	2-95

5-2 List of High-level Instructions

Name	Operation unit	Mnemonic		Operand(s)	Overview of Functions	Execution condition		Page
						Level	↑	
Data Compare Instructions								
Data Compare	US, SS, UL, SL, SF, DF	CMP	(P)	S1, S2	S1 and S2 are compared, and the result is output to system relays SRA to SRC. (S1) > (S2) → SRA:ON (S1) = (S2) → SRB:ON (S1) < (S2) → SRC:ON	●	●	3-2
Band Compare	US, SS, UL, SL, SF, DF	WIN	(P)	S1, S2, S3	The value of S1 is compared to the lower limit S2 and to the upper limit S3, and the result is output to system relays SRA to SRC. (S1) > (S3) → SRA:ON (S2) ≦ (S1) ≦ (S3) → SRB:ON (S1) < (S2) → SRC:ON	●	●	3-5
Data Transfer Instructions								
Data transfer	US, SS, UL, SL, SF, DF	MV	(P)	S, D	(S) → (D)	●	●	3-8
Inversion and Transfer	US, SS, UL, SL, SF, DF	MV/	(P)	S, D	$\overline{(S)}$ → (D)	●	●	3-10
Block Transfer	US, SS, UL, SL, SF, DF	BKMV	(P)	S1, S2, D	Data between S1 and S2 are transferred to the area starting with D.	●	●	3-12
Block Copy	US, SS, UL, SL, SF, DF	COPY	(P)	S, D1, D2	Data of S are transferred to all the areas between D1 and D2.	●	●	3-14
Reset	US, SS, UL, SL, SF, DF	RST	(P)	D	Area in D is reset to "0".	●	●	3-16
Bit Block Transfer	US, SS, UL, SL, SF, DF	BTM	(P)	S1, S2, D	Data between the bit addresses S1 and S2 are transferred to the area ending with the bit specified by D.	●	●	3-18
Block Clear	bit	ZRST	(P)	D1, D2	The area specified by bit addresses D1 and D2 is reset to "0".	●	●	3-20
Digit Data Transfer	–	DGT	(P)	S, S1, n, D, D1	Data of "n" digits from the digits specified by S and S1 are stored, starting with the digits specified by D and D1.	●	●	3-22
Data Exchange	US, SS, UL, SL, SF, DF	XCH	(P)	D1, D2	(D1)→(D2) , (D2)→(D1)	●	●	3-25
Exchange of Higher Bytes and Lower Bytes	US, SS	SWAP	(P)	S, D	Higher bytes and lower byte of data in S are exchanged and stored in D.	●	●	3-27
Index Register Operation Instructions								
Index Register Backup	UL, SL	PUSHIX	(P)	D	(I0) to (IE) → (D) to (D＋29)	●	●	3-28
Index Register Recovery	UL, SL	POPIX	(P)	S	(S) to (S＋29) → (I0) to (IE)	●	●	3-29

Name	Operation Unit	Mnemonic		Operand (s)	Overview of Functions	Execution Condition		Page
						Level	↑	
Arithmetic Operation Instructions								
Addition	US, SS, UL, SL, SF, DF	ADD	(P)	S1, S2, D	(S1) + (S2) → (D)	●	●	3-30
Subtraction	US, SS, UL, SL, SF, DF	SUB	(P)	S1, S2, D	(S1) - (S2) → (D)	●	●	3-32
Multiplication	US, SS, UL, SL, SF, DF	MUL	(P)	S1, S2, D	(S1) × (S2) → (D, D+n) n varies by operation unit.	●	●	3-34
Division	US, SS, UL, SL, SF, DF	DIV	(P)	S1, S2, D	(S1) / (S2) → Quotient (D)	●	●	3-36
Division	US, SS, UL, SL	DIVMOD	(P)	S1, S2, D	(S1) / (S2) → Quotient (D), Residue (D+n) n varies by operation unit.	●	●	3-38
Increment	US, SS, UL, SL, SF, DF	INC	(P)	D	(D) + 1 → (D)	●	●	3-40
Decrement	US, SS, UL, SL, SF, DF	DEC	(P)	D	(D) + 1 → (D)	●	●	3-41
Arithmetic Instructions (BCD)								
BCD Data Addition	US, UL	BCDADD	(P)	S1, S2, D	(S1) + (S2) → (D)	●	●	3-42
BCD Data Subtraction	US, UL	BCDSUB	(P)	S1, S2, D	(S1) - (S2) → (D)	●	●	3-44
BCD Data Multiplication	US, UL	BCDMUL	(P)	S1, S2, D	(S1) × (S2) → (D, D+n) n varies by operation unit.	●	●	3-46
BCD Data Division	US, UL	BCDDIV	(P)	S1, S2, D	(S1) / (S2) → Quotient (D), Residue (D+n) n varies by operation unit.	●	●	3-48
BCD Data Increment	US, UL	BCDINC	(P)	D	(D) + 1 → (D)	●	●	3-50
BCD Data Decrement	US, UL	BCDDEC	(P)	D	(D) + 1 → (D)	●	●	3-52
Boolean instructions								
Logical Conjunction	US, SS, UL, SL	AND	(P)	S1, S2, D	(S1) ∧ (S2) → (D)	●	●	3-54
Logical Disjunction	US, SS, UL, SL	OR	(P)	S1, S2, D	(S1) ∨ (S2) → (D)	●	●	3-56
Exclusive OR	US, SS, UL, SL	XOR	(P)	S1, S2, D	{(S1) ∧ $\overline{(S2)}$ ∨ $\overline{(S1)}$ ∧ (S2)} → (D)	●	●	3-58
Exclusive NOR	US, SS, UL, SL	XNR	(P)	S1, S2, D	{(S1) ∧ (S2) ∨ $\overline{(S1)}$ ∧ $\overline{(S2)}$} → (D)	●	●	3-60
Combination	US, SS, UL, SL	COMB	(P)	S1, S2, S3, D	{(S1) ∧ (S3)} ∨ {(S2) ∧/(S3)} → (D)	●	●	3-62

Name	Operation Unit	Mnemonic		Operand (s)	Overview of Functions	Execution Condition		Page
						Level	↑	
String Conversion Instructions								
Block Check Code Calculation	US, SS	BCC	(P)	S1, S2, S3, D	Calculate the block check code of the data specified by S2 and S3, and store the result in the area starting with D. The calculation method is specified by S1.	●	●	3-64
CRC Code Calculation	US, SS	CRC	(P)	S1, S2, S3, D	Calculate the CRC code of the data specified by S2 and S3, and store the result in the area starting with D. The calculation method is specified by S1.	●	●	3-67
Conversion: HEX → Hexadecimal ASCII	US, SS, UL, SL	HEXA	(P)	S1, S2, D	Convert the hexadecimal data specified by S1 and S2 into ASCII code, and store the result in the area starting with D. Example) HABCD→H 42 41 44 43 B A D C	●	●	3-70
Conversion: Hexadecimal ASCII → HEX	US, SS, UL, SL	AHEX	(P)	S1, S2, D	Convert the ASCII code specified by S1 and S2 into hexadecimal data, and store the result in the area starting with D. Example) H 44 43 42 41→HCDAB D C B A	●	●	3-72
Conversion: BCD → Decimal ASCII	US, UL	BCDA	(P)	S1, S2, D	Convert the BCD data specified by S1 and S2 into ASCII code, and store the result in the area starting with D. The conversion method is specified by S2. Example 1) H1234→H 32 31 34 33 2 1 4 3 Example 2) H1234→H 34 33 32 31 4 3 2 1	●	●	3-75
Conversion: Decimal ASCII → BCD	US, UL	ABCD	(P)	S1, S2, D	Convert the ASCII code specified by S1 and S2 into BCD data, and store the result in the area starting with D. The conversion method is specified by S2. Example 1) H 34 33 32 31→H 3412 4 3 2 1 Example 2) H 34 33 32 31→H 1234 4 3 2 1	●	●	3-78
Conversion: BIN → Decimal ASCII	US, SS, UL, SL	BINA	(P)	S1, S2, D	Convert the BIN data expressing a decimal figure specified by S1 into ASCII code, and store the result in the area starting with D. No. of bytes for the conversion result is specified by S2. Example) K-100→H 30 30 31 2D 20 20 0 0 1 —	●	●	3-82
Conversion: Decimal ASCII → BIN	US, SS, UL, SL	ABIN	(P)	S1, S2, D	Convert the ASCII code starting with S1 into BIN data expressing a decimal figure, and store the result in the area starting with D. No. of bytes to be converted is specified by S2. Example) H 30 30 31 2D 20 20 → K-100 0 0 1 —	●	●	3-84
Conversion: BIN → ASCII	US, SS, UL, SL, SF, DF	BTOA	(P)	S1, S2, N, D	Convert the BIN data stored in the area starting with S2 into ASCII data, and store the result in the area starting with D. The conversion method is specified by S1 and N.	●	●	3-86
Conversion: ASCII → BIN	US, SS, UL, SL, SF, DF	ATOB	(P)	S1, S2, N, D	Convert the ASCII data stored in the area starting with S2 into BIN data, and store the result in the area starting with D. The conversion method is specified by S1 and N.	●	●	3-98
ASCII Data Check	US, SS, UL, SL, SF, DF	ACHK	(P)	S1, S2, N	Check that ASCII data can be normally converted by the ATOB instruction.	●	●	3-110

Name	Operation Unit	Mnemonic		Operand (s)	Overview of Functions	Execution Condition		Page
						Level	↑	
Conversion Instructions								
Data Inversion	US, SS, UL, SL	INV	(P)	D	Logically invert each bit of data in D, and store the result in D.	●	●	3-112
Sign Inversion	SS, SL	NEG	(P)	S, D	Invert the sign of data in S, and store the result in D.	●	●	3-113
Absolute Value	US, SS, UL, SL	ABS	(P)	S, D	Calculate the absolute value of data in S, and store the result in D.	●	●	3-114
Sign Extension	US, SS	EXT	(P)	S, D	While maintaining the sign of 16-bit BIN data in S, convert the data into 32-bit BIN data, and store the result in (D+1, D).	●	●	3-115
Conversion: BCD Data	US, UL	BCD	(P)	S, D	Convert the BIN data specified by S into BCD data, and store the result in the area starting with D. Example) K100→H100	●	●	3-116
Conversion: BCD → BIN	US, UL	BIN	(P)	S, D	Convert the BCD data specified by S into BIN data, and store the result in the area starting with D. Example) H100→K100	●	●	3-117
Decoding	–	DECO	(P)	S, n, D	Decode a part of the data in S, and store the result in the area starting with D. The part to be converted is specified by "n".	●	●	3-118
7-Segment Decoding	US	SEGT	(P)	S, D	Convert the data in S for use in a 7-segment display, and store the result in (D+1, D).	●	●	3-120
Encoding	–	ENCO	(P)	S, n, D	Encode a part of the data in S, and store the result in the area starting with D. The part to be converted is specified by "n".	●	●	3-122
Digit Unification	US	UNIT	(P)	S, n, D	Unify the lowest digits of 16-bit data for "n" words starting with S, and store the result in D.	●	●	3-124
Digit Disintegration	US	DIST	(P)	S, n, D	Disintegrate each digit of the data in S, and store the result in the respective lowest digit of 16-bit data starting with D.	●	●	3-125
Byte Data Unification	US, UL	BUNI	(P)	S, n, D	Unify the lower bytes of 16-bit data for "n" words starting with S, and store the result in D.	●	●	3-126
Byte Data Disintegration	US, UL	BDIS	(P)	S, n, D	Disintegrate data in the area starting with S into bytes, and store the result in the respective lower byte of 16-bit data starting with D.	●	●	3-128
Conversion: Binary → Gray Code	US, UL	GRY	(P)	S, D	Convert the BIN data stored in S into gray code data, and store the result in the area starting with D.	●	●	3-130
Conversion: Gray Code → BIN	US, UL	GBIN	(P)	S, D	Convert the gray code data stored in S into BIN data, and store the result in the area starting with D.	●	●	3-132
Conversion: Bit Line → Bit Column	US	COLM	(P)	S, n, D	The values of Bits 0 to 15 in S are stored in the respective Bit "n" of D to D+15.	●	●	3-134
Conversion: Bit Column → Bit Line	US	LINE	(P)	S, n, D	The value of Bit "n" in S to S+15 is stored in Bit 0 to 15 of D.	●	●	3-136

Name	Operation Unit	Mnemonic		Operand(s)	Overview of Functions	Execution Condition		Page
						Level	↑	
Data-shift Instructions								
n-bit Right Shift	US, SS, UL, SL	SHR	(P)	D, n	Shifts the “n” bits of D to the right.	●	●	3-138
n-bit Left Shift	US, SS, UL, SL	SHL	(P)	D, n	Shifts the “n” bits of D to the left.	●	●	3-140
n-digit Right Shift	US, SS, UL, SL	BSR	(P)	D, n	Shifts the “n” digits of D to the right.	●	●	3-142
n-digit Left Shift	US, SS, UL, SL	BSL	(P)	D, n	Shifts the “n” digits of D to the left.	●	●	3-144
n-bit Right Shift of Multiple Devices	bit	BITR	(P)	D1, D2, n	Shifts the “n” bits of the area D1 to D2 to the right.	●	●	3-146
n-bit Left Shift of Multiple Devices	Bit	BITL	(P)	D1, D2, n	Shifts the “n” bits of the area D1 to D2 to the left.	●	●	3-148
n-word Right Shift of a Block Area	US, SS	WSHR	(P)	D1, D2, n	Shifts the “n” words of the area D1 to D2 to the right.	●	●	3-150
n-word Left Shift of a Block Area	US, SS	WSHL	(P)	D1, D2, n	Shifts the “n” words of the area D1 to D2 to the left .	●	●	3-152
Data Rotation Instructions								
Right Rotation of Data	US, UL	ROR	(P)	D, n	Rotates the “n” bits of D to the right.	●	●	3-154
Left Rotation of Data	US, UL	ROL	(P)	D, n	Rotates the “n” bits of D to the left.	●	●	3-156
Right Rotation of Data with Carry-Flag Data	US, UL	RCR	(P)	D, n	Rotates the "n" bits of the area D, added with the CY flag (SR9), to the right.	●	●	3-158
Left Rotation of Data with Carry-Flag Data	US, UL	RCL	(P)	D, n	Rotates the "n" bits of the area D, added with the CY flag (SR9), to the left.	●	●	3-160
Data Buffer Instructions								
Data Table Shift-out and Compress	US, SS	CMPR	(P)	D1, D2, D3	Transfer D2 to D3. Any parts of the data between D1 and D2 that are 0 are compressed, and shifted in order toward D2.	●	●	3-162
Data Table Shift-in and Compress	US, SS	CMPW	(P)	S, D1, D2	Transfer S to D1. Any parts of the data between D1 and D2 that are 0 are compressed, and shifted in order toward D2.	●	●	3-164
Buffer Definition	US, SS	DEFBUF	(P)	n, D	The "n" words of area starting with D is defined in the data buffer area used by the FIFR/BUFW/LIFR instruction.	●	●	3-166
Data Read (First-In-First-Out)	US, SS	FIFR	(P)	S, D	Data are read from the area specified by the read pointer of the FIFO buffer starting with S, and stored in D.	●	●	3-168
Data Write	US, SS	BUFW	(P)	S, D	The value of S is stored in the area specified by the read pointer of the buffer starting with D.	●	●	3-170
Data Read (Last-In-First-Out)	US, SS	LIFR	(P)	S, D	Data are read from the area specified by the LIFO pointer of the LIFO buffer starting with S, and stored in D.	●	●	3-172

Name	Operation Unit	Mnemonic		Operand(s)	Overview of Functions	Execution Condition		Page
						Level	↑	
Bitwise boolean instruction								
Bit Inversion	US	BTI	(P)	D, n	Invert the value of bit position “n” of the data in “D”.	●	●	3-174
Bit Test	US	BTT	(P)	D, n	When the value of bit position “n” of the data in “D” is '0', the system relay (SRB) is turned on. When the value is '1', the (SRB) is turned off.	●	●	3-175
Carry-flag Set	−	STC	(P)		Turn on the CY flag (SR9).	●	●	3-177
Carry-flag Reset	−	CLC	(P)		Turn off the CY flag (SR9).	●	●	3-178
String Operation Instructions								
Conversion: Character Constant → ASCII Code	−	SSET	(P)	S, D	Convert the character constant specified by S into ASCII code, and store the result in the area starting with D.	●	●	3-179
String Compare	−	SCMP	(P)	S1, S2	The strings in S1 and S2 are compared, and the result is output to system relays SRA to SRC. (S1) > (S2) → SRA:ON (S1) = (S2) → SRB:ON (S1) < (S2) → SRC:ON	●	●	3-182
String Addition	−	SADD	(P)	S1, S2, D	Connect the string in S2 to the string in S1, and store the result in D.	●	●	3-184
Obtainment of String Length	−	LEN	(P)	S, D	The no. of characters, stored in the data of the string length table starting with S, is stored in D.	●	●	3-185
String Search	−	SSRC	(P)	S1, S2, D	The data of the string length table starting with S2 are searched for Data S1, and the relative position of the first match is stored in D.	●	●	3-186
Takeout of the Right Side of a String	−	RIGHT	(P)	S1, S2, D	The string with the no. of characters as specified by S2 is taken out from the right side (with a larger device address) of the string length table starting with S1, and stored in D.	●	●	3-188
Takeout of the Left Side of a String	−	LEFT	(P)	S1, S2, D	The string with the no. of characters as specified by S2 is taken out from the left side (with a smaller device address) of the string length table starting with S1, and stored in D.	●	●	3-190
Data Read from a Given Position in the String	−	MIDR	(P)	S1, S2, S3, D	The strings of the number as specified by S3, starting from the byte position as specified by S2 in the string length table starting with S1, are read and stored in D.	●	●	3-192
Rewrite from a Given Position in the String	−	MIDW	(P)	S1, S2, D, n	The string data with the no. of characters as specified by S2 in the string table starting with S1, are read and stored in D, starting the nth-byte position.	●	●	3-194
Replacement of a String	−	SREP	(P)	S, D, p, n	The string table starting with D is replaced with the string in S1. The position and the no. of characters for replacement are respectively specified by "p" and "n".	●	●	3-197

Name	Operation Unit	Mnemonic		Operand (s)	Overview of Functions	Execution Condition		Page
						Level	↑	
Data Processing Instructions								
Data Search	US, SS, UL, SL, SF, DF	SRC	(P)	S1, S2, S3, D	The data stored in S2 and S3 are searched for data S1, and the number of matches and the relative position of the first match are stored in the area starting with D.	●	●	3-200
ON Bits Count	US, UL	BCU	(P)	S, D	The number of ON bits in the data of S is counted and stored in D.	●	●	3-203
Obtainment of the Max. Value	US, SS, UL, SL, SF, DF	MAX	(P)	S1, S2, D	The data stored in S1 and S2 are searched for the max. value, and the resulting max. value and the relative position of the first detection are stored in the area starting with D.	●	●	3-205
Obtainment of the Min. Value	US, SS, UL, SL, SF, DF	MIN	(P)	S1, S2, D	The data stored in S1 and S2 are searched for the min. value, and the resulting min. value and the relative position of the first detection are stored in the area starting with D.	●	●	3-209
Obtainment of the Total and the Mean	US, SS, UL, SL, SF, DF	MEAN	(P)	S1, S2, D	The total and the mean of the data stored in S1 and S2 are calculated, and stored in the area starting with D.	●	●	3-214
Sort	US, SS, UL, SL, SF, DF	SORT	(P)	S1, S2, S3	Data stored in S1 and S2 are sorted. Sort condition (ascending order or descending order) is specified by S3.	●	●	3-218
Linearization	US, SS, UL, SL, SF, DF	SCAL	(P)	S1, S2, D	Prepare a data table to be used for scaling (linearization) in the area starting with S2. Linearize the data in S1 (input value X), and store the result in D (output value Y).	●	●	3-221
Event Count Instructions								
Instruction to Count the No. of Events	UL	EVENTC		S1, S2, n, D	Count the number of ONs in the data of "n" bits as specified by S1, and store the result in the area starting with D.	●	–	3-224
Instruction to Count the Time of Events	UL	EVENTT		S1, S2, n, D	Count the time, in seconds, of ON in the data of "n" bits as specified by S1, and store the result in the area starting with D.	●	–	3-227
PID Instruction								
PID Operation	–	PID		S	Execute PID operation in accordance with the parameters of S to S+29. The operation result is stored in (S+3) as the manipulated value (MV).	(Note 1)	(Note 1)	3-230
PID Operation: PWM Output Available	–	EZPID		S1, S2, S3, S4	Execute PID operation in accordance with the parameters of S1, S2, S3 to S3+3, and S4 to S4+29. The operation result is stored in S4 as the manipulated value (MV). Describe the OUT instruction immediately after, and the PWM output becomes available.	(Note 1)	(Note 1)	3-235

(Note 1) For the PID instruction and the EZPID instruction, execution conditions must be maintained in the ON state, in order to hold the operation data.

Name	Operation Unit	Mnemonic		Operand (s)	Overview of Functions	Execution Condition		Page
						Level	↑	
Data Control Instructions								
Data Revision Detection	US, SS, UL, SL, SF, DF	DTR	(P)	S, D	Detect data revision in S, and reflect the result into the CY flag (SR9). D is used as the area to hold the previous value data.	●	●	3-244
Ramp Output	US, SS, UL, SL, SF, DF	RAMP		S1, S2, S3, D	Executes the linear output for the specified time S3 from the specified initial value S1 to the target value S2.	(Note 1)	(Note 1)	3-246
Upper and Lower Limit Control	US, SS, UL, SL, SF, DF	LIMT	(P)	S1, S2, S3, D	Output is given based on whether the input value S3 falls within the range delimited by the upper limit S1 and the lower limit S2, and the result is stored in D. In case of (S1) > (S3), (S1)→(D). In case of (S2) < (S3), (S2)→(D). In case of (S1) ≦ (S3) ≦ (S2), (S3)→(D).	●	●	3-248
Deadband Control	SS, SL, SF, DF	BAND	(P)	S1, S2, S3, D	Output is given based on whether the input value S3 falls within the deadband delimited by S1 and S2, and the result is stored in D. In case of (S1) > (S3), (S3)-(S1)→(D). In case of (S2) > (S3), (S3)-(S2)→(D). In case of (S1) ≦ (S3) ≦ (S2), 0→(D).	●	●	3-250
Zone Control	US, SS, UL, SL, SF, DF	ZONE	(P)	S1, S2, S3, D	Depending on the range of the input value S3, an offset value as specified either by S1 or by S2 is added, and the result is stored in D. In case of (S3) < 0, (S3)+(S1)→(D) In case of (S3) = 0, 0→(D) In case of (S3) > 0, (S3)+(S2)→(D)	●	●	3-252
Time Constant Processing Instructions								
Time Constant Processing	–	FILTR		S1, S2, S3, D	The data filtering process is executed, as specified by S1 and S2, and the result is stored in D.	(Note 1)	(Note 1)	3-254
Floating Point Real Number Function Instructions								
Sine Operation	SF, DF	SIN	(P)	S, D	SIN(S)→(D)	●	●	3-256
Cosine Operation	SF, DF	COS	(P)	S, D	COS(S)→(D)	●	●	3-257
Tangent Operation	SF, DF	TAN	(P)	S, D	TAN(S)→(D)	●	●	3-258
Arcsine Operation	SF, DF	ASIN	(P)	S, D	$SIN^{-1}(S)$→(D)	●	●	3-259
Arccosine Operation	SF, DF	ACOS	(P)	S, D	$COS^{-1}(S)$→(D)	●	●	3-260
Arctangent Operation	SF, DF	ATAN	(P)	S, D	$TAN^{-1}(S)$→(D)	●	●	3-261
Conversion: Coordinate Data → Angle Radian	SF, DF	ATAN2	(P)	S1, S2, D	The arctangent is calculated from S1 (X coordinate) and S2 (Y coordinate), and stored in D.	●	●	3-262
Hyperbolic Sine Operation	SF, DF	SINH	(P)	S, D	SINH(S)→(D)	●	●	3-263
Hyperbolic Cosine Operation	SF, DF	COSH	(P)	S, D	COSH(S)→(D)	●	●	3-264

Name	Operation Unit	Mnemonic		Operand (s)	Overview of Functions	Execution Condition		Page
						Level	↑	
Hyperbolic Tangent Operation	SF, DF	TANH	(P)	S, D	TANH(S)→(D)	●	●	3-265
Exponential Operation	SF, DF	EXP	(P)	S, D	EXP(S)→(D)	●	●	3-266
Natural Logarithmic Operation	SF, DF	LN	(P)	S, D	LN(S)→(D)	●	●	3-267
Common Logarithmic Operation	SF, DF	LOG	(P)	S, D	LOG(S)→(D)	●	●	3-268
Power Operation	SF, DF	PWR	(P)	S1, S2, D	(S1)^(S2) → (D)	●	●	3-269
Square Root Operation	SF, DF	SQR	(P)	S, D	SQR(S) → (D)	●	●	3-270

Note 1) For the RAMP instruction and the FILTER instruction, execution conditions must be maintained in the ON state, in order to hold the operation data.

Name	Operation Unit	Mnemonic		Operand (s)	Overview of Functions	Execution Condition		Page
						Level	↑	
Floating Point Real Number Conversion Instructions								
Conversion: Degrees → Radian	SF, DF	RAD	(P)	S, D	Angle data [degrees] of S is converted into angle data [radian], and the result is stored in D.	●	●	3-271
Conversion: Radian → Degrees	SF, DF	DEG	(P)	S, D	Angle data [radian] of S is converted into angle data [degrees], and the result is stored in D.	●	●	3-272
Floating Point Real Number Data - Rounding the First Decimal Point Down	SF, DF	FINT	(P)	S, D	Round the first decimal point down for the real number stored in S, and store the result in D.	●	●	3-273
Floating Point Real Number Data - Rounding the First Decimal Point Off	SF, DF	FRINT	(P)	S, D	Round the first decimal point off for the real number stored in S, and store the result in D.	●	●	3-274
Floating Point Real Number Data - Sign Changes (Negative/Positive Conversion)	SF, DF	FNEG	(P)	S, D	The sign of the real number stored in S is converted (negative/positive), and the result is stored in D.	●	●	3-275
Floating Point Real Number Data - Absolute Value	SF, DF	FABS	(P)	S, D	The absolute value of the real number stored in S is calculated, and the result is stored in D.	●	●	3-276
Conversion: Integer → Single Precision Floating Point Real Number Data	US, SS, UL, SL	FLT	(P)	S, D	The integer data stored in S is converted into single precision floating point real number data, and the result is stored in D.	●	●	3-277
Conversion: Single Point Floating Point Real Number Data → Integer (Max. Value Not Exceeding the Original Data)	US, SS, UL, SL	INT	(P)	S, D	The single precision floating point real number stored in S is converted into integer (max. value not exceeding the original data), and the result is stored in D.	●	●	3-279
Conversion: Single Point Floating Point Real Number Data → Integer (Rounding the First Decimal Point Down)	US, SS, UL, SL	FIX	(P)	S, D	The single precision floating point real number stored in S is converted into integer (rounding the first decimal point down), and the result is stored in D.	●	●	3-282
Conversion: Single Precision Floating Point Real Number Data → Integer (Rounding the First Decimal Point Off)	US, SS, UL, SL	ROFF	(P)	S, D	The single precision floating point real number stored in S is converted into integer (rounding the first decimal point off), and the result is stored in D.	●	●	3-285

Name	Operation Unit	Mnemonic		Operand (s)	Overview of Functions	Execution Condition		Page
						Level	↑	
Clock/Calendar Instructions								
Conversion: Time Data (Hours, Minutes and Seconds) → Seconds Data	−	HMSS	(P)	S, D	The BIN data indicating hours, minutes and seconds stored in S to S+2 are converted into seconds, and stored in (D+1, D).	●	●	3-288
Conversion: Seconds Data → Time Data (Hours, Minutes and Seconds)	−	SHMS	(P)	S, D	The seconds data stored in (S+1. S) is converted into hours, minutes and seconds, and stored as BIN data in D to D+2.	●	●	3-289
Clock Addition	−	CADD	(P)	S1, S2, D	The time data stored in S2 to S2+2 (hours, minutes and seconds) is added to the clock data stored in S1 to S1+5 (year, month, day, hours, minutes and seconds), and the result is stored in D to D+5.	●	●	3-290
Clock Subtraction	−	CSUB	(P)	S1, S2, D	The time data stored in S2 to S2+2 (hours, minutes and seconds) is subtracted from the clock data stored in S1 to S1+5 (year, month, day, hours, minutes and seconds), and the result is stored in D to D+5.	●	●	3-291
Clock Data → Seconds Data from the Base Time	−	TMSEC	(P)	S, D	The clock data specified by S is converted into seconds data from the base time (January 1, 2001), and the result is stored in (D+1, D).	●	●	3-292
Seconds Data from the Base Time → Clock Data	−	SECTM	(P)	S, D	The seconds data from the base time (January 1, 2001) as stored in (S+1, S) is converted into clock data, and the result is stored in D to D+5.	●	●	3-293
Setting of Clock/Calendar	−	TIMEWT	(P)	S	The clock data (year, month, day, hours, minutes and seconds) as stored in S to S+6 is set as RTC data for the CPU unit.	●	●	3-294
Special Instructions								
Self-Diagnostic Error Code Set	US	ERR	(P)	n	In a user program, set an optional error code [n]. The self-diagnostic error code is stored in the system data register (SD0). Set the self-diagnostic abnormality flag (SR0).	●	●	3-296
Watchdog Timer Reset	−	WDTRES	(P)		The watchdog timer is reset.	●	●	3-297

Name	Operation Unit	Mnemonic		Operand (s)	Overview of Functions	Execution Condition		Page
						Level	↑	
Communication Instructions								
Specification of a Communication Unit Slot Port	–	UNITSEL	(P)	S1, S2	To be described immediately before the following communication instructions, to specify the targets of execution. GPSEND, GPRECV, SEND, RECV, RDET, PMSET, PMGET	●	●	3-298
General-Purpose Communication Send Instruction	US, SS	GPSEND		S, n, D	Send the data of "n" bytes, stored in the area starting with S, to an external device via the unit's COM port (SCU) or LAN port (ET-LAN).	(Note 1)	(Note 1)	3-300
General-Purpose Communication Receive Instruction	US, SS	GPRECV	(P)	D1, D2	Copy the data received from an external device, via the unit's COM port (SCU) or LAN port (ET-LAN), to the range D1 and D2.	●	●	3-306
MEWTOCOL /MODBUS Master Send Instruction	US, SS	SEND	(P)	S, n, D1, D2, D3	Data are read from the device in the master unit, starting with S, and stored in the address starting with D2 of the partner unit D1. The sent data amount is specified by "n".	●	●	3-312
MODBUS Master Send Instruction : Function Code Specification	US, SS	SEND	(P)	S, n, D1 D2, D3	Data are read from the device in the master unit, starting with S, and stored in the address starting with D2 of the partner unit D1. The MODBUS function code and the unit no. are specified by D1, and the sent data amount is specified by "n".	●	●	3-318
MEWTOCOL /MODBUS Master Receive Instruction	US, SS	RECV	(P)	S1, S2, n D1, D2	Data are read from the address starting with S2 in the partner unit no. S1, and stored in the area starting with D1 in the master unit. The sent data amount is specified by "n".	●	●	3-324
MODBUS Master Receive Instruction : Function Code Specification	US, SS	RECV	(P)	S1, S2, n D1, D2	Data are read from the address starting with S2 in the partner unit no. S1, and stored in the area starting with D1 in the master unit. The MODBUS function code and the unit no. are specified by S1, and the sent data amount is specified by "n".	●	●	3-331
Change of SCU Parameters	–	PMSET		S, n, D	The data stored in the area of "n" words, starting with S, are set as communication parameters for the unit's COM port (SCU).	(Note 2)	(Note 2)	3-337
Obtainment of SCU Parameters	–	PMGET	(P)	S, D	Read the communication parameters that are set in the unit's COM port (SCU), and store them in the area starting with D.	●	●	3-340
Read the ET-LAN Status	–	RDET	(P)	D	Read the statuses of all connections of the unit's LAN port (ET-LAN), and store them in D.	●	●	3-343

(Note 1) For the GPSEND instruction, execution conditions must be set to ON after the instruction is triggered, and until the sending flag turns OFF.

(Note 2) For the PMSET instruction, execution conditions must be set to ON after the instruction is triggered, and until the process in-progress flag turns OFF.

Name	Operation Unit	Mnemonic		Operand (s)	Overview of Functions	Execution Condition		Page
Positioning Unit Control Instructions								
Setting of Positioning Starting Table	−	POSSET	(P)	S1, S2, S3	Describe it immediately before the program to start up positioning, and set the positioning data table to be started up. Specify the slot no., axis no., and table no. respectively in S1, S2 and S3.	●	●	3-345
Obtainment of Axis Status	−	PSTRD	(P)	S1, S2, D	With respect to the axis no. (S2) of the positioning unit attached to the slot no. (S1), read the statuses of key flags as "axis status", and store them in D.	●	●	3-347
Obtainment of Error/Warning in the Positioning Unit	−	PERRD	(P)	S1, S2, D	With respect to the axis no. (S2) of the positioning unit attached to the slot no. (S1), read the error codes and warning codes stored in the notification buffer 1, and store them respectively in D and D1.	●	●	3-349
Unit Control Instructions								
Error and Warning Clear	−	UCLR		S	Clear an error/warning in the unit attached to the slot no. specified by S.	●	●	3-350

Index

F

G

H

I

J

K

L

M

N

O

P

R

S

T

U

W

X

Y

Z

Record of changes

Manual No.	Date	Record of Changes
WUME-FP7CPUPGR-01	JUL.2013	First Edition
WUME-FP7CPUPGR-02	DEC.2013	2nd Edition Revised the description contents of communication instructions UNITSEL,GPSEND, GPRECV, SEND, RECV, PMSET and PMGET along with the addition of a new product Serial Communication Unit and new models of communication cassettes.

Please contact .....

Panasonic Industrial Devices SUNX Co., Ltd.

■ Overseas Sales Division (Head Office): 2431-1 Ushiyama-cho, Kasugai-shi, Aichi, 486-0901, Japan
■ Telephone: +81-568-33-7861 ■ Facsimile: +81-568-33-8591

panasonic.net/id/pidsx/global

About our sale network, please visit our website.

Dec, 2013 PRINTED IN JAPAN

WUME-FP7CPUPGR-02