FANUC Series 30*i*/31*i*/32*i*/35*i*-MODEL B
FANUC Power Motion *i*-MODEL A
FANUC Series 0*i*-MODEL F

PMC Supplemental Programming Manual

Type of applied technical documents

Name	FANUC Series 30*i*/31*i*/32*i*/35*i*-MODEL B FANUC Power Motion *i*-MODEL A FANUC Series 0*i*-MODEL F PMC Programming Manual
Spec. No. /Ed.	B-64513EN/04

Summary of Change

Group	Name/Outline	New, Add, Correct, Delete	Applied Date
Basic Function	Addition of functional instructions for character string instruction and RS-232C control	Add	Feb. 2019
Optional Function			
Unit			
Maintenance Parts			
Notice	Addition of a caution to the window functions Modification in notes on using subroutines		
Correction			
Another	Correction of error (PID functional instruction)		

FANUC Series 30*i*/31*i*/32*i*/35*i*-MODEL B
FANUC Power Motion *i*-MODEL A
FANUC Series 0*i*-MODEL F
PMC Supplemental Programming Manual

Contents

1 SUMMARY

The following functions have been added.

- Add of functional instructions for character string instruction and RS-232C control

This document is a supplemental manual for the above.
Please refer to the following manual about existing functions and operations.

Manual	Spec.
FANUC Series 30*i*/31*i*/32*i*/35*i*-MODEL B FANUC Power Motion *i*-MODEL A FANUC Series 0*i*-MODEL F PMC Programming Manual	B-64513EN / 04

In this document, the following abbreviations are used.

Name	Abbreviation
FANUC Series 30*i*/31*i*/32*i*/35*i*-MODEL B	30*i*/31*i*/32*i*/35*i*-B

2 APPLIED SOFTWARE

The new features will be applied to the following software.

PMC System software

Name	Series	Edition
Series 30*i*/31*i*/32*i*/35*i*-MODEL B PMC Power Motion *i*-MODEL A PMC (A02B-0323-H580#40A7)	40A7	02 or later

CNC System software

Name	Series	Edition
Series 30*i*-MODEL B (A02B-0323-H501#xxxx)	G303, G313, G323, G333, G353	19 or later
Series 31*i*-MODEL B5 (A02B-0326-H501#xxxx)	G423, G424, G433, G483, G4H3	19 or later
Series 31*i*-MODEL B (A02B-0327-H501#xxxx)	G403, G404, G413, G453, G4G3	19 or later
Series 32*i*-MODEL B (A02B-0328-H501#xxxx)	G503, G513, G523	19 or later

FANUC LADDER-III

Name	Drawing number	Edition
FANUC LADDER-III	A08B-9210-J505	8.70 or later
FANUC LADDER-III (10 users)	A08B-9210-J541	8.70 or later
FANUC LADDER-III (20 users)	A08B-9210-J542	8.70 or later
FANUC LADDER-III (Site license)	A08B-9210-J543	8.70 or later
FANUC LADDER-III (Update)	A08B-9210-J544	8.70 or later

3 OVERVIEW OF PMC

3.1 EXECUTION OF SEQUENCE PROGRAM

3.1.1 Structured Sequence Program

3.1.1.1 Notes on using subroutines

Functional instructions list to be noted (1)

- EXIN
- WINDR (low-speed type only)
- WINDW (low-speed type only)
- AXCTL
- RSOPN
- RSRCV
- RSSND
- RSCLS

For the above functional instructions, ACT = 1 must be held until transfer completion information (coil) is set to 1.

When using these functional instructions in subprograms, note the following prohibition:

(1) When one of the above functional instructions is being used within a subprogram and is not yet completed (processing is in progress), the subprogram call is canceled. (ACT for the CALL instruction is set to 0.)

> ⚠ **CAUTION**
> When you operate as in (1) above, the subsequent operation of the above functional instruction is not guaranteed.

(2) When one of the above functional instructions is being used within a subprogram and is not yet completed (processing is in progress), the subroutine is called from another subprogram.

> ⚠ **CAUTION**
> When you operate as in (2) above, because the preceding function is being processed, the subsequent operation of the above functional instruction is not guaranteed.

When a subprogram using the above functional instructions is called from more than one place, exclusive control is required. An example of using the WINDR instruction (low-speed type) is given below.

**** omitted the middle****

Functional instructions list to be noted (2)

- DISPB

In addition, for the above functional instruction, ACT = 1 must be held while using the function.

When using these functional instructions in subprograms, note the following prohibition:

(1) When the above functional instruction is being used within a subprogram, the subprogram call is canceled. (ACT for the CALL instruction is set to 0.)

> ⚠ **CAUTION**
> When you operate as in (1) above,, the subsequent operation of the above functional instruction is not guaranteed.

4 PMC SPECIFICATIONS

4.1 SPECIFICATIONS

4.1.1 Functional instructions (Arranged in Sequence of Instruction Group)

Table 4.1 (a) PMC functional instructions (Arranged in Sequence of Instruction Group)

Instruction group		Instruction Name	SUB No.	Processing	Required memory size (byte)	1st to 5th PMC	0*i*-F PMC/L	DCS PMC
Character string instructions	1	SLEN	461	Calculating length of Character string	16	●	●	●
	2	SLEFT	462	Extracting the left part of Character string	28	●	●	●
	3	SRIGH	463	Extracting the right part of Character string	28	●	●	●
	4	SMID	464	Extracting a part of Character string	32	●	●	●
	5	SCON	465	Concatenating two Character strings	28	●	●	●
	6	SINS	466	Inserting a sub-string in Character string	36	●	●	●
	7	SDEL	467	Deleting a part of Character string	32	●	●	●
	8	SREP	468	Replacing a part of Character string	40	●	●	●
	9	SFIND	469	Finding a sub-string in Character string	24	●	●	●
	10	BTOS	470	Conversion (1-byte signed binary→Character string)	20	●	●	●
	11	WTOS	471	Conversion (2-byte signed binary→Character string)	20	●	●	●
	12	DTOS	472	Conversion (4-byte signed binary→Character string)	20	●	●	●
	13	RTOS	473	Conversion (Single Precision Real→ Character string)	20	◊	×	×
	14	LTOS	474	Conversion (Double Precision Real→ Character string)	24	◊	×	×
	15	STOB	475	Conversion (Character string→1-byte signed binary)	16	●	●	●
	16	STOW	476	Conversion (Character string→2-byte signed binary)	16	●	●	●
	17	STOD	477	Conversion (Character string→4-byte signed binary)	16	●	●	●
	18	STOR	478	Conversion (Character string→Single Precision Real)	16	◊	×	×
	19	STOL	479	Conversion (Character string→Double Precision Real)	16	◊	×	×
	20	BTOSH	480	Conversion (1-byte signed binary→ Hexadecimal character string)	20	●	●	●
	21	WTOSH	481	Conversion (2-byte signed binary→ Hexadecimal character string)	20	●	●	●

Instruction group		Instruction Name	SUB No.	Processing	Required memory size (byte)	1st to 5th PMC	0*i*-F PMC/L	DCS PMC
	22	DTOSH	482	Conversion (4-byte signed binary→ Hexadecimal character string)	20	●	●	●
	23	SHTOB	483	Conversion (Hexadecimal character string→ 1-byte signed binary)	16	●	●	●
	24	SHTOW	484	Conversion (Hexadecimal character string→ 2-byte signed binary)	16	●	●	●
	25	SHTOD	485	Conversion (Hexadecimal character string→ 4-byte signed binary)	16	●	●	●
	26	MOVS	486	Copying Character string	20	●	●	●
	27	SCOMP	487	Comparing two Character strings	20	●	●	●
	28	UPPER	488	Character string conversion (Lower case→ Upper case)	20	●	●	●
	29	LOWER	489	Character string conversion (Upper case→ Lower case)	20	●	●	●
	30	SFILL	490	Filling Character string	24	●	●	●
RS-232C control instructions	1	RSOPN	491	Opening RS-232C channel	24	●	●	×
	2	RSRCV	492	Receiving data from RS-232C channel	28	●	●	×
	3	RSSND	493	Sending data to RS-232C channel	24	●	●	×
	4	RSCLS	494	Closing RS-232C channel	12	●	●	×

(○: Usable, ●: The Extended PMC Ladder Instruction Function, ◊: The Extended Floating Point, Δ: Executed as NOP instruction, ×: Unusable.)

5 LADDER LANGUAGE

5.1 FUNCTIONAL INSTRUCTION

5.1.1 Addresses for Signals Between the PMC and CNC (F, G)

This section describes available numerical data in PMC.

(a) BCD format data
(b) Integer data
(c) Real number data
(d) String data

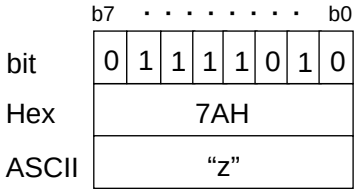
*** *omitted the middle* ***

String data

PMC supports string data that conform to ASCII code.
A single character is represented by one byte.

String data is stored in the PMC address as follows.

Example of the bit pattern representing a character.



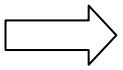
The maximum number of characters in string data is 255 characters.

To use a string data, it is necessary to allocate a memory area which has the size of actual string length plus one byte. For example, when using a 255 characters string data (maximum specification), a 256 bytes memory is necessary.

As for the string data, each character is set to the corresponding byte from the specified address, and a code 00H is set to follow the string data.

Example of the string data “PMC” stored in addresses from R0000 to R0003.

R0000	50H	(“P”)
R0001	4DH	(“M”)
R0002	43H	(“C”)
R0003	00H	(NULL)



These data represent three characters as "PMC".

The available ASCII codes for string data are as follows.

Table 5.1.1(a) ASCII code table

	2	3	4	5	6	7
0	(※1)	0	@	P	‘	p
1	!	1	A	Q	a	q
2	”	2	B	R	b	r
3	#	3	C	S	c	s
4	$	4	D	T	d	t
5	%	5	E	U	e	u
6	&	6	F	V	f	v
7	’	7	G	W	g	w
8	(	8	H	X	h	x
9	)	9	I	Y	i	y
A	*	:	J	Z	j	z
B	+	;	K	[	k	{
C	,	<	L	¥	l	¦
D	ー (※2)	=	M	]	m	}
E	. (※3)	>	N	^	n	~
F	/	?	O	_ (※4)	o	(※1)

(※1)Blank (※2)Minus (※3)Dot (※4)Underscore

NOTE

The code of “00H” is treated as the NULL and the code of “0aH” is treated as the line feed. A character code that is not shown in the ASCII code table will be displayed as a blank on the CNC screen.

5.2 OPERATION INSTRUCTIONS

5.2.1 PID (PID Control : SUB 460)

This instruction executes PID operation.

> **⚠ CAUTION**
> This instruction cannot be used in some version of PMC software. Executing this instruction on the unsupported version of PMC software will raise the warning "WN58 UNSUPPORTED FUNCTION" on PMC alarm screen, and the ladder program is executed excluding this instruction.

**** omitted the middle ****

Parameters

**** omitted the middle ****

(e) Work memory address
Specify a work memory address of 20 bytes length which is used for PID operation. R address will be used typically.
The work memory is used for preserving the progress data of the PID operation. If you specify a nonvolatile memory for the work memory, you have to initialize the area to 0 before the initial execution of the instruction.

> **⚠ CAUTION**
> 1 If the instruction starts working with the work memory that is not initialized to "0", the result of the PID operation will be incorrect.
> 2 Do not modify the contents of the work memory while the instruction is active.
> 3 If you change the work memory address by ladder editor, the new work memory has to be initialized before starting the PID operation. In this case, the PID operation starts from the initial state.

**** omitted below ****

5.3 CHARACTER STRING INSTRUCTIONS

The following table shows the kinds of Character string instructions.

	Instruction Name	SUB No.	Processing
1	SLEN	461	Calculating length of Character string
2	SLEFT	462	Extracting the left part of Character string
3	SRIGH	463	Extracting the right part of Character string
4	SMID	464	Extracting a part of Character string
5	SCON	465	Concatenating two Character strings
6	SINS	466	Inserting a sub-string in Character string
7	SDEL	467	Deleting a part of Character string
8	SREP	468	Replacing a part of Character string
9	SFIND	469	Finding a sub-string in Character string
10	BTOS	470	Conversion (1-byte signed binary→Character string)
11	WTOS	471	Conversion (2-byte signed binary→Character string)
12	DTOS	472	Conversion (4-byte signed binary→Character string)
13	RTOS	473	Conversion (Single Precision Real→Character string)
14	LTOS	474	Conversion (Double Precision Real→Character string)
15	STOB	475	Conversion (Character string→1-byte signed binary)
16	STOW	476	Conversion (Character string→2-byte signed binary)
17	STOD	477	Conversion (Character string→4-byte signed binary)
18	STOR	478	Conversion (Character string→Single Precision Real)
19	STOL	479	Conversion (Character string→Double Precision Real)
20	BTOSH	480	Conversion (1-byte signed binary→Hexadecimal character string)
21	WTOSH	481	Conversion (2-byte signed Binary→Hexadecimal character string)
22	DTOSH	482	Conversion (4-byte signed Binary→Hexadecimal character string)
23	SHTOB	483	Conversion (Hexadecimal character string→1-byte signed binary)
24	SHTOW	484	Conversion (Hexadecimal character string→2-byte signed binary)
25	SHTOD	485	Conversion (Hexadecimal character string→4-byte signed binary)
26	MOVS	486	Copying Character string
27	SCOMP	487	Comparing two Character strings
28	UPPER	488	Character string conversion (Lower case→Upper case)
29	LOWER	489	Character string conversion (Upper case→Lower case)
30	SFILL	490	Filling Character string

5.3.1 SLEN (Calculating length of Character string : SUB 461)

This instruction counts the length of the specified string data and outputs it to the specified address.
If the length of the string data is longer than 255 bytes, 255 is outputted to the result.

NOTE
"00H" at the end of the string data is not counted.

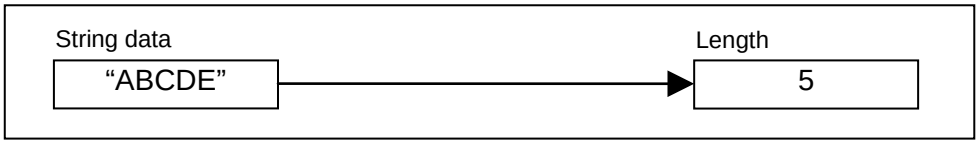


Fig. 5.3.1 (a) Operation of SLEN instruction

Format

The ladder format and the mnemonic format of ths instruction are as follows.

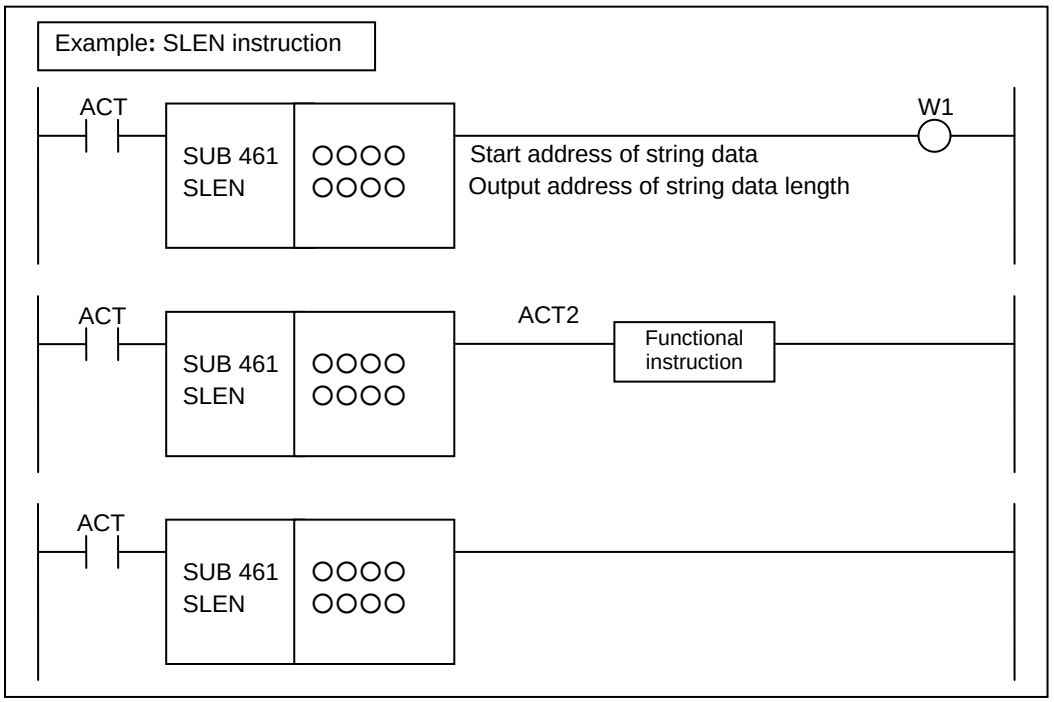


Fig. 5.3.1 (b) Format of SLEN instruction

Table 5.3.1 (a) Mnemonic of SLEN instruction

Mnemonic format					Memory status of control condition			
Step number	Instruction	Address No.	Bit No.	Remarks	ST3	ST2	ST1	ST0
1	RD	○○○○	.○	ACT				ACT
2	SUB	461		SUB No. (SLEN instruction)				↓
3	(PRM)	○○○○		Start address of string data				
4	(PRM)	○○○○		Output address of string data length				↓
5	WRT	○○○○	.○	Normal end output				W1

Control conditions

(a) Input signal (ACT)
ACT=0: Do not execute the instruction.
ACT=1: Execute the instruction.

Parameters

(a) Start address of string data
Specify the start address of the string data.

(b) Output address of string data length
Specify PMC memory address where the 2-byte data of the string data length will be stored.

NOTE
Refer to the section “5.1.1 String data" for the specification of the string data.

⚠ CAUTION
Be careful not to exceed the range of the PMC address, when setting the above addresses.

OUTPUT (W1)

W1=1: The operation has finished normally.
W1=0: The operation was not executed (ACT=0).

NOTE
W1 can be omitted. Or it can be also connected to another functional instruction instead of a coil.

5.3.2 SLEFT (Extracting the left part of Character string : SUB 462)

This instruction extracts the string data of the specified length from the head of the original string data, and outputs the extracted string data to the specified address.
00H (NULL) is appended at the end of the outputted string data.
When the length of the original string data is shorter than the specified number, all of the original string data is copied.

NOTE
If a minus value is set to the number of characters to be extracted, W1 = 0 and no data is output.

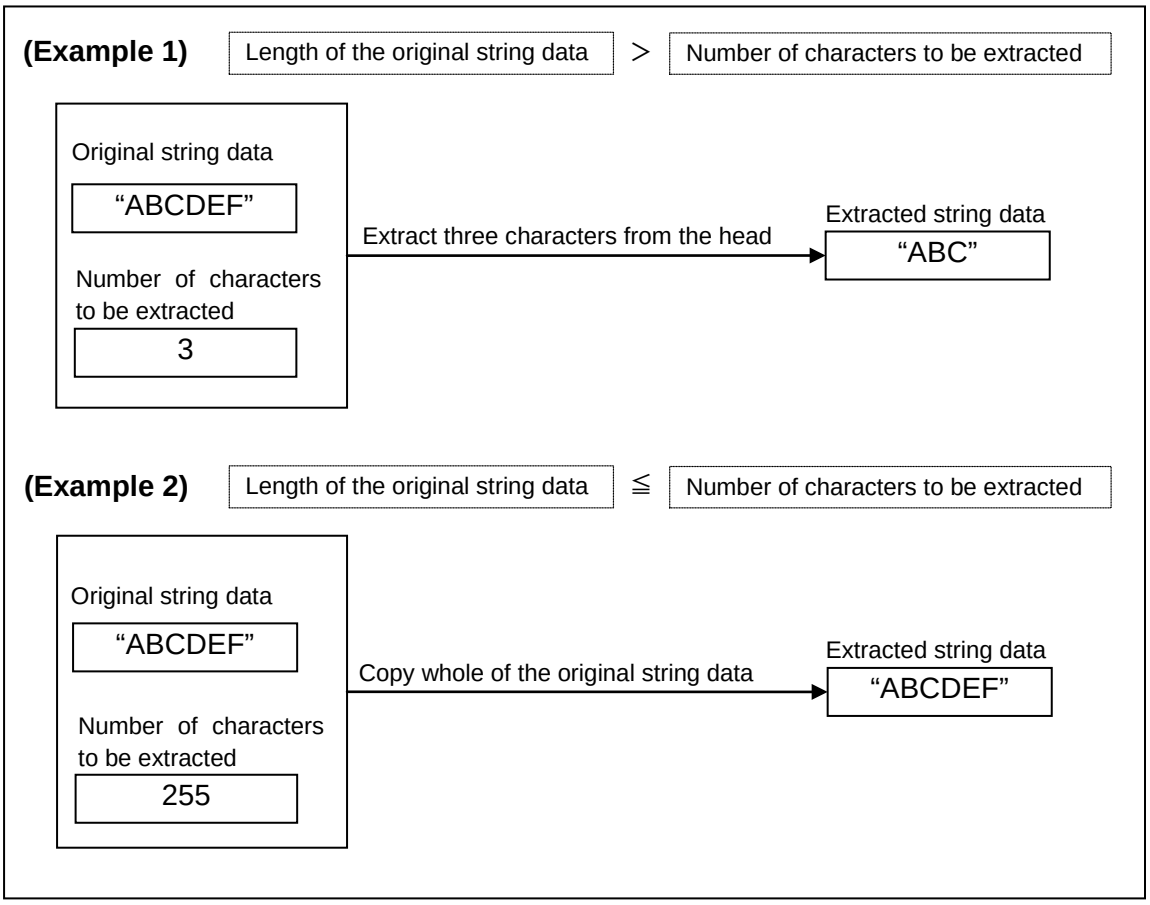


Fig. 5.3.2 (a) Operation of SLEFT Instruction

Format

The ladder format and the mnemonic format of this instruction are as follows.

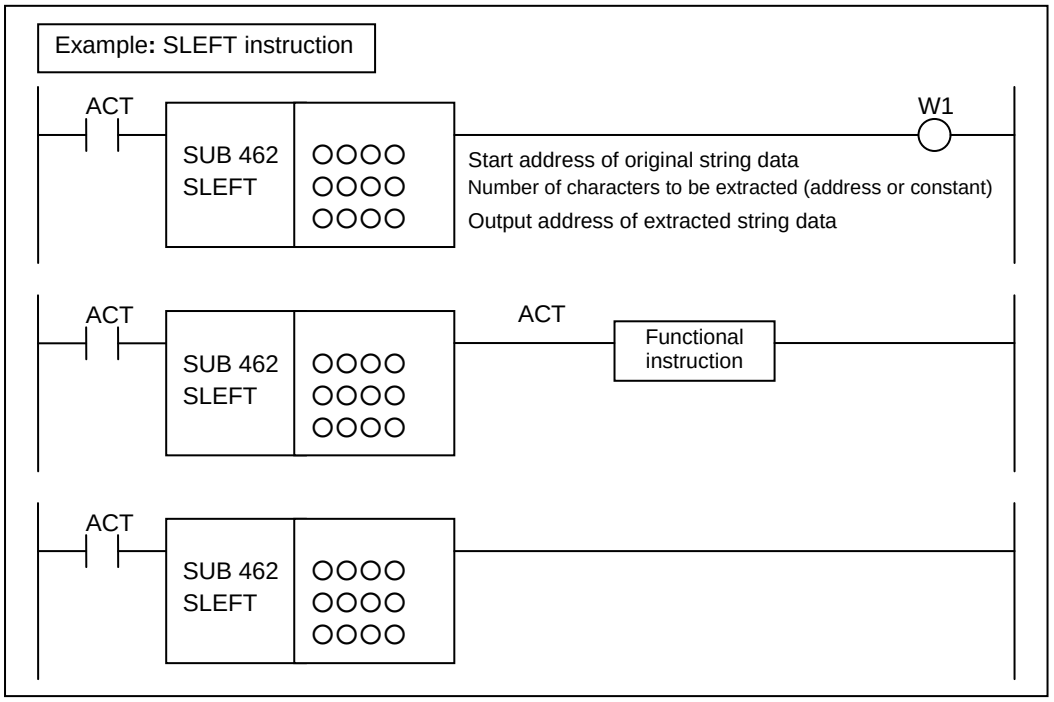


Fig. 5.3.2 (b) Format of SLEFT instruction

Table 5.3.2 (a) Mnemonic of SLEFT instruction

Mnemonic format					Memory status of control condition			
Step number	Instruction	Address No.	Bit No.	Remarks	ST3	ST2	ST1	ST0
1	RD	OOOO	.O	ACT				ACT
2	SUB	462		SUB No. (SLEFT instruction)				↓
3	(PRM)	OOOO		Start address of original string data				
4	(PRM)	OOOO		Number of characters to be extracted (address or constant)				
5	(PRM)	OOOO		Output address of extracted string data				↓
6	WRT	OOOO	.O	Normal end output				W1

Control conditions

(a) Input signal (ACT)
ACT=0: Do not execute the instruction.
ACT=1: Execute the instruction.

Parameters

(a) Start address of string data
Specify the start address of the string data.

(b) Number of characters to be extracted
Specify the number of characters to be extracted. In this parameter, a constant or a PMC memory address for storing 2-byte data can be specified.

(c) Output address of extracted string data
Specify PMC memory address where the extracted string data will be stored.

NOTE
Refer to the section “5.1.1 String data" for the specification of the string data.

⚠ CAUTION
Be careful not to exceed the range of the PMC address, when setting the above addresses.

OUTPUT (W1)

W1=1 : The operation has finished normally.
W1=0 : The operation was not executed (ACT=0).
A minus value was specified to the number of characters to be extracted.

NOTE
W1 can be omitted. Or it can be also connected to another functional instruction instead of a coil.

5.3.3 SRIGH (Extracting the right part of Character string : SUB 463)

This instruction extracts the string data of the specified length from the end of the original string data, and outputs the extracted string data to the specified address.
00H (NULL) is appended at the end of the outputted string data.
When the length of the original string data is shorter than the specified number, all of the original string data is copied.
If the length of the original string data is 256 or more characters, it extracts backward from 255th character.

NOTE
If a minus value is set to the number of characters to be extracted, W1 = 0 and no data is output.

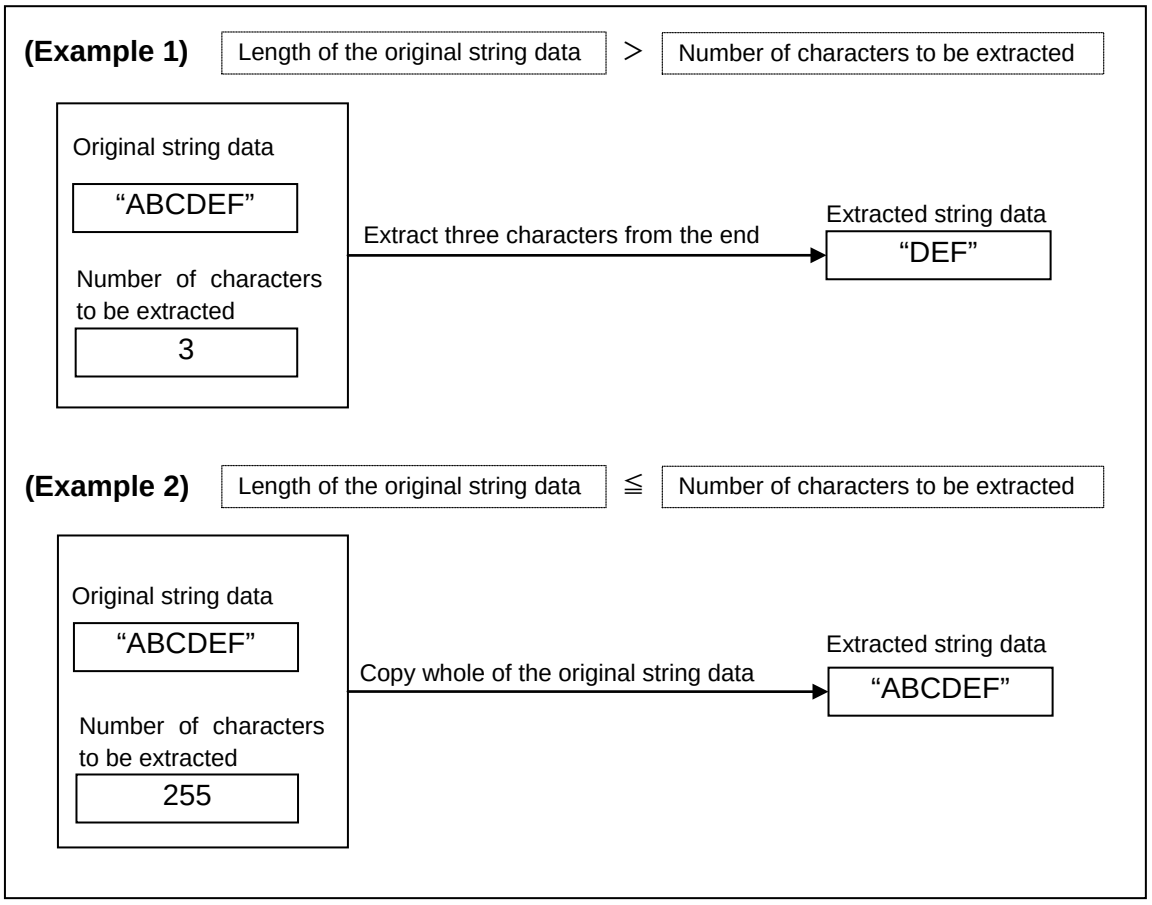


Fig. 5.3.3 (a) Operation of SRIGH Instruction

Format

The ladder format and the mnemonic format of this instruction are as follows.

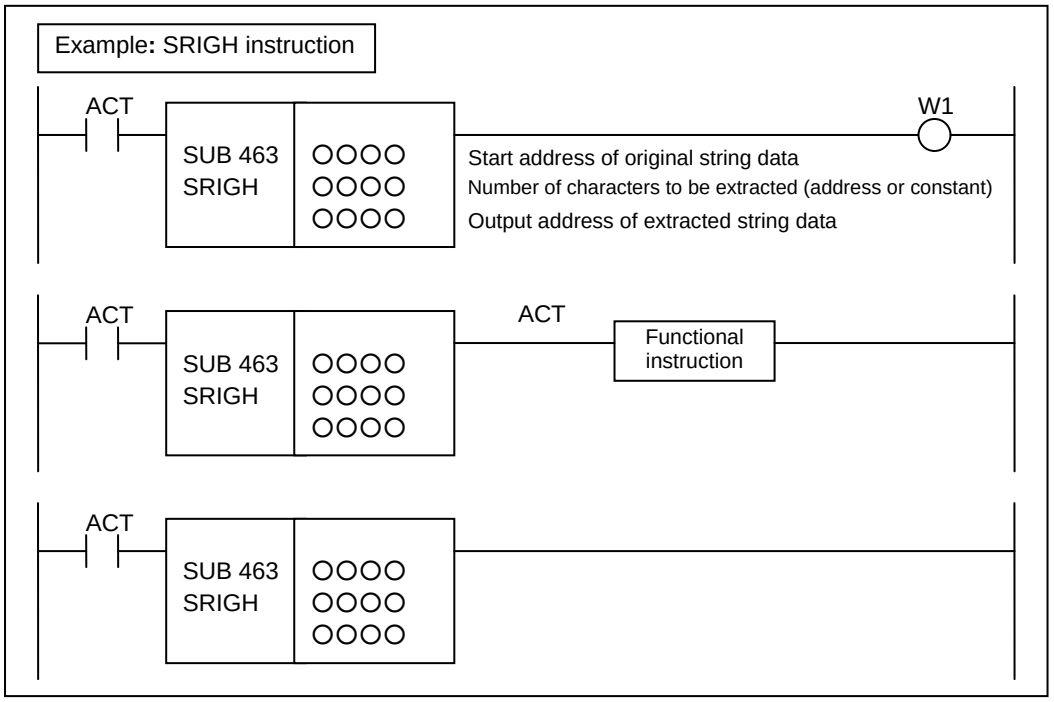


Fig. 5.3.3 (b) Format of SRIGH instruction

Table 5.3.3 (a) Mnemonic of SRIGH instruction

Mnemonic format					Memory status of control condition			
Step number	Instruction	Address No.	Bit No.	Remarks	ST3	ST2	ST1	ST0
1	RD	OOOO	.O	ACT				ACT
2	SUB	463		SUB No. (SRIGH instruction)				
3	(PRM)	OOOO		Start address of original string data				
4	(PRM)	OOOO		Number of characters to be extracted (address or constant)				
5	(PRM)	OOOO		Output address of extracted string data				
6	WRT	OOOO	.O	Normal end output				W1

Control conditions

(a) Input signal (ACT)
ACT=0: Do not execute the instruction.
ACT=1: Execute the instruction.

Parameters

(a) Start address of string data
Specify the start address of the string data.

(b) Number of characters to be extracted
Specify the number of characters to be extracted. In this parameter, a constant or a PMC memory address for storing 2-byte data can be specified.

(c) Output address of extracted string data
Specify PMC memory address where the extracted string data will be stored.

NOTE
Refer to the section “5.1.1 String data" for the specification of the string data.

⚠ CAUTION
Be careful not to exceed the range of the PMC address, when setting the above addresses.

OUTPUT (W1)

W1=1 : The operation has finished normally.
W1=0 : The operation was not executed (ACT=0).
A minus value was specified to the number of characters to be extracted.

NOTE
W1 can be omitted. Or it can be also connected to another functional instruction instead of a coil.

5.3.4 SMID (Extracting a part of Character string : SUB 464)

This instruction extracts the string data of the specified length from the specified position of the original string data, and outputs the extracted string data to the specified address.
00H (NULL) is appended at the end of the outputted string data.
If the number of characters to be extracted exceeds the string data length of the extraction source string data from the extraction start position, all source string data are output from the extraction start position.
If the extraction start position exceeds the length of the original string data, 00H (NULL) is output.

NOTE
If a minus value is set to the number of characters to be extracted, or the value of extraction start position less than or equal to 0 is specified, W1 = 0 and no data is output.

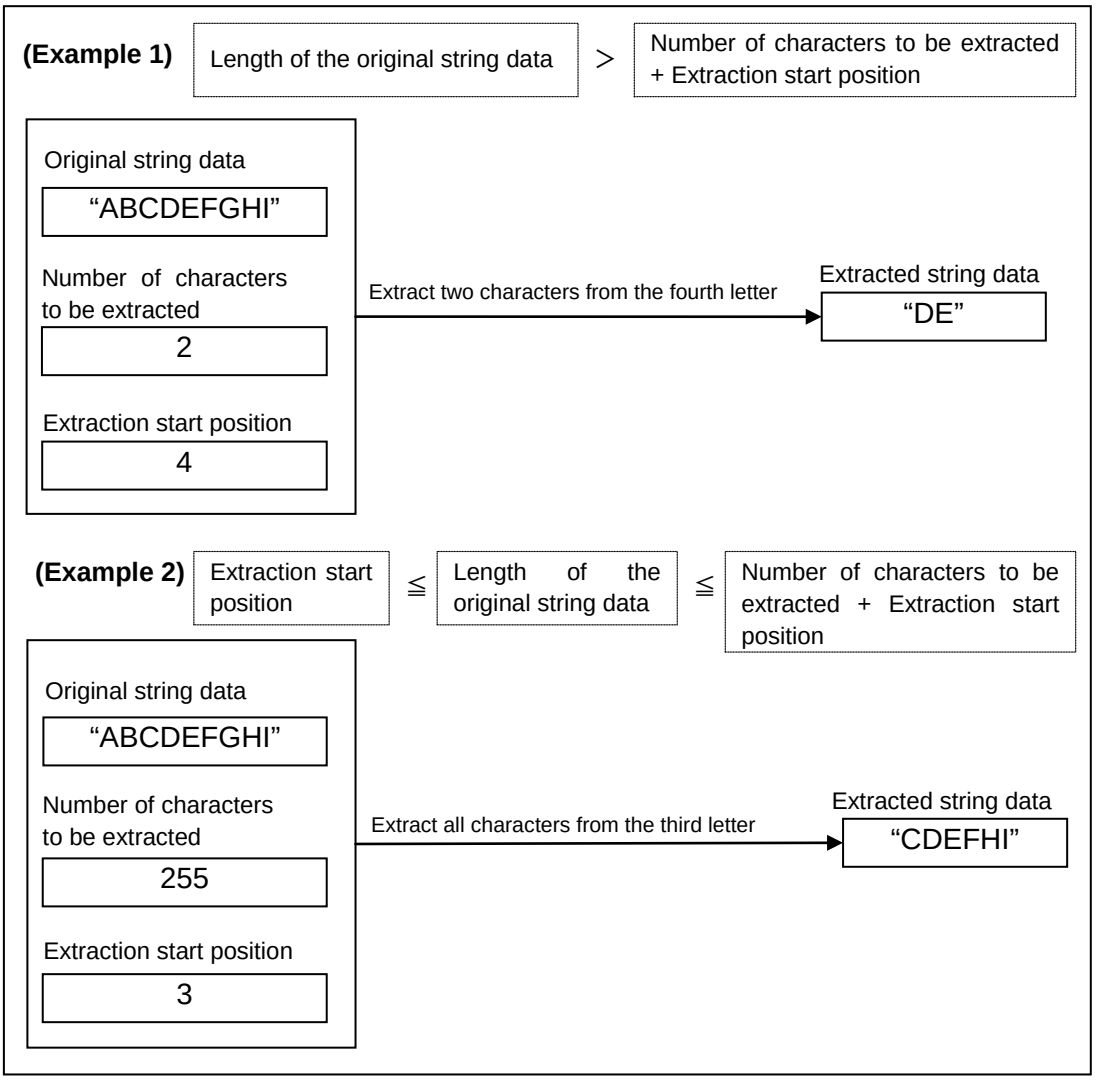


Fig. 5.3.4 (a) Operation of SMID Instruction-1

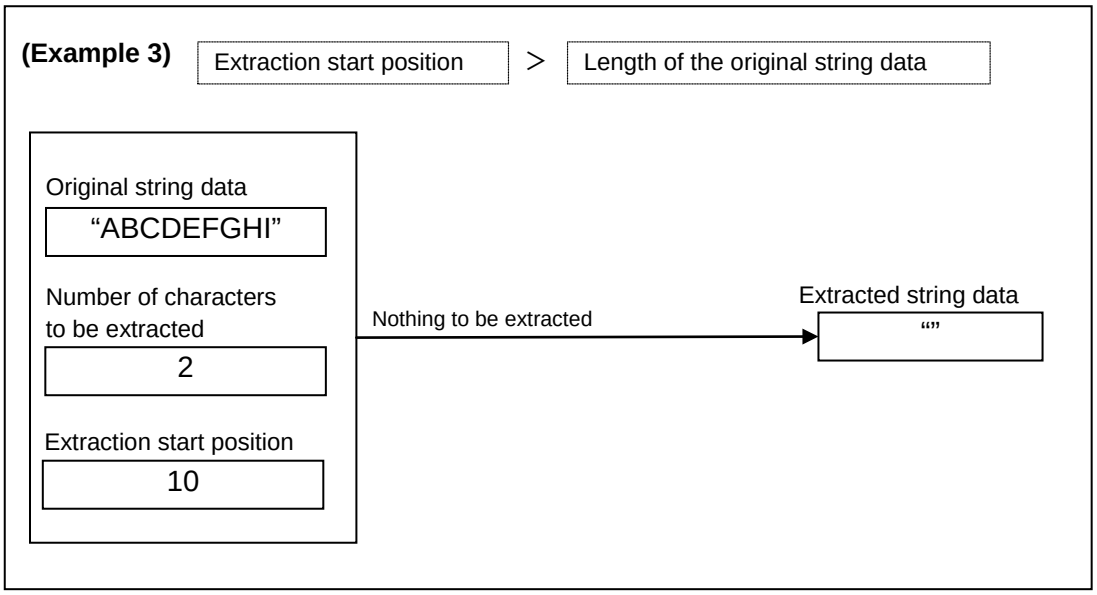


Fig. 5.3.4 (b) Operation of SMID Instruction-2

Format

The ladder format and the mnemonic format of this instruction are as follows.

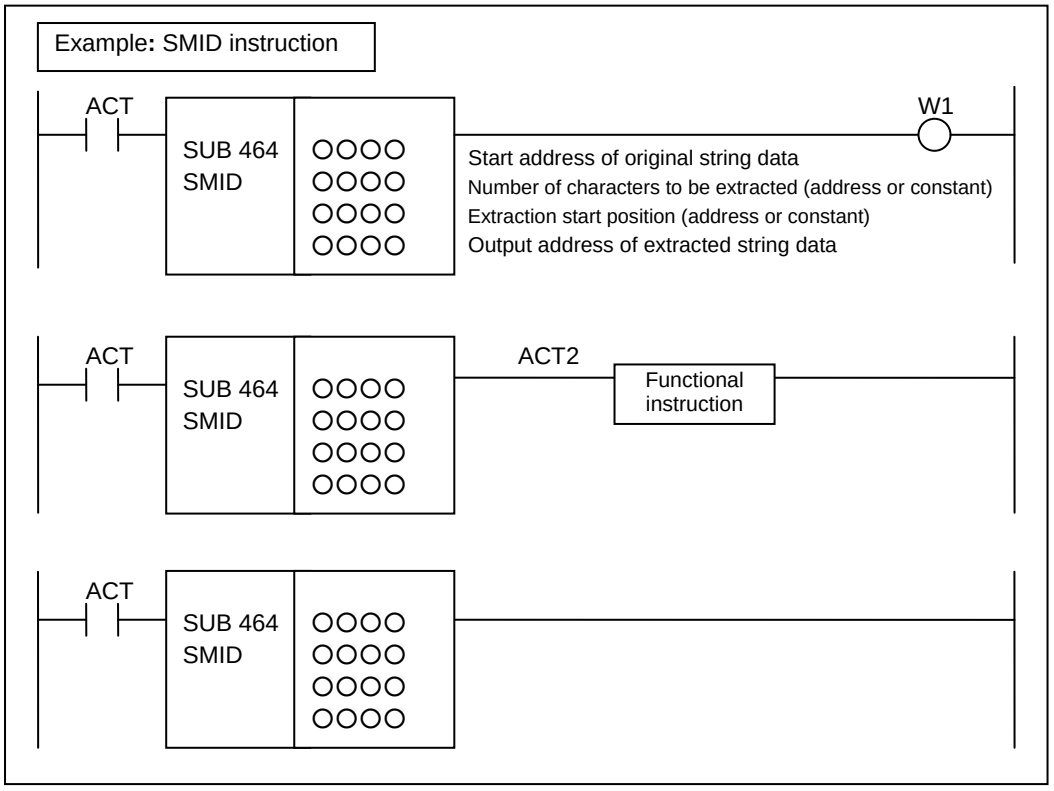


Fig. 5.3.4 (c) Format of SMID instruction

Table 5.3.4 (a) Mnemonic of SMID instruction

Mnemonic format					Memory status of control condition			
Step number	Instruction	Address No.	Bit No.	Remarks	ST3	ST2	ST1	ST0
1	RD	○○○○	.○	ACT				ACT
2	SUB	464		SUB No. (SLEFT instruction)				
3	(PRM)	○○○○		Start address of original string data				
4	(PRM)	○○○○		Number of characters to be extracted (address or constant)				
5	(PRM)	○○○○		Extraction start position (address or constant)				
6	(PRM)	○○○○		Output address of extracted string data				
7	WRT	○○○○	.○	Normal end output				W1

Control conditions

(a) Input signal (ACT)
ACT=0: Do not execute the instruction.
ACT=1: Execute the instruction.

Parameters

(a) Start address of string data
Specify the start address of the string data.

(b) Number of characters to be extracted
Specify the number of characters to be extracted. In this parameter, a constant or a PMC memory address for storing 2-byte data can be specified.

(c) Extraction start position
Specify the value of extraction start position. In this parameter, a constant or a PMC memory address for storing 2-byte data can be specified.

(d) Output address of extracted string data
Specify PMC memory address where the extracted string data will be stored.

NOTE
Refer to the section “5.1.1 String data" for the specification of the string data.

⚠ CAUTION
Be careful not to exceed the range of the PMC address, when setting the above addresses.

OUTPUT (W1)

W1=1 : The operation has finished normally.
W1=0 : The operation was not executed (ACT=0).
A minus value was specified for the number of characters to be extracted or the value of extraction start position less than or equal to 0 is specified.

NOTE
W1 can be omitted. Or it can be also connected to another functional instruction instead of a coil.

5.3.5 SCON (Concatenating two Character strings : SUB 465)

This instruction concatenates two specified string data and outputs it to the specified address.
00H (NULL) is appended at the end of the outputted string data.
If the length of concatenated string data exceeds 255 characters, it will be truncated to 255 characters and outputted.

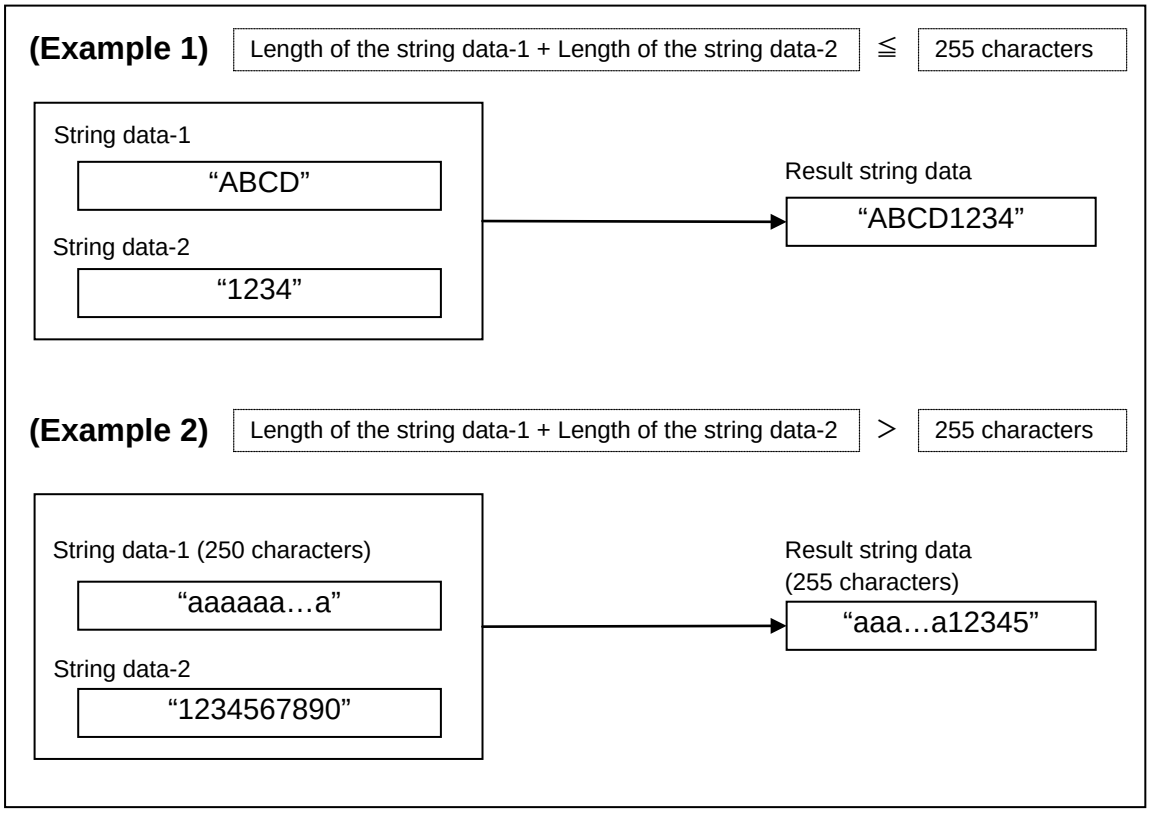


Fig. 5.3.5 (a) Operation of SCON Instruction

Format

The ladder format and the mnemonic format of this instruction are as follows.

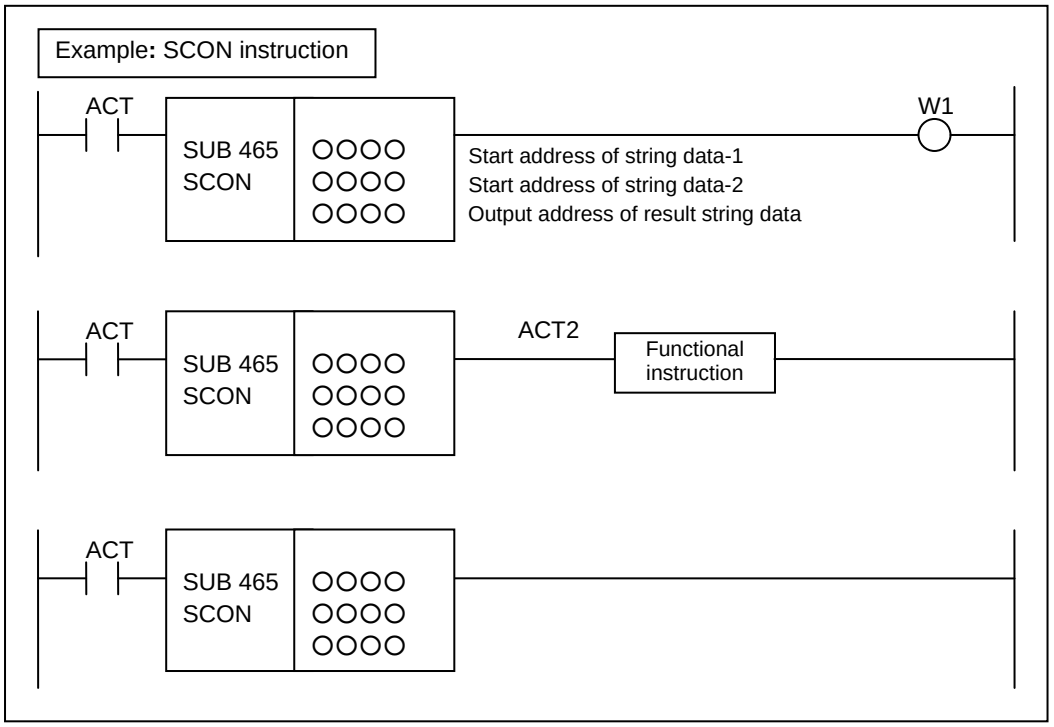


Fig. 5.3.5 (b) Format of SCON instruction

Table 5.3.5 (a) Mnemonic of SCON instruction

Mnemonic format					Memory status of control condition			
Step number	**Instruction**	**Address No.**	**Bit No.**	**Remarks**	**ST3**	**ST2**	**ST1**	**ST0**
1	RD	OOOO	.O	ACT				ACT
2	SUB	465		SUB No. (SCON instruction)				↓
3	(PRM)	OOOO		Start address of string data-1				
4	(PRM)	OOOO		Start address of string data-2				
5	(PRM)	OOOO		Output address of result string data				
6	WRT	OOOO	.O	Normal end output				W1

Control conditions

(a) Input signal (ACT)
ACT=0: Do not execute the instruction.
ACT=1: Execute the instruction.

Parameters

(a) Start address of string data-1
Specify the start address of the first string data to be concatenated.

(b) Start address of string data-2
Specify the start address of the second string data to be concatenated.

(c) Output address of modified string data
Specify PMC memory address where the concatenated string data will be stored.

NOTE
Refer to the section “5.1.1 String data" for the specification of the string data.

⚠ CAUTION
Be careful not to exceed the range of the PMC address, when setting the above addresses.

OUTPUT (W1)

W1=1 : The operation has finished normally.
W1=0 : The operation was not executed (ACT=0).

NOTE
W1 can be omitted. Or it can be also connected to another functional instruction instead of a coil.

5.3.6 SINS (Inserting a sub-string in Character string : SUB 466)

This instruction inserts the string data to the original string data at the specified position and outputs it to the specified address. The specified position is counted from the head of the original string data.
00H (NULL) is appended at the end of the outputted string data.
If the insertion start position is greater than the length of the original string data, the original string data and the string data to be inserted will be concatenated and outputted.
If the length of result string data exceeds 255 characters, it will be truncated to 255 characters and outputted.

NOTE
If the value of insertion start position less than or equal to 0 is specified, W1 = 0 and no data is output.

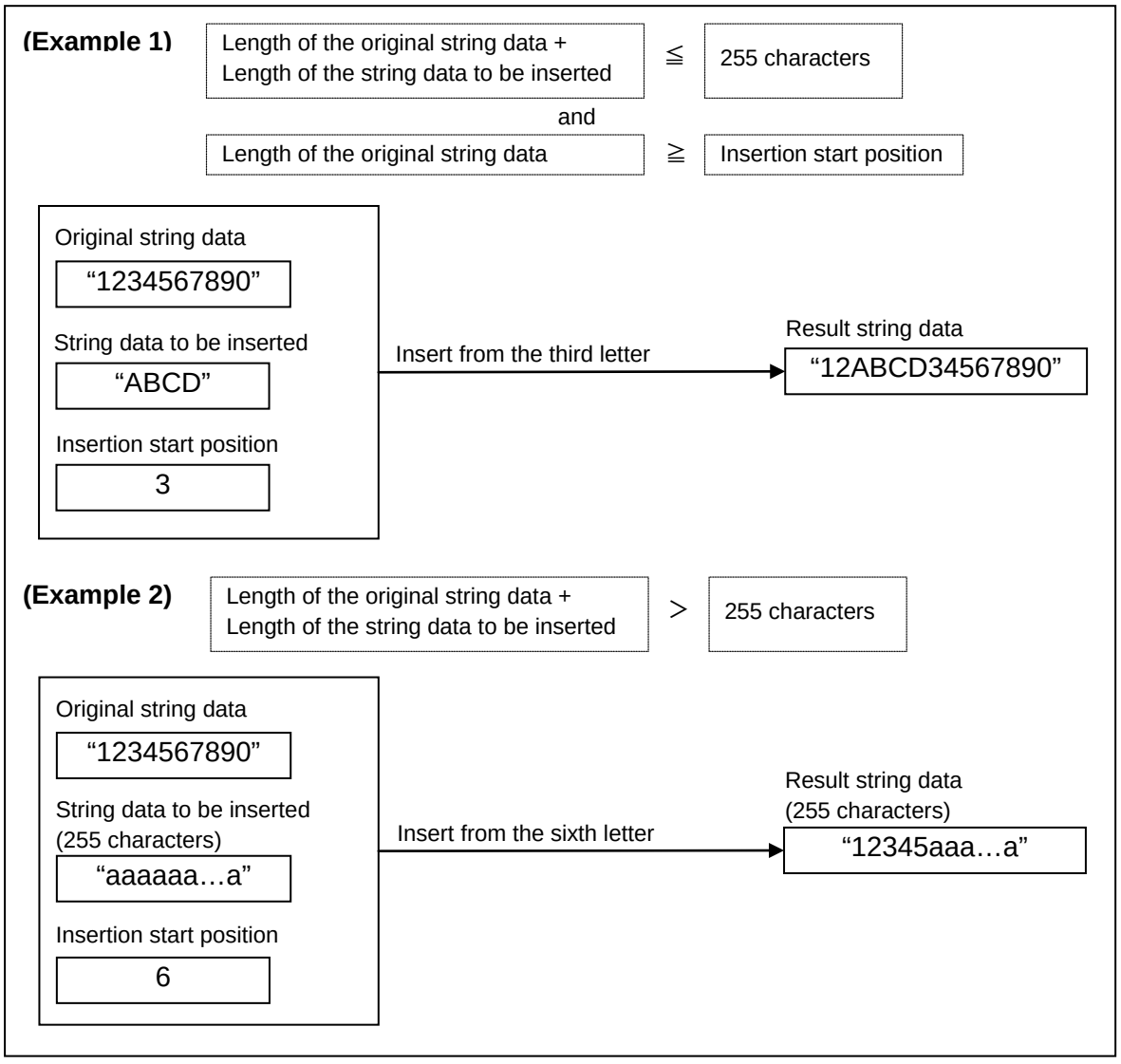


Fig. 5.3.6 (a) Operation of SINS Instruction-1

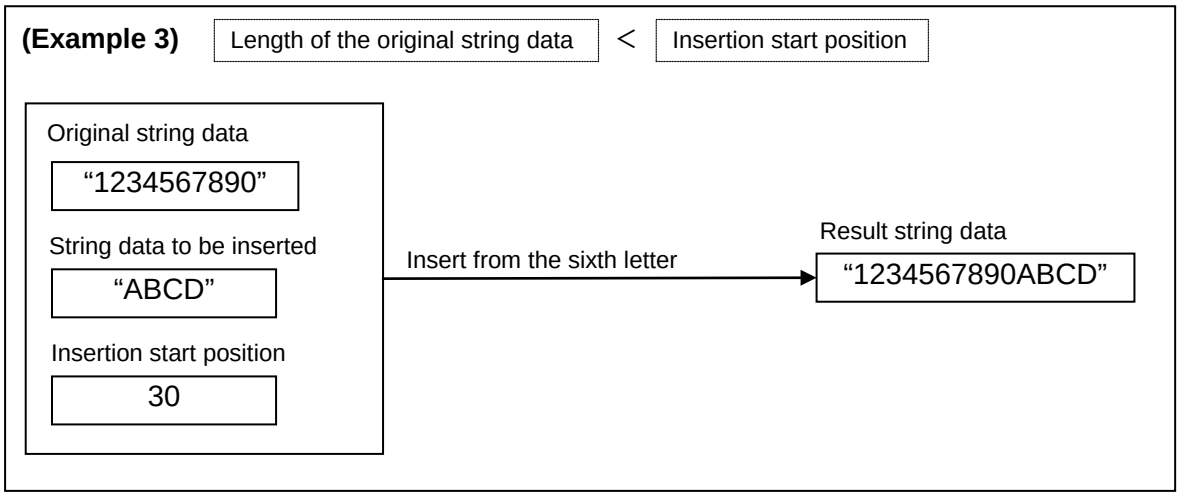


Fig. 5.3.6 (b) Operation of SINS Instruction-2

Format

The ladder format and the mnemonic format of this instruction are as follows.

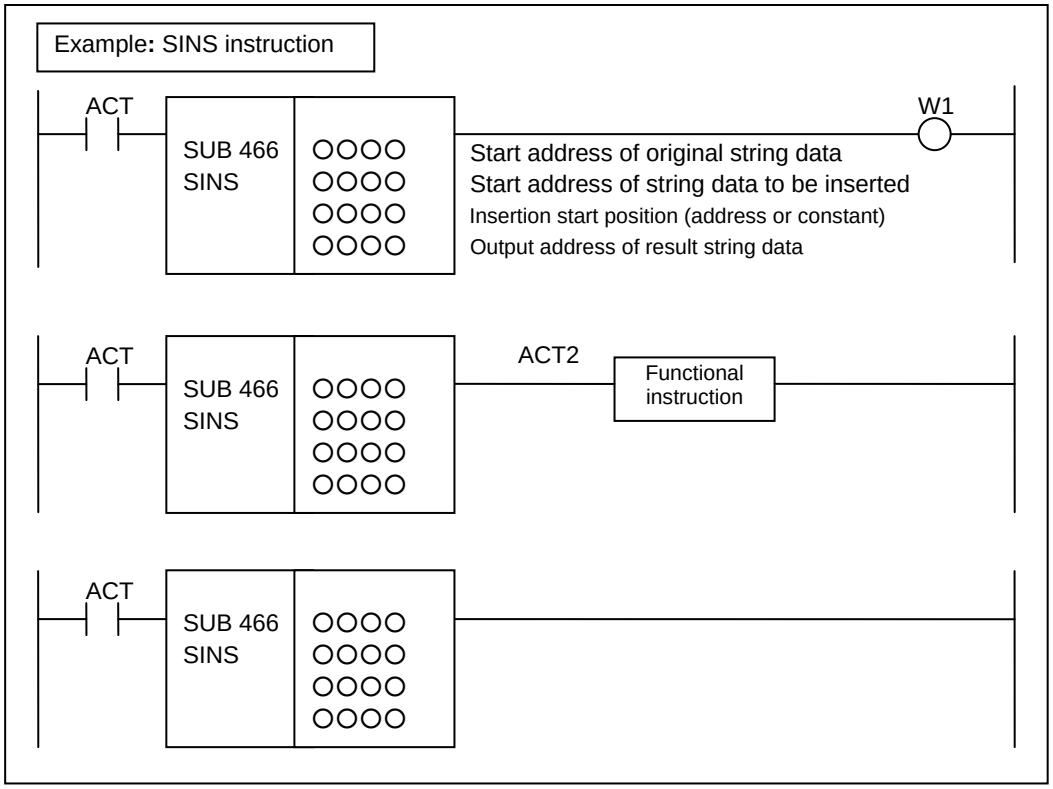


Fig. 5.3.6 (c) Format of SINS instruction

Table 5.3.6 (a) Mnemonic of SINS instruction

Mnemonic format

Step number	Instruction	Address No.	Bit No.	Remarks
1	RD	OOOO	.O	ACT
2	SUB	466		SUB No. (SINS instruction)
3	(PRM)	OOOO		Start address of original string data
4	(PRM)	OOOO		Start address of string data to be inserted
5	(PRM)	OOOO		Insertion start position (address or constant)
6	(PRM)	OOOO		Output address of result string data
7	WRT	OOOO	.O	Normal end output

Memory status of control condition

ST3	ST2	ST1	ST0
			ACT
			↓
			↓
			↓
			↓
			↓
			↓
			W1

Control conditions

(a) Input signal (ACT)
ACT=0: Do not execute the instruction.
ACT=1: Execute the instruction.

Parameters

(a) Start address of original string data
Specify the start address of the original string data.

(b) Start address of string data to be inserted
Specify the start address of the string data to be inserted.

(c) Insertion start position
Specify the value of insertion start position. In this parameter, a constant or a PMC memory address for storing 2-byte data can be specified.

(d) Output address of modified string data
Specify PMC memory address where the result string data will be stored.

NOTE
Refer to the section “5.1.1 String data" for the specification of the string data.

⚠ CAUTION
Be careful not to exceed the range of the PMC address, when setting the above addresses.

OUTPUT (W1)

W1=1 : The operation has finished normally.
W1=0 : The operation was not executed (ACT=0).
The value less than or equal to 0 is specified to the insertion start position.

NOTE
W1 can be omitted. Or it can be also connected to another functional instruction instead of a coil.

5.3.7 SDEL (Deleting a part of Character string : SUB 467)

This instruction deletes the string data of the specified length at the specified position, and outputs the result to the specified address. The specified position is counted from the head of the string data.
00H (NULL) is appended at the end of the outputted string data.
If the deletion start position exceeds the length of original string data, all original string data is output.

NOTE
If a minus value is set to the number of characters to be deleted, or the value of deletion start position less than or equal to 0 is specified, W1 = 0 and no data is output.

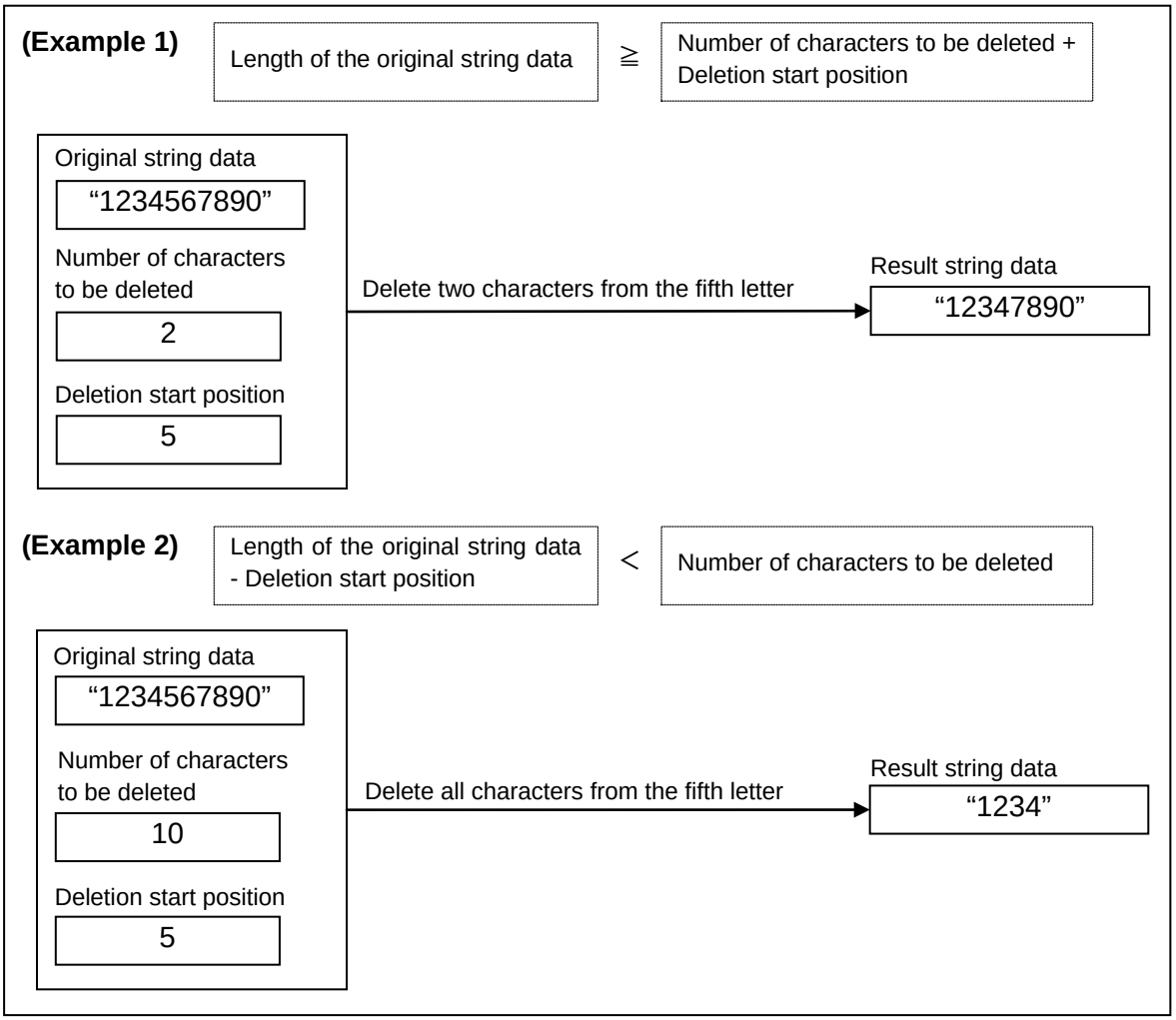


Fig. 5.3.7 (a) Operation of SDEL Instruction-1

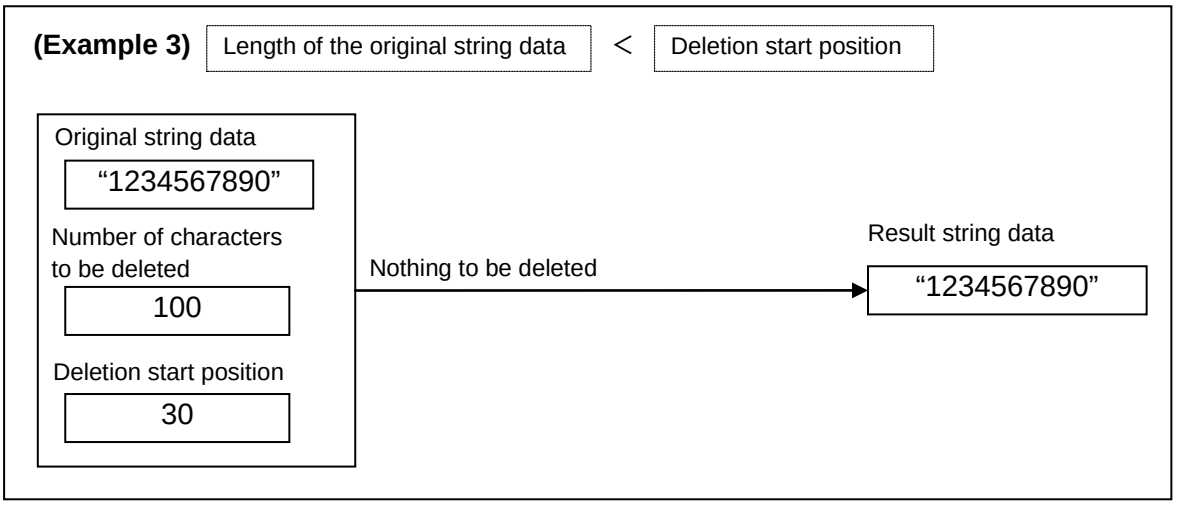


Fig. 5.3.7 (b) Operation of SDEL Instruction-2

Format

The ladder format and the mnemonic format of this instruction are as follows.

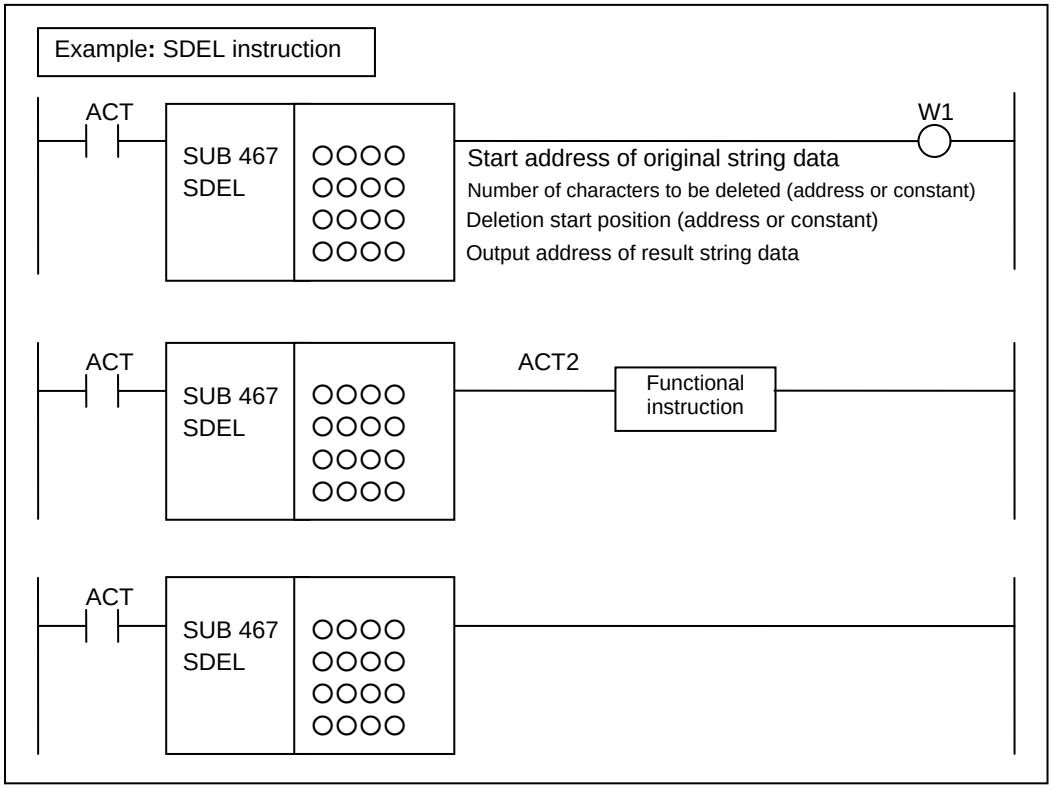


Fig. 5.3.7 (c) Format of SDEL instruction

Table 5.3.7 (a) Mnemonic of SDEL instruction

Mnemonic format					Memory status of control condition			
Step number	Instruction	Address No.	Bit No.	Remarks	ST3	ST2	ST1	ST0
1	RD	○○○○	.○	ACT				ACT
2	SUB	467		SUB No. (SDEL instruction)				
3	(PRM)	○○○○		Start address of original string data				
4	(PRM)	○○○○		Number of characters to be deleted (address or constant)				
5	(PRM)	○○○○		Deletion start position (address or constant)				
6	(PRM)	○○○○		Output address of result string data				
7	WRT	○○○○	.○	Normal end output				W1

Control conditions

(a) Input signal (ACT)
ACT=0: Do not execute the instruction.
ACT=1: Execute the instruction.

Parameters

(a) Start address of original string data
Specify the start address of the original string data.

(b) Number of characters to be deleted
Specify the number of characters to be deleted. In this parameter, a constant or a PMC memory address for storing 2-byte data can be specified.

(c) Deletion start position
Specify the value of deletion start position. In this parameter, a constant or a PMC memory address for storing 2-byte data can be specified.

(d) Output address of modified string data
Specify PMC memory address where the result string data will be stored.

NOTE
Refer to the section “5.1.1 String data" for the specification of the string data.

⚠ CAUTION
Be careful not to exceed the range of the PMC address, when setting the above addresses.

OUTPUT (W1)

W1=1 : The operation has finished normally.
W1=0 : The operation was not executed (ACT=0).
A minus value was specified for the number of characters to be deleted or the value of deletion start position less than or equal to 0 is specified.

NOTE

W1 can be omitted. Or it can be also connected to another functional instruction instead of a coil.

5.3.8 SREP (Replacing a part of Character string : SUB 468)

This instruction replaces the string data of the specified length at the specified position and outputs it to the specified address. The specified position is counted from the head of the original string data.
00H (NULL) is appended at the end of the outputted string data.
If the replacement start position is greater than the length of the original string data, the original string data and the string data to be replaced with will be concatenated and outputted.
If the number of string data to be replaced exceeds the original string data length, it deletes all after the replacement start position of the original string data, then, appends the string data to be replacement and output it.
If the length of result string data exceeds 255 characters, it will be truncated to 255 characters and outputted.

NOTE
If a minus value is set to the number of characters to be replaced, or the value of replacement start position less than or equal to 0 is specified, W1 = 0 and no data is output.

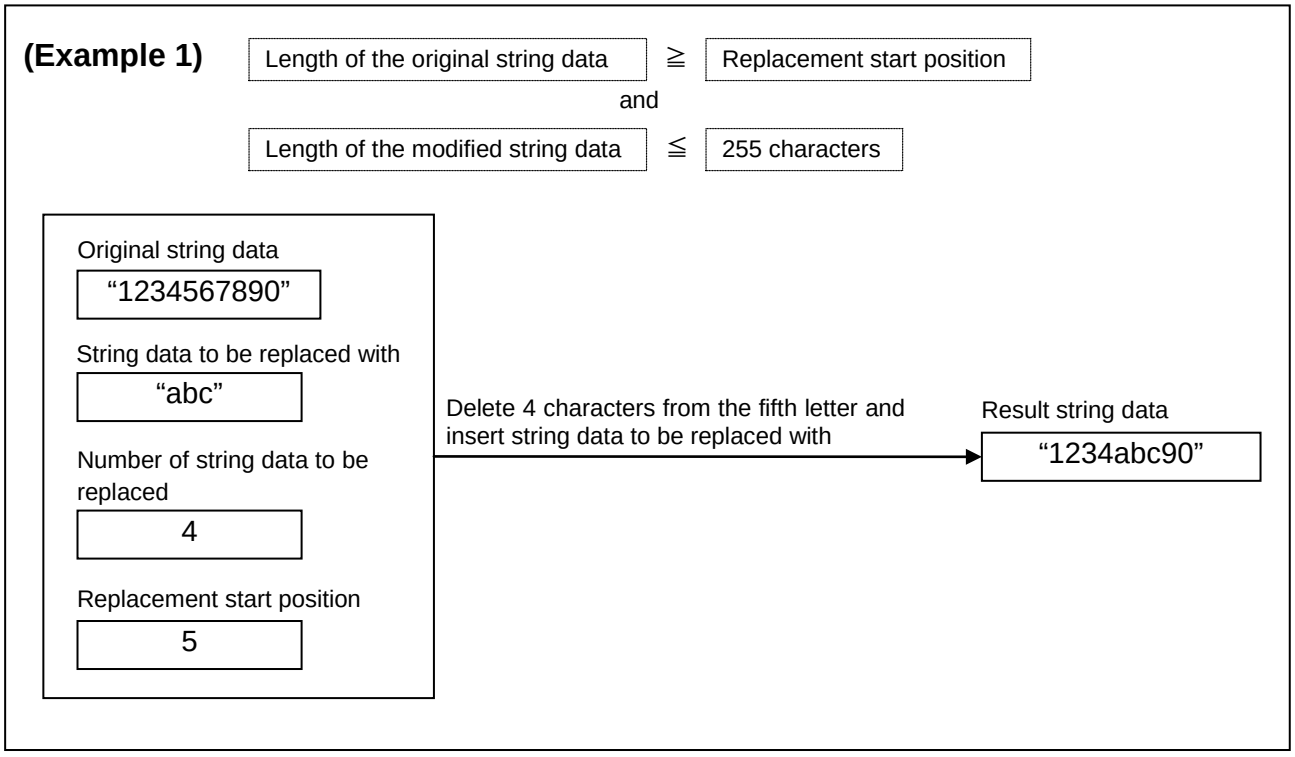


Fig. 5.3.8 (a) Operation of SREP Instruction-1

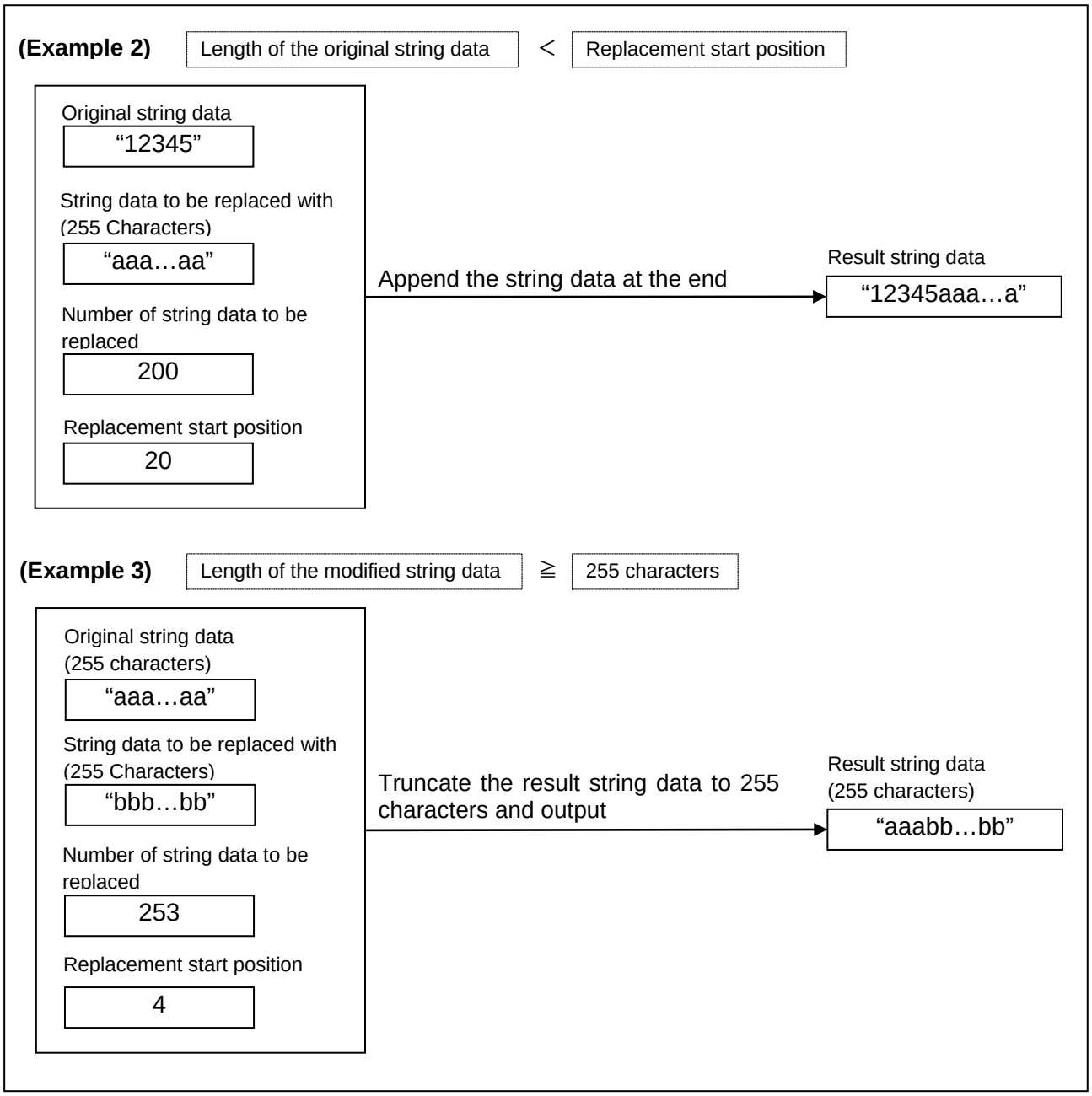


Fig. 5.3.8 (b) Operation of SREP Instruction-2

Format

The ladder format and the mnemonic format of this instruction are as follows.

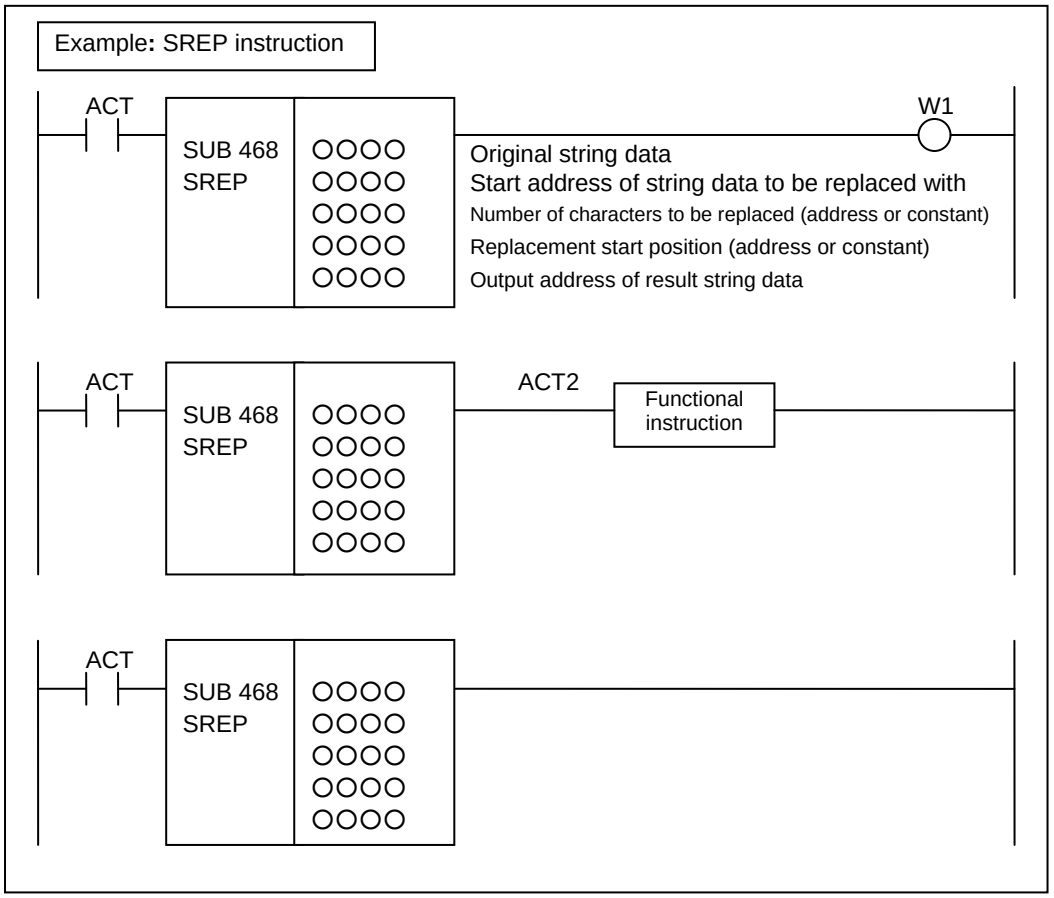


Fig. 5.3.8 (c) Format of SREP instruction

Table 5.3.8 (a) Mnemonic of SREP instruction

	Mnemonic format				Memory status of control condition			
Step number	**Instruction**	**Address No.**	**Bit No.**	**Remarks**	**ST3**	**ST2**	**ST1**	**ST0**
1	RD	OOOO	.O	ACT				ACT
2	SUB	468		SUB No. (SREP instruction)				
3	(PRM)	OOOO		Original string data				
4	(PRM)	OOOO		Start address of string data to be replaced with				
5	(PRM)	OOOO		Number of characters to be replaced (address or constant)				
6	(PRM)	OOOO		Replacement start position (address or constant)				
7	(PRM)	OOOO		Output address of result string data				
8	WRT	OOOO	.O	Normal end output				W1

Control conditions

(a) Input signal (ACT)
ACT=0: Do not execute the instruction.
ACT=1: Execute the instruction.

Parameters

(a) Start address of original string data
Specify the start address of the original string data.

(b) Start address of string data to be replaced with
Specify the start address of the string data to be replaced with.

(c) Number of characters to be replaced
Specify the number of characters to be replaced. In this parameter, a constant or a PMC memory address for storing 2-byte data can be specified.

(d) Replacement start position
Specify the value of replacement start position. In this parameter, a constant or a PMC memory address for storing 2-byte data can be specified.

(e) Output address of modified string data
Specify PMC memory address where the result string data will be stored.

NOTE
Refer to the section “5.1.1 String data" for the specification of the string data.

⚠ CAUTION
Be careful not to exceed the range of the PMC address, when setting the above addresses.

OUTPUT (W1)

W1=1: The operation has finished normally.
W1=0: The operation was not executed (ACT=0).
A minus value was specified for the number of characters to be replaced or the value of replacement start position less than or equal to 0 is specified.

NOTE
W1 can be omitted. Or it can be also connected to another functional instruction instead of a coil.

5.3.9 SFIND(Finding a sub-string in Character string : SUB 469)

This instruction searches the sub-string in the string data, and outputs the position where the target string data is found to the specified address.
If the target string is not found, 0 is output to the result.

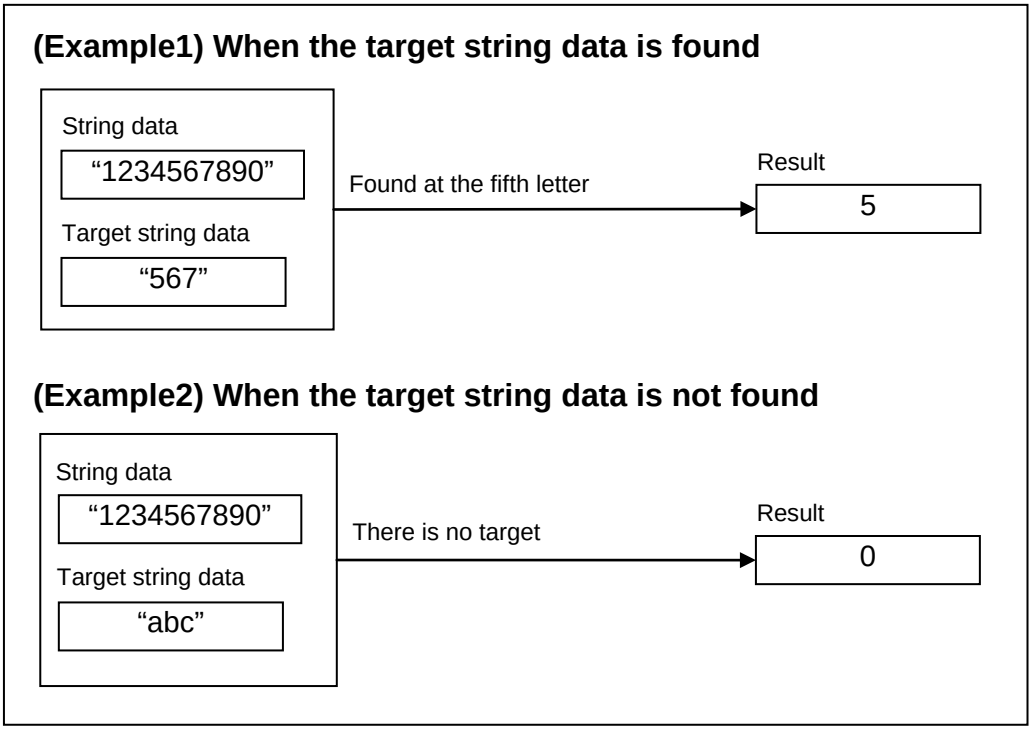


Fig. 5.3.9 (a) Operation of SFIND Instruction

Format

The ladder format and the mnemonic format of this instruction are as follows.

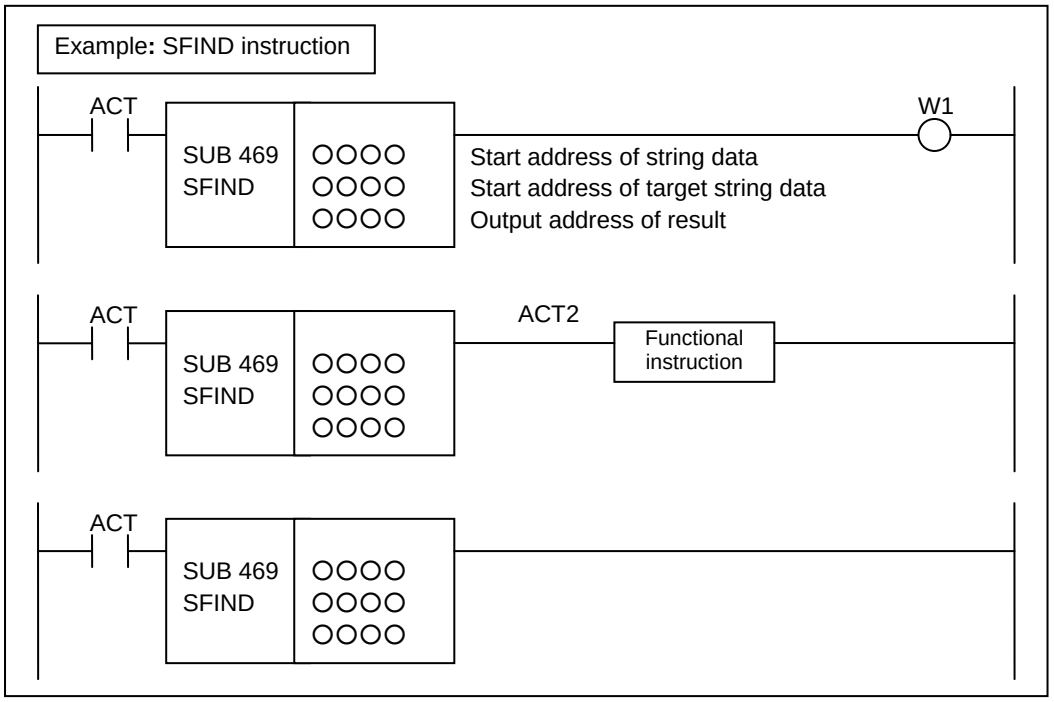


Fig. 5.3.9 (b) Format of SFIND instruction

Table 5.3.9 (a) Mnemonic of SFIND instruction

Mnemonic format					Memory status of control condition			
Step number	Instruction	Address No.	Bit No.	Remarks	ST3	ST2	ST1	ST0
1	RD	OOOO	.O	ACT				ACT
2	SUB	469		SUB No. (SFIND instruction)				↓
3	(PRM)	OOOO		Start address of string data				
4	(PRM)	OOOO		Start address of target string data				
5	(PRM)	OOOO		Output address of result				↓
6	WRT	OOOO	.O	Normal end output				W1

Control conditions

(a) Input signal (ACT)
ACT=0: Do not execute the instruction.
ACT=1: Execute the instruction.

Parameters

(a) Start address of string data
Specify the start address of the string data.

(b) Start address of target string data
Specify the start address of target string data.

(c) Output address of result
Specify PMC memory address where the 2-byte data of result is stored.

NOTE
Refer to the section “5.1.1 String data" for the specification of the string data.

⚠ CAUTION
Be careful not to exceed the range of the PMC address, when setting the above addresses.

OUTPUT (W1)

W1=1 : The operation has finished normally.
W1=0 : The operation was not executed (ACT=0).

NOTE
W1 can be omitted. Or it can be also connected to another functional instruction instead of a coil.

5.3.10 BTOS (Conversion (1-byte signed binary→Character string) : SUB 470) WTOS (Conversion (2-byte signed binary→Character string) : SUB 471) DTOS (Conversion (4-byte signed binary→Character string) : SUB 472)

These instructions convert a 1, 2 or 4 byte signed binary data to string data.
00H (NULL) is appended at the end of the outputted string data.

The following instructions are available according to the data type of the parameter.

Table 5.3.10 (a) Kinds of data conversion instruction

	Instruction name	SUB No.	Data type
1	BTOS	470	1-byte signed binary
2	WTOS	471	2-byte signed binary
3	DTOS	472	4-byte signed binary

Fig. 5.3.10 (a) Operation of BTOS, WTOS, DTOS instruction

Format

The ladder format and the mnemonic format of these instructions are as follows.

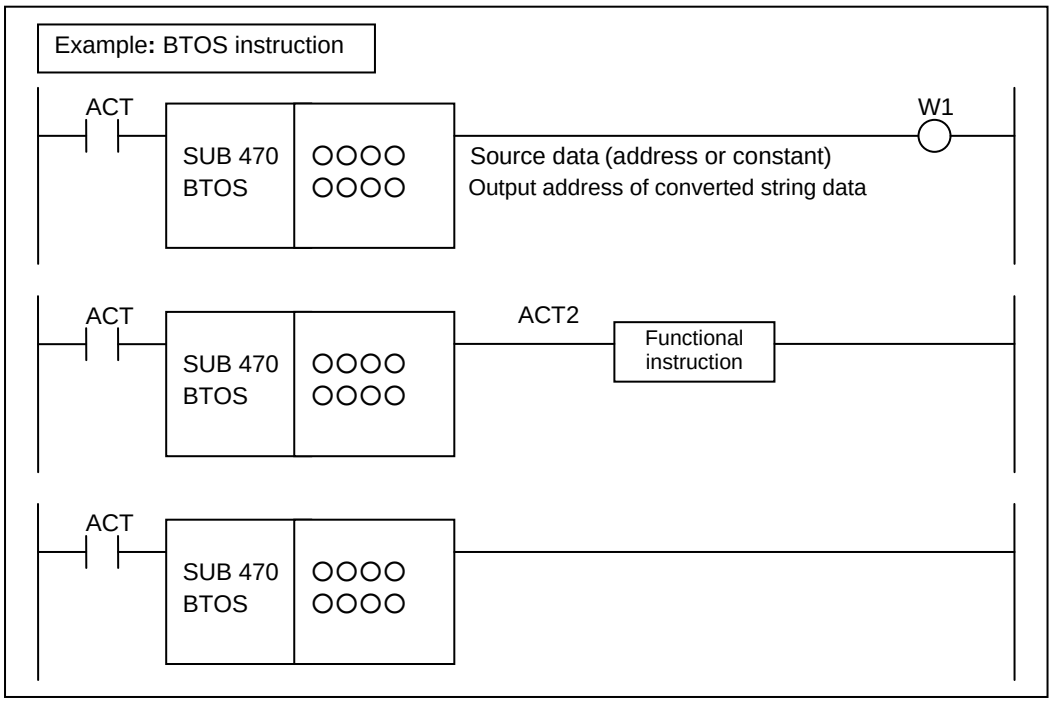


Fig. 5.3.10 (b) Format of BTOS, WTOS, DTOS instructions

Table 5.3.10 (b) Mnemonic of BTOS, WTOS, DTOS instructions

Mnemonic format

Step number	Instruction	Address No.	Bit No.	Remarks
1	RD	OOOO	.O	ACT
2	SUB	470		SUB No. (BTOS instruction)
3	(PRM)	OOOO		Source data (address or constant)
4	(PRM)	OOOO		Output address of converted string data
5	WRT	OOOO	.O	Normal end output

Memory status of control condition

ST3	ST2	ST1	ST0
			ACT
			↓
			↓
			↓
			W1

Control conditions

(a) Input signal (ACT)
ACT=0: Do not execute the instruction.
ACT=1: Execute the instruction.

Parameters

(a) Source data
Specify the source data. You can specify the constant or PMC memory address.

(b) Output address of conversion result
Specify PMC memory address where the conversion result will be stored.

NOTE
Refer to the section “5.1.1 String data" for the specification of the string data.

⚠ CAUTION
Be careful not to exceed the range of the PMC address, when setting the above addresses.

OUTPUT (W1)

W1=1 : The operation has finished normally.
W1=0 : The operation was not executed (ACT=0).

NOTE
W1 can be omitted. Or it can be also connected to another functional instruction instead of a coil.

5.3.11 RTOS (Conversion (Single Precision Real→Character string) : SUB 473)
LTOS (Conversion (Double Precision Real→Character string) : SUB 474)

These instructions convert a real number to string data.
Depending on the size of the number, they are output in either decimal notation or exponential notation format.
00H (NULL) is appended at the end of the outputted string data.

The following instructions are available according to the data type of the parameter.

Table 5.3.11 (a) Kinds of data conversion instruction

	Instruction name	SUB No.	Data type
1	RTOS	473	Single precision real (4 bytes)
2	LTOS	474	Double precision real (8 bytes)

NOTE
Because the processing time of these instructions can be long depending on the data to convert, please check the scan time when you use them.

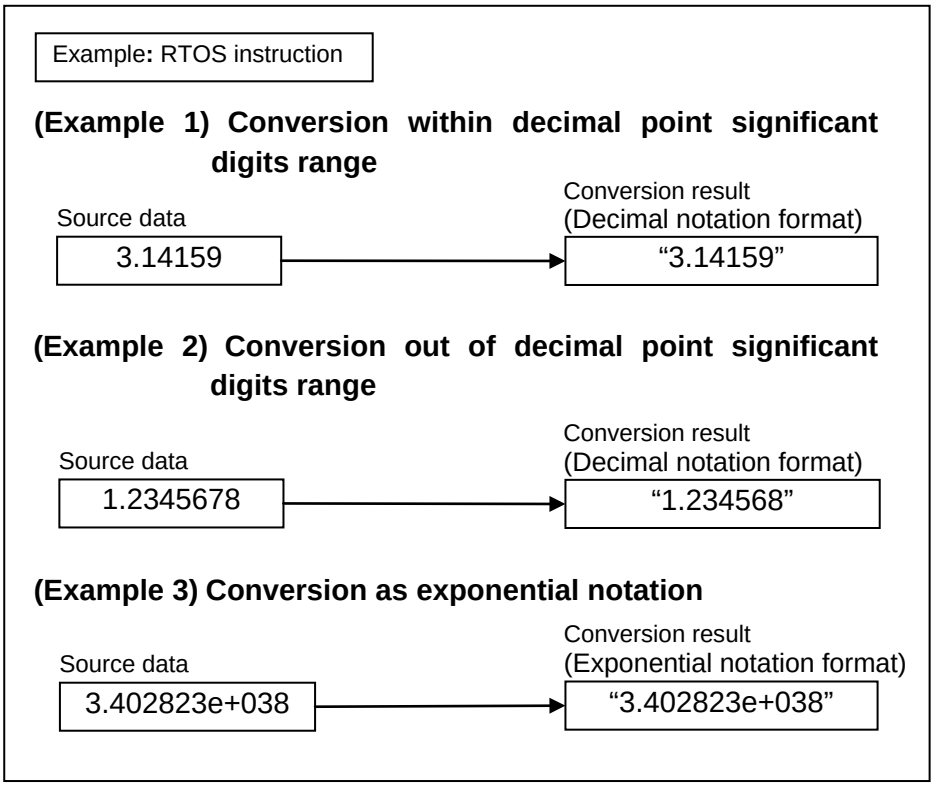


Fig. 5.3.11 (a) Operation of RTOS, LTOS instruction

Format

The ladder format and the mnemonic format of these instructions are as follows.

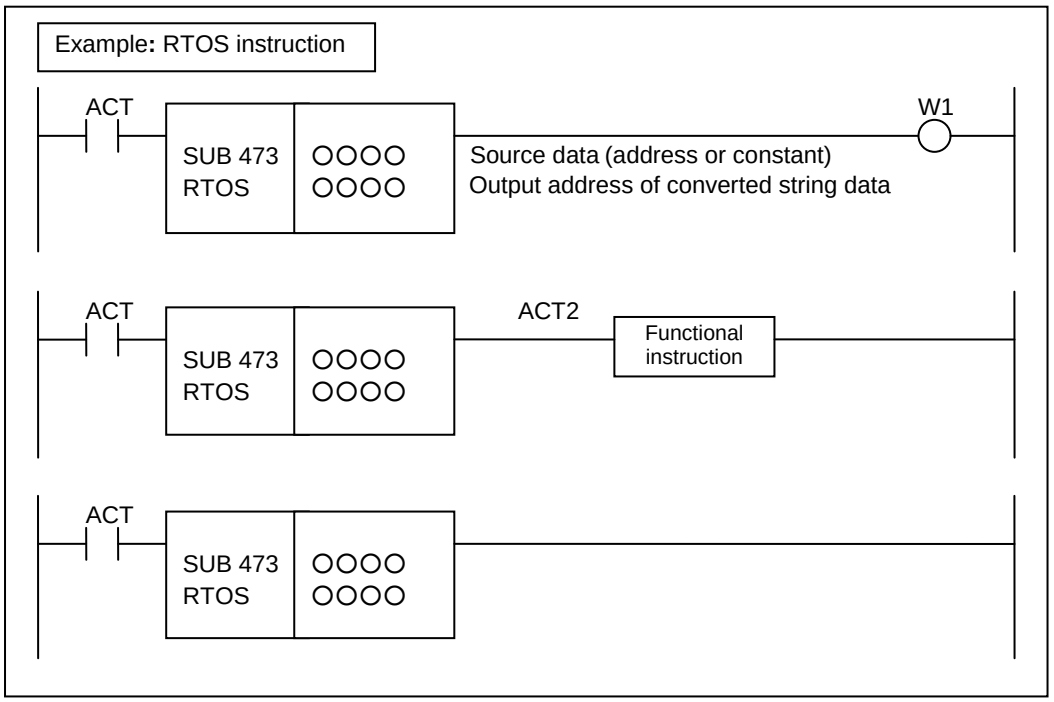


Fig. 5.3.11 (b) Format of RTOS, LTOS instruction

Table 5.3.11 (b) Mnemonic of RTOS, LTOS instruction

Mnemonic format

Step number	Instruction	Address No.	Bit No.	Remarks
1	RD	OOOO	.O	ACT
2	SUB	473		SUB No. (RTOS instruction)
3	(PRM)	OOOO		Source data (address or constant)
4	(PRM)	OOOO		Output address of conversion result
5	WRT	OOOO	.O	Normal end output

Memory status of control condition

ST3	ST2	ST1	ST0
			ACT
			↓
			↓
			↓
			W1

Control conditions

(a) Input signal (ACT)
ACT=0: Do not execute the instruction.
ACT=1: Execute the instruction.

Parameters

(a) Source data
Specify the source data. You can specify the constant or PMC memory address.

(b) Output address of conversion result
Specify PMC memory address where the conversion result will be stored.

NOTE
Refer to the section “5.1.1 String data" for the specification of the string data.

⚠ CAUTION
Be careful not to exceed the range of the PMC address, when setting the above addresses.

NOTE
1 An +infinity data is set in the PMC parameter, “1.#INF” is output.
2 An -infinity data is set in the PMC parameter, “-1.#INF” is output.
3 A NAN data (Not a Number) is set in the PMC parameter, “1.#NAN” or “-1.#NAN” is output.

OUTPUT (W1)

W1=1 : The operation has finished normally.
W1=0 : The operation was not executed (ACT=0).

NOTE
W1 can be omitted. Or it can be also connected to another functional instruction instead of a coil.

5.3.12 STOB (Conversion(Character string→1-byte signed binary) : SUB 475)
STOW (Conversion(Character string→2-byte signed binary) : SUB 476)
STOD (Conversion(Character string→4-byte signed binary) : SUB 477)

These instructions convert the string data to 1, 2 or 4 byte signed binary data and output it.
At the beginning of the string, you can use "+" for positive numbers and "-" for negative numbers.
When a character that cannot be converted to an integer is in the string data, it outputs the integer converted from the string data before the invalid character.

There are the following instructions according to the data type of parameters.

Table 5.3.12 (a) Kinds of data type conversion instruction

	Instruction name	Sub No.	Data type
1	STOB	475	1-byte signed data(-128 ~ 127)
2	STOW	476	2-byte signed data (-32768 ~ 32767)
3	STOD	477	4-byte signed data (-2147483648 ~ 2147483647)

NOTE
When the string data is converted to the number that is out of data type range, W1 = 0 and no data is outputted.

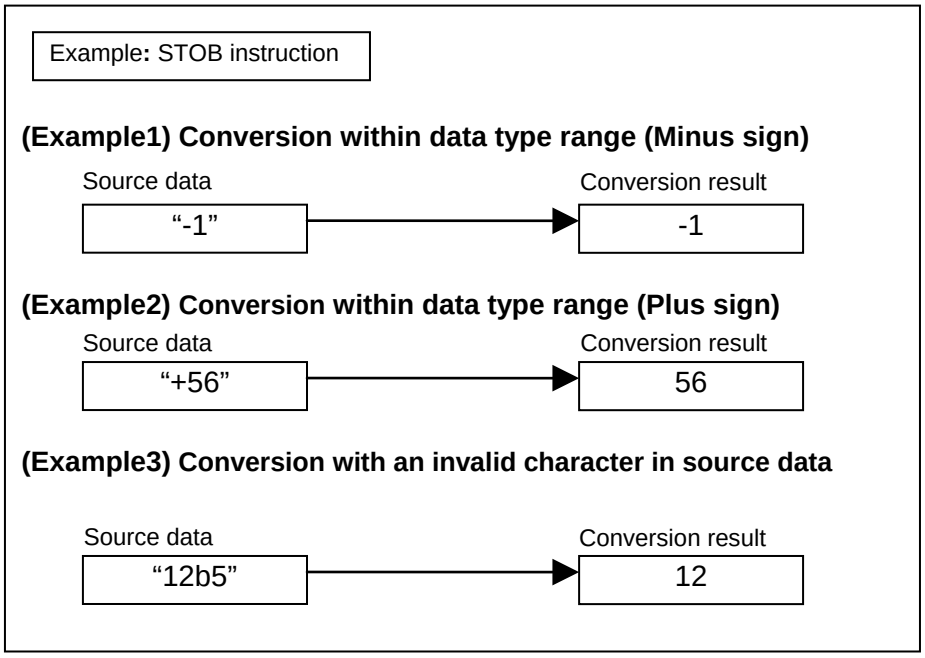


Fig. 5.3.12 (a) Operation of STOB, STOW, STOD instruction

Format

The ladder format and the mnemonic format of these instructions are as follows.

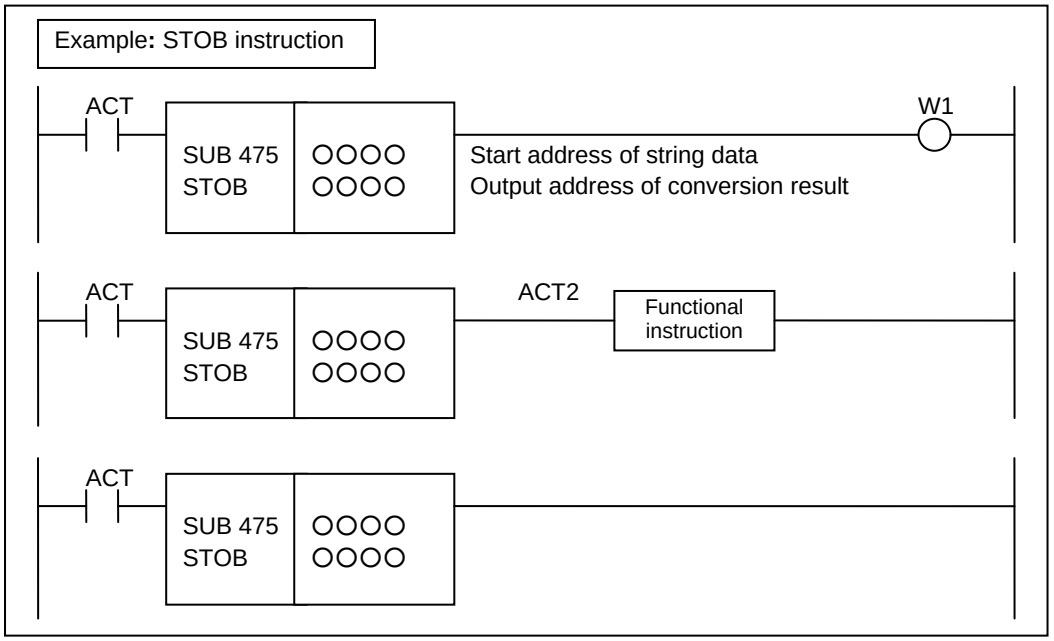


Fig. 5.3.12 (b) Format of STOB, STOW, STOD instructions

Table 5.3.12 (b) Mnemonic of STOB, STOW, STOD instruction

Mnemonic format

Step number	Instruction	Address No.	Bit No.	Remarks
1	RD	OOOO	.O	ACT
2	SUB	475		SUB No. (STOB instruction)
3	(PRM)	OOOO		Start address of string data
4	(PRM)	OOOO		Output address of conversion result
5	WRT	OOOO	.O	Normal end output

Memory status of control condition

ST3	ST2	ST1	ST0
			ACT
			↓
			↓
			↓
			W1

Control conditions

(a) Input signal (ACT)
ACT=0: Do not execute the instruction.
ACT=1: Execute the instruction.

Parameters

(a) Start address of string data
Specify the start address of the string data.

(b) Output address of conversion result
Specify PMC memory address where the conversion result will be stored.

NOTE
Refer to the section “5.1.1 String data" for the specification of the string data.

⚠ CAUTION
Be careful not to exceed the range of the PMC address, when setting the above addresses.

OUTPUT (W1)

W1=1 : The operation has finished normally.
W1=0 : The operation was not executed (ACT=0).
Or the number is outside the range of the data type.

NOTE
W1 can be omitted. Or it can be also connected to another functional instruction instead of a coil.

5.3.13 STOR (Conversion (Character string→Single Precision Real) : SUB 478)
STOL (Conversion (Character string→Double Precision Real) : SUB 479)

These instructions convert the string data to real number data and output it.
At the beginning of the string, you can use "+" for positive numbers and "-" for negative numbers.
When a character that cannot be converted to real number is in the string data, it outputs the real number converted from the string data before the invalid character.

There are the following instructions according to the data type of parameters.

Table 5.3.13 (a) Kinds of data type conversion instruction

	Instruction name	SUB No.	Data type
1	STOR	478	Single precision real (4 bytes)
2	STOL	479	Double precision real (8 bytes)

NOTE
1 When the string data is converted to a value that is out of the real number range, an error is generated in the conversion result and the conversion result is rounded into valid digits of the real number.
2 Because the processing time of these instructions can be long depending on the data to convert, please check the scan time when you use them.

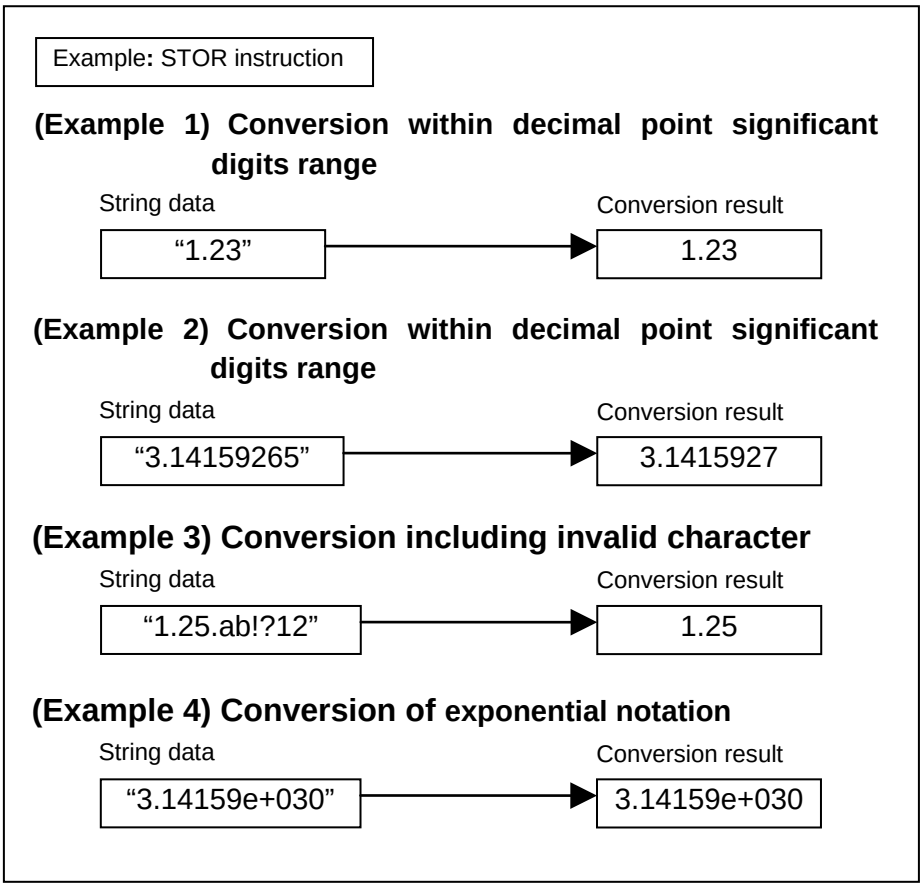


Fig. 5.3.13 (a) Operation of STOR, STOL instruction

Format

The ladder format and the mnemonic format of these instructions are as follows.

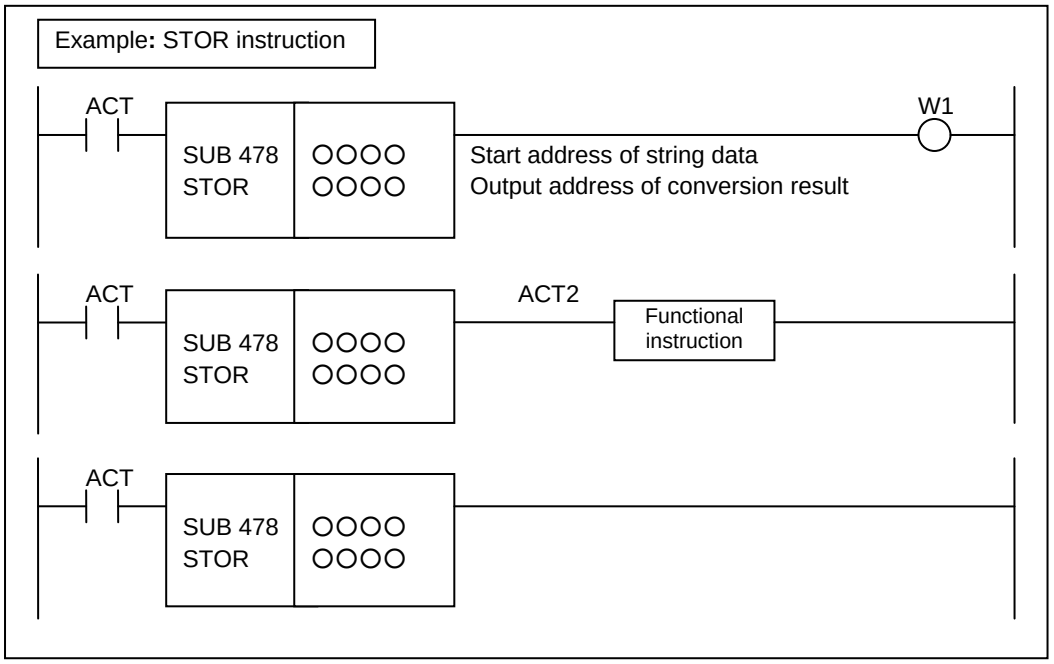


Fig. 5.3.13 (b) Format of STOR, STOL instruction

Table 5.3.13 (b) Mnemonic of STOR, STOL instruction

Mnemonic format

Step number	Instruction	Address No.	Bit No.	Remarks
1	RD	OOOO	.O	ACT
2	SUB	478		SUB No. (STOR instruction)
3	(PRM)	OOOO		Start address of string data
4	(PRM)	OOOO		Output address of conversion result
5	WRT	OOOO	.O	Normal end output

Memory status of control condition

ST3	ST2	ST1	ST0
			ACT
			↓
			↓
			W1

Control conditions

(a) Input signal (ACT)
ACT=0: Do not execute the instruction.
ACT=1: Execute the instruction.

Parameters

(a) Start address of string data
Specify the start address of the string data.

(b) Output address of conversion result
Specify PMC memory address where the conversion result will be stored.

NOTE
Refer to the section “5.1.1 String data" for the specification of the string data.

⚠ CAUTION
Be careful not to exceed the range of the PMC address, when setting the above addresses.

OUTPUT (W1)

W1=1 : The operation has finished normally.
W1=0 : The operation was not executed (ACT=0).

NOTE
W1 can be omitted. Or it can be also connected to another functional instruction instead of a coil.

5.3.14 BTOSH (Conversion (1-byte signed binary→Hexadecimal character string) : SUB 480)
WTOSH (Conversion (2-byte signed binary→Hexadecimal character string) : SUB 481)
DTOSH (Conversion (4-byte signed binary→Hexadecimal character string) : SUB 482)

These instructions convert a 1, 2 or 4 byte signed binary data to hexadecimal string data.
The values of AH to FH will be converted to uppercase alphabet of "A" to "F".
00H (NULL) is appended at the end of the outputted string data.

There are the following instructions according to the data type of parameters.

Table 5.3.14 (a) Kinds of data conversion instruction

	Instruction name	SUB No.	Data type
1	BTOSH	480	1-byte signed binary
2	WTOSH	481	2-byte signed binary
3	DTOSH	482	4-byte signed binary

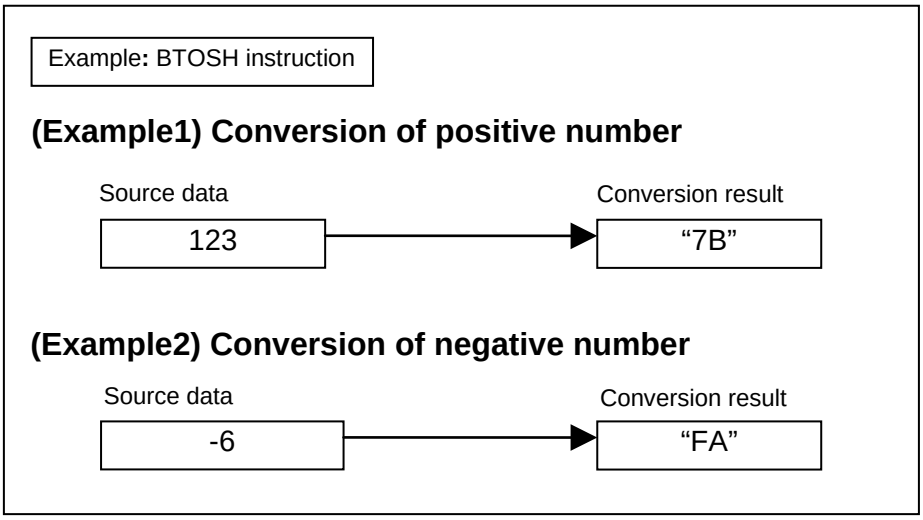


Fig. 5.3.14 (a) Operation of BTOSH, WTOSH, DTOSH instruction

Format

The ladder format and the mnemonic format of these instructions are as follows.

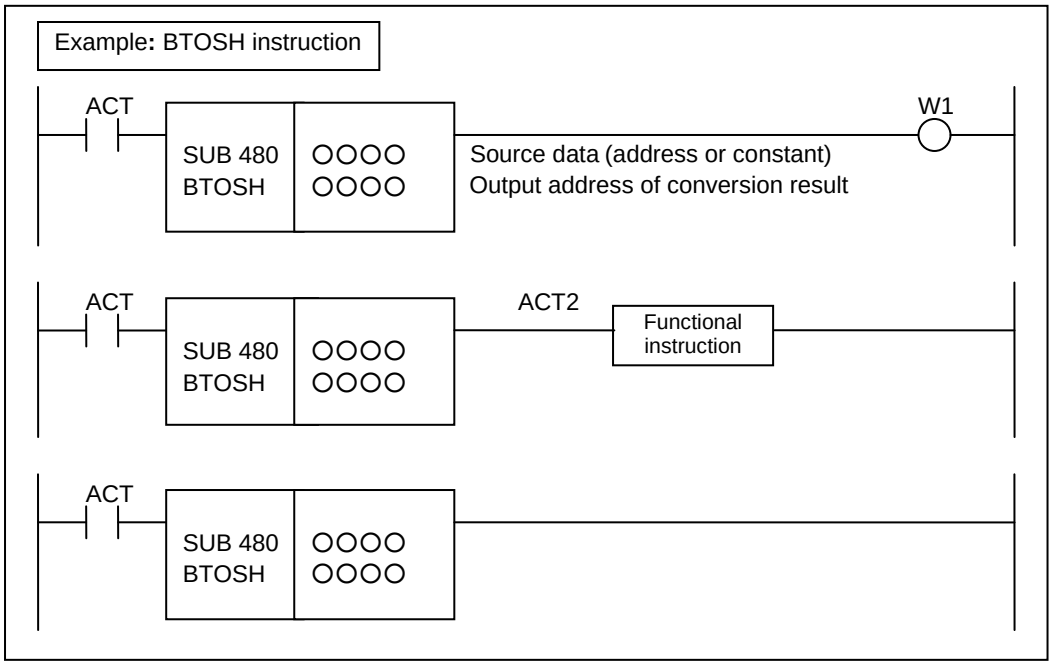


Fig. 5.3.14 (b) Format of BTOSH, WTOSH, DTOSH instruction

Table 5.3.14 (b) Mnemonic of BTOSH, WTOSH, DTOSH instruction

Mnemonic format

Step number	Instruction	Address No.	Bit No.	Remarks
1	RD	OOOO	.O	ACT
2	SUB	480		SUB No. (BTOSH instruction)
3	(PRM)	OOOO		Source data (address or constant)
4	(PRM)	OOOO		Output address of conversion result
5	WRT	OOOO	.O	Normal end output

Memory status of control condition

ST3	ST2	ST1	ST0
			ACT
			↓
			↓
			↓
			W1

Control conditions

(a) Input signal (ACT)
ACT=0: Do not execute the instruction.
ACT=1: Execute the instruction.

Parameters

(a) Source data
Specify the source data. You can specify the constant or PMC memory address.

(b) Output address of conversion result
Specify PMC memory address where the conversion result will be stored.

> ⚠ **CAUTION**
> Be careful not to exceed the range of the PMC address, when setting the above addresses.

OUTPUT (W1)

W1=1 : The operation has finished normally.
W1=0 : The operation was not executed (ACT=0).

> **NOTE**
> W1 can be omitted. Or it can be also connected to another functional instruction instead of a coil.

5.3.15 SHTOB (Conversion(Hexadecimal character string→1-byte signed binary) : SUB 483)
SHTOW (Conversion(Hexadecimal character string→2-byte signed binary) : SUB 484)
SHTOD (Conversion(Hexadecimal character string→4-byte signed binary) : SUB 485)

These instructions convert a hexadecimal string data to a 1, 2 or 4 byte signed binary data
When a character that cannot be converted to an integer is in the string data, it outputs the integer converted from the string data before the invalid character.

There are the following instructions according to the data type of parameters.

Table 5.3.15 (a) Kinds of data type conversion instruction

	Instruction name	Sub No.	Data type
1	SHTOB	483	1-byte signed binary
2	SHTOW	484	2-byte signed binary
3	SHTOD	485	4-byte signed binary

NOTE
When the string data is converted to the number that is out of data type range, W1 = 0 and no data is outputted.

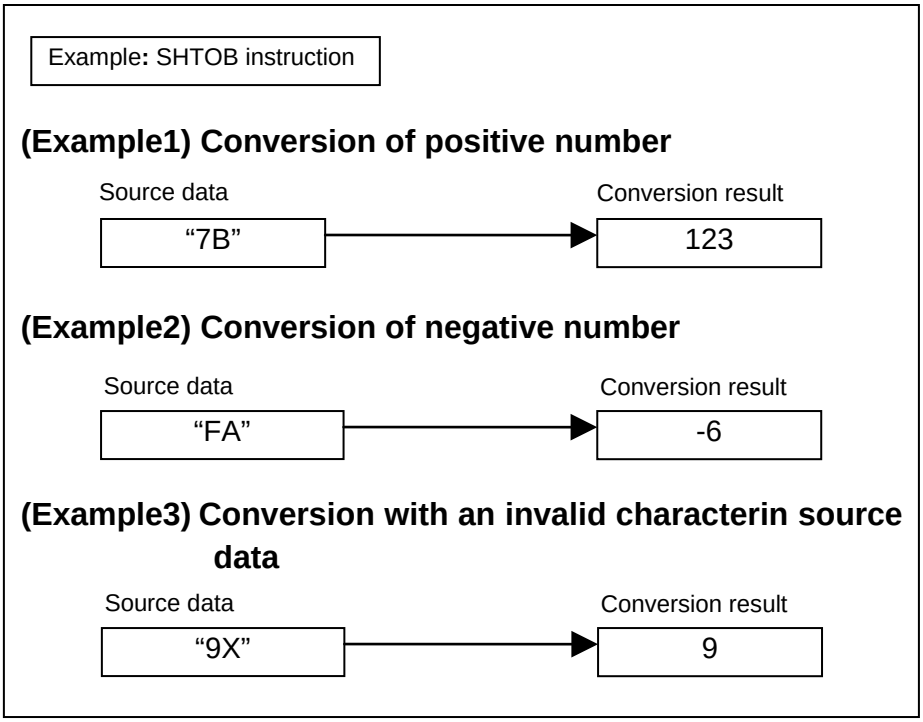


Fig. 5.3.15 (a) Operation of SHTOB, SHTOW, SHTOD instruction

Format

The ladder format and the mnemonic format of these instructions are as follows.

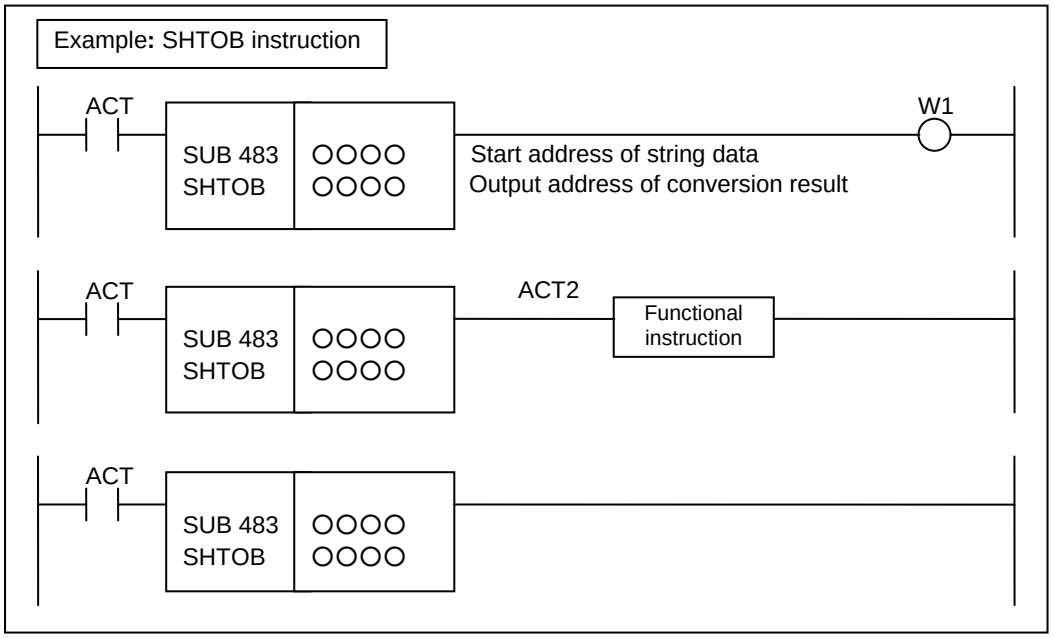


Fig. 5.3.15 (b) Format of SHTOB, SHTOW, SHTOD instructions

Table 5.3.15 (b) Mnemonic of SHTOB, SHTOW, SHTOD instruction

Mnemonic format

Step number	Instruction	Address No.	Bit No.	Remarks
1	RD	OOOO	.O	ACT
2	SUB	483		SUB No. (SHTOB instruction)
3	(PRM)	OOOO		Start address of string data
4	(PRM)	OOOO		Output address of conversion result
5	WRT	OOOO	.O	Normal end output

Memory status of control condition

ST3	ST2	ST1	ST0
			ACT
			↓
			↓
			↓
			W1

Control conditions

(a) Input signal (ACT)
ACT=0: Do not execute the instruction.
ACT=1: Execute the instruction.

Parameters

(a) Start address of string data
Specify the start address of the string data.

(b) Output address of conversion result
Specify PMC memory address where the conversion result will be stored.

NOTE
Refer to the section “5.1.1 String data" for the specification of the string data.

⚠ CAUTION
Be careful not to exceed the range of the PMC address, when setting the above addresses.

OUTPUT (W1)

W1=1 : The operation has finished normally.
W1=0 : The operation was not executed (ACT=0).
Or the number is outside the range of the data type.

NOTE
W1 can be omitted. Or it can be also connected to another functional instruction instead of a coil.

5.3.16 MOVS (Copying Character string : SUB 486)

This instruction copies string data from the specified source address to the specified destination address.

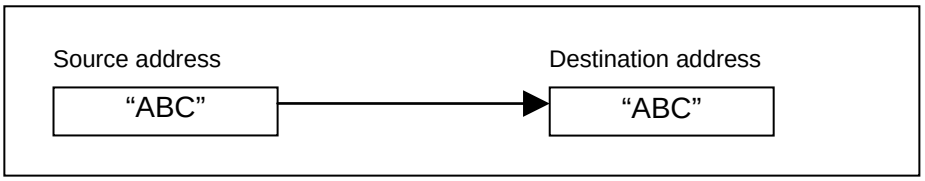


Fig. 5.3.16 (a) Operation of MOVS instruction

Format

The ladder format and the mnemonic format of this instruction are as follows.

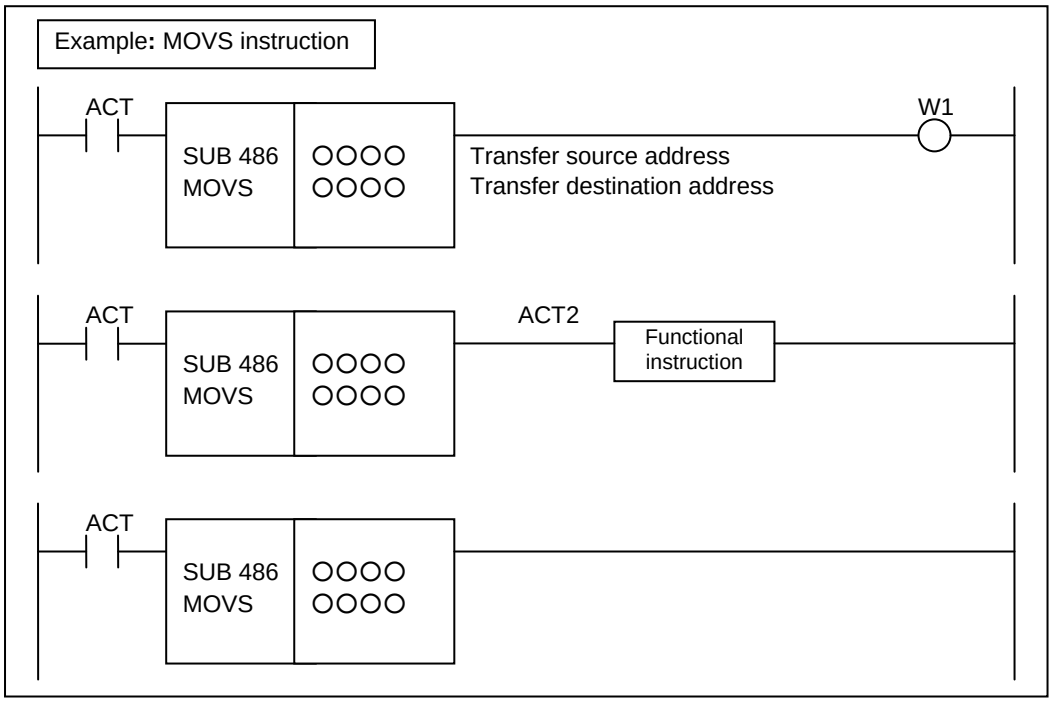


Fig. 5.3.16 (b) Format of MOVS instructions

Table 5.3.16 (a) Mnemonic of MOVS instruction

Mnemonic format

Step number	Instruction	Address No.	Bit No.	Remarks
1	RD	OOOO	.O	ACT
2	SUB	486		SUB No. (MOVS instruction)
3	(PRM)	OOOO		Transfer source address
4	(PRM)	OOOO		Transfer destination address
5	WRT	OOOO	.O	Normal end output

Memory status of control condition

ST3	ST2	ST1	ST0
			ACT
			↓
			W1

Control conditions

(a) Input signal (ACT)
ACT=0: Do not execute the instruction.
ACT=1: Execute the instruction.

Parameters

(a) Transfer source address
Specify the source string data address for the transfer.

(b) Transfer destination address
Specify the destination string data address for the transfer.

NOTE
Refer to the section “5.1.1 String data" for the specification of the string data.

⚠ CAUTION
Be careful not to exceed the range of the PMC address, when setting the above addresses.

OUTPUT (W1)

W1=1 : The operation has finished normally.
W1=0 : The operation was not executed (ACT=0).

NOTE
W1 can be omitted. Or it can be also connected to another functional instruction instead of a coil.

5.3.17 SCOMP(Comparing two Character strings : SUB 487)

This instruction compares “comparison string data 1” and “comparison string data 2” to judge whether they are identical or not.

Format

The ladder format and the mnemonic format of this instruction are as follows.

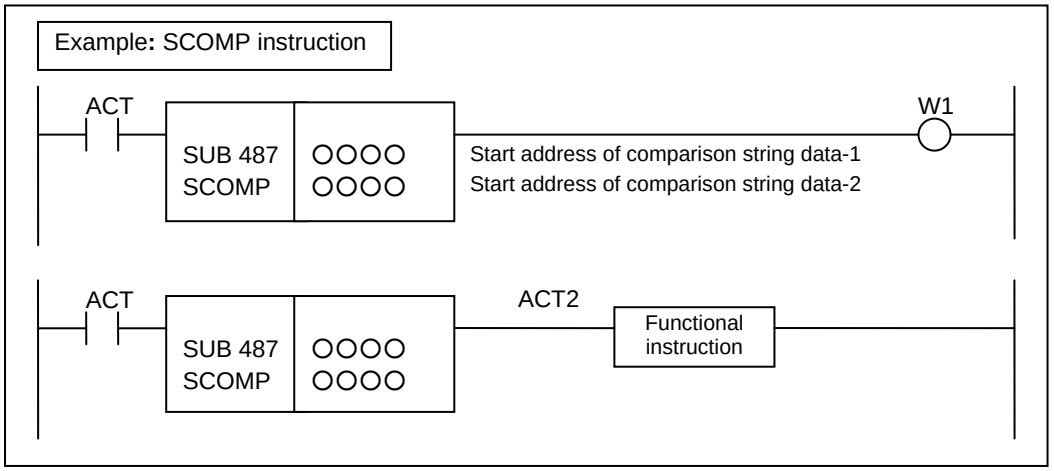


Fig. 5.3.17 (a) Format of SCOMP instructions

Table 5.3.17 (a) Mnemonic of SCOMP instruction

Mnemonic format					Memory status of control condition			
Step number	Instruction	Address No.	Bit No.	Remarks	ST3	ST2	ST1	ST0
1	RD	OOOO	.O	ACT				ACT
2	SUB	487		SUB No. (SCOMP instruction)				↓
3	(PRM)	OOOO		Start address of comparison string data-1				↓
4	(PRM)	OOOO		Start address of comparison string data-2				↓
5	WRT	OOOO	.O	Result				W1

Control conditions

(a) Input signal (ACT)
ACT=0: Do not execute the instruction.
ACT=1: Execute the instruction. The result is output to W1.

Parameters

(a) Start address of the comparison string data-1
Specify the start address of comparison string data-1 data.

(b) Start address of the comparison string data-2
Specify the start address of comparison string data-2 data.

NOTE
Refer to the section “5.1.1 String data" for the specification of the string data.

OUTPUT (W1)

The result is output to W1.

W1=1 : “Comparison string data-1”＝“Comparison string data-2”.

W1=0 : The operation was not executed (ACT=0).
Or “Comparison string data-1”≠“Comparison string data-2” (ACT=1).

5.3.18 UPPER (Character string Conversion (Lower case→Upper case) : SUB 488)

This instruction converts lowercase characters into uppercase characters and output the converted string data to the specified address.
00H (NULL) is appended at the end of the outputted string data.
Characters that cannot be converted are output as they are.

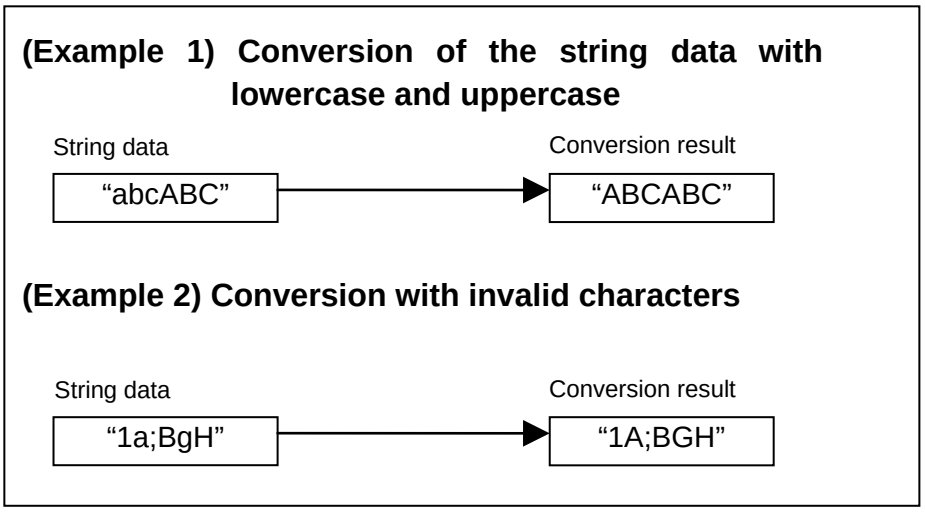


Fig. 5.3.18 (a) Operation of UPPER instruction

Format

The ladder format and the mnemonic format of this instruction are as follows.

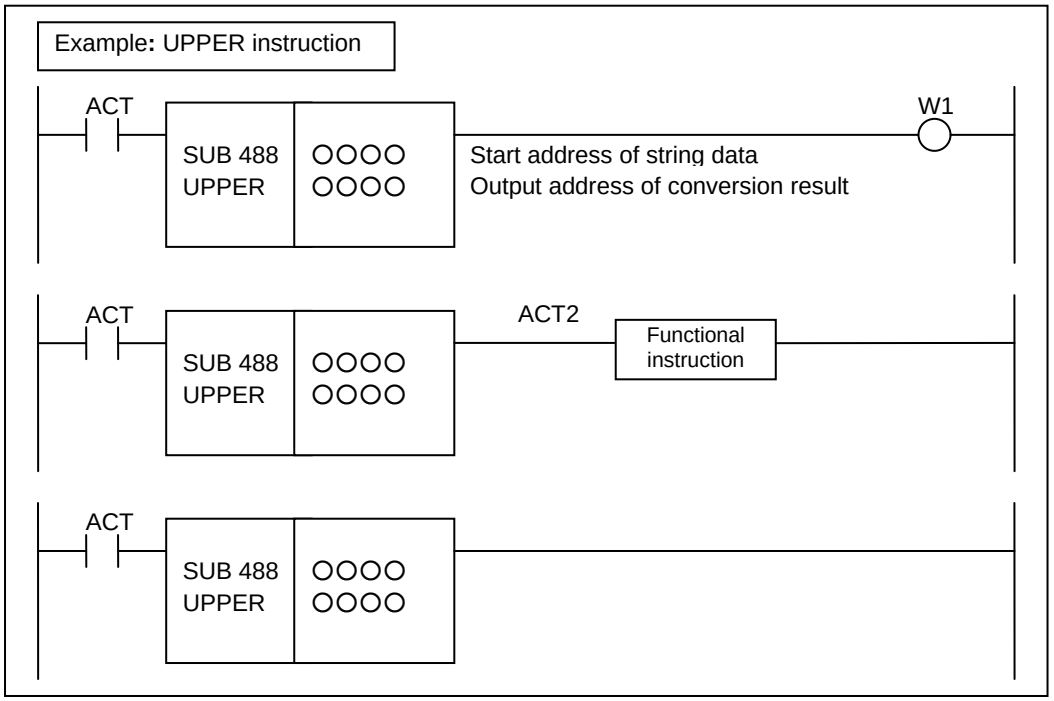


Fig. 5.3.18 (b) Format of UPPER instruction

Table 5.3.18 (a) Mnemonic of UPPER instruction

Mnemonic format					Memory status of control condition			
Step number	Instruction	Address No.	Bit No.	Remarks	ST3	ST2	ST1	ST0
1	RD	OOOO	.O	ACT				ACT
2	SUB	488		SUB No. (UPPER instruction)				↓
3	(PRM)	OOOO		Start address of string data				↓
4	(PRM)	OOOO		Output address of conversion result				↓
5	WRT	OOOO	.O	Normal end output				W1

Control conditions

(a) Input signal (ACT)
ACT=0: Do not execute the instruction.
ACT=1: Execute the instruction.

Parameters

(a) Start address of string data
Specify the start address of the string data.

(b) Output address of conversion result
Specify PMC memory address where the conversion result will be stored.

NOTE
Refer to the section “5.1.1 String data" for the specification of the string data.

⚠ CAUTION
Be careful not to exceed the range of the PMC address, when setting the above addresses.

OUTPUT (W1)

W1=1 : The operation has finished normally.
W1=0 : The operation was not executed (ACT=0).

NOTE
W1 can be omitted. Or it can be also connected to another functional instruction instead of a coil.

5.3.19 LOWER (Character string Conversion (Upper case→Lower case) : SUB 489)

This instruction converts uppercase characters into lowercase characters and output the converted string data to the specified address.
00H (NULL) is appended at the end of the outputted string data.
Characters that cannot be converted are output as they are.

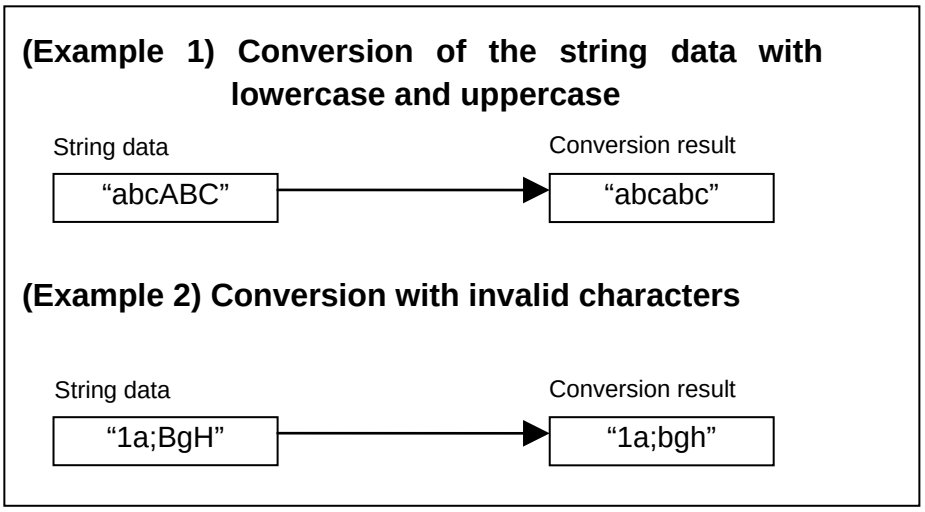


Fig. 5.3.19 (a) Operation of LOWER instruction

Format

The ladder format and the mnemonic format of this instruction are as follows.

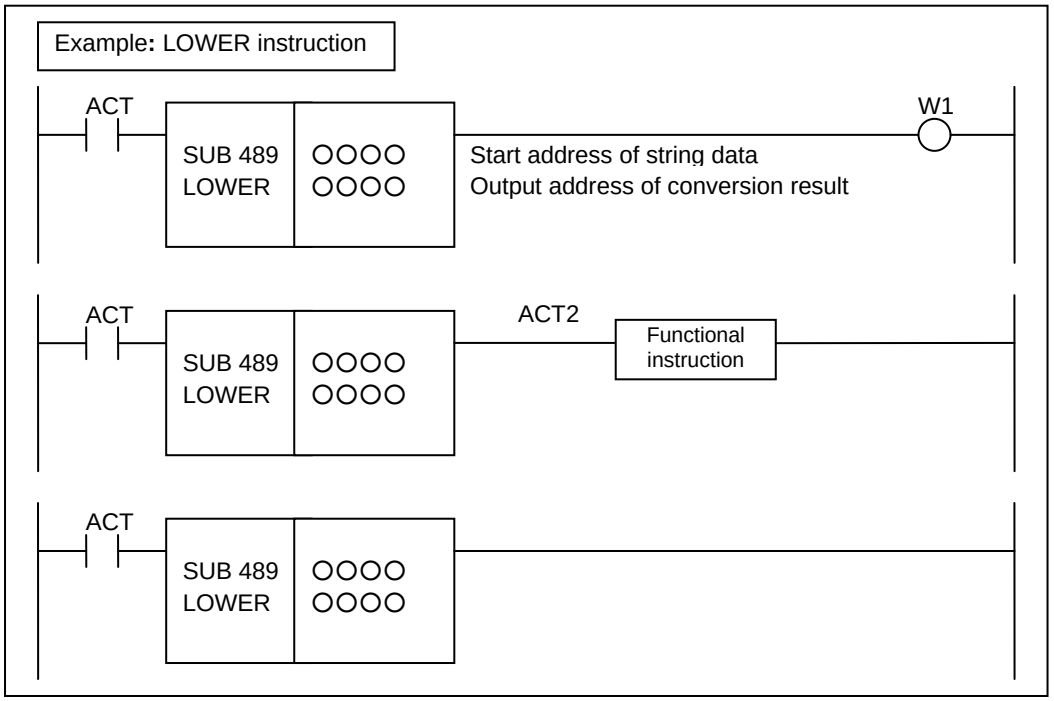


Fig. 5.3.19 (b) Format of LOWER instruction

Table 5.3.19 (a) Mnemonic of LOWER instruction

Mnemonic format

Step number	Instruction	Address No.	Bit No.	Remarks
1	RD	○○○○	.○	ACT
2	SUB	489		SUB No. (LOWER instruction)
3	(PRM)	○○○○		Start address of string data
4	(PRM)	○○○○		Output address of conversion result
5	WRT	○○○○	.○	Normal end output

Memory status of control condition

ST3	ST2	ST1	ST0
			ACT
			↓
			↓
			↓
			W1

Control conditions

(a) Input signal (ACT)
ACT=0: Do not execute the instruction.
ACT=1: Execute the instruction.

Parameters

(a) Start address of string data
Specify the start address of the string data.

(b) Output address of conversion result
Specify PMC memory address where the conversion result will be stored.

NOTE
Refer to the section “5.1.1 String data" for the specification of the string data.

⚠ CAUTION
Be careful not to exceed the range of the PMC address, when setting the above addresses.

OUTPUT (W1)

W1=1 : The operation has finished normally.
W1=0 : The operation was not executed (ACT=0).

NOTE
W1 can be omitted. Or it can be also connected to another functional instruction instead of a coil.

5.3.20 SFILL (Filling Character string : SUB 490)

This instruction outputs the specified number of the specified characters in series.
00H (NULL) is appended at the end of the outputted string data.
When 0 is specified as the specified number of characters, 00H (NULL) is output.
When the number of characters is 256 or more, 255 characters will be output.

NOTE
If a minus value is set to the number of characters, W1 = 0 and no data is output.

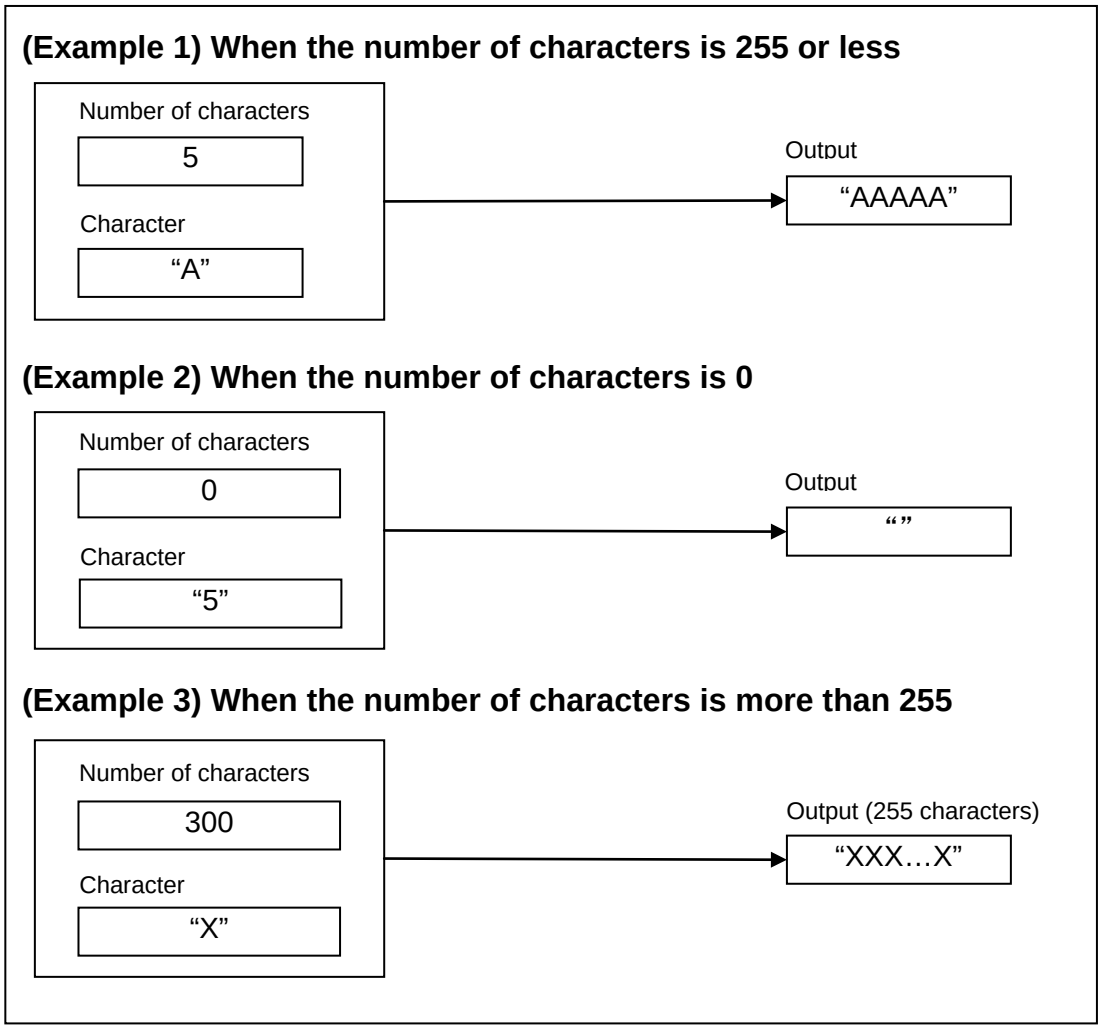


Fig. 5.3.20 (a) Operation of SFILL instruction

Format

The ladder format and the mnemonic format of this instruction are as follows.

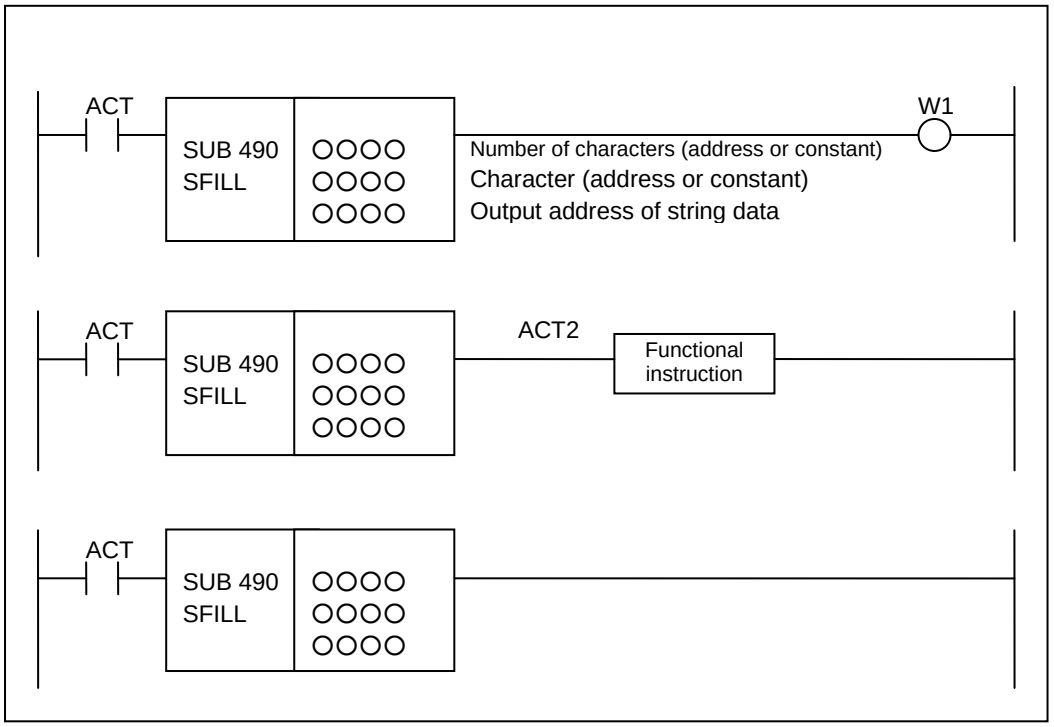


Fig. 5.3.20 (b) Format of SFILL instruction

Table 5.3.20 (a) Mnemonic of SFILL instruction

Mnemonic format					Memory status of control condition			
Step number	**Instruction**	**Address No.**	**Bit No.**	**Remarks**	**ST3**	**ST2**	**ST1**	**ST0**
1	RD	OOOO	.O	ACT				ACT
2	SUB	490		SUB No. (SFILL instruction)				
3	(PRM)	OOOO		Number of characters (address or constant)				
4	(PRM)	OOOO		Character (address or constant)				
5	(PRM)	OOOO		Output address of string data				
7	WRT	OOOO	.O	Normal end output				W1

Control conditions

(a) Input signal (ACT)
ACT=0: Do not execute the instruction..
ACT=1: Execute the instruction.

Parameters

(a) Number of characters
Specify the number of characters to be filled. In this parameter, a constant or a PMC memory address for storing 2-byte data can be specified.

(b) Character
Specify the characters to be filled with. In this parameter, a constant or a PMC memory address for storing 1-byte data can be specified.

(c) Output address of string data
Specify PMC memory address where the characters will be stored from.

NOTE
Refer to the section “5.1.1 String data" for the specification of the string data.

⚠ CAUTION
Be careful not to exceed the range of the PMC address, when setting the above addresses.

OUTPUT (W1)

W1=1 : The operation has finished normally.
W1=0 : The operation was not executed (ACT=0).
A minus value was specified for the number of characters.

NOTE
W1 can be omitted. Or it can be also connected to another functional instruction instead of a coil.

5.4 RS-232C control instructions

The following types of RS-232C control instruction are available.

	Instruction name	SUB No.	Processing
1	RSOPN	491	Opening RS-232C channel
2	RSRCV	492	Receiving data from RS-232C channel
3	RSSND	493	Sending data to RS-232C channel
4	RSCLS	494	Closing RS-232C channel

5.4.1 Execution of RS-232C control instructions

The RS-232C control instructions are the function that sends or receives a data between CNC and a serial device with RS-232C cable. The instructions may need several scans of sequence program to complete the operation.

NOTE
Depending on the status of CNC and the amount of data to transfer, the number of scans to complete the operation may change.

When using the RS-232C control instruction, turn ACT to 0 immediately after the completion information (W1) is set to 1. Please refer to the following caution for details.

⚠ CAUTION
The RS-232C control instructions are exclusively controlled with other RS-232C control instructions. Turn ACT to 0 immediately after the completion information (W1) is set to 1 during continuous transmission / reception processing. If you leave ACT=1 of an RS-232C control instruction with W1=1, other RS-232C control instructions do not work.

Cable connection example

Please modify the wiring between CNC and a serial device according to the specification of the device.

NOTE

These instructions don't support hardware flow control. Please short-circuit the RS and CS signal terminals.

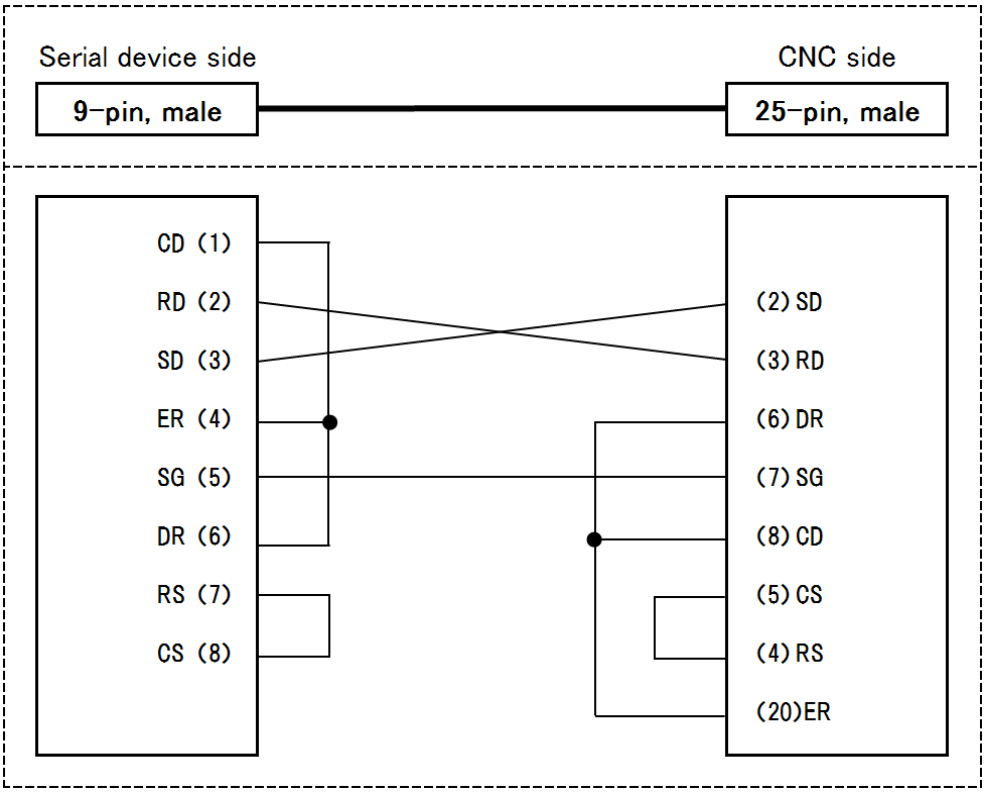


Fig. 5.4.1 (a) cable wiring example

5.4.2 RSOPN (Opening RS-232C channel : SUB 491)

This instruction opens a serial interface with the channel No. and the RS-232C control data.
Data communication via RS-232C will be available by opening the serial interface by this instruction.
Only one channel can be opened at the same time.
If opening a channel when other channel is already opened, it will fail and the completion code 5 will be returned. Please call this instruction again after closing the currently opening channel.

NOTE
1 Close a channel with the RSCLS (SUB 491) instruction after finishing the RS-232C processing on the channel that is opened with this instruction. One channel cannot be double-opened. Other functions cannot access the channel while the process of opening the channel is executed.
2 Opening a channel should be executed while no data is sent or received.

Format

The ladder format and the mnemonic format of this instruction are as follows.

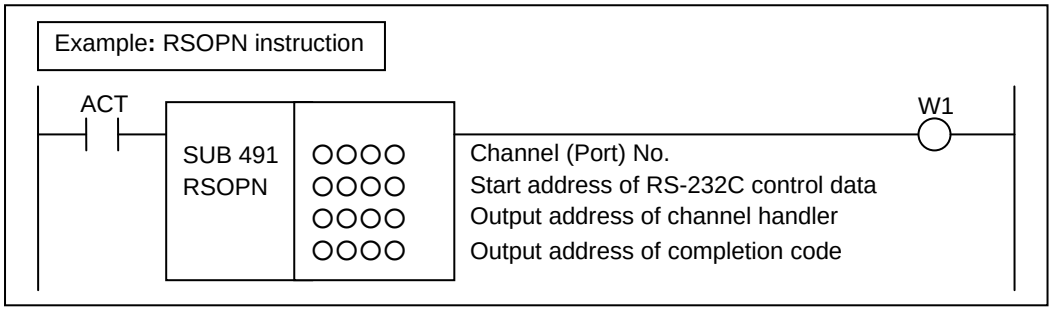


Fig. 5.4.2 (a) Format of SFIND instruction

Table 5.4.2 (a) Mnemonic of RSOPN instruction

Mnemonic format					Memory status of control condition			
Step number	Instruction	Address No.	Bit No.	Remarks	ST3	ST2	ST1	ST0
1	RD	OOOO	.O	ACT				ACT
2	SUB	491		SUB No. (RSOPN Instruction)				
3	(PRM)	OOOO		Channel (Port) No.				
4	(PRM)	OOOO		Start address of RS-232C control data				
5	(PRM)	OOOO		Output address of channel handler				
6	(PRM)	OOOO		Output address of completion code				
7	WRT	OOOO	.O	Completion output				W1

Control conditions

(a) Input signal (ACT)
ACT=0: Do not execute the instruction.
ACT=1: Execute the instruction.

Parameters

(a) Channel (Port)

Specify the channel (port) to be used.
In this parameter, a constant or a PMC memory address for storing 2-byte data can be specified.
Available data range is as follows.
If the specified data is out of the following range, the channel cannot be opened.

Setting data	RS-232C channel (Port)
1	JD56A
2	JD36A/JD54

(b) Start address of RS-232C control data

Specify the PMC memory address where the following RS-232C control data is stored.
RS-232C control data is the setting data table for RS-232C communication.
Unless these data are properly set, the channel cannot be opened.
First, you have to configure the control data properly, referring to the explanation in this section.

(i) Baud rate

Set the baud rate.
1: 1200
2: 2400
3: 4800
4: 9600
5: 19200

(ii) Stop bit

Set bit length which indicates the end of a character.
0: 1 bit
1: 2 bit

(iii) Receiving timeouts

Set the timeout value (in seconds) used for receiving data.
0: No timeout
1～127: Specified number of seconds

(iv) Sending timeouts

Set the timeout value (in seconds) used for sending data.
0: No timeout
1～127: Specified number of seconds

(v) Terminate ASCII code

Set the ASCII code that indicates the end of received data.
0: None
1: CR (0DH)
2: LF (0AH)

NOTE
Hardware and software flow controls are not supported and cannot be set.

Structure of RS-232C control data

Start address		
+0	(Reserved) 0 (1 byte)	Set 0 to this field.
+1	Baud rate (1 byte)	
+2	Stop bit (1 byte)	
+3	(Reserved) 0 (3 bytes)	Set 0 to this field.
+6	Receiving timeouts (1 byte)	
+7	Sending timeouts (1 byte)	
+8	Terminate ASCII code (1 byte)	
+9	(Reserved) 0 (7 bytes)	Set 0 to this field.
+15		

(c) Output address of channel handler
Specify the PMC memory address where the 4-byte data of channel handler will be stored.

(d) Output address of completion code
Specify the PMC memory address where the 1-byte data of completion code will be stored.
When the operation has completed, the following completion code is set to the specified address.

Completion code	Details
0	Normal end
1	The specified channel is invalid
2	RS-232C control data is invalid
3	This channel is being used in other function
4	The communication device is not connected or an error occurred
5	Other channel has been opened
101	Failed to open the channel

OUTPUT (W1)

W1=1 : The operation has been completed.
W1=0 : The operation is not executed (ACT=0) or is being processed now.

5.4.3 RSRCV (Receiving data from RS-232C channel : SUB 492)

This instruction receives the data from the serial interface opened in RSOPN (SUB 491) instruction.

NOTE
The channel handler used in this instruction is given in the RSOPN (SUB 491) instruction.

Format

The ladder format and the mnemonic format of this instruction are as follows.

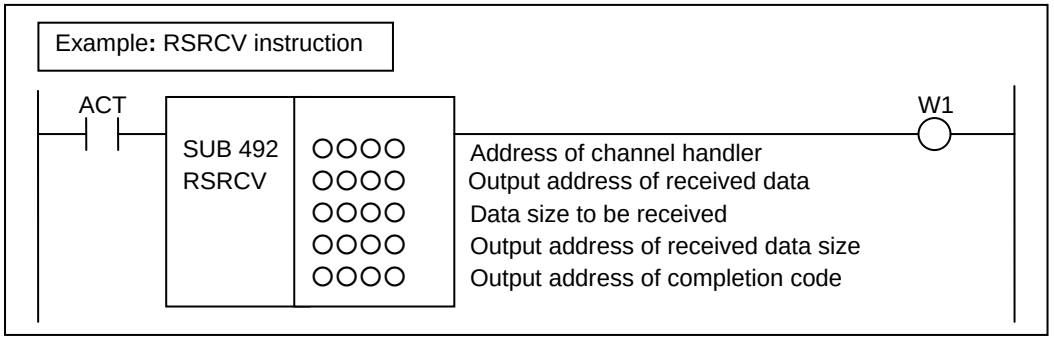


Fig. 5.4.3 (a) Format of RSRCV instruction

Table 5.4.3 (a) Mnemonic of RSRCV instruction

Mnemonic format

Step number	Instruction	Address No.	Bit No.	Remarks
1	RD	OOOO	.O	ACT
2	SUB	492		SUB No. (RSRCV instruction)
3	(PRM)	OOOO		Address of channel handler
4	(PRM)	OOOO		Output address of received data
5	(PRM)	OOOO		Data size to be received
6	(PRM)	OOOO		Output address of received data size
7	(PRM)	OOOO		Output address of completion code
8	WRT	OOOO	.O	Completion output

Memory status of control condition

ST3	ST2	ST1	ST0
			ACT
			↓
			↓
			↓
			↓
			↓
			↓
			W1

Control conditions

(a) Input signal (ACT)
ACT=0: Do not execute the instruction.
ACT=1: Execute the instruction.

Parameters

(a) Address of channel handler
Specify the PMC memory address where the 4-byte data of channel handler given in the RSOPN (SUB 491) instruction has been stored.

(b) Output address of received data
Specify the PMC memory address where the received data from RS-232C will be stored.

(c) Data size to be received
Specify the data size to be received.
In this parameter, a constant or a PMC memory address for storing 2-byte data can be specified.
Available data range is 1 to 512.

(d) Output address of received data size
Specify the PMC memory address where the 2-byte data of received data size will be stored.

(e) Output address of completion code
Specify the PMC memory address where the 1-byte data of completion code will be stored.
When the operation has completed, the following completion code is set to the specified address.

Completion code	Details
0	Normal end
4	The communication device is not connected or an error occurred (NOTE)
6	The specified channel handler is invalid
7	Timeout occurred in the data receiving
8	The specified data size to be received is invalid
9	Received data is invalid. Please check the RS-232C control data (NOTE)
10	Overrun occurred in the data receiving (NOTE)
11	The data to be received will exceed the specified PMC memory area
102	Failed to receive data (NOTE)

⚠ CAUTION
Set the character string address and the output address appropriately not to exceed the PMC address area.

NOTE
When those completion codes were returned, the communication cannot be performed since then. To restart the communication, close the channel with RSCLS (SUB 494) instruction and reopen the channel with RSOPN (SUB 491) instruction.

OUTPUT (W1)

W1=1 : The operation has been completed.
W1=0 : The operation is not executed (ACT=0) or is being processed now.

5.4.4 RSSND (Sending data to RS-232C channel : SUB 493)

This instruction sends the data to the serial interface opened in RSOPN (SUB 491) instruction.

NOTE
The channel handler used in this instruction is given in the RSOPN (SUB 491) instruction.

Format

The ladder format and the mnemonic format of this instruction are as follows.

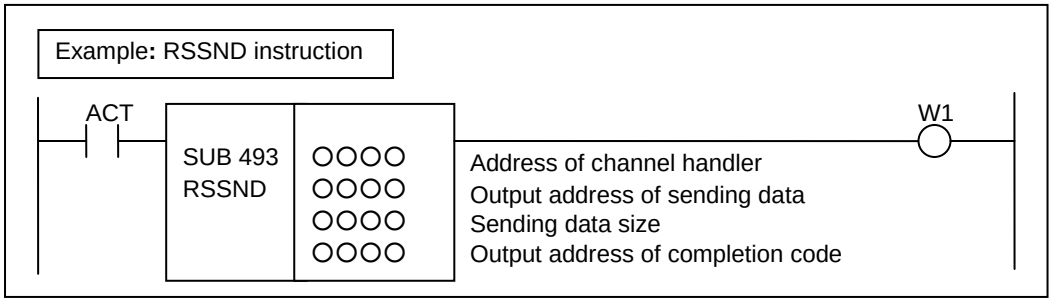


Fig. 5.4.4 (b) Format of RSSND instruction

Table 5.4.4 (a) Mnemonic of RSSND instruction

Mnemonic format					Memory status of control condition			
Step number	Instruction	Address No.	Bit No.	Remarks	ST3	ST2	ST1	ST0
1	RD	OOOO	.O	ACT				ACT
2	SUB	493		SUB No. (RSSND instruction)				
3	(PRM)	OOOO		Address of Channel handler				
4	(PRM)	OOOO		Stored address of sending data				
5	(PRM)	OOOO		Sending data size				
6	(PRM)	OOOO		Output address of completion code				
7	WRT	OOOO	.O	Completion output				W1

Control conditions

(a) Input signal (ACT)
ACT=0: Do not execute the instruction.
ACT=1: Execute the instruction.

Parameters

(a) Address of Channel handler
Specify the PMC memory address where the 4-byte data of channel handler given in the RSOPN (SUB 491) instruction has been stored.

(b) Stored address of sending data
Specify the PMC memory address where the sending data to RS-232C is stored.

(c) Sending data size
Specify the data size that wants to send.
In this parameter, a constant or a PMC memory address for storing 2-byte data can be specified.
Available data range is 1 to 512.

(d) Output address of completion code
Specify the PMC memory address where the 1-byte data of completion code will be stored.
When the operation has completed, the following completion code is set to the specified address.

Completion code	Details
0	Normal end
4	The communication device is not connected or an error occurred (NOTE)
6	The specified channel handler is invalid
9	Received data is invalid. Check the RS-232C control data (NOTE)
10	Overrun occurred in the data receiving (NOTE)
12	Timeout occurred in the data sending
13	The specified sending data size is invalid
14	The PMC memory of sending data is exceeded
103	Failed to sending data (NOTE)

NOTE
When those completion codes were returned, the communication cannot be performed since then. To restart the communication, close the channel with RSCLS (SUB 494) instruction and reopen the channel with RSOPN (SUB 491) instruction.

OUTPUT (W1)

W1=1 : The operation has been completed.
W1=0 : The operation is not executed (ACT=0) or is being processed now.

5.4.5 RSCLS (Closing RS-232C channel : SUB 494)

This instruction closes the serial interface opened in RSOPN (SUB 491) instruction.

NOTE
The channel handler used in this instruction is given in the RSOPN (SUB 491) instruction.

Format

The ladder format and the mnemonic format of this instruction are as follows.

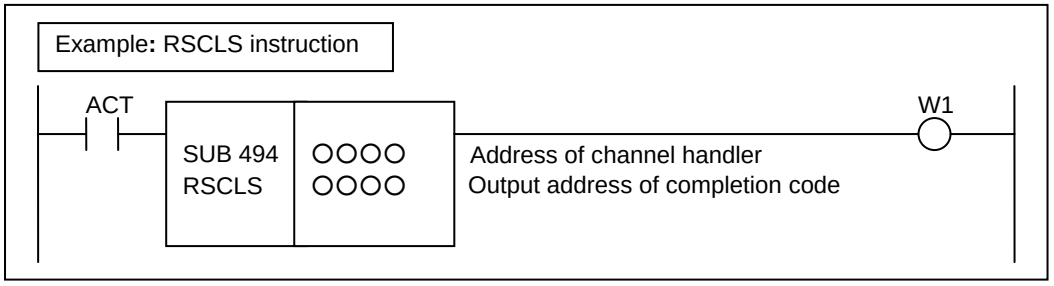


Fig. 5.4.5 (a) Format of RSCLS instruction

Table 5.4.5 (a) Mnemonic of RSCLS instruction

Mnemonic format					Memory status of control condition			
Step number	Instruction	Address No.	Bit No.	Remarks	ST3	ST2	ST1	ST0
1	RD	OOOO	.O	ACT				ACT
2	SUB	494		SUB No. (RSCLS instruction)				↓
3	(PRM)	OOOO		Address of channel handler				↓
4	(PRM)	OOOO		Output address of completion code				↓
5	WRT	OOOO	.O	Completion output				W1

Control conditions

(a) Input signal (ACT)
ACT=0: Do not execute the instruction.
ACT=1: Execute the instruction.

Parameters

(a) Address of channel handler
Specify the PMC memory address where the 4-byte data of channel handler given in the RSOPN (SUB 491) instruction has been stored.

(b) Output address of completion code
Specify the PMC memory address where the 1-byte data of completion code is stored.
When the operation has completed, the following completion code is set to the specified address.

Completion code	Details
0	Normal end
4	The communication device is not connected or an error occurred
6	The specified channel handler is invalid
104	Failed to close the serial interface

OUTPUT (W1)

W1=1 : The operation has been completed.
W1=0 : The operation is not executed (ACT=0) or is being processed now.

5.4.6 Example of programming

The followings are the example to receive data using RS-232C control instructions.

The setting of parameters

－RS-232C Communication signal:	R0000.0	0: Communication OFF	1: Communication ON
－RSOPN Executing signal:	R0000.1	0: Not executed	1: Executing
－RSOPN Completion signal	R0000.2	0: Not completed	1: Completed
－RSRCV Executing signal:	R0000.3	0: Not executed	1: Executing
－RSRCV Completion signal:	R0000.4	0: Not completed	1: Completed
－RSCLS Executing signal:	R0000.5	0: Not executed	1: Executing
－RSCLS Completion signal:	R0000.6	0: Not completed	1: Completed
－Port open signal:	R0001.0	0: Not opened	1: Opened

(a) The parameters for RSOPN(SUB491)

- Channel No.:	1	// Using JD56A port
- Start address of RS-232C control data	D0000	
Contents of the table		
Reserved	D0000	Set 0
Baud rate	D0001	1: 1200 2: 2400 3: 4800 4: 9600 5: 19200
Stop bit	D0002	0: 1bit 1: 2bits
Reserved	D0003－D0005	Set 0
Receiving time outs	D0006	Receiving time out (Second)
Sending time outs	D0007	Sending time out (Second)
Terminate ASCII code	D0008	0: None 1: CR 2: LF
Reserved	D0009－D0015	Set 0
- Output address of channel handler:	R0010	Channel handler
- Output address of completion code:	R2000	Completion code

(b) The parameters for RSRCV(SUB492)

- Address of channel handler:	R0010	Channel handler
- Output address of received data:	R0100	Received data
- Data size to be received:	255	The data up to 255 bytes
－ Output address of received data size:	R0500	Received data size
- Output address of completion code:	R2001	Completion code

(c) The parameters for RSCLS(SUB494)

－ Address of channel handler:	R0010	Channel handler
- Output address of completion code:	R2002	Completion code

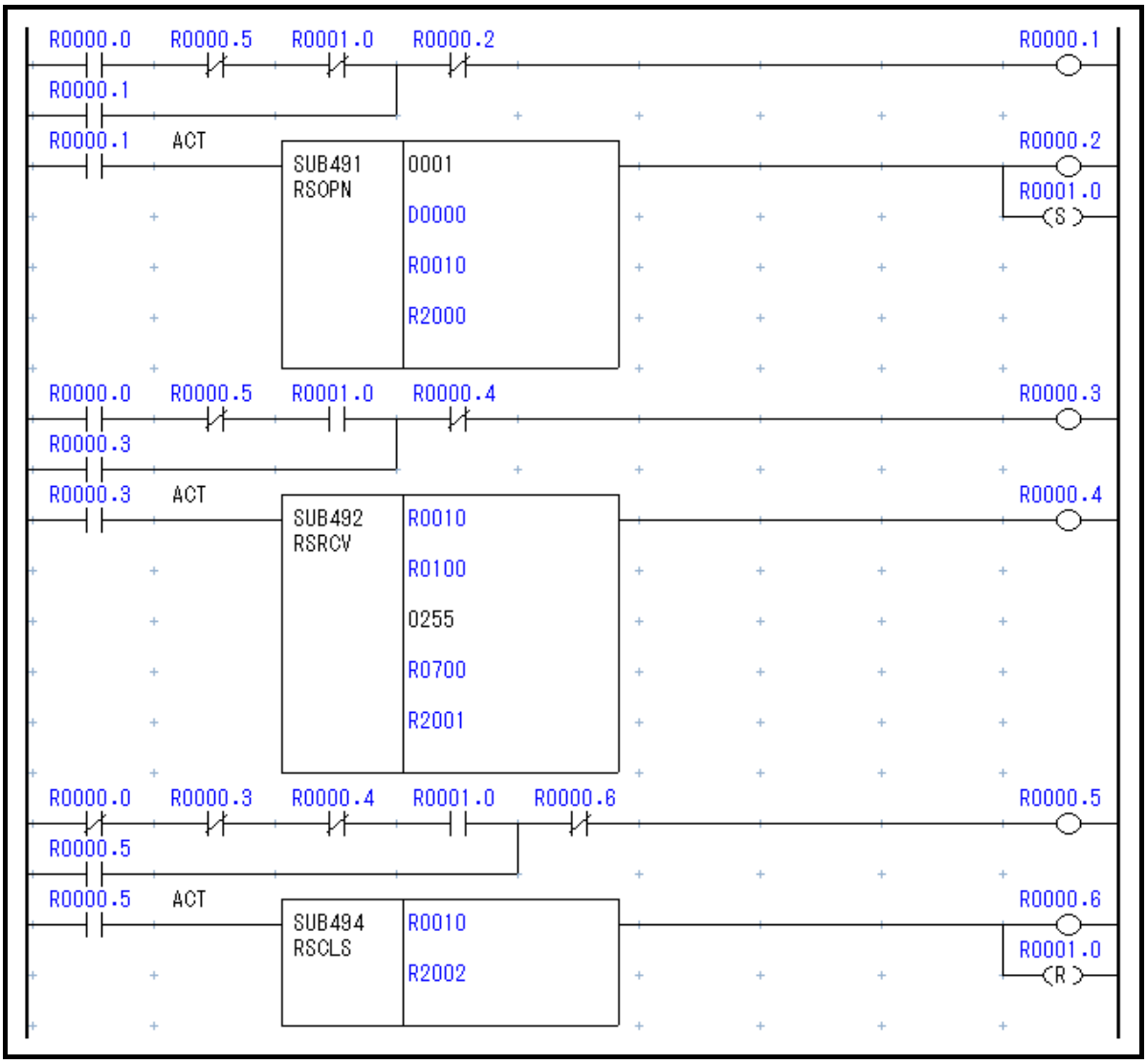


Fig. 5.4.6 (a) Example of Ladder diagram for RS-232C control

1. When the RS-232C communication signal is set to 1 (R0000.0=1), the RSOPN instruction is executed.
   a) When the process has been completed, the RSOPN completion signal becomes 1 (R0000.2=1) and ACT is set to OFF (R0000.1=0) in next ladder execution period. The port open signal is also set to 1 (R0001.0=1).
   b) In case that the port has been opened or the RSCLS is executing (R0000.5=1), the RSOPN is not executed.
2. Because the RS-232C communication signal is 1 (R0000.0=1) and the port open signal is 1 (R0001.0=1), the RSRCV instruction is executed.
   a) When the communication is completed, the RSRCV completion signal becomes 1 (R0000.4=1). ACT is set to OFF (R0000.3=0) in next ladder execution period.
   b) The data reception is repeated on each ladder scanning while the RS-232C communication signal is 1 (R0000.0=1).
   c) In case that the RSCLS is executing (R0000.5=1), the RSRCV is not executed.
3. When the RS-232C communication signal is set to 0 (R0000.0=0) and the RSRCV instruction is completed (R0000.3=0 and R0000.4=0), the RSCLS instruction is executed.

a) When the process has been completed, the RSCLS completion signal becomes 1 (R0000.6=1) and ACT is set to OFF (R0000.5=0) in next ladder execution period. The port open signal is also set to 0 (R0001.0=0).
b) To restart RS-232C communication, set RS-232C communication signal to 1 (R0000.0=1).

5.5 NOTE ON PROGRAMMING

Some functional instructions may cause the ladder program to take a long time to update or stop, or the PMC alarm “WN03 ABORT NC-WINDOW/EXIN”, “WN12 RSOPN/RSRCV/RSSND/RSCLS IS ABORTED” may occur, or the ladder program is not able to update or stop, if their ACT or RST condition remains on for no apparent reason.
To avoid such problems, when you code a ladder program using those functional instructions, you need to design the ladder structure based on a thorough understanding of the control conditions of the individual instructions you use.
Listed below are typical cases in which the ladder program will not stop.

- A low-speed window function is used for a WINDR or WINDW functional instruction, and its ACT condition remains on after the end of command.
- In an EXIN instruction, its ACT condition remains on after the end of command.
- In an AXCTL instruction, its ACT condition or RST condition remains on after the end of command.
- In an RSOPN / RSRCV / RSSND / RSCLS instruction, its ACT condition remains on after the end of command.
- The same processing is repeated by JMPB instruction and JMPC instruction.

It takes time for the following operations or the operation cannot be completed by the above-mentioned.

- Stopping the ladder program using a soft key on the screen
- Reading a new ladder program from a memory card or other medium, by using the data input and output screen
- Updating the ladder program with changes made using the ladder diagram edit screen

If any of the above phenomena occurs, the functional instruction causing the problem needs to be fixed. Check the functional instructions mentioned above to see whether there is any ACT or RST condition remaining on, and correct the ladder program according to the following procedure.

1. Put the machine in safe condition and turn off the power of the CNC.
2. Turn on the power of the CNC while holding down the "CAN" and "Z" keys simultaneously, to restart the NC with the ladder program halted.
3. In the ladder diagram edit screen, redesign the logic associated with the problematic functional instruction. When done, set the ACT or RST condition to off. If the same operation is repeating because of an inadequate JMPB or JMPC instruction, review the jump condition and, if necessary, change the ladder structure.
4. Write the resulting logic to flash ROM using the I/O screen.
5. Run the ladder program.

If the ladder program does not stop or cannot be changed even after you make the correction, there may be other functional instructions that have the same condition settings. Check for other functional instructions having the same condition settings, besides the one you have corrected, and repeat the above procedure to correct them all.

6 WINDOW FUNCTIONS

6.1 LOW-SPEED RESPONSE AND HIGH-SPEED RESPONSE

There are two types of window function - one executed at high speed and the other executed at low speed.

TYPE	Number of scans to be executed until the window instruction is completed
LOW	TWO SCAN TIMES OR MORE (Depends on the CNC processing priority and operation status.)
HIGH	1 SCAN TIME

When using the low-speed response window function, set ACT to 0 immediately after the data transfer end data (W1) is set to 1 for the window instruction. For details, see "CAUTION" below.

⚠ CAUTION

1 The window instruction of a low-speed response is controlled exclusively with the other window instructions of low-speed response.
Therefore, when the data is read or written continuously, it is necessary to clear ACT of the functional instruction to 0 once when the completion information (W1) become 1.
If you leave a window instruction of a low-speed response with W1=1 and ACT=1, other window instructions of low-speed response do not work.
The window instruction of a high-speed response is not exclusively controlled like a low-speed response. Therefore, when the data is read or written continuously, you need not make ACT=0.

2 The completion of the other window instruction of low-speed response may be delayed by the execution of application using FOCAS2 functions and C language executor application. If some window instruction of a low speed response is executed continuously with a high frequency, the completion of the other window instruction of low-speed response will be delayed, or may not be done. You should take care that the execution of application using FOCAS2 functions and C language executor application may be delayed. The window instruction of a low-speed response should be executed with the lowest required frequency.

3 Because window functions of low-speed response may take long time to complete, you should make sure that the conditions to start some actions, which need to wait for the completion of a window function, include the completion of the window instruction.

7 PMC ALARM MESSAGES AND ACTIONS TO TAKE

7.1 ALARM MESSAGE LIST

7.1.1 Messages That May Be Displayed on the PMC Alarm Screen

Add the following alarm

Alarm number	Faulty location/corrective action	Contents
WN12 RSOPN/RSRCV/RSSND/RSCLS IS ABORTED	Modify the ladder program and turn on the power of CNC. Refer to section “4.17” in the PMC programming manual for detail.	The ladder program was stopped while RS-232C control communication is executing. This alarm may cause the RSOPN, RSRCV, RSSND and RSCLS instruction to malfunction.