www.cimon.com

CIMON
Total Solution for Industrial Automation

CIMON-PLC

INSTRUCTION USER MANUAL

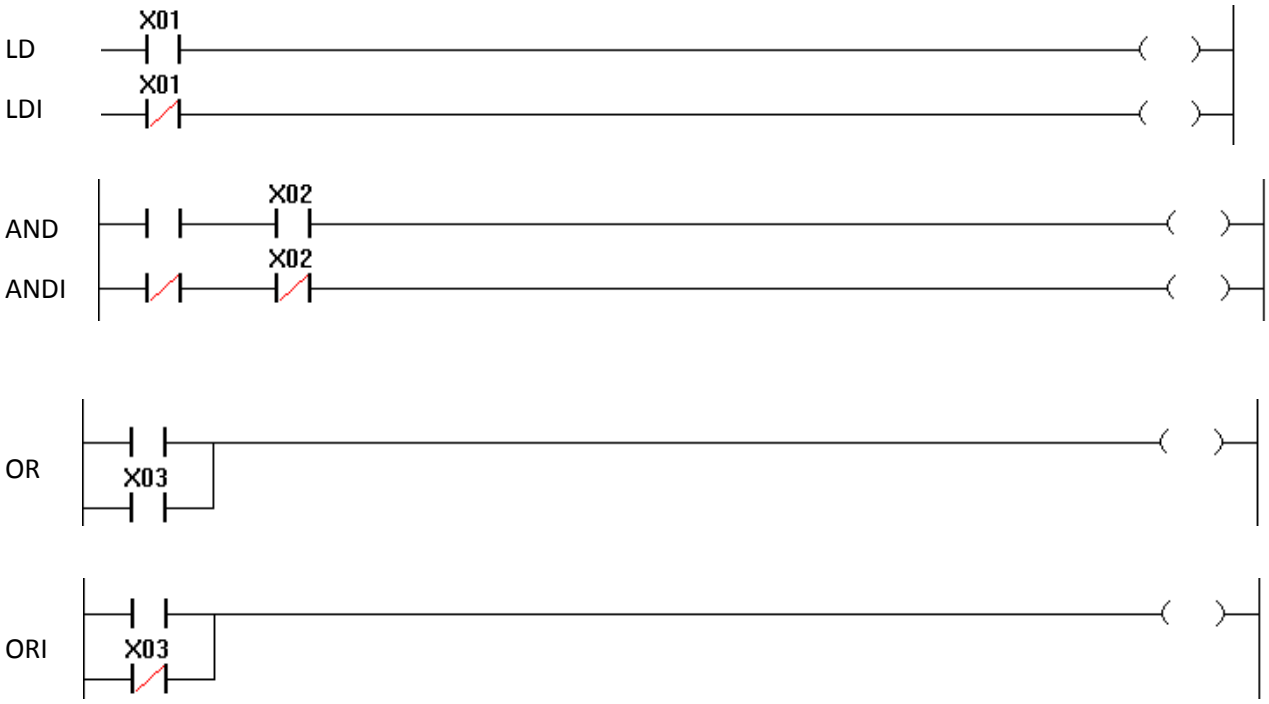
Table of Contents

1. BASIC INSTRUCTION

1.1. Logical Instruction

1.1.1. Start Operation, Series Connection, Parallel Connection: LD, LDI, AND, ANDI, OR, ORI

Instruction		Device address													No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
LD LDI AND ANDI OR ORI	S	o	o	o	o	o	o	o	o	o	-	-	-	-	1	-	-	-

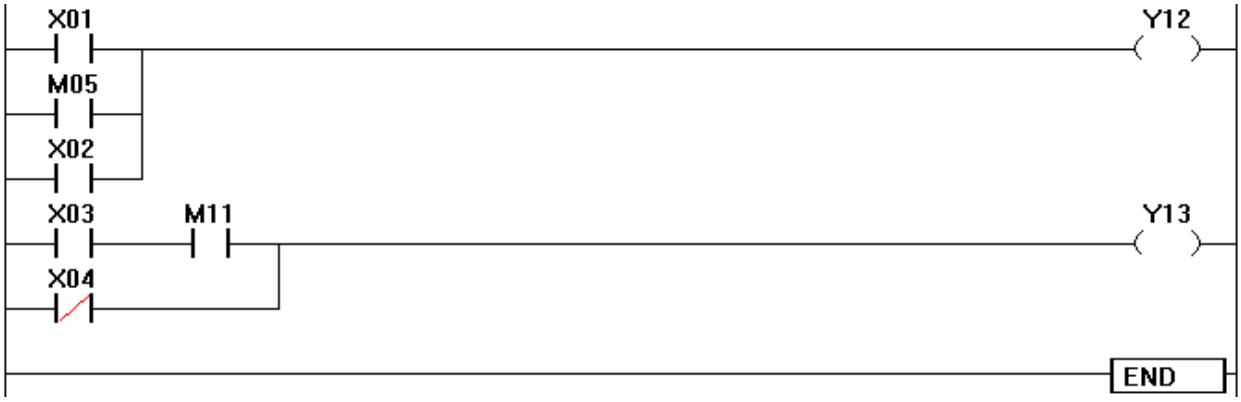


1) LD (Load), LDI (Load Inverse)

Function:

LD is used to start the operation of Contact A while LDI is used to start the operation of Contact B.

The information (ON/OFF) on the assigned device is read and taken as the result of their operation.



List Mode

Steps	Instruction	Device
0	LD	X0001
1	OR	M0005
2	OR	X0002
3	OUT	Y0012
4	LD	X0003
5	AND	M0011
6	ORI	X0004
7	OUT	Y0013
8	END	

2) AND, ANDI (AND Inverse)

Function:

AND is an instruction to connect in series with Contact A while ANDI is an instruction to connect in series with Contact B. The information (ON/OFF) on the assigned device is read, and the result of the AND operation is taken as the result of their operation.

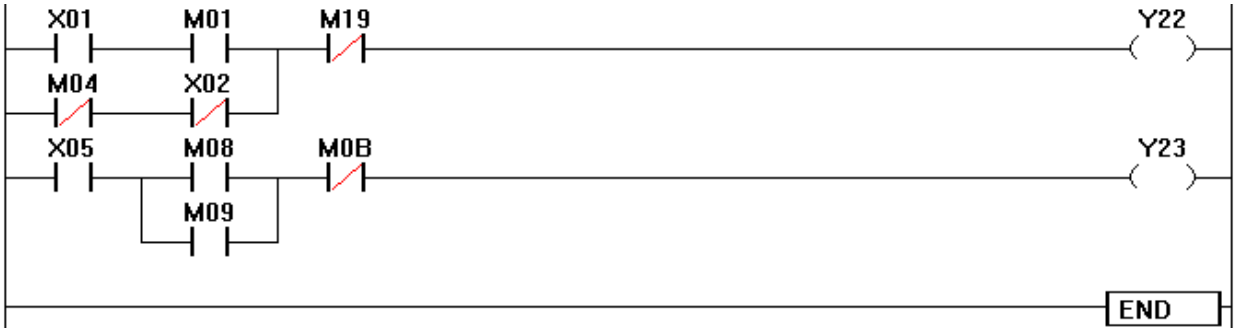
List Mode

Steps	Instruction	Device
0	LD	X0001
1	OUT	Y0021
2	AND	X0002
3	OUT	Y0022
4	ANDI	X0003
5	OUT	Y0023
6	END	

3) OR, ORI (OR Inverse)

Function:

OR is an instruction to connect in parallel with Contact A while ORI is an instruction to connect in parallel with Contact B. The information (ON/OFF) on the assigned device is read, and the result of the OR operation is taken as the result of their operation.



List Mode

Steps	Instruction	Device
0	LD	X0001
1	AND	M0001
2	LDI	M0004
3	ANDI	X0002
4	ORB	
5	ANDI	M0019
6	OUT	Y0022
7	LD	X0005
8	LD	M0008
10	OR	M0009
11	ANB	
12	ANDI	M000B
13	OUT	Y0023
14	END	

1.1.2. Start Operation, Series Connection, Parallel Connection for Positive-Negative Transition Contact: LDP, LDF, ANDP, ANDF, ORP, ORF

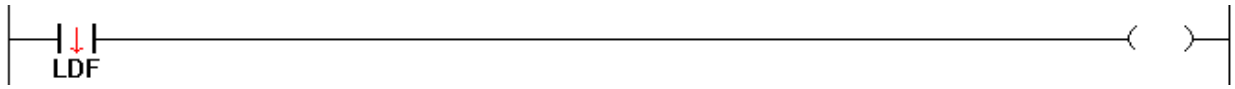
Instruction		Device address												No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
LDP LDF ANDP ANDF ORP ORF	S	o	o	o	o	o	o	o	o	-	-	-	-	-	1	-	-	-

1) LDP (Load Pulse)

• This function is used to read the positive transition contact of a circuit.
• If an input condition is turned ON from OFF, this turns the device ON for one scan.

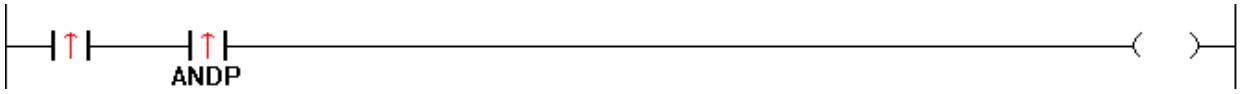
2) LDF (Load Falling Pulse)

- This function is used to read the negative transition contact of a circuit.
- If an input condition is turned OFF from ON, this turns the device ON for one scan.



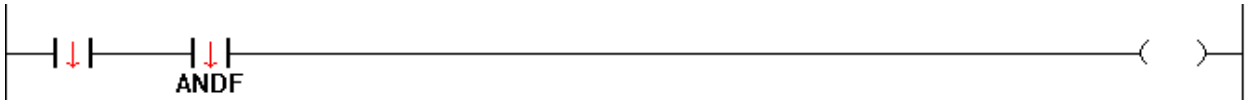
3) ANDP (AND Pulse)

- This function is used to read the positive transition contact in series connection.
- If an input condition is turned ON from OFF, this turns the device ON for one scan.
- If the left connection is turned ON, the right connection turns ON for one scan.



4) ANDF (AND Falling Pulse)

- This function is used to read the negative transition contact in series connection.
- If an input condition is turned OFF from ON, this turns the device ON for one scan.
- If the left connection is turned ON, the right connection turns ON for one scan.

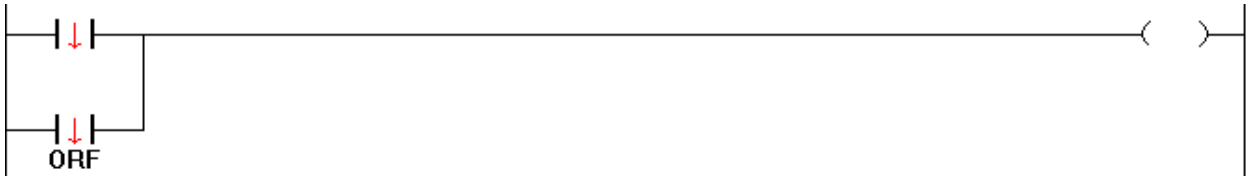


5) ORP (OR Pulse)

- This function is used to read the positive transition contact in parallel connection.
- If an input condition is turned ON from OFF, this turns the device ON for one scan.
- If the below connection is turned ON, the above connection turns ON for one scan.

6) ORF (OR Falling Pulse)

• This function is used to read the negative transition contact in parallel connection.
• If an input condition is turned OFF from ON, this turns the device ON for one scan.
• If the below connection is turned ON, the above connection turns ON for one scan.

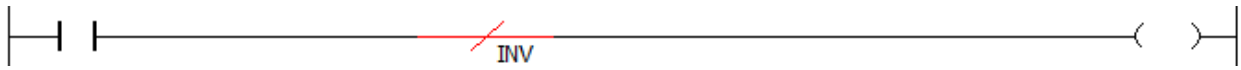


1.1.3. Invert the Result of the Operation: INV (Inverse)

Instruction	Device address													No. of Steps	Flag		
	M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
INV	-	-	-	-	-	-	-	-	-	-	-	-	-	1	-	-	-

Function:

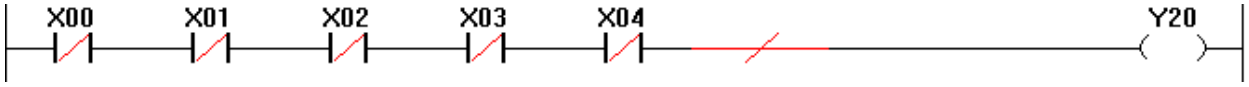
If you use the INV instruction, Contact “A” circuit is inverted to Contact “B” circuit and Contact “B” circuit is inverted to Contact “A” circuit for the circuit on the left of the INV instruction. Likewise, series connection circuit is inverted to parallel connection circuit while parallel connection circuit is inverted to series connection circuit.



Example)

Program A and B are examples that output the same result.

- Program A



- Program B

1.1.4. Data Device (D) Bit Control: LDBT, LDBTI, ANDBT, ANDBTI, ORBT, ORBTI

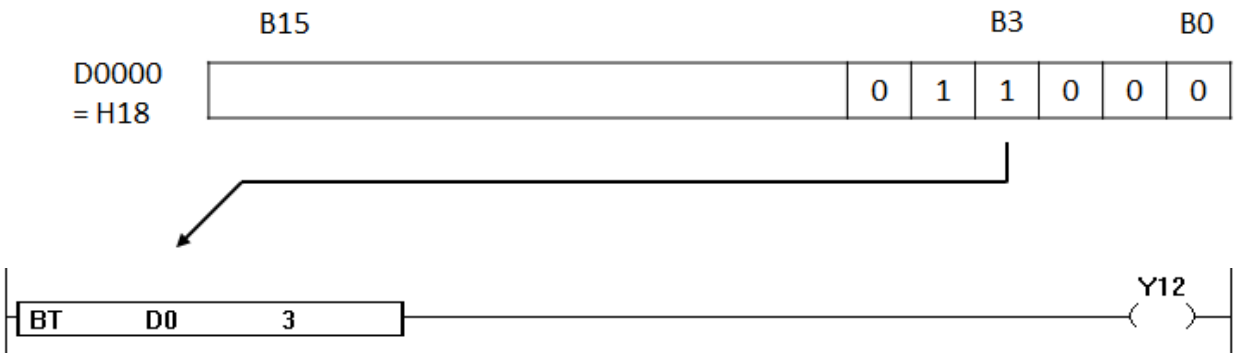
Instruction		Device address												No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
LDBT(I) ANDBT(I) ORBT(I)	S1	-	-	-	-	-	-	-	-	-	o	o	o	-	3	-	-	-
	S2	o	o	o	o	o	o	o	o	-	o	o	o	o				

1) LDBT (Load Bit), LDBTI (Load Bit Inverse)

Function:

These instructions are used to read the bit data assigned to S2 among the word addresses in Device D assigned to S1, starting the operation for Contact A (LDBT) and the one for Contact B (LDBTI). The information (On/Off) on the assigned device are received and taken as the result of their operation.

Example)



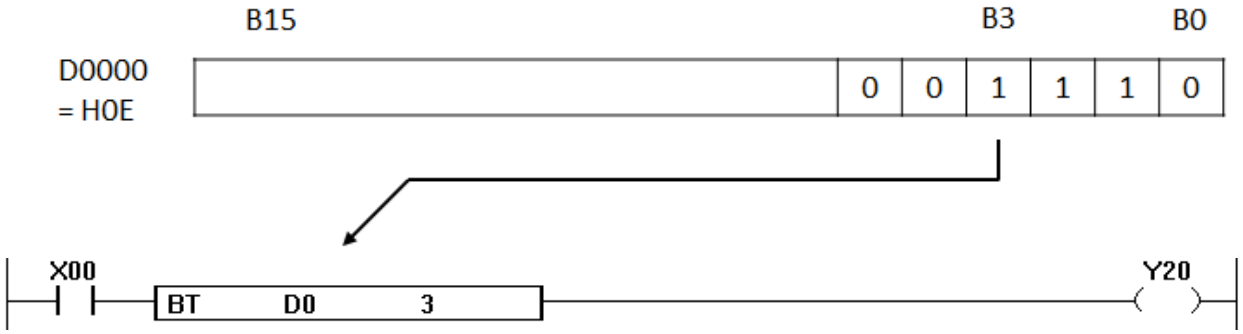
If the 3rd bit of D0000 is turned on, Output Y12 is turned on.

2) ANDBT (AND Bit), ANDBTI (AND Bit Inverse)

Function:

These instructions are used to read the bit data assigned to S2 among the word addresses in Device D assigned to S1, connecting in series with Contact A (ANDBT) and Contact B (ANDBTI). The information (On/Off) on the assigned device are received and taken as the result of their operation.

Example)



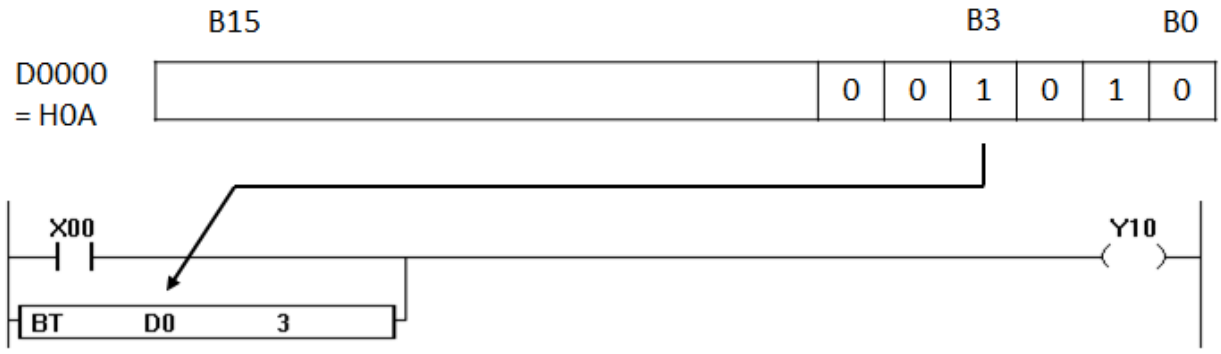
Turning on X00 and the 3rd bit of D00 turns Y20 on.

3) ORBT (OR Bit), ORBTI (OR Bit Inverse)

Function:

These instructions are used to read the bit data assigned to S2 among the word addresses in Device D assigned to S1, connecting in parallel with Contact A (ORBT) and Contact B (ORBTI). The information (On/Off) on the assigned device are received and taken as the result of their operation.

Example)



Either turning on X00 or the 3rd bit of D00 turns Y10 on.

1.2. Connection Instruction

1.2.1. Series Connection and Parallel Connection of Blocks: ANB, ORB

ANB and ORB instructions are not contact symbols but connection symbols that connect blocks either in series or parallel.

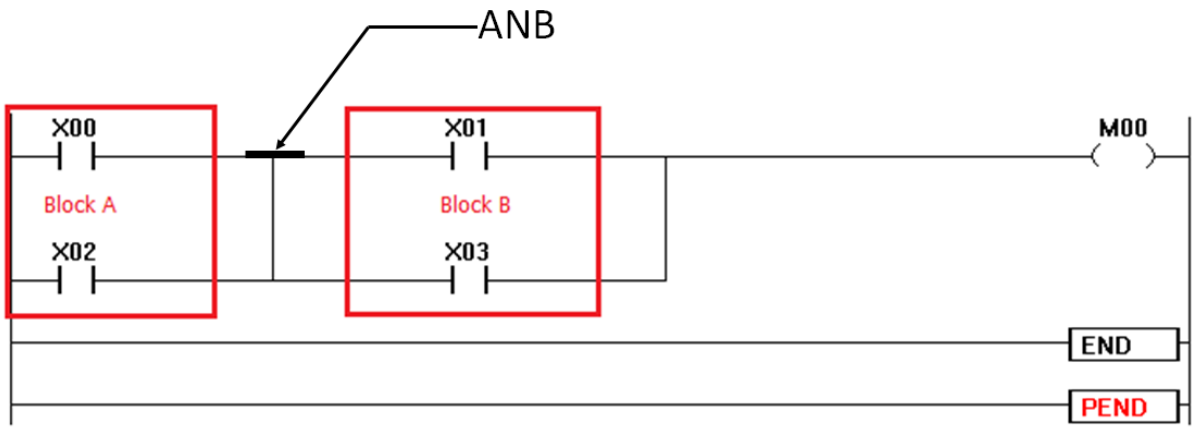
Instruction	Device address													No. of Steps	Flag		
	M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
ANB ORB	-	-	-	-	-	-	-	-	-	-	-	-	-	1	-	-	-

1) ANB (AND Block)

Function:

- Connect 'Block A' and 'Block B' in series and take the result to the output.
- ANB instruction is not a contact symbol but a connection symbol.
- A maximum of 15 instructions (16 blocks) are available in case of writing the ANB continuously.

Example)

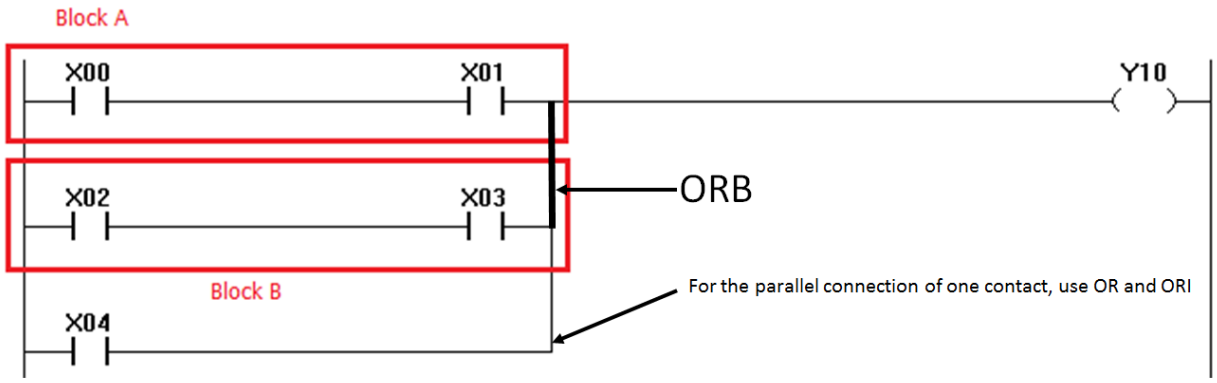


2) ORB (OR Block)

Function:

- Connect 'Block A' and 'Block B' in parallel and take the result to the output.
- ORB instruction is used to connect the circuit block over two contacts in parallel. On the other hand, to connect the circuit block of one contact, use the OR and the ORI. ORB is not necessary in this case.
- ORB instruction is not a contact symbol but a connection symbol.
- A maximum of 15 instructions (16 blocks) are available in case of writing the ORB continuously.

Example)



1.2.2. Multiple Branch: MPS, MRD, MPP

Multiple branch instructions are used to connect output coils to the left hand side of a contact. Without these instructions, connections can only be made to the right hand side of the last contact.

For every MPS instruction, there must be a corresponding MPP instruction.
The last contact or coil circuit must connect to an MPP instruction.

Instruction		Device address													No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
MPS MRD MPP	S	-	-	-	-	-	-	-	-	-	-	-	-	-	1	-	-	-

1) MPS (Multiple Point Store)

Function:

This is used to start branching in a ladder. MPS stores the connection point of the ladder circuit so that further coil branches can recall the value later.

2) MRD (Multiple Read)

Function:

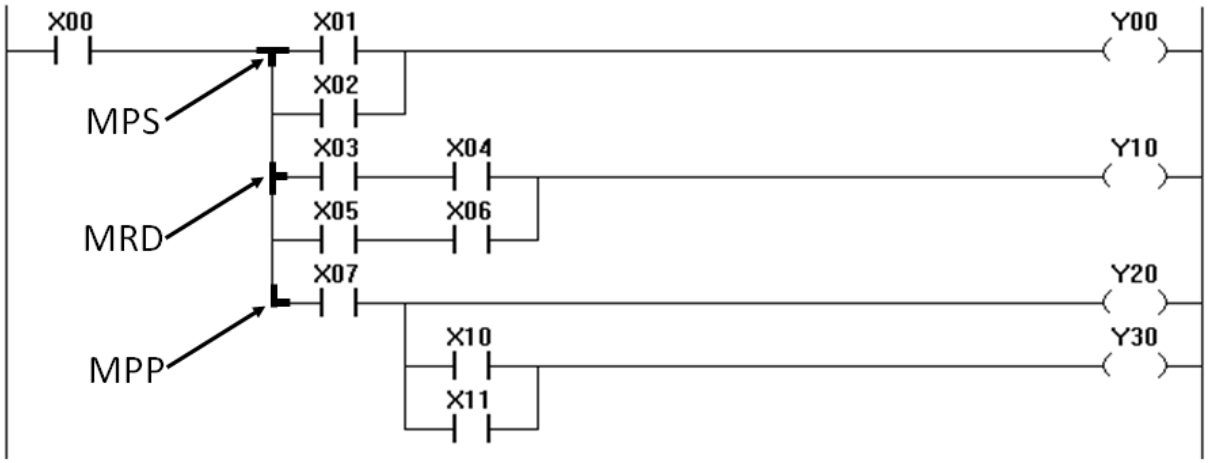
This is used to relay branching in a ladder. MRD recalls or reads the previously stored connection point data and forces the next contact to connect to it.

3) MPP (Multiple Pop)

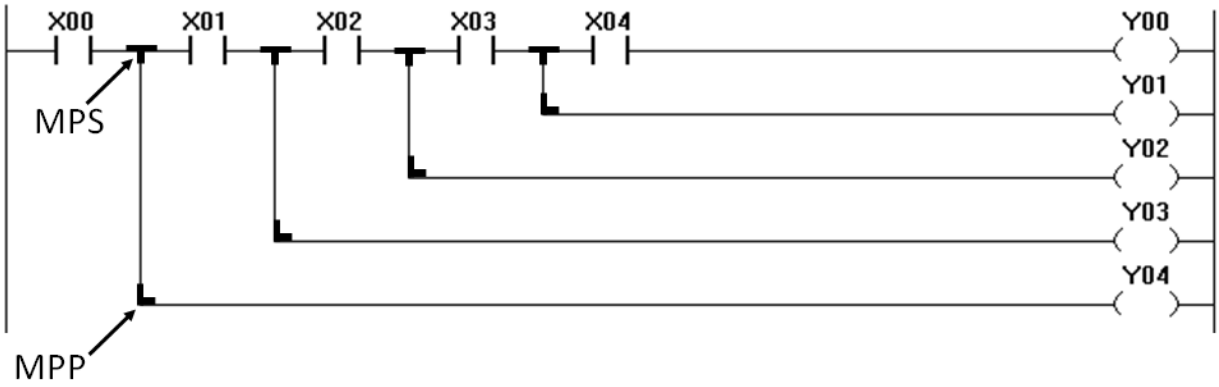
Function:

This is used to finish branching in a ladder. MPP pops (recalls and removes) the stored connection point. First, it connects the next contact, and then it removes the point from the temporary storage area.

Examples)



Steps	Instruction	Device	Steps	Instruction	Device
0	LD	X00	11	ORB	
1	**MPS**		12	ANB	
2	LD	X01	13	OUT	Y10
3	OR	X02	14	**MPP**	
4	ANB		15	AND	X07
5	OUT	Y00	16	OUT	Y20
6	**MRD**		17	LD	X10
7	LD	X03	18	OR	X11
8	AND	X04	19	ANB	
9	LD	X05	20	OUT	Y30
10	AND	X06			



Steps	Instruction	Device	Steps	Instruction	Device
0	LD	X00	9	OUT	Y00
1	**MPS**		10	**MPP**	
2	AND	X01	11	OUT	Y01
3	**MPS**		12	**MPP**	
4	AND	X02	13	OUT	Y02
5	**MPS**		14	**MPP**	
6	AND	X03	15	OUT	Y03
7	**MPS**		16	**MPP**	
8	AND	X04	17	OUT	Y04

1.3. Output Instruction

1.3.1. Bit Device, Timer, Counter: OUT

OUT instruction is the final logical operation type coil drive and is used to output the final result of the operation to the assigned device. It is impossible to use the OUT instruction to drive 'X' type input devices. However, it is possible to connect multiple OUT instructions in parallel.

Instruction		Device address													No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
OUT	S	o	-	o	o	o	-	-	-	o	-	-	-	-	1	-	-	-

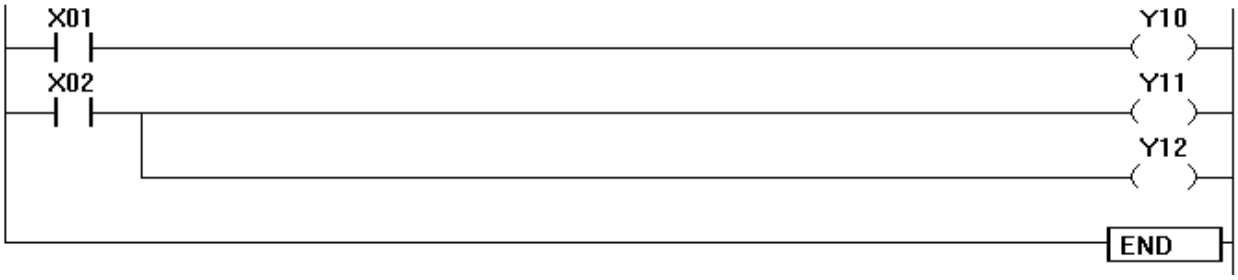
Function:

Outputs the result of the operation prior to the OUT instruction to the assigned device.

Result of Operation	OUT Instruction		
	Coil	Contact	
		Contact A	Contact B
OFF	OFF	OFF	ON
ON	ON	ON	OFF

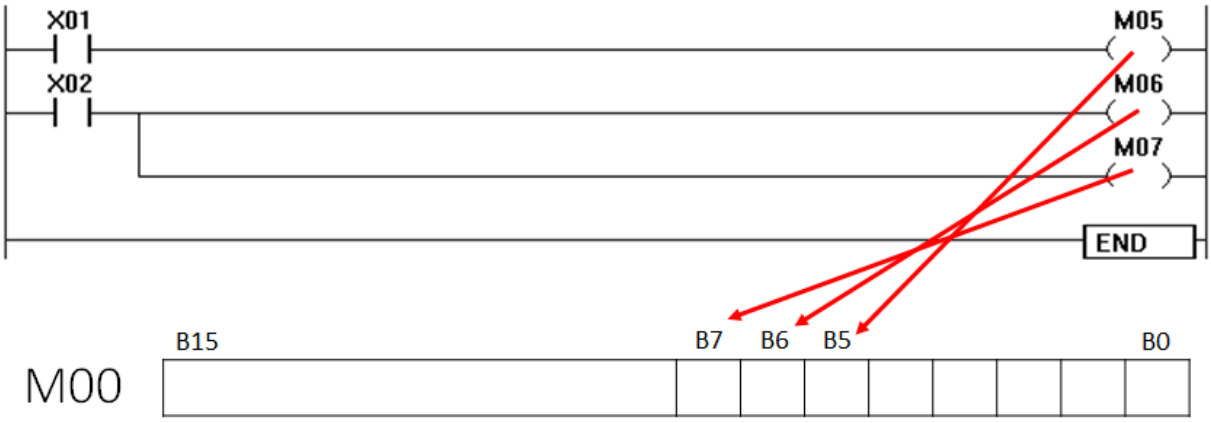
Example)

1) A program that outputs with an output unit



Steps	Instruction	Device
0	LD	X0001
1	OUT	Y0010
2	LD	X0002
3	OUT	Y0011
4	OUT	Y0012
5	END	

2) A program that writes a word device made of a bit instruction



Steps	Instruction	Device
0	LD	X0001
1	OUT	M0005
2	LD	X0002
3	OUT	M0006
4	OUT	M0007
5	END	

1.3.2. Set and Reset Bit Device: SET, RST

Function:

- SET instruction sets a bit device permanently ON. The turned-on device remains the turned-on status even after SET instruction is turned OFF.
- RST instruction resets a bit device permanently OFF. The turned-on device by SET instruction can be turned OFF and reset by RST instruction.

Instruction		Device address												No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
SET	S	o	-	o	o	o	-	-	-	o	-	-	-	-	1	-	-	-
RST		o	-	o	o	o	-	o	o	o	-	-	-	-				

Example)

-Turning ON X00 causes Y00 to turn ON. Y00 remains ON even after X00 turns OFF.
-Turning ON X01 causes Y00 to turn OFF. Y00 remains OFF even after X01 turns OFF.

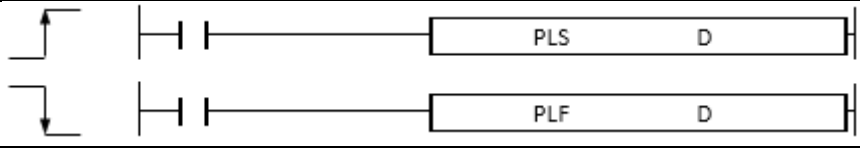
-SET and RST instructions can be used for the same device as many times as necessary. However, the last instruction activated determines the current status.

Steps	Instruction	Device
0	LD	X00
1	**SET**	Y00
2	LD	X01
3	**RST**	Y00
4	LD	X02
5	**SET**	M00
6	LD	X03
7	**RST**	M00
8	LD	X04
9	**SET**	L00
10	LD	X05
11	**RST**	L00

1.3.3. Rising Edge and Falling Edge Pulse Output for a Scan: PLS, PLF

When a PLS instruction is executed, object devices operate for one operation cycle after the drive input signal has turned ON. Likewise, when a PLF instruction is executed, object devices operate for one operation cycle after the drive input signal has turned OFF.

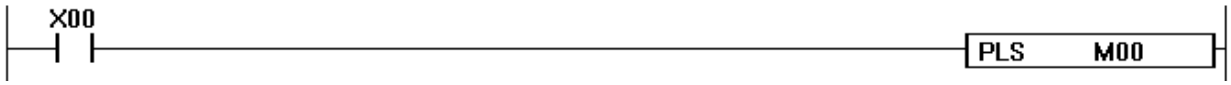
Instruction		Device address													No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
PLS PLF	S	o	-	o	o	o	-	-	-	o	-	-	-	-	1	-	-	-



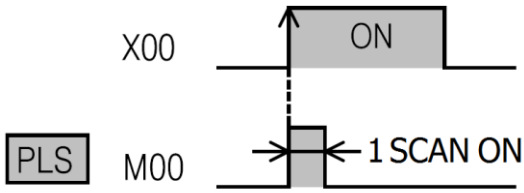
D	Source

1) PLS (Pulse – Rising edge pulse)

Function:
Turns the assigned device ON for one scan when the PLS instruction is turned ON from OFF.



For example, when X00 is turned ON from OFF by PLS instruction, the device “M00” is turned ON only for one scan.

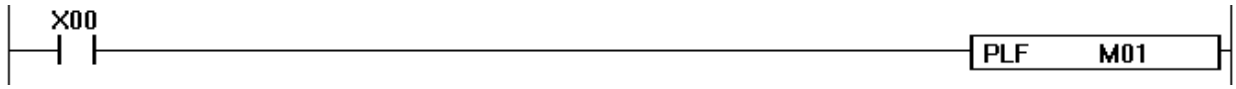


When the PLS instruction is executed on the device specified as Latch Relay (L) and the power is turned OFF and ON again, the device will output the previous value.

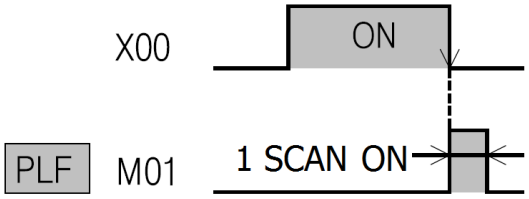
Though PLC is switched from RUN to STOP and STOP to RUN after the PLS instruction, it will not activate the PLS instruction.

2) PLF (Pulse Falling – Falling edge pulse)

Function:
Turns the assigned device ON for one scan when the PLF instruction is turned OFF from ON.



For example, when X00 is turned OFF from ON by PLF instruction, the device “M01” is turned ON only for one scan.



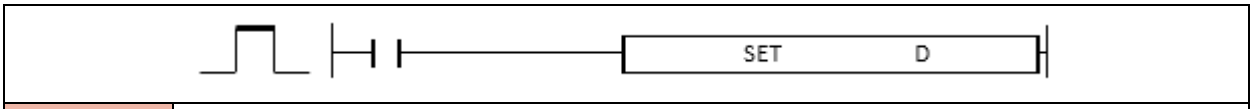
Though PLC is switched from RUN to STOP and STOP to RUN after the PLF instruction, it will not activate the PLF instruction.

1.4. Step Control Instruction

1.4.1. Step Control: SET Sxx.xx

The step control (SET instruction) outputs the value when its previous step is ON and its own condition contact is ON.

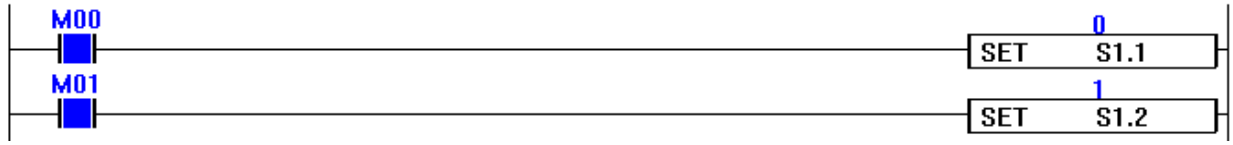
Instruction		Device address													No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
SET	D	-	-	-	-	-	-	-	-	o	-	-	-	-	1	-	-	-

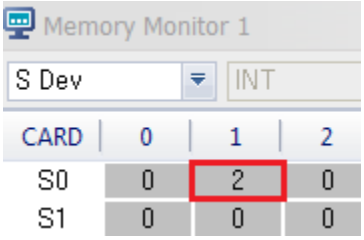


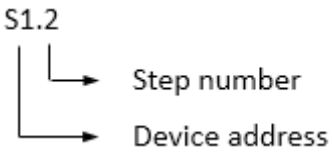
D	Source

*S device: This device is specifically to be used for step control relay, including step control (SET instruction) or last-in and first-out control (OUT instruction).

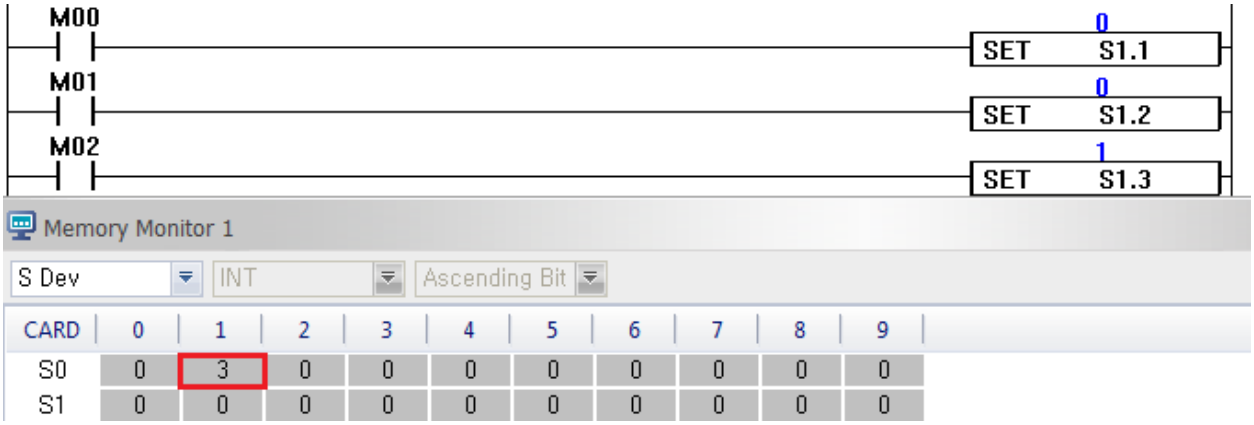
Function:
When the previous step number (S1.1) is ON, the current step number (S1.2) will be ON.



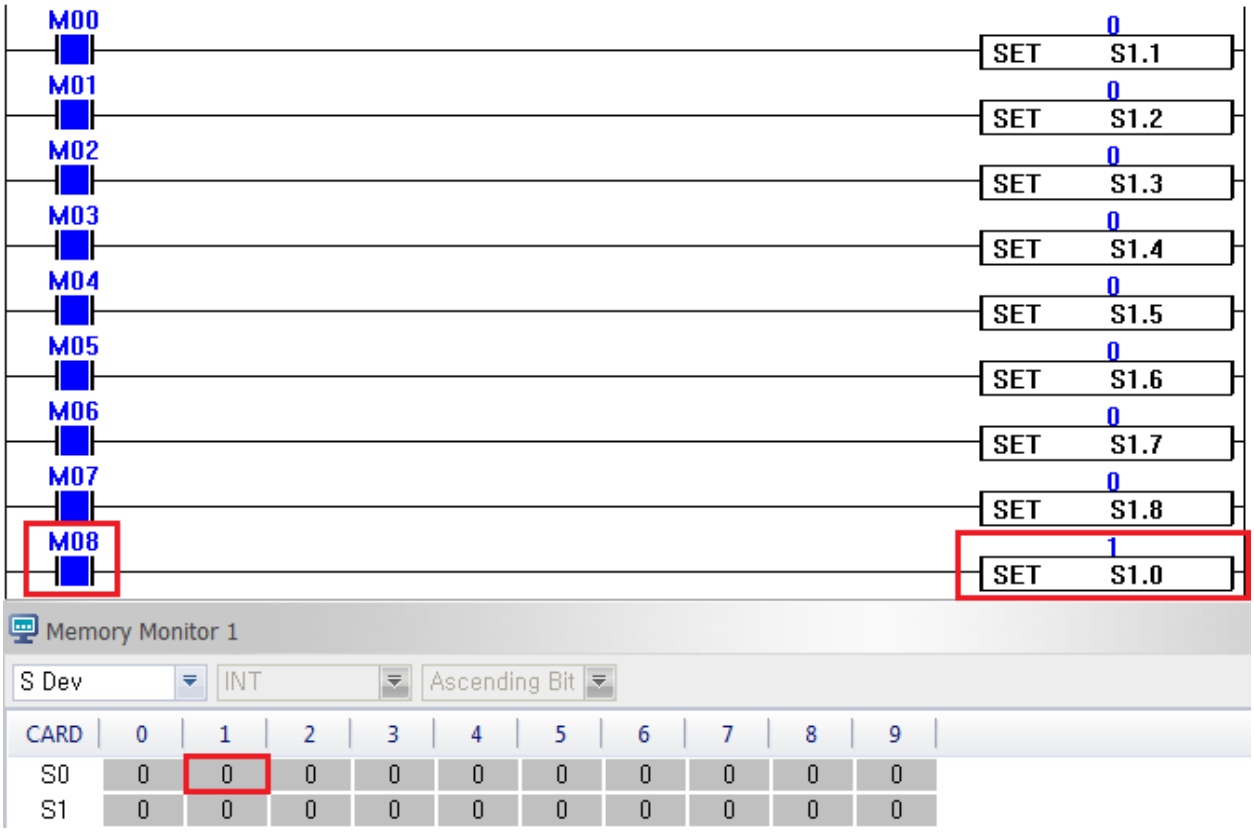




If the current step number (S1.3) is ON, it will maintain the value and keep the ON status even after the input contact (M02) is OFF.



The SET instruction gets cleared by turning ON the input contact for Sxx.00.
Notice that the value of "S1" becomes 0 by turning ON the input contact (M08) for SET S1.0.



1.4.2. Last-In and First-Out Control: OUT Sxx.xx

The Last-In and First-Out control (OUT instruction) outputs the value entered last with first priority.

Instruction		Device address													No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
OUT	S	-	-	-	-	-	-	-	-	o	-	-	-	-	1	-	-	-

S	This device is specifically to be used for step control relay such as step control (SET instruction) or last-in and first-out control (OUT instruction).

Function:

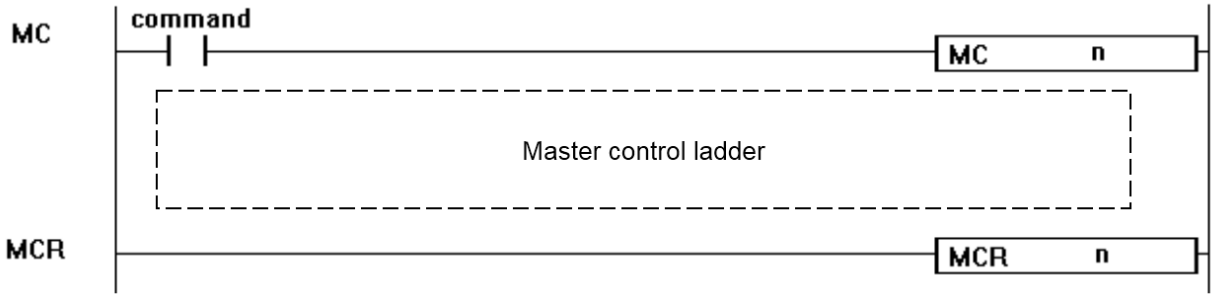
- Even if multiple input contacts are ON, only one step number will be ON.
- Among the multiple turned-on input contacts, the last-entered input contact gets to output first.
- If the current step number is ON, it will maintain the value and keep the ON status even after the input contact is OFF.
- The OUT instruction gets cleared by turning ON the input contact for Sxx.00.

Example)

1.5. Master Control Instruction

1.5.1. Set or Reset Master Control: MC, MCR

Instruction		Device address													No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
MC, MCR	S	-	-	-	-	-	-	-	-	-	-	-	-	o	1	-	-	-



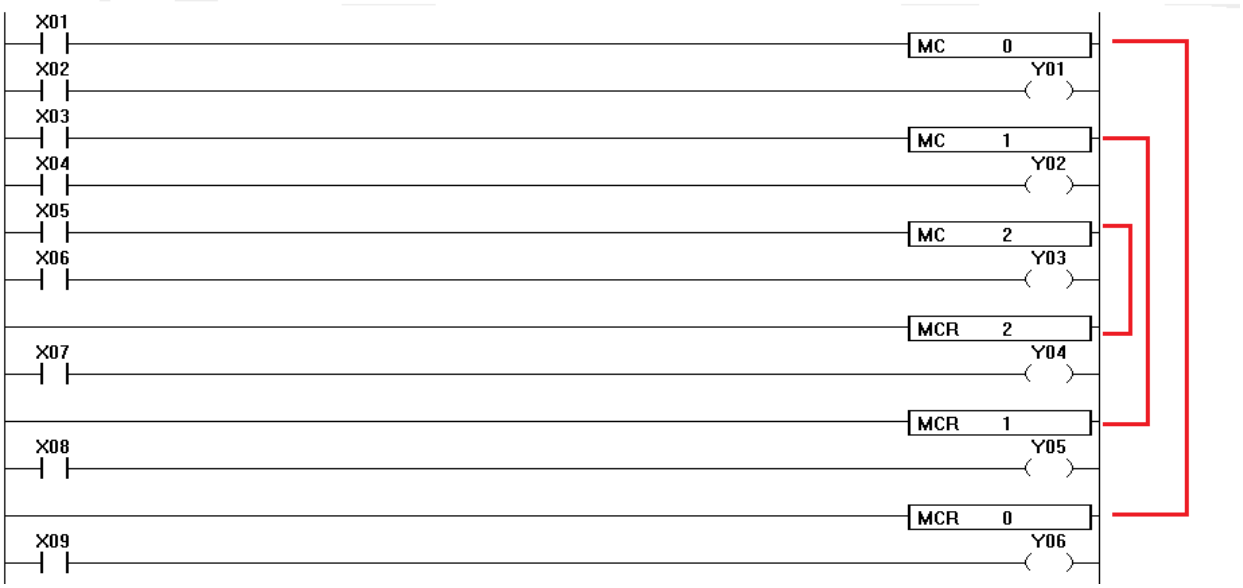
n	Nesting (0 – 7)

Functions:

- If the input condition of the MC is turned ON, instructions within the boundary from MC (n) to MCR (n) are executed. If the input condition is turned OFF, no instructions will be operated.
- Nesting is available up to 8 pairs (n0 – n7). In case of nesting, the MC is operated from a lower nesting number while the MCR is executed from a higher nesting number.
- The MCR instruction is used to terminate and mark the end (reset) of the Master Control instruction. MCR terminates from the specified nesting (n) number and thereafter.
- If the MC-operated block with a higher priority is terminated, the MC-operated blocks with lower priorities also get terminated.

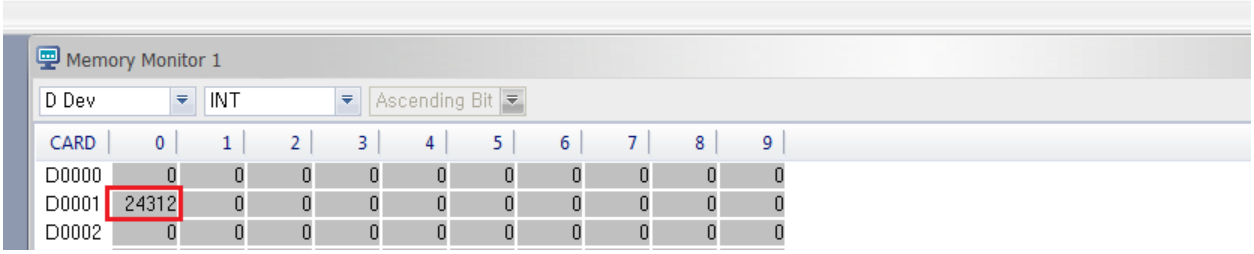
Example 1)

- If X01 is turned ON, the MC is set. If it is turned OFF, the MC is reset.
- X01 has to be ON in order to execute MC 0.
- In order to execute MC 1, MC 0 has to be operated first and then X03 has to be ON.
- In order to execute MC 2, MC 0 has to be operated first, then MC 1 the second, and X05 the last.



Example 2)

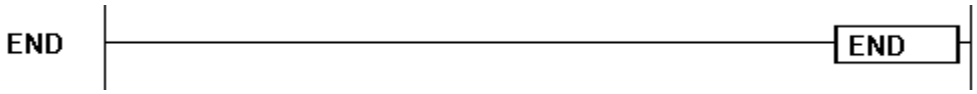
- Turning on the input condition (M01) of MC 0 allows D10 to increase as the time passes by.
- On the other hand, because the input condition (M02) of MC 1 is turned OFF, D11 will not increase its value. Notice how D11 stays '0.'



1.6. End Instruction

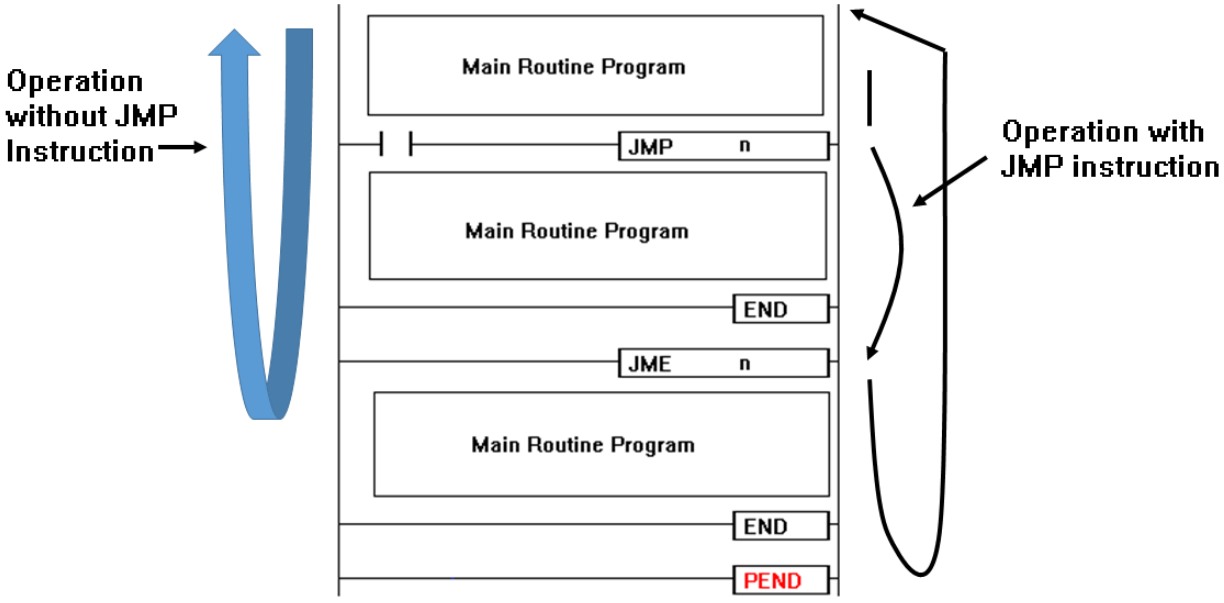
1.6.1. End Main Routine Program: END

Instruction		Device address													No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
END	S	-	-	-	-	-	-	-	-	-	-	-	-	-	1	-	-	-



Function:

- Used to end main routine program.
- When the END instruction is enabled, PLC goes back to 0 Step after operating the execution of the PEND instruction (Timer, Counter, Check, and etc.) and execute the operation again.



Example - A program using the JMP instruction)

• When M01 is ON, it will jump to Step 8 and execute the operation after the JME 1 step.
• Notice how Y30 and Y31 are OFF due to the JMP instruction even though M10 and M11 are ON.

Memory Monitor 1

Y Dev | INT | Ascending Bit

CARD	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	DEC	HEX
Y000	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	H0000
Y001	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	H0000
Y002	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	H0001
Y003	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	H0000
Y004	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	H0000
Y005	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	H0000

- Y30 and Y31 will be ON when M01 gets OFF and deactivates the JMP instruction.
- Notice how Y30 and Y31 are ON and have the value of 1 in the memory monitor.

Memory Monitor 1

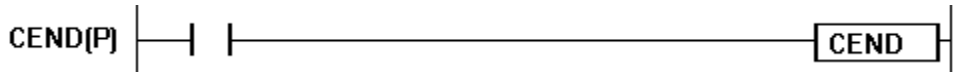
Y Dev | INT | Ascending Bit

CARD	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
Y000	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Y001	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Y002	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Y003	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Y004	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Y005	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Y0030

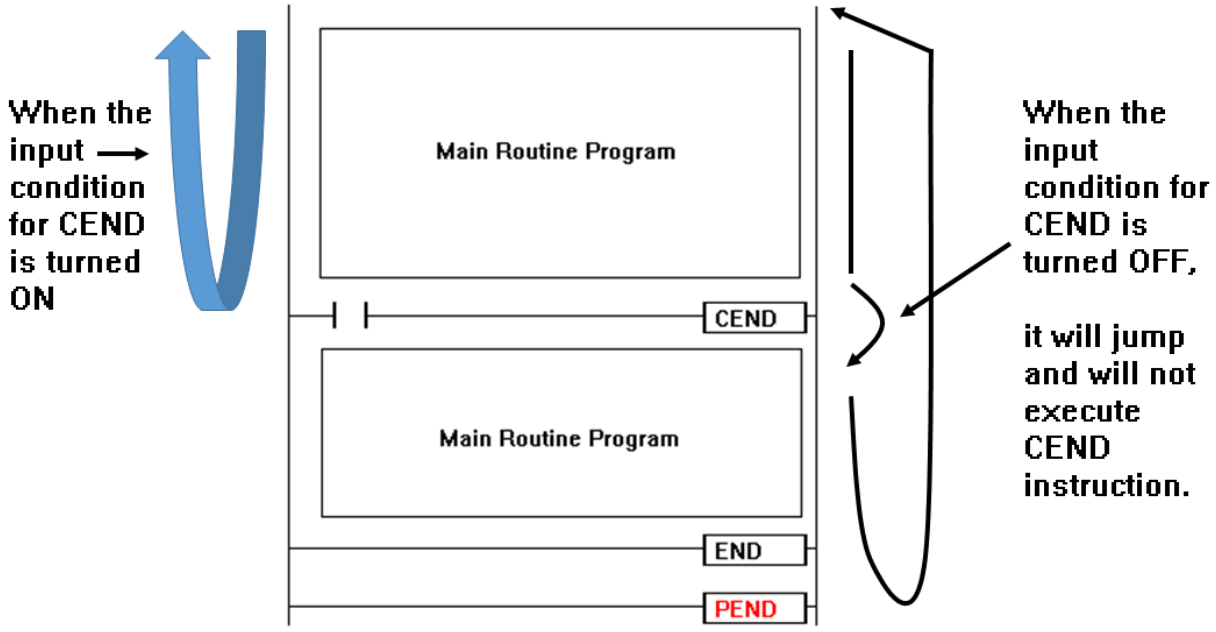
1.6.2. End Main Routine Program Conditionally: CEND, CENDP

Instruction	Device address													No. of Steps	Flag		
	M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
CEND (P)	-	-	-	-	-	-	-	-	-	-	-	-	-	1	-	-	-



Function:

- If the command execution condition is turned ON, the CEND(P) instruction will end the main routine program.
- The difference with END instruction is that CEND (P) must have a condition to get activated.
- At the end of the Main Routine Program must be an END instruction.



Example)

- When the input condition for CEND (M05) is turned ON, a main routine program will end.
- Notice how Y31 is OFF due to the CEND instruction, which has ended the main routine program, even though M11 is turned ON.

Memory Monitor 1

Y Dev | INT | Ascending Bit

CARD	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	DEC	HEX
Y000	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	H0000
Y001	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	H0000
Y002	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3	H0003
Y003	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	H0000
Y004	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	H0000

Y0021

- When the input condition for CEND (M05) is turned OFF, it will jump and will not execute CEND instruction. Now that the main routine program is not ended, Y31 gets turned ON.

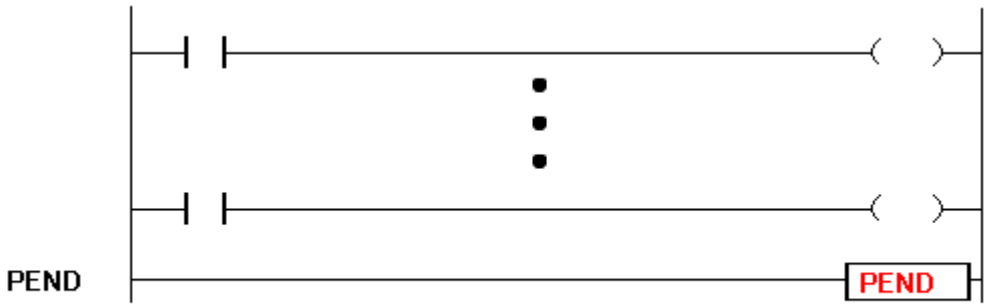
Memory Monitor 1

Y Dev | INT | Ascending Bit

CARD	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	DEC	HEX
Y000	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	H0000
Y001	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	H0000
Y002	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3	H0003
Y003	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	H0002
Y004	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	H0000

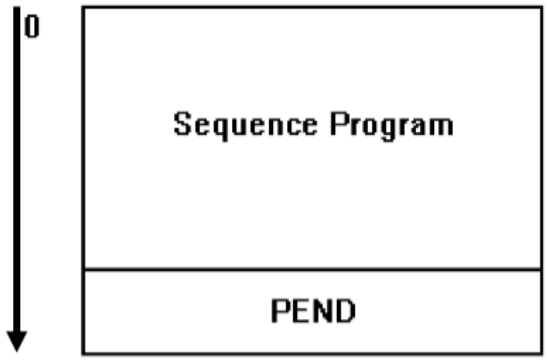
1.6.3. End Sequence Program: PEND

Instruction		Device address													No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
PEND	S	-	-	-	-	-	-	-	-	-	-	-	-	-	1	-	-	-



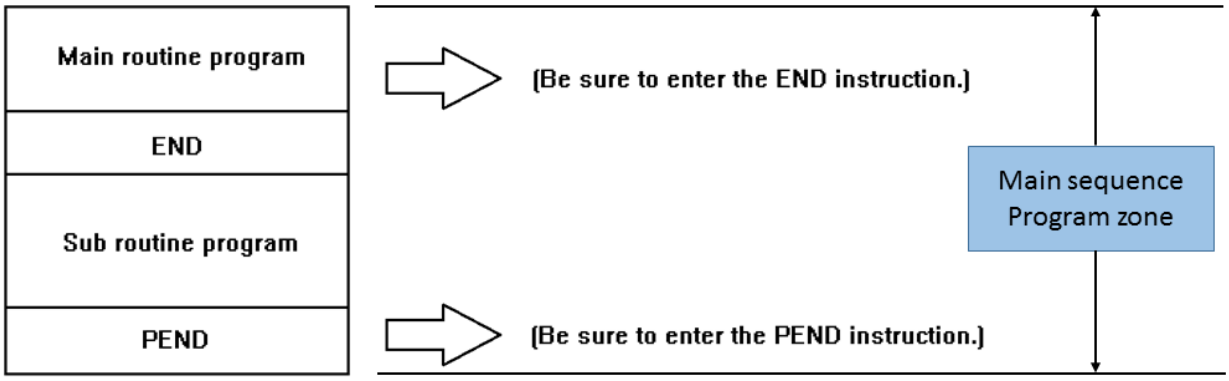
Functions:

- PEND instruction marks the end of a program. Scanning ends here and goes back to 0 step.



- While main sequence program and sub sequence program are running, it is impossible to use the PEND instruction. Instead, use the END instruction if needed to end the program while running.

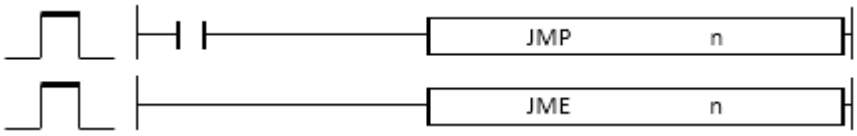
- In case of having main routine program, sub routine program, interrupt program, and sub sequence program all together, how to use the END instruction and the PEND instruction is shown in the below image.



1.7. Program Branch Instruction

1.7.1. Jump Instruction: JMP, JMPP, JME

Instruction		Device address												No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
JMP (P) JME	n	-	-	-	-	-	-	-	-	-	-	-	-	o	1	-	-	-

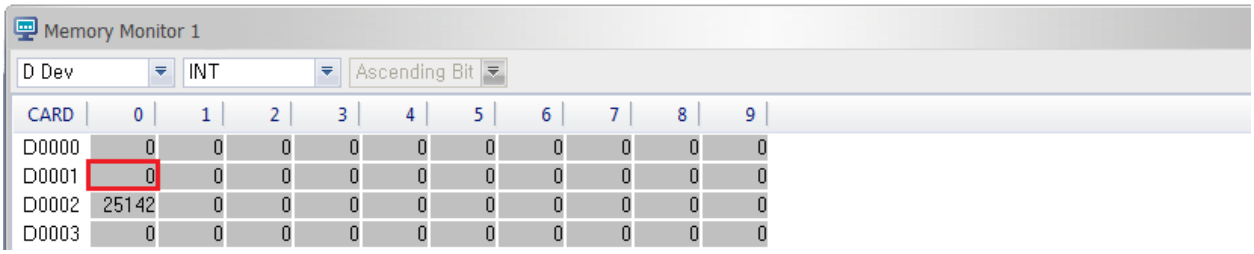


Functions:

- When the input for the JMP instruction is turned ON, the execution of a program jumps into the JME instruction, disabling and skipping all the instructions between JMP n and JME n.

- For example, when the input (M00) for the JMP 1 is turned ON, the program skips and will not operate the INC D10 instruction, resulting in '0' value for the device D10 because it is between JMP 1 and JME 1.

- While D10 stays '0' value due to the JMP command, notice how the value of D20 is +25792 because the INC D20 instruction is not included between JMP 1 and JME 1.



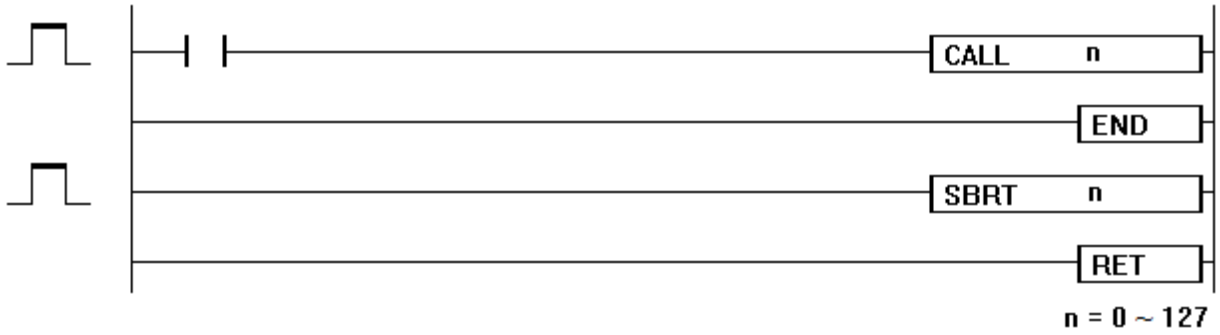
- Multiple JMP instructions can pair with one JME instruction.

- Though the number from 0 to 127 can be used for the JMP Number n, this number should not be interchangeable with the number used for sub routine (SBRT, RET) function.

- Please note that the JMP instruction between SBRT and RET blocks is considered as an error.

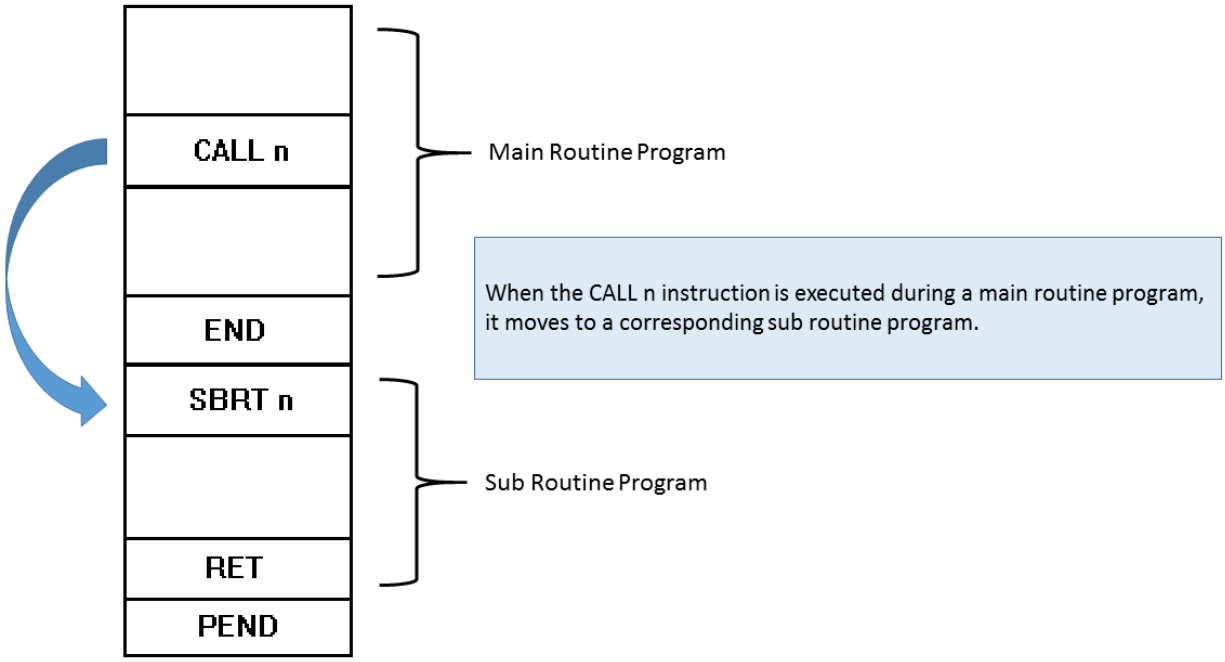
1.7.2. Sub Routine Instruction: CALL, CALLP, SBRT, RET

Instruction		Device address													No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
CALL (P)	n	-	-	-	-	-	-	-	-	-	-	-	-	o	1	-	-	-
SBRT	n	-	-	-	-	-	-	-	-	-	-	-	-	o	1	-	-	-
RET	-	-	-	-	-	-	-	-	-	-	-	-	-	-	1	-	-	-



Functions:

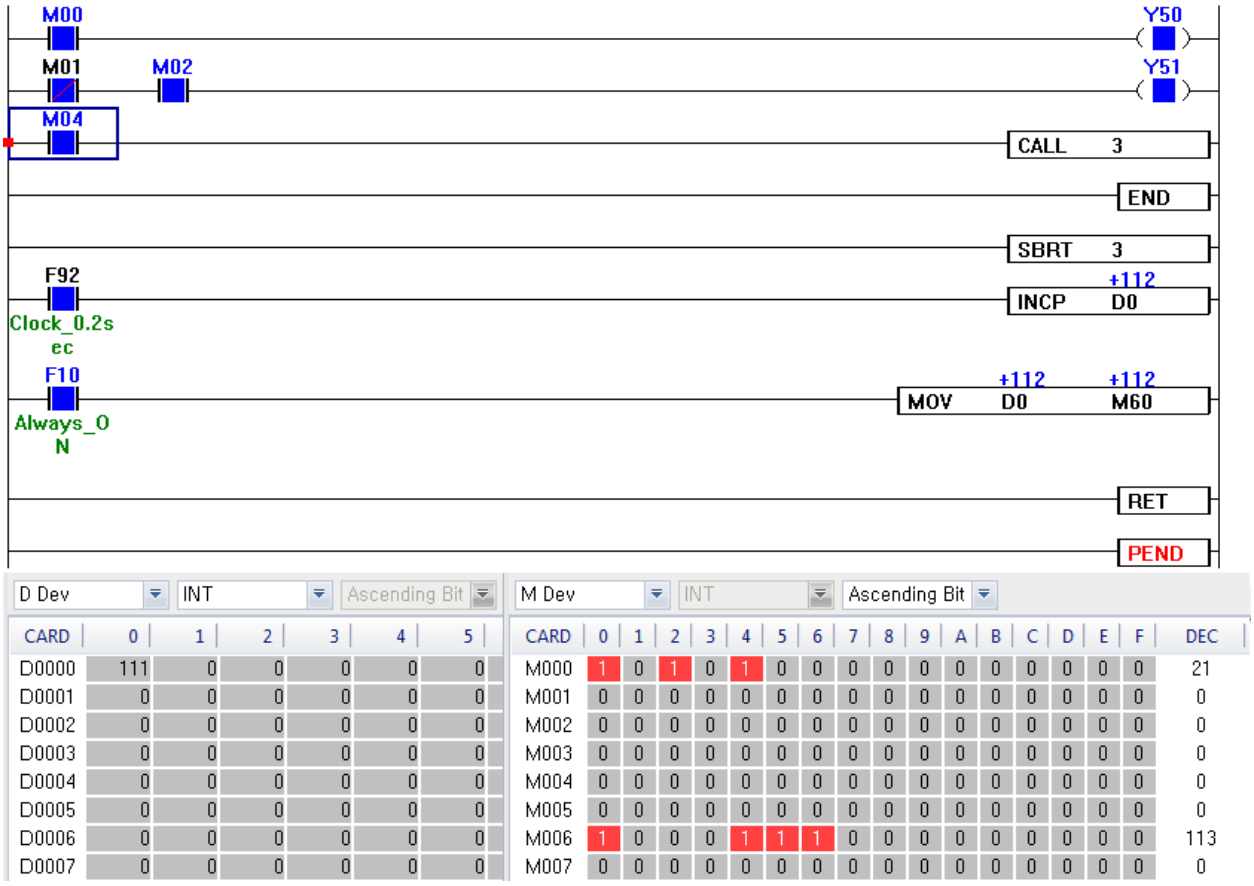
- When the input condition for the CALL n instruction is turned ON, a program executes the called operations between the SBRT n and the RET according to the CALL n.



- CALL No. can be used by nesting.
- The instructions between the SBRT n and the RET must be located after the END instruction.
- Conditions processed as an error:
  a. When a value of n is entered other than 0 ~ 127.
  b. When the CALL n instruction exists without the SBRT n.
  c. When the SBRT n or the RET stands alone.
- It is possible to CALL another SBRT within a SBRT up to maximum of 16 times.

Example)

- When the input condition (M04) for the CALL 3 is turned ON, the program executes the called operations (F92 -> INCP D0 / F10 -> MOV D0 M60) between the SBRT 3 and the RET according to the CALL 3 instruction.



CARD	0	1	2	3	4	5
D0000	111	0	0	0	0	0
D0001	0	0	0	0	0	0
D0002	0	0	0	0	0	0
D0003	0	0	0	0	0	0
D0004	0	0	0	0	0	0
D0005	0	0	0	0	0	0
D0006	0	0	0	0	0	0
D0007	0	0	0	0	0	0

CARD	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	DEC
M000	1	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	21
M001	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
M002	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
M003	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
M004	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
M005	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
M006	1	0	0	0	1	1	1	0	0	0	0	0	0	0	0	0	113
M007	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

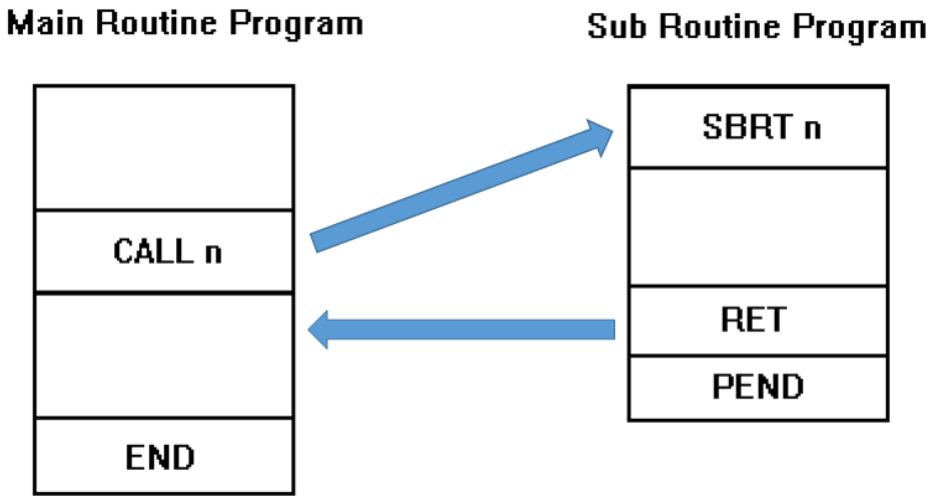
- As the clock ticks every 0.2 seconds (F92), it will increase the value of D0 by 1.
- As F10 is always ON, it will move the value of D0 to the device M60.

1.7.3. Sub Routine Calls Between Program Files: ECALL, ECALLP, SBRT, RET

Instruction		Device address													No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
ECALL (P)	ID	-	-	-	-	-	-	-	-	-	-	-	-	o	3	-	-	-
	n	-	-	-	-	-	-	-	-	-	-	-	-	o				
SBRT	n	-	-	-	-	-	-	-	-	-	-	-	-	o	1	-	-	-
RET	-	-	-	-	-	-	-	-	-	-	-	-	-	-	1	-	-	-

Function:

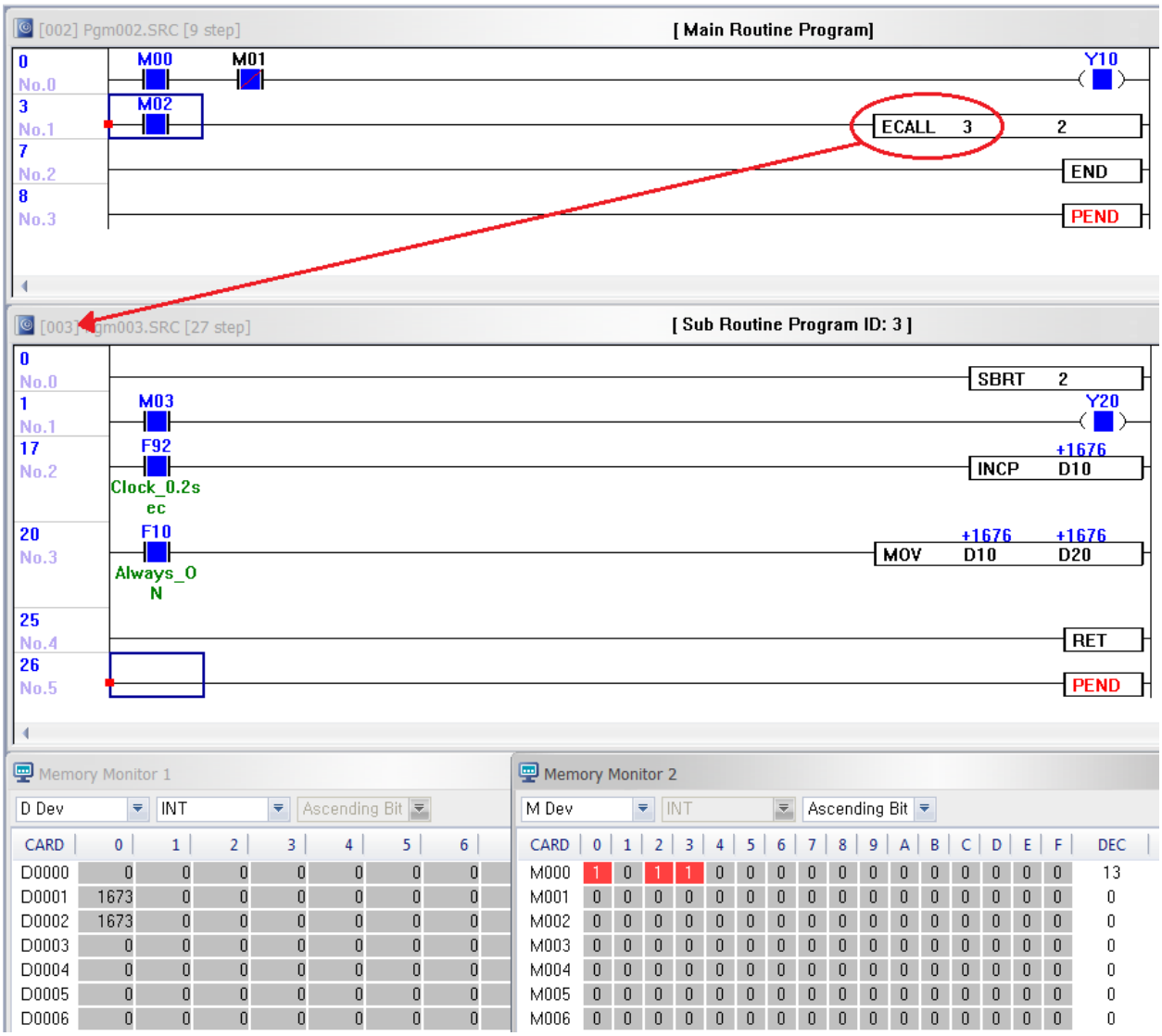
- When the input condition for the ECALL ID No. n instruction is turned ON, it moves to the corresponding sub routine ID program and executes the called operations between SBRT n and RET inside the sub routine program according to the ECALL ID No. n instruction.



- ECALL ID No. n can be used by nesting.
- The SBRT n and RET program must have the number corresponding to the ECALL ID No. n.
- Conditions processed as an error:
  a. When a value of n is entered other than 0 ~ 127.
  b. When the ECALL ID No. n instruction exists without the corresponding program file.
  c. When the SBRT n or the RET instruction stands alone.
- It is possible to CALL or ECALL another SBRT within a SBRT.

Example)

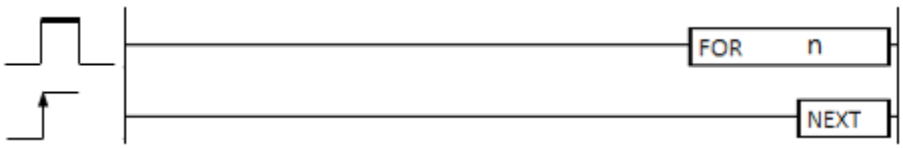
- When the input condition (M02) for the ECALL 3 (Program ID) 2 (Sub Routine) is turned ON, it moves to the corresponding sub routine program ID '3' and executes the called operations between the SBRT 2 and the RET according to the ECALL 3 2 instruction.


[002] Pgm002.SRC [9 step]
[Main Routine Program]
0
No.0
M00
M01
Y10
3
No.1
M02
ECALL 3 2
7
No.2
END
8
No.3
PEND
[003] Pgm003.SRC [27 step]
[Sub Routine Program ID: 3]
0
No.0
SBRT 2
1
No.1
M03
Y20
17
No.2
F92
Clock_0.2sec
+1676
INCP D10
20
No.3
F10
Always_ON
+1676 +1676
MOV D10 D20
25
No.4
RET
26
No.5
PEND
Memory Monitor 1
D Dev
INT
Ascending Bit
CARD 0 1 2 3 4 5 6
D0000 0 0 0 0 0 0 0
D0001 1673 0 0 0 0 0 0
D0002 1673 0 0 0 0 0 0
D0003 0 0 0 0 0 0 0
D0004 0 0 0 0 0 0 0
D0005 0 0 0 0 0 0 0
D0006 0 0 0 0 0 0 0
Memory Monitor 2
M Dev
INT
Ascending Bit
CARD 0 1 2 3 4 5 6 7 8 9 A B C D E F DEC
M000 1 0 1 1 0 0 0 0 0 0 0 0 0 0 0 0 13
M001 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
M002 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
M003 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
M004 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
M005 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
M006 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

1.8. Structure Instruction

1.8.1. FOR-NEXT Repetition: FOR, NEXT

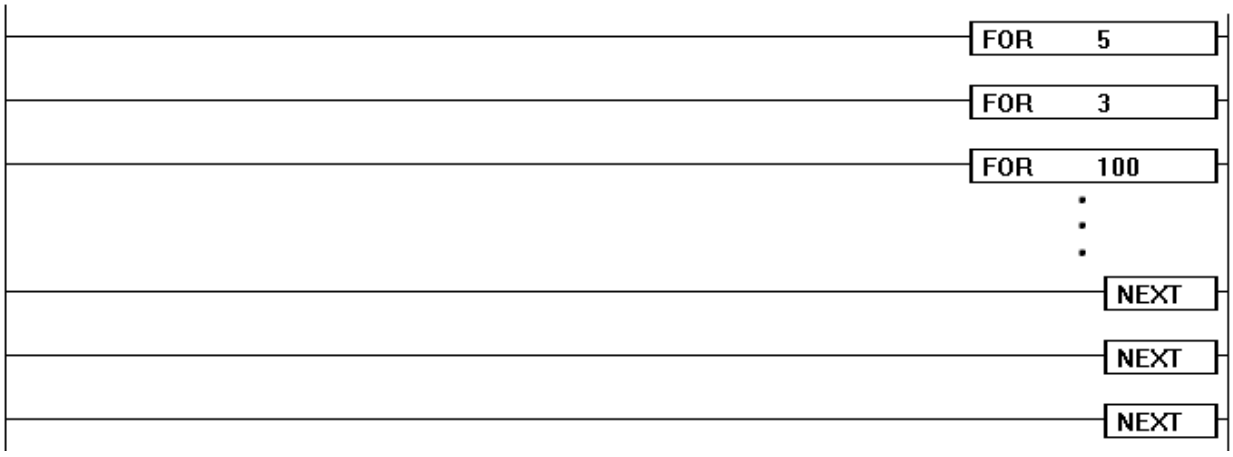
Instruction		Device address													No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
FOR	S	o	o	o	o	o	o	o	o	-	o	o	o	o	2	o	-	-
NEXT	S	-	-	-	-	-	-	-	-	-	-	-	-	-	1			



n	Number for Repeat ladder logic between FOR and NEXT (1 ~ 32767).
Error (F110)	Occurs when a device is assigned to @D and the device range limit exceeds.

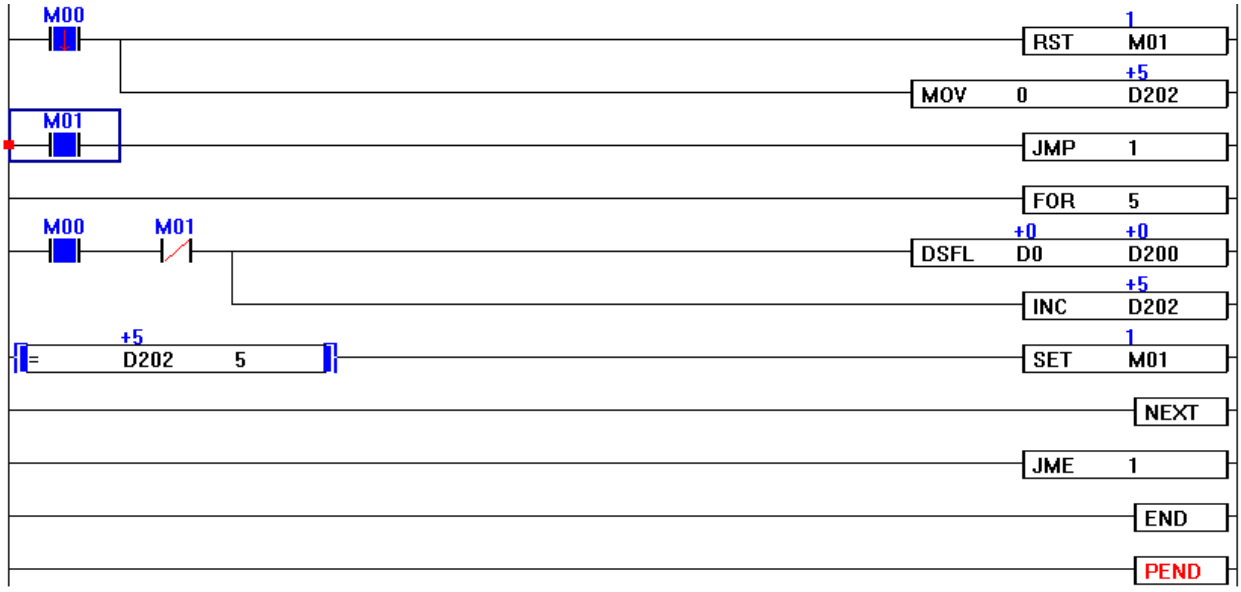
Functions:

- Repeats the instructions between the FOR and the NEXT for the designated number of times.
- The repeat count (n) can be assigned to one of the numbers from 1 to 32767.
- When the repeat count (n) number is from -32767 to 0, this instruction is operated as n=1.
- The nesting for the FOR instruction is possible up to 16 times.



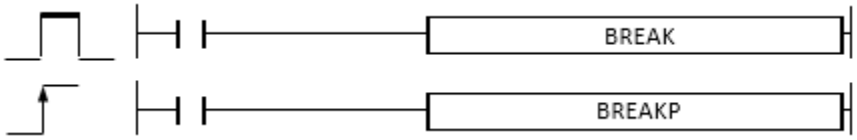
Example)

- When M01 is turned OFF, the FOR-NEXT instruction is executed.
- When M01 is turned ON, the FOR-NEXT instruction is NOT executed.



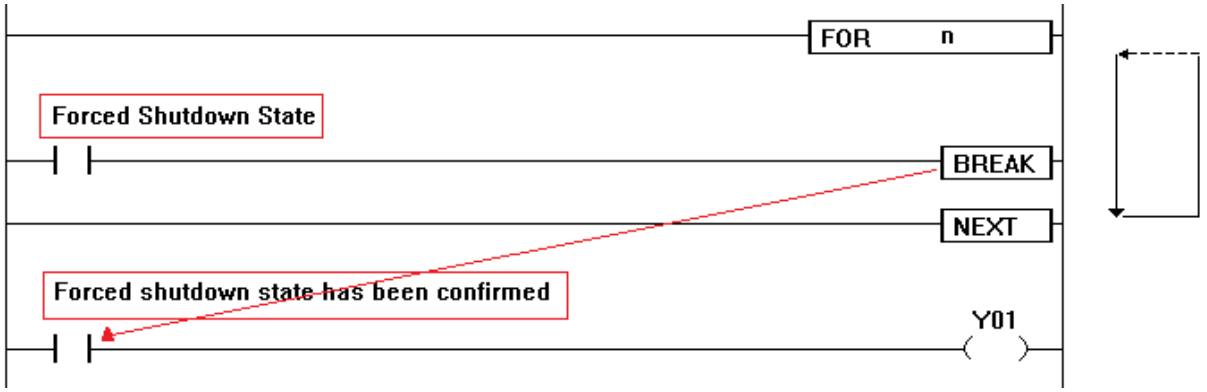
1.8.2. Force FOR-NEXT Repetition to End: BREAK, BREAKP

Instruction		Device address												No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
BREAK (P)	S	-	-	-	-	-	-	-	-	-	-	-	-	-	1	-	-	-



Function:

- Forcefully end the FOR-NEXT instruction to repeat the operations.
- The BREAK instruction can only be used while the FOR-NEXT is operated.



- If not executed, the BREAK instruction cannot stop the FOR-NEXT instruction from repeating the operation for the designated number of times.

Example)

- When M08 is turned OFF, the FOR-NEXT instruction will be operated.
- When M08 is turned ON, the BREAK instruction will prevent the FOR-NEXT instruction from operating.

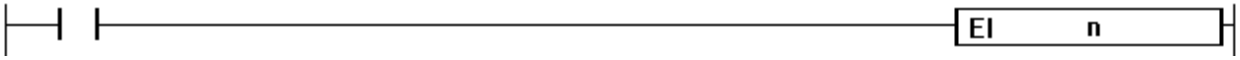
1.9. Program Execution Control Instruction

1.9.1. Enable, Disable and End Interrupt: EI, DI, GEI, GDI, IRET

Instruction		Device address												No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
EI, DI	n	-	-	-	-	-	-	-	-	-	-	-	-	o	1	-	-	-
GEI, GDI	-	-	-	-	-	-	-	-	-	-	-	-	-	-	1	-	-	-
IRET	-	-	-	-	-	-	-	-	-	-	-	-	-	-	1	-	-	-

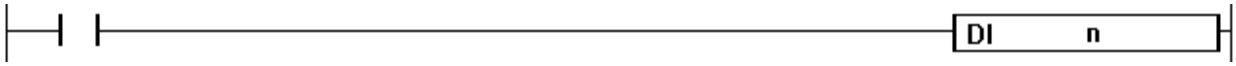
1) EI (Enable Interrupt)

- Enables the Interrupt ID program assigned to n to operate its Time Driven Interrupt.
- The Interrupt set up by a parameter can be operated after this instruction is executed.
- When converted to RUN Mode, an interrupt program is in the DI (Disable Interrupt) status.
- To use an interrupt program, the GEI should be executed and then the EI should be executed.



2) DI (Disable Interrupt)

- Disables the Interrupt ID program assigned to n to stop its Time Driven Interrupt.
- After this instruction is operated, the Interrupt ID program assigned to n will not be executed.



3) GEI

- Enables all the programs (that are assigned to interrupt) to operate their Time Driven Interrupt.
- The interrupt set up by a parameter can be operated after this instruction is operated.
- When converted to RUN Mode, all interrupt programs are in the DI (Disable Interrupt) status.
- To use all the interrupt program routines, the GEI should be executed.

4) GDI

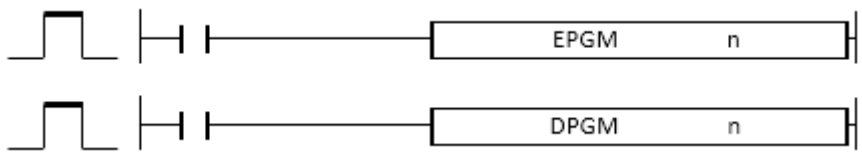
- Disables all the Time Driven Interrupts.
- After this instruction is operated, all the interrupt ID program routines will not be executed.

5) IRET

- Displays the end of an interrupt program.
- Used in the interrupt program specifically assigned to Time Driven Interrupt.

1.9.2. Enable, Disable Scan Program: EPGM, DPGM

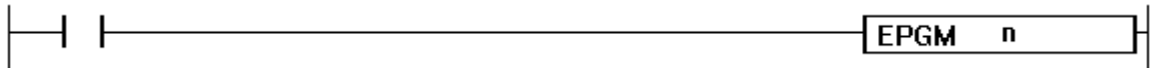
Instruction		Device address												No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
EPGM DPGM	n	-	-	-	-	-	-	-	-	-	-	-	-	o	1	-	-	-



n	Scan Program ID

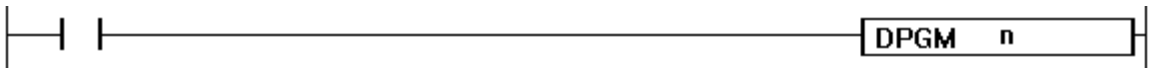
1) EPGM (Enable Scan Program)

- Enables to run the scan program that corresponds to ID number 'n.'
- Used when to enable again the previously disabled scan program by DPGM command.
- If the Disable Program (DPGM) command is not entered when converted to RUN Mode, all the operations of the program are enabled.



2) DPGM (Disable Scan Program)

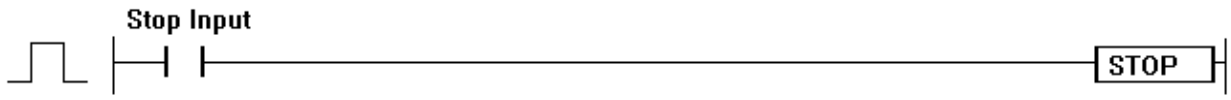
- Disables to run the scan program that corresponds to ID number 'n.'
- In order to run the disabled scan program again, the EPGM command must be used.



1.10. Other Basic Instructions

1.10.1. Stop Sequence Program: STOP

Instruction		Device address												No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
STOP	S	-	-	-	-	-	-	-	-	-	-	-	-	-	1	-	-	-



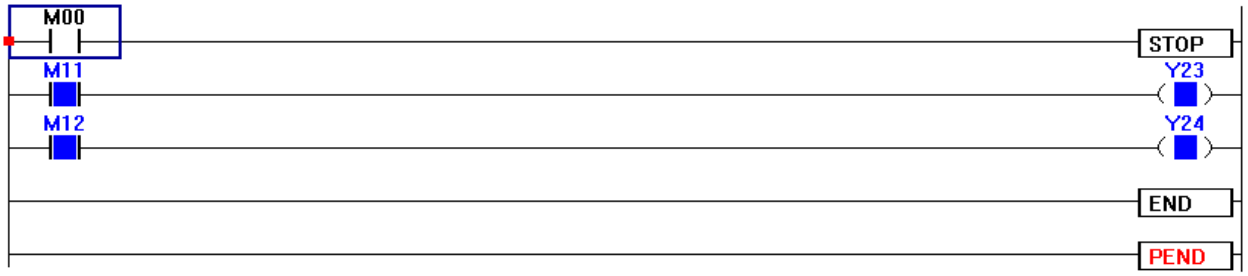
1) STOP

- The STOP command stops the operation of the PLC when the Stop Input is turned ON.
- This function is the same as shifting the Key Switch from RUN to STOP as shown below.

- To restart the operation of the PLC, shift the PLC Mode from RUN to STOP to RUN.
- Do not use the STOP command in the Interrupt Program, Sub Routine Program and FOR-NEXT.

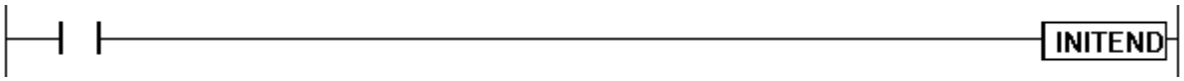
Example)

When the Stop Input (M00) is turned ON, it will STOP the sequence program (Y23 and Y24).



1.10.2. End Initialization Program and Execute Scan Program: INITEND

Instruction		Device address												No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
INITEND	S	-	-	-	-	-	-	-	-	-	-	-	-	-	1	-	-	-

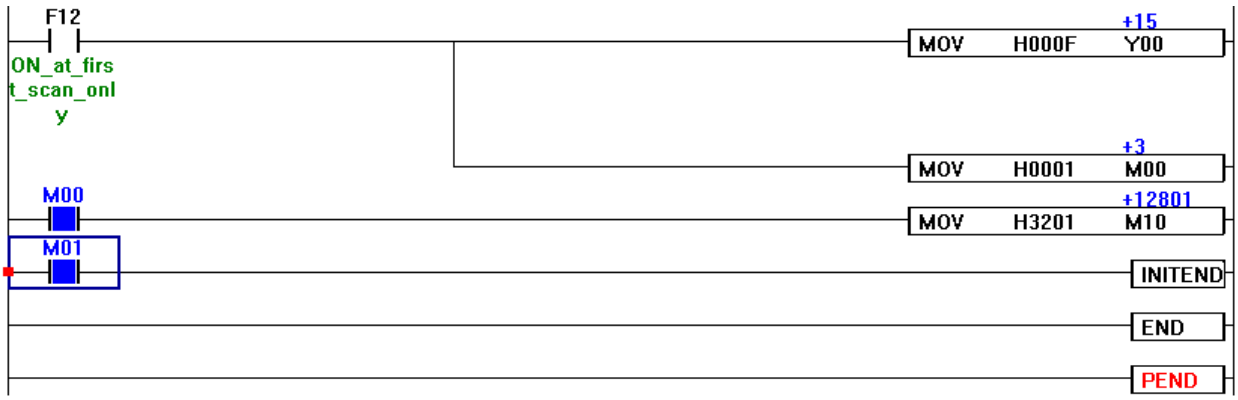


1) INITEND

- Ends an Initialization Program after the current scan and execute a scan program.
- Unless the INITEND command is executed, the Initialization Program will run continuously.
- The INITEND command can only be used for the Initialization Program.

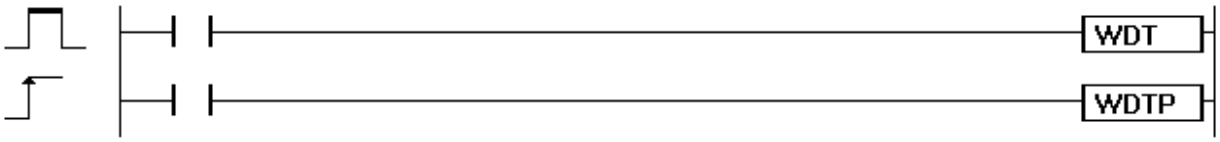
Example)

A program to end an Initialization Program



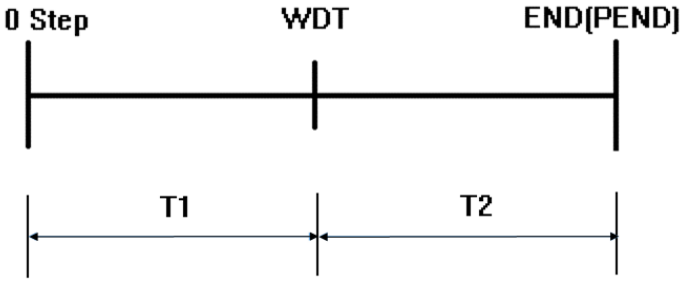
1.10.3. Reset Watch Dog Timer: WDT, WDTP

Instruction	Device address													No. of Steps	Flag		
	M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
WDT(P)	-	-	-	-	-	-	-	-	-	-	-	-	-	1	-	-	-



1) WDT

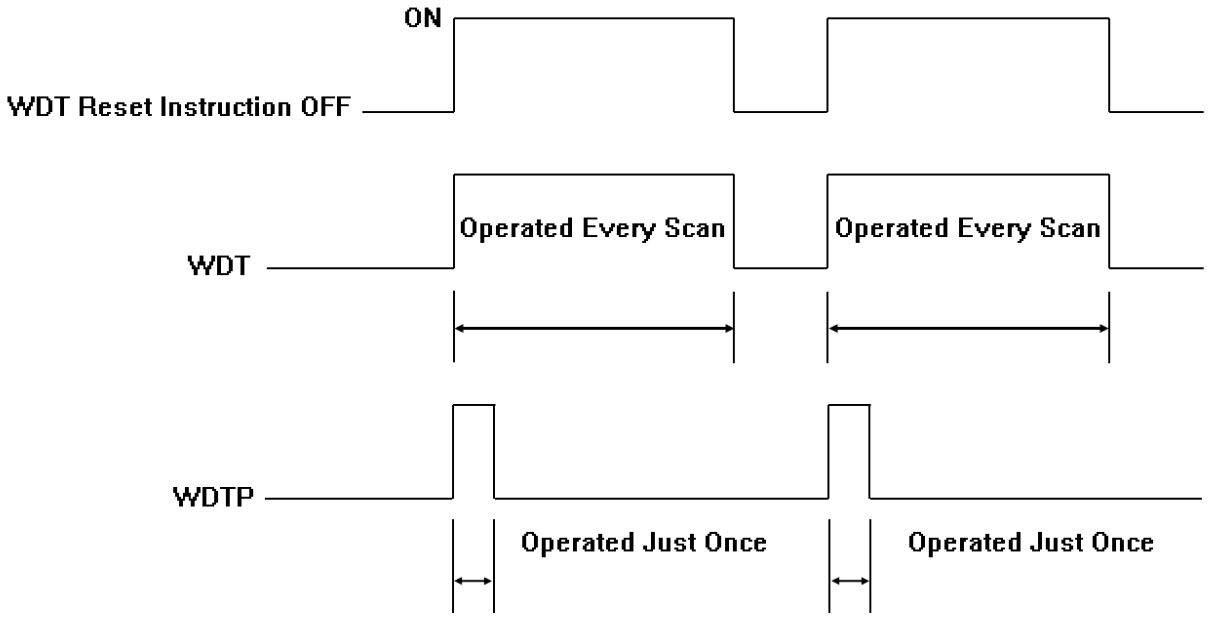
- Resets Watch Dog Timer in a sequence program.
- Used when the time from Step 0 to END(PEND) is over the set value of Watch Dog Timer according to conditions in a sequence program.
- When Scan Timer exceeds every set value of the Scan Watch Dog Timer, the set value of Watch Dog Timer should be changed in PLC Parameter.
- Must ensure that neither T1 (0 Step to WDT) nor T2 (WDT to END(PEND) exceeds the set value of Watch Dog Timer.



- Though the WDT command can be used over twice for one scan, ensure not to take too long to reset the output in case of an error.

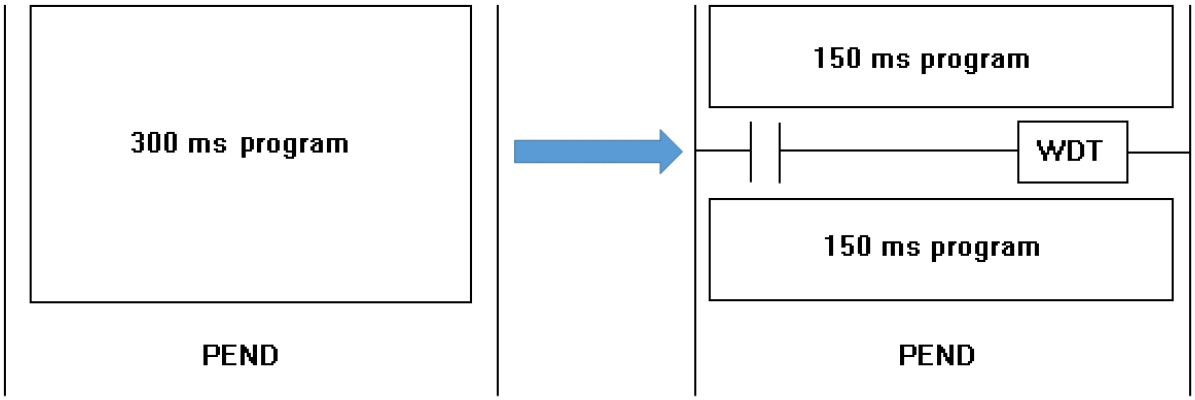
Conditions)

- The conditions to reset WDT are as follows:



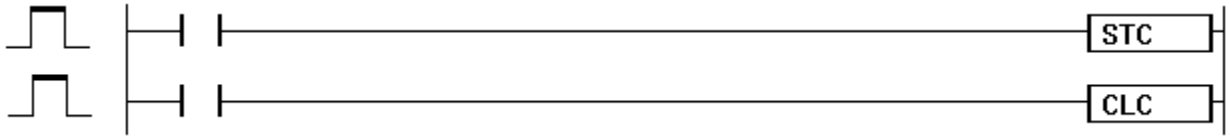
Example)

- The set value of Watch Dog Timer is 150 ms.
- It takes 300 ms from 0 Step to END(PEND).



1.10.4. Set or Reset Carry Flag: STC, CLC

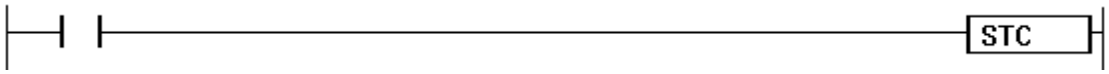
Instruction		Device address												No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
STC, CLC	n	-	-	-	-	-	-	-	-	-	-	-	-	-	1	-	-	o



Carry (F112)	If an input condition for STC is turned ON, Carry Flag is set.
	If an input condition for CLC is turned ON, Carry Flag is reset.
	If an input condition for STC or CLC is turned OFF, there is no change.

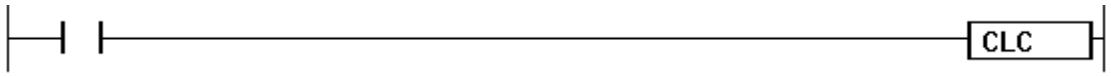
1) STC

- If an input condition for STC is turned ON, Carry Flag (F112) is SET (ON).



2) CLC

- If an input condition for CLC is turned ON, Carry Flag (F112) is RESET (OFF).

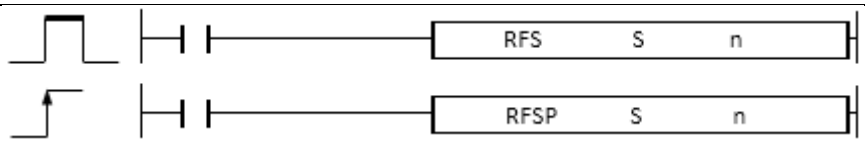


Example)

When an input condition (X01) for STC is turned ON, Carry Flag (F112) is SET.

1.10.5. Refresh I/O: RFS, RFSP

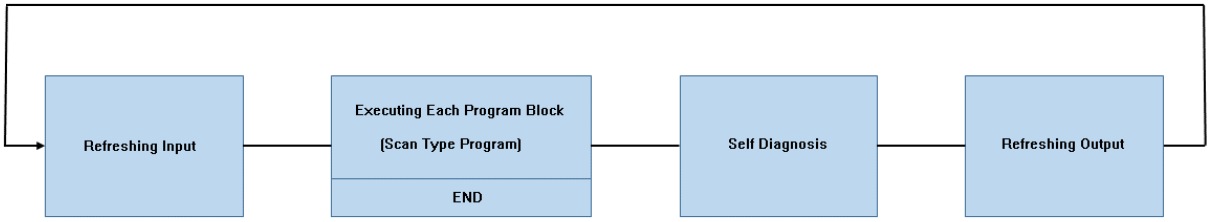
Instruction		Device address												No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
RFS(P)	S	-	o	o	-	-	-	-	-	-	-	-	-	-	3	-	-	-
	n	o	o	o	o	o	o	o	o	-	o	o	o	o				



S	Device to perform the Refresh Instruction
n	Number of devices to perform the Refresh Instruction

1) RFS, RFSP

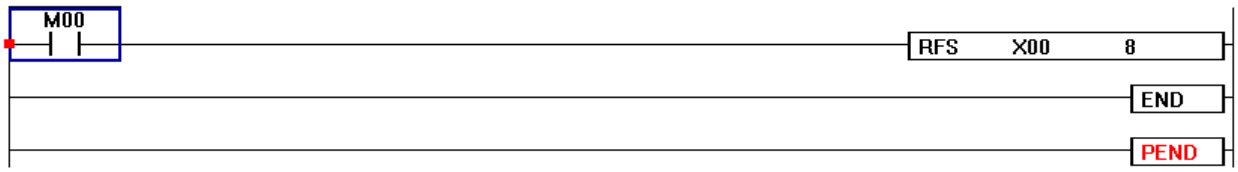
- When activated, the RFS command refreshes the devices of input and output modules while scanning.



- If the RFS command is activated while the scan is in progress in the above order, it will forcefully refresh input and output again during the execution of program block.
- By performing the refresh command, the newly authorized inputs and outputs can be identified.
- After the refresh instruction is executed, it performs the command according to the new input.

Example)

If M00 is turned ON, it will refresh the input modules of 8 word starting from X00.



2. APPLICATION INSTRUCTION

2.1. Comparison Operation Instruction

2.1.1. Relational operators with Logical operators for 16Bit(Word) and 32Bit(Double word) Data

Relational operators compare two values or strings to provide a true and false result.

If the comparison is true, the result is 1. (Instruction will carry on)

If the comparison is false, the result is 0. (Instruction will no carry on)

Format : Relational Operator(=,<>, <, >, <=, >=) value(S1) value(S2)

(Example : <= X00 D1)

If relational operator is double word (LDD, ANDD, ORD), write D in front of relational operator.

(Example : D<= 1000000 D10)

Instruction		Device address													No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
LD x, LDD x AND X, ANDD x, OR x, ORD x	S1	o	o	o	o	o	o	o	o	-	o	o	o	o	3	o	-	-
	S2	o	o	o	o	o	o	o	o	-	o	o	o	o				

Error(F110) :

1) LD x, LDD x (x : =, <, >, <=, >=, <>)

Functions
- Compare two values S1 and S2 and if the comparison is true, the instruction will carry on.

- S1 and S2 are signed number.

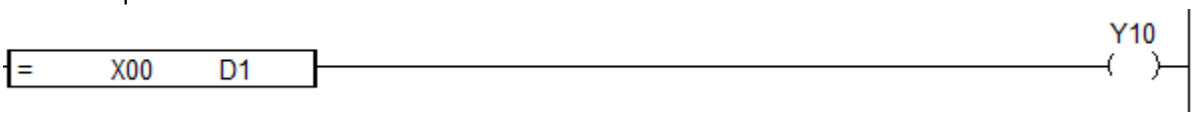
LD x : h8000 (-32768) ~ hFFFF (-1) < 0 ~ h7FFF (32767)

LDD x : h80000000 (-2147483648) ~ hFFFFFFFF (-1) < 0 ~ h7FFFFFFF (2147483647)

Comparison	Operator	Result
= Equal	S1 = S2	True
< Less than	S1 < S2	True
<= Less than or equal	S1 <= S2	True
> Greater than	S1 > S2	True
>= Greater than or equal	S1 >= S2	True
<> Not equal	S1 ≠ S2	True

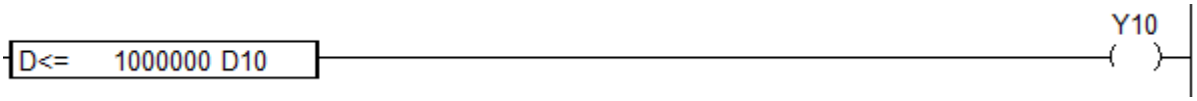
Example)

• If X00 equals D01 then Y10 will be turned ON.



• If D1 is greater than or equals to 1000 then Y10 will be turned ON.

• If 1000000 is less than or equal to D10 then Y10 will be turned ON.



2) AND x, ANDD x (x : =, <, >, <=, >=, <>)

Functions

- Compare two values S1 and S2 and if the comparison is true and multiple conditions are also true then the instruction will carry on.

- S1 and S2 are signed number.

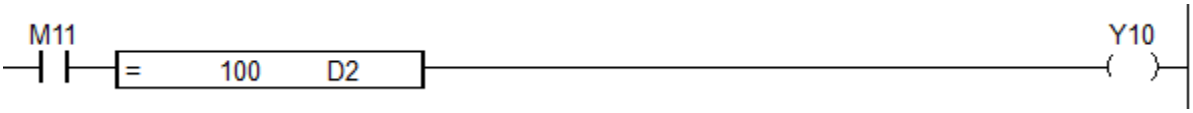
AND x : h8000 (-32768) ~ hFFFF (-1) < 0 ~ h7FFF (32767)

ANDD x : h80000000 (-2147483648) ~ hFFFFFFFF (-1) < 0 ~ h7FFFFFFF (2147483647)

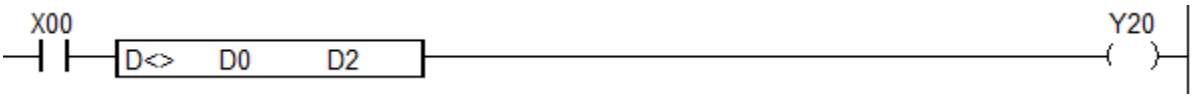
Comparison	Operator	Result
= Equal	S1 = S2	True
< Less than	S1 < S2	True
<= Less than or equal	S1 <= S2	True
> Greater than	S1 > S2	True
>= Greater than or equal	S1 >= S2	True
<> Not equal	S1 ≠ S2	True

Example)

• If M11 is ON and 100 equal to D2 then Y10 will be turned ON.



• If X00 is ON and D0(double word) is not equal to D2(double word) then Y20 will be turned ON.



3) OR x, ORD x (x : =, <, >, <=, >=, <>)

Functions

- Compare two values S1 and S2 and if either the comparison is true or other condition is true then the instruction will carry on.

- S1 and S2 are signed number.

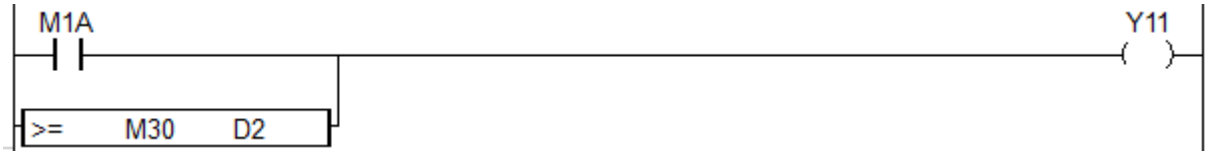
AND x : h8000 (-32768) ~ hFFFF (-1) < 0 ~ h7FFF (32767)

ANDD x : h80000000 (-2147483648) ~ hFFFFFFFF (-1) < 0 ~ h7FFFFFFF (2147483647)

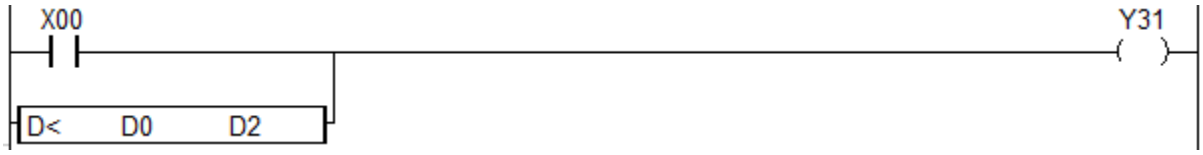
Comparison	Operator	Result
= Equal	S1 = S2	True
< Less than	S1 < S2	True
<= Less than or equal	S1 <= S2	True
> Greater than	S1 > S2	True
>= Greater than or equal	S1 >= S2	True
<> Not equal	S1 ≠ S2	True

Example)

• If either M1A is ON or M30 is greater than or equal to D2 then Y11 will be turned ON.

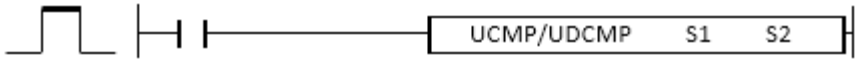


• If either X0 is ON or D0 is less than D2, Y31 will be turned ON.



2.1.2. Relational operators (Unsigned operations) for 16Bit(Word) and 32Bit(Double word) Data

Relational operators compare two values or strings to provide a true and false result.
If the comparison is true, the result is 1. (Instruction will carry on)
If the comparison is false, the result is 0. (Instruction will no carry on)



S1	Value to compare
S2	Value to compare

Instruction		Device address													No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
UCMP UDCMP	S1	o	o	o	o	o	-	-	-	-	o	o	o	o	3	o	-	-
	S2	o	-	o	o	o	-	-	-	-	o	o	o	o				

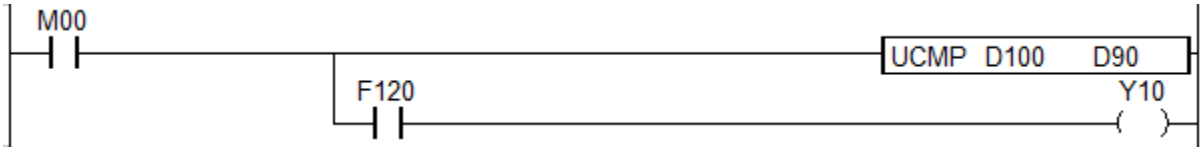
1) UCMP

Functions
- Compare two values S1 and S2 and if the comparison is true then set the Flag (F120~125) to 1.
- S1 and S2 are unsigned number.
- S1 =, <, >, <=, >=, <> S2 → Flag (F120~125)

Comparison	Operator	Flag
< Less than	S1 < S2	F120
<= Less than or equal	S1 <= S2	F121
= equal to	S1 = S2	F122
> Greater than	S1 > S2	F123
>= Greater than or equal	S1 >= S2	F124
<> Not equal	S1 ≠ S2	F125

Example)

• If M00 is ON and D100 is less than D90 then Y10 will be turned ON.



2) UDCMP

Functions

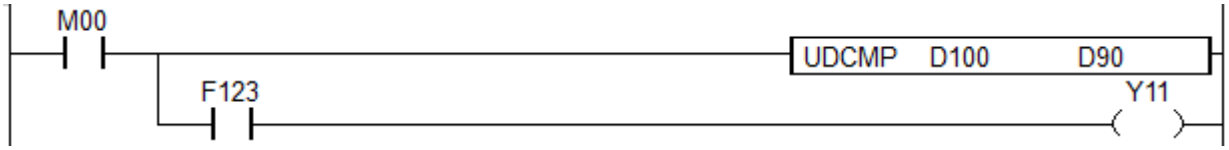
- Compare four values (double words) S1, S1+1 and S2, S2+1 and if the comparison is true then set the Flag (F120~125) to 1.
- S1, S1+1 and S2, S2+1 are unsigned number.

- (S1, S1+1) =, <, >, <=, >=, <> (S2, S2+1) → Flag (F120~125)

Comparison	Operator	Flag
< Less than	S1 < S2	F120
<= Less than or equal	S1 <= S2	F121
= equal to	S1 = S2	F122
> Greater than	S1 > S2	F123
>= Greater than or equal	S1 >= S2	F124
<> Not equal	S1 ≠ S2	F125

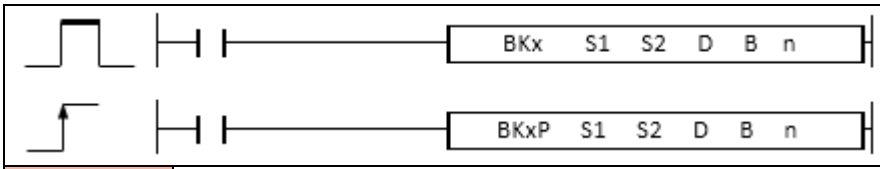
Example)

• If M00 is ON and D100 and D101 are greater than D90 and D91 then Y11 will be turned ON.



2.1.3. Relational operators (Block comparison) for 16Bit(Word) and 32Bit(Double word) Data

Relational operators compare two values or strings to provide a true and false result.



S1	Value to compare
S2	Value to compare
D	Destination where comparison result is stored
B	Starting bit address for D
n	Number of block for comparison

Instruction		Device address													No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
BK(P)	S1	o	o	o	o	o	o	o	o	-	o	o	o	o	6	o	-	-
	S2	o	o	o	o	o	o	o	o	-	o	o	o	o				
	D	o	-	o	o	o	-	-	-	-	o	o	o	-				
	B	o	o	o	o	o	o	o	o	-	o	o	o	o				
	n	o	o	o	o	o	o	o	o	-	o	o	o	-				

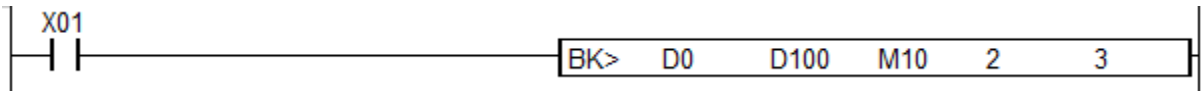
1) BK

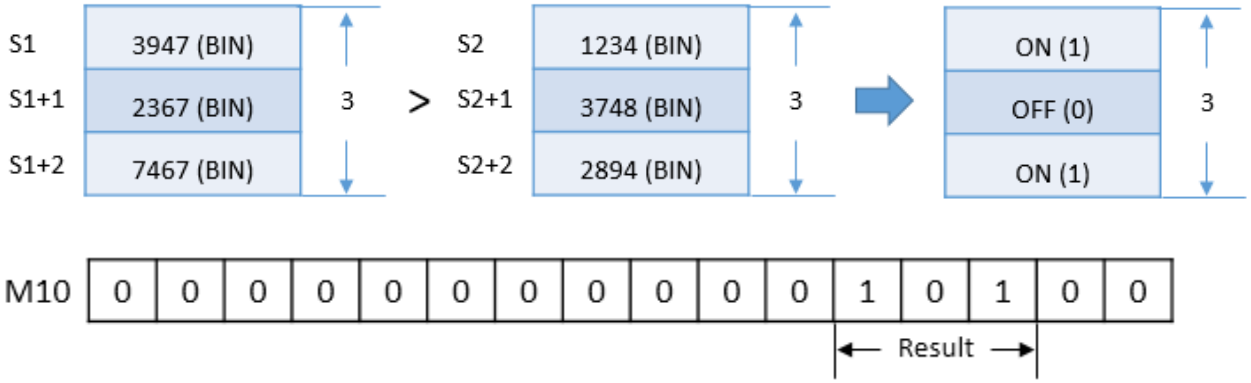
Functions
- Compare n number of S1 and n number of S2 and save the result from D2 of assigned D1 in sequence.
- If the result is true then the assigned bit on D will be turned ON.
- If the result is false then the assigned bit on D will be turned OFF.
- Relational operation is word type.
- The range of S1 is -32768 ~ 32767.
- S1 =, <, >, <=, >=, <> S2 → Flag (F120~125)

Comparison	Operator	Result
BK< Less than	S1 < S2	ON
BK<= Less than or equal	S1 <= S2	ON
BK= equal to	S1 = S2	ON
BK> Greater than	S1 > S2	ON
BK>= Greater than or equal	S1 >= S2	ON
BK<> Not equal	S1 ≠ S2	ON

Example)

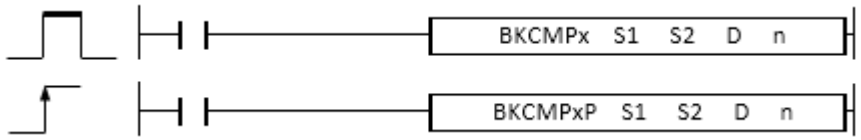
• If X01 is ON and D0, D1 and D2 are greater than D100, D101 and D103 then set the M12, M13 and M14 to 1 in sequence.





2.1.4. Relational operators (Block comparison)

Relational operators compare two values or strings to provide a true and false result.



S1	Value to compare
S2	Value to compare
D	Destination where comparison result is stored
n	Number of block for comparison

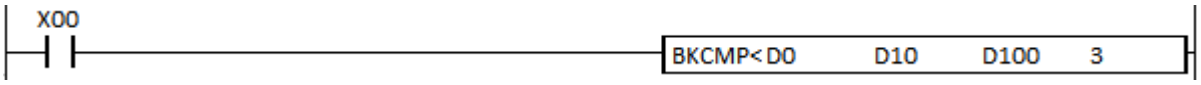
Instruction		Device address													No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
BKCMP BKCMPP	S1	o	o	o	o	o	o	o	o	-	o	o	o	o	5	o	-	-
	S2	o	o	o	o	o	o	o	o	-	o	o	o	o				
	D	o	-	o	o	o	-	o	o	-	o	o	o	o				
	n	o	o	o	o	o	o	o	o	o	o	o	o	o				

- Compare n number of S1 and n number of S2 and save the result to the Destination.
- If the result is true then the assigned bit on D will be turned ON.
- If the result is false then the assigned bit on D will be turned OFF.

Comparison	Operator	Result
BKCMP< Less than	S1 < S2	ON
BKCMP<= Less than or equal	S1 <= S2	ON
BKCMP= equal to	S1 = S2	ON
BKCMP> Greater than	S1 > S2	ON
BKCMP>= Greater than or equal	S1 >= S2	ON
BKCMP<> Not equal	S1 ≠ S2	ON

Example)

• If X00 is ON and D0, D1 and D2 are less than D10, D11 and D13 then set the D100, D101 and D103 to 1 in sequence.



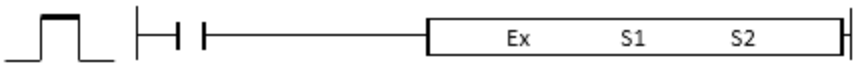
2.1.5. Relational operators with Logical operators for float data

(It is supported only in XP and PLC-S CPU series)

Relational operators compare two float values to provide a true and false result.

If the comparison is true, the result is 1. (Instruction will carry on)

If the comparison is false, the result is 0. (Instruction will no carry on)



S1	Value to compare
S2	Value to compare

Instruction		Device address													No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
LDE x, ANDE X, ORE x,	S1	o	o	o	o	o	-	-	-	-	o	o	o	-	3	o	-	-
	S2	o	o	o	o	o	-	-	-	-	o	o	o	-				

1) LDE x (x : =, <, >, <=, >=, <>)

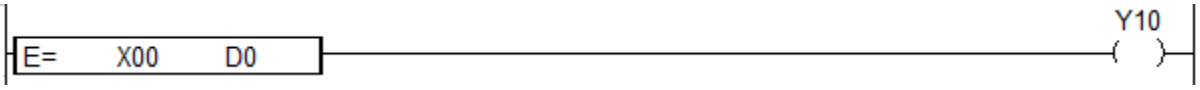
Functions

- Compare two float values S1 and S2 and if the comparison is true, the instruction will carry on.
- S1 and S2 are signed number.

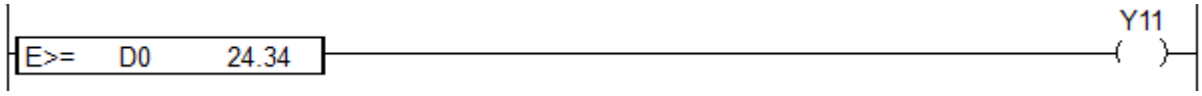
Comparison	Operator	Result
= Equal	S1 = S2	True
< Less than	S1 < S2	True
<= Less than or equal	S1 <= S2	True
> Greater than	S1 > S2	True
>= Greater than or equal	S1 >= S2	True
<> Not equal	S1 ≠ S2	True

Example)

• If X00 equals to D0 then Y10 will be turned ON.



• If D0 is greater than or equals to 24.34 then Y11 will be turned ON.



2) ANDE x (x : =, <, >, <=, >=, <>)

Functions

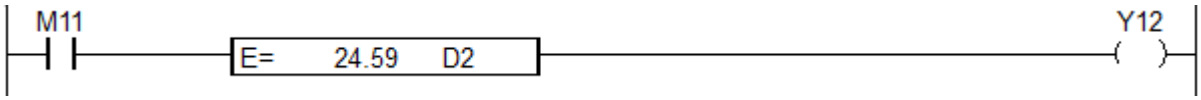
- Compare two float values S1 and S2 and if the comparison is true and multiple conditions are also true then the instruction will carry on.

- S1 and S2 are signed number.

Comparison	Operator	Result
= Equal	S1 = S2	True
< Less than	S1 < S2	True
<= Less than or equal	S1 <= S2	True
> Greater than	S1 > S2	True
>= Greater than or equal	S1 >= S2	True
<> Not equal	S1 ≠ S2	True

Example)

• If M11 is ON and 24.59 equal D2 then Y12 will be turned ON.



3) ORE x (x : =, <, >, <=, >=, <>)

Functions

- Compare two float values S1 and S2 and if either the comparison is true or other condition is true then the instruction will carry on.
- S1 and S2 are signed number.

Comparison	Operator	Result
= Equal	S1 = S2	True
< Less than	S1 < S2	True
<= Less than or equal	S1 <= S2	True
> Greater than	S1 > S2	True
>= Greater than or equal	S1 >= S2	True
<> Not equal	S1 ≠ S2	True

Example)

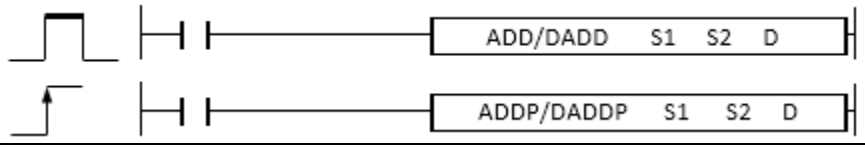
• If either M1A is ON or M30 is greater than or equal to D2 then Y13 will be turned ON.

• If either X0 is ON or D0 is less than D2, Y14 will be turned ON.

2.2. Arithmetic Operation Instruction

2.2.1. ADD, DADD, ADDP, DADDP (Binary)

ADD instruction adds two binary values (S1 and S2) and saves the result in Destination (assigned device address).

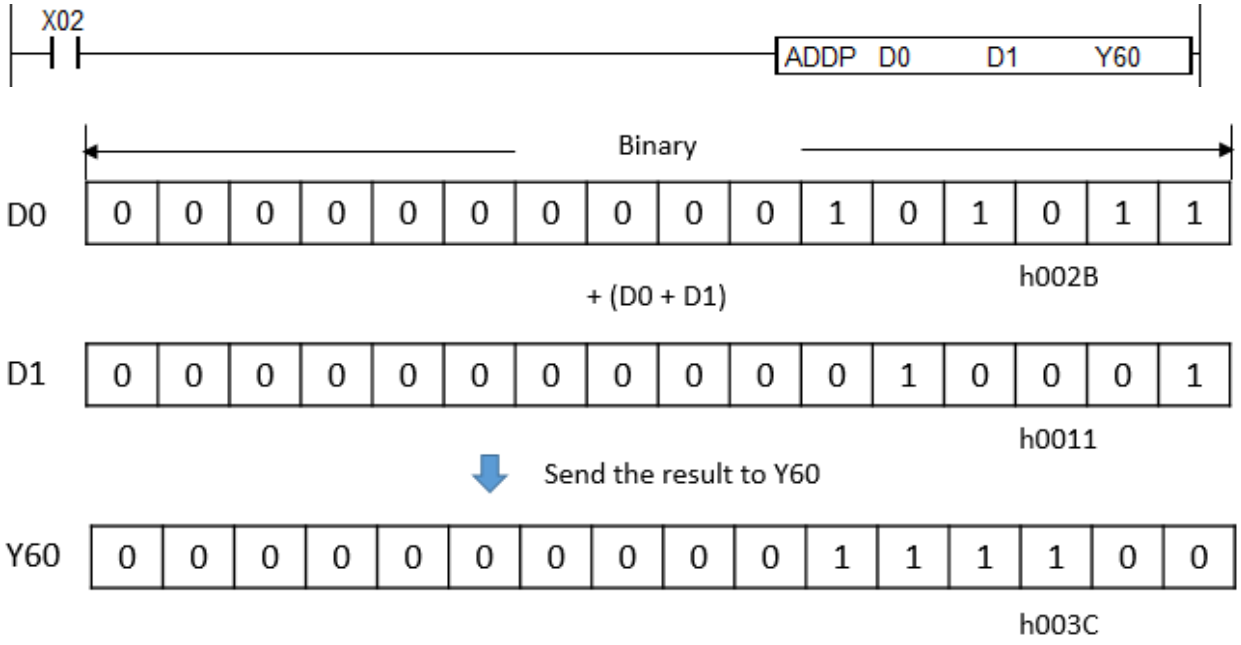


S1, S2	Source 1 and 2
D	Destination

Instruction		Device address													No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
ADD(P) DADD(P)	S1	o	o	o	o	o	o	o	o	-	o	o	o	o	4	o	-	-
	S2	o	o	o	o	o	o	o	o	-	o	o	o	o				
	D	o	-	o	o	o	-	o	o	-	o	o	o	-				

Example)

• If X02 is ON, add D0 and D1 and then send the result to Y60.

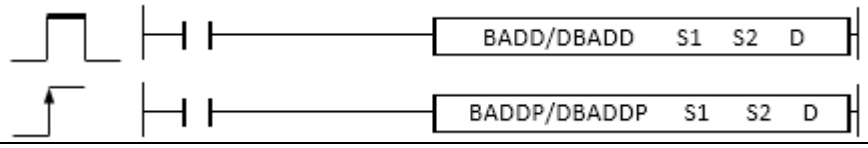


2.2.2. BADD, DBADD, BADDP, DBADDP (BCD code)

ADD instruction adds two BCD values (S1 and S2) and save the result in Destination (assigned device address).

- DBADD is Double word / BADDP and DBADDP are pulse.

Format: BADD value(S1) value(S2) Destination

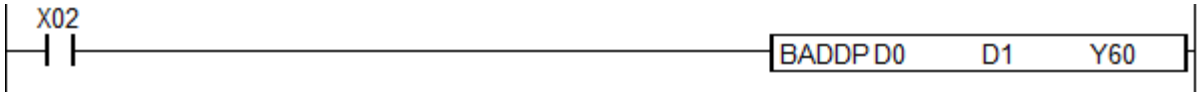


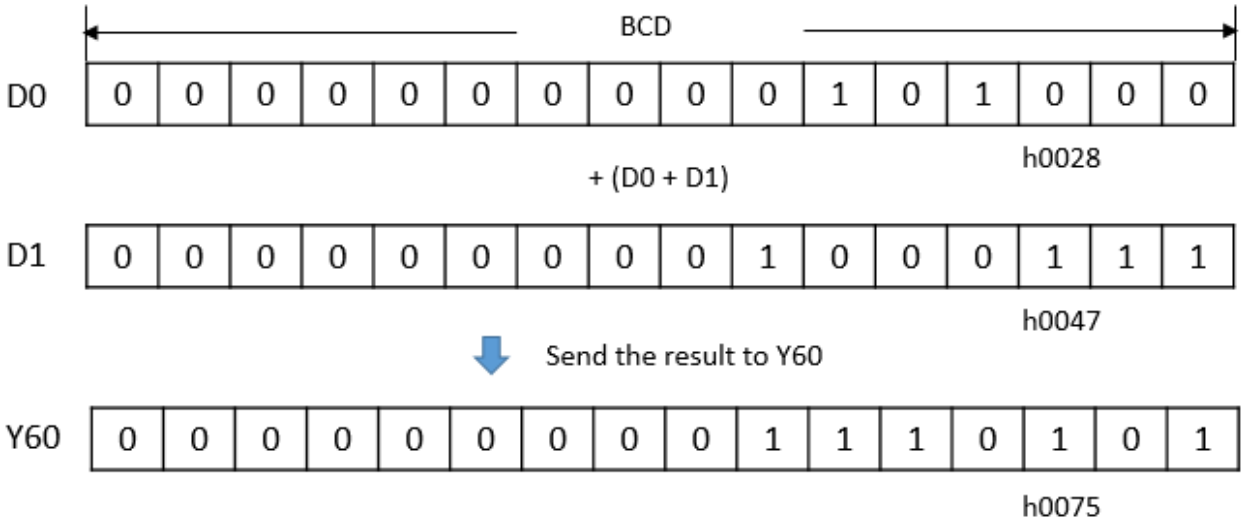
S1, S2	Source 1 and 2
D	Destination

Instruction		Device address													No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
BADD(P) DBADD(P)	S1	o	o	o	o	o	o	o	o	-	o	o	o	o	4	o	-	-
	S2	o	o	o	o	o	o	o	o	-	o	o	o	o				
	D	o	-	o	o	o	-	o	o	-	o	o	o	-				

Example)

• If X02 is ON, add D0 and D1 and then send the result to Y60.

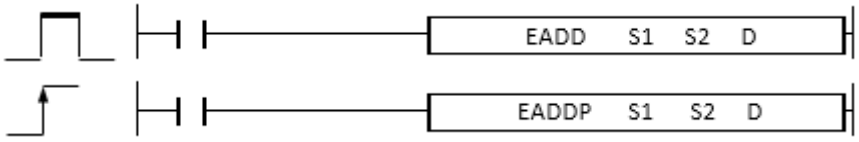




2.2.3. EADD, EADDP (Float)

ADD instruction adds two float values (S1 and S2) and save the result in Destination (assigned device address).

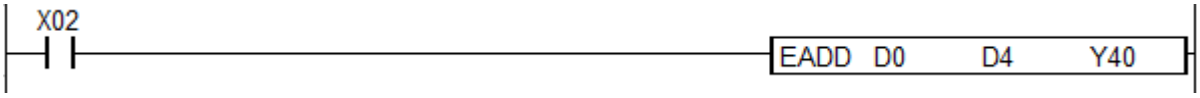
(It is supported only in XP and PLC-S CPU series)



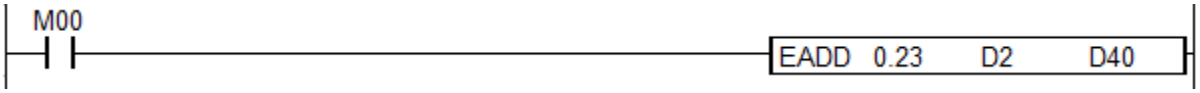
S1, S2	Source 1 and 2
D	Destination

Example)

• If X02 is ON, add D0 and D4 and then send the result to Y40.



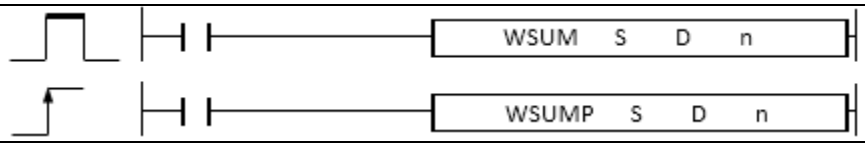
• If M00 is ON, add 0.23 and D2 and then send the result to D40.



2.2.4. WSUM, WSUMP (16bit data)

ADD (n) number of word data and save the result in Destination (double word)

(It is supported only in XP and PLC-S CPU series)

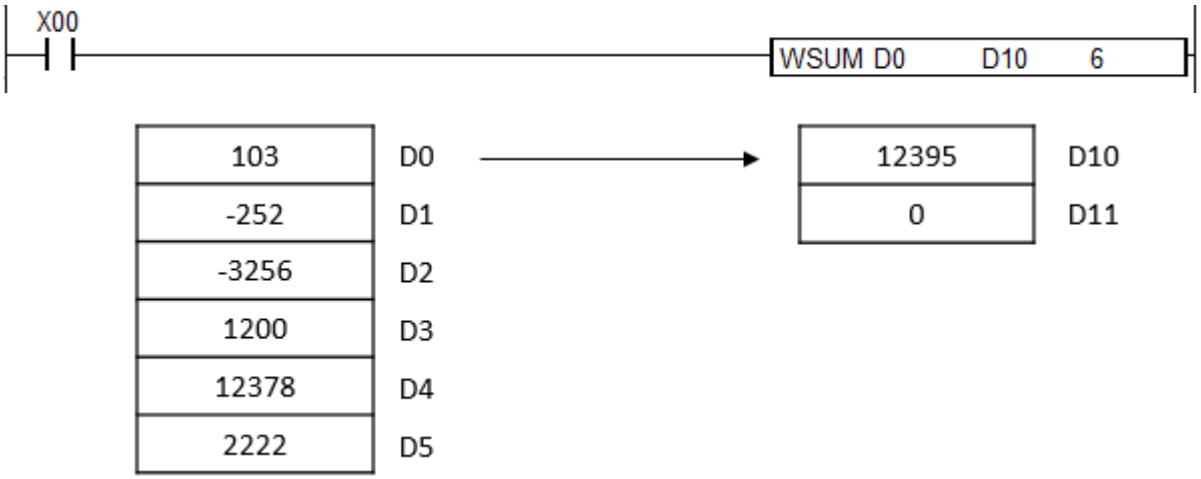


S	Starting address of Source
D	Destination
n	Number of source

Instruction		Device address													No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
WSUM WSUMP	S	o	o	o	o	o	o	o	o	-	o	o	o	-	4	o	-	-
	D	o	o	o	o	o	-	o	o	-	o	o	o	-				
	n	o	-	o	o	o	-	o	o	-	o	o	o	o				

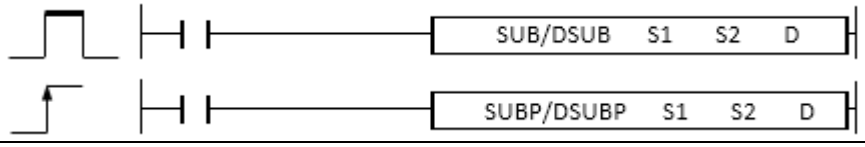
Example)

• If X00 is ON, add D0, D1, D2, D3, D4, and D5 and then send the result to D10 and D11.



2.2.5. SUB, SUBP, DSUB, DSUBP (Binary)

SUB instruction subtracts S2 from S1 and save the result in the Destination (assigned device address).

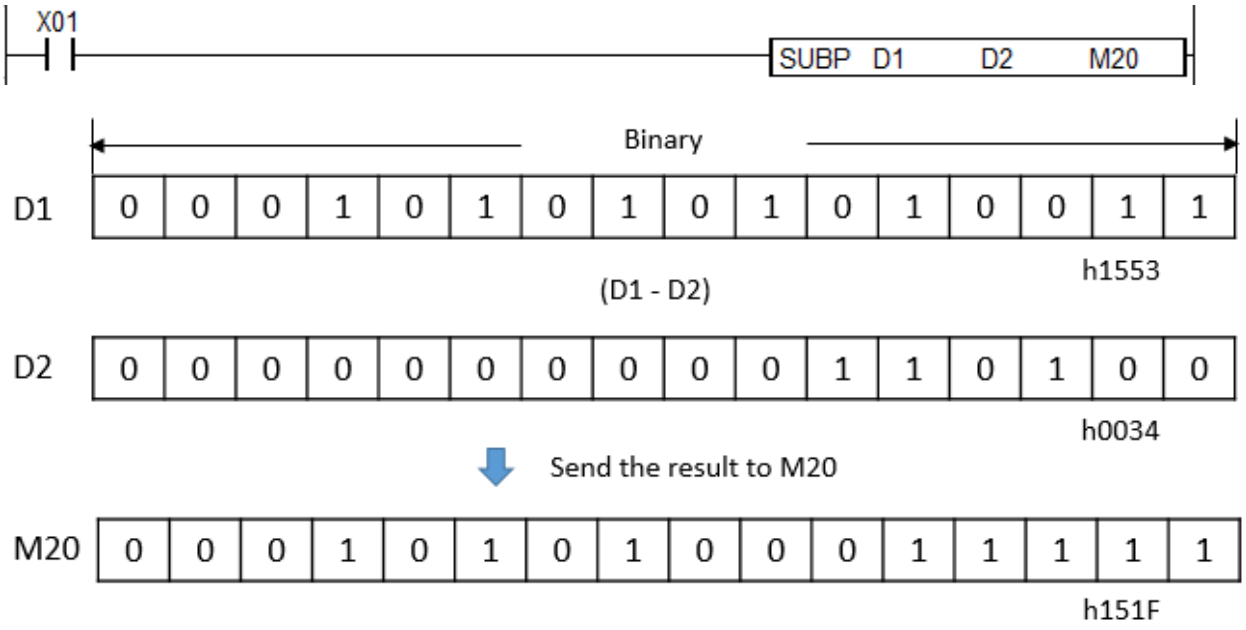


S1, S2	Source
D	Destination

Instruction		Device address													No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
SUB(P) DSUB(P)	S1	o	o	o	o	o	o	o	o	-	o	o	o	o	4	o	-	-
	S2	o	o	o	o	o	o	o	o	-	o	o	o	o				
	D	o	-	o	o	o	-	o	o	-	o	o	o	-				

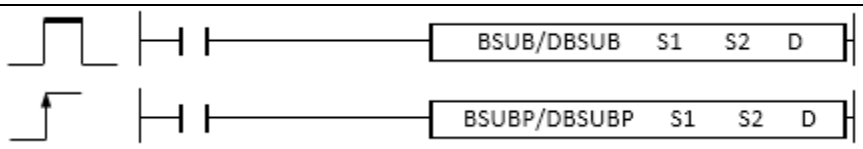
Example)

• If X01 is ON, subtract D2 from D1 and then send the result to M20.



2.2.6. BSUB, BSUBP, DBSUB, DBSUBP (BCD code)

SUB instruction subtracts S2 from S1 and save the result in the Destination (assigned device address).

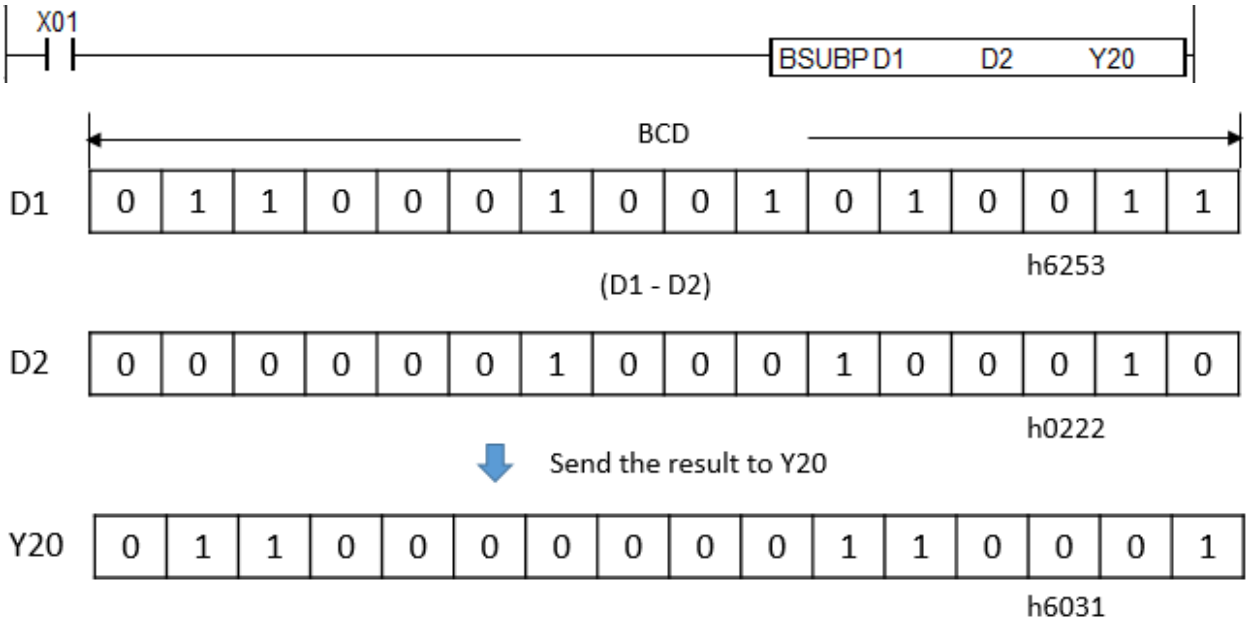


S1, S2	BDC value Source
D	Destination

Instruction		Device address													No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
BSUB(P) DBSUB(P)	S1	o	o	o	o	o	o	o	o	-	o	o	o	o	4	o	-	-
	S2	o	o	o	o	o	o	o	o	-	o	o	o	o				
	D	o	-	o	o	o	-	o	o	-	o	o	o	-				

Example)

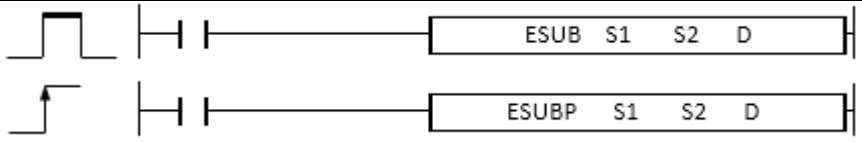
• If X01 is ON, subtract D2 from D1 and then send the result to Y20.



2.2.7. ESUB, ESUBP (Float)

SUB instruction subtracts S2 from S1 and save the result in the Destination (assigned device address).

(It is supported only in XP and PLCS CPU series)

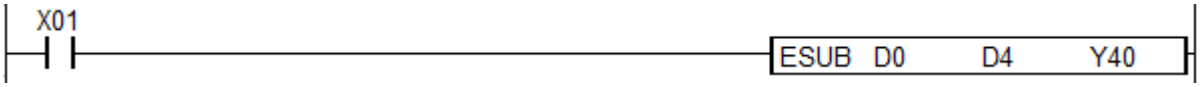


S1, S2	Float value
D	Destination

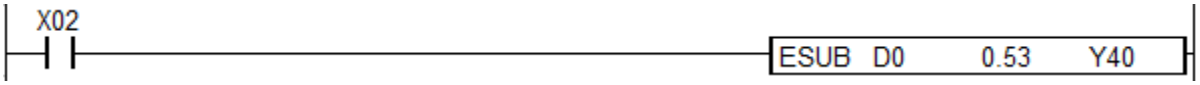
Instruction		Device address													No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
ESUB(P)	S1	o	o	o	o	o	-	-	-	-	o	o	o	-	4	o	-	-
	S2	o	o	o	o	o	-	-	-	-	o	o	o	-				
	D	o	-	o	o	o	-	-	-	-	o	o	o	-				

Example)

• If X01 is ON, subtract D4 from D0 and then send the result to Y40.

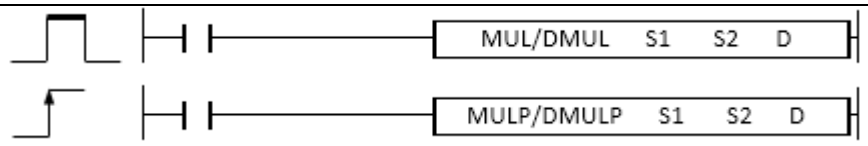


• If X02 is ON, subtract 0.53 from D0 and then send the result to Y40.



2.2.8. MUL, MULP, DMUL, DMULP (Binary)

MUL instruction multiplies S1 with S2 and save the result in Destination (assigned device address).



S1, S2	Source
D	Destination

Instruction		Device address													No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
MUL(P) DMUL(P)	S1	o	o	o	o	o	o	o	o	-	o	o	o	o	4	o	-	-
	S2	o	o	o	o	o	o	o	o	-	o	o	o	o				
	D	o	-	o	o	o	-	o	o	-	o	o	o	-				

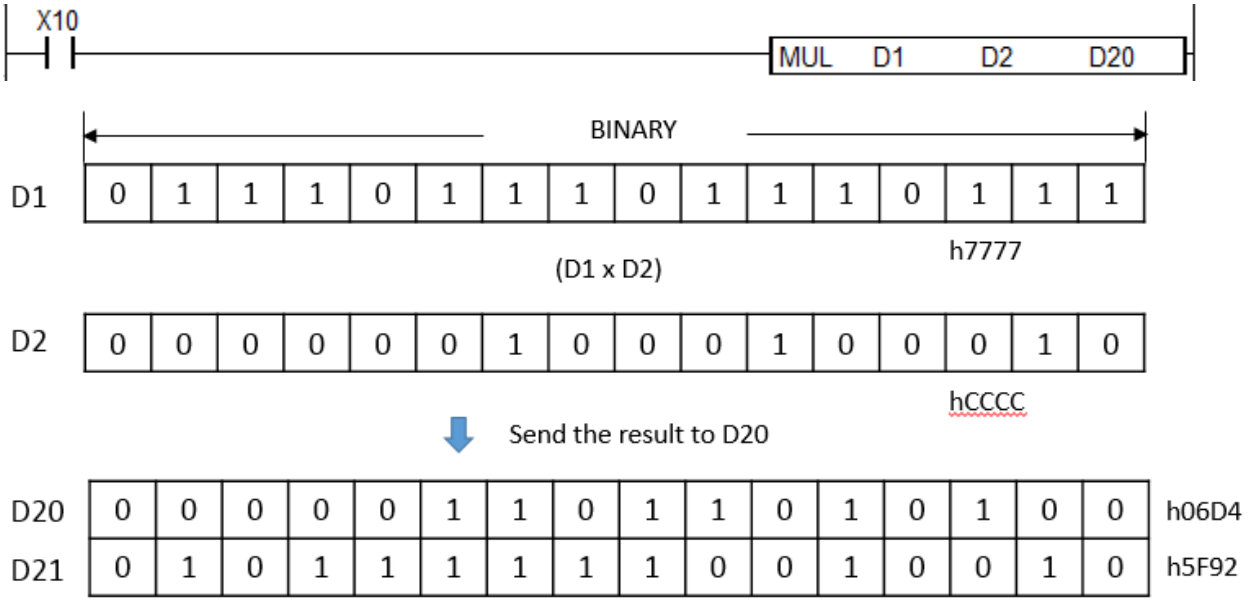
1) MUL

Function

- Multiply S1 with S2 and save the result as double word type to D (Low byte) and D+1 (High byte).

Example)

• If X10 is ON, multiply H777(D1) with HCCC(D2) and save low byte(h06D4) to D20 and save high byte(h5F92) to D21.



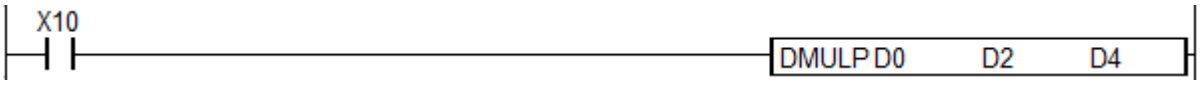
2) DMUL

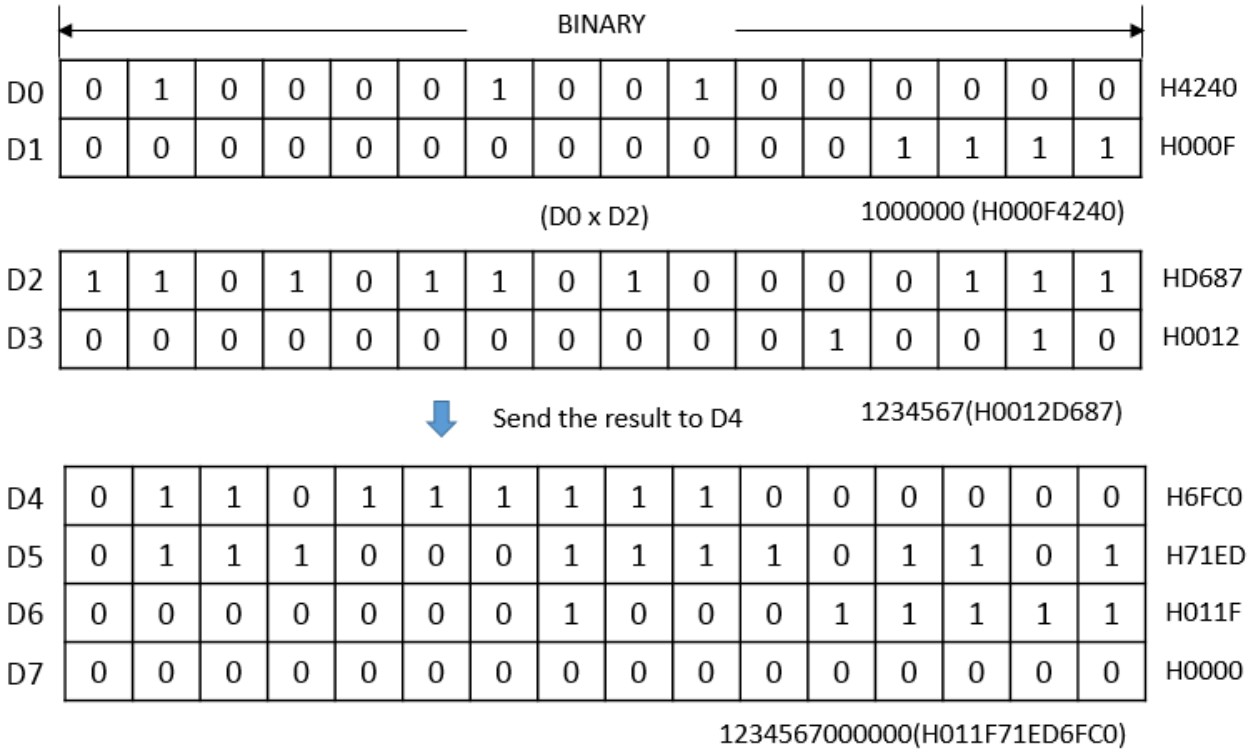
Function

- Multiply S1(Double word) with S2(Double word) and save the result as 4 word type to D, D+1, D+2, and D+3 in sequence.

Example)

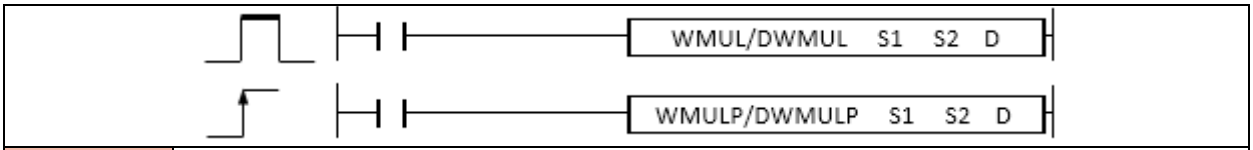
• If X10 is ON, multiply H000F4240(D0) with H0012D687(D2) and then save H6FC0 to D4, H71ED to D5, H011F to D6, and H0000 to D7 in sequence.





2.2.9. WMUL, WMULP, DWMUL, DWMULP (Binary)

WMUL instruction multiplies S1 with S2 and save the result in Destination (assigned device address).



S1, S2	Source
D	Destination

Instruction		Device address													No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
WMUL(P) DWMUL(P)	S1	o	o	o	o	o	o	o	o	-	o	o	o	o	4	o	-	-
	S2	o	o	o	o	o	o	o	o	-	o	o	o	o				
	D	o	-	o	o	o	-	o	o	-	o	o	o	-				

1) WMUL

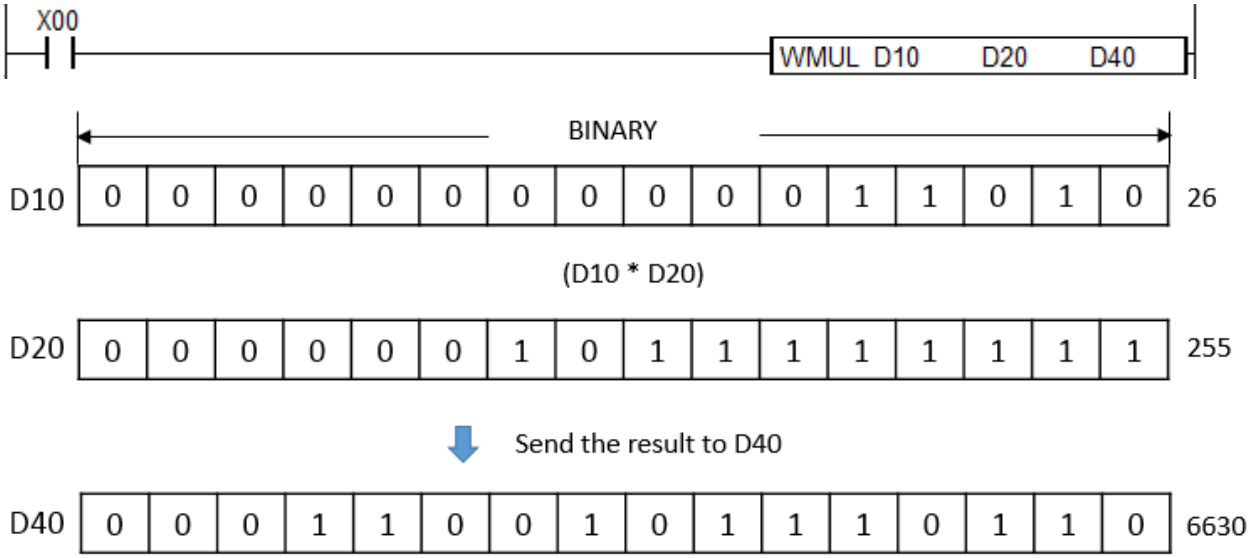
Function

- Multiply S1 with S2 and save the result as word type to D.

- If the value is out of range -32,768 ~ 32,767, multiply instruction will not work and set error flag (F110).

Example)

• If X00 is ON, multiply 26(D10) with 255(D20) and save 6630 to D40.



2) DWMUL

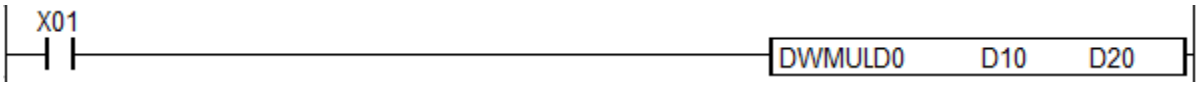
Function

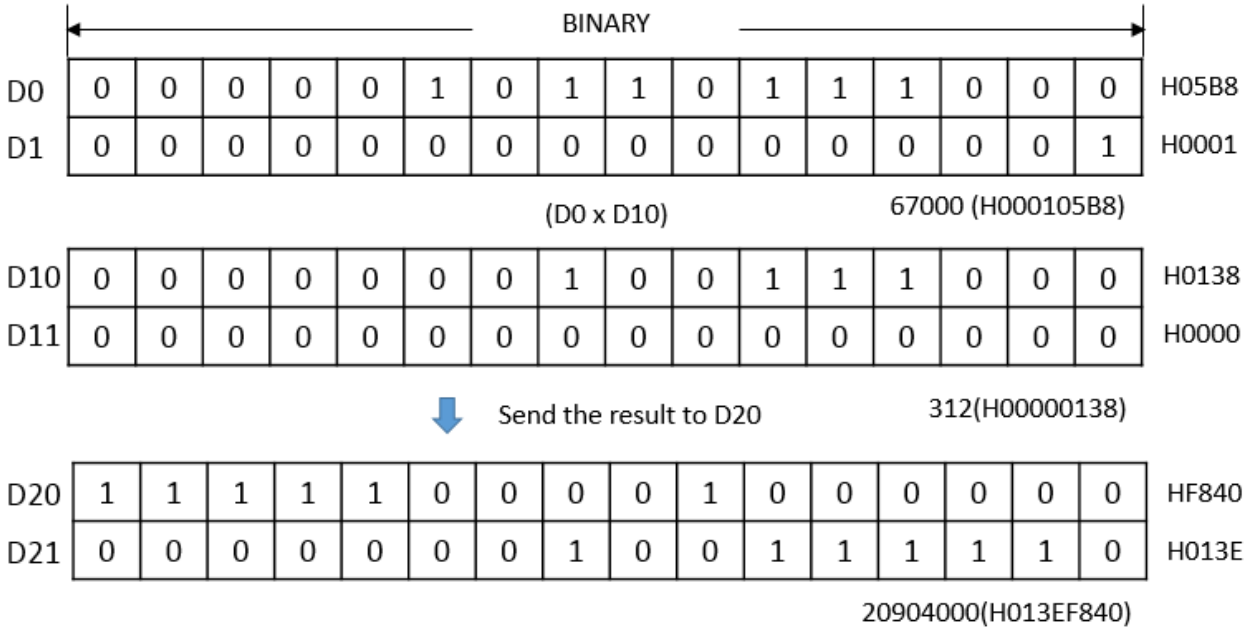
- Multiply S1(Double word) with S2(Double word) and save the result as double word type to D and D+1.

- If the value is out of range -2,147,483,648 ~ 2,147,483,647, multiply instruction will not work and set error flag (F110).

Example)

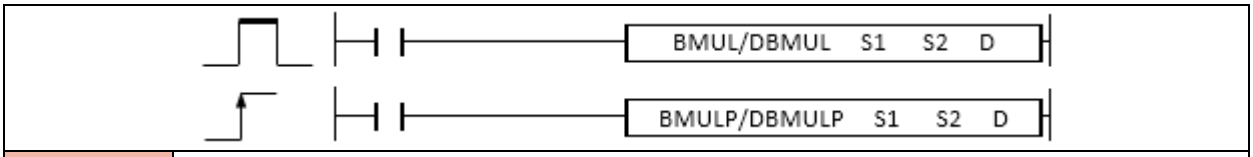
• If X01 is ON, multiply 67000(D0) with 312(D10) and then save HF840 to D20 and H013E to D21 in sequence.





2.2.10. BMUL, BMULP, DBMUL, DBMULP (BCD)

BMUL instruction multiplies S1 with S2 and save the result in Destination (assigned device address).



S1, S2	Source (BCD value)
D	Destination

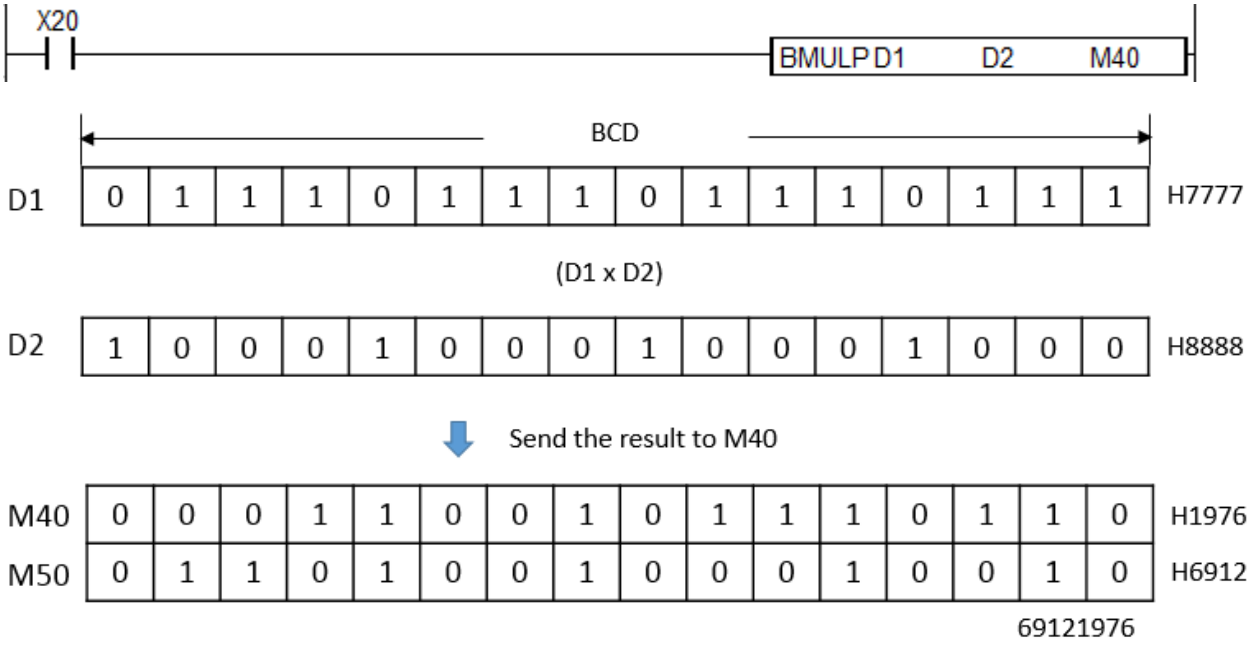
Instruction		Device address													No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
BMUL(P) DBMUL(P)	S1	o	o	o	o	o	o	o	o	-	o	o	o	o	4	o	-	-
	S2	o	o	o	o	o	o	o	o	-	o	o	o	o				
	D	o	-	o	o	o	-	o	o	-	o	o	o	-				

Function

- Multiply S1 with S2 and save the result as double word type to D (Low byte) and D+1 (High byte).

Example)

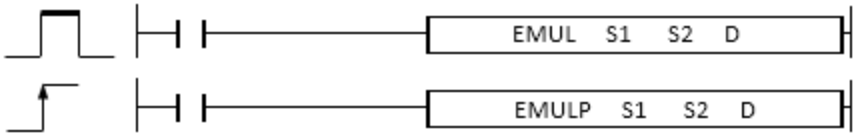
• If X20 is ON, multiply 7777(D1) with 8888(D2) and save 1976 to M40 and 6912 to M50.



2.2.11. EMUL, EMULP (FLOAT)

EMUL instruction multiplies S1 with S2 and save the result in Destination (assigned device address).

(It is supported only in XP and PLCS CPU series.)



S1, S2	Source (Float value)
D	Destination

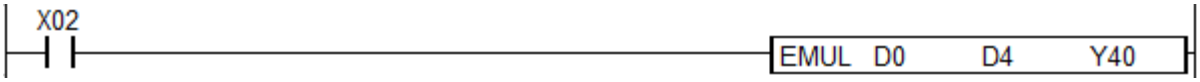
Instruction		Device address													No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
EMUL(P)	S1	o	o	o	o	o	-	-	-	-	o	o	o	-	4	o	-	-
	S2	o	o	o	o	o	-	-	-	-	o	o	o	-				
	D	o	-	o	o	o	-	-	-	-	o	o	o	-				

Function

- Multiply S1 with S2 and save the result to Destination.

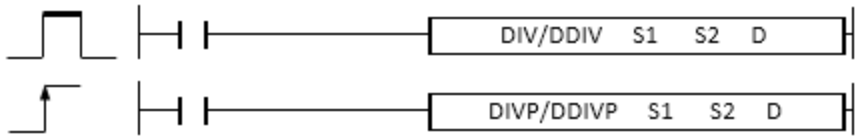
Example)

• If X02 is ON, multiply D0 with D4 and save the result to Y40.



2.2.12. DIV, DIVP, DDIV, DDIVP (Binary)

DIV instruction divides S1 by S2 and save the result in the Destination (assigned device address).



S1	Value of the dividend
S2	Value of the Divisor
D	Destination

Instruction		Device address													No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
DIV(P) DDIV(P)	S1	o	o	o	o	o	o	o	o	-	o	o	o	o	4	o	-	-
	S2	o	o	o	o	o	o	o	o	-	o	o	o	o				
	D	o	-	o	o	o	-	o	o	-	o	o	o	-				

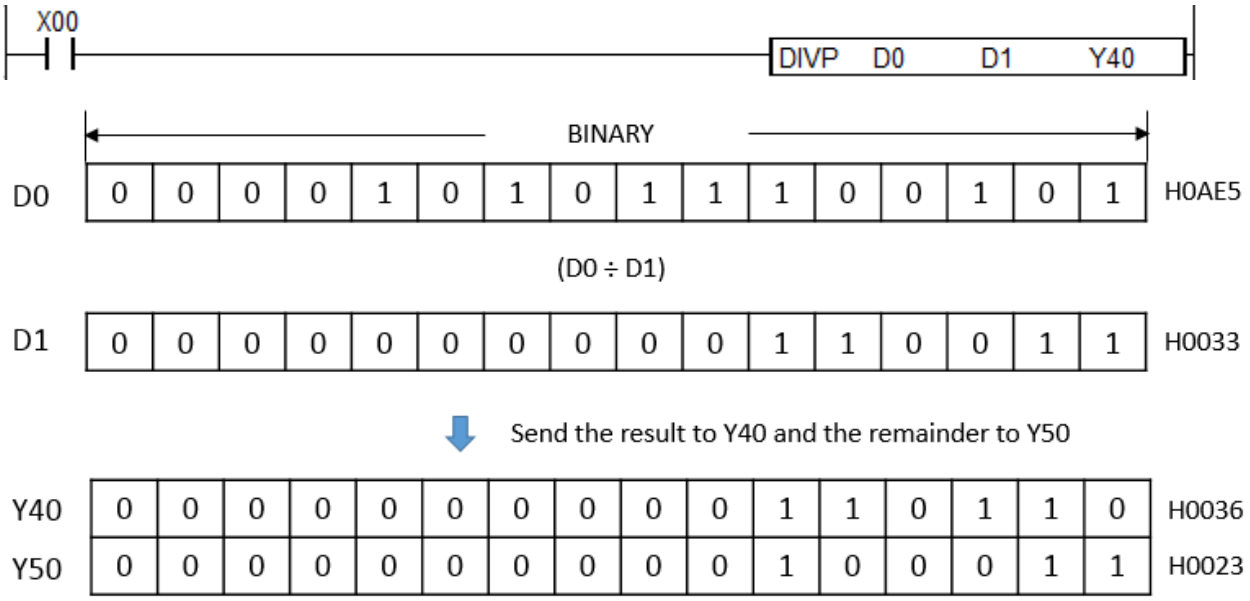
1) DIV

Function

- Divide S1 by S2 and save the result to D and the remainder to D+1

Example)

• If X00 is ON, divide H0AE5(D0) by H0033(D1) and save the result H0036(54) to Y40 and the remainder H0023(35) to Y50.



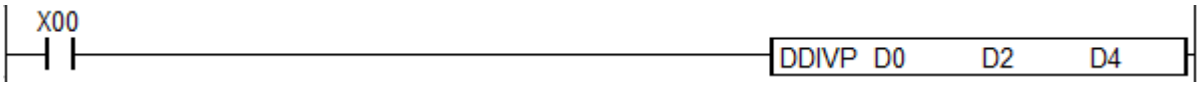
2) DDIV

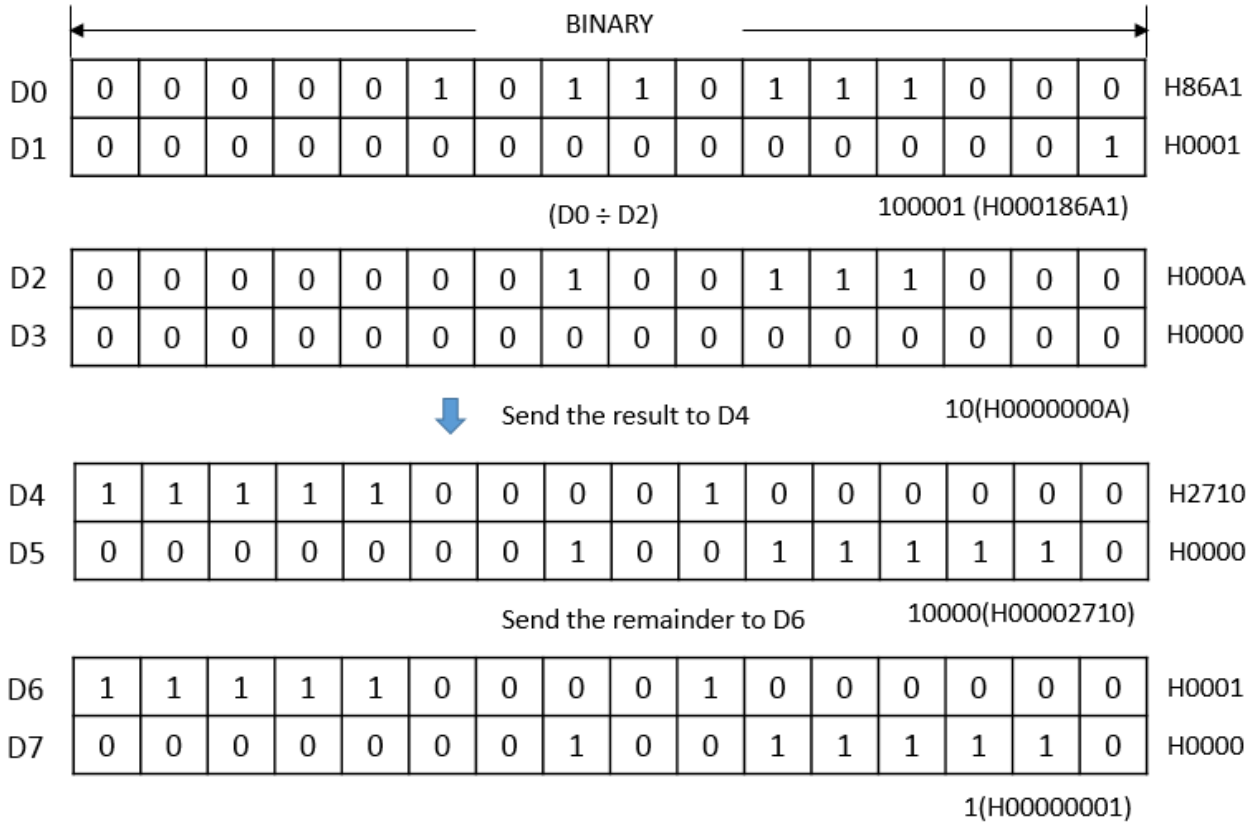
Function

- Divide S1(double word) by S2(double word) and then save the result to D and D+1 and the remainder to D+2 and D+3.

Example)

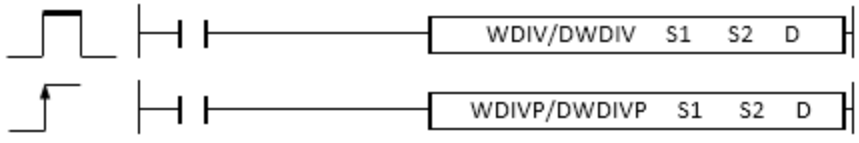
• If X00 is ON, divide H000186A1(D0, D1) by H0000000A(D2, D3) and save the result H00002710(10000) to D4 and D5 and the remainder H00000001(1) to D6 and D7.





2.2.13. WDIV, WDIVP, DWDIV, DWDIVP (Binary)

WDIV instruction divides S1 by S2 and save the result in the Destination (assigned device address) and delete the remainder.



S1	Value of the dividend
S2	Value of the Divisor
D	Destination

Instruction		Device address													No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
WDIV(P) DWDIV(P)	S1	o	o	o	o	o	o	o	o	-	o	o	o	o	4	o	-	-
	S2	o	o	o	o	o	o	o	o	-	o	o	o	o				
	D	o	-	o	o	o	-	o	o	-	o	o	o	-				

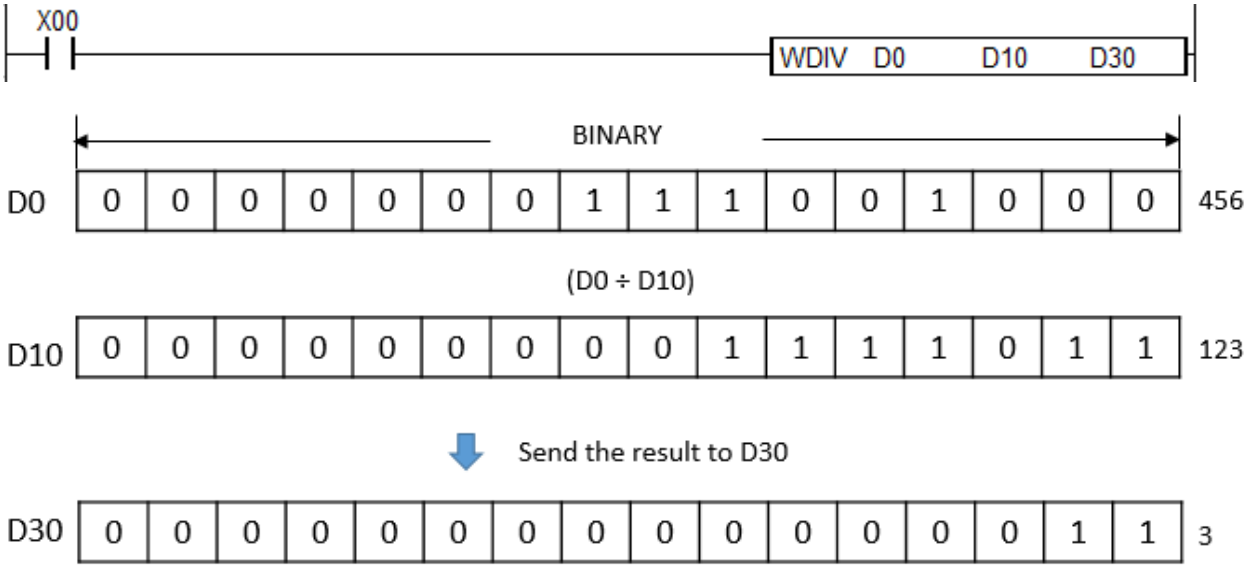
1) WDIV

Function

- Divide S1 by S2 and save the result to D and delete the remainder.

Example)

• If X00 is ON, divide 456(D0) by 123(D10) and save the result 3 to D30 and delete remainder.



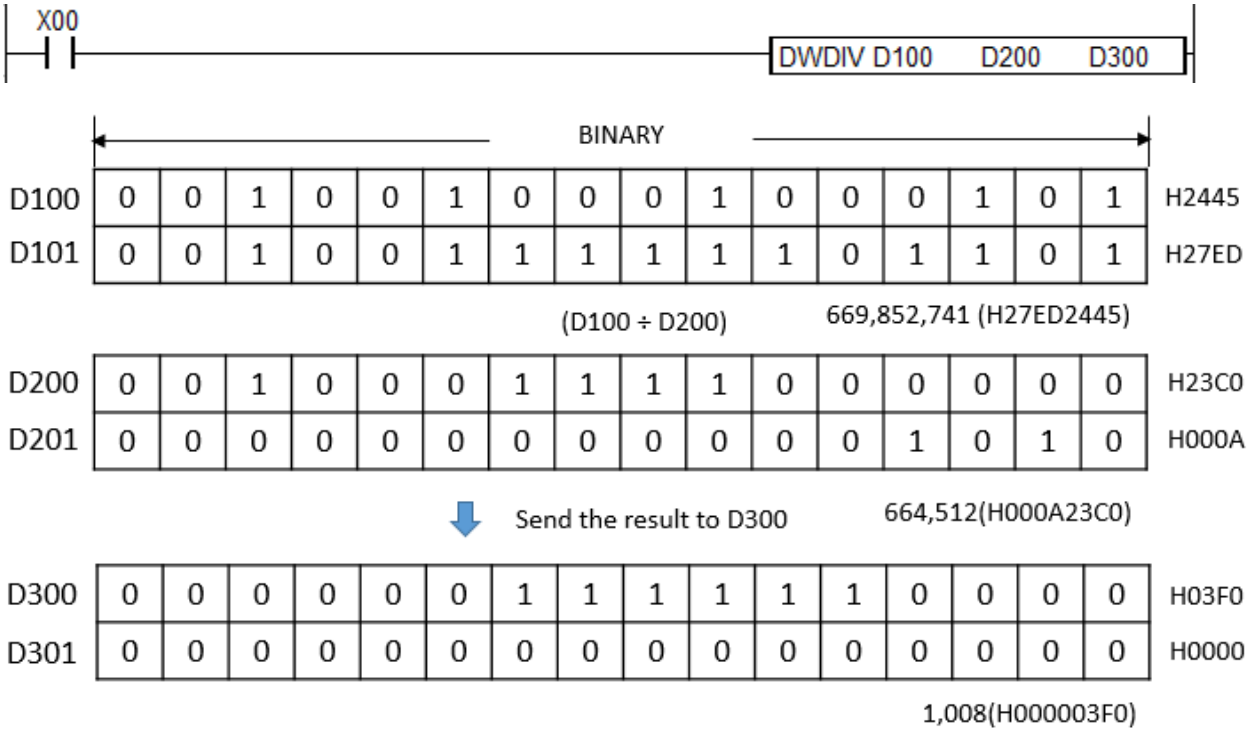
2) DWDIV

Function

- Divide S1(double word) by S2(double word) and then save the result to D and D+1 and delete the remainder.

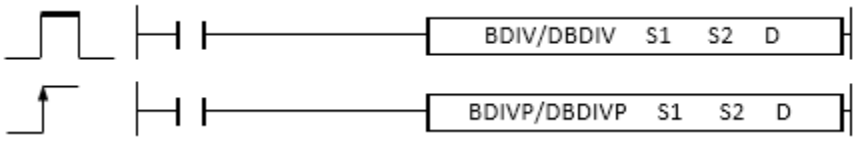
Example)

• If X00 is ON, divide H27ED2445(D100, D101) by H000A23C0(D200, D201) and save the result H000003F0(1,008) to D300 and D301 and delete the remainder.



2.2.14. BDIV, BDIVP, DBDIV, DBDIVP (BCD)

BDIV instruction divides S1 by S2 and save the result in the Destination (assigned device address).



S1	Value of the dividend
S2	Value of the Divisor
D	Destination

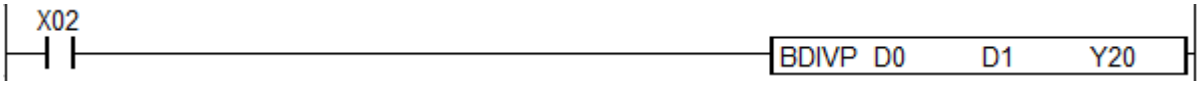
Instruction		Device address													No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
BDIV(P) DBDIV(P)	S1	o	o	o	o	o	o	o	o	-	o	o	o	o	4	o	-	-
	S2	o	o	o	o	o	o	o	o	-	o	o	o	o				
	D	o	-	o	o	o	-	o	o	-	o	o	o	-				

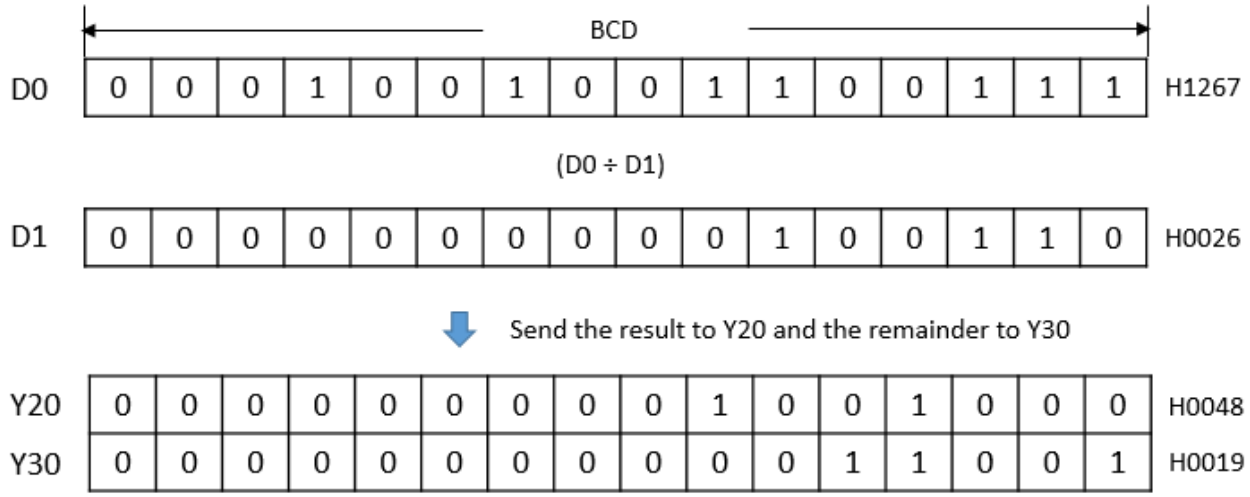
1) BDIV

When enabled, BDIV instruction divides S1(BCD value) by S2(BCD value) and saves the result to D and the remainder to D+1.

Example)

• If X02 is ON, divide 1267(D0) by 26(D1) and save the result H0048 to Y20 and the remainder H0019 to Y30.

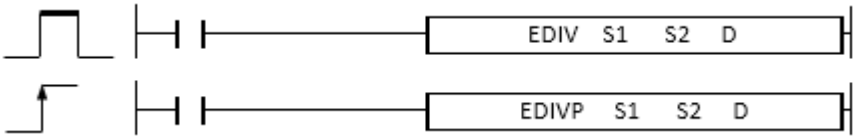




2.2.15. EDIV, EDIVP (FLOAT)

EDIV instruction divides S1 by S2 and save the result in the Destination (assigned device address).

(It is supported only in XP and PLCS CPU series.)



S1	Value of the dividend
S2	Value of the Divisor
D	Destination

Instruction		Device address													No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
EDIV(P)	S1	o	o	o	o	o	-	-	-	-	o	o	o	-	4	o	-	-
	S2	o	o	o	o	o	-	-	-	-	o	o	o	-				
	D	o	-	o	o	o	-	-	o	-	o	o	o	-				

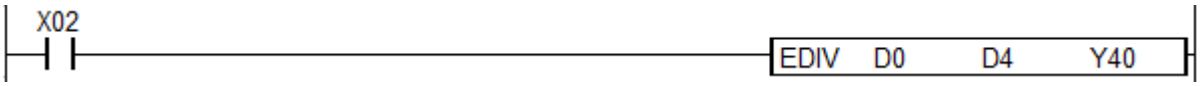
1) EDIV

Function

- Divide S1(Float value) by S2(Float value) and save the result to D and the remainder to D+1.

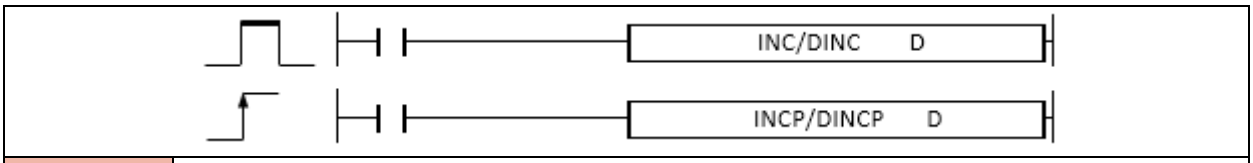
Example)

• If X02 is ON, divide D0 by D4 and save the result to Y40.



2.2.16. INC, INCP, DINC, DINCP (Binary)

INC instruction increments the value of Destination by one.



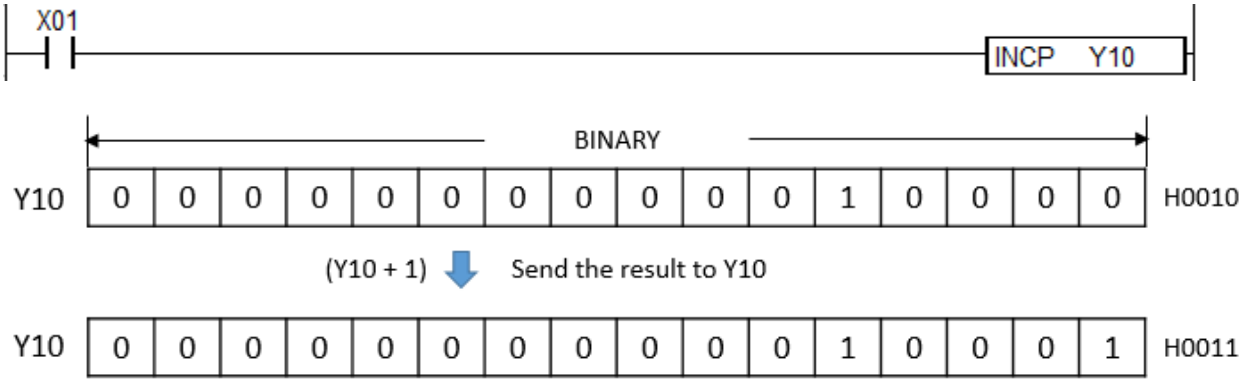
D	Destination

Instruction		Device address													No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
INC(P) DINC(P)	D	o	o	o	o	o	o	o	o	-	o	o	o	-	2	o	-	-

When enabled, INC instruction adds 1 to D and saves the result to D again.

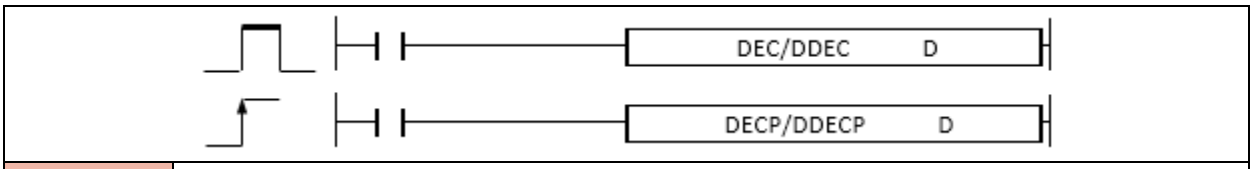
Example)

• If X01 is ON, the value of Y10(H0018) increments by 1.



2.2.17. DEC, DECP, DDEC, DDECP (Binary)

DEC instruction decrements the value of Destination by one.



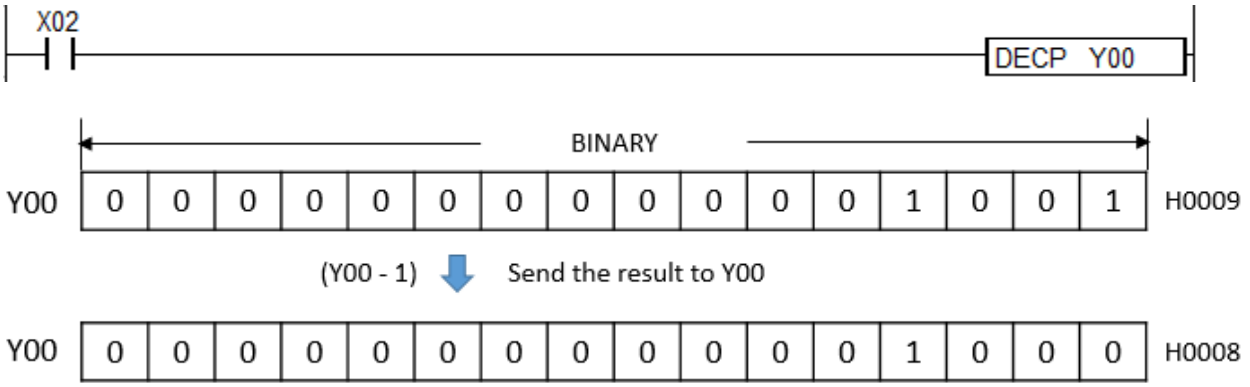
D	Destination

Instruction		Device address													No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
DEC(P) DDEC(P)	D	o	-	o	o	o	-	o	o	-	o	o	o	-	2	o	-	-

When enabled, the DEC instruction Subtracts 1 from D and saves the result to D again.

Example)

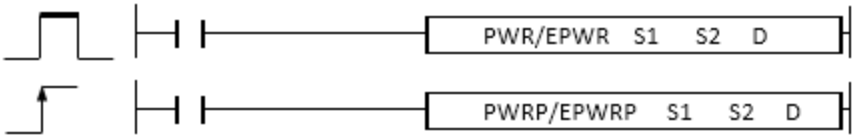
• If X02 is ON, the value of Y00(H0009) decrements by 1.



2.2.18. PWR. PWRP, EPWR. EPWRP

PWR instruction multiplies a base number(S1) by an exponent(S2) and save the result in the Destination (assigned device address).

(It is supported only in XP and PLCS CPU series.)



S1	Base number
S2	Exponent
D	Destination

Instruction		Device address													No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
PWR(P)	S1	o	o	o	o	o	o	o	o	-	o	o	o	o	4	o	-	-
	S2	o	o	o	o	o	o	o	o	-	o	o	o	o				
	D	o	-	o	o	o	-	o	o	-	o	o	o	-				

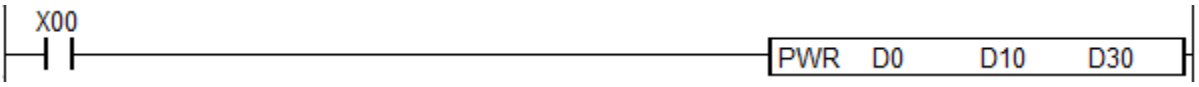
1) PWR/PWRP

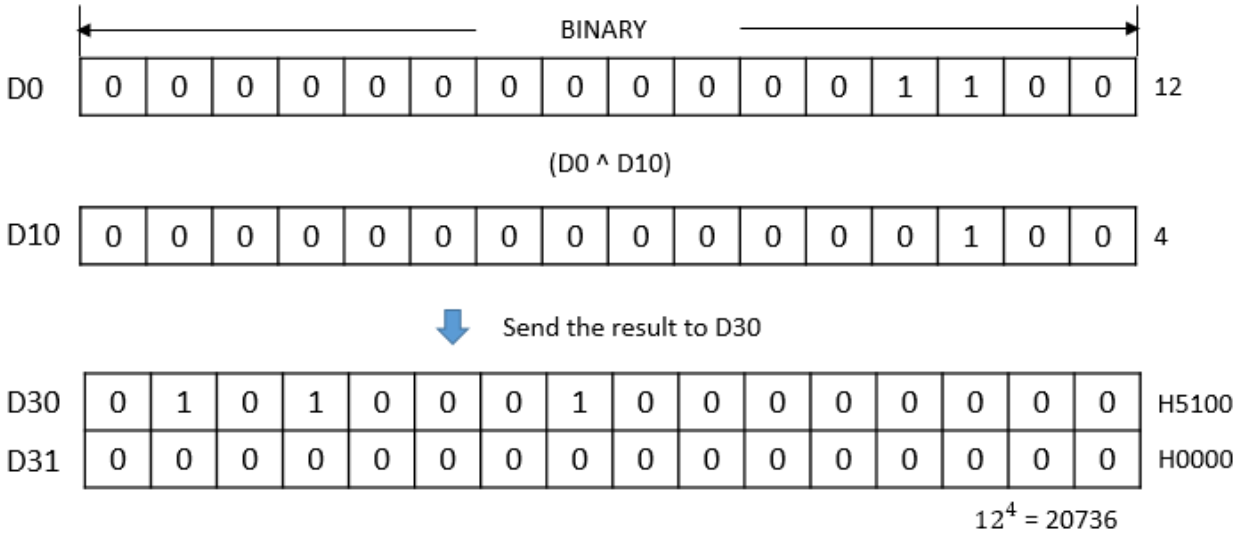
Function

- S1 is being multiplied by itself S2 times and save the result in the Destination.

Example)

• If X00 is ON, D0(12) is being multiplied by itself D10(4)times (12 to the power of fourth) and save the result(20736) in D30





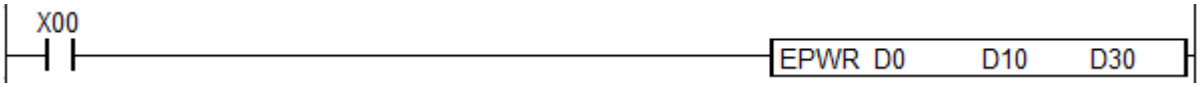
2) EPWR/EPWRP

Function

- S1(Float value) is being multiplied by itself S2 times and save the result in the Destination.

Example)

• If X00 is ON, D0(3.1) is being multiplied by itself D10(3)times (3.1 cubed) and save the result(29.791) in D30

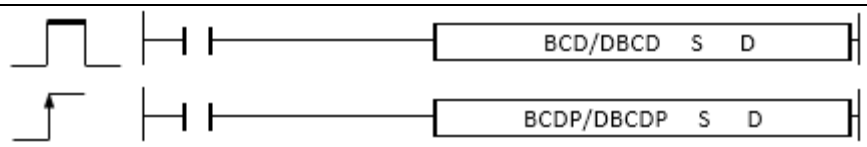


2.3. Data Conversion Instruction

2.3.1. BCD, BCDP, DBCD, DBCDP

BCD instruction converts a source value to a BCD value and save the BCD value in Destination (assigned device address).

BCD is the Binary Coded Decimal number system that expresses individual decimal digits in a 40bit binary notation.



S	Source (Binary data)
D	Destination (BCD value)

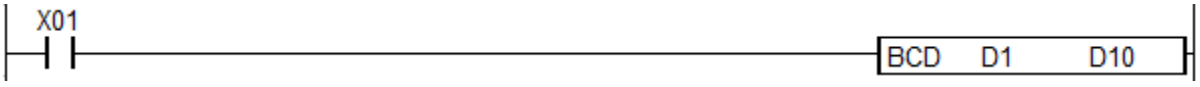
Instruction		Device address													No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
BCD(P) DBCD(P)	S	o	o	o	o	o	o	o	o	-	o	o	o	o	3	o	-	-
	D	o	-	o	o	o	-	o	o	-	o	o	o	-				

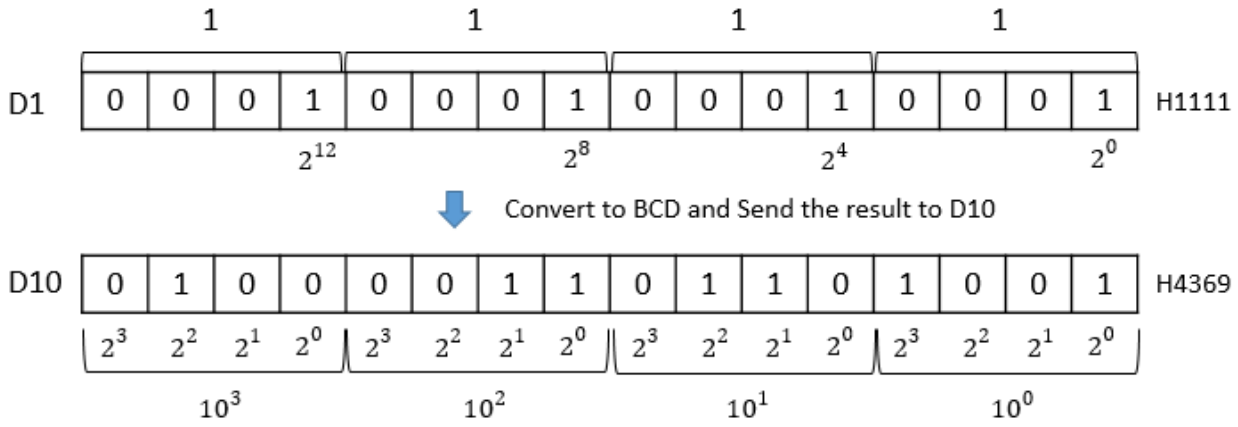
Error(F110) : If BCD Data is out of range, Error Flag(F110) will be set.

Instruction	Binary Data range	
BCD BCDP	16bit	0 ~ H270F 0 ~ 9999
DBCD DBCDP	32bit	0 ~ H05F5E0FF 0 ~ 99999999

Example)

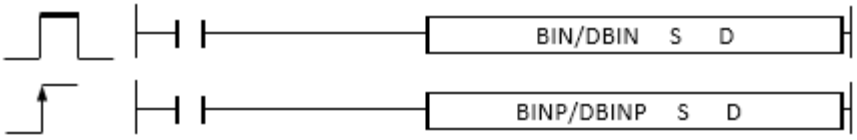
• If X01 is ON, convert D1(H1111) to BCD value(H4369) and save it to D10





2.3.2. BIN, BINP, DBIN, DBINP

BIN instruction converts a BCD value to a Binary data and save the result in Destination (assigned device address).



S	Source (BCD value)
D	Destination (Binary data)

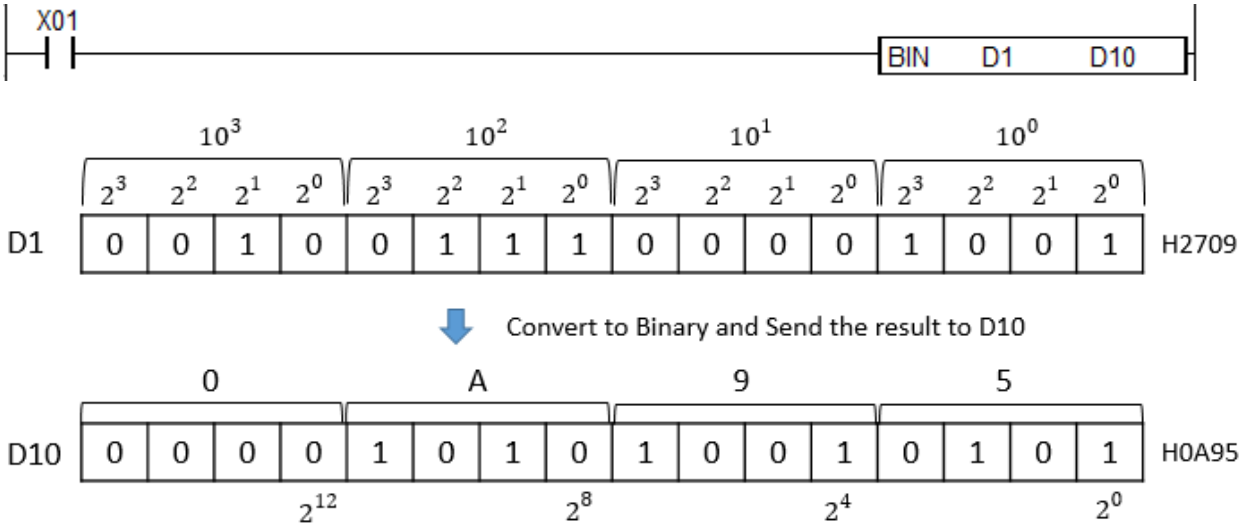
Instruction		Device address													No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
BIN(P)	S	o	o	o	o	o	o	o	o	-	o	o	o	o	3	o	-	-
	D	o	-	o	o	o	-	o	o	-	o	o	o	-				
DBIN(P)	S	o	o	o	o	o	o	o	o	-	o	o	o	o	3			
	D	o	-	o	o	o	-	o	o	-	o	o	o	-				

Error(F110) : If BCD Data is out of range, Error Flag(F110) will be set.

Instruction	Binary Data range	
BIN BINP	16bit	0 ~ H270F 0 ~ 9999
DBIN DBINP	32bit	0 ~ H05F5E0FF 0 ~ 99999999

Example)

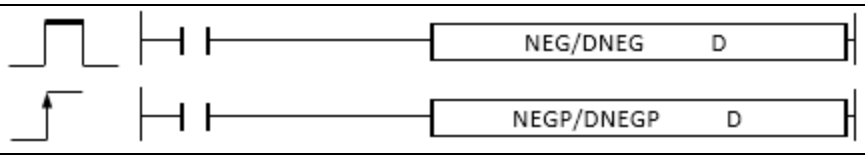
• If X01 is ON, convert BCD value D1(H2709) to Binary value(H0A95) and save it to D10



2.3.3. NEG, NEGP, DNEG, DNEGP

NEG instruction changes the sign of the value and save the result in Destination (assigned device address).

If you negate a negative value, the result is positive. If you negate a positive value, the result is negative.

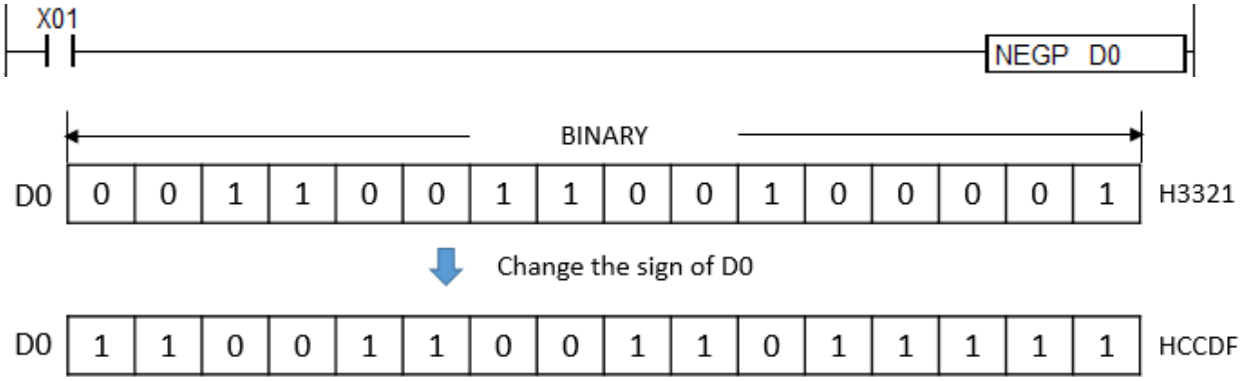


D	Destination

Instruction		Device address													No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
NEG(P) DNEG(P)	D	o	-	o	o	o	-	o	o	-	o	o	o	-	2	o	-	-

Example)

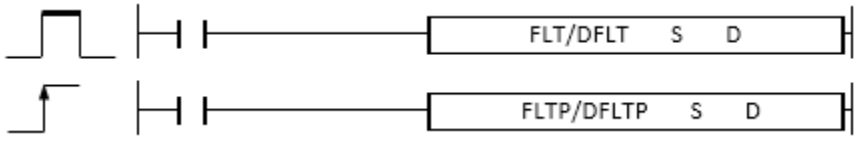
• If X01 is ON, change the sign of DO(H3321) value to minus value and save it in D0 again.



2.3.4. FLT, FLTP, DFLT, DFLTP

FLT instruction converts binary data to float value and save the result in Destination (assigned device address).

(It is supported only in XP and PLC-S CPU series)



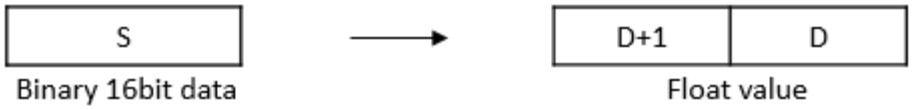
S	Source
D	Destination (Float value)

Instruction		Device address													No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
FLT(P) DFLT(P)	S	o	o	o	o	o	o	o	o	-	o	o	o	o	4	o	-	-
	D	o	-	o	o	o	-	-	-	-	o	o	o	-				

1) FLT(P)

Functions

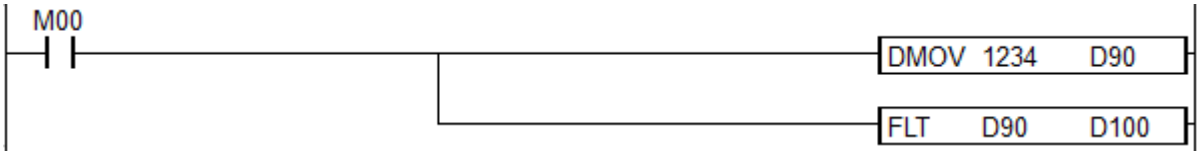
- Convert S(16bit binary data) to float value and save the result in the Destination.



- The range of binary data in S is -32,768 ~ 32,767.

Example)

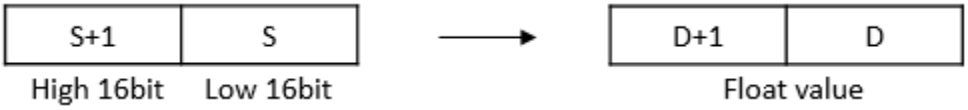
• If M00 is ON, save 1234 to D90 and convert D90 to float value and then save the result in D100.



2) DFLT(P)

Functions

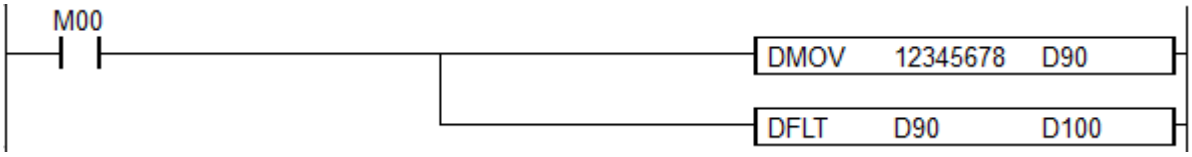
- Convert S(32bit binary data) to float value and save the result in the Destination.



- The range of binary data in S is -2,147,483,648 ~ 2,147,483,647.

Example)

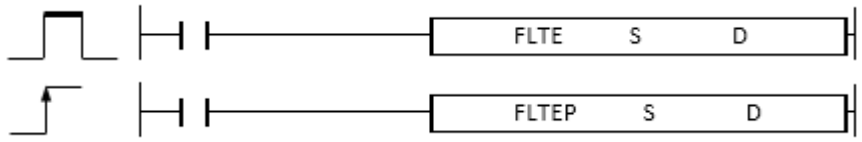
• If M00 is ON, save 12345678 to D90 and convert D90 to float value and then save the result in D100.



2.3.5. FLTE, FLTEP

FLTE instruction converts 64bit float value to 32bit float value and save the result in Destination.

(It is supported only in XP and PLC-S CPU series)



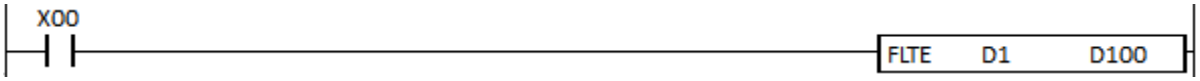
S	Source (64bit Float value
D	Destination (32bit Float value)

Instruction		Device address													No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
FLTE FLTEP	S	o	o	o	o	o	o	o	o	-	o	o	o	o	4	o	-	-
	D	o	-	o	o	o	-	-	-	-	o	o	o	-				

- Convert S(64bit float value) to 32bit float value and save the result in the Destination (2 word).

Example)

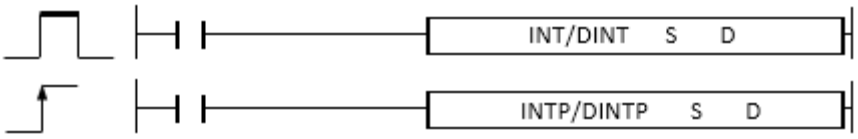
• If X00 is ON, convert D1 (64bit float data) to 32 bit float value and then save the result in D100 (2 word).



2.3.6. INT, INTP, DINT, DINTP

INT instruction converts a float value to an integer value and save the result in Destination (assigned device address).

(It is supported only in XP and PLC-S CPU series)



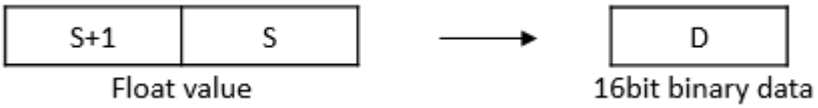
S	Source (Float value)
D	Destination

Instruction		Device address													No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
INT(P) DINT(P)	S	o	o	o	o	o	-	-	-	-	o	o	o	-	4	o	-	-
	D	o	-	o	o	o	-	o	o	-	o	o	o	o				

1) INT(P)

Functions

- Convert S(Float value) to Integer(16bit binary data) and save the result in the Destination.

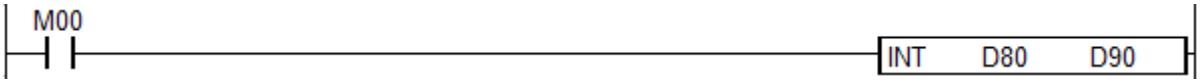


- The range of float in S is -32,768 ~ 32,767.

- The result is rounded off to zero decimal place.

Example)

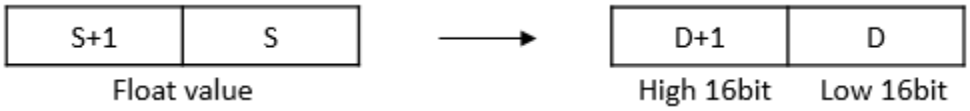
• If M00 is ON, convert D80 (float value) to integer (16bit binary) and then save the result in D90.



2) DINT(P)

Functions

- Convert S(Float value) to Integer(32bit binary data) and save the result in the Destination.

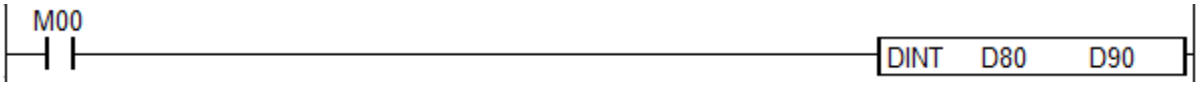


- The range of float in S is -2,147,483,648 ~ 2,147,483,647.

- The result is rounded off to zero decimal place.

Example)

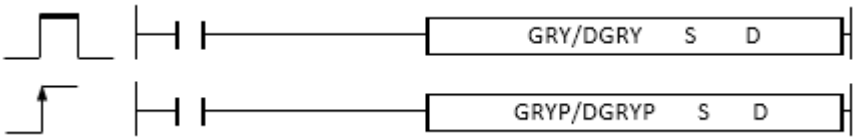
• If M00 is ON, convert D80 (float value) to integer (32bit binary) and then save the result in D90.



2.3.7. GRY, GRYP, DGRY, DGRYP

GRY instruction converts a binary data to a Gray code and save the result in the Destination (assigned device address).

(It is supported only in XP and PLC-S CPU series)



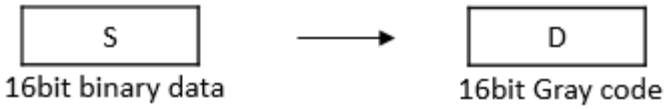
S	Source
D	Destination

Instruction		Device address													No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
GRY(P) DGRY(P)	S	o	o	o	o	o	o	o	o	-	o	o	o	o	4	o	-	-
	D	o	-	o	o	o	-	o	o	-	o	o	o	o				

1) GRY(P)

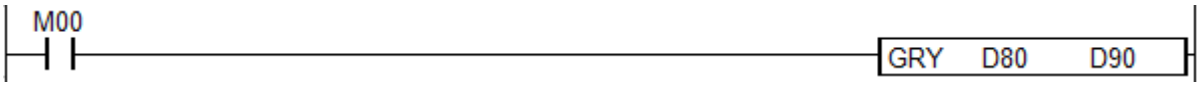
Functions

- Convert S(16bit binary data) to 16bit gray code data and save the result in the Destination.



Example)

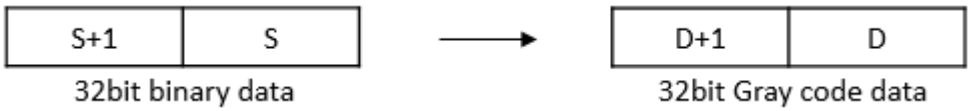
• If M00 is ON, convert D80 (16bit binary data) to 16bit Gray code and then save the result in D90.



2) DGRY(P)

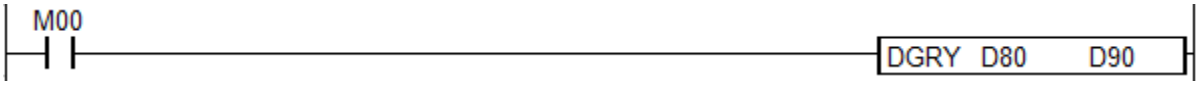
Functions

- Convert S(32bit binary data) to 32bit Gray code and save the result in Destination.



Example)

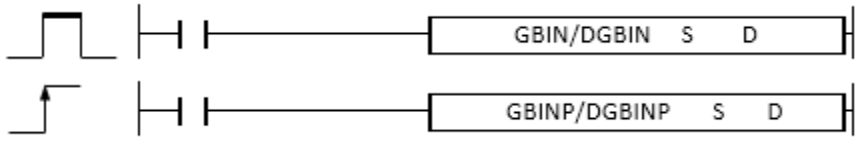
• If M00 is ON, convert D80 (32bit binary data) to 32bit Gray code and then save the result in D90.



2.3.8. GBIN, GBINP, DGBIN, DGBINP

GBIN instruction converts a Gray code to a Binary data and save the result in the Destination (assigned device address).

(It is supported only in XP and PLC-S CPU series)



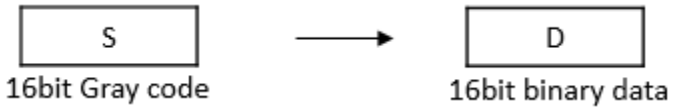
S	Source (Gray code)
D	Destination

Instruction		Device address													No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
GBIN(P) DGBIN(P)	S	o	o	o	o	o	o	o	o	-	o	o	o	o	4	o	-	-
	D	o	-	o	o	o	-	o	o	-	o	o	o	o				

1) GBIN(P)

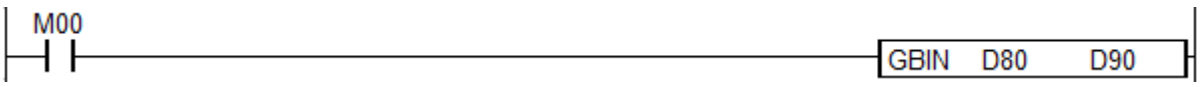
Functions

- Convert S(16bit Gray data) to 16bit binary code data and save the result in the Destination.



Example)

• If M00 is ON, convert D80 (16bit Gray code data) to 16bit binary data and then save the result in D90.



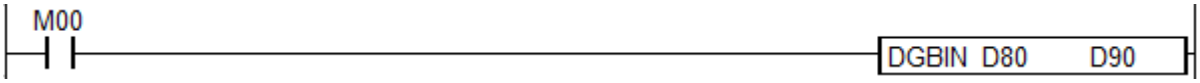
2) DGBIN(P)

Functions

- Convert S(32bit Gray code) to 32bit binary data and save the result in the Destination.

Example)

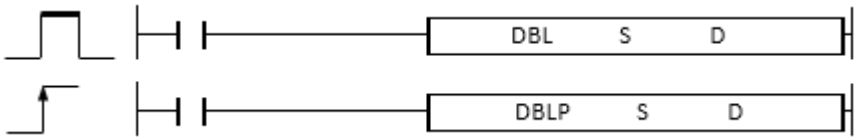
• If M00 is ON, convert D80 (32bit Gray code) to 32bit binary data and then save the result in D90.



2.3.9. DBL, DBLP

DBL instruction converts 2 word BCD source to 64bit float value and saves the result in Destination.

(It is supported only in XP and PLC-S CPU series)



S	Source (2 word BCD data)
D	Destination (Float value)

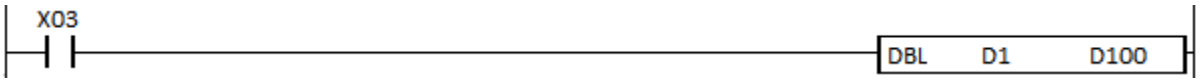
Instruction		Device address													No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
DBL DBLP	S	o	-	o	o	o	-	-	-	-	o	o	o	o	3	o	-	-
	D	o	o	o	o	o	o	o	o	o	o	o	o	o				

Error(F110) : If BCD Data is out of range, Error Flag(F110) will be set.

Instruction	Binary Data range	
DBL DBLP	32bit	0 ~ H05F5E0FF 0 ~ 99999999

Example)

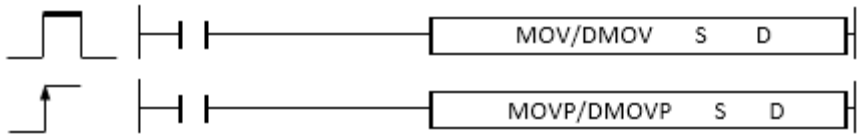
• If X03 is ON, the DBL instruction converts D1 (2word) to 64 bit Float value and saves it to D100



2.4. Data Transfer Instruction

2.4.1. MOV, MOVP, DMOV, DMOVP

MOV instruction copies the source to the Destination (assigned device address).



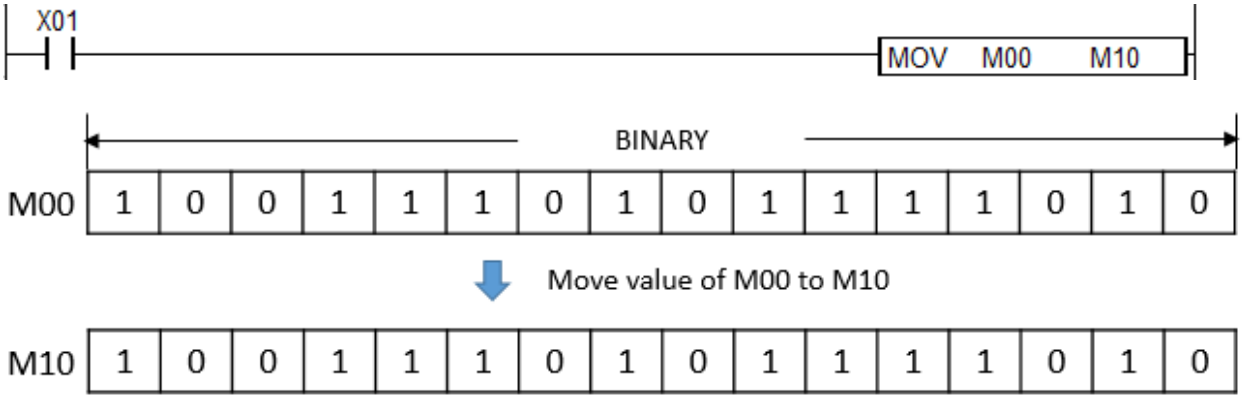
S	Source
D	Destination

Instruction		Device address													No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
MOV(P)	S	o	o	o	o	o	o	o	o	-	o	o	o	o	3	o	-	-
	D	o	-	o	o	o	-	o	o	-	o	o	o	-				
DMOV(P)	S	o	o	o	o	o	o	o	o	-	o	o	o	o	3			
	D	o	-	o	o	o	-	o	o	-	o	o	o	-				

- The MOV(P) instruction copies 16 bit data and the DMOV(P) copies 32 bit data to the Destination.

Example)

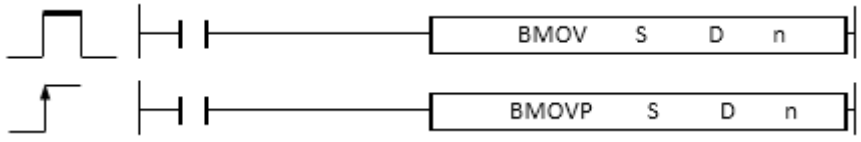
• If X01 is ON, the MOV instruction copies the data of M00 to the M10.



2.4.2. BMOV, BMOVP

BMOV instruction copies a group of words from the source to the Destination (assigned device address).

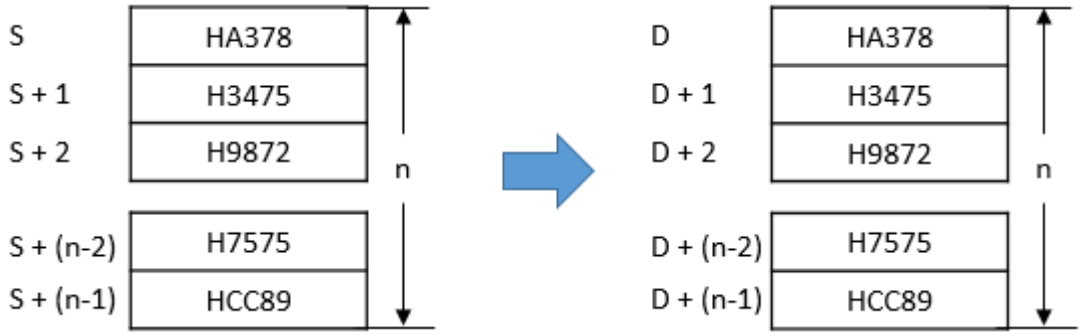
Block (Group of words) MOVE.



S	Source
D	Destination
n	number of Words to copy

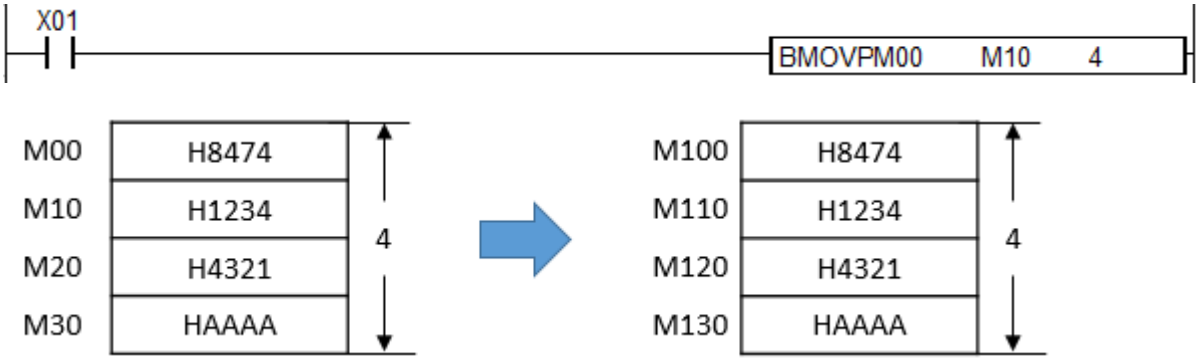
Instruction		Device address													No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
BMOV(P)	S	o	o	o	o	o	o	o	o	-	o	o	o	-	4	o	-	-
	D	o	-	o	o	o	-	o	o	-	o	o	o	-				
	n	o	o	o	o	o	o	o	o	-	o	o	o	o				

- BMOV(P) copies (n) number of words from the source(S) and writes the words into the Destination(D)



Example)

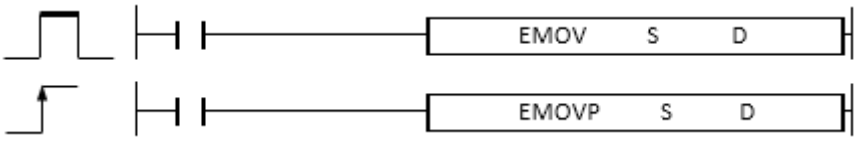
• If X01 is ON, copy 4 words (M00 to M30) and save them to the M100, M110, M120, and M130.



2.4.3. EMOV, EMOVP

EMOV instruction copies the source (float type) to the Destination (assigned device address).

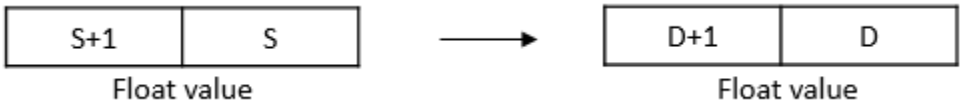
(It is supported only in XP and PLCS CPU series.)



S	Source
D	Destination

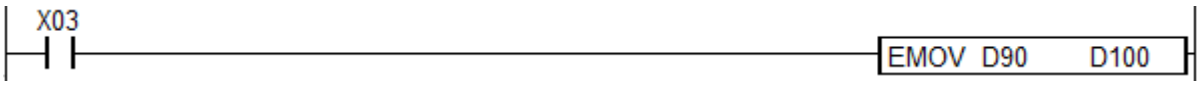
Instruction		Device address													No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
EMOV(P)	S	o	o	o	o	o	-	-	-	-	o	o	o	-	4	o	-	-
	D	o	-	o	o	o	-	-	-	-	o	o	o	-				

- The EMOV(P) instruction copies float type value and writes the value into the Destination(D)



Example)

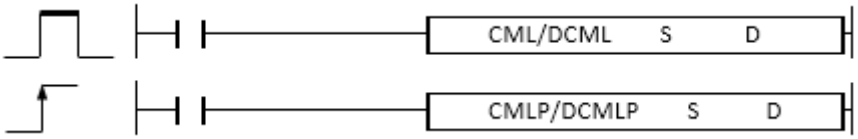
• If X03 is ON, the EMOV instruction copies float value of D90 to D100.



2.4.4. CML, CMLP, DCML, DCMLP

CML instruction negates the value of source and moves it to the Destination (assigned device address).

If you negate 0, the result is 1. If you negate 1, the result is 0.

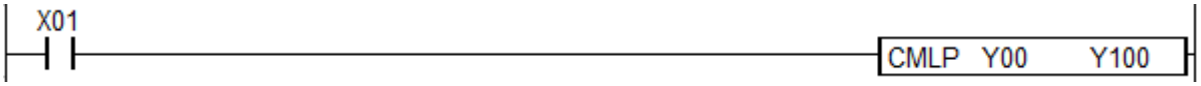


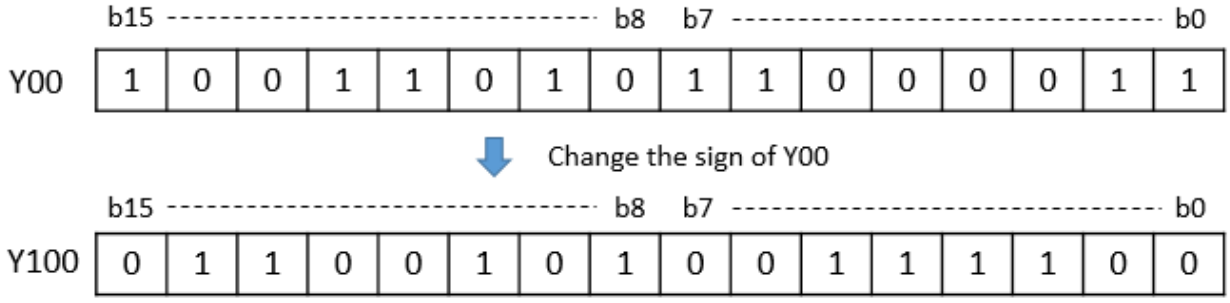
S	Source
D	Destination

Instruction		Device address													No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
CML(P) DCML(P)	S	o	o	o	o	o	o	o	o	-	o	o	o	o	3	o	-	-
	D	o	-	o	o	o	-	o	o	-	o	o	o	-				

Example)

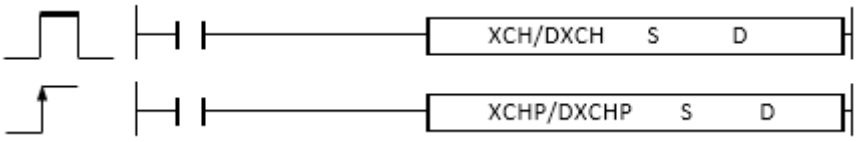
• If X01 is ON, the CMLP instruction negates the value of Y00 value and saves the result in Y100.





2.4.5. XCH, XCHP, DXCH, DXCHP

XCH instruction exchanges between the data of source 1 and source 2.

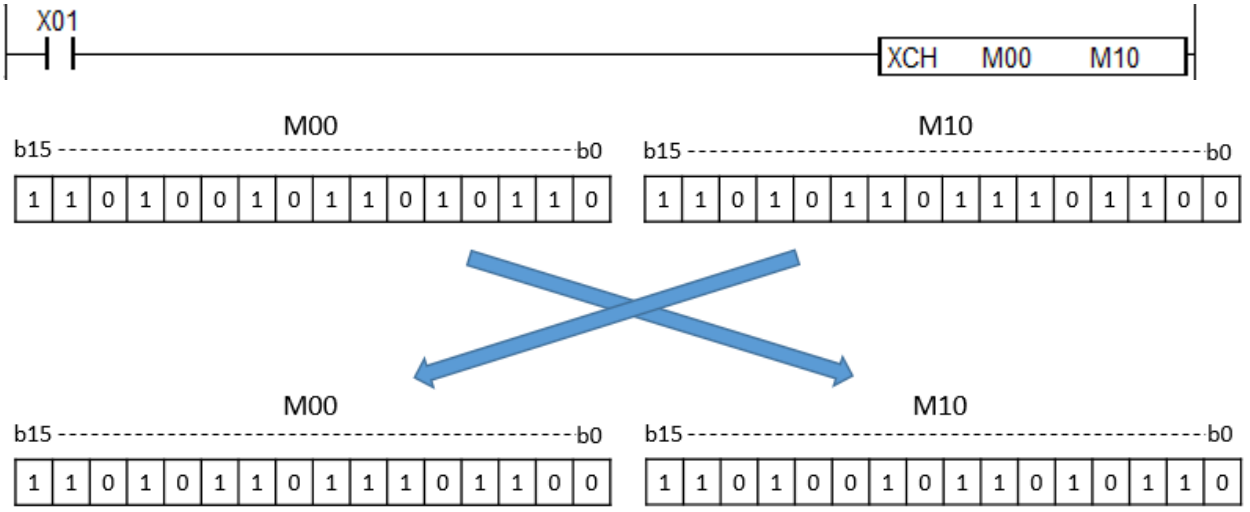


S	Source 1
D	Source 2

Instruction		Device address													No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
XCH(P) DXCH(P)	S	o	-	o	o	o	-	o	o	-	o	o	o	-	3	o	-	-
	D	o	-	o	o	o	-	o	o	-	o	o	o	-				

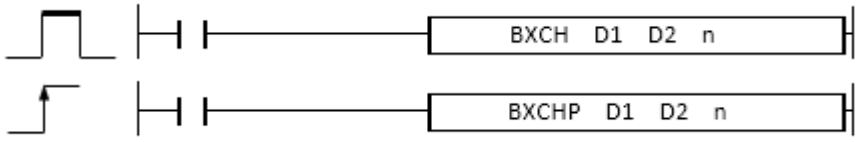
Example)

• If X01 is ON, the XCH instruction exchanges between M00 and M10.



2.4.6. BXCH, BXCHP

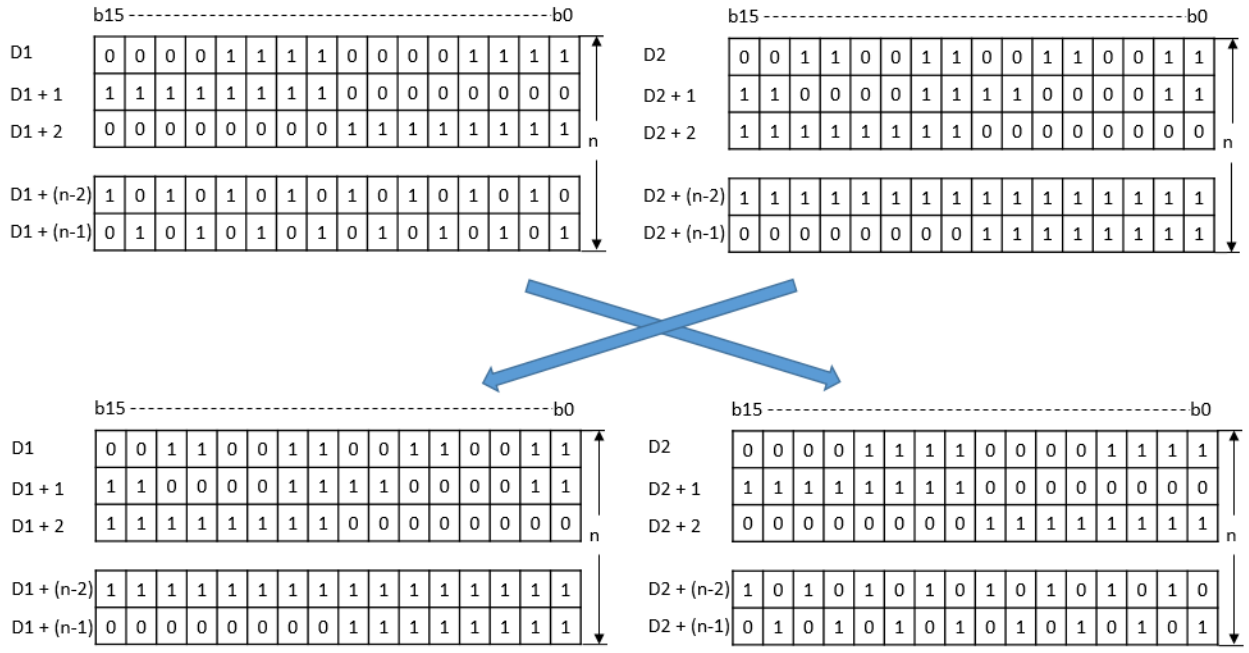
XCH instruction exchanges between the specified data of source 1 and source 2.



D1,D2	Source 1 and Source 2
n	Number of source

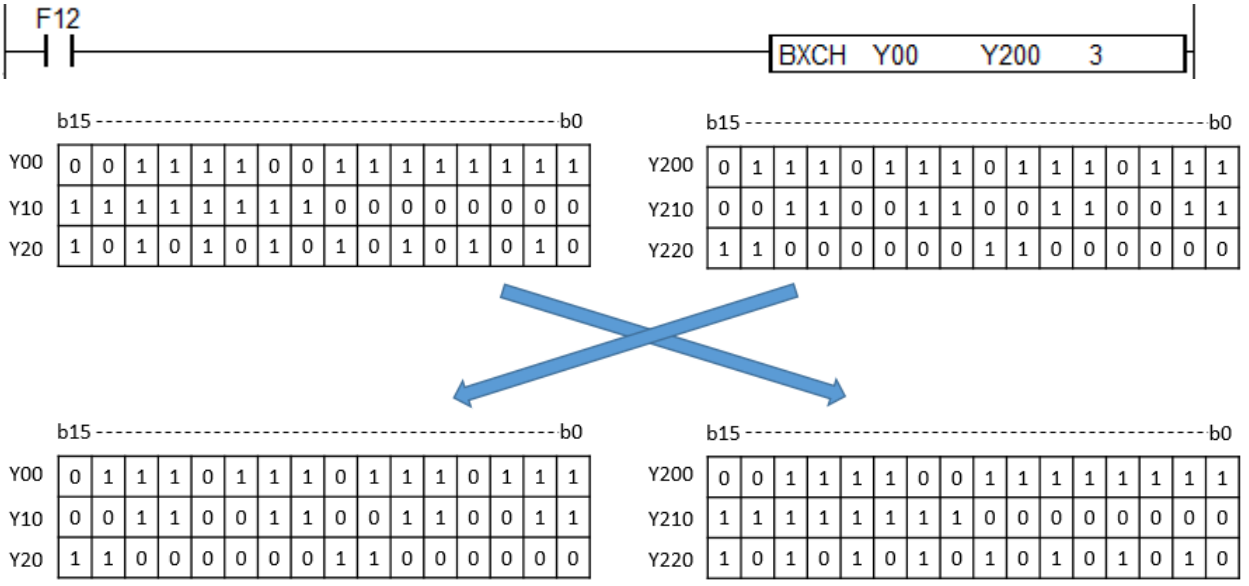
Instruction		Device address													No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
BXCH BXCHP	D1	o	-	o	o	o	-	o	o	-	o	o	o	-	4	o	-	-
	D2	o	-	o	o	o	-	o	o	-	o	o	o	-				
	n	o	o	o	o	o	o	o	o	-	o	o	o	o				

- The BXCH instruction exchanges (n)number of words from D1 to the D2.



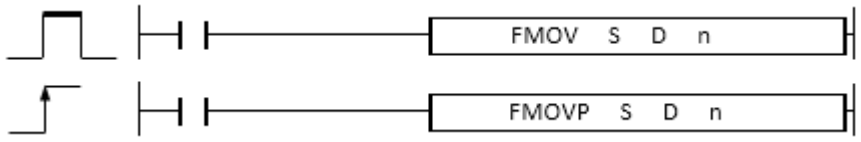
Example)

• If F12 is ON, the BXCH instruction exchanges between 3 bytes of Y00 (Y00, Y10 and Y20) and 3 bytes of Y200 (Y200, Y210 and Y220)



2.4.7. FMOV, FMOVP

FMOV instruction copies the source and writes it in the specified Destination.



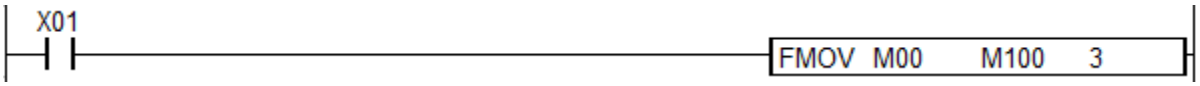
S	Source
D	Destination
n	Number of Destination

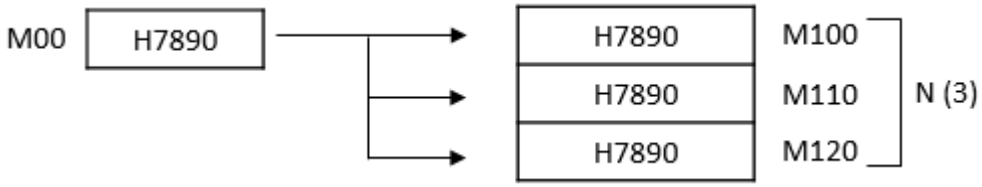
Instruction		Device address													No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
FMOV FMOVP	S	o	o	o	o	o	o	o	o	-	o	o	o	o	4	o	-	-
	D	o	-	o	o	o	-	o	o	-	o	o	o	-				
	n	o	o	o	o	o	o	o	o	-	o	o	o	o				

- FMOV instruction copies 16 bit data and writes it in (n) number of Destination.

Example)

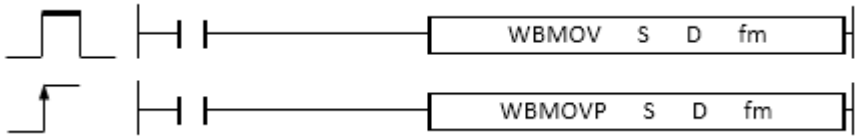
• If X01 is ON, the RMOV instruction copies M00 (H7890) and writes it in M100, M110 and M120.





2.4.8. WBMOV, WBMOVP

WBMOV instruction copies the specified bits from the source to the Destination.

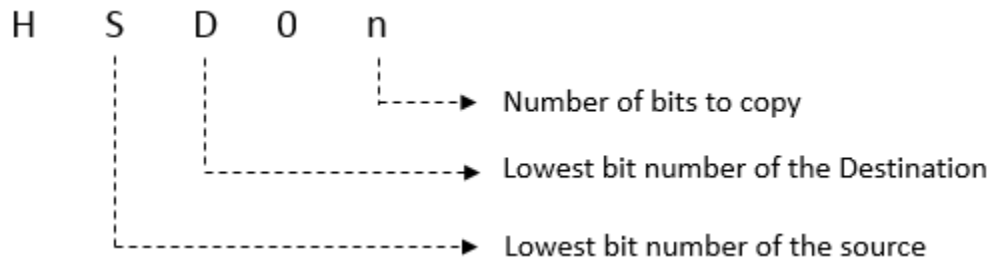


S	Source
D	Destination
fm	Lowest bit number of the group and number of bits

Instruction		Device address													No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
WBMOV WBMOVP	S	o	o	o	o	o	o	o	o	-	o	o	o	-	4	o	-	-
	D	o	-	o	o	o	-	o	o	-	o	o	o	-				
	fm	o	o	o	o	o	o	o	o	-	o	o	o	o				

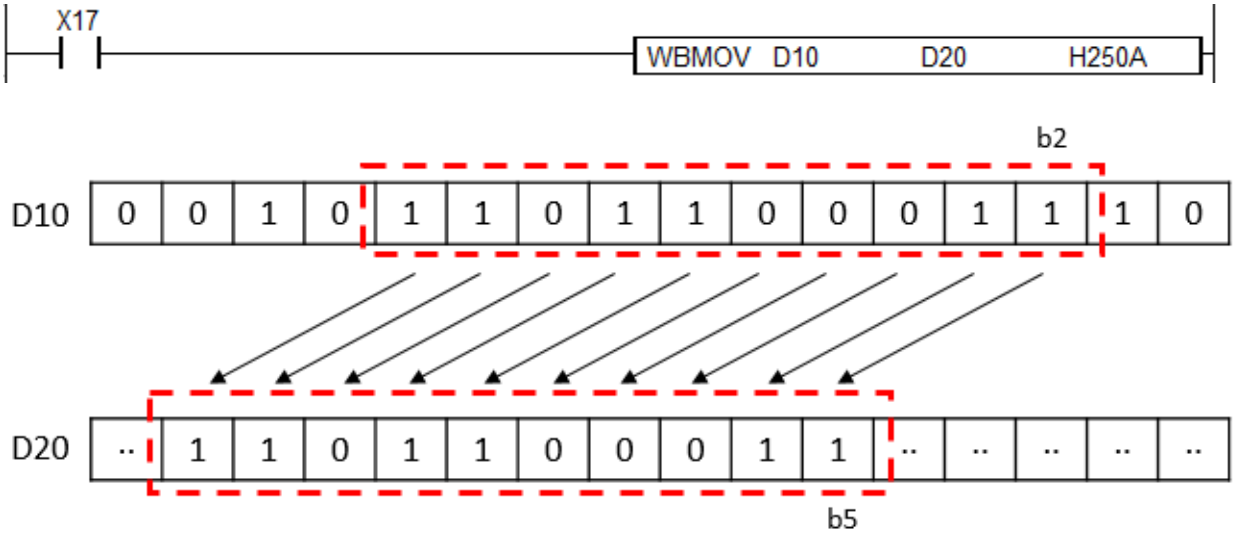
- WBMOV instruction copies a group of bits from (S) and write them in Destination in accordance with fm format.

- fm format :

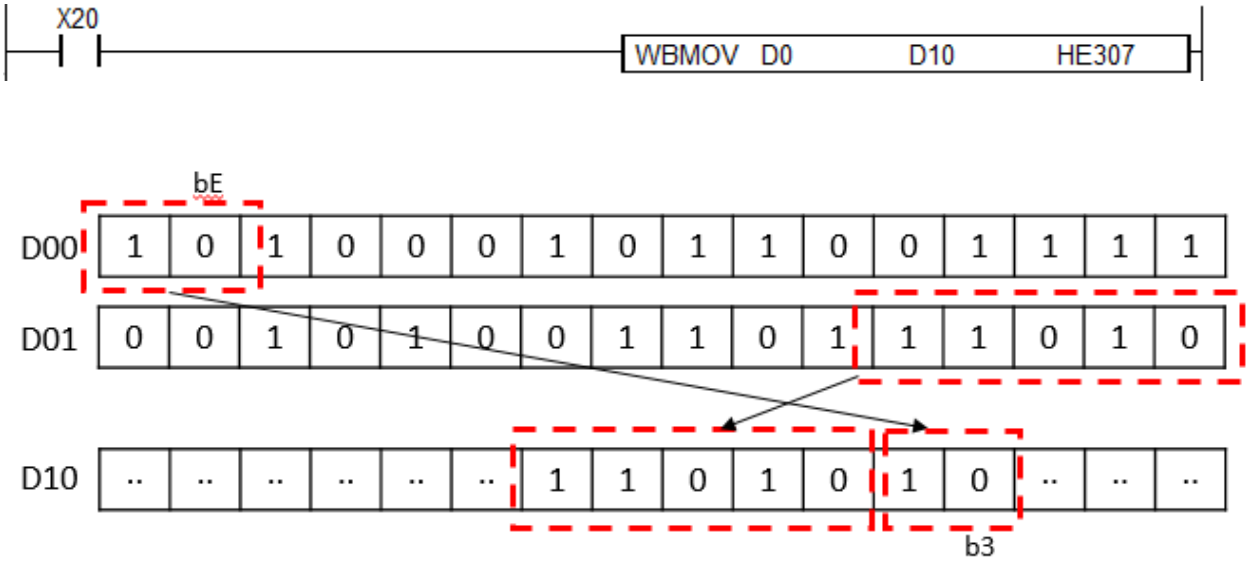


Example)

• If X17 is ON, copy 10 bits from bit number 2 of D10 and write them in D20 starting from bit number 5.

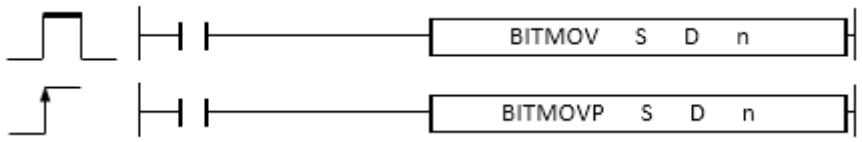


• If X20 is ON, copy 7 bits from bit number E of D00 and write them in D10 starting from bit number 3.



2.4.9. BITMOV, BITMOVP

BITMOV instruction copies the specified bits from the source to the Destination.



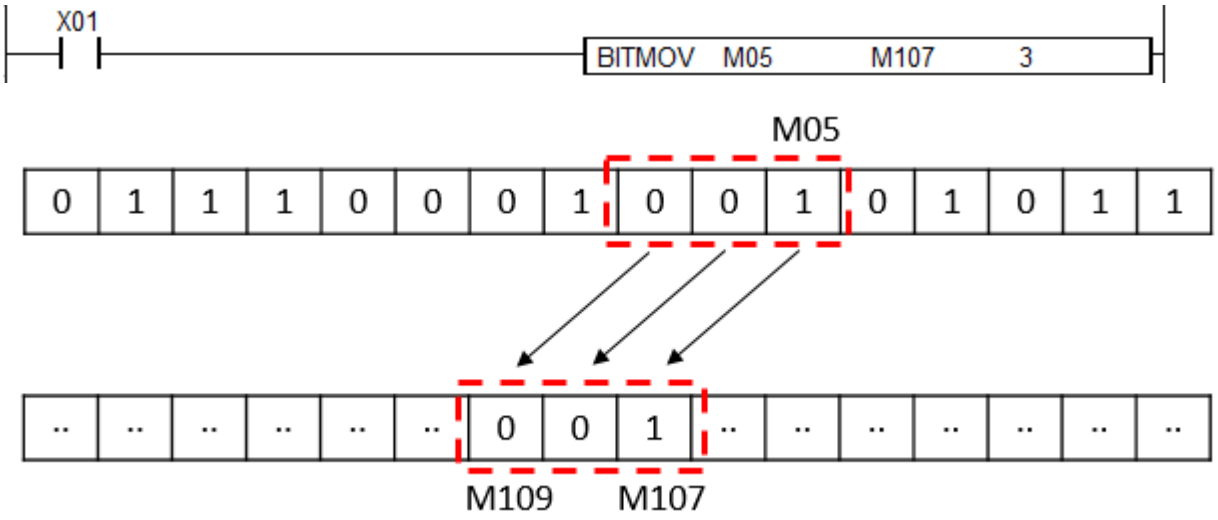
S	Source
D	Destination
n	Number of bits to copy

Instruction		Device address													No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
BITMOV BITMOVP	S	o	o	o	o	o	o	-	-	-	-	-	-	-	4	o	-	-
	D	o	-	o	o	o	-	-	-	-	-	-	-	-				
	n	o	o	o	o	o	o	o	o	-	o	o	o	o				

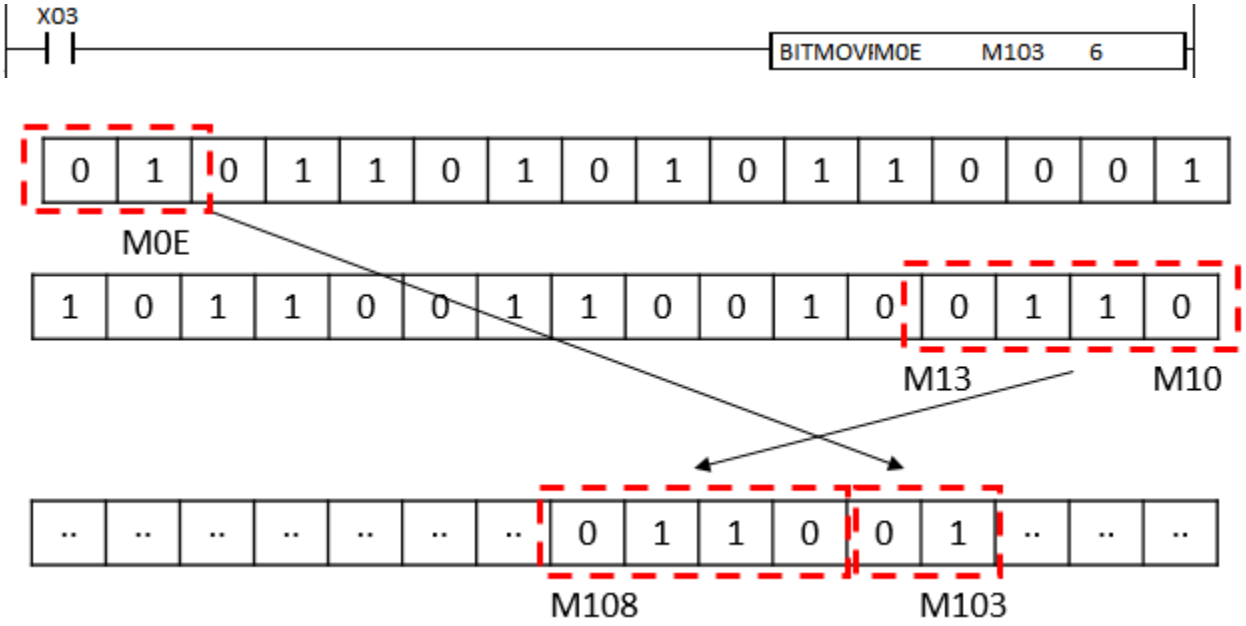
- BITMOV instruction copies (n) number of bits from (S) and writes them the lowest bit number to start with in the Destination.

Example)

• If X01 is ON, the BITMOV instruction copies 3 bits from M05 and writes them in M107 in sequence.

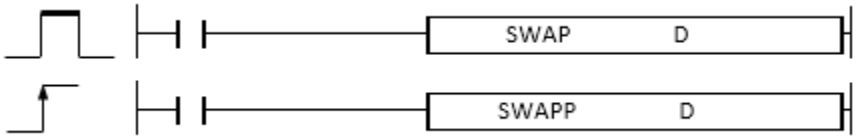


• If X03 is ON, the BITMOVP copies 6 bits from M0E and writes them in M103 in sequence.



2.4.10. SWAP, SWAPP

SWAP instruction reverses the bytes of the source and save the result in the Destination.

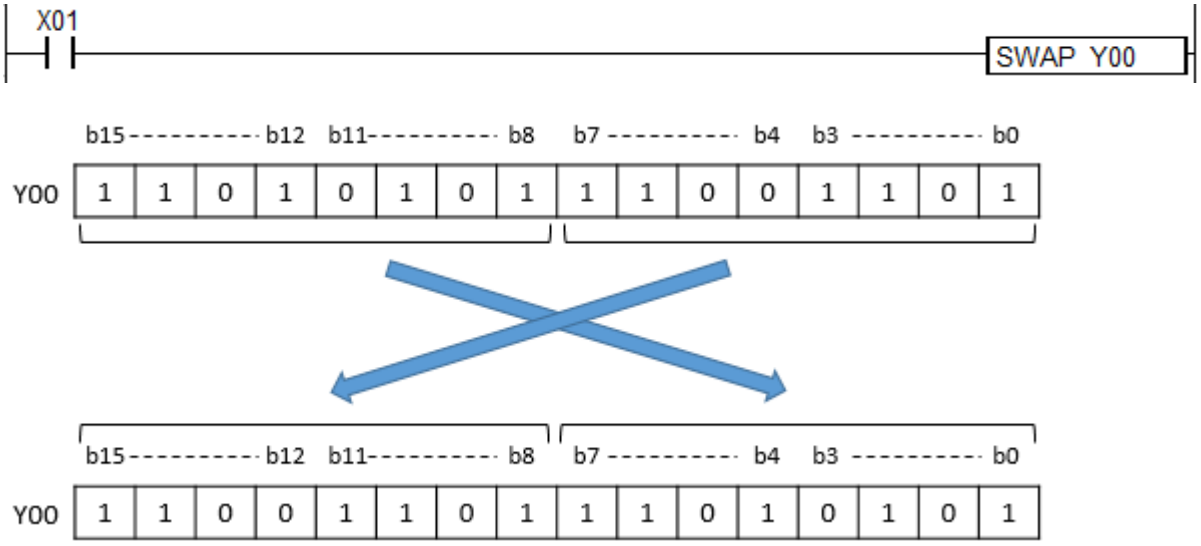


D	Source and Destination

Instruction		Device address													No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
SWAP SWAPP	D	o	-	o	o	o	-	o	o	-	o	o	o	-	2	o	-	-

Example)

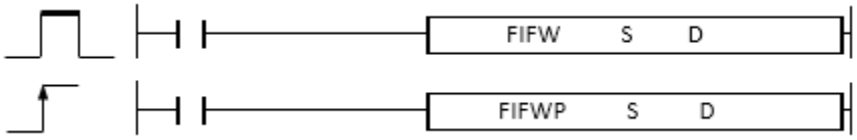
• If X01 is ON, the SWAP instruction reverses the order of the high bytes and low bytes of the Y00



2.5. Data Table Operation Instruction

2.5.1. FIFW, FIFWP

FIFW instruction copies the source to the Destination (assigned device address).

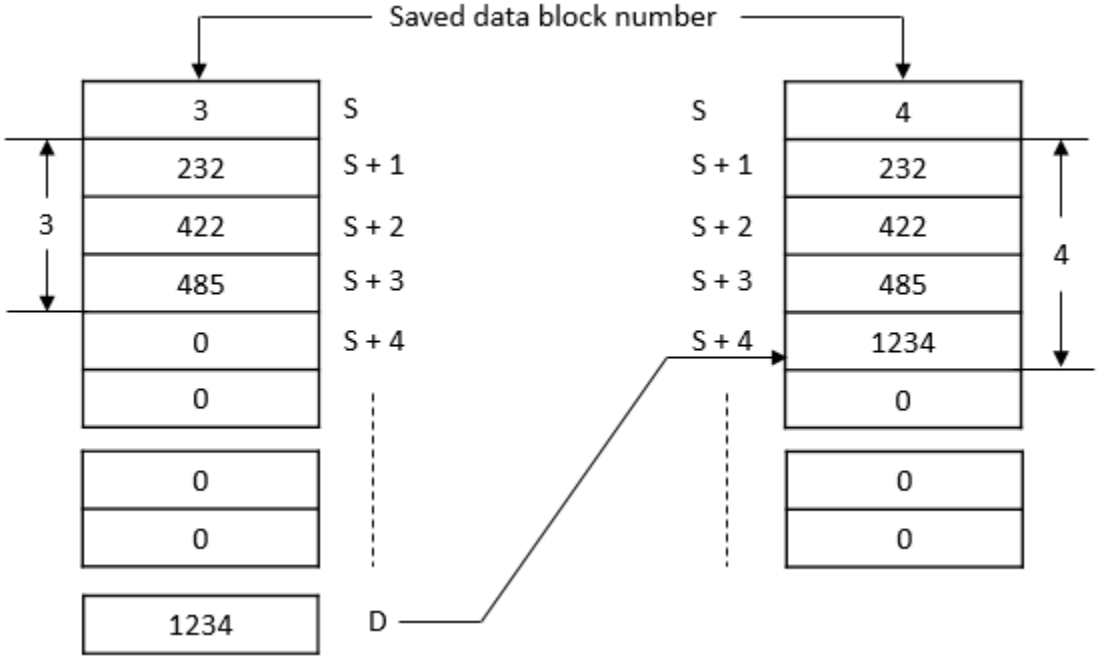


S	Source to copy
D	Destination

Instruction		Device address													No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
FIFW FIFWP	S	o	o	o	o	o	o	o	o	-	o	o	o	o	3	o	-	-
	D	-	-	-	-	-	-	-	-	-	o	o	o	-				

- When enabled, the number of word device(S) is increased by one word.

-FIFW(P) instruction shift each element of array (S+1 ~ S+3) into the same position within array S and copies 1234 (D) and places at the end of element of array S (S+4).

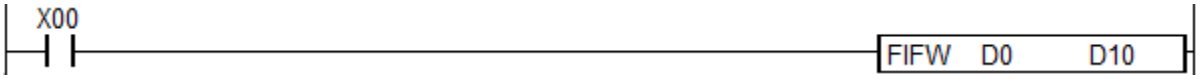


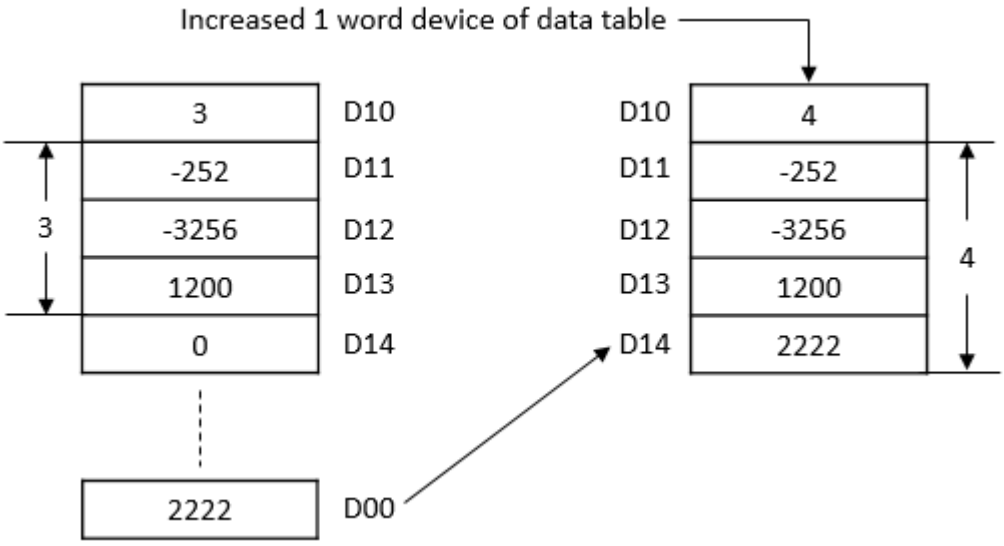
Example)

• If X00 is ON, copy D11, D12 and D13 (3 array) and write them to the same position.

The number of word device 3 in D10 is increased to 4 word.

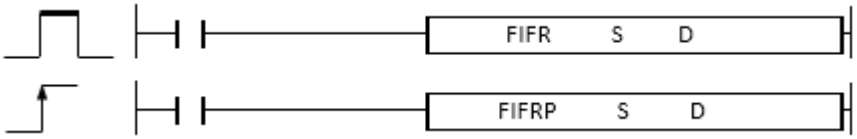
And then FIFW copies 2222 (D00) to D14.





2.5.2. FIFR, FIFRP

FIFR instruction copies the array to the Destination and moves specific word to the other position.

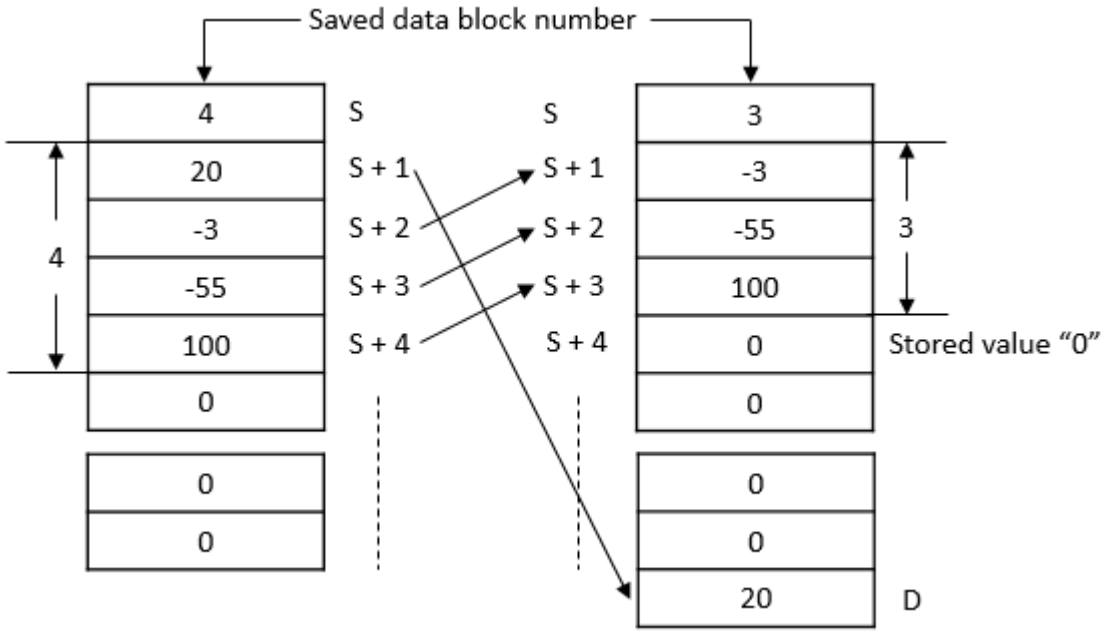


S	Source to copy
D	Destination

Instruction		Device address													No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
FIFR FIFRP	D	o	o	o	o	o	o	o	o	-	o	o	o	o	3	o	-	-
	S	-	-	-	-	-	-	-	-	-	o	o	o	-				

- When enabled, the number of word device(S) is decreased by one word.

- FIFR(P) instruction shifts each elements of array(S+2 ~ S+4) within S+1 ~ S+3 and copies 20 (S+1) to D.

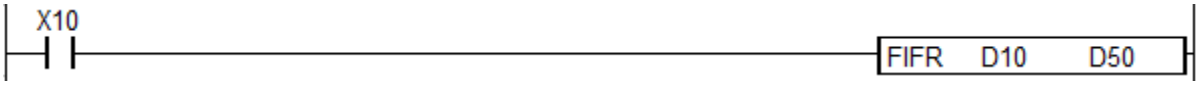


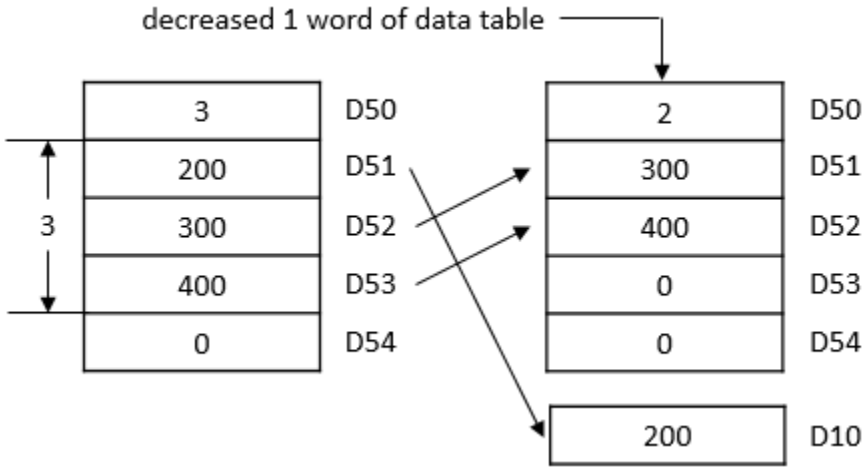
Example)

• If X10 is ON, copy D52 and D53 (2 array) and write them to D51 and D52.

The number of word device 3 in D50 is decreased to 2 word.

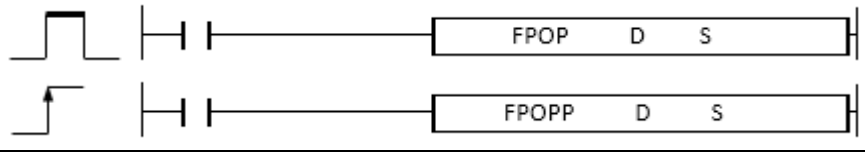
And then FIFR copies 200 (D51) to D10.





2.5.3. FPOP, FPOPP

FPOP instruction overwrites the array of Source to the Source and copies the last word of source to the Destination.

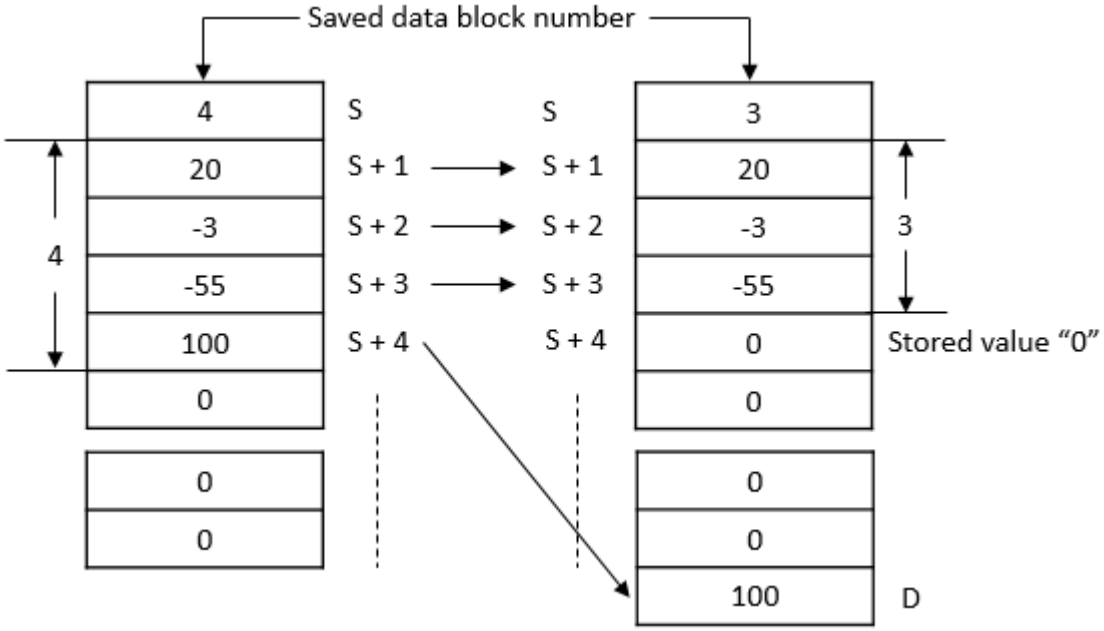


D	Destination
S	Source

Instruction		Device address													No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
FPOP FPOPP	D	o	o	o	o	o	o	o	o	-	o	o	o	o	3	o	-	-
	S	-	-	-	-	-	-	-	-	-	o	o	o	-				

- When enabled, the number of word device(S) is decreased by one word.

- FPOP(P) instruction shifts each elements of array(S+1 ~ S+3) within S+1 ~ S+3 and copies the last word S+4 (100) to the Destination.

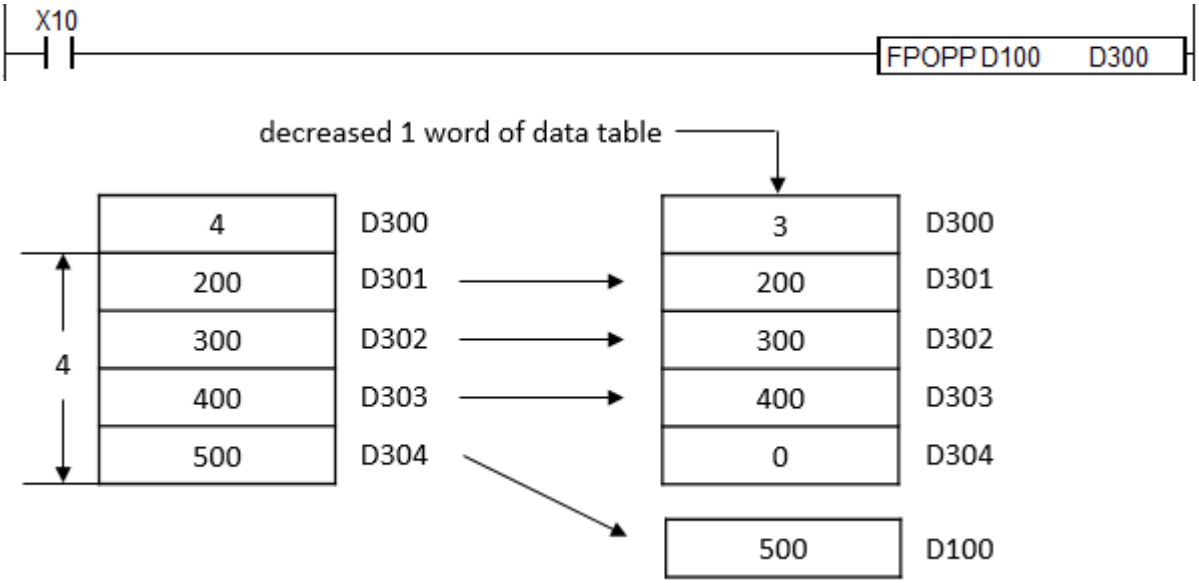


Example)

• If X10 is ON, copy D301, D302, and D303, (3 array) and write them to the same position D301, D302 and D303.

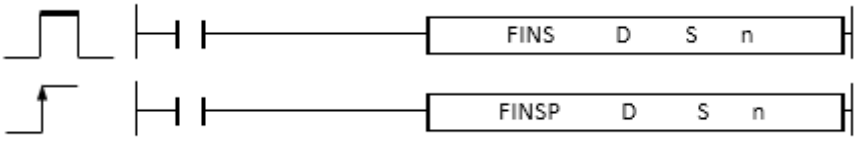
The number of word device 4 in D300 is decreased to 3 word.

And then FPOP copies the last word 500 (D304) to D100.



2.5.4. FINS, FINSP

FINS instruction overwrites the Source and copies the Destination to the (n) position of source.



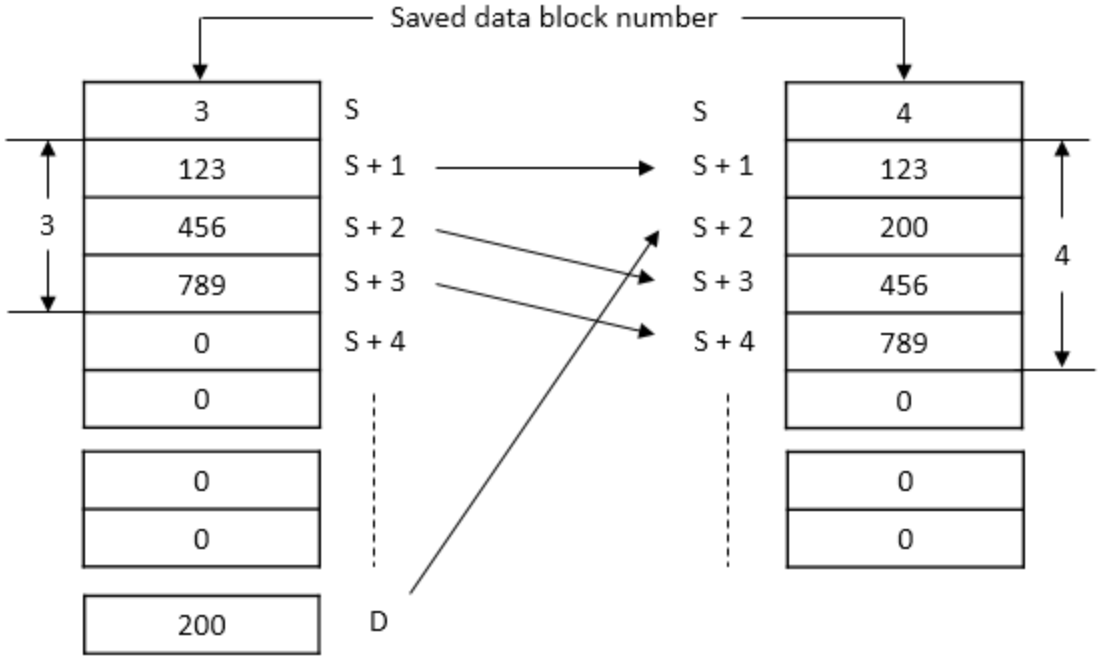
D	Destination
S	Source
n	Position of Destination where D is stored

Instruction		Device address													No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
FINS FINSP	D	o	o	o	o	o	o	o	o	-	o	o	o	o	4	o	-	-
	S	-	-	-	-	-	-	-	-	-	o	o	o	-				
	n	o	o	o	o	o	o	o	o	-	o	o	o	o				

- When enabled, the number of word device(S) is increased by one word.

- FINS(P) instruction copies 200(D) to S+2. (n is 2 which means 2nd word from S)

S+1 shifts to S+1. As S+2 is occupied by D, S+2 and S+3 move to S+3 and S+4 each.

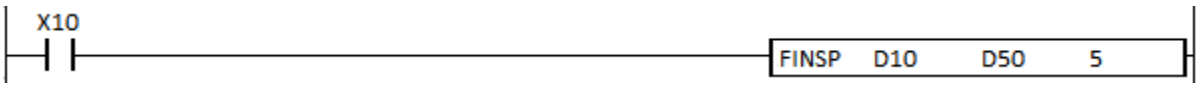


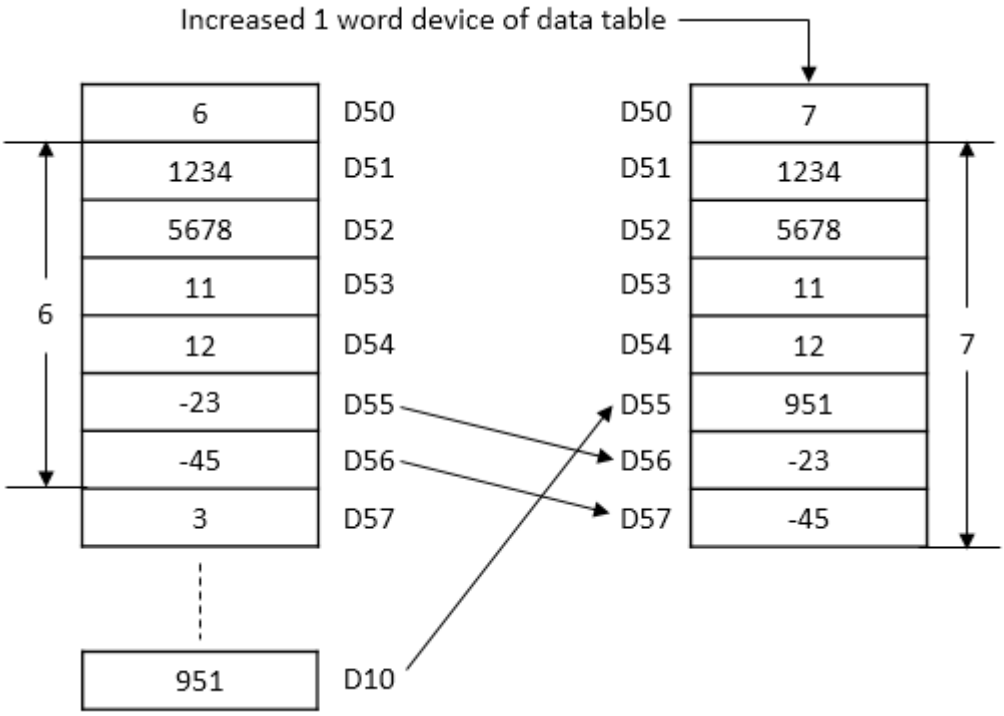
Example)

• If X10 is ON, copy D10 (951) to D55 and shift D51, D52, D53, and D54 to the same position.

As 951 (D10) is already occupied in D55, D55 and D56 are shifted to D56 and D57 each.

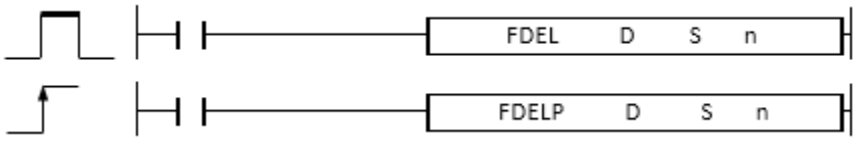
The number of word device 6 in D50 is increased to 7 word.





2.5.5. FDEL, FDELP

FDEL instruction copies the array to the Destination and move specific word to the other position.



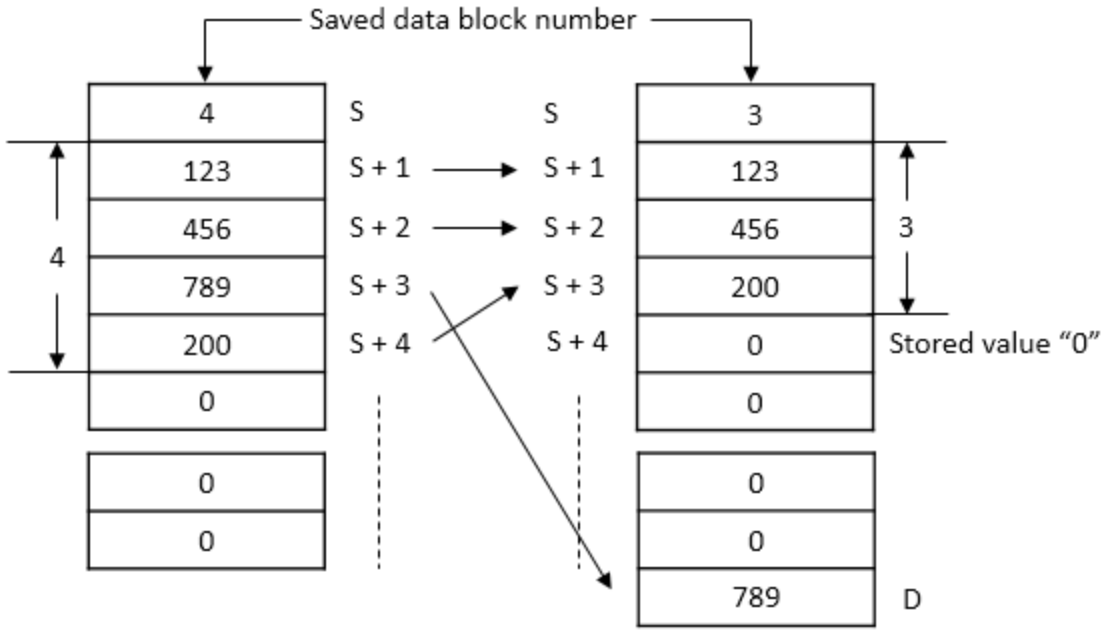
D	Destination
S	Source
n	Position umber of source to copy

Instruction		Device address													No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
FDEL FDELP	D	o	o	o	o	o	o	o	o	-	o	o	o	o	4	o	-	-
	S	-	-	-	-	-	-	-	-	-	o	o	o	-				
	n	o	o	o	o	o	o	o	o	-	o	o	o	o				

- When enabled, copy 789 (S+3) to the Destination. (n is 3 which means 3rd word from S).

- FDEL(P) instruction shifts S+1 and S+2 to the same position and shifts 200 (S+4) to S+3.

The number of word device 4 in S is decreased to 3 word.

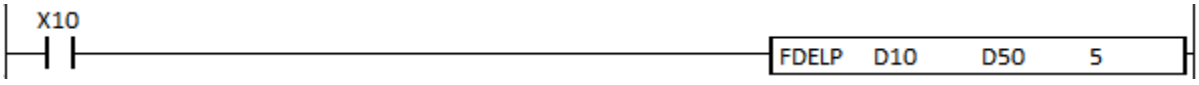


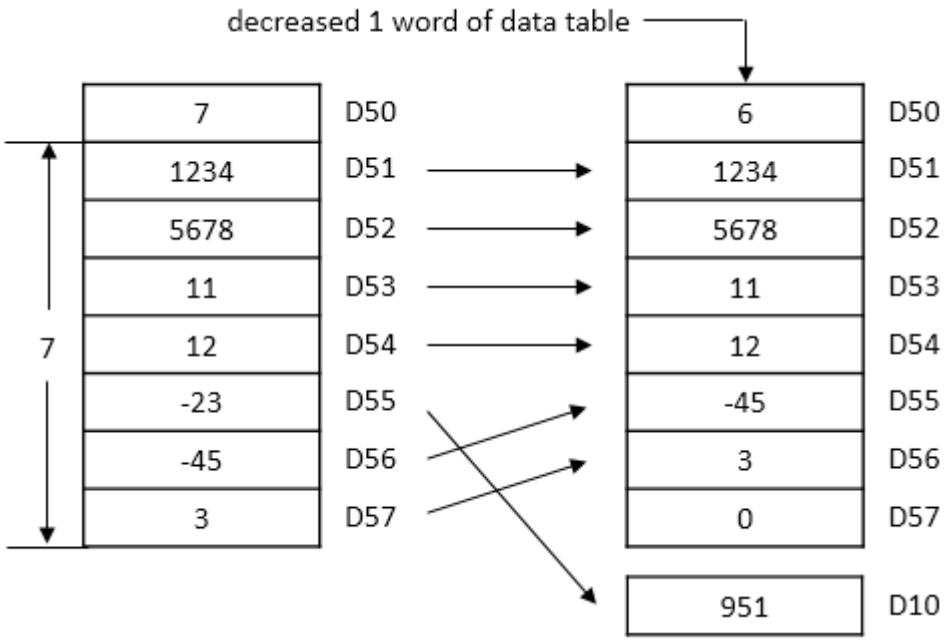
Example)

• If X10 is ON, copy -23 (5th word from D50) to D10 and shift D51, D52, D53, and D54 to the same position.

As -23 (D55) is already deleted, D56 and D57 are shifted to D55 and D56 each.

The number of word device 7 in D50 is decreased to 6 word.



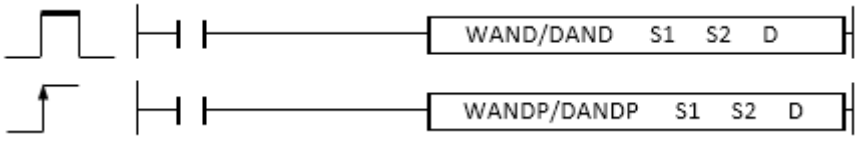


2.6. Logic Operation Instruction

2.6.1. WAND, WANDP, DAND, DANDP

WAND instruction performs a bitwise AND operation on Source 1 and Source 2 and places the result in the Destination. The bitwise AND operation returns 1, if the matching bits from both the operands are 1, otherwise it returns 0.

- WAND(WANDP) is for 16bit source and DAND(DANDP) is for 32bit source.



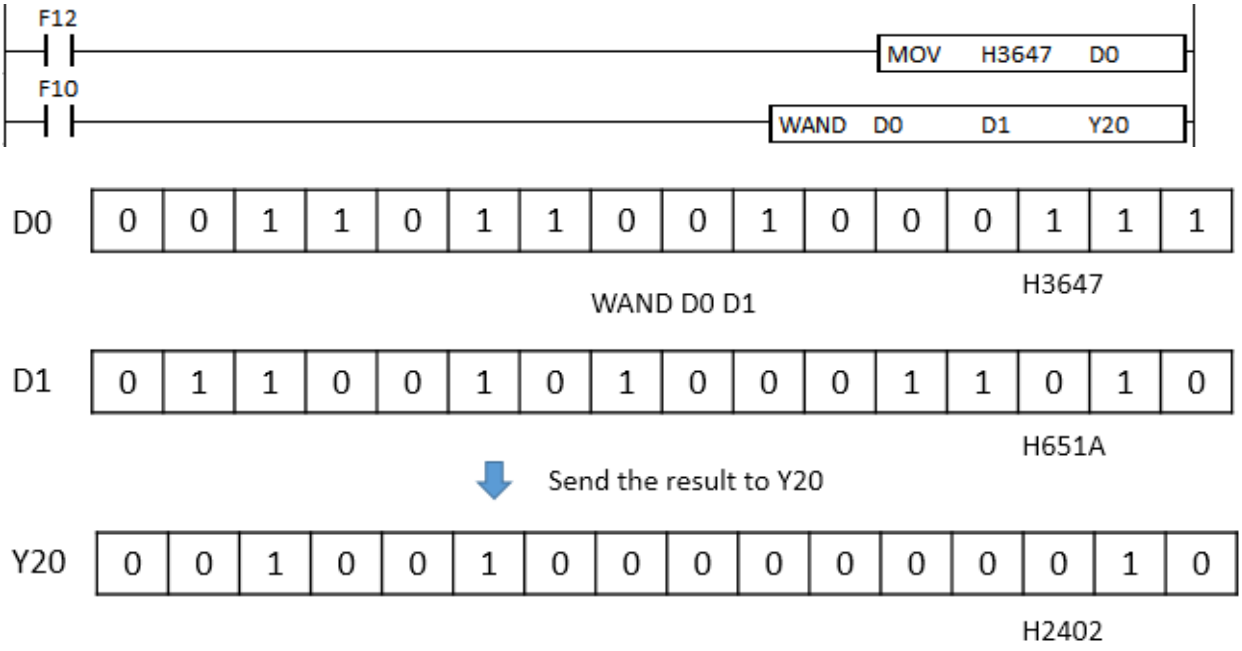
S1, S2	Source
D	Destination

Instruction		Device address													No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
WAND WANDP DAND DANDP	S1	o	o	o	o	o	o	o	o	-	o	o	o	o	4	o	-	-
	S2	o	o	o	o	o	o	o	o	-	o	o	o	o				
	D	o	-	o	o	o	-	o	o	-	-	o	o	-				

Example)

• If F12 is ON, H3647 moves to D0.

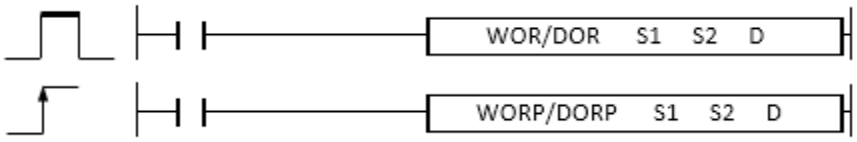
• If F10 is ON, the WAND performs a bitwise AND operation by using the 16bits in H3647 (D0) and H651A (D1) and places the result H2402 in the Y20.



2.6.2. WOR, WORP, DOR, DORP

WOR instruction performs a bitwise OR operation on Source 1 and Source 2 and places the result in the Destination. The bitwise OR operation returns 1, if the matching bits from either or both operands are 1. It returns 0, if both the bits are 0.

- WOR(WORP) is for 16bit source and DOR(DORP) is for 32bit source.



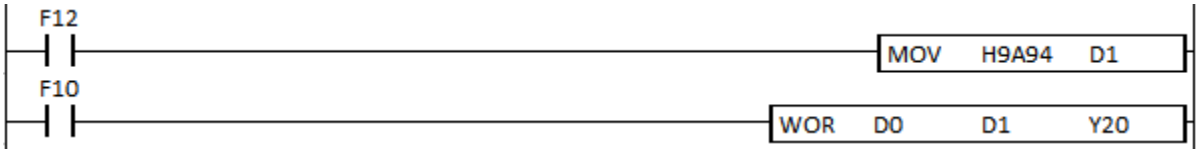
S1, S2	Source
D	Destination

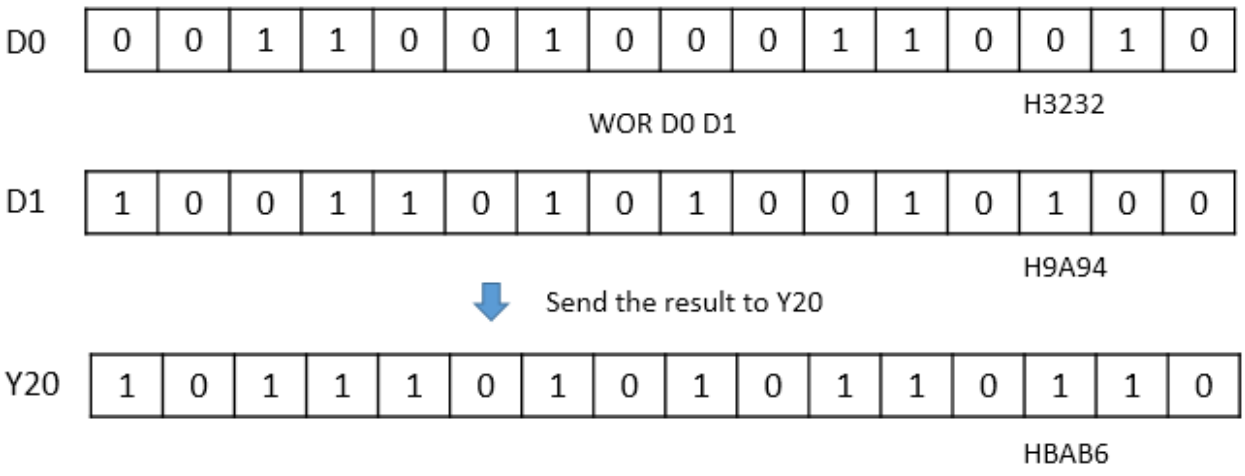
Instruction		Device address													No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
WOR WORP DOR DORP	S1	o	o	o	o	o	o	o	o	-	o	o	o	o	4	o	-	-
	S2	o	o	o	o	o	o	o	o	-	o	o	o	o				
	D	o	-	o	o	o	-	o	o	-	-	o	o	-				

Example)

• If F12 is ON, H9A94 moves to D1.

• If F10 is ON, the WOR performs a bitwise OR operation by using the 16bits in H3232 (D0) and H9A94 (D1) and places the result HBAB6 in the Y20.

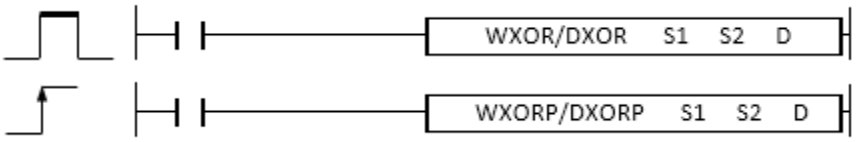




2.6.3. WXOR, WXORP, DXOR, DXORP

WXOR instruction performs a bitwise XOR operation on Source 1 and Source 2 and places the result in the Destination. The bitwise XOR operation returns 1, if the bits from the operands are different. It returns 0, if the bits from the operands are the same (both 0 or both 1).

- WXOR(WXORP) is for 16bit source and DXOR(DXORP) is for 32bit source.



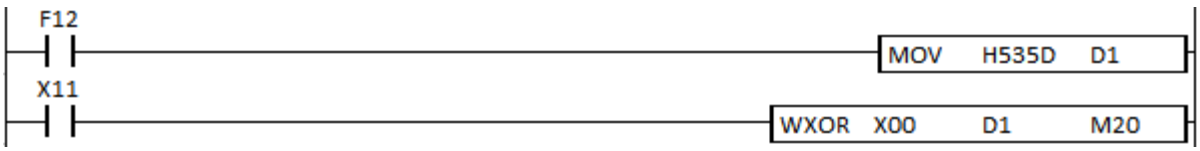
S1, S2	Source
D	Destination

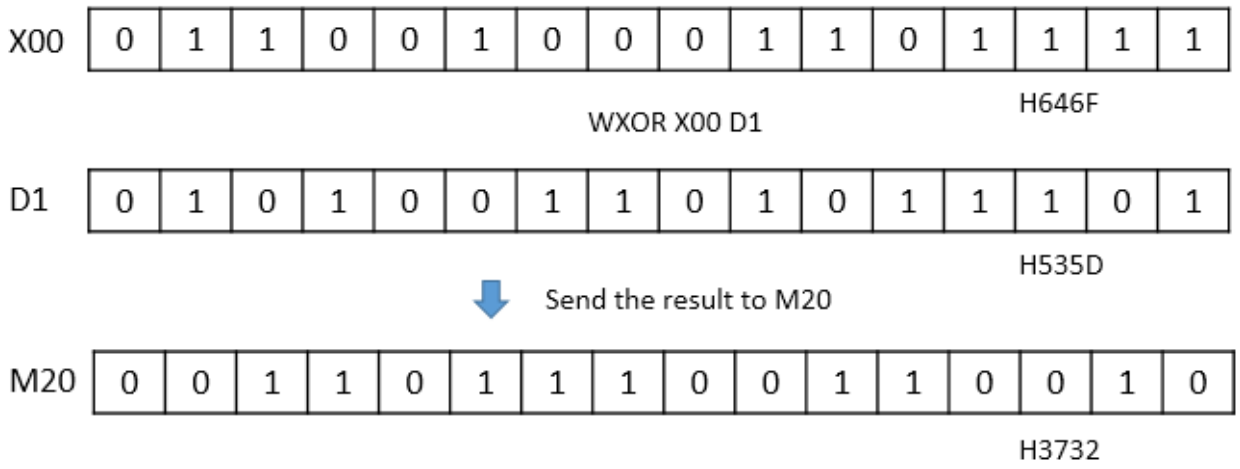
Instruction		Device address													No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
WXOR WXORP DXOR DXORP	S1	o	o	o	o	o	o	o	o	o	-	o	o	o	4	o	-	-
	S2	o	o	o	o	o	o	o	o	o	-	o	o	o				
	D	o	-	o	o	o	-	o	o	-	-	o	o	-				

Example)

• If F12 is ON, H535D moves to D1.

• If X11 is ON, the WXOR performs a bitwise XOR operation by using the 16bits in H646F (X00) and H535D (D1) and places the result H3732 in the M20.

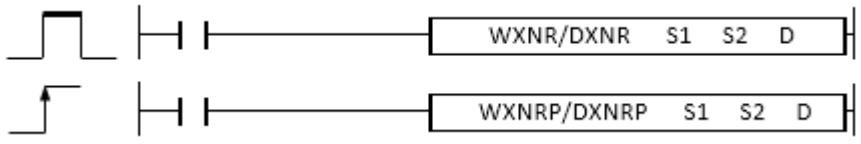




2.6.4. WXNR, WXNRP, DXNR, DXNRP

WXNR instruction performs a bitwise XNOR operation on Source 1 and Source 2 and places the result in the Destination. The bitwise XNOR operation returns 0, if the bits from the operands are different. It returns 1, if the bits from the operands are the same (both 0 or both 1).

- WXNR(WXNRP) is for 16bit source and DXNR(DXNRP) is for 32bit source.



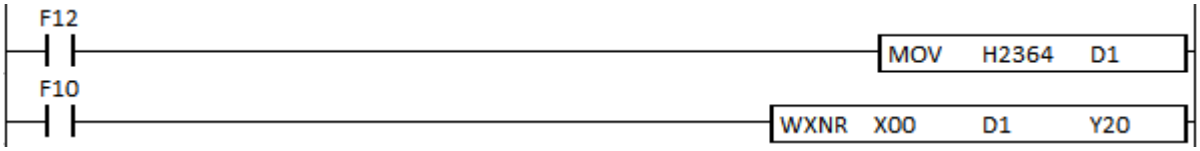
S1, S2	Source
D	Destination

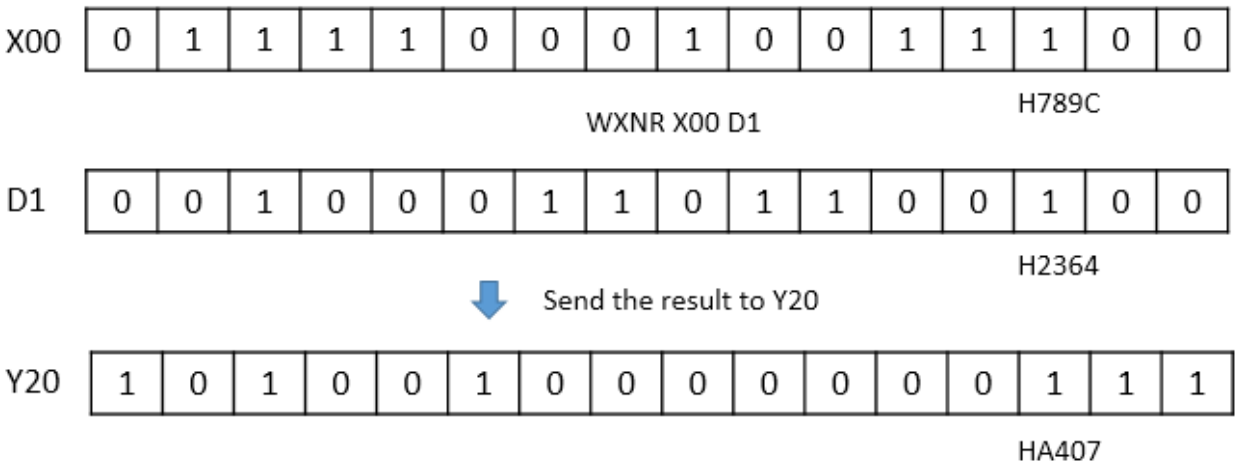
Instruction		Device address													No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
WXNR WXNRP DXNR DXNRP	S1	o	o	o	o	o	o	o	o	-	-	o	o	o	4	o	-	-
	S2	o	o	o	o	o	o	o	o	-	-	o	o	o				
	D	o	-	o	o	o	-	o	o	-	-	o	o	-				

Example)

• If F12 is ON, H2364 moves to D1.

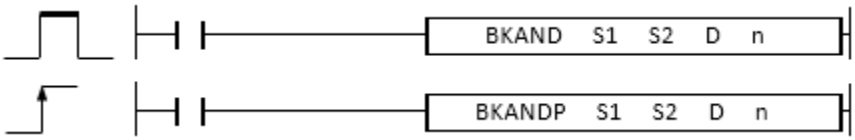
• If F10 is ON, the WXNR performs a bitwise XNOR operation by using the 16bits in H789C (X00) and H2364 (D1) and places the result HA407 in the Y20.





2.6.5. BKAND, BKANDP

BKAND instruction performs a bitwise AND operation by using specified word (16bit) in Source 1 and Source 2 and places the result in the Destination. The bitwise AND operation returns 1, if the matching bits from both the operands are 1, otherwise it returns 0.



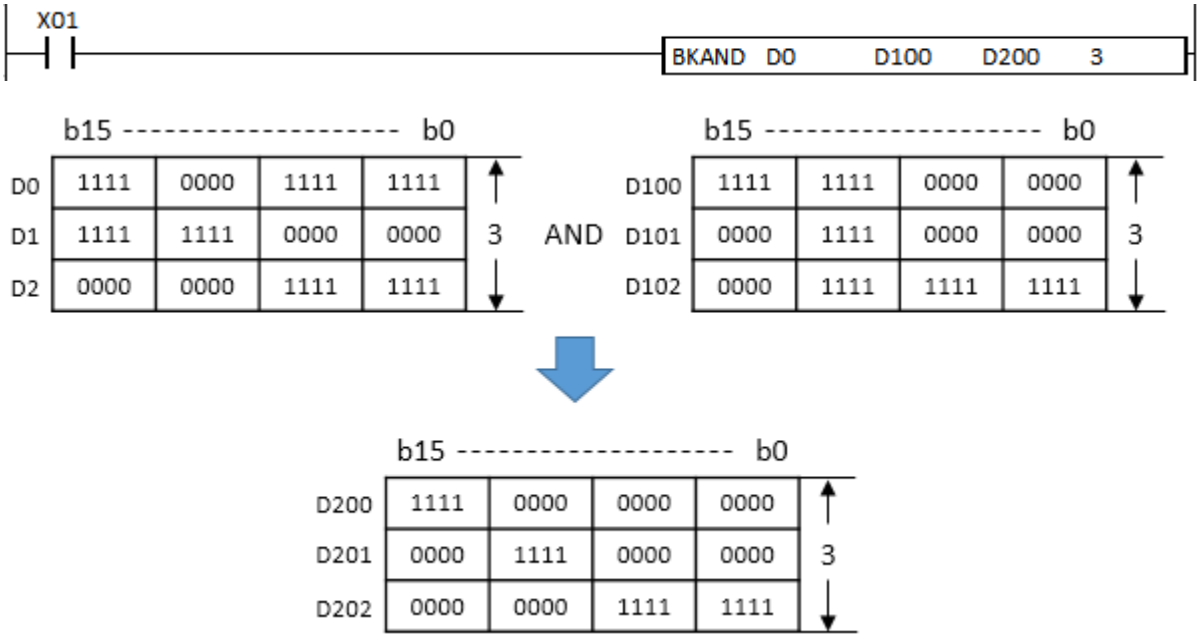
S1, S2	Source
D	Destination
n	Number of words

Instruction		Device address													No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
BKAND BKANDP	S1	o	o	o	o	o	o	o	o	-	o	o	o	o	5	o	-	-
	S2	o	o	o	o	o	o	o	o	-	o	o	o	o				
	D	o	-	o	o	o	-	o	o	-	o	o	o	-				
	n	o	o	o	o	o	o	o	o	-	o	o	o	o				

Example)

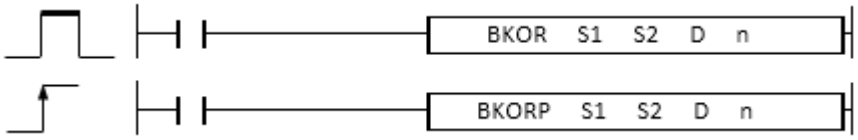
• If X01 is ON, the BKAND performs a bitwise AND operation by using the 3 word of D0(D0, D1, and D2) and D100(D100, D101, and D102) and places the results in D200, D201, and D202.

- Source 1 : D0, D1, and D2
- Source 2 : D100, D101, and D102
- Destination : D200, D201, and D202
- Number of word : 3 word



2.6.6. BKOR, BKORP

BKOR instruction performs a bitwise OR operation by using (n)number of word (16bit) on Source 1 and Source 2 and places the result in the Destination. The bitwise OR operation returns 1, if the matching bits from either or both operands are 1. It returns 0, if both the bits are 0.



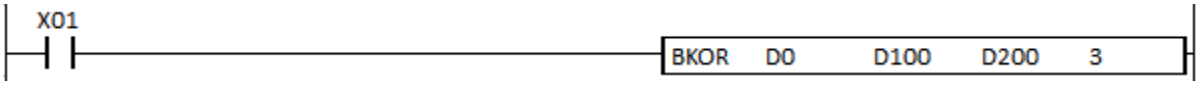
S1, S2	Source
D	Destination
n	Number of words

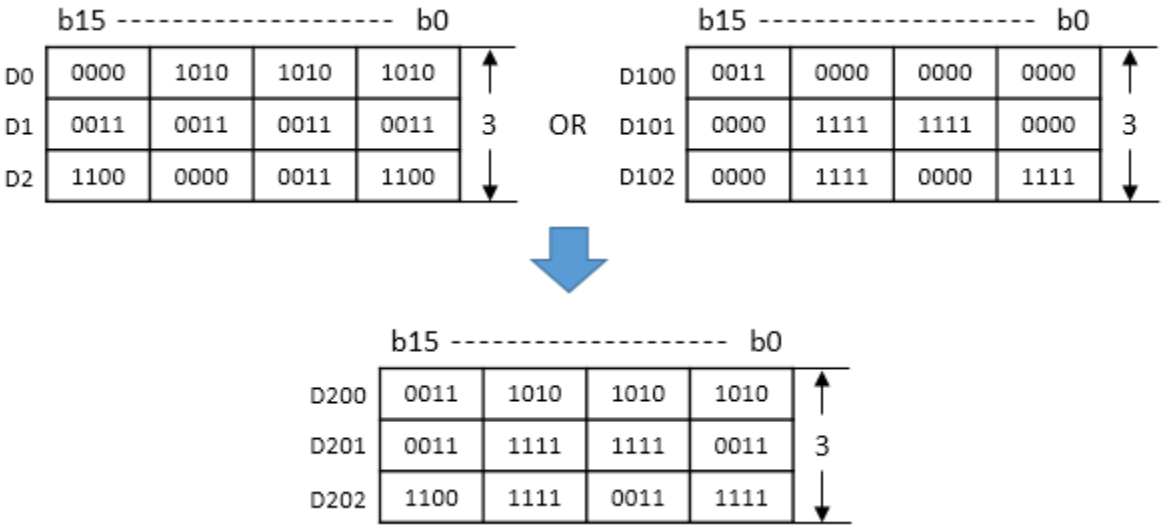
Instruction		Device address													No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
BKOR BKORP	S1	o	o	o	o	o	o	o	o	-	o	o	o	o	5	o	-	-
	S2	o	o	o	o	o	o	o	o	-	o	o	o	o				
	D	o	-	o	o	o	-	o	o	-	o	o	o	-				
	n	o	o	o	o	o	o	o	o	-	o	o	o	o				

Example)

• If X01 is ON, the BKOR performs a bitwise OR operation by using the 3 word of D0(D0, D1, and D2) and D100(D100, D101, and D102) and places the results in D200, D201, and D202.

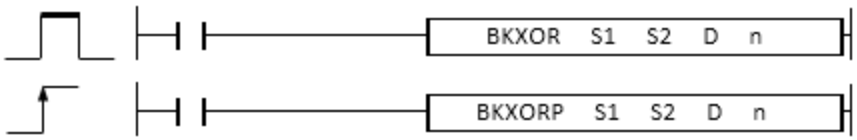
- Source 1 : D0, D1, and D2
- Source 2 : D100, D101, and D102
- Destination : D200, D201, and D202
- Number of word : 3 word





2.6.7. BKXOR, BKXORP

BKXOR instruction performs a bitwise XOR operation by using (n)number of word(16bit) on Source 1 and Source 2 and places the result in the Destination. The bitwise XOR operation returns 1, if the bits from the operands are different. It returns 0, if the bits from the operands are the same (both 0 or both 1).



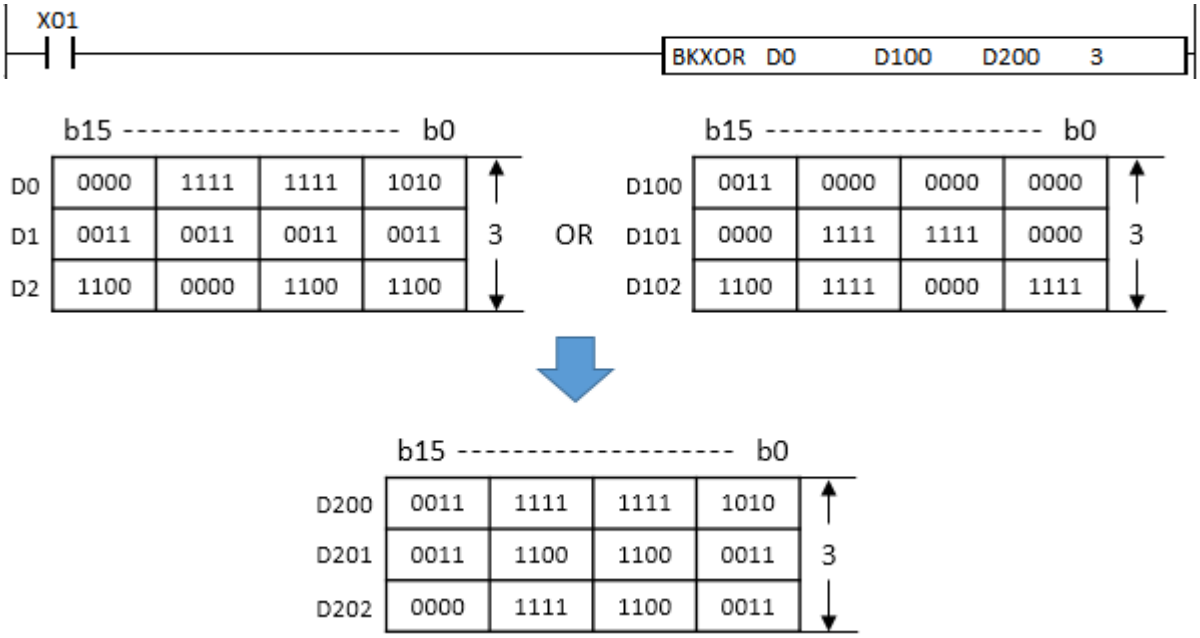
S1, S2	Source
D	Destination
n	Number of words

Instruction		Device address													No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
BKXOR BKXORP	S1	o	o	o	o	o	o	o	o	-	o	o	o	o	5	o	-	-
	S2	o	o	o	o	o	o	o	o	-	o	o	o	o				
	D	o	-	o	o	o	-	o	o	-	o	o	o	-				
	n	o	o	o	o	o	o	o	o	-	o	o	o	o				

Example)

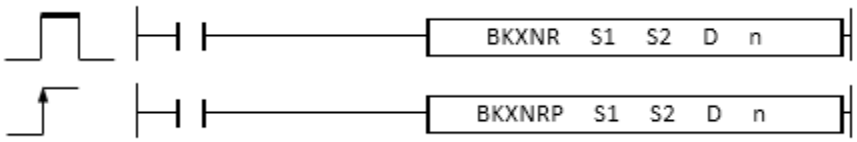
• If X01 is ON, the BKXOR performs a bitwise XOR operation by using the 3 word of D0(D0, D1, and D2) and D100(D100, D101, and D102) and places the results in D200, D201, and D202.

- Source 1 : D0, D1, and D2
- Source 2 : D100, D101, and D102
- Destination : D200, D201, and D202
- Number of word : 3 word



2.6.8. BKXNR, BKXNRP

BKXNR instruction performs a bitwise XNOR operation by using (n) number of word (16bit) on Source 1 and Source 2 and places the result in the Destination. The bitwise XNOR operation returns 0, if the bits from the operands are different. It returns 1, if the bits from the operands are the same (both 0 or 1).



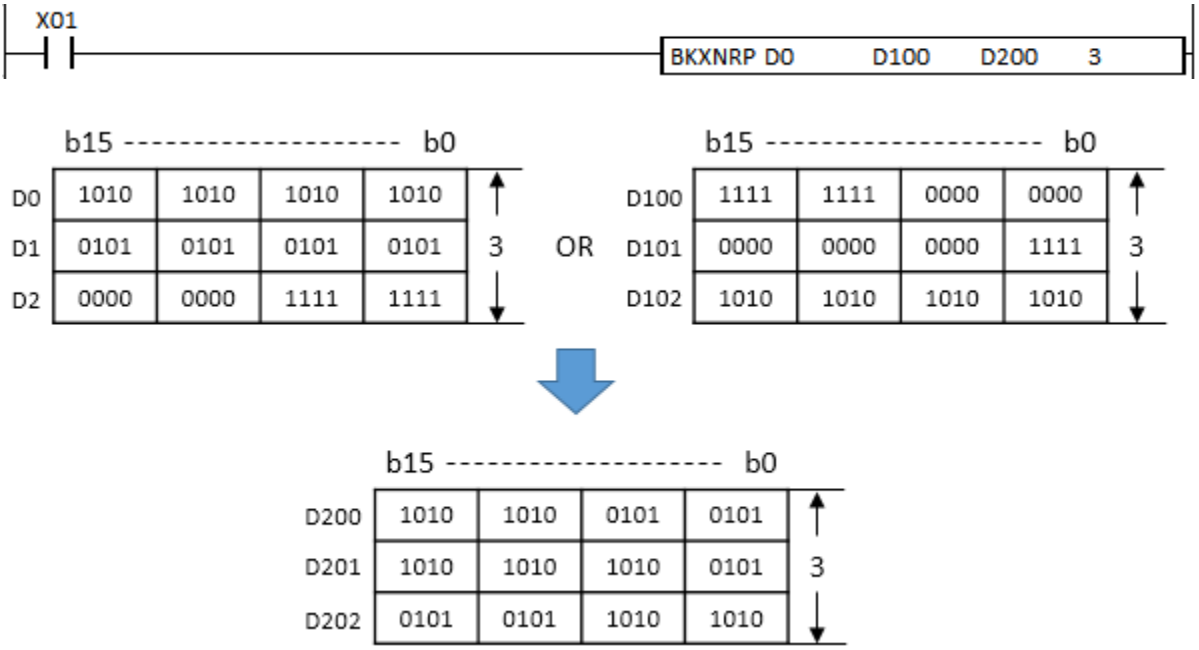
S1, S2	Source
D	Destination
n	Number of words

Instruction		Device address													No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
BKXNR BKXNRP	S1	o	o	o	o	o	o	o	o	-	o	o	o	o	5	o	-	-
	S2	o	o	o	o	o	o	o	o	-	o	o	o	o				
	D	o	-	o	o	o	-	o	o	-	o	o	o	-				
	n	o	o	o	o	o	o	o	o	-	o	o	o	o				

Example)

• If X01 is ON, the BKXNR performs a bitwise XNOR operation by using the 3 word of D0(D0, D1, and D2) and D100(D100, D101, and D102) and places the results in D200, D201, and D202.

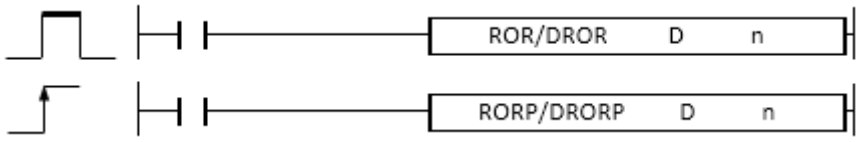
- Source 1 : D0, D1, and D2
- Source 2 : D100, D101, and D102
- Destination : D200, D201, and D202
- Number of word : 3 word



2.7. Rotation Instruction

2.7.1. ROR, RORP, DROR, DRORP

ROR instruction rotates the (D) data source towards the right by (n) bits without carry flag.
- ROR(RORP) is for 16bit source and DROR(DRORP) is for 32bit source.

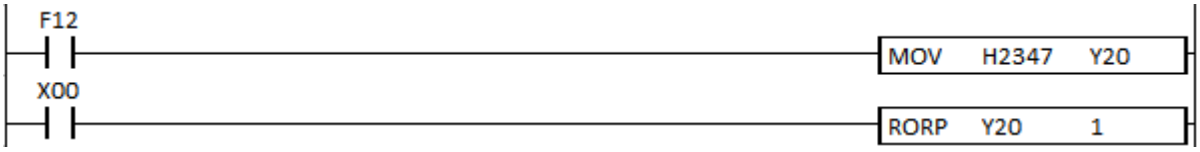


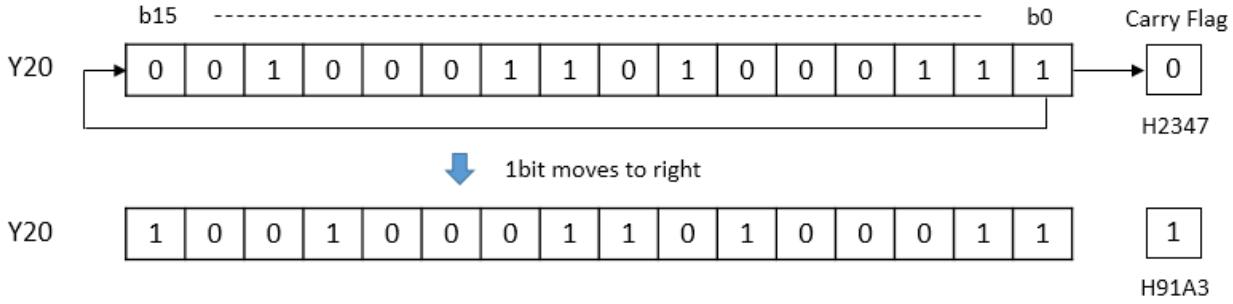
D	Source
n	Number of bits to move right

Instruction		Device address													No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
ROR(P) DROR(P)	D	o	-	o	o	o	-	o	o	-	o	o	o	-	3	o	-	o
	n	o	o	o	o	o	o	o	o	-	o	o	o	o				

Example)

• If F12 is ON, H2347 moves to Y20.
• If X00 is ON, the RORP instruction moves 1bit to the right side within Y20(16bit).
The lowest bit (b0) moves to the highest bit (b15) as rotation.

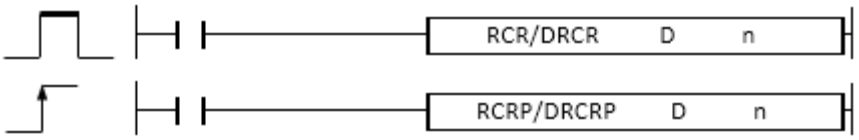




2.7.2. RCR, RCRP, DRCR, DRCRP

RCR instruction rotates the (D) data source towards the right by (n) bits with carry flag (F112).

- RCR(RCRP) is for 16bit source and DRCR(DRCRP) is for 32bit source.



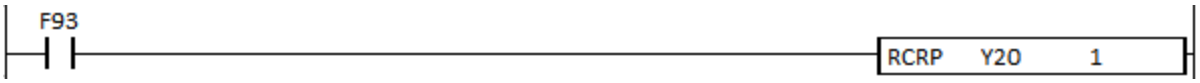
D	Source
n	Number of bits to move right

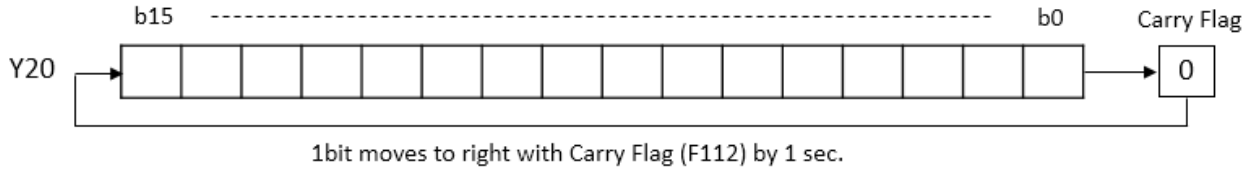
Instruction		Device address													No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
RCR(P) DRCR(P)	D	o	-	o	o	o	-	o	o	-	o	o	o	-	3	o	-	o
	n	o	o	o	o	o	o	o	o	-	o	o	o	o				

Example)

• If F93 is ON, the RCRP instruction moves 1bit to the right side within Y20 (16bit) by every 1 sec.

The lowest bit (b0) moves to carry flag and the carry flag moves to the highest bit (b15).



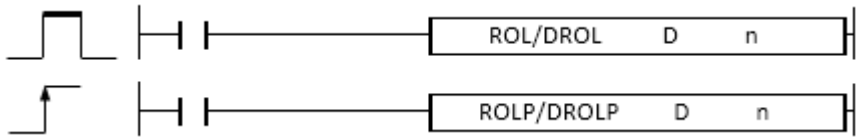


1bit moves to right with Carry Flag (F112) by 1 sec.

2.7.3. ROL, ROLP, DROL, DROLP

ROL instruction rotates the (D) data source towards the left by (n) bits without carry flag.

- ROL(ROLP) is for 16bit source and DROL(DROLP) is for 32bit source.



D	Source
n	Number of bits to move left

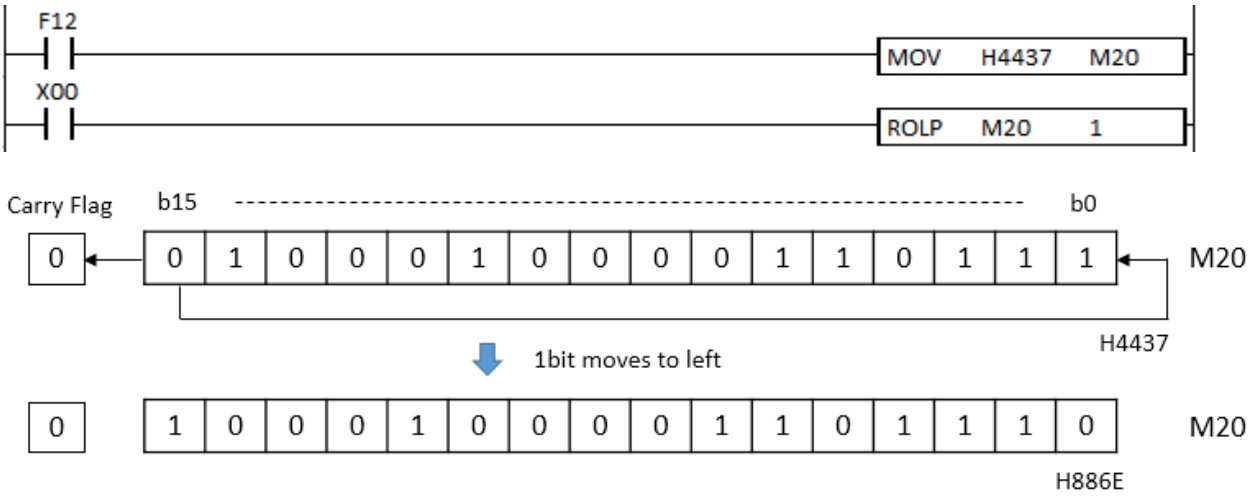
Instruction		Device address													No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
ROL(P) DROL(P)	D	o	-	o	o	o	-	o	o	-	o	o	o	-	3	o	-	o
	n	o	o	o	o	o	o	o	o	-	o	o	o	o				

Example)

• If F12 is ON, H4437 moves to M20.

• If X00 is ON, the ROLP instruction moves 1bit to the left side within M20 (16bit).

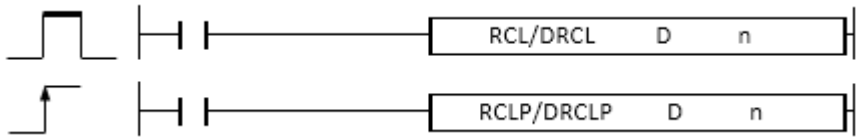
The highest bit (b15) moves to the lowest bit (b0) as rotation.



2.7.4. RCL, RCLP, DRCL, DRCLP

RCL instruction rotates the (D) data source towards the left by (n) bits with carry flag (F112).

- RCL (RCLP) is for 16bit source and DRCL (DRCLP) is for 32bit source.



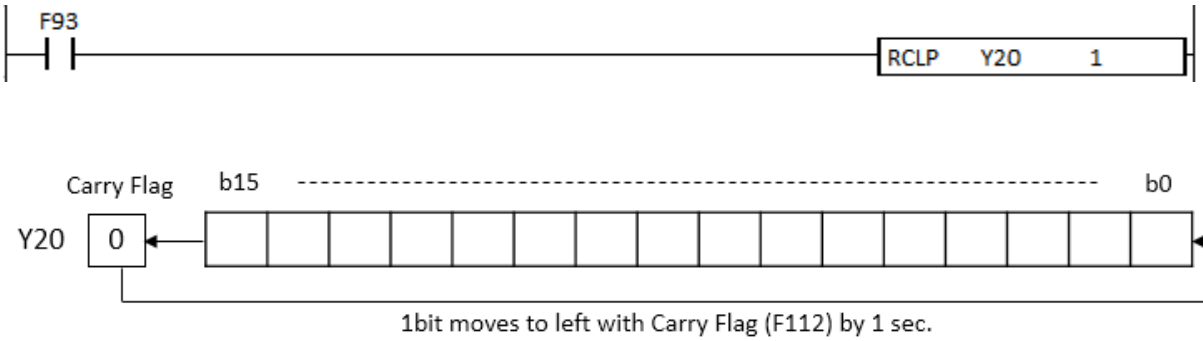
D	Source
n	Number of bits to move left

Instruction		Device address													No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
RCL(P) DRCL(P)	D	o	-	o	o	o	-	o	o	-	o	o	o	-	3	o	-	o
	n	o	o	o	o	o	o	o	o	-	o	o	o	o				

Example)

• If F93 is ON, the RCLP instruction moves 1bit to the left side within Y20 (16bit) by every 1 sec.

The highest bit (b15) moves to carry flag and the carry flag moves to the lowest bit (b0).

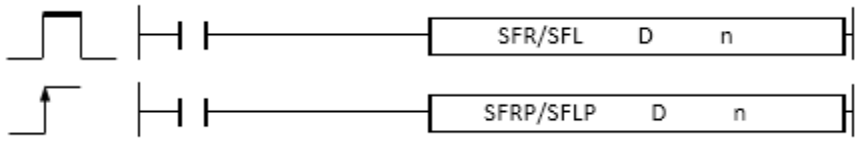


2.8. Shift Instruction

2.8.1. SFR, SFRP, SFL, SFLP

SFR(P) instruction shifts the specified bits(D) towards the right by (n) bits and loads Source bit into bit 0 of Array.

SFL(P) instruction shifts the specified bits(D) towards the left by (n) bits and loads Source bit into bit 0 of Array.



D	Source
n	Number of bits to move right(SFR) or left(SFL)

* In case of T and C device, count value can be shifted. (Set value cannot be shifted.)

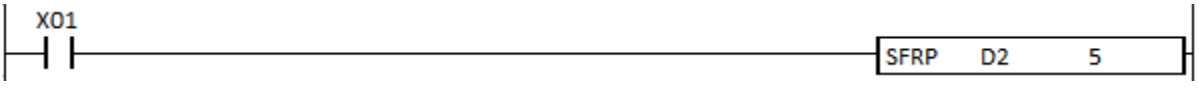
Instruction		Device address													No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
SFR(P) SFL(P)	D	o	-	o	o	o	-	o	o	-	o	o	o	-	3	o	-	o
	n	o	o	o	o	o	o	o	o	-	o	o	o	o				

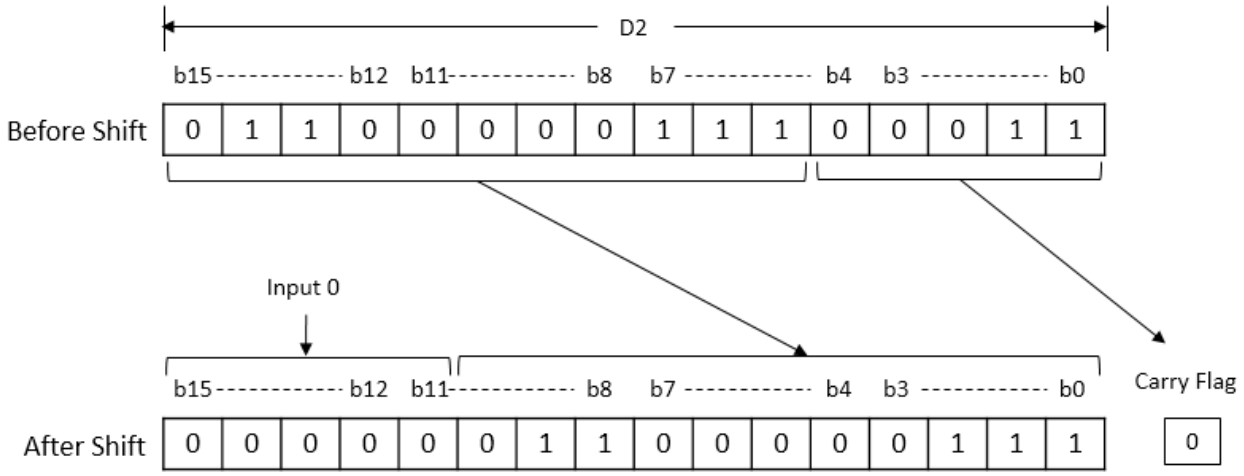
1) SFR(P)

Format: SFR D(Source) n(number of bit to move right)

Example)

• If X01 is ON, the SFRP instruction shifts D2 (16bit) 5 position right side and loads 0 into b11~b15.



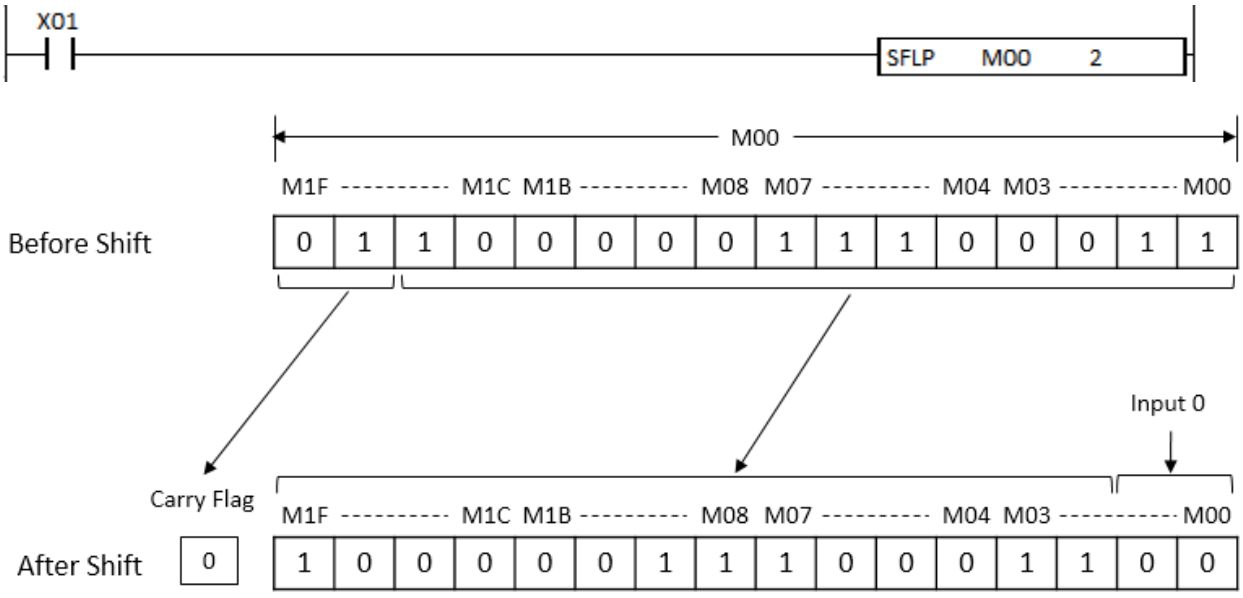


2) SFL(P)

Format : SFL D(Source) n(number of bit to move left)

Example)

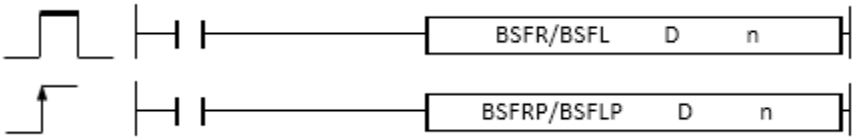
• If X01 is ON, the SFLP instruction shifts M00 (16bit) 2 position left side and loads 0 into M00~M01.



2.8.2. BSFR, BSFRP, BSFL, BSFLP

BSFR(P) instruction shifts the specified bits(D) towards the 1 bit right side and loads Source bit into bit 0 of Array.

BSFL(P) instruction shifts the specified bits(D) towards the 1 bit left side and loads Source bit into bit 0 of Array.



D	Source
n	Number of bits to move right(BSFR) or left(BSFL)

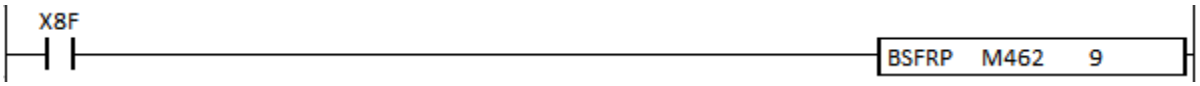
Instruction		Device address													No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
BSFR(P) BSFL(P)	D	o	-	o	o	o	-	-	-	-	-	-	-	-	3	o	-	o
	n	o	o	o	o	o	o	o	o	-	o	o	o	o				

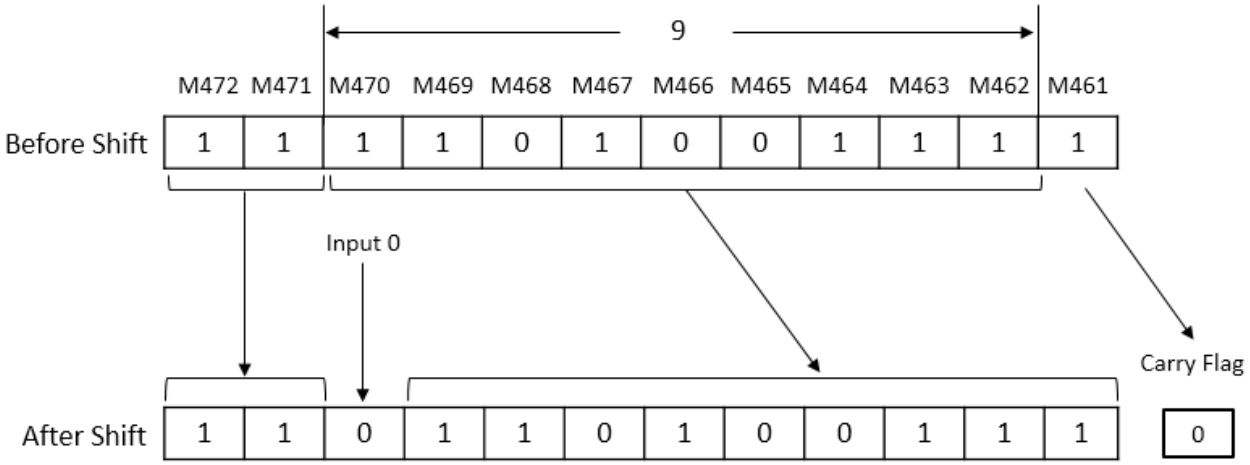
1) BSFR(P)

Format: BSFRP D(starting device address to move right) n(number of source device to move right)

Example)

• If X8F is ON, the BSFRP instruction shifts M462~M470 (9 device value) 1 bit right side and loads 0 into M470.



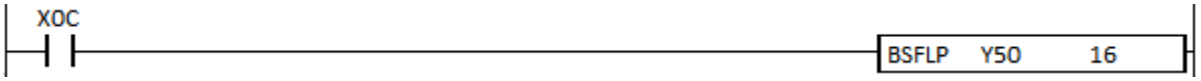


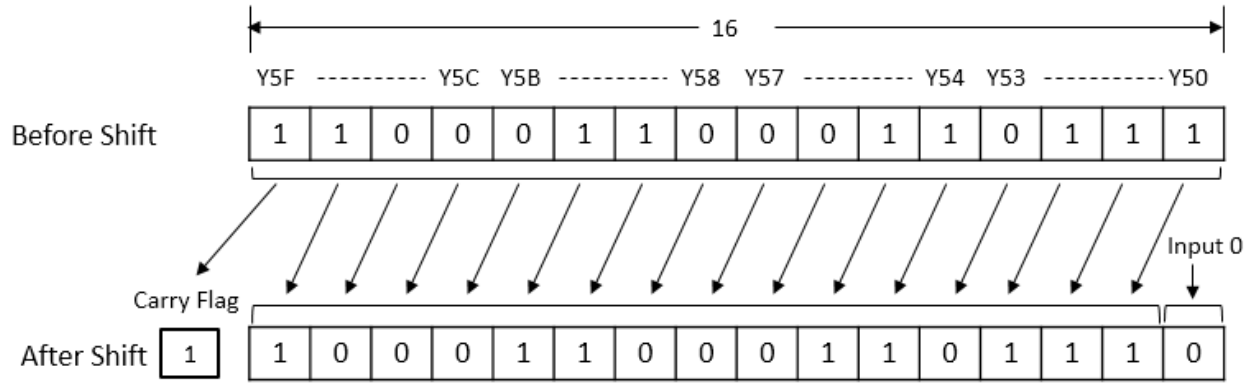
2) BSFL(P)

Format : BSFL D(starting device address to move left) n(number of source device to move left)

Example)

• If X0C is ON, the BSFLP instruction shifts Y50~Y5F (16 device value) 1 bit left side and loads 0 into Y50.

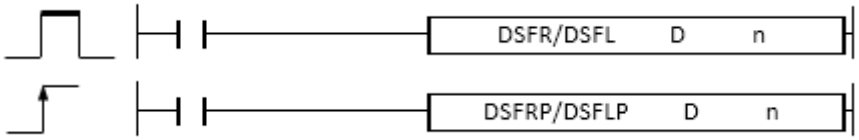




2.8.3. DSFR, DSFRP, DSFL, DSFLP

DSFR(P) instruction shifts the specified WORD data(D) towards the 1 WORD right side and loads Source Word into Word 0 of Array.

DSFL(P) instruction shifts the specified WORD data(D) towards the 1 WORD left side and loads Source Word into Word 0 of Array.



D	Source
n	Number of bits to move right(BSFR) or left(BSFL)

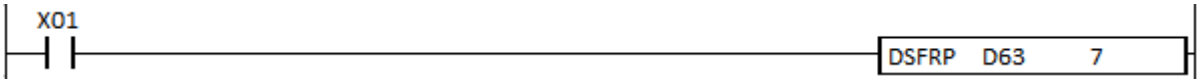
Instruction		Device address													No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
DSFR(P) DSFL(P)	D	o	-	o	o	o	-	o	o	-	o	o	o	-	3	o	-	o
	n	o	o	o	o	o	o	o	o	-	o	o	o	o				

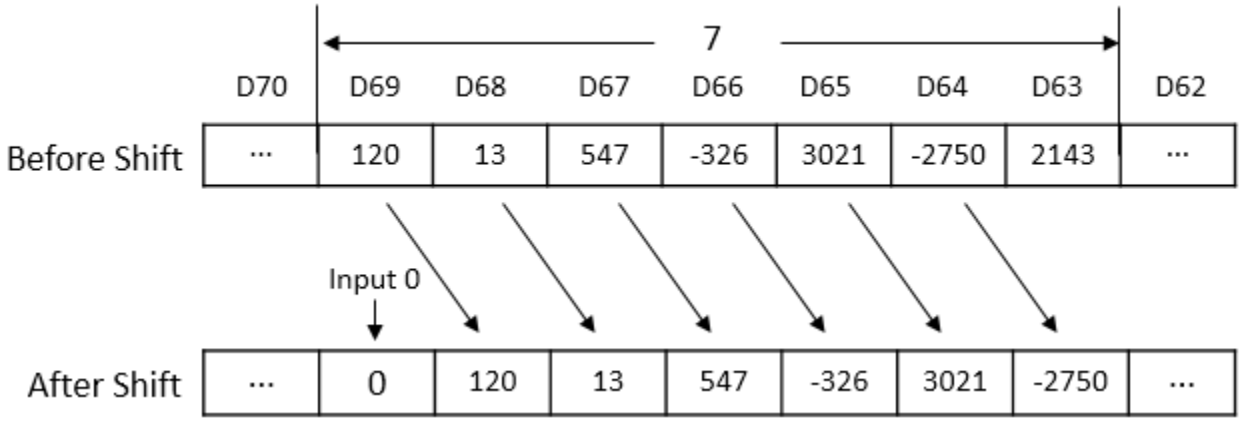
1) DSFR(P)

Format : DSFRP D(starting device address to move right) n(number of source device to move right)

Example)

• If X01 is ON, the DSFRP instruction shifts D63~D69 (7 device value) 1 Word right side and loads 0 into D69.



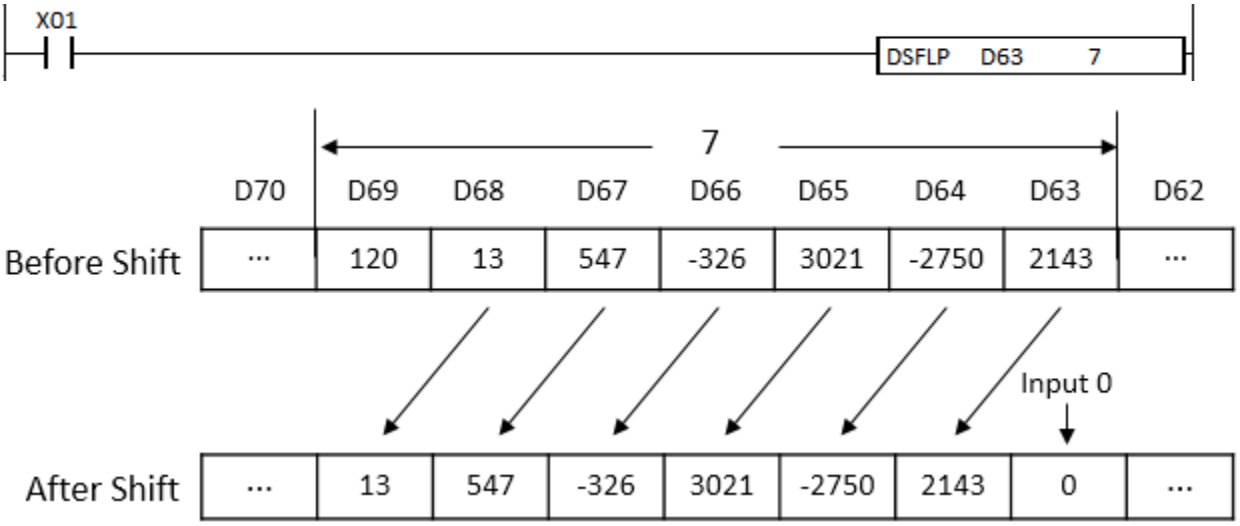


2) DSFL(P)

Format : DSFL D(starting device address to move left) n(number of source device to move left)

Example)

• If X01 is ON, the DSFLP instruction shifts D63~D69 (7 device value) 1 Word left side and loads 0 into D63.

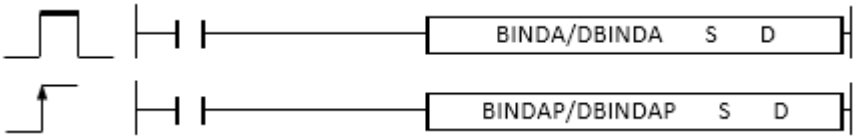


2.9. Character String Processing Instruction

2.9.1. BINDA, BINDAP, DBINDA, DBINDAP

BINDA instruction converts a BIN 16bit value to a decimal (ASCII code) and saves the result in Destination (assigned device address).

- DBINDA and DBINDAP are Double Word (32bit).

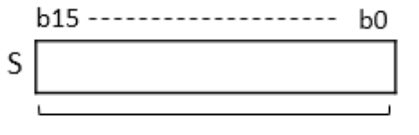


S	Source
D	Starting device address of the Destination

Instruction		Device address													No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
BINDA(P) DBINDA(P)	S	o	o	o	o	o	o	o	o	-	o	o	o	o	3	o	-	-
	D	o	-	o	o	o	-	o	o	-	o	o	o	-				

1) BINDA(P)

The BINDA instruction converts 16bit BIN to a decimal ASCII code and places the result in the Destination.

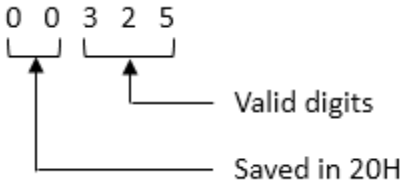


	b15 ---------- b8	b7 ---------- b0
D	ASCII code in the ten thousands place	Sign
D+1	ASCII code in the hundreds place	ASCII code in the thousands place
D+2	ASCII code in the ones place	ASCII code in the tens place

- The range of 16bit BIN data is -32768 ~ 32767.

- If BIN data is +, sign is saved as 20H(space) and if BIN data is -, sign is saved as 2DH(-).

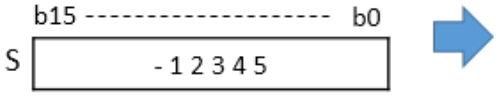
The “0” is saved as 20H(space).



- The “0” is saved in D+3.

Example)

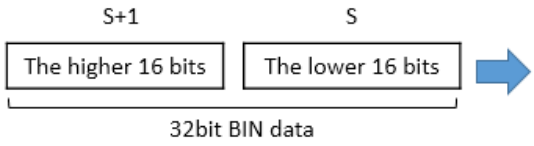
• If the S value is -12345, its decimal ASCII code is saved in D~D+2.



	b15 ---------- b8	b7 ---------- b0
D	31H (1)	2DH (-)
D+1	33H (3)	32H (2)
D+2	35H (5)	34H (4)

2) DBINDA(P)

The DBINDA instruction converts 32bit BIN to a decimal ASCII code and places the result in the Destination.

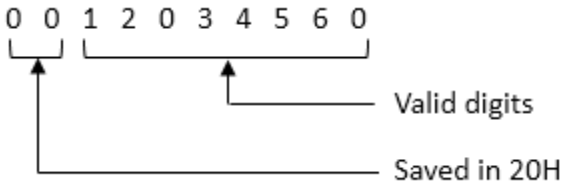


	b15 ---------- b8	b7 ---------- b0
D	ASCII code in the billions place	Sign
D+1	ASCII code in the ten millions place	ASCII code in the hundred millions place
D+2	ASCII code in the hundred thousands place	ASCII code in the millions place
D+3	ASCII code in the thousands place	ASCII code in the ten thousands place
D+4	ASCII code in the tens place	ASCII code in the hundreds place
D+5	20H	ASCII code in the ones place

- The range of 32bit BIN data is -2147483648 ~ 2147483647.

- If BIN data is +, sign is saved as 20H(space) and if BIN data is -, sign is saved as 2DH(-).

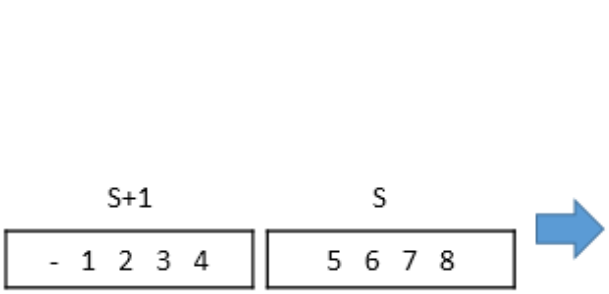
The “0” is saved as 20H(space).



- The “0” is saved in higher bit of D+5.

Example)

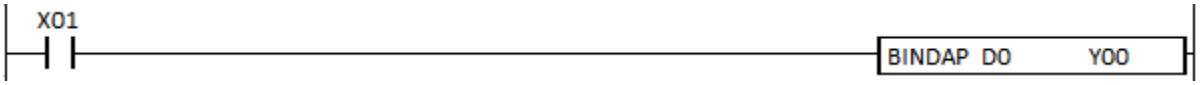
• If the S value is -12345678, its decimal ASCII code is saved in D~D+5.



	b15 ---------- b8	b7 ---------- b0
D	20H (space)	2DH (-)
D+1	31H (1)	20H (space)
D+2	33H (3)	32H (2)
D+3	35H (5)	34H (4)
D+4	37H (7)	36H (6)
D+5	20H	38H (8)

Example)

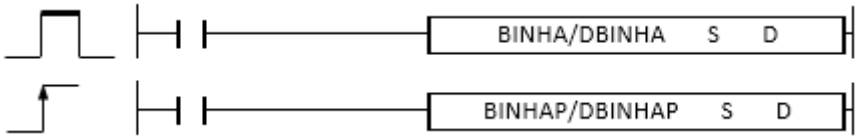
If X01 is ON, BINDAP converts a BIN 16bit value of D0 to a decimal (ASCII code) and save the result in Y00.



2.9.2. BINHA, BINHAP, DBINHA, DBINHAP

BINHA instruction converts a BIN 16bit value to a Hexadecimal (ASCII code) and saves the result in Destination (assigned device address).

- DBINHA and DBINHAP are Double Word (32bit).

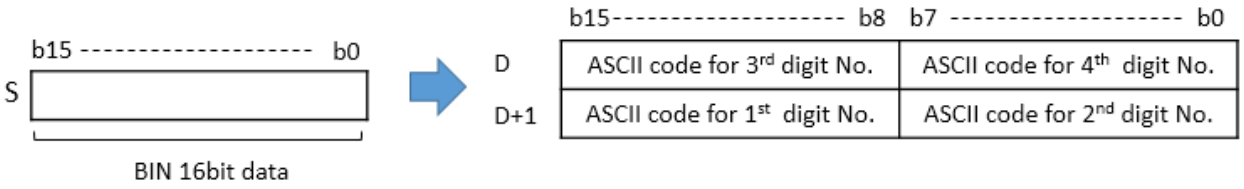


S	Source
D	Starting device address of the Destination

Instruction		Device address													No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
BINHA(P) DBINHA(P)	S	o	o	o	o	o	o	o	o	-	o	o	o	o	3	o	-	-
	D	o	-	o	o	o	-	o	o	-	o	o	o	-				

1) BINHA(P)

The BINHA instruction converts 16bit BIN to a hexadecimal ASCII code and places the result in the Destination.

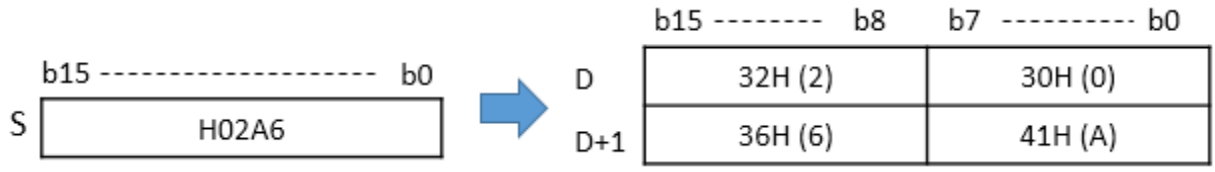


- The range of 16bit BIN data is 0 ~ FFFFH.

- The “0” is saved in D+2.

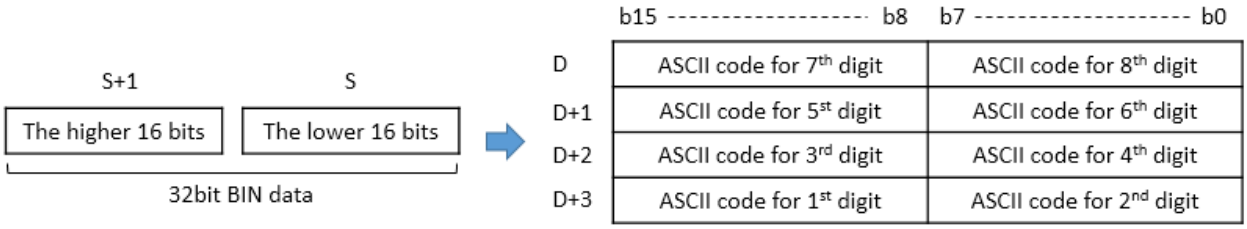
Example)

• If the S value is 02A6H, its ASCII code is saved in D~D+1.



2) DBINHA(P)

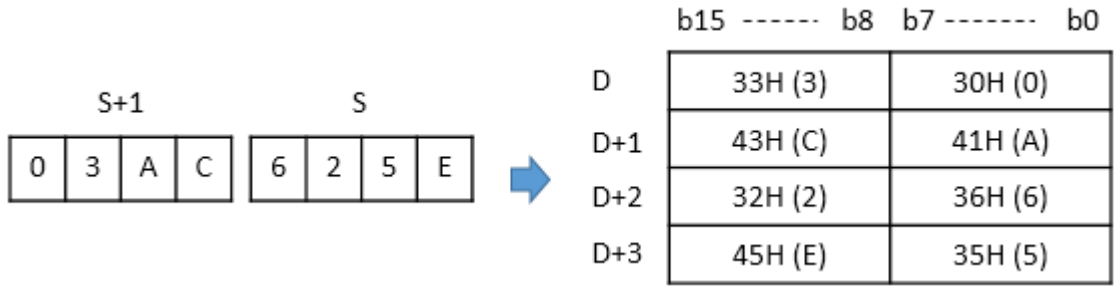
The DBINHA instruction converts 32bit BIN to a Hexadecimal ASCII code and places the result in the Destination.



- The range of 32bit BIN data is 0 ~ FFFFFFFFH.

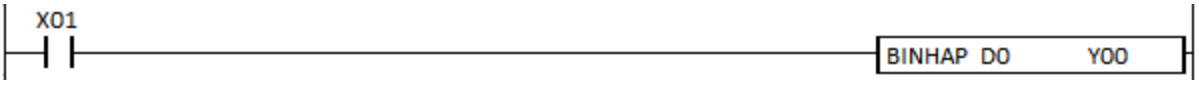
Example)

• If the S value is 03AC625EH, its Hexadecimal ASCII code is saved in D~D+3.



Example)

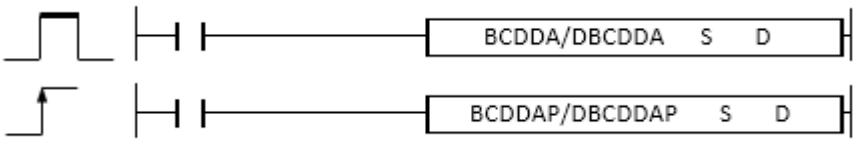
If X01 is ON, BINHAP converts a BIN 16bit value of D0 to a Hexadecimal (ASCII code) and save the result in Y00.



2.9.3. BCDDA, BCDDAP, DBCDDA, DBCDDAP

BCDDA instruction converts a BCD value to an ASCII code and saves the result in Destination (assigned device address).

- DBCDDA and DBCDDAP are Double Word (32bit).

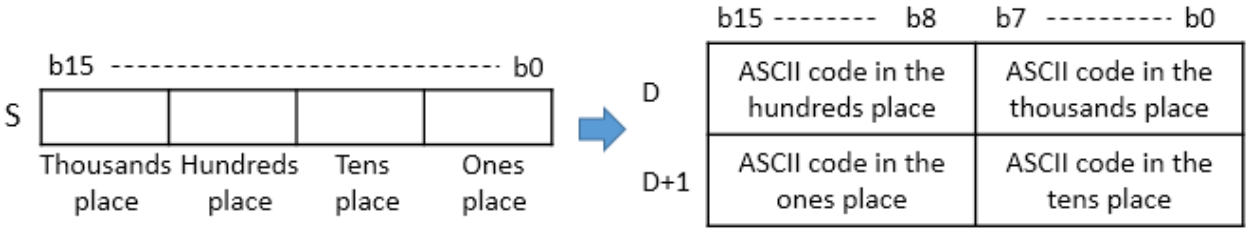


S	Source (BCD)
D	Starting device address of the Destination

Instruction		Device address													No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
BCDDA(P) DBCDDA(P)	S	o	o	o	o	o	o	o	o	-	o	o	o	o	3	o	-	-
	D	o	-	o	o	o	-	o	o	-	o	o	o	-				

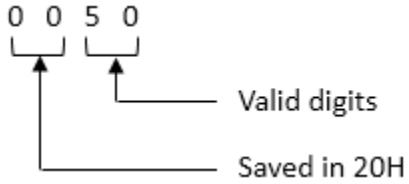
1) BCDDA(P)

The BCDDA instruction converts a BCD value to an ASCII code and places the result in the Destination.



- The range of BCD data is 0 ~ 9999.

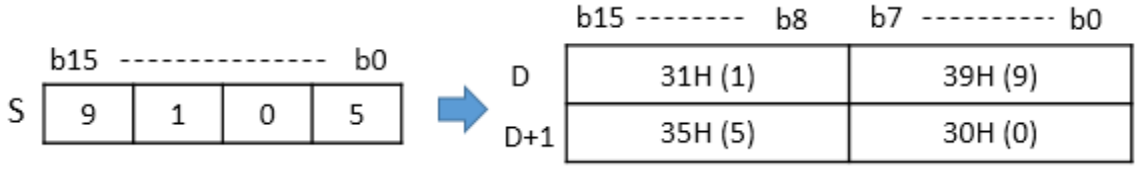
- The “0” is saved as 20H.



- The “0” is saved in D+2.

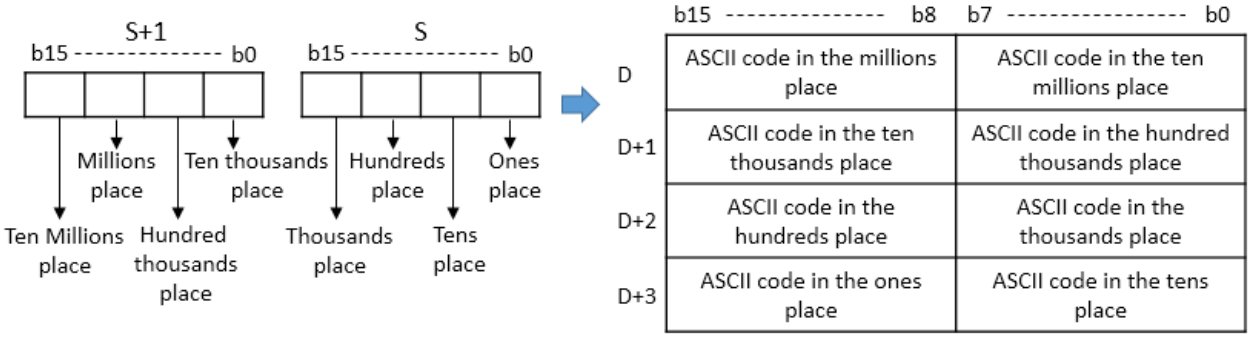
Example)

• If the S value is 9105, its ASCII code is saved in D~D+1.



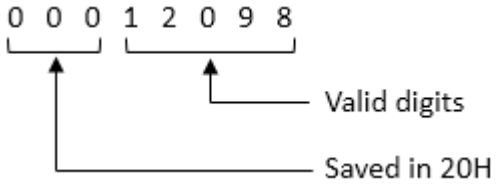
2) DBCDDA(P)

The DBCDDA instruction converts 32bit BCD to an ASCII code and places the result in the Destination.



- The range of 32bit BCD data is 0 ~ 99999999.

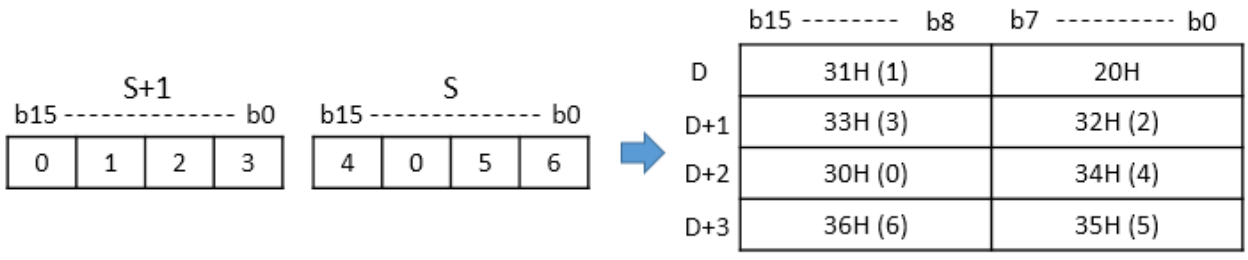
- The “0” is saved as 20H.



- The “0” is saved in D+4.

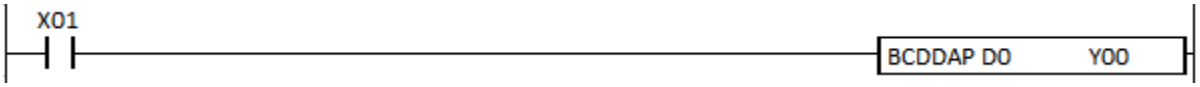
Example)

• If the S value is 01234056, its ASCII code is saved in D~D+3.



Example)

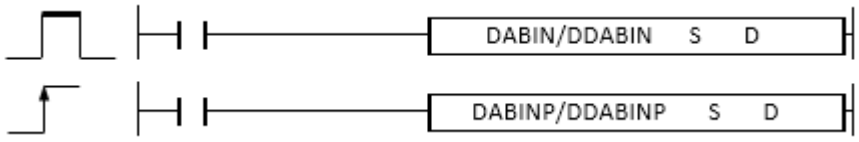
If X01 is ON, BCDDAP converts a BCD value of D0 to an ASCII code and save the result in Y00.



2.9.4. DABIN, DABINP, DDABIN, DDABINP

DABIN instruction converts an ASCII code to a decimal value and saves the 16bit binary result in Destination (assigned device address).

- DDABIN and DDABINP are Double Word (32bit).

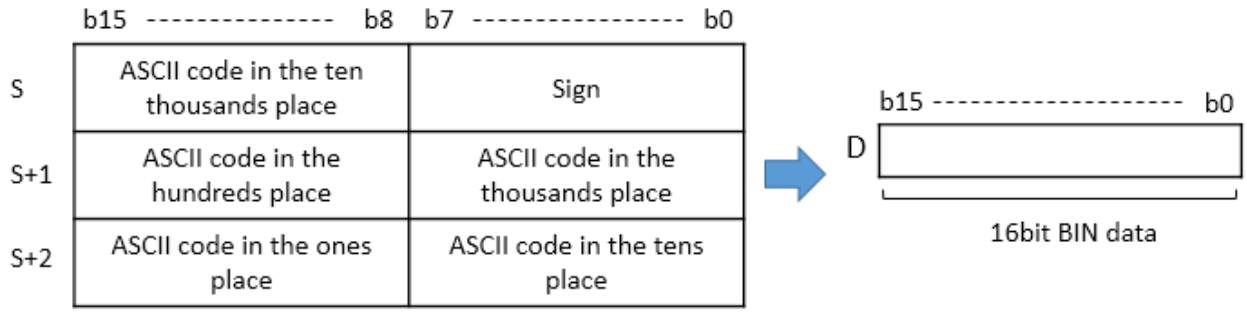


S	Source (Decimal ASCII code)
D	Starting device address of the Destination

Instruction		Device address													No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
DABIN(P) DDABIN(P)	S	o	o	o	o	o	o	o	o	-	o	o	o	o	3	o	-	-
	D	o	-	o	o	o	-	o	o	-	o	o	o	-				

1) DABIN(P)

The DABIN instruction converts an ASCII code to a decimal value and places the 16bit binary result in the Destination.



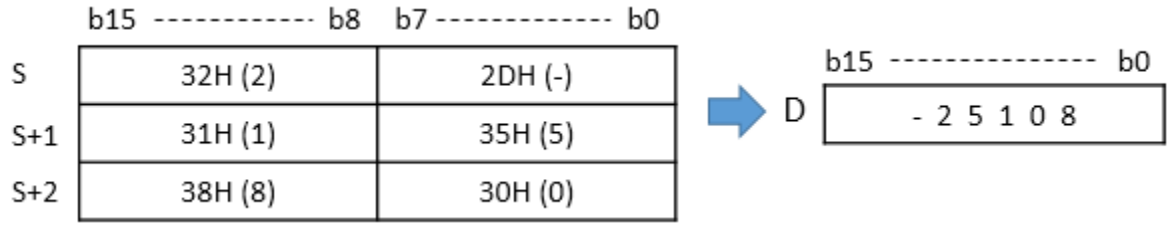
- The range of Decimal data is -32768 ~ 32767.

- If BIN data is +, sign is saved as 20H(space) and if BIN data is -, sign is saved as 2DH(-).

- If the position of ASCII code is 20H and 00H each, it becomes 30H.

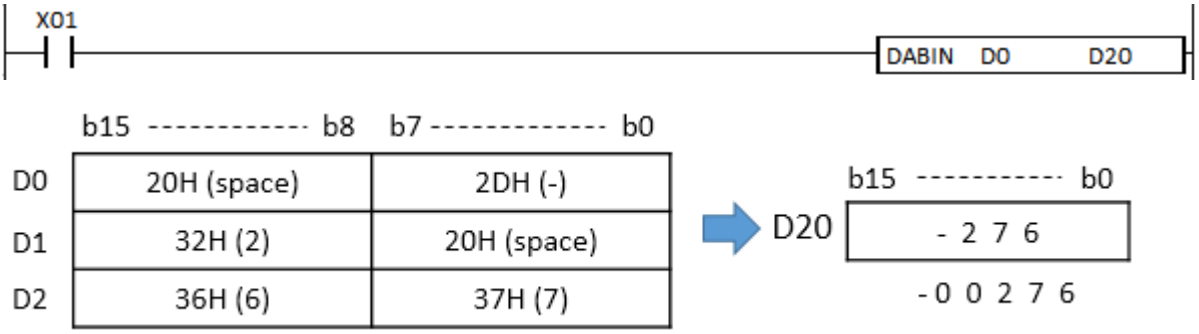
Example)

• If the ASCII code of S is -25108, its decimal value is saved in D.



Example)

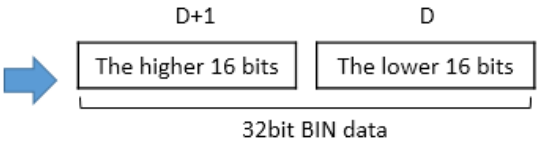
If X01 is ON, DABIN converts an ASCII code of D0~D2 to a decimal value and save the result -276 in D20.



2) DDABIN(P)

The DDABIN instruction converts 32bit ASCII code to a decimal value and places the result in the Destination.

	b15 ---------- b8	b7 ---------- b0
S	ASCII code in the billions place	Sign
S+1	ASCII code in the ten millions place	ASCII code in the hundred millions place
S+2	ASCII code in the hundred thousands place	ASCII code in the millions place
S+3	ASCII code in the thousands place	ASCII code in the ten thousands place
S+4	ASCII code in the tens place	ASCII code in the hundreds place
S+5	(Ignored)	ASCII code in the ones place



- The range of Decimal data is -2147483648 ~ 2147483647.

- If BIN data is +, sign is saved as 20H(space) and if BIN data is -, sign is saved as 2DH(-).

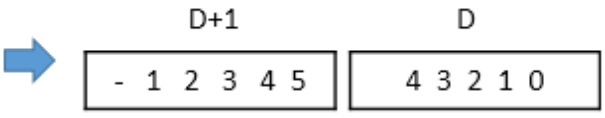
- The data in higher byte of S+5 is ignored.

- If the position of ASCII code is 20H and 00H each, it becomes 30H.

Example)

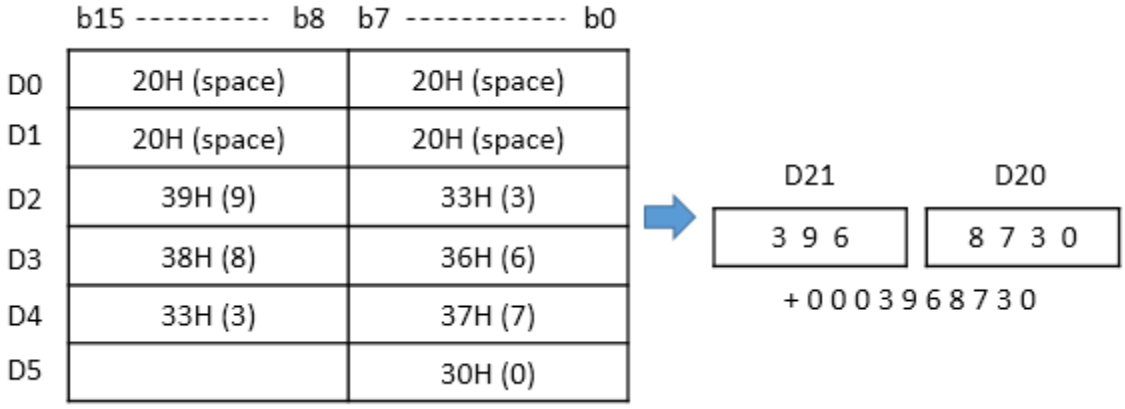
• If the ASCII code of S is -123454321, its decimal value is saved in D and D+1.

	b15 ---------- b8	b7 ---------- b0
S	31H (1)	2DH (-)
S+1	33H (3)	32H (2)
S+2	35H (5)	34H (4)
S+3	33H (3)	34H (4)
S+4	31H (1)	32H (2)
S+5		30H (0)



Example)

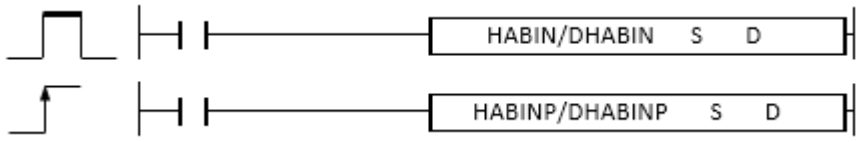
If X01 is ON, DDABIN converts an ASCII code of D0~D5 to a decimal value and save the result 3968730 in D20 and D21.



2.9.5. HABIN, HABINP, DHABIN, DHABINP

HABIN instruction converts an ASCII code to a Hexadecimal value and saves the 16bit binary result in Destination (assigned device address).

- DHABIN and DHABINP are Double Word (32bit).

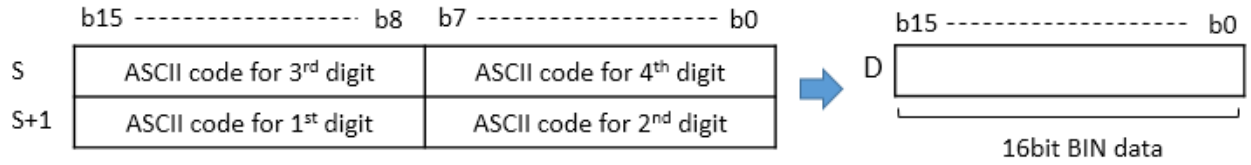


S	Source (Hexadecimal ASCII code)
D	Starting device address of the Destination

Instruction		Device address													No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
HABIN(P) DHABIN(P)	S	o	o	o	o	o	o	o	o	-	o	o	o	o	3	o	-	-
	D	o	-	o	o	o	-	o	o	-	o	o	o	-				

1) HABIN(P)

The HABIN instruction converts an ASCII code to a Hexadecimal value and places the 16bit binary result in the Destination.

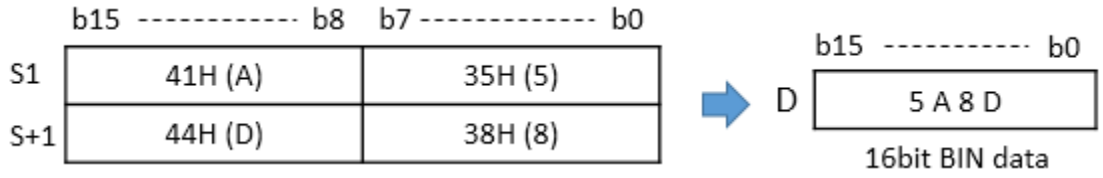


- The range of Hexadecimal data is 0000H ~ FFFFH.

- If BIN data is +, sign is saved as 20H(space) and if BIN data is -, sign is saved as 2DH(-).

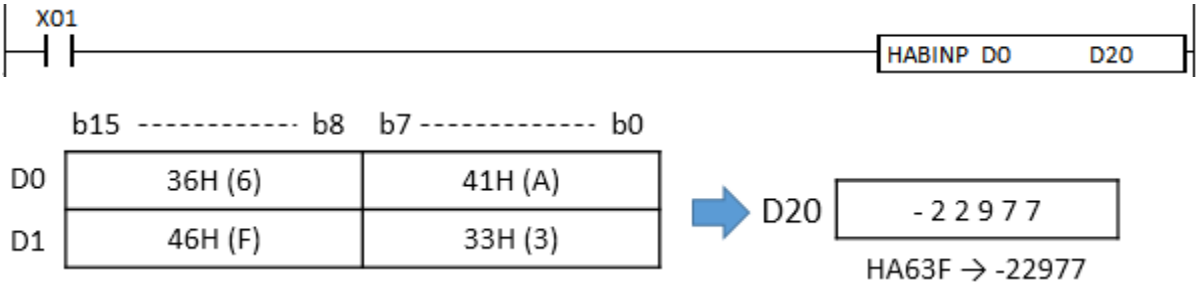
Example)

• If the ASCII code of S is 5A8DH, its hexadecimal value is saved in D.



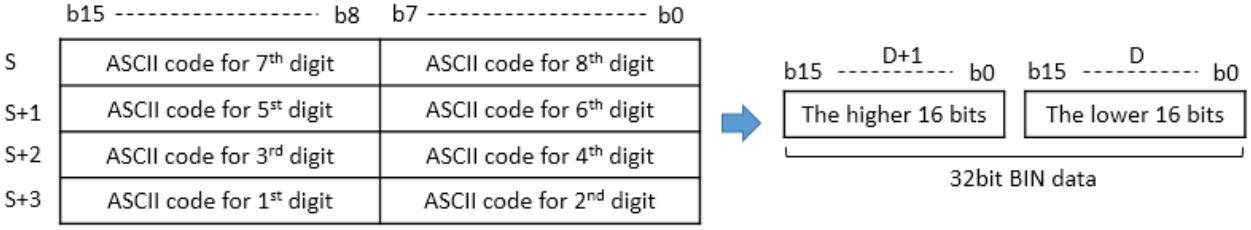
Example)

If X01 is ON, HABIN converts an ASCII code of D0~D1 to a hexadecimal value(HA63F) and save the decimal value -272977 in D20.



2) DHABIN(P)

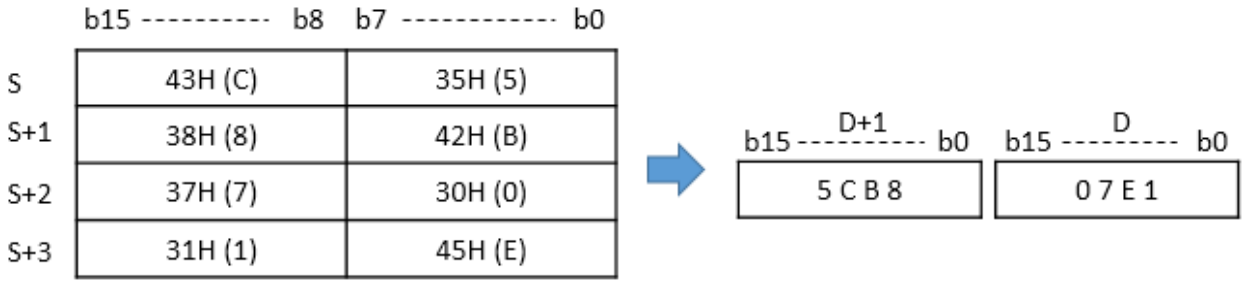
The DHABIN instruction converts 32bit ASCII code to a hexadecimal value and places the result in the Destination.



- The range of Hexadecimal data is 00000000H ~ FFFFFFFFH.

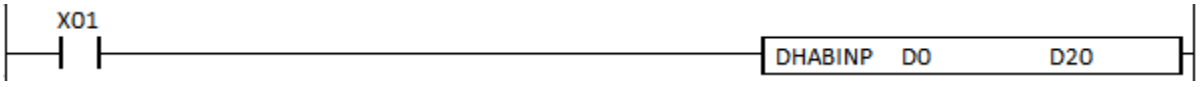
Example)

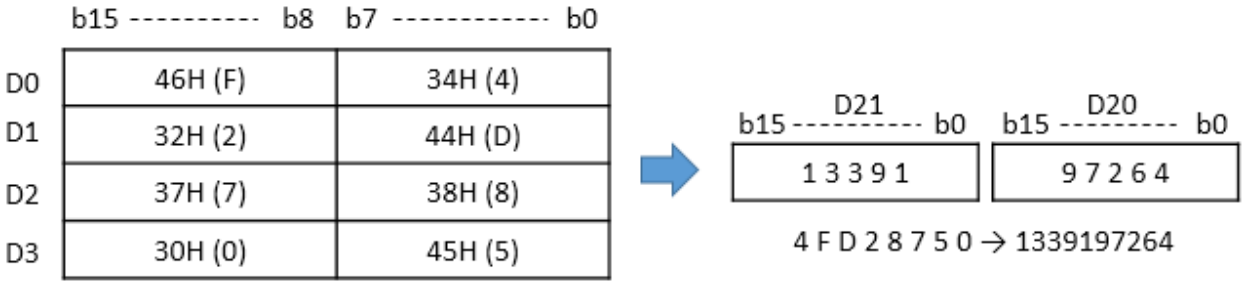
• If the ASCII code of S is 5CB807E1H, its hexadecimal value is saved in D and D+1.



Example)

If X01 is ON, DHABIN converts an ASCII code of D0~D3 to a hexadecimal value and save the decimal value 1339197264 in D20 and D21.



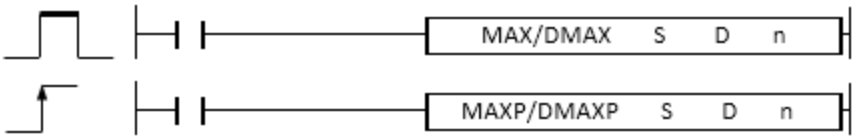


2.10. Data Processing Instruction

2.10.1. MAX, MAXP, DMAX, DMAXP

MAX instruction searches the maximum value from the source and saves the maximum value, the position number of the maximum value, and the total number of the maximum value to the Destination (assigned device address).

- DMAX and DMAXP are double words.



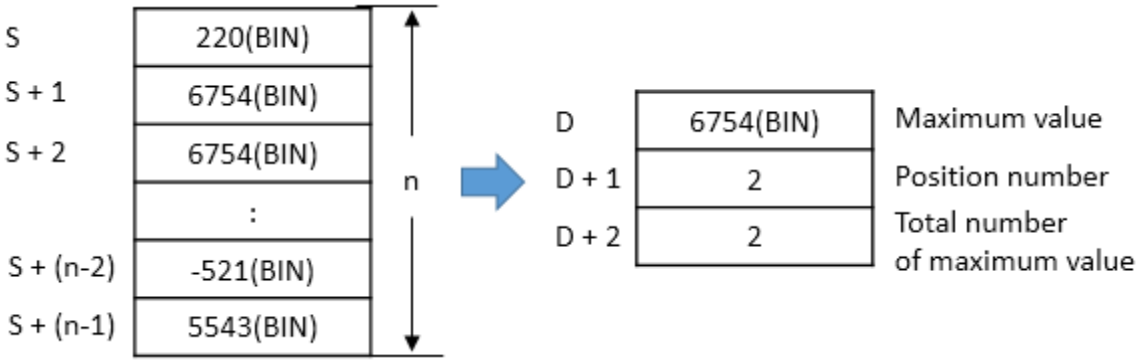
S	Source
D	Destination
n	Number of data block to search

Instruction		Device address													No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
MAX MAXP DMAX DMAXP	S	o	o	o	o	o	o	o	o	-	o	o	o	-	4	o	-	-
	D	o	-	o	o	o	-	o	o	-	o	o	o	-				
	n	o	o	o	o	o	o	o	o	-	o	o	o	-				

1) MAX

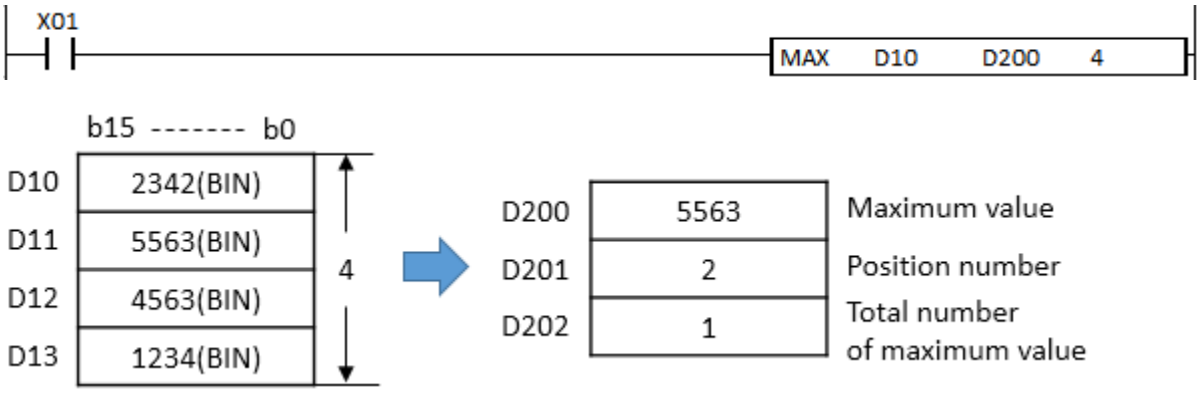
Functions

- MAX(P) searches the maximum value from the 16 bit data source and saves the maximum value (6575) to D, the position number of the maximum value 2 (S+1, 2nd block from S) to D+1, and the total number of the maximum value (2) to D+2.



Example)

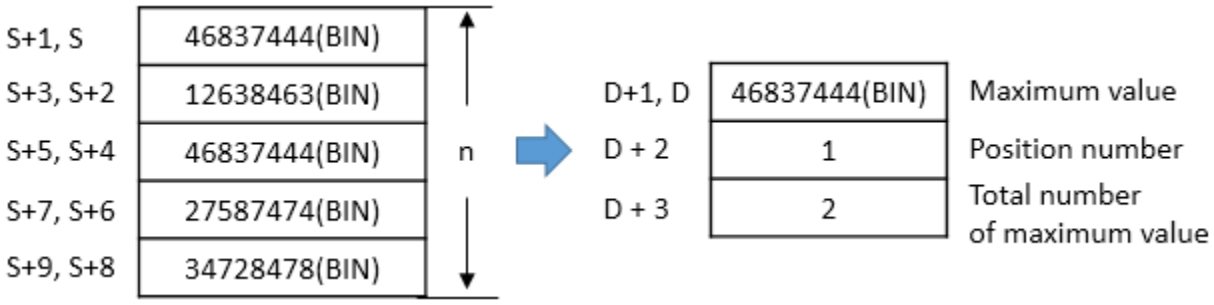
• If X01 is ON, MAX instruction searches maximum values from D10 to D13 (4 blocks) and saves the maximum value 5563 to D200, the position number 2 (D11, 2nd block from D10) where the maximum number is located to D201, and the total number of the maximum value 1 to D202.



2) DMAX

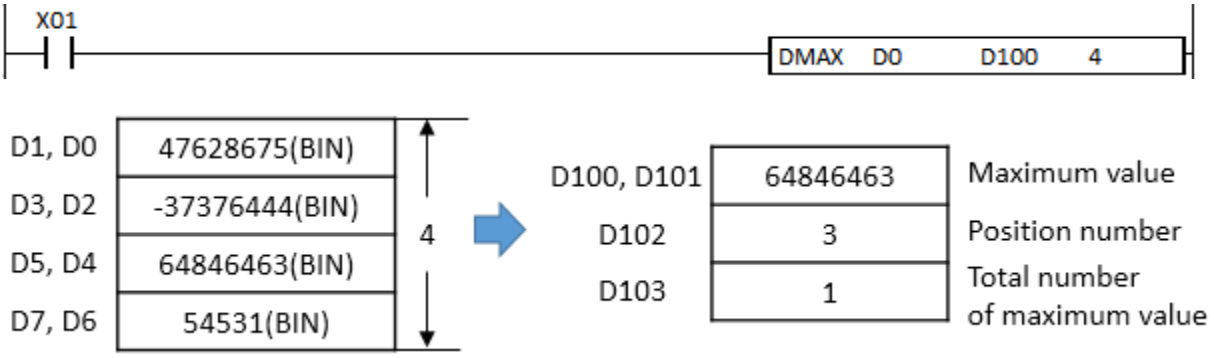
Functions

- DMAX(P) searches the maximum value from the 32 bit data source and saves the maximum value (46837444) to D and D+1, the position number of the maximum value (1) to D+2, and the total number of the maximum value (2) to D+3.



Example)

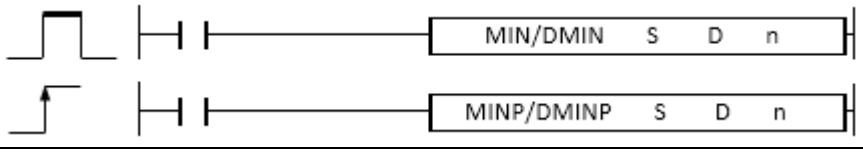
• If X01 is ON, DMAX instruction searches maximum values from D0 to D7 (4 blocks) and save the maximum value 64846463 to D100 and D101, position number 3 (D4 and D5, 3rd block from D0) where the maximum number is located to D102, and the total number of the maximum value 1 to D103.



2.10.2. MIN, MINP, DMIN, DMINP

MIN instruction searches the minimum value from the source and saves the minimum value, the position number of the minimum value, and the total number of the minimum value to the Destination (assigned device address).

- DMIN and DMINP are double words.

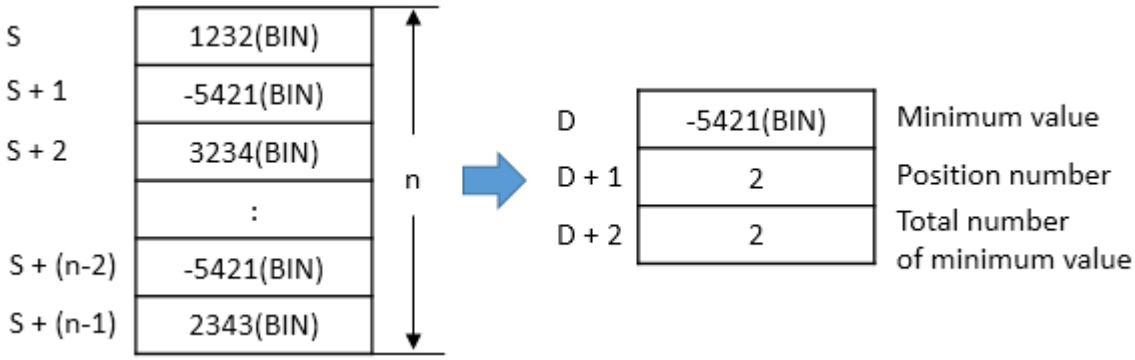


S	Source
D	Destination
n	Number of data block to search

Instruction		Device address													No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
MIN MINP DMINX DMINP	S	o	o	o	o	o	o	o	o	-	o	o	o	-	4	o	-	-
	D	o	-	o	o	o	-	o	o	-	o	o	o	-				
	n	o	o	o	o	o	o	o	o	-	o	o	o	-				

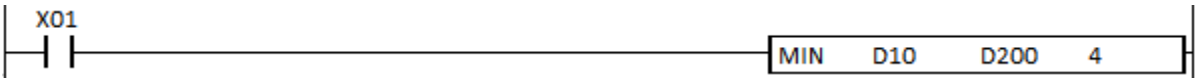
1) MIN

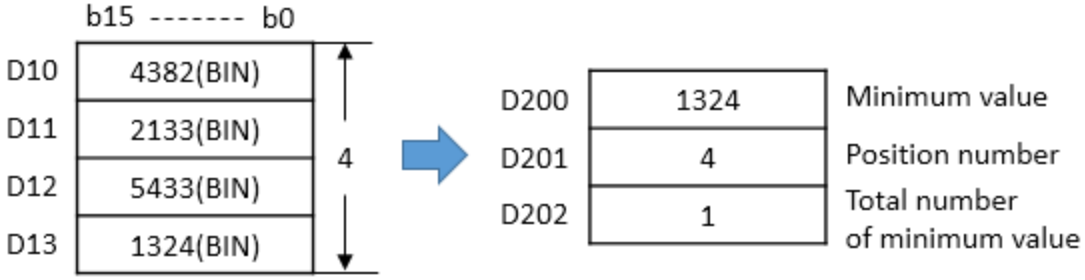
- MIN(P) searches the minimum value from the 16 bit data source and saves the minimum value (-5421) to D, the position number of the minimum value 2 (S+1, 2nd block from S) to D+1, and the total number of the minimum value (2) to D+2.



Example)

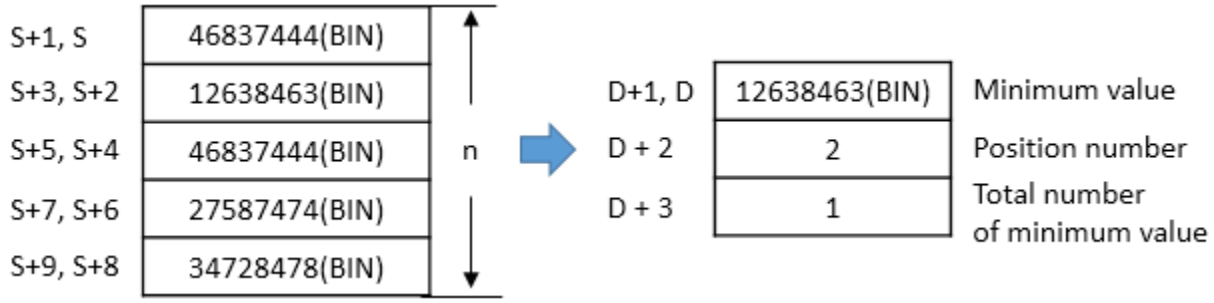
• If X01 is ON, MIN instruction searches minimum values from D10 to D13 (4 blocks) and saves the minimum value 1324 to D200, the position number 4 (D13, 4th block from D10) where the minimum number is located to D201, and the total number of the minimum value 1 to D202.





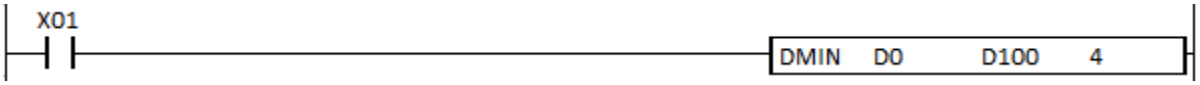
2) DMIN

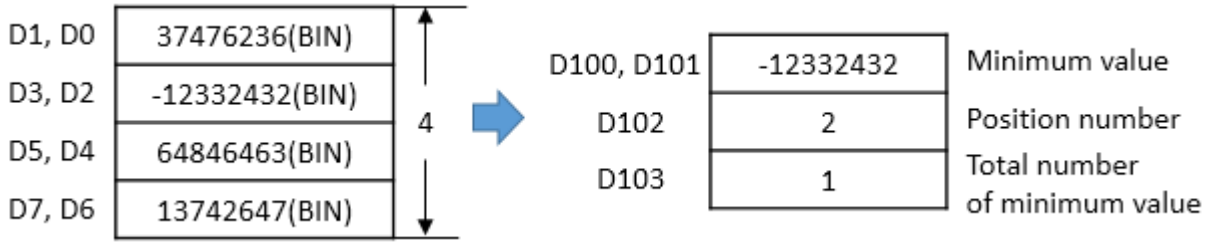
- DMIN(P) searches the minimum value from the 32 bit data source and saves the minimum value (12638463) to D and D+1, the position number of the minimum value 2 (S+2 and S+3, 2nd block from S) to D+2, and the total number of the minimum value (1) to D+3.



Example)

• If X01 is ON, DMIN instruction searches minimum values from D0 to D7 (4 blocks) and saves the minimum value -12332432 to D100 and D101, the position number 2 (D2and D3, 2nd block from D0) where the minimum number is located to D102, and the total number of the minimum value 1 to D103.

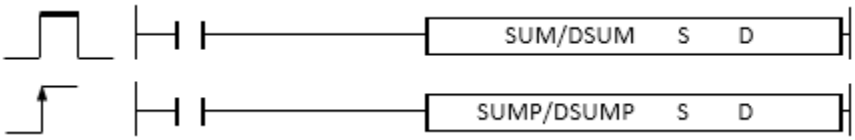




2.10.3. SUM, SUMP, DSUM, DSUMP

SUM instruction counts 1(bit=1) in 16bit data source and saves the result in Destination.

- DSUM and DSUMP are double words.



S	Source
D	Destination

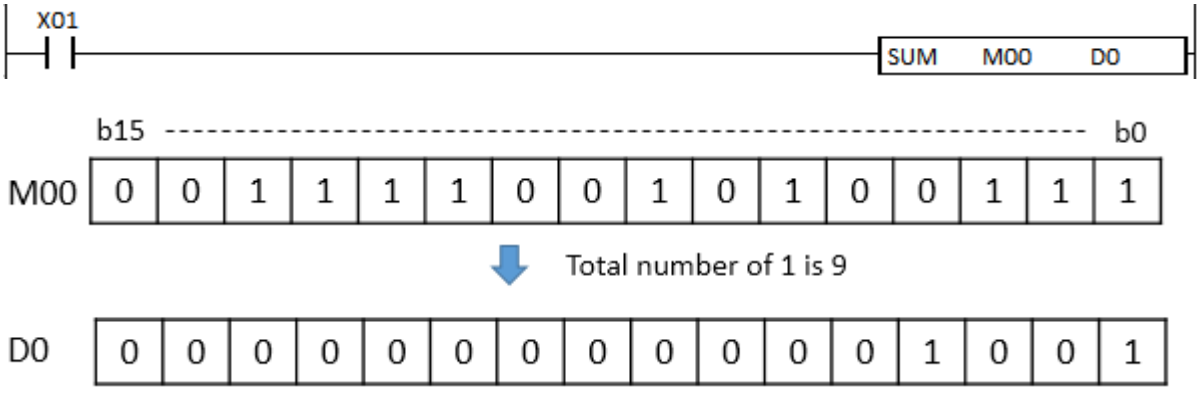
Instruction		Device address													No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
SUM(P) DSUM(P)	S	o	o	o	o	o	o	o	o	-	o	o	o	o	3	o	-	-
	D	o	-	o	o	o	-	o	o	-	o	o	o	-				

1) SUM

- When enabled, the SUM instruction counts 1 in 16bit data source and places the result in the Destination.

Example)

• If X01 is ON, SUM instruction counts 1 in M00 and save the result 9 in D0.

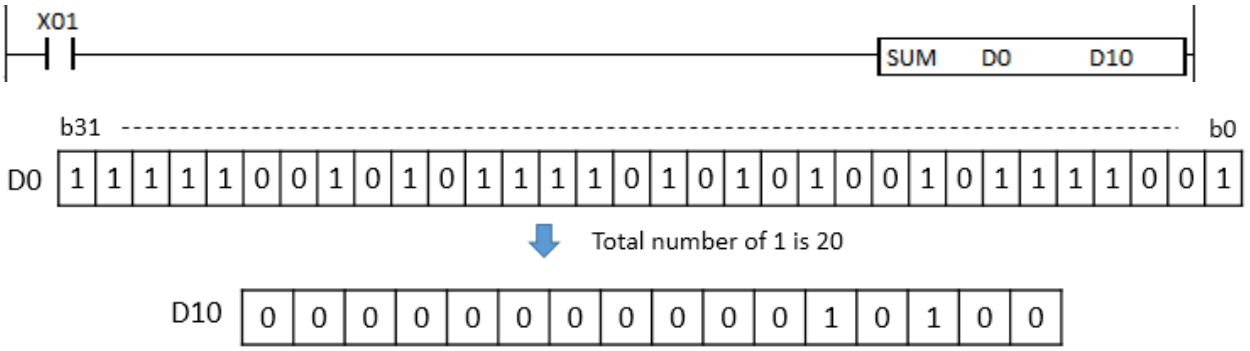


2) DSUM

- When enabled, the DSUM instruction counts 1 in 32bit data source and places the result in the Destination.

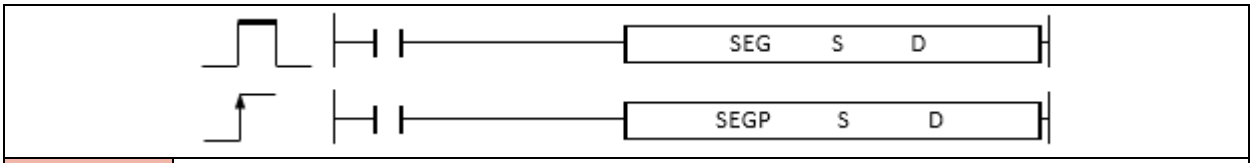
Example)

• If X01 is ON, DSUM instruction counts 1 in D0 (double words) and saves the result 20 in D10.



2.10.4. SEG, SEGP

SEG instruction converts source data to 7-segment code and saves the code into the byte of Destination.

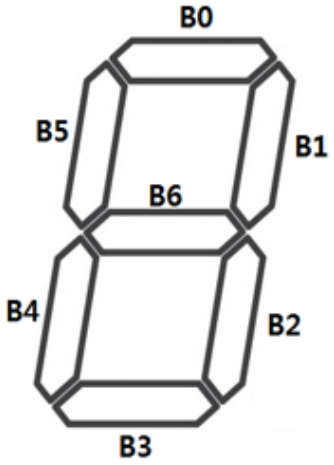


S	Source
D	Destination

Instruction		Device address													No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
SEG SEGP	S	o	o	o	o	o	o	o	o	-	o	o	o	o	3	o	-	-
	D	o	-	o	o	o	-	o	o	-	o	o	o	-				

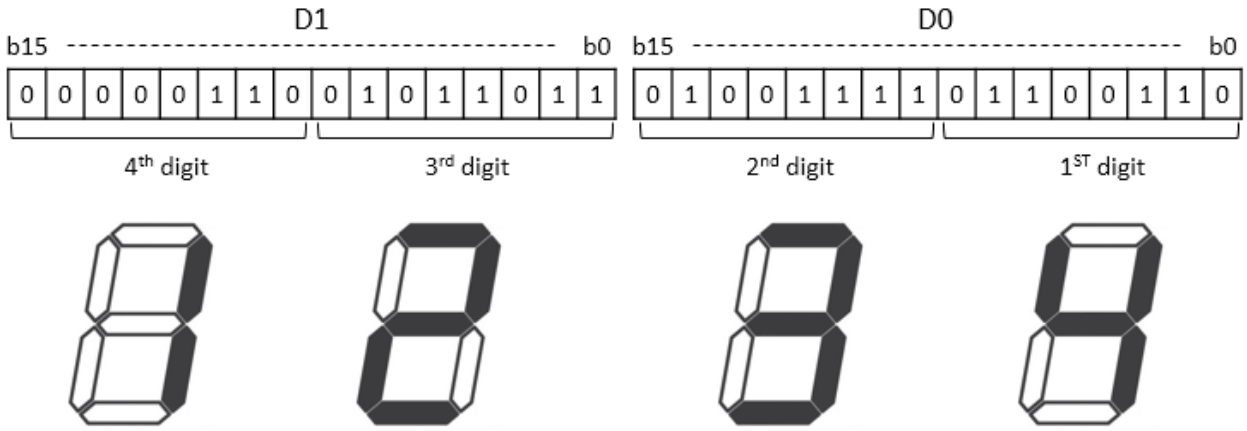
- When enabled, the SEG instruction converts data source to 7-segment code and places the code in the Destination.

• 7-Segment Display Table

Source		7-Segment Display	Byte of D							
Hex	Bit		B7	B6	B5	B4	B3	B2	B1	B0
0	0000		0	0	1	1	1	1	1	1
1	0001		0	0	0	0	0	1	1	0
2	0010		0	1	0	1	1	0	1	1
3	0011		0	1	0	0	1	1	1	1
4	0100		0	1	1	0	0	1	1	0
5	0101		0	1	1	0	1	1	0	1
6	0110		0	1	1	1	1	1	0	1
7	0111		0	0	1	0	0	1	1	1
8	1000		0	1	1	1	1	1	1	1
9	1001		0	1	1	0	1	1	1	1
A	1010		0	1	1	1	0	1	1	1
B	1011		0	1	1	1	1	1	0	0
C	1100		0	0	1	1	1	0	0	1
D	1101		0	1	0	1	1	1	1	0
E	1110		0	1	1	1	1	0	0	1
F	1111		0	1	1	1	0	0	0	1

Example)

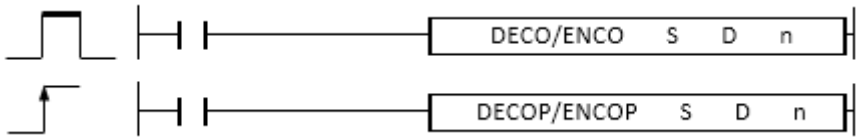
• If X01 is ON, the SEG instruction converts H1234 to 7-segments code and places to D0 and D1.



2.10.5. DECO, DECOP, ENCO, ENCOP

• DECO instruction decodes (n) number of source and sets the decoded bit(2^n) among the total of bits in the Destination.

• ENCO instruction encodes source as 2^n and sets the encoded bit(n) to the Destination (encoding start point is b0).



S	Source
D	Destination
n	Number of bit to be decoded or encoded

Instruction		Device address													No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
DECO(P) ENCO(P)	S	o	o	o	o	o	o	o	o	-	o	o	o	o	4	o	-	-
	D	o	-	o	o	o	-	o	o	-	o	o	o	-				
	n	o	o	o	o	o	o	o	o	-	o	o	o	o				

1) DECO, DECOP

- When enabled, the DECO instruction decodes (n) number of source and sets the decoded bit (2^n) among the total of bits in the Destination.

- You can choose 1~8 for (n).

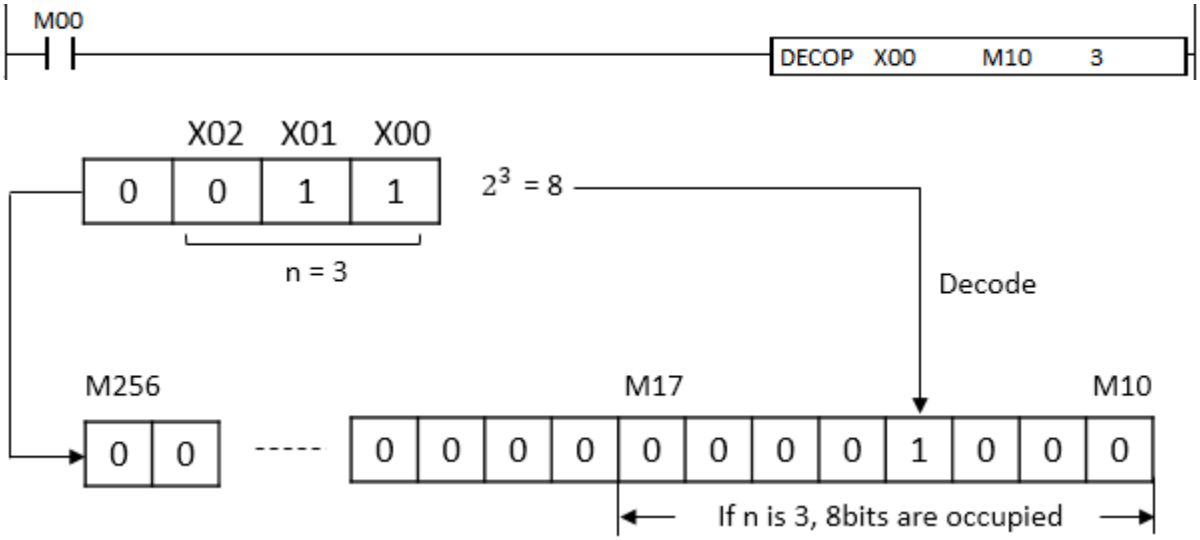
- If n is 0, there will be no change in the Destination.

Example)

• If M00 is ON, DECOP instruction decodes 3 number of source (X00, X01, and X02).

As X00 and X01 are 1, the decimal value is 3 so decoded value 2^3 is 8.

In 16bit binary, 8 is 4th bit. Therefore, M13 is set to 1.



2) ENCO, ENCOP

- When enabled, the ENCO instruction encodes source as 2^n and sets the encoded bit(n) to the Destination (encoding start point is b0).

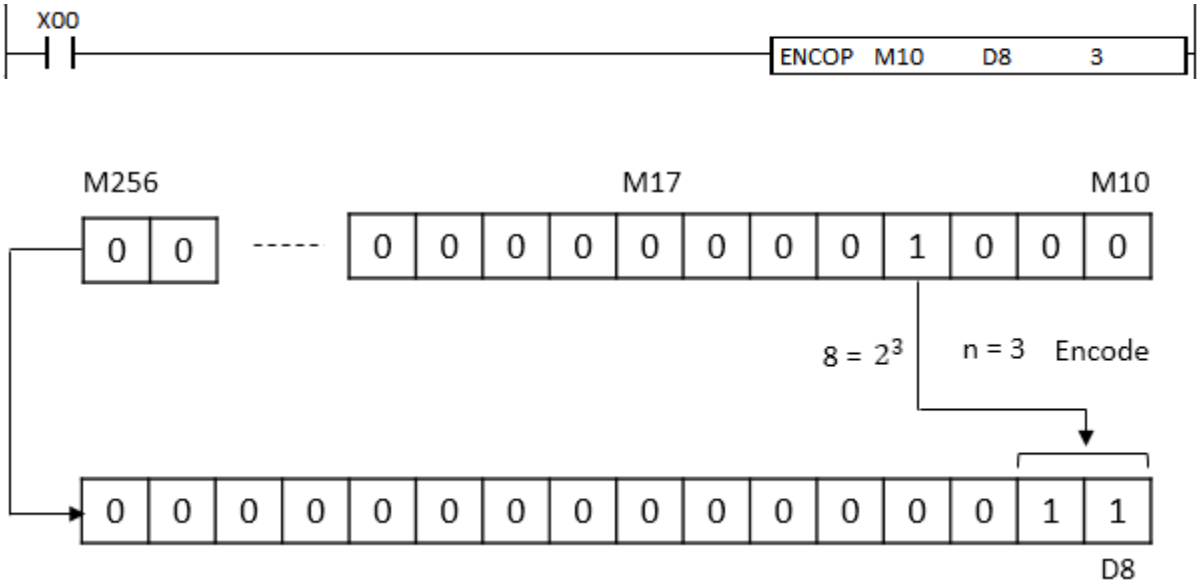
- You can choose 1~8 for (n).

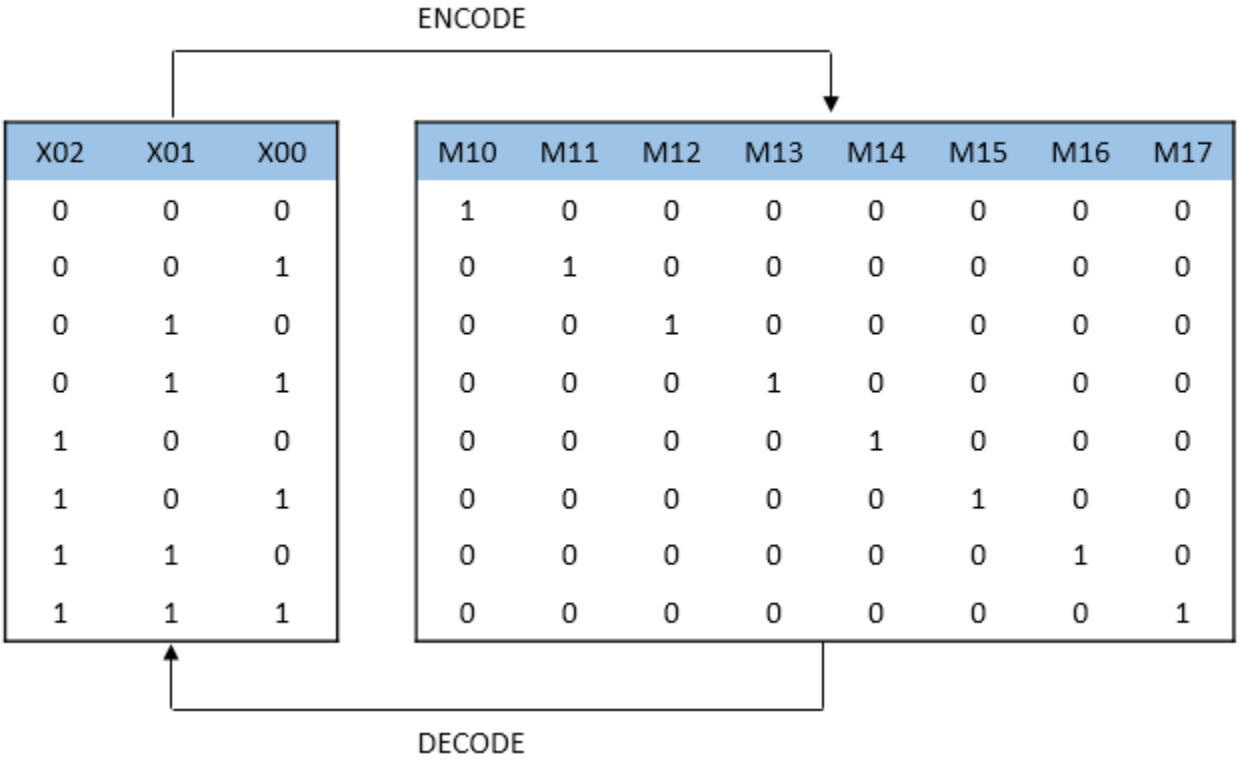
- If n is 0, there will be no change in the Destination.

Example)

• If X00 is ON, ENCOP instruction encodes2^3.

In 16bit binary, D8 and D9 are set to 1.

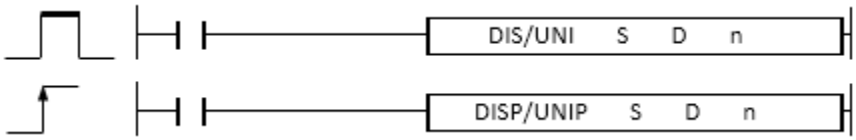




2.10.6. DIS, DISP, UNI, UNIP

• DIS instruction moves 4 bits of source to the lowest bit of each number of Destination.

• UNI instruction moves the lowest 4 bits of each source to the Destination.



S	Source
D	Destination
n	Number of 4bits to move

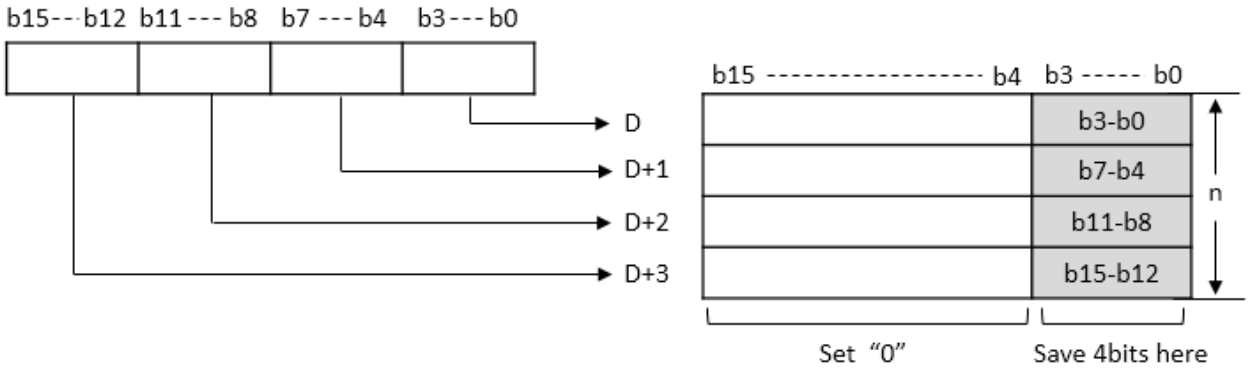
Instruction		Device address													No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
DIS(P) UNI(P)	S	o	o	o	o	o	o	o	o	-	o	o	o	o	4	o	-	-
	D	o	-	o	o	o	-	o	o	-	o	o	o	-				
	n	o	o	o	o	o	o	o	o	-	o	o	o	o				

1) DIS, DISP

- When enabled, the DIS instruction moves each 4 bits of source to the lowest bit of number (n) of the Destination.

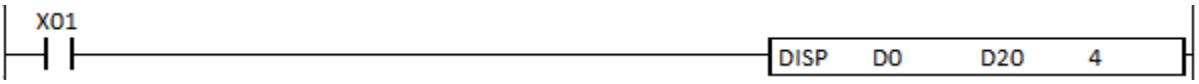
- You can choose 1~4 for (n). The bits from b4 to b15 become 0.

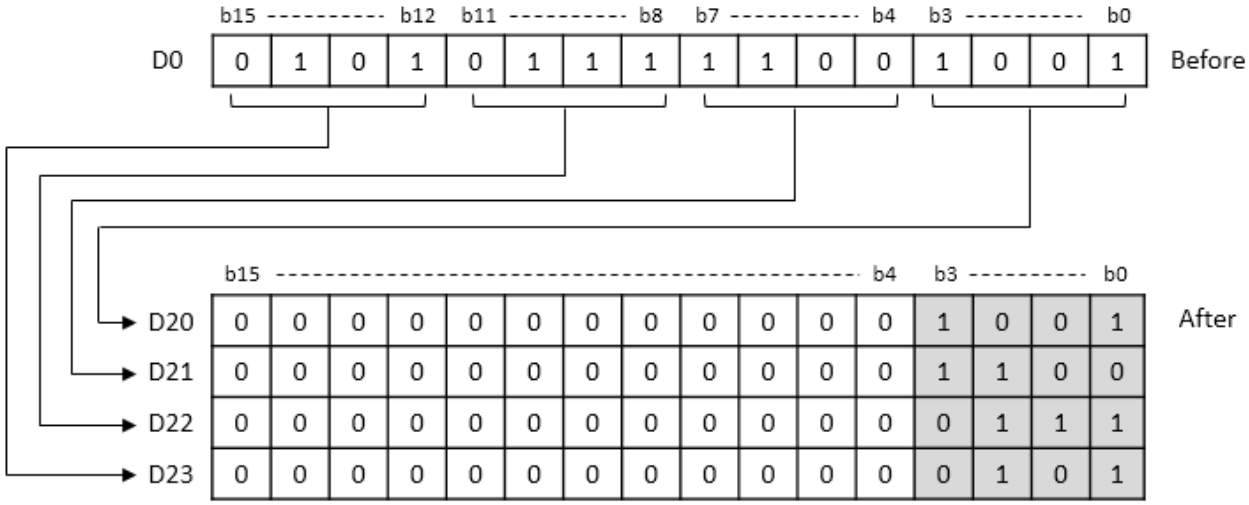
- If n is 0, there will be no change in the Destination.



Example)

• If X01 is ON, DISP instruction moves 4bits of D0 to the lowest bit of D20, D21, D22, and D23 (n=4). The lowest 4bits of D0 is stored from starting device D20.



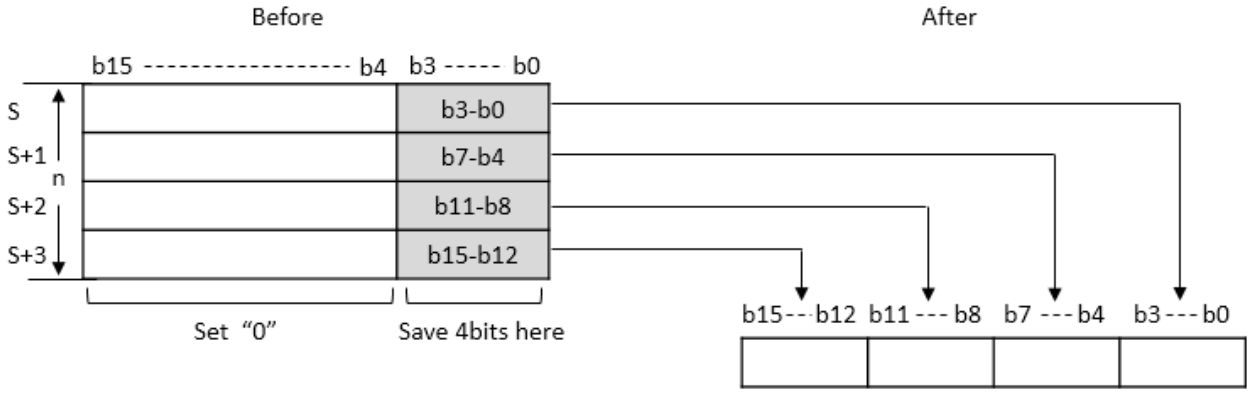


2) UNI, UNIP

- When enabled, the UNI instruction moves the lowest 4bits from (n) number of source to the Destination.

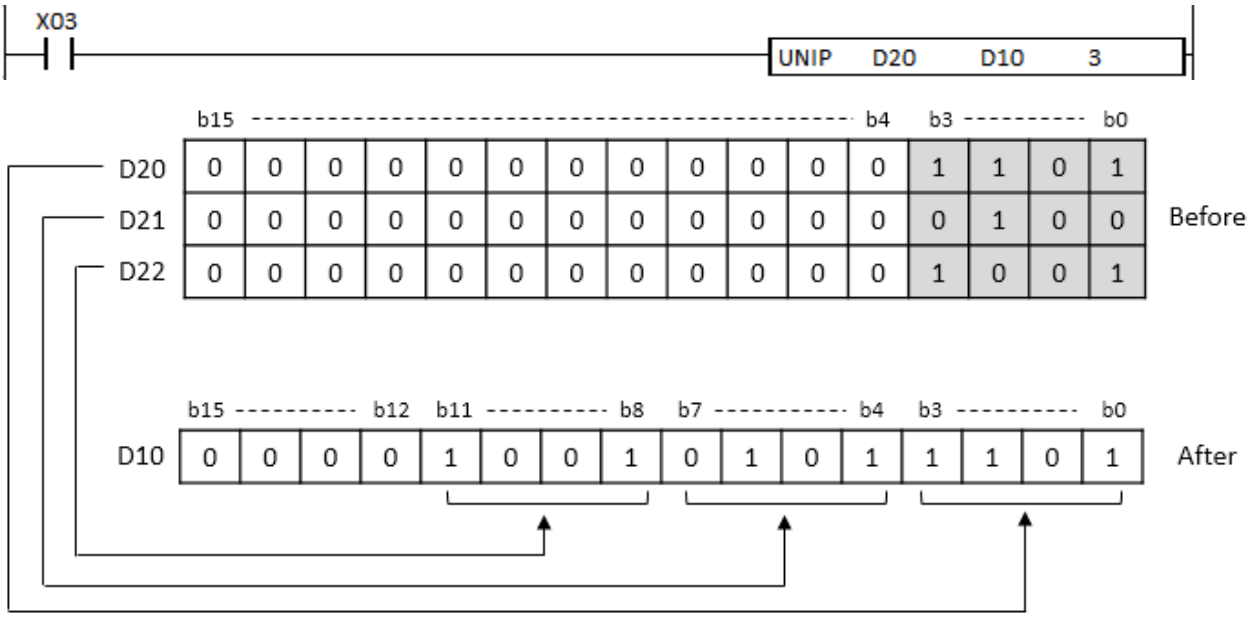
- You can choose 1~4 for (n). The bits from b4 to b15 become 0.

- If n is 0, there will be no change in the Destination.



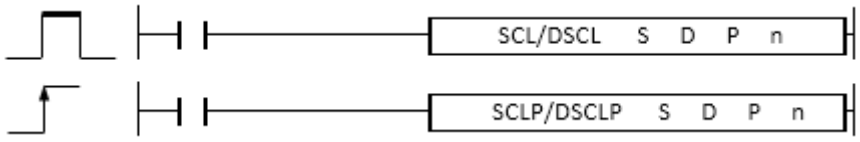
Example)

• If X03 is ON, UNIPS instruction moves each 4 bits of D20, D21, and D22 (n=3) to D10. The starting device D20 is stored from the lowest bit of D10.



2.10.7. SCL, SCLP, DSCL, DSCLP

SCL instruction scales number of source according to condition of T and saves the scaled value to the Destination.



S	Source
D	Destination
P	Scale condition Table
n	Number of data to scale

Instruction		Device address													No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
SCL(P) DSCL(P)	S	o	o	o	o	o	o	o	o	-	o	o	o	o	5	o	-	-
	D	o	-	o	o	o	-	-	-	-	o	o	o	-				
	P	o	-	-	o	o	-	-	-	-	o	o	o	-				
	n	o	o	o	o	o	o	o	o	-	o	o	o	o				

1) SCL

SCL instruction scales number of word source according to condition of P and saves the scaled value to the Destination.

• Scale condition (P)

P	Minimum value of source to be scaled (Start Source)
P + 1	Maximum value of source to be scaled (Last Source)
P + 2	Scaled minimum value of destination (Start Dest.)
P + 3	Scaled maximum value of destination (Last Dest.)

2) DSCL

DSCL instruction scales number of double word source according to condition of P and saves the scaled value to the Destination.

• Scale condition (P)

P, P + 1	Minimum value of source to be scaled (Start Source)
P + 2, P + 3	Maximum value of source to be scaled (Last Source)
P + 4, P + 5	Scaled minimum value of destination (Start Dest.)
P + 6, P + 7	Scaled maximum value of destination (Last Dest.)

Reference

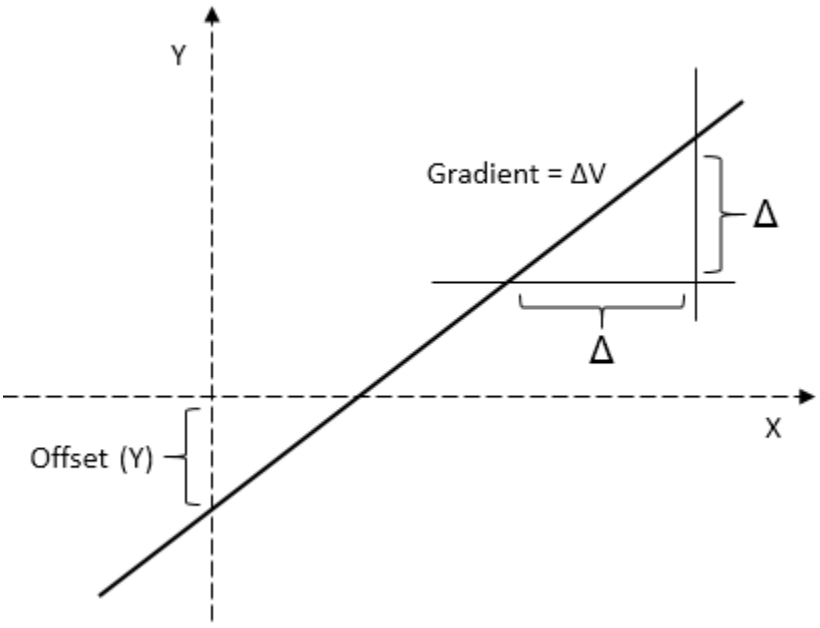
• Scale formula

1. Destination = Source x Gradient(2) + Offset(3)

2. Gradient = $\frac{\text{Last Dest.} - \text{Start Dest.}}{\text{Last source} - \text{Start source}}$

3. Offset (Y intercept) = Last Dest. – Gradient(2) x Last source

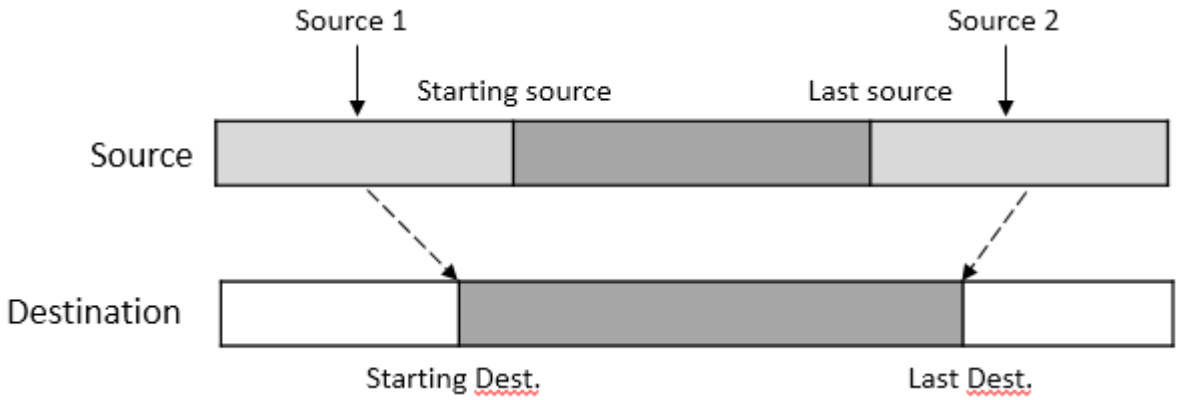
• Scale graph



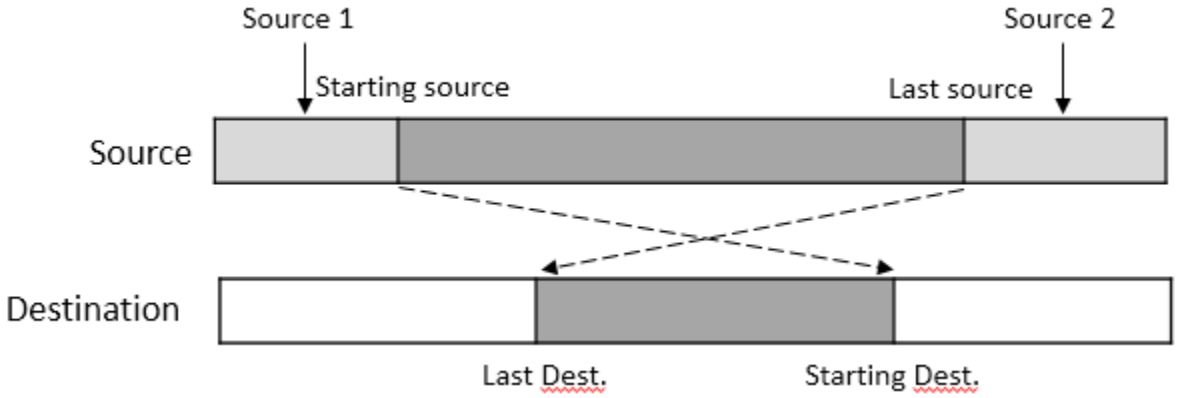
• Scale Range

- If the source data is out of range, its value becomes the maximum and minimum value of Destination.

Source 1 is scaled to Starting value of destination and Source 2 is scaled to Last value of destination.

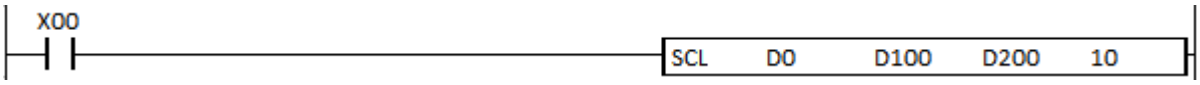


- If the last source is less than starting source and the last destination is less than starting destination, its scaled value is opposite. (In case that Gradient has minus (-) value)



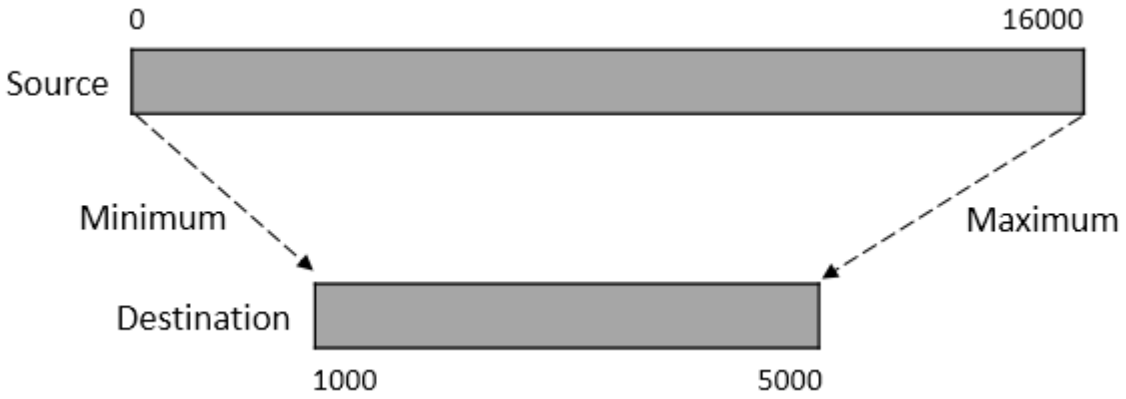
Example)

• If X00 is ON, SCL instruction scales 10 words (D0~D9) according to scale condition (D200 to D203) and saves the scaled value to D100~D109.



- Scale condition

D200	0 : minimum value of source
D201	160000 : maximum value of source
D202	1000 : minimum scaled value
D203	5000 : maximum scaled value

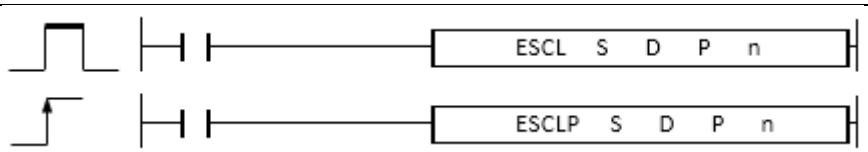


Source	Unscaled value
D0	-192
D1	0
D2	500
D3	4000
D4	8000
D5	10000
D6	12000
D7	13500
D8	16000
D9	16192

Destination	Scaled value
D100	1000
D101	1000
D102	1125
D103	2000
D104	3000
D105	3500
D106	4000
D107	4375
D108	5000
D109	5000

2.10.8. ESCL, ESCLP

ESCL instruction scales number of float source according to condition of P and saves the scaled value to the Destination.



S	Source
D	Destination
P	Scale condition Table
n	Number of data to scale

Instruction		Device address													No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
ESCL ESCLP	S	o	o	o	o	o	o	o	o	-	o	o	o	o	5	o	-	-
	D	o	-	o	o	o	-	-	-	-	o	o	o	-				
	P	o	-	-	o	o	-	-	-	-	o	o	o	-				
	n	o	o	o	o	o	o	o	o	-	o	o	o	o				

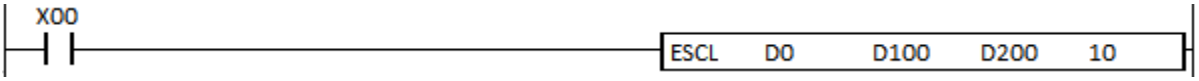
ESCL instruction scales number of float source according to condition of P and saves the scaled value to the Destination.

• Scale condition (P)

P	Minimum value of source to be scaled (Start Source)
P + 1	Maximum value of source to be scaled (Last Source)
P + 2	Scaled minimum value of destination (Start Dest.)
P + 3	Scaled maximum value of destination (Last Dest.)

Example)

• If X00 is ON, ESCL instruction scales 10 float value (D0~D9) according to scale condition (D200 to D203) and saves the scaled value to D100~D109.



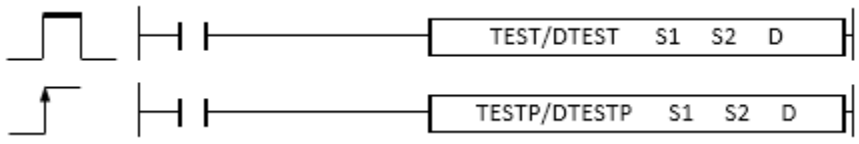
- Scale condition

D200	Minimum value of source to be scaled (Start Source)
D201	Maximum value of source to be scaled (Last Source)
D202	Scaled minimum value of destination (Start Dest.)
D203	Scaled maximum value of destination (Last Dest.)

2.11. Bit Processing Instruction

2.11.1. TEST, TESTP, DTEST, DTESTP

TEST instruction reads the specific bits of word source and writes its value to the Destination to set or clear the data bit.



S1	Source
S2	Bit number
D	Destination to be set or cleared

Instruction		Device address													No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
TEST(P) DTEST(P)	S1	o	o	o	o	o	o	o	o	-	o	o	o	-	4	o	-	-
	S2	o	o	o	o	o	o	o	o	-	o	o	o	o				
	D	o	-	o	o	o	-	o	o	-	-	-	-	-				

1) TEST

TEST instruction reads the specific bits (S1)of word source(S1) and writes its value to the Destination(D) to set or clear the data bit.

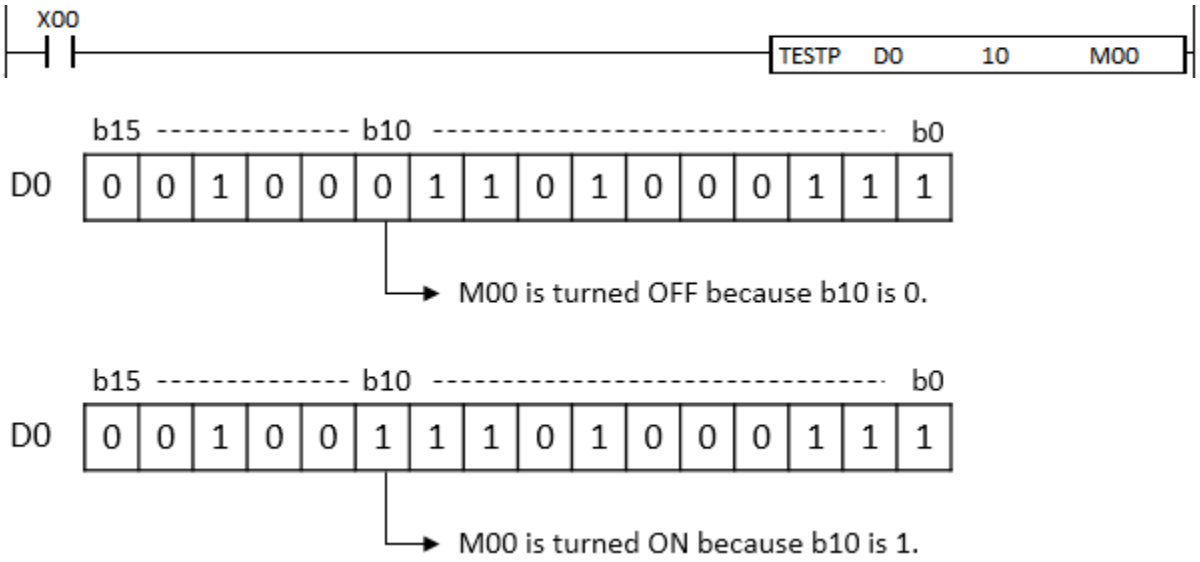
- The range of S2 is from 0 to 15 (bit0~bit15)

- If D is 1, its bit will be turned ON and if D is 0, its bit will be turned OFF.

Example)

• If X00 is ON, TEST instruction reads bit10 of D0.

According to the value of the bit10, M00 will be set (1) or cleared (0).



2) DTEST

DTEST instruction reads the specific bits (S1) of double word source (S1) and writes its value to the Destination(D) to set or clear the data bit.

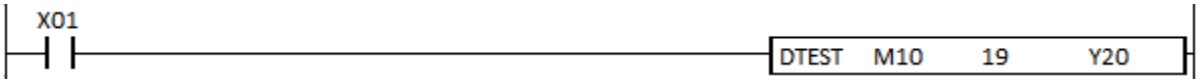
- The range of S2 is from 0 to 31 (bit0~bit31)

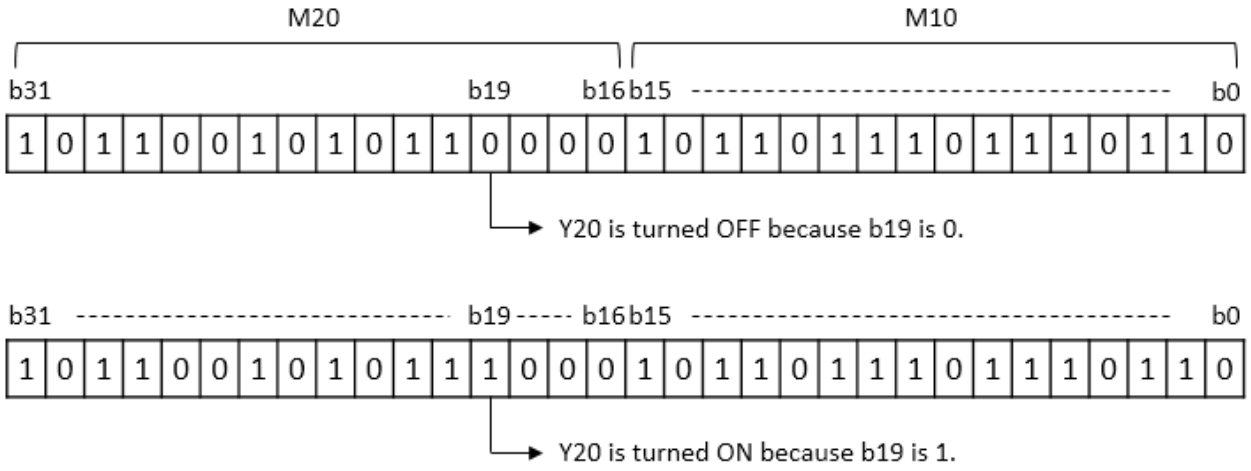
- If D is 1, its bit will be turned ON and if D is 0, its bit will be turned OFF.

Example)

• If X01 is ON, DTEST instruction reads bit19 of M10.

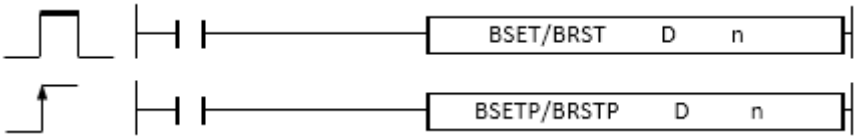
According to the value of the bit19, M0 will be set (1) or cleared (0).





2.11.2. BSET, BSETP, BRST, BRSTP

BSET instruction sets the specific bit of Destination.



D	Destination
n	Bit number of Destination

Instruction		Device address												No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
BSET(P) BRST(P)	D	o	-	o	o	o	-	o	o	-	o	o	o	-	3	o	-	-
	n	o	o	o	o	o	o	o	o	-	o	o	o	o				

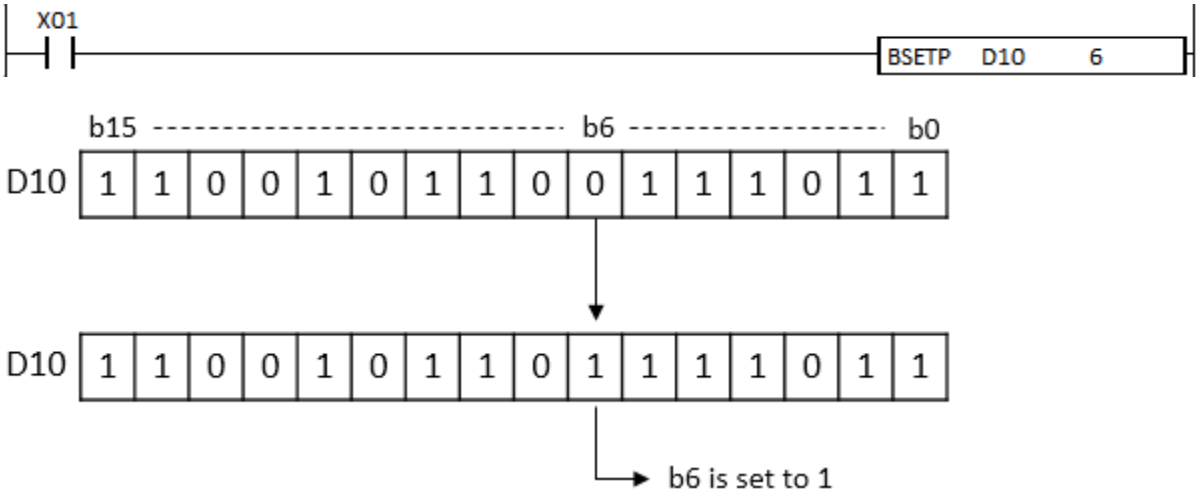
1) BSET

BSET instruction sets (1) the specific bit (n) of Destination.

- The range of n is from 0 to 15 (bit0~bit15)

Example)

• If X01 is ON, BSET instruction sets b6 of D10 to 1.



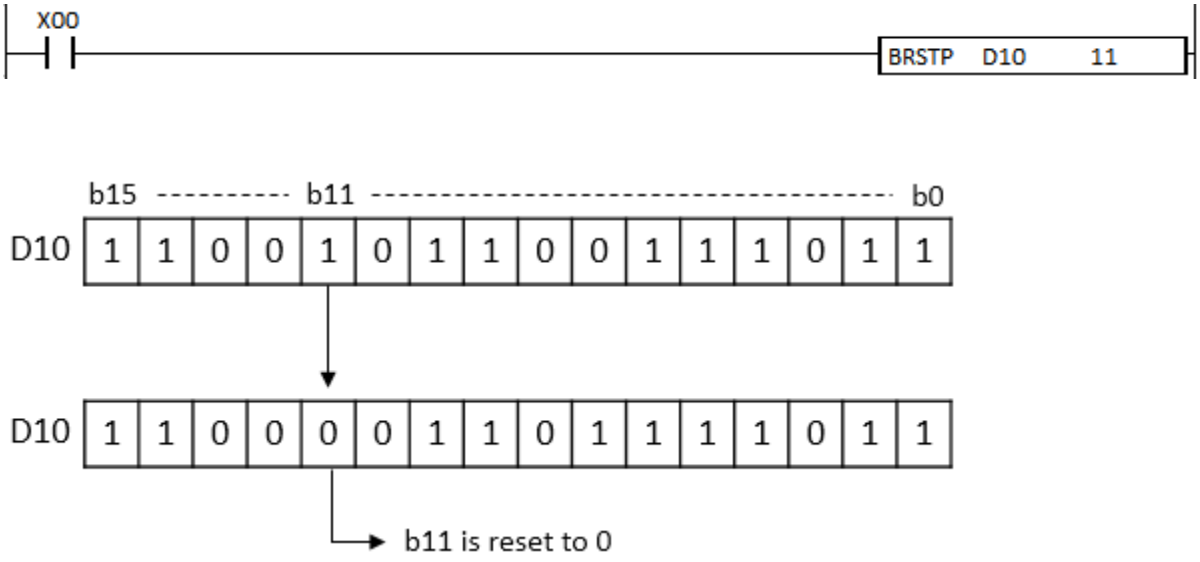
2) BRST

BSET instruction resets (0) the specific bit (n) of Destination.

- The range of n is from 0 to 15 (bit0~bit15)

Example)

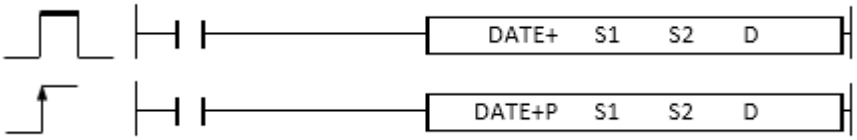
• If X00 is ON, BRST instruction resets b11 of D10 to 0.



2.12. Clock Processing Instruction

2.12.1. DATE+, DATE+P

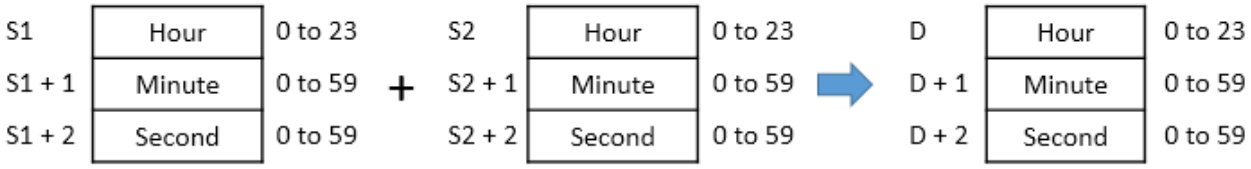
DATE+ instruction adds two clock data (S1 and S2) and saves the result to the Destination.



S1	Starting address of clock data
S2	Starting address of clock data
D	Destination

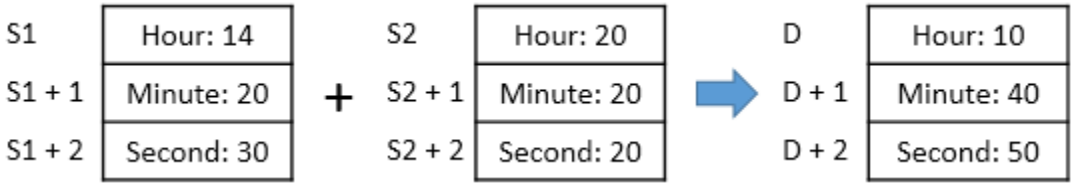
Instruction		Device address													No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
DATE+ DATE+P	S1	o	o	o	o	o	o	o	o	-	o	o	o	-	4	o	-	-
	S2	o	o	o	o	o	o	o	o	-	o	o	o	-				
	D	o	-	o	o	o	-	o	o	-	o	o	o	-				

- When enabled, DATE+ instruction adds clock date S1 and S2 and saves the result to the Destination.



- If the result is more than 24 hours, subtract 24 from the result.

For example, in case of 20:20:20 + 14:40:50, the result is 10:40:50. (Not 34:40:50)

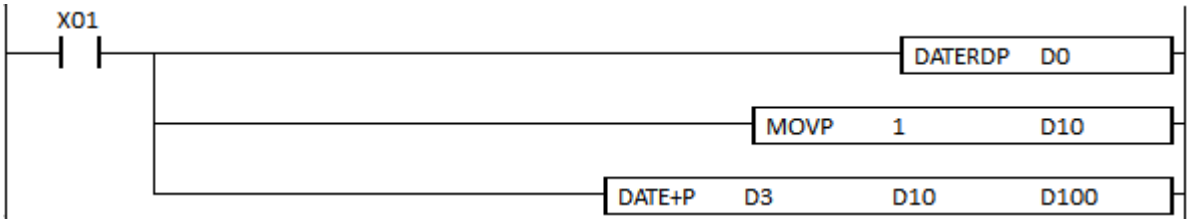


- If S1 and S2 are out or range, Error flag will be turned ON.

Example)

• If X01 is ON, DATERDP instruction writes a date value to D0 ~ D6 and MOVP instruction writes 1 to D10.

Then, DATE+P instruction adds time data (D3~D5) and (D10~D12) and saves the result to the D100~D102.



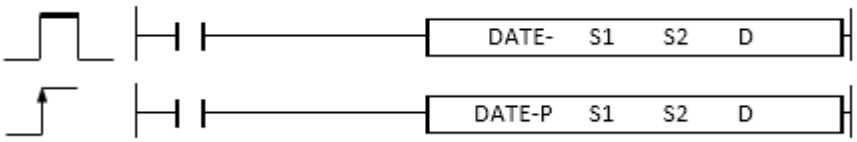
D0	2015	Year
D1	8	Month
D2	25	Day
D3	10	Hour
D4	23	Minute
D5	41	Second
D6	2	Day of week

Time data

D3	Hour: 10		D10	Hour: 1		D100	Hour: 11
D4	Minute: 23	+	D11	Minute: 0	⇒	D101	Minute: 23
D5	Second: 41		D12	Second: 0		D102	Second: 41

2.12.2. DATE-, DATE-P

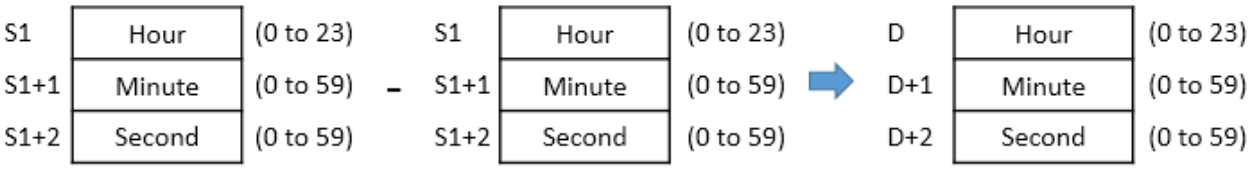
DATE- instruction subtracts clock data S2 from S1 and saves the result to the Destination.



S1	Starting address of clock data
S2	Starting address of clock data
D	Destination

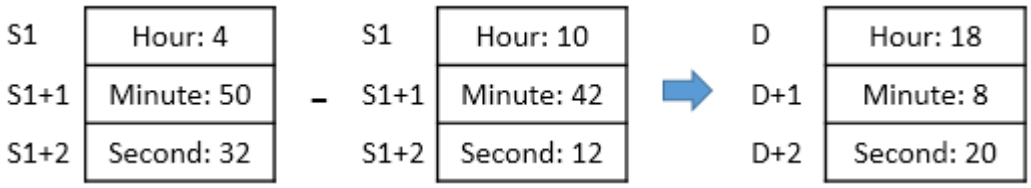
Instruction		Device address												No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
DATE- DATE-P	S1	o	o	o	o	o	o	o	o	-	o	o	o	-	4	o	-	-
	S2	o	o	o	o	o	o	o	o	-	o	o	o	-				
	D	o	-	o	o	o	-	o	o	-	o	o	o	-				

- When enabled, DATE- instruction subtracts clock date S2 from S1 and saves the result to the Destination.



- If the result has (-)minus value, add 24 to the result.

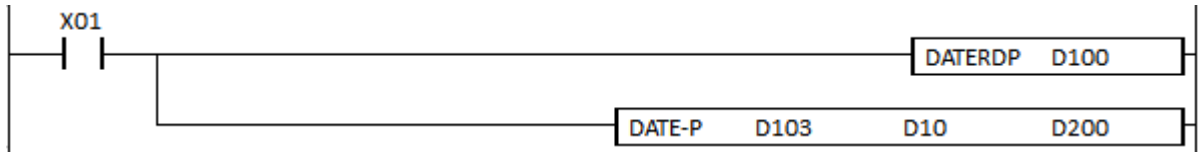
For example, in case of 4:50:32 - 10:42:12, the result is 18:8:20. (Not -6:8:20)



- If S1 and S2 are out or range, Error flag will be turned ON.

Example)

• If X01 is ON, the DATERDP instruction writes a date value to D100 ~ D106 and then DATE-P instruction subtracts time data (D10~D12) from (D103~D105) and saves the result to the D200~D202.



D100	2015	Year
D101	8	Month
D102	25	Day
D103	10	Hour
D104	40	Minute
D105	20	Second
D106	1	Day of week

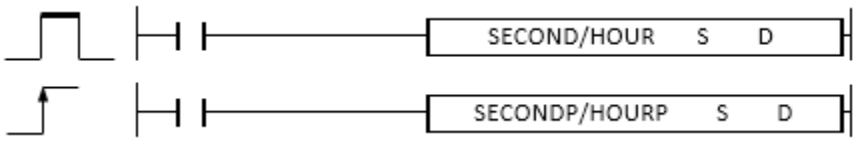
Time data

D103	Hour: 10	-	D10	Hour: 3	→	D200	Hour: 6
D104	Minute: 40		D11	Minute: 50		D201	Minute: 50
D105	Second: 20		D12	Second: 10		D202	Second: 10

2.12.3. SECOND, SECONDP, HOUR, HOURP

SECOND instruction converts time data(S) to second value and saves the result to the Destination (D).

HOUR instruction converts second value(S) to time data (Hour, Minute and Second value) and saves the result to the Destination (D).

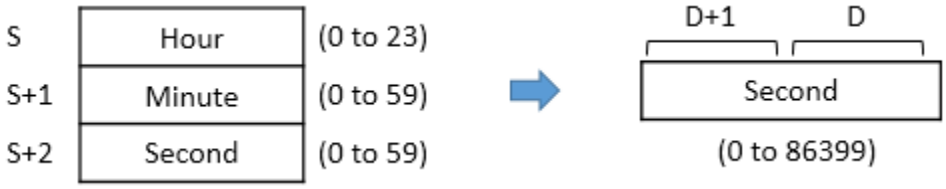


S	Time data
D	Destination

Instruction		Device address													No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
SECOND SECONDP HOUR HOURP	S	o	o	o	o	o	o	o	o	-	o	o	o	-	3	o	-	-
	D	o	-	o	o	o	-	o	o	-	o	o	o	-				

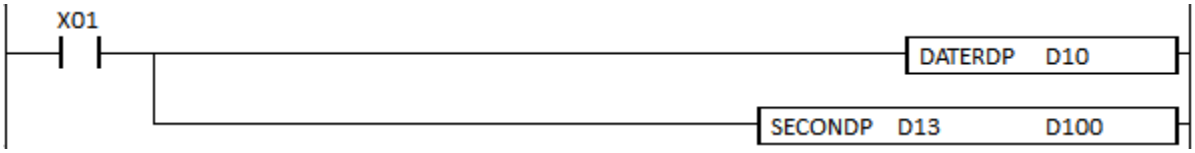
1) SECOND

- When enabled, SECOND instruction converts time data(S) to second value and saves the result to the Destination (D).



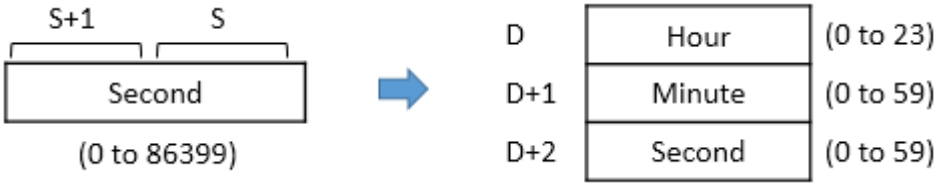
Example)

• If X01 is ON, DATERDP instruction write a date value to D10 ~ D16 and then SECOND instruction converts hour, minute and second (D13 to D15) to second value (73283) and then save the second value to D100 and D101.



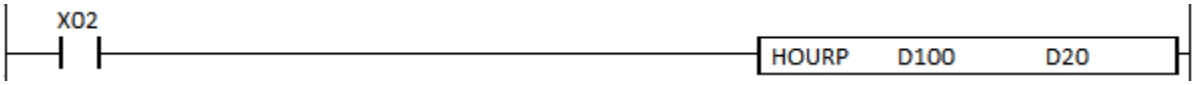
2) HOUR

- When enabled, HOUR instruction converts second value(S) to time data (Hour, Minute and Second) and saves the result to the Destination (D).



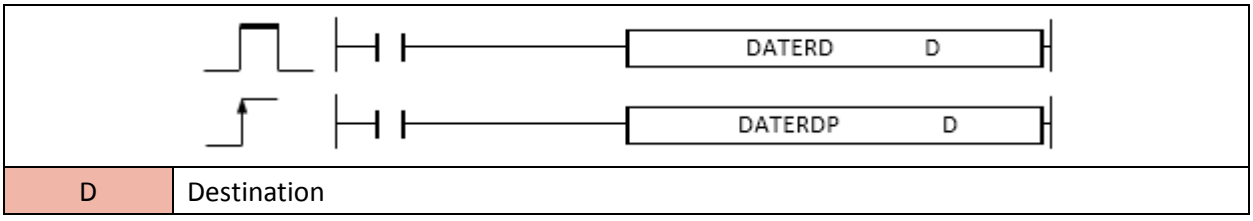
Example)

• If X02 is ON, HOURP instruction converts second value (40000) to time data (11:6:40) and then saves the result to D20(11), D21(6), and D22(40).



2.12.4. DATERD, DATERDP

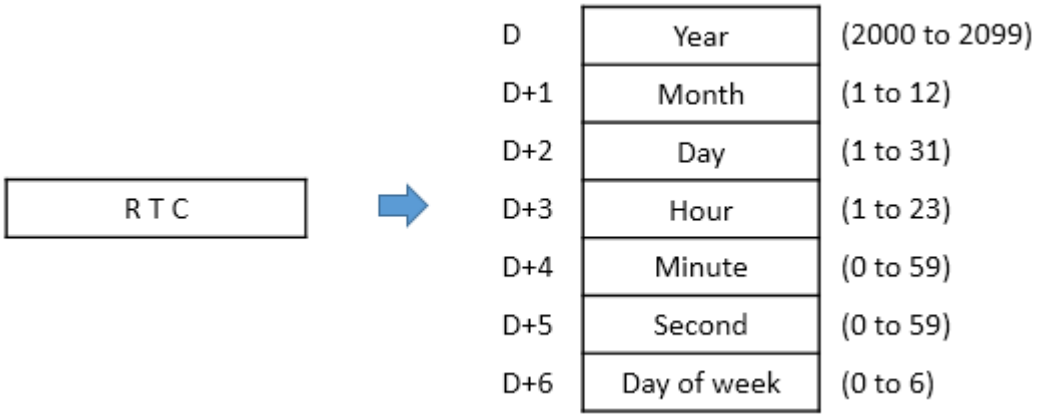
DATERD instruction reads RTC and saves the result to the Destination.



D	Destination

Instruction		Device address													No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
DATERD DATERDP	D	o	-	o	o	o	-	o	o	-	o	o	o	-	2	o	-	-

- When enabled, DATERD instruction reads RTC and saves the result to the Destination.



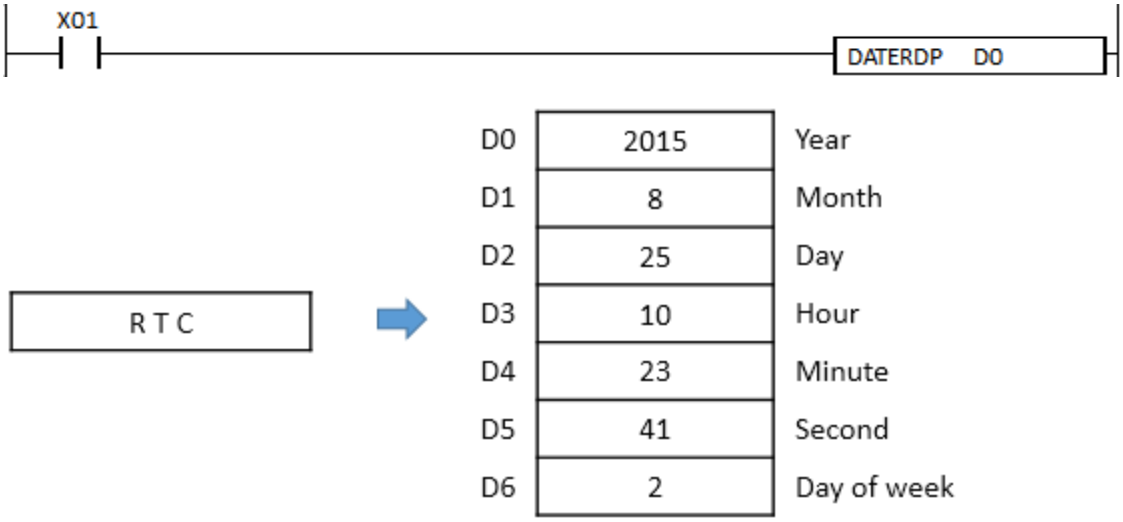
- Year has 4 digits.

- Day of week (D+6) shows Sunday to Saturday as below.

Day of week	Sunday	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday
Saved value	0	1	2	3	4	5	6

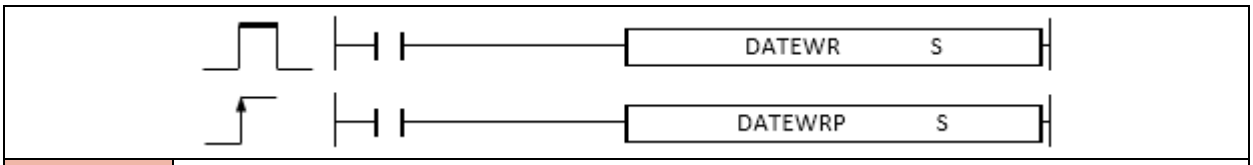
Example)

• If X01 is ON, DATERDP instruction reads RTC and writes a date value to D0 ~ D6.



2.12.5. DATEWR, DATEWRP

DATEWR instruction writes time data (Year, Month, Day, Hour, Minute, Second and Day of week) to the RTC.

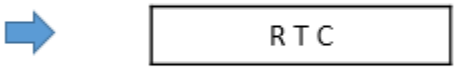


S	Time data

Instruction		Device address												No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
DATEWR DATEWRP	S	o	o	o	o	o	o	o	o	-	o	o	o	-	2	o	-	-

- When enabled, the DATEWR instruction writes time data to the RTC.

S	Year	(2000 to 2099)
S+1	Month	(1 to 12)
S+2	Day	(1 to 31)
S+3	Hour	(1 to 23)
S+4	Minute	(0 to 59)
S+5	Second	(0 to 59)
S+6	Day of week	(0 to 6)



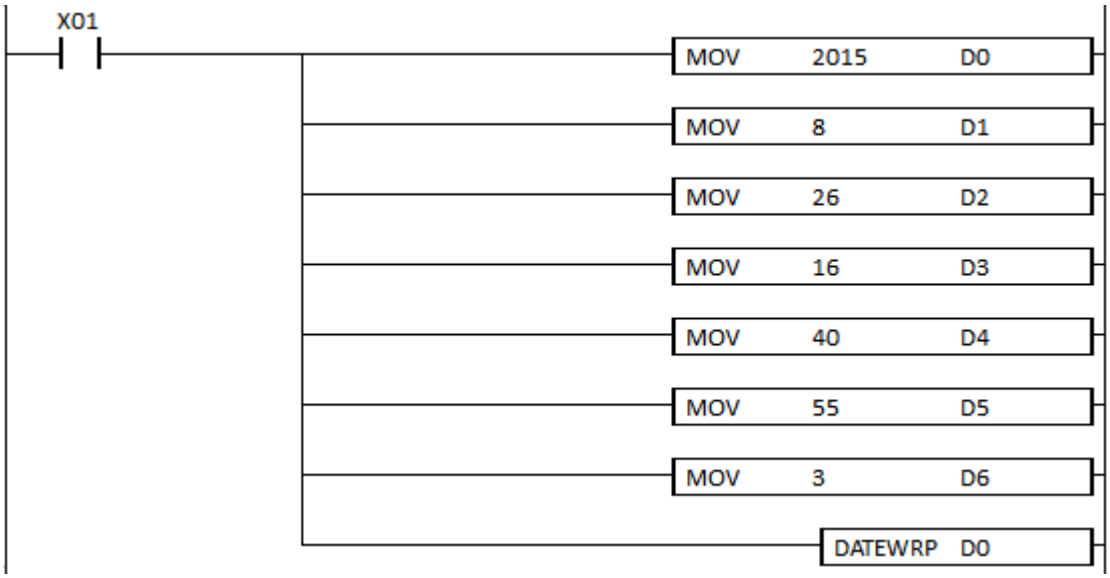
- Year has 4 digits.

- Day of week (S+6) shows Sunday to Saturday as below.

Day of week	Sunday	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday
Saved value	0	1	2	3	4	5	6

Example)

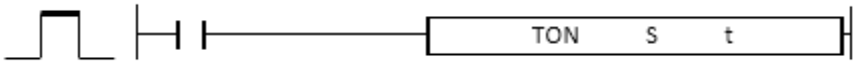
• If X01 is ON, MOV instruction moves data to each device and then the DATEWRP instruction stores D0~D6 value to the RTC.



2.13. Timer and Counter

2.13.1. TON (Timer On Delay)

TON instruction accumulates time (ACC value) to set timer(S).

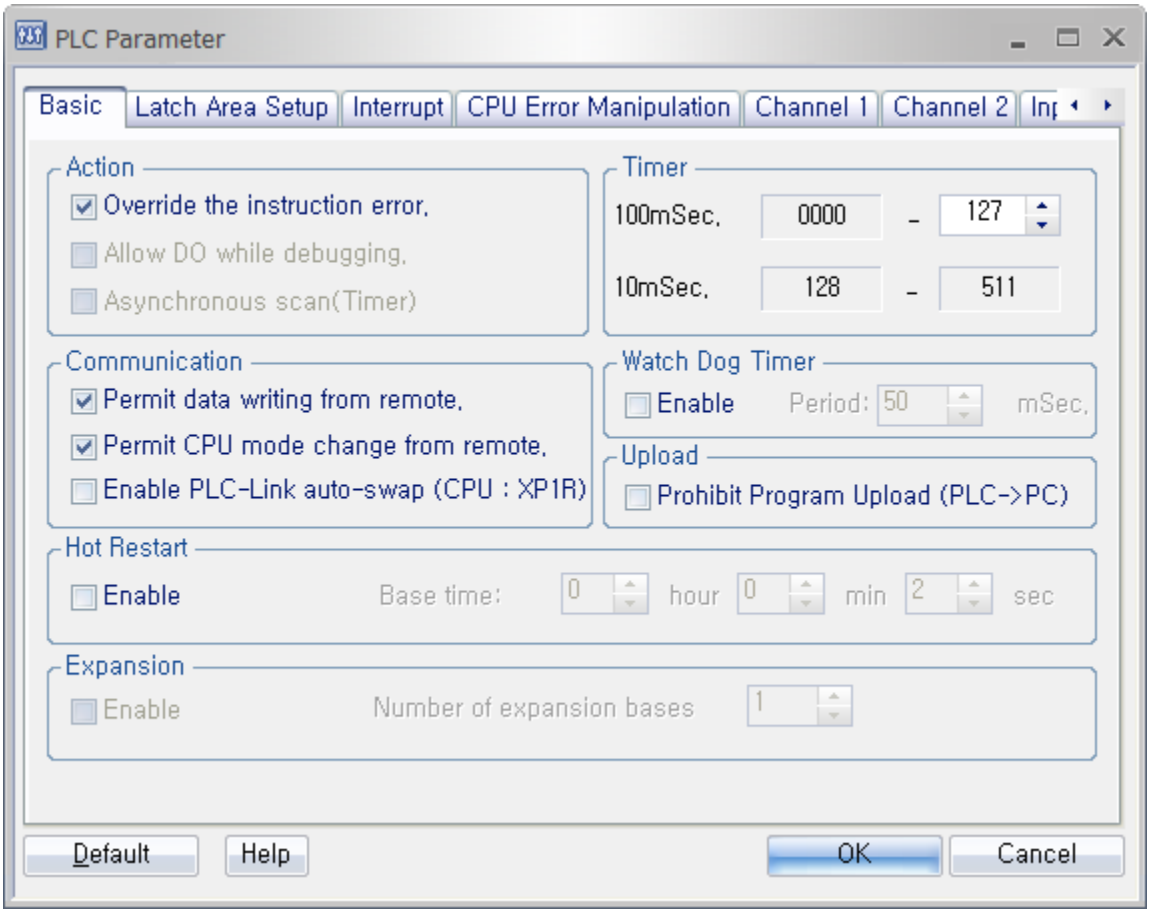


S	Timer device address to set
t	Preset value

Instruction		Device address													No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
TON	S	-	-	-	-	-	-	o	-	-	-	-	-	-	3	-	-	-
	t	-	-	-	-	-	-	-	-	-	-	o	-	o				

When enabled, the TON instruction counts preset value (t) and when the accumulated value (ACC value) reaches preset value (t), Timer device(S) will be set.

- You can set up the time base (t) in PLC Parameter.



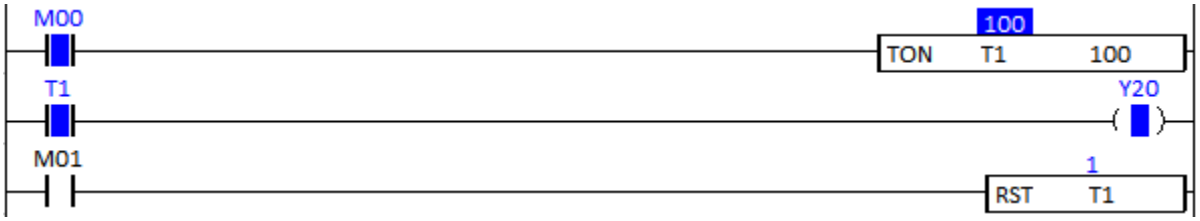
According to Timer parameter, T0~T127 is 100ms and T128~T511 is 10ms.

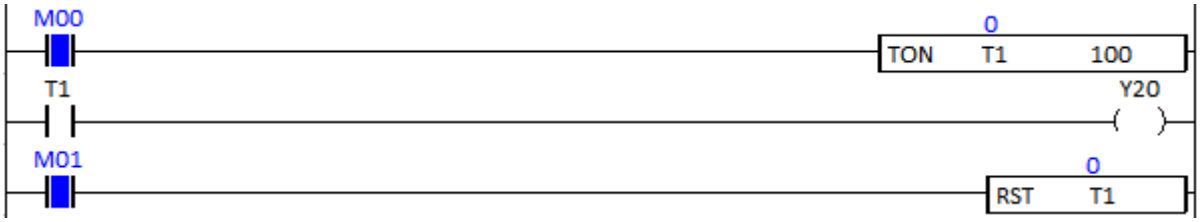
(EX. TON T5 100: 10,000ms = 10sec.)

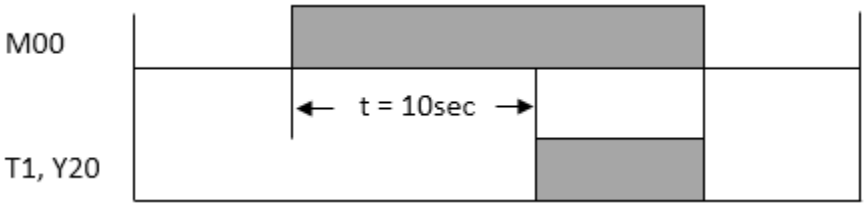
- The TON is a non-retentive timer. When the TON instruction is disabled, ACC value is cleared.
- The range of preset (t): 0 ~ 65535

Example)

• If M00 is ON, the TON instruction counts up to 10sec. and when it reaches 10sec, T1 and Y20 are turned ON. If M01 is ON (TON instruction is disabled), T1 (ACC value) is cleared.

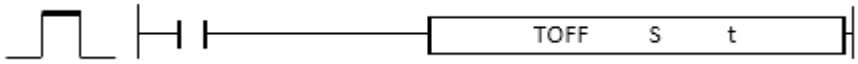






2.13.2. TOFF (Timer Off Delay)

TOFF instruction counts downward time (Preset value) to set timer(S).

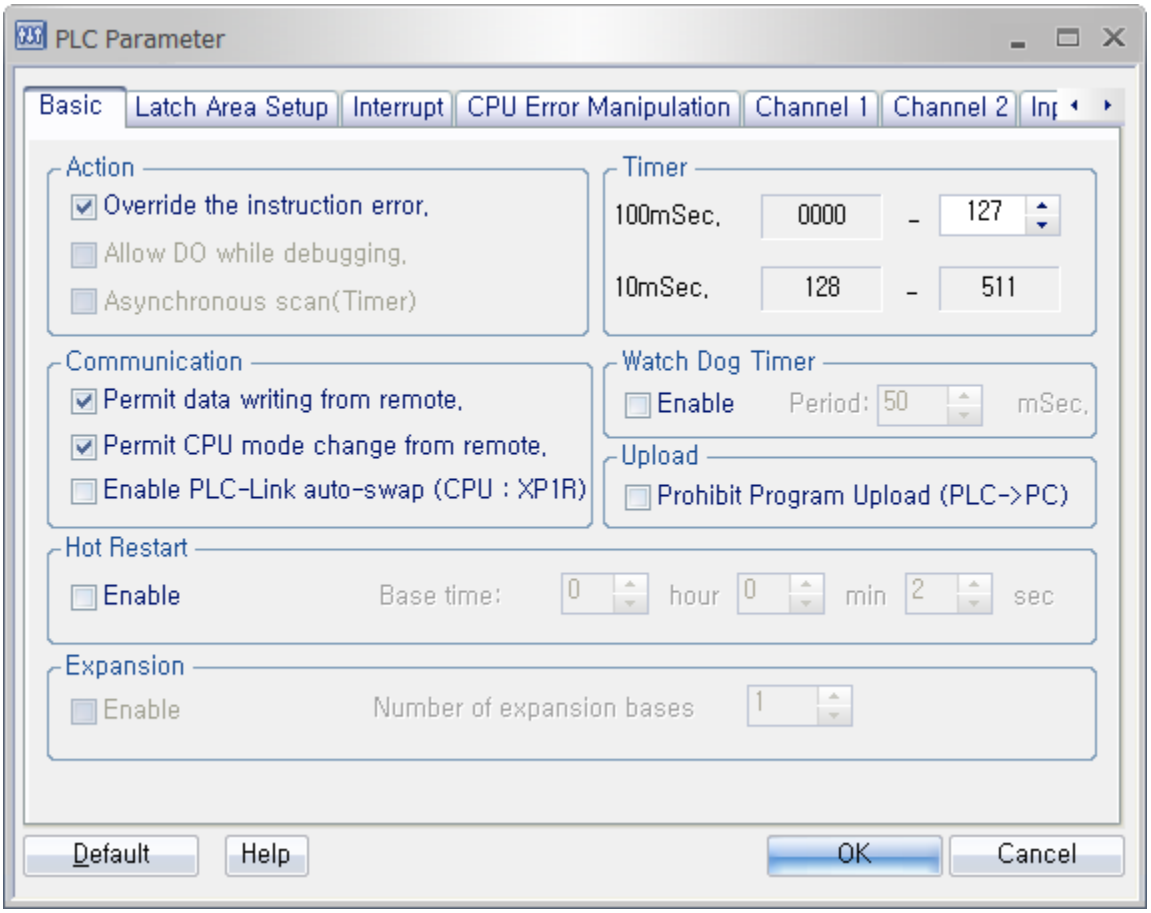


S	Timer device address to set
t	Preset value

Instruction		Device address												No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
TOFF	S	-	-	-	-	-	-	o	-	-	-	-	-	-	3	-	-	-
	t	-	-	-	-	-	-	-	-	-	-	o	-	o				

When disabled, TOFF instruction decrements preset value (t) and when the accumulated value (ACC value) reaches 0sec, timer device(S) will be turned OFF.

You can set up the time base (t) in PLC Parameter.



According to Timer parameter, T0~T127 is 100ms and T128~T511 is 10ms.

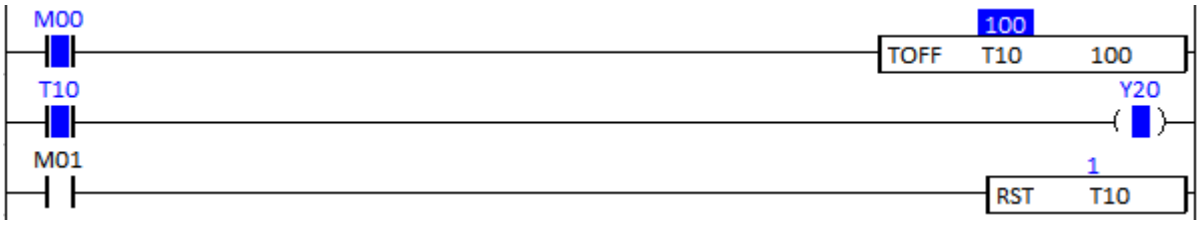
(EX. TOFF T5 100: 10,000ms = 10sec.)

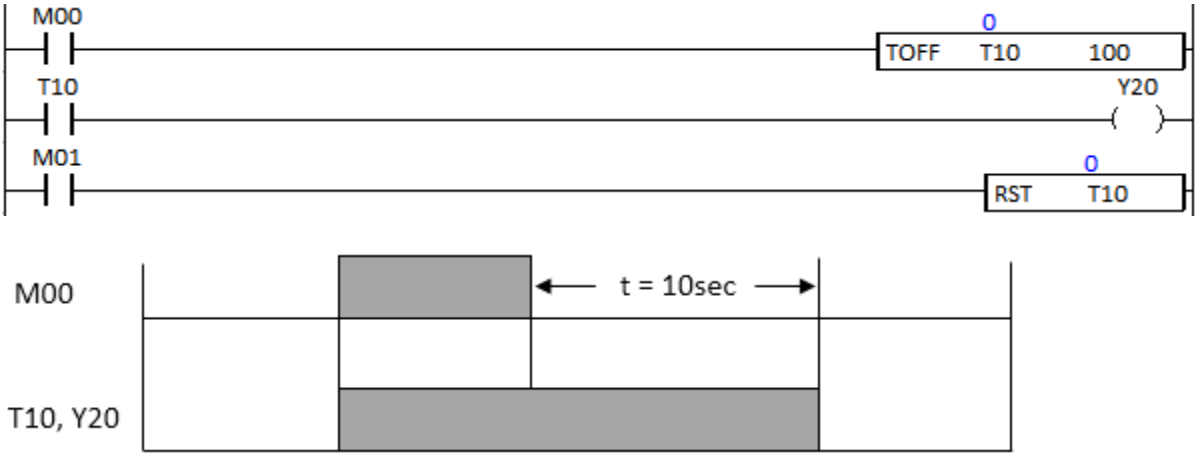
- The TOFF is a non-retentive timer. When the TOFF instruction is disabled, ACC value go back to preset value (t).

- The range of preset (t): 0 ~ 65535

Example)

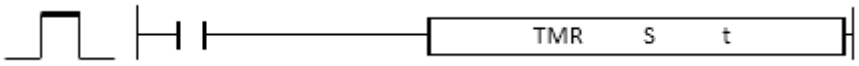
• If M00 is ON, T10 and Y20 are turned ON. When M00 is OFF, TOFF decrements the preset value (from 10 to 0sec) and when it reaches 0sec, T10 and Y20 are turned OFF.





2.13.3. TMR

TMR instruction accumulates time (set value) to set timer(S).

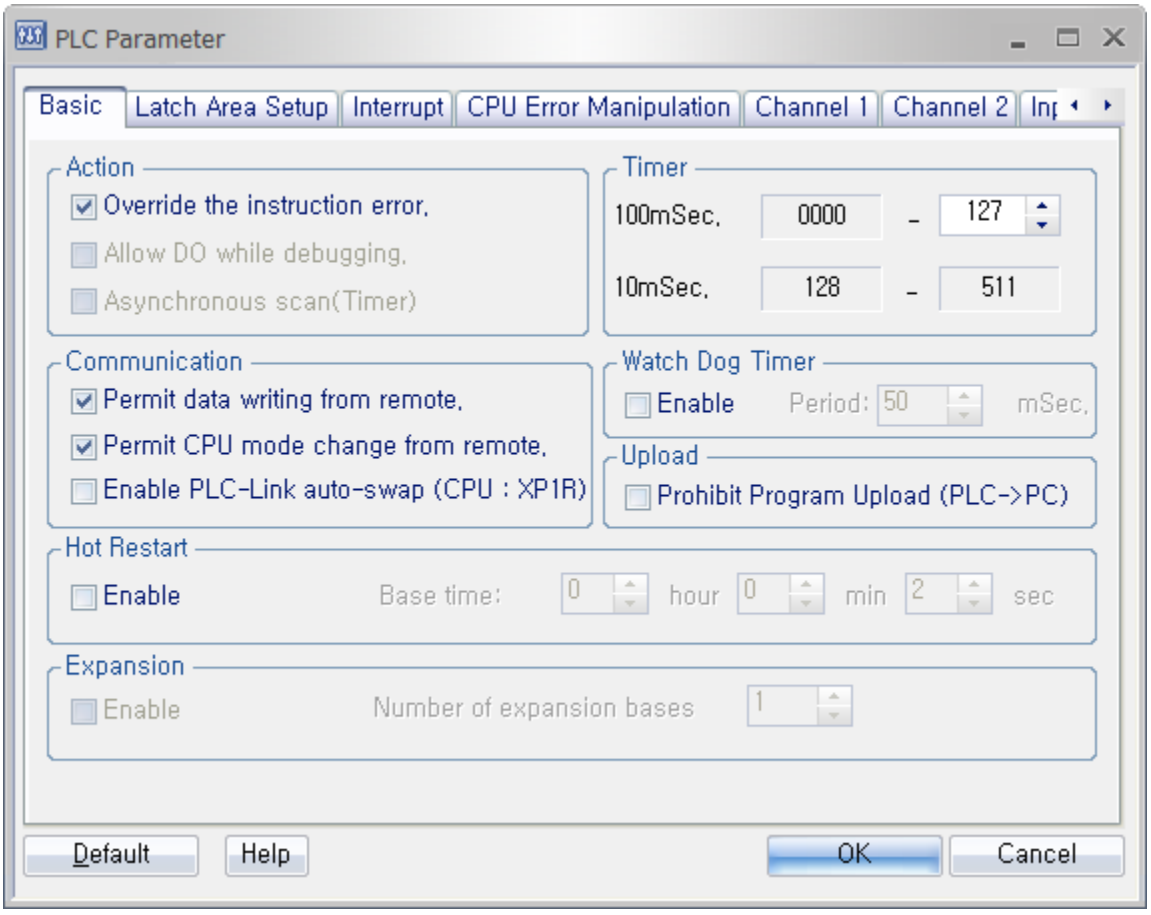


S	Timer device address to set
t	Accumulated (ACC) preset value

Instruction		Device address												No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
TMR	S	-	-	-	-	-	-	o	-	-	-	-	-	-	3	-	-	-
	t	-	-	-	-	-	-	-	-	-	-	o	-	o				

When enabled, TMR instruction counts preset value (t) and when the accumulated value (ACC value) reaches preset value (t), Timer device(S) will be turned ON

You can set up the time base (t) in PLC Parameter.



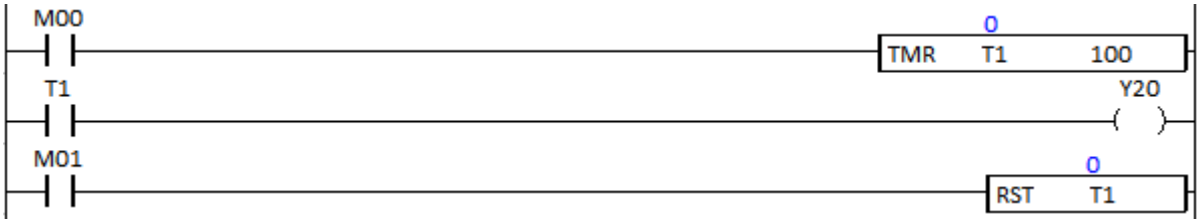
According to Timer parameter, T0~T127 is 100ms and T128~T511 is 10ms.

(EX. TMR T5 100 : 10,000ms = 10sec.)

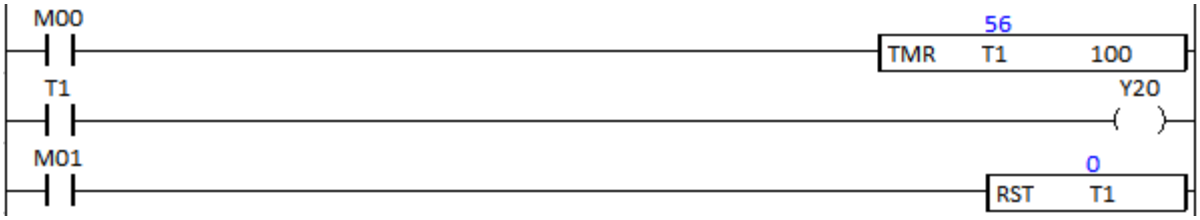
- The TMR is a retentive timer. When the TMR instruction is disabled, it retains its ACC value.
- The range of preset (t) : 0 ~ 65535

Example)

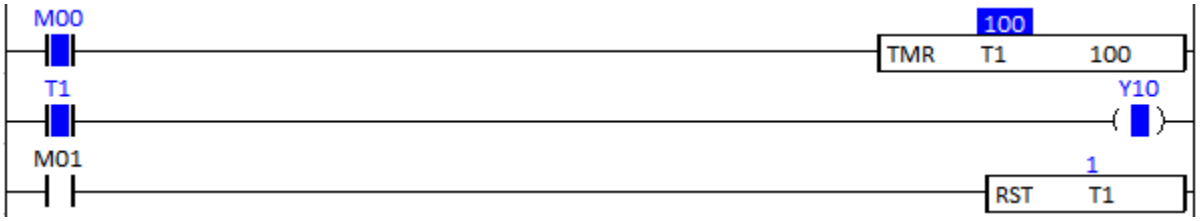
• If M00 is ON, the TMR instruction counts up to 10sec. Even if M00 is disabled, TMR instruction retains its ACC value. When it reaches 10sec, T1 and Y10 are turned ON. If M01 is ON, T1 (ACC value) is cleared.

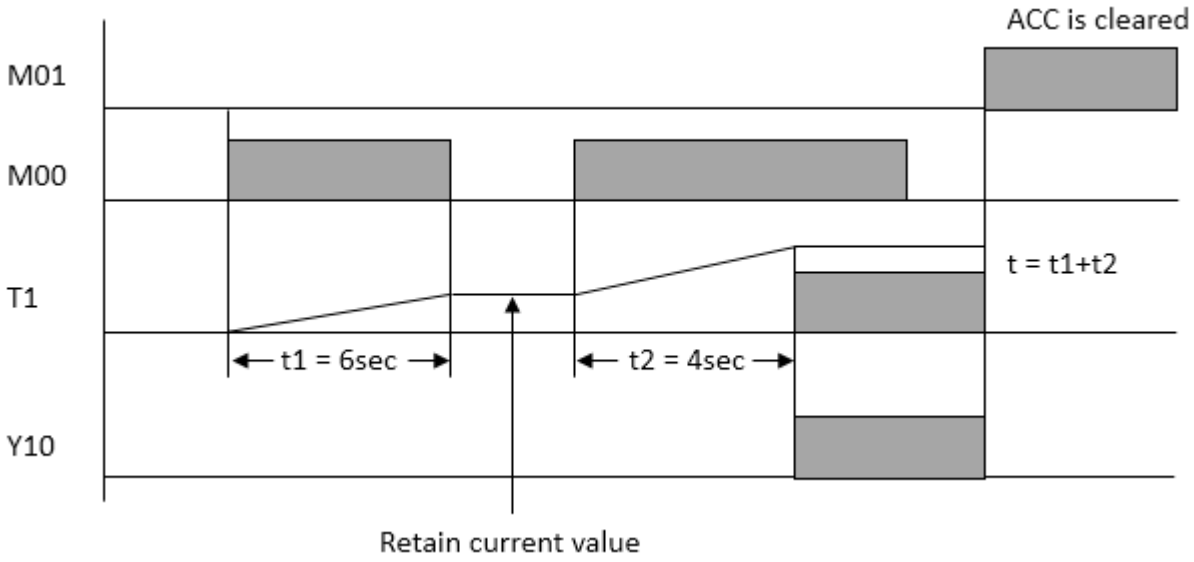


• TMR retains ACC value (56) even if M00 is disabled.



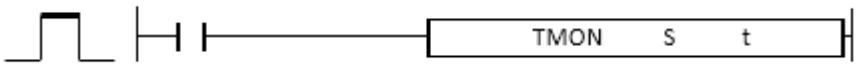
• When it reaches 10sec, T1 and Y10 are turned ON.





2.13.4. TMON

TMON instruction counts downward up to preset value.

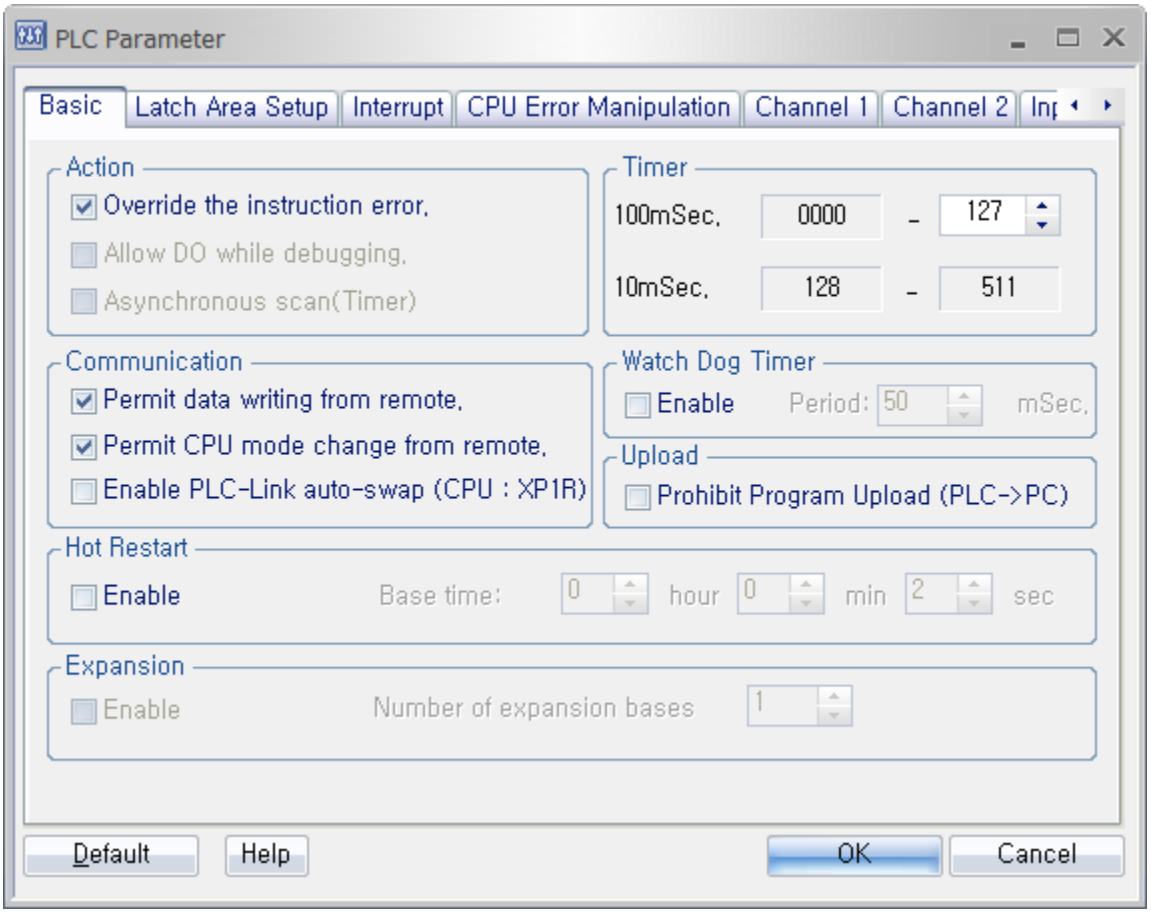


S	Timer device address to set
t	Preset value

Instruction		Device address												No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
TMON	S	-	-	-	-	-	-	o	-	-	-	-	-	-	3	-	-	-
	t	-	-	-	-	-	-	-	-	-	-	o	-	o				

When enabled, TMON instruction counts downward preset value (t) and when the accumulated value (ACC value) reaches preset value (t), Timer device(S) will be turned OFF.

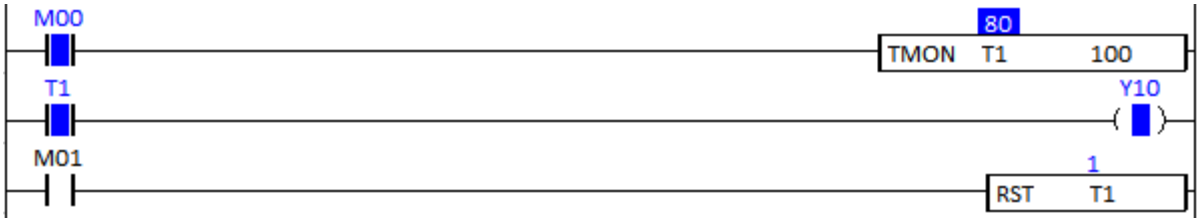
You can set up the time base (t) in PLC Parameter.



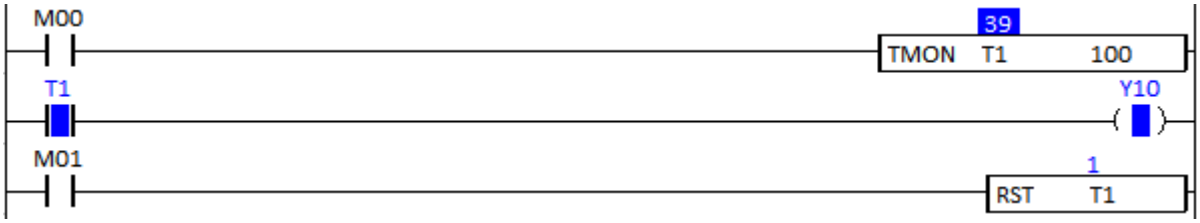
According to Timer parameter, T0~T127 is 100ms and T128~T511 is 10ms.
(EX. TMON T5 100 : 10,000ms = 10sec.)
- The TMON is a retentive timer. Even if the TMON instruction is disabled, it keeps decrementing the counter up to preset value (t)
- The range of preset (t) : 0 ~ 65535

Example)

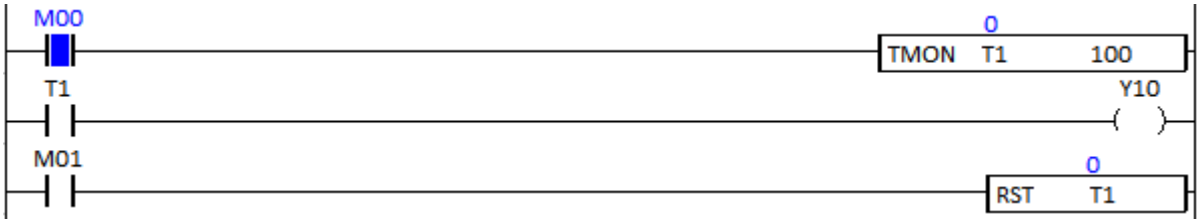
• If M00 is ON, T1 and Y10 are turned ON and TMON instruction decrements the preset value (from 10 to 0sec). When it reaches 0sec, T1 and Y10 are turned OFF. Even if M00 is disabled while ACC value decrements, the TMON instruction keeps decrementing preset value.

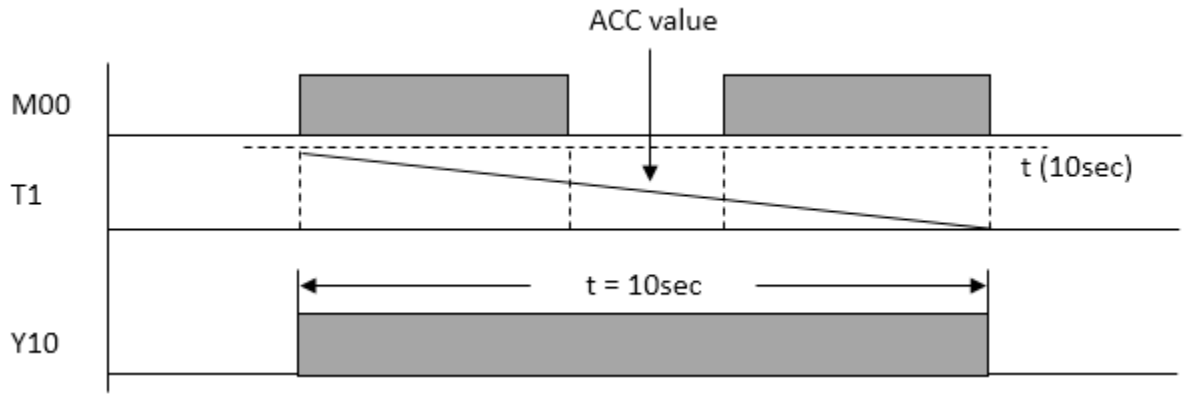


• Even if M00 is OFF, the TMON instruction counts downward up to 0.



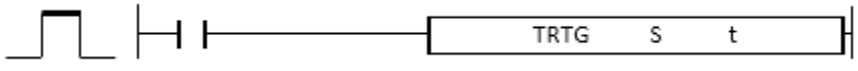
• When it reaches 0sec, T1 and Y10 are turned OFF.





2.13.5. TRTG

TRTG instruction counts downward up to preset value.

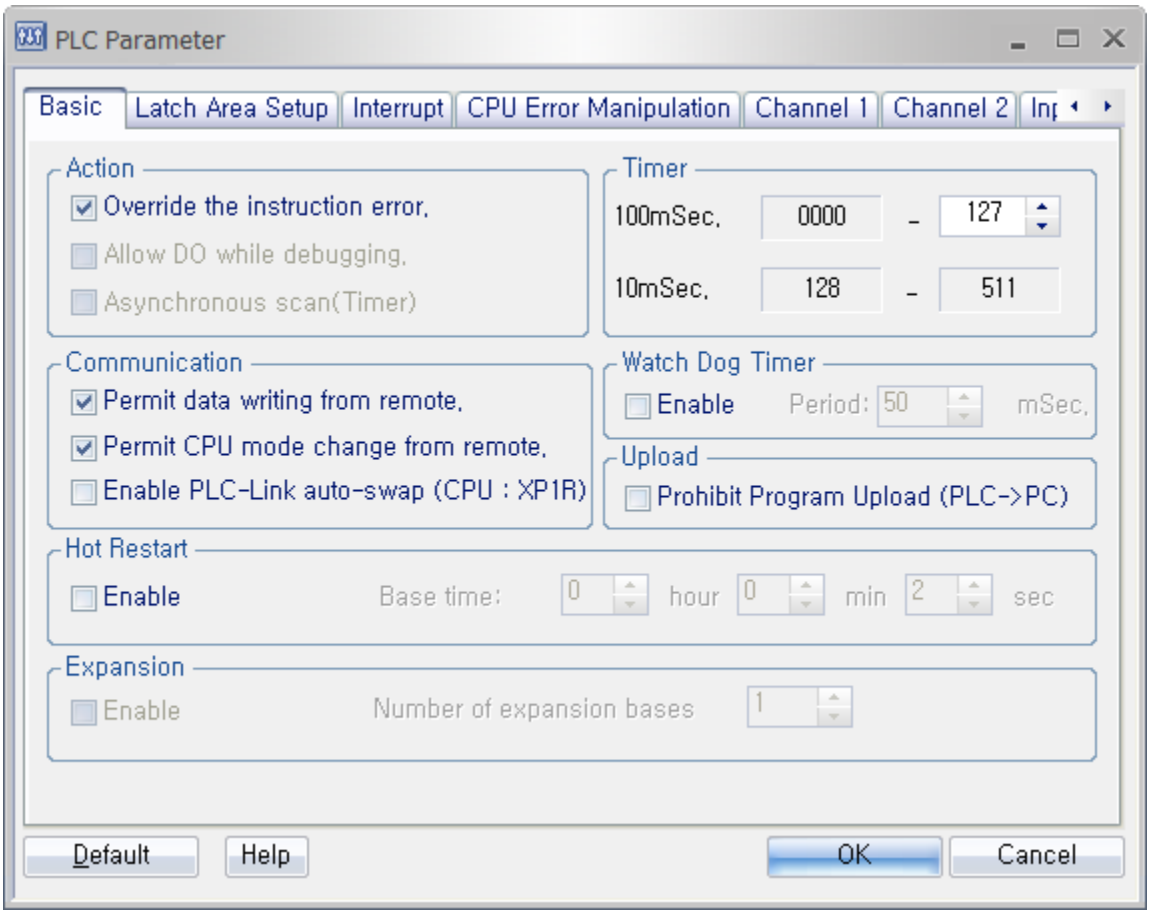


S	Timer device address to set
t	Preset value

Instruction		Device address													No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
TRTG	S	-	-	-	-	-	-	o	-	-	-	-	-	-	3	-	-	-
	t	-	-	-	-	-	-	-	-	-	-	o	-	o				

When enabled, TRTG instruction counts downward preset value (t) and when the accumulated value (ACC value) reaches preset value (t), Timer device(S) will be turned OFF.

- You can set up the time base (t) in PLC Parameter.



According to Timer parameter, T0~T127 is 100ms and T128~T511 is 10ms.

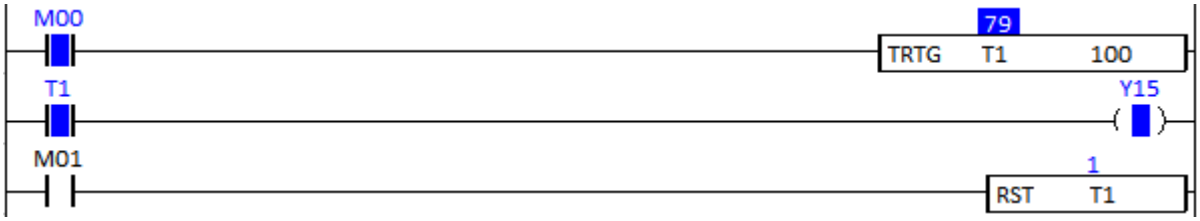
(EX. TRTG T1 100 : 10,000ms = 10sec.)

- The TRTG is a non-retentive timer. When the TRTG instruction is disabled, it clears current value and decrements again from the beginning of preset value (t)

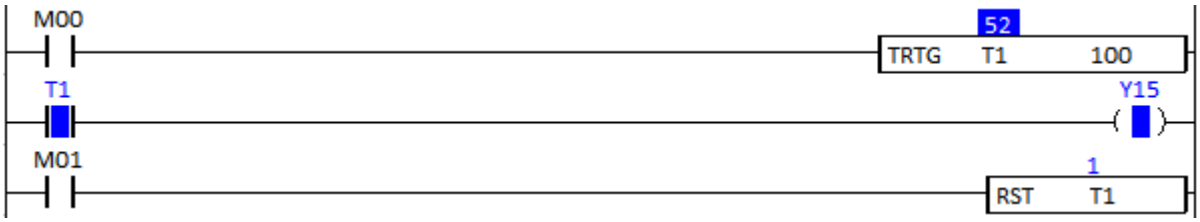
- The range of preset (t) : 0 ~ 65535

Example)

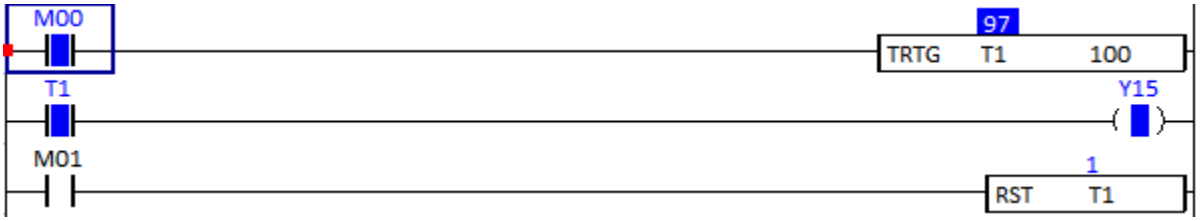
• If M00 is ON, T1 and Y15 are turned ON and TRTG instruction decrements the preset value (from 10 to 0sec). When it reaches 0sec, T1 and Y10 are turned OFF. Even if M00 is disabled while ACC value decrements, the TRTG instruction keeps decrementing preset value. When M00 is enabled again, the TRTG instruction clears current value and decrements the preset value (t) again from the beginning.



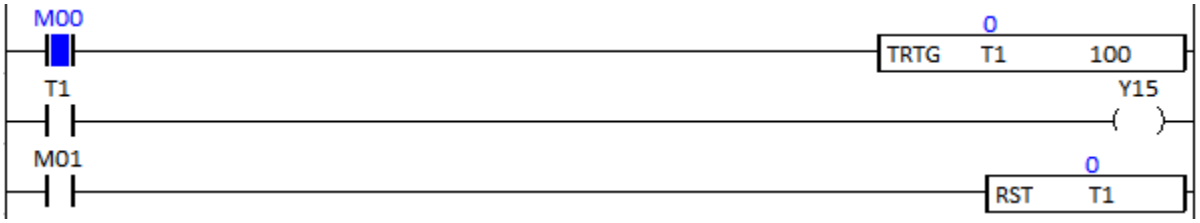
• Even if M00 is OFF, the TRTG instruction keep decrementing preset value.

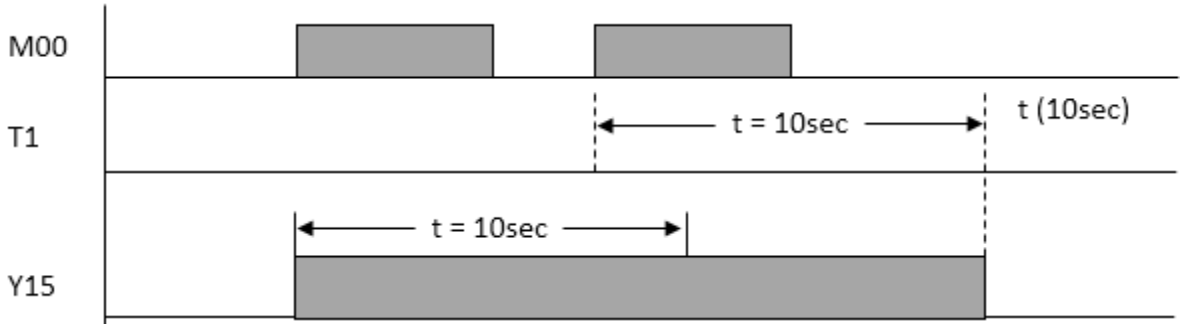


• When M00 is ON again, the TRTG instruction counts downward from 10sec again.



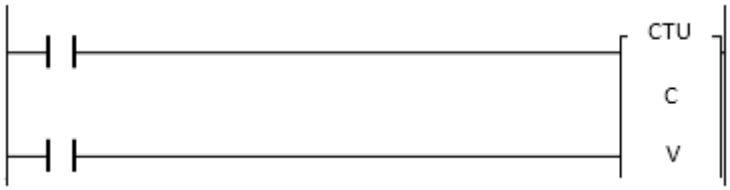
• When it reaches 0sec, T1 and Y15 are turned OFF.





2.13.6. CTU (Count Up)

CTU instruction counts upward by one. When disabled, the CTU instruction retains its ACC value.



C	Counter device address to set
V	Preset value

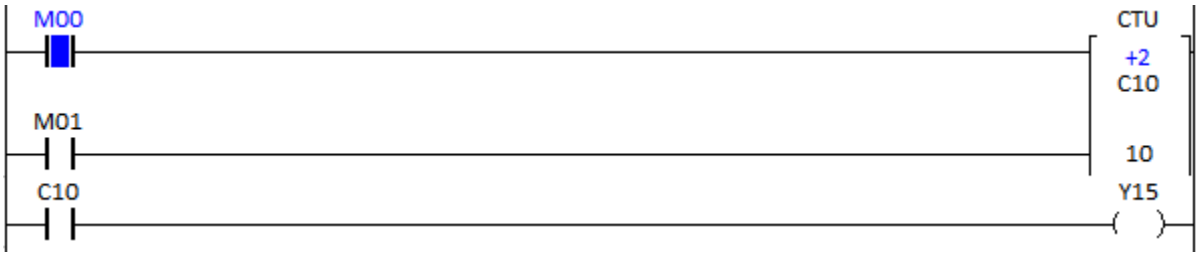
Instruction		Device address													No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
CTU	C	-	-	-	-	-	-	-	o	-	-	-	-	-	3	-	-	-
	V	-	-	-	-	-	-	-	-	-	-	o	-	o				

When enabled, CTU instruction increments the counter by one. Even if the CTU instruction is disabled, it retains its ACC value. When ACC value reaches preset value, counter bit is set.

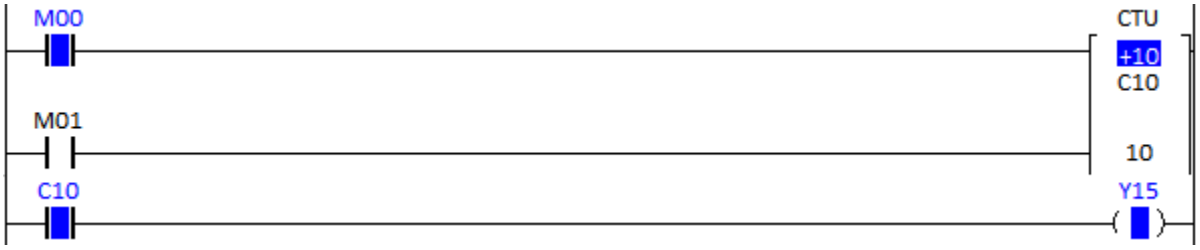
- The range of preset (v): 0 ~ 65535

Example)

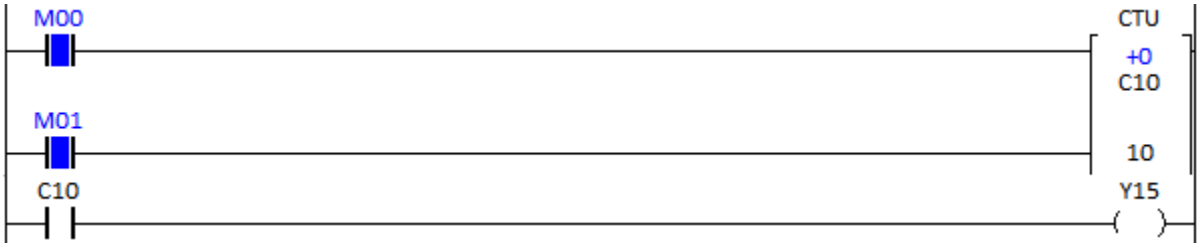
• Whenever M00 is ON, the CTU instruction increments the counter by one. When ACC value reaches 10, C10 and Y15 will be set.

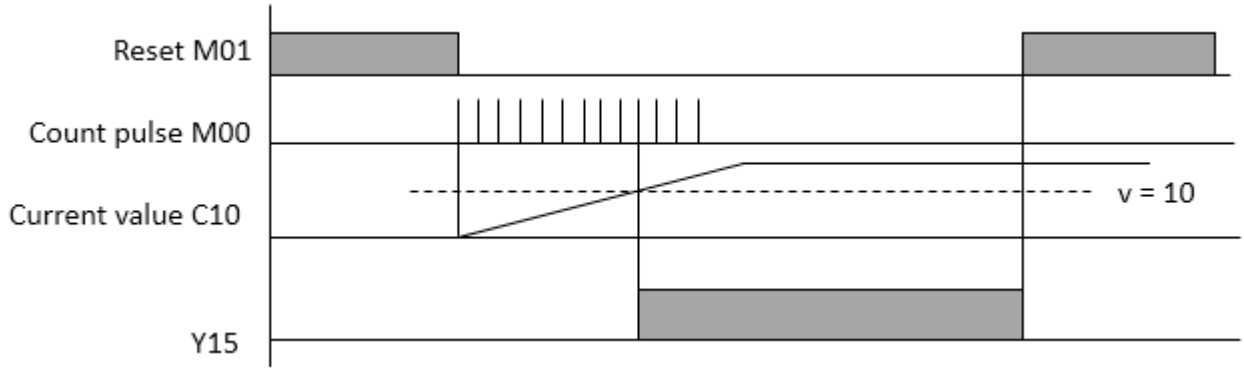


• When ACC value reaches 10, C10 and Y15 will be set.



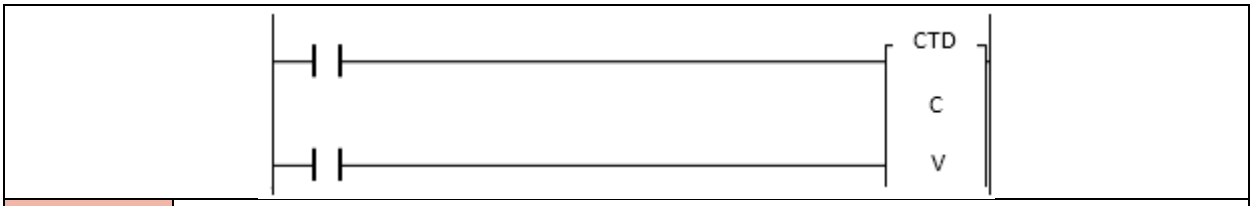
• When M01 is ON, the ACC value is cleared.





2.13.7. CTD (Count Down)

CTD instruction counts downward by one. When disabled, the CTD instruction retains its ACC value.



C	Counter device address to set
V	Preset value

Instruction		Device address												No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
CTD	C	-	-	-	-	-	-	-	o	-	-	-	-	-	3	-	-	-
	V	-	-	-	-	-	-	-	-	-	-	o	-	o				

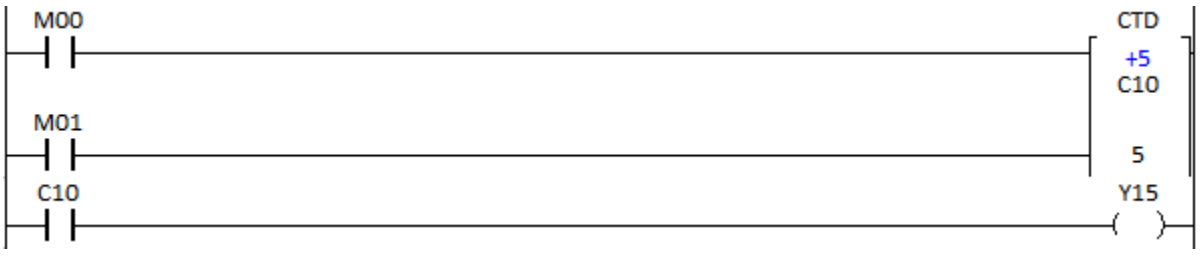
When enabled, CTD instruction decrements the counter by one. Even if the CTD instruction is disabled, it retains its ACC value. When ACC value reaches 0, counter bit is set.

- The range of preset (v) : 0 ~ 65535

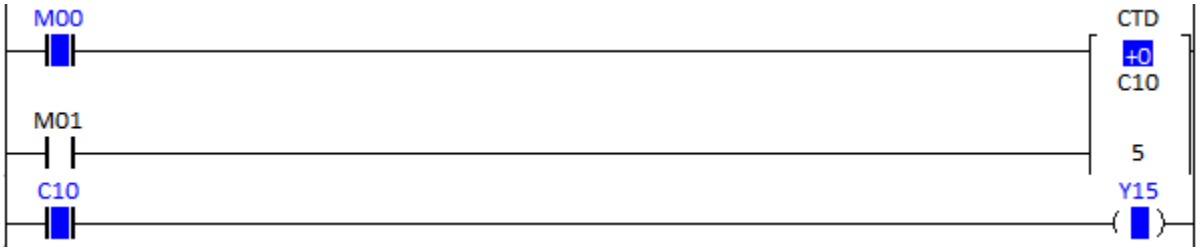
- In order to assign preset value to counter device, reset must be set at first after the ladder program downloaded to PLC.

Example)

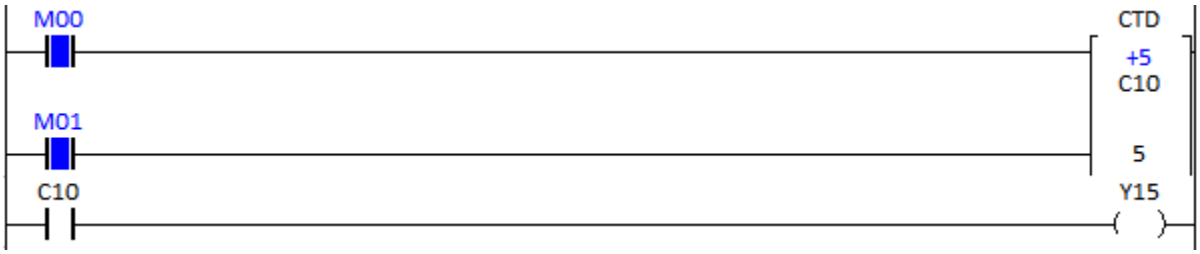
• Whenever M00 is ON, the CTD instruction decrements the counter by one. When ACC value reaches 0, C10 and Y15 will be set.

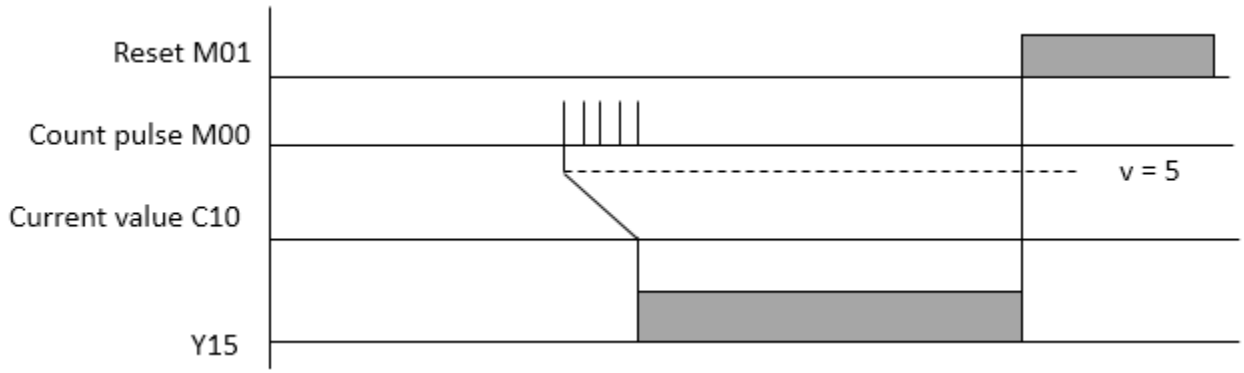


• When ACC value reaches 0, C10 and Y15 will be set.



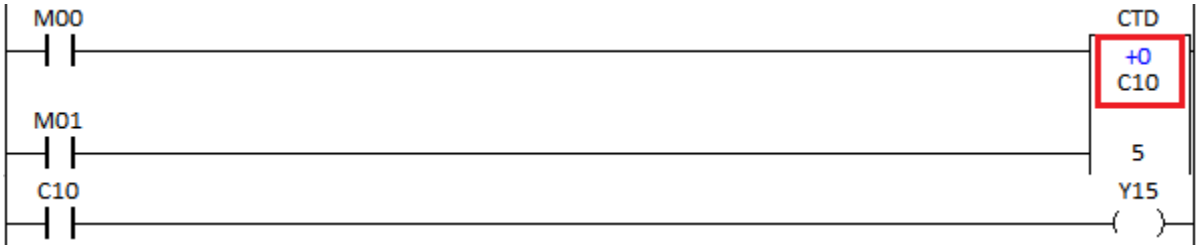
• When M01 is ON, the current value goes back to preset value 5.



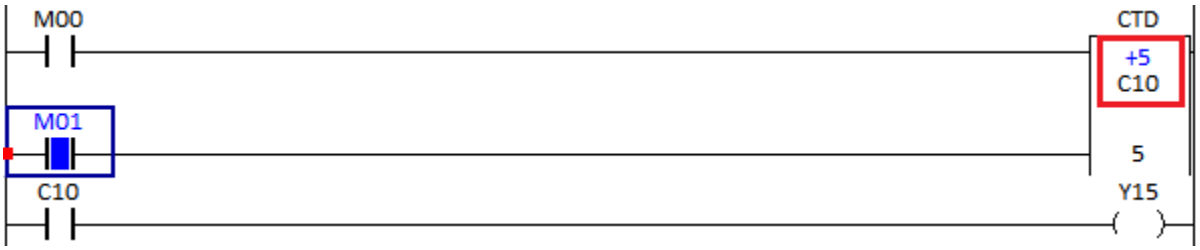


*Notice: In order to assign preset value 5 to C10, reset (M01) must be set at first after the ladder program downloaded to PLC.

- Before

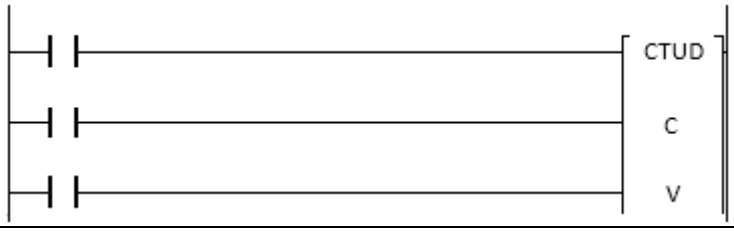


- After



2.13.8. CTUD (Count Up / Down)

CTUD instruction counts upward by one when Counter Up bit is set and the instruction counts downward by one when Counter Down bit is set.



C	Counter device address to set
V	Preset value

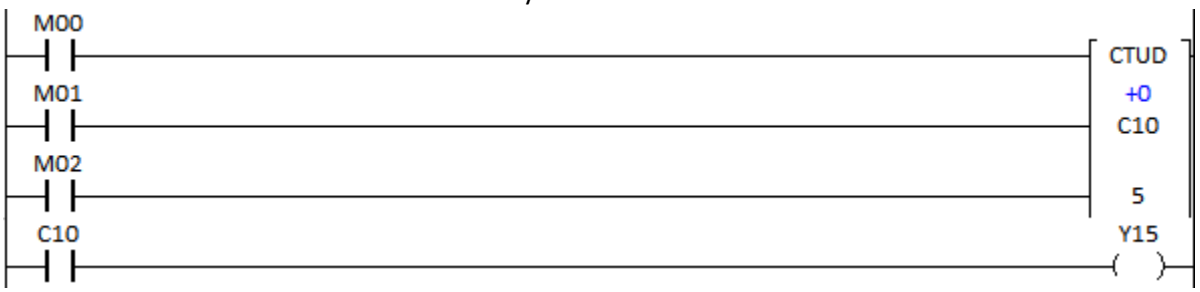
Instruction		Device address													No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
CTD	C	-	-	-	-	-	-	-	o	-	-	-	-	-	3	-	-	-
	V	-	-	-	-	-	-	-	-	-	-	o	-	o				

When Counter Up bit is enable, CTUD instruction increments the counter by one. When Counter Down bit is enable, CTUD instruction decrements the counter by one. When disabled, the CTUD instruction retains its ACC value. When ACC value reaches preset value, counter bit is set.

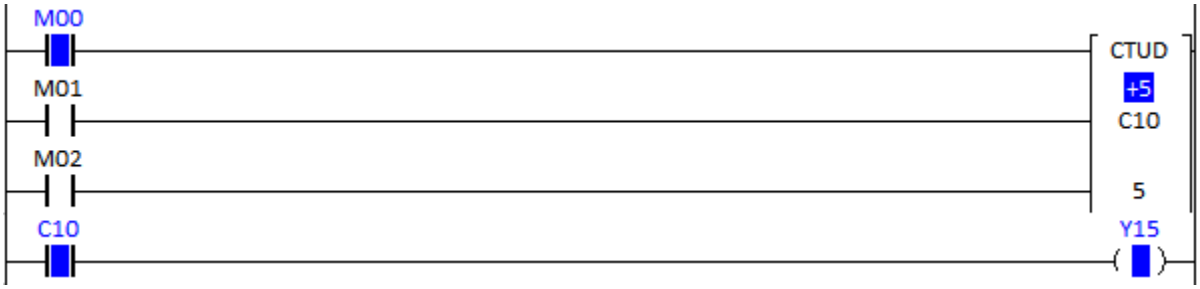
- The range of preset (v): 0 ~ 65535
- If Counter Up and Counter Down bit are set at the same time, the ACC value does not change.

Example)

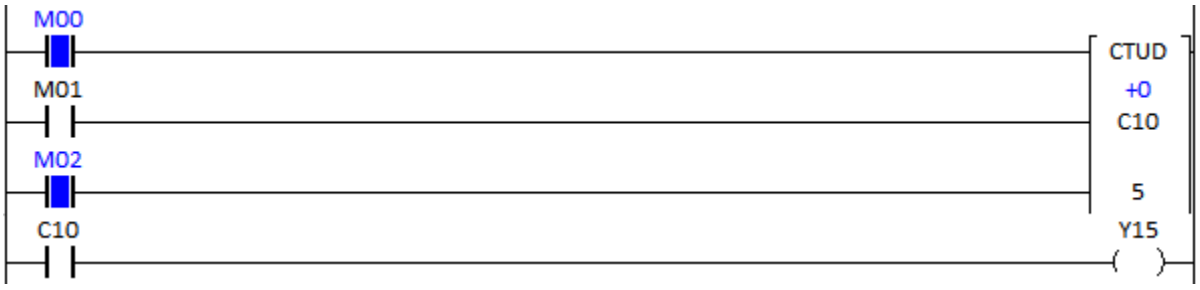
• Whenever M00 is ON, the CTUD instruction increments the counter by one. Whenever M01 is ON, the CTUD instruction decrements the counter by one.

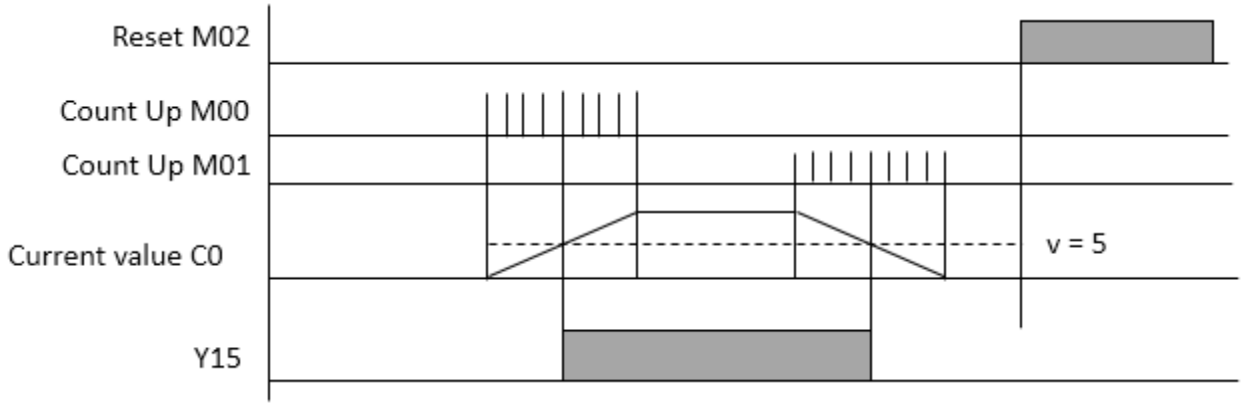


• When ACC value reaches 5, C10 and Y15 are set. (ACC ≥ preset : C10 and Y15 are set)



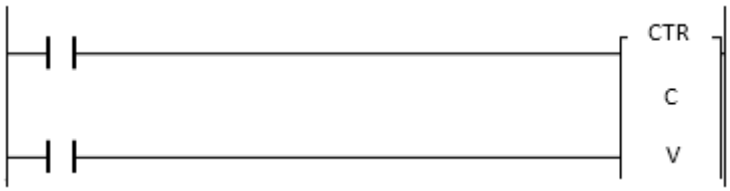
• When M02 is ON, the ACC value is cleared.





2.13.9. CTR (Count Up)

CTR instruction counts upward by one. When disabled, the CTR instruction retains its ACC value. Only if ACC value reaches preset value, counter device is set and ACC value is cleared.



C	Counter device address to set
V	Preset value

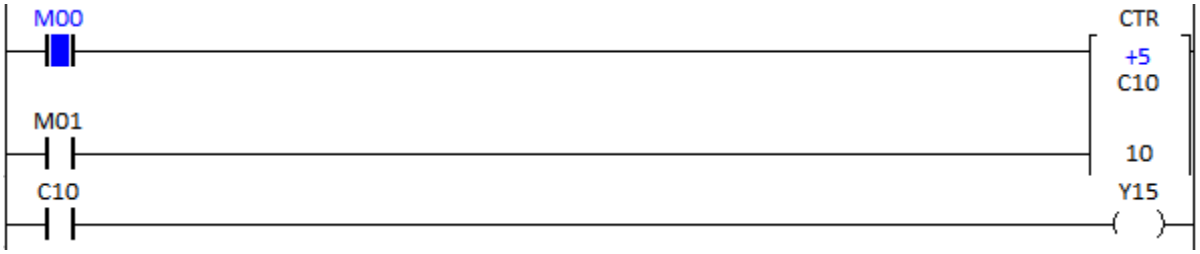
Instruction		Device address												No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
CTR	C	-	-	-	-	-	-	-	o	-	-	-	-	-	3	-	-	-
	V	-	-	-	-	-	-	-	-	-	-	o	-	o				

When enabled, CTR instruction increments the counter by one. Even if the CTU instruction is disabled, it retains its ACC value. Only if ACC value reaches preset value, counter bit is set and ACC value is cleared.

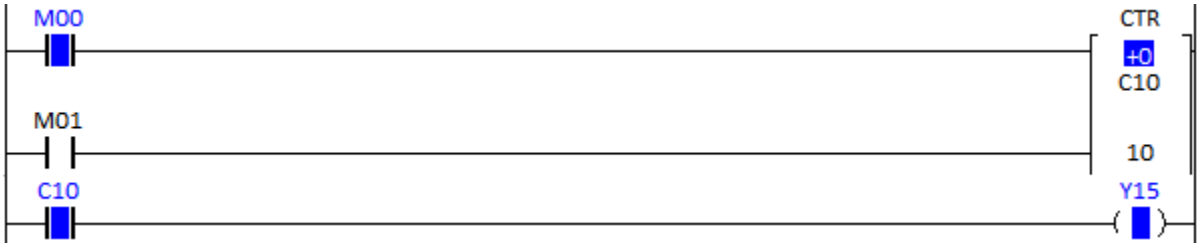
- The range of preset (v) : 0 ~ 65535

Example)

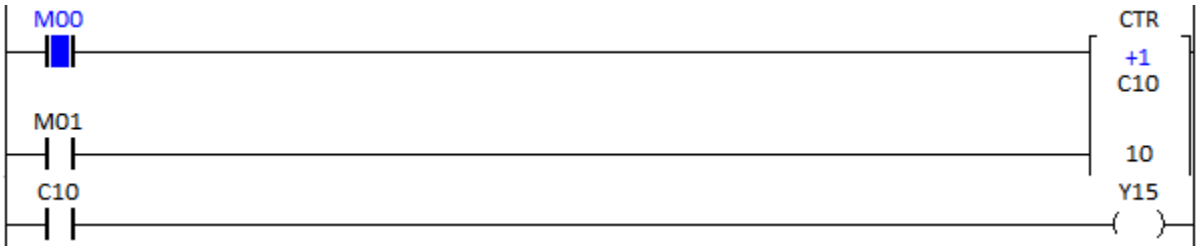
• Whenever M00 is ON, the CTR instruction increments the counter by one. When ACC value reaches 10, C10 and Y15 will be set.

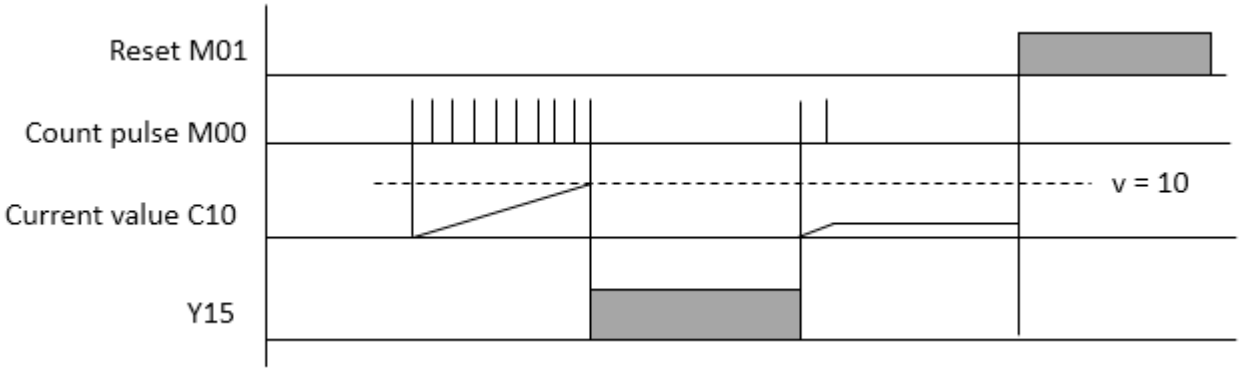


• When ACC value reaches 10, Y15 is set and the ACC value is cleared.



• When M00 is ON, C10 and Y15 are OFF and the CTR instruction counts upward by one again.



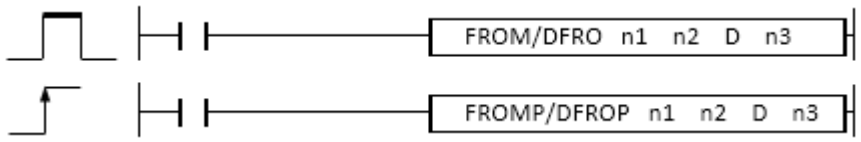


2.14. Buffer Memory Processing Instruction

2.14.1. FROM, FROMP, DFRO, DFROP

FROM instruction reads a value of buffer memory and saves its value to device address.

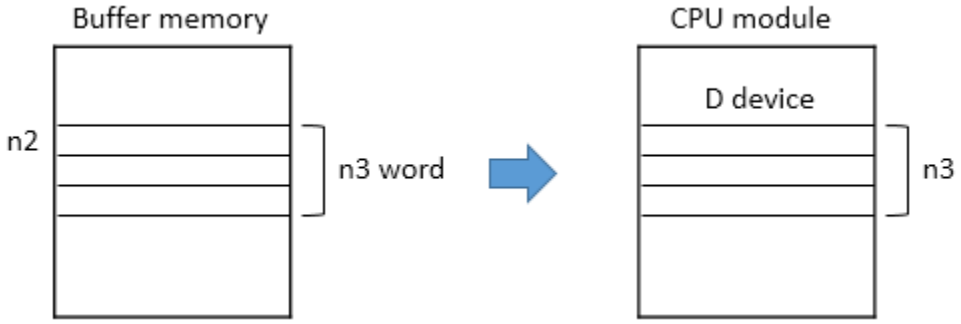
- DFRO and DFROP are Double word.



n1	Slot number of module to be read
n2	Buffer memory to be read
D	Destination
n3	Number of words to be read

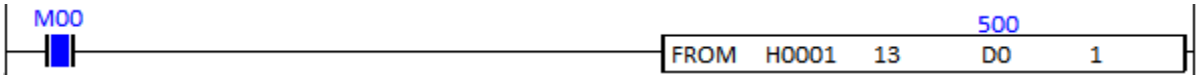
Instruction		Device address													No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
FROM FROMP DFRO DFROP	n1	o	o	o	o	o	o	o	o	-	o	o	o	o	5	o	-	-
	n2	o	o	o	o	o	o	o	o	-	o	o	o	o				
	D	o	-	o	o	o	-	o	o	-	o	o	o	-				
	n3	o	o	o	o	o	o	o	o	-	o	o	o	o				

The FROM instruction copies the word source (value of buffer memory from module) to a Destination.

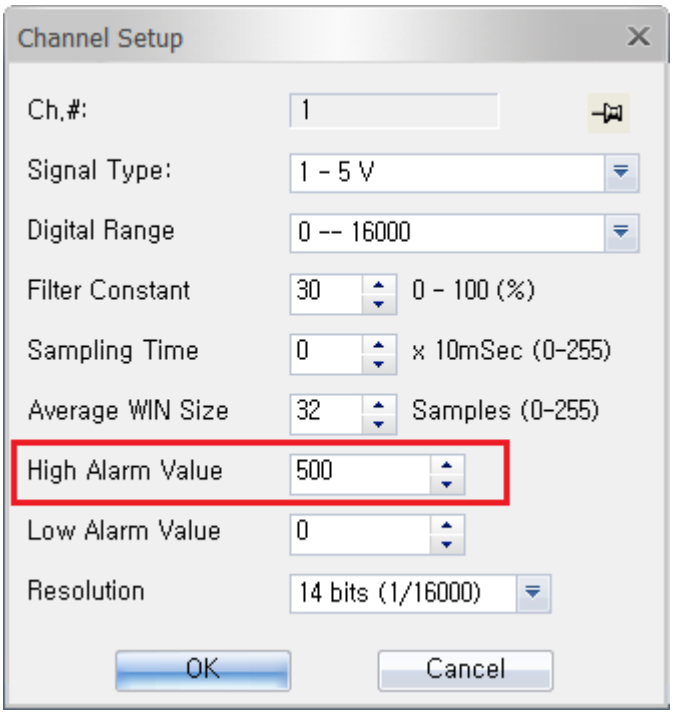


Example)

• If M00 is ON, the FROM instruction copies the value (500) of buffer memory 13 from slot number 1 module (H0001) and saves 500 to D0.

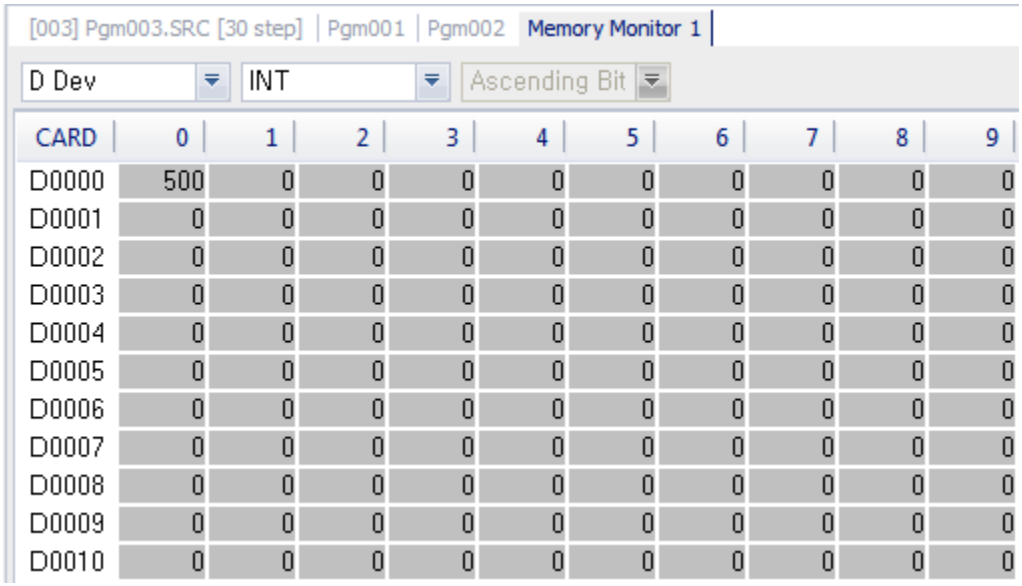


In this example, CM3-SP04EAA module is installed at the slot number 1 and the buffer memory 13 is High alarm value of Channel 1.



*Please refer to buffer memory of each module for more details.

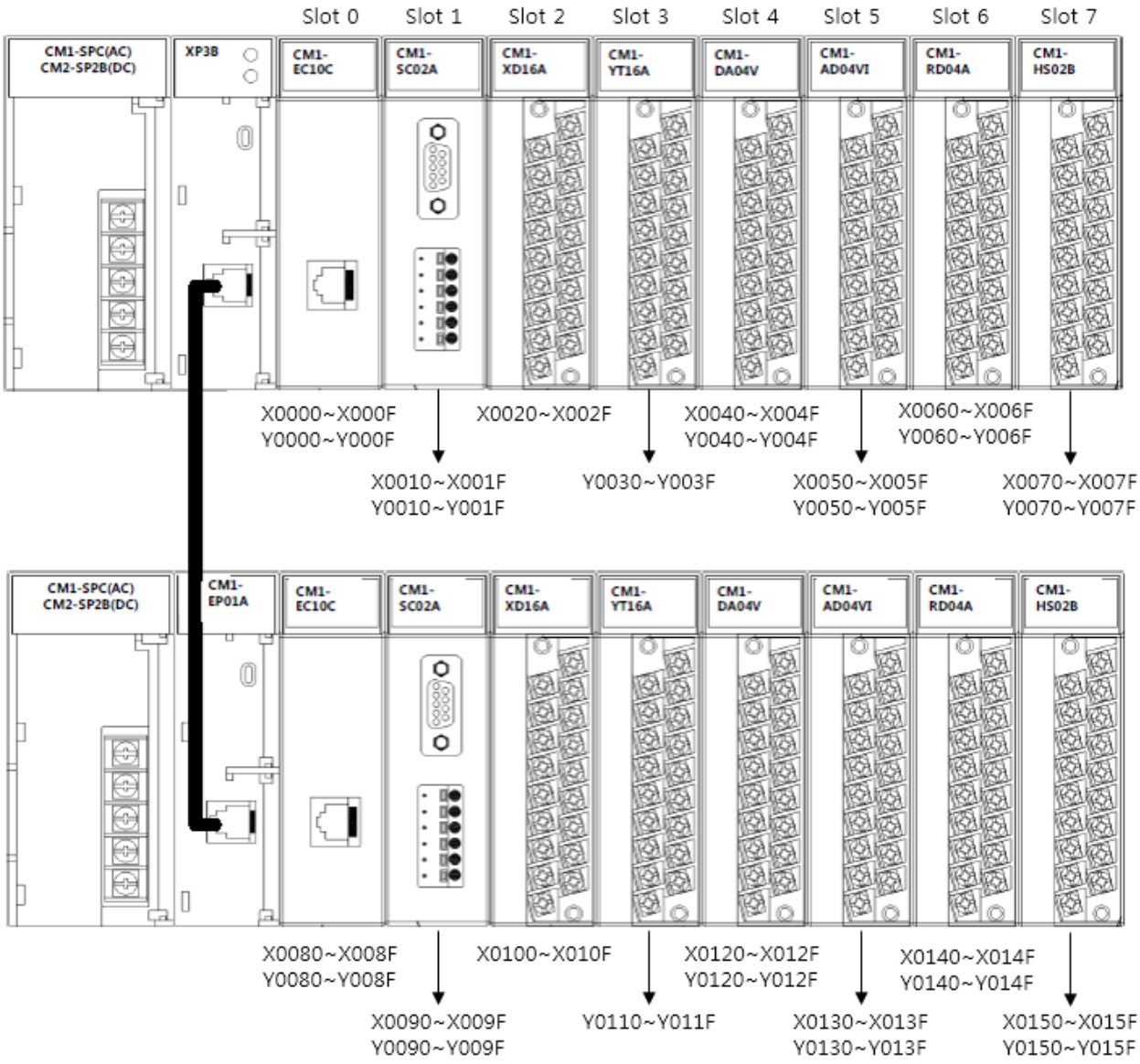
• If you check memory monitor, you can find out D0 has 500 value.



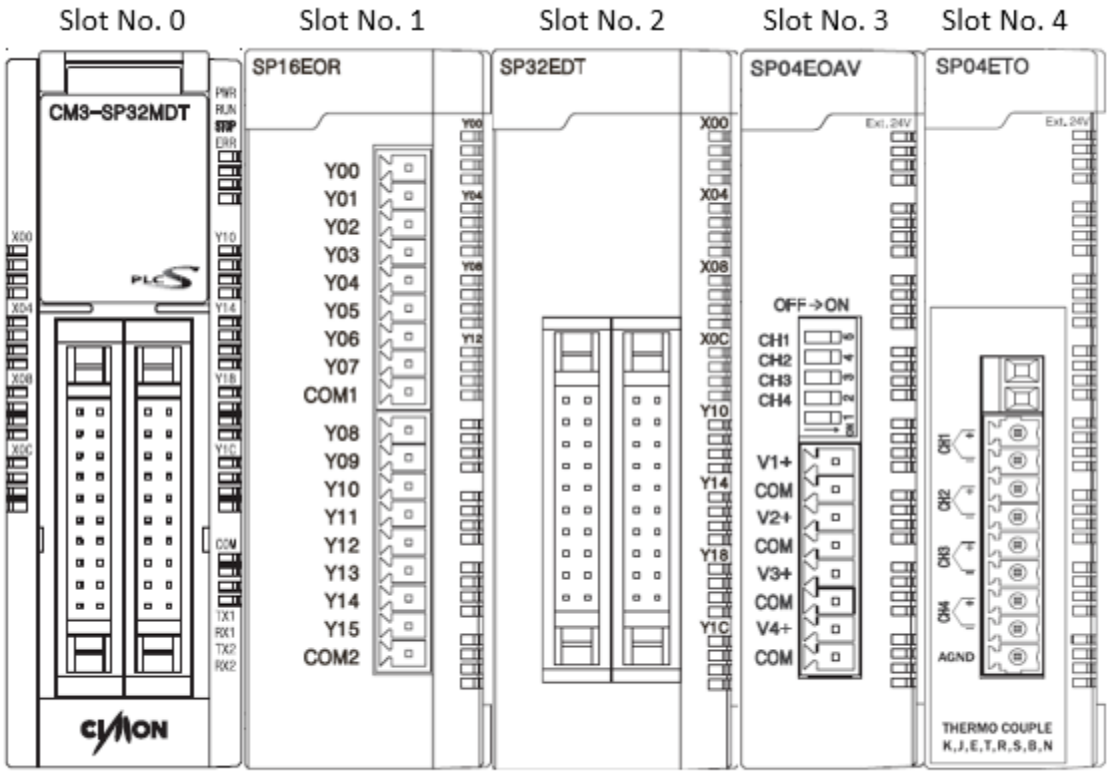
CARD	0	1	2	3	4	5	6	7	8	9
D0000	500	0	0	0	0	0	0	0	0	0
D0001	0	0	0	0	0	0	0	0	0	0
D0002	0	0	0	0	0	0	0	0	0	0
D0003	0	0	0	0	0	0	0	0	0	0
D0004	0	0	0	0	0	0	0	0	0	0
D0005	0	0	0	0	0	0	0	0	0	0
D0006	0	0	0	0	0	0	0	0	0	0
D0007	0	0	0	0	0	0	0	0	0	0
D0008	0	0	0	0	0	0	0	0	0	0
D0009	0	0	0	0	0	0	0	0	0	0
D0010	0	0	0	0	0	0	0	0	0	0

• Slot number and address assignment for CM1 and CM3 series.

1) Slot number of CM1 series



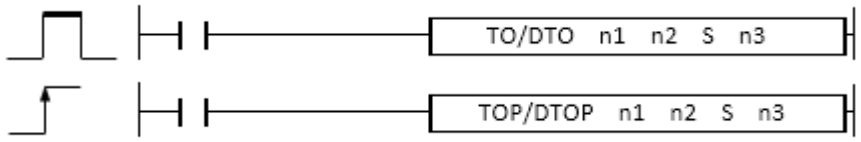
2) Slot number of CM3 Series



2.14.2. TO, TOP, DTO, DTOP

TO instruction reads a value of device address and saves its value to the buffer memory of module.

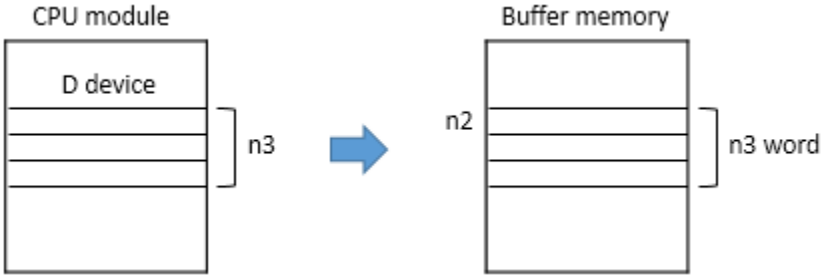
- DTO and DTOP are Double word.



n1	Slot number of module to be written
n2	Buffer memory to be written
S	Source value to write
n3	Number of words to be written

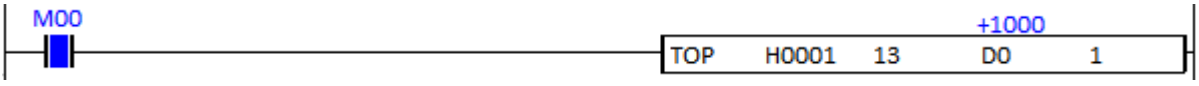
Instruction		Device address													No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
TO TOP DTO DTOP	n1	o	o	o	o	o	o	o	o	-	o	o	o	o	5	o	-	-
	n2	o	o	o	o	o	o	o	o	-	o	o	o	o				
	S	o	o	o	o	o	o	o	o	-	o	o	o	o				
	n3	o	o	o	o	o	o	o	o	-	o	o	o	o				

The TO instruction copies the source value and saves it to the buffer memory of module.

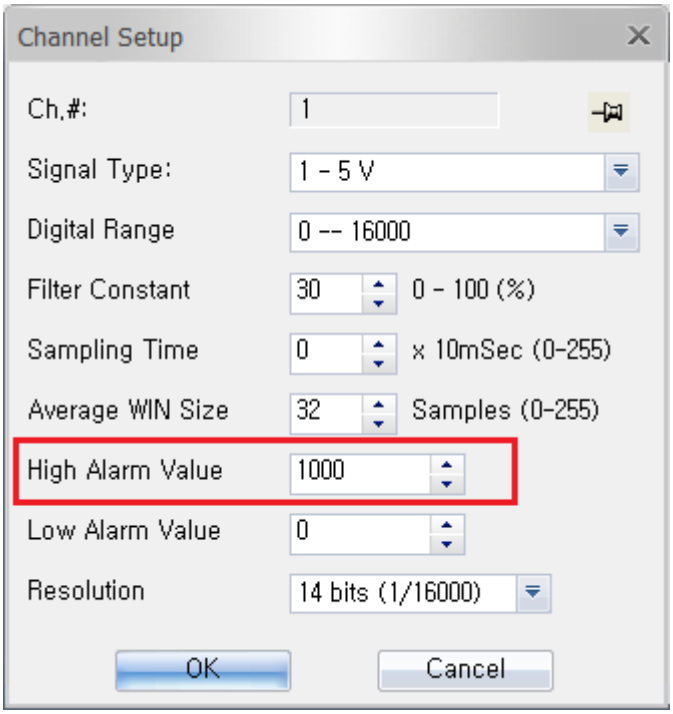


Example)

• If M00 is ON, the TOP instruction copies the value (1000) of D0 and saves 1000 to buffer memory 13 of slot number 1 module (H0001).



In this example, CM3-SP04EAA module is installed at the slot number 1 and the buffer memory 13 is High alarm value of Channel 1.



*Please refer to buffer memory of each module for more details.

• If you check memory monitor, you can find out D0 has 1000 value.

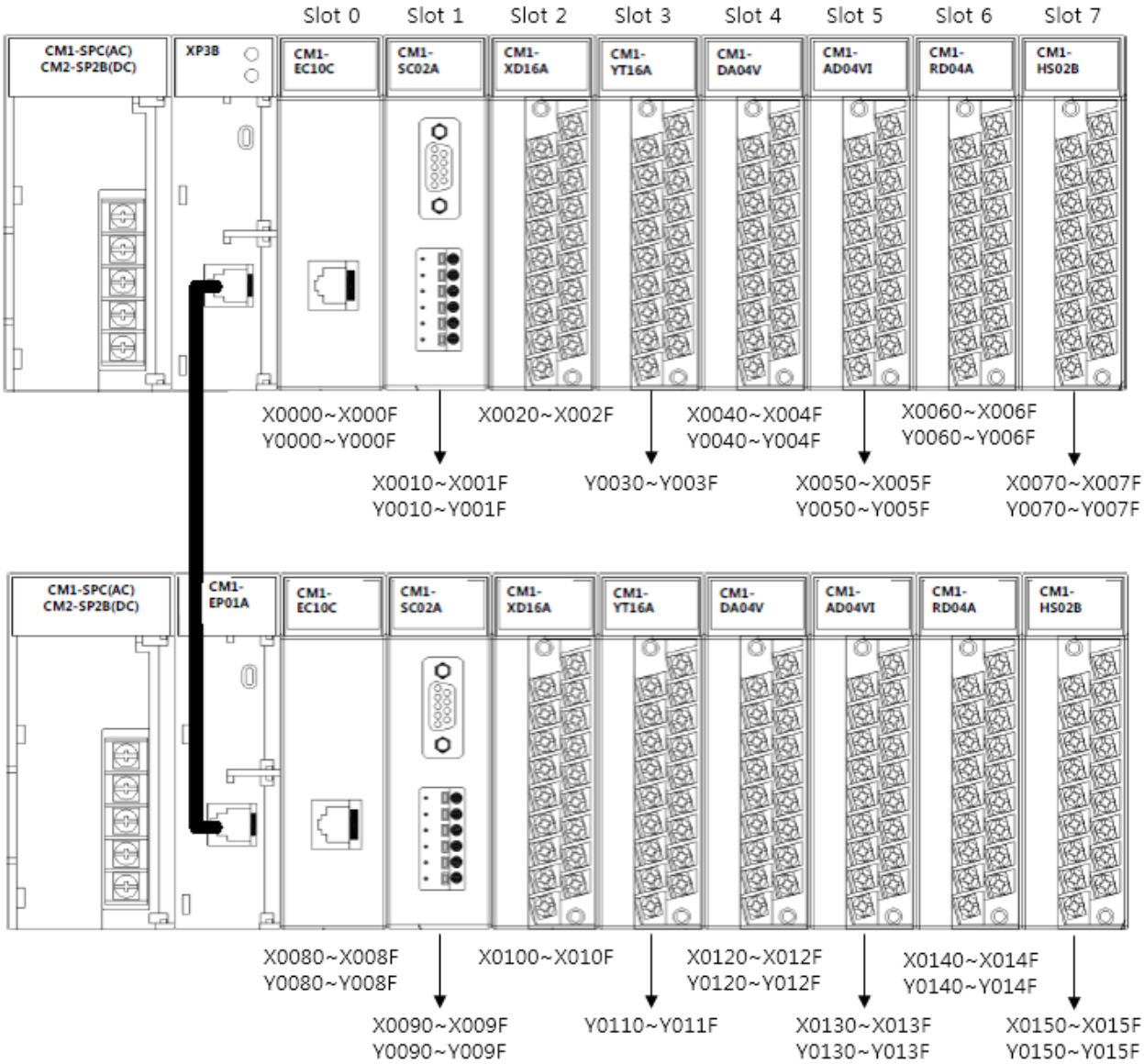
[003] Pgm003.SRC [30 step] | Pgm001 | Pgm002 | Memory Monitor 1

D Dev | INT | Ascending Bit

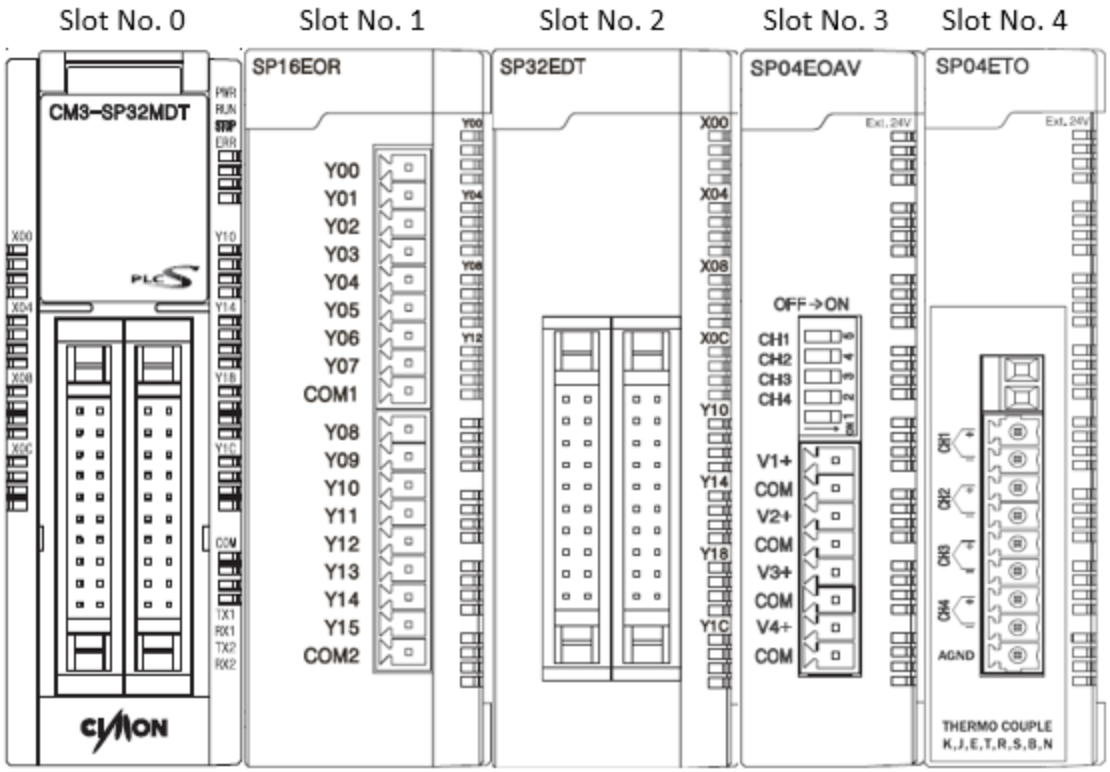
CARD	0	1	2	3	4	5	6	7	8	9
D0000	1000	0	0	0	0	0	0	0	0	0
D0001	0	0	0	0	0	0	0	0	0	0
D0002	0	0	0	0	0	0	0	0	0	0
D0003	0	0	0	0	0	0	0	0	0	0
D0004	0	0	0	0	0	0	0	0	0	0
D0005	0	0	0	0	0	0	0	0	0	0
D0006	0	0	0	0	0	0	0	0	0	0
D0007	0	0	0	0	0	0	0	0	0	0
D0008	0	0	0	0	0	0	0	0	0	0
D0009	0	0	0	0	0	0	0	0	0	0
D0010	0	0	0	0	0	0	0	0	0	0

• Slot number and address assignment for CM1 and CM3 series.

1) Slot number of CM1 series



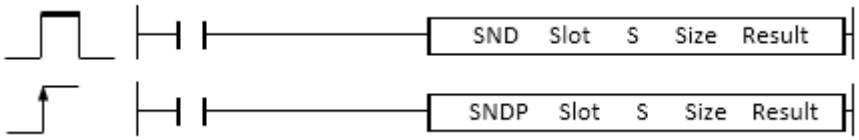
2) Slot number of CM3 Series



2.15. Data Link Instruction

2.15.1. SND, SNDP

SND instruction sends the frame data of Master station to Slave station.



Slot	Slot number of communication module
S	Source
Size	Data size to be sent
Result	Result of sending data

Instruction		Device address													No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
SND SNDP	Slot	o	o	o	o	o	o	o	o	-	o	o	o	o	5	o	-	-
	S	o	o	o	o	o	-	o	o	-	o	o	o	o				
	Size	o	o	o	o	o	-	o	o	-	o	o	o	o				
	Result	o	o	o	o	o	-	o	o	-	o	o	o	-				

When enabled, the SND instruction sends the frame data of Master module to Slave module.

Format : SND Slot (Slot no. of Comm. Module) S(Source) Size (Data size to be sent) Result

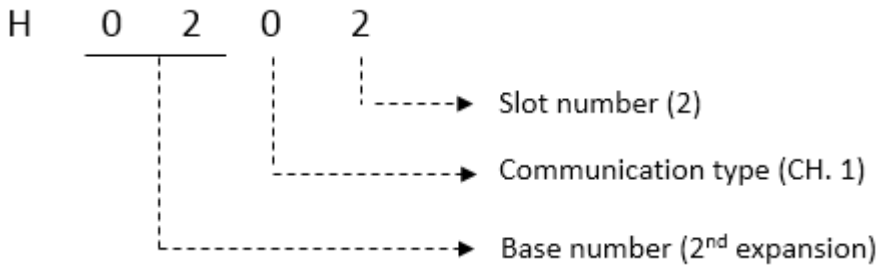
- Slot

a. It is hexadecimal.

b. Bit8 ~ F : Base expansion number (Local : 0, Expansion : 1~F)

c. Bit 4 ~ 7 : Communication type (CH.1 : 0, CH.2 : 1)

d. Bit 0 ~ 3 : Slot number



- Source (Tx Data) : Starting address which has the frame data

- Size : size(Byte) of frame data (Maximum 500Bytes)

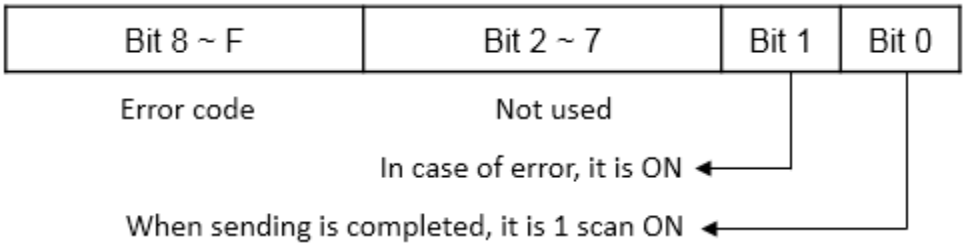
- Result : Device address which will have a communication result of sending frame data.

a. Bit 0 : When sending frame data is completed, bit 0 is 1 scan ON.

b. Bit 1 : When sending frame data is failed, bit 1 is always ON.

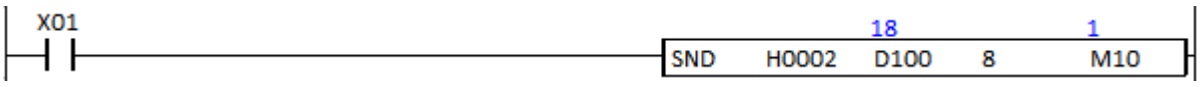
c. Bit 2 ~ 7 : OFF (Not used)

d. Bit 8 ~ F : Error Code (0 = No Error)



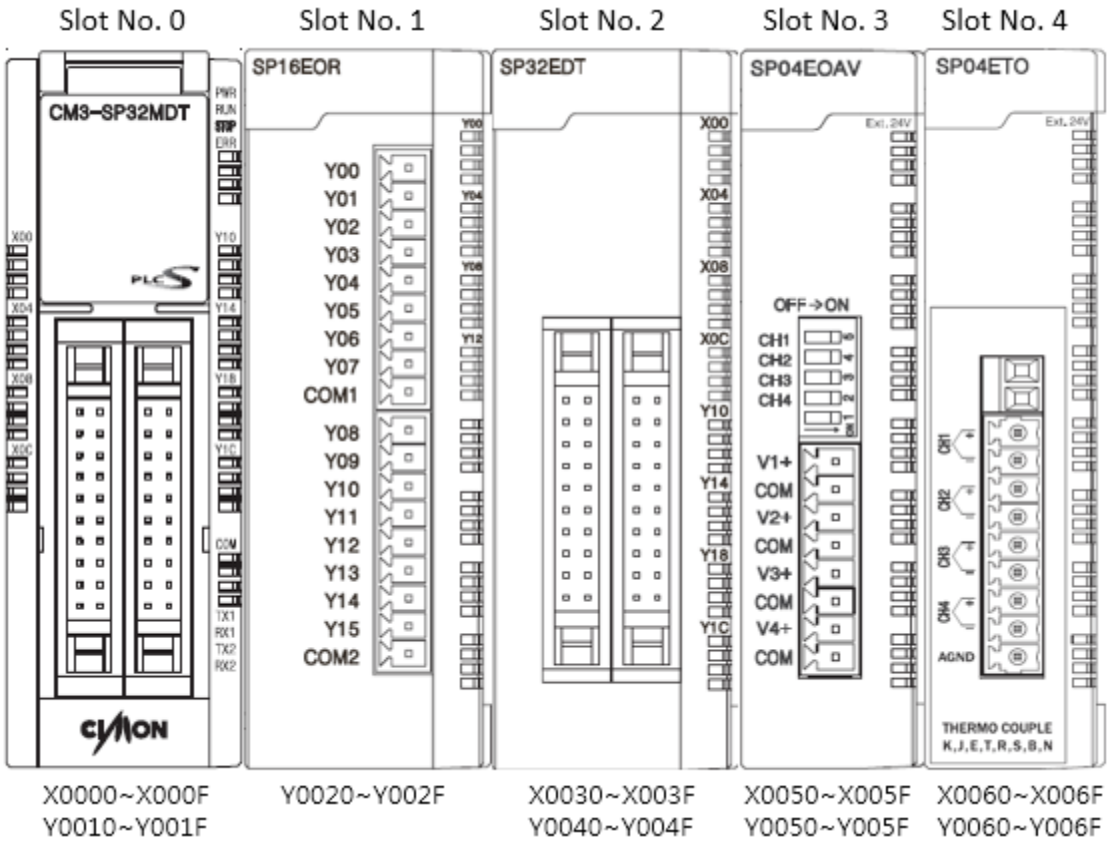
Example)

• When X01 is ON, the SND instruction sends D100~D103(4word) to slave station through channel 1 of the serial module where is installed in slot number 2 in local base. The communication result is saved in M10.

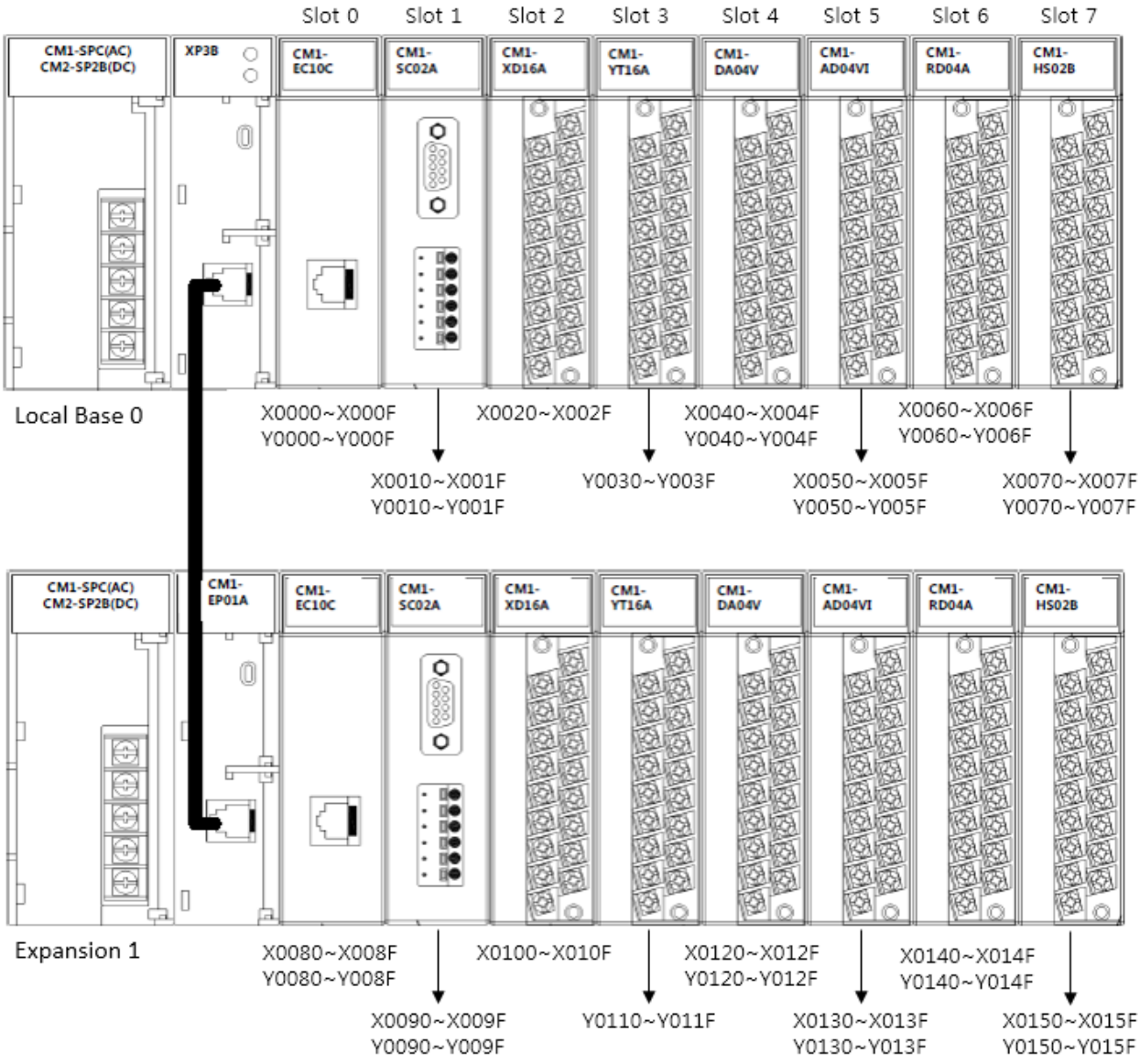


• Slot number and address assignment for CM1 and CM3 series.

1) Slot number of CM3 Series

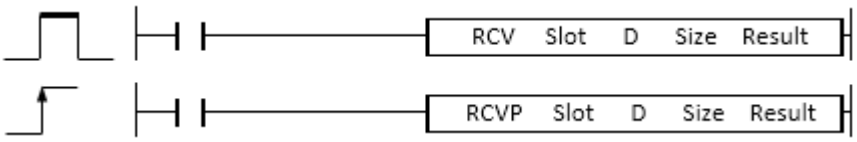


2) Slot number of CM1 Series



2.15.2. RCV, RCVP

RCV instruction receives frame data from Master station to Slave station.



Slot	Slot number of communication module
D	Destination
Size	Data size to be sent
Result	Result of receiving data

Instruction		Device address													No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
RCV RCVP	Slot	o	o	o	o	o	o	o	o	-	o	o	o	o	5	o	-	-
	D	o	o	o	o	o	-	o	o	-	o	o	o	o				
	Size	o	o	o	o	o	-	o	o	-	o	o	o	o				
	Result	o	o	o	o	o	-	o	o	-	o	o	o	-				

When enabled, the RCV instruction receives the frame data from Master module to Slave module.

Format : RCV Slot (Slot no. of Comm. Module) D(Destination) Size (Data size to be sent) Result

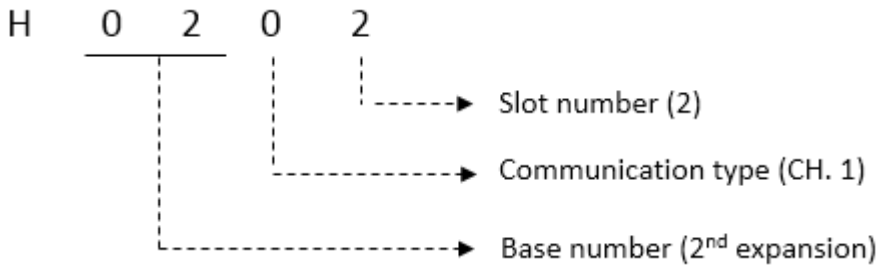
- Slot

a. It is hexadecimal.

b. Bit8 ~ F : Base expansion number (Local : 0, Expansion : 1~F)

c. Bit 4 ~ 7 : Communication type (CH.1 : 0, CH.2 : 1)

d. Bit 0 ~ 3 : Slot number



- Destination (Rx Data) : Starting device address that saves the receiving frame data.

- Size : size(Byte) of frame data (Maximum 500Bytes)

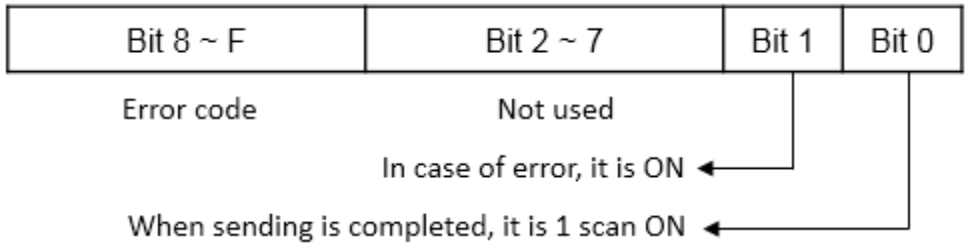
- Result : Device address which will have a communication result of receiving frame data.

a. Bit 0 : When receiving frame data is completed, bit 0 is 1 scan ON.

b. Bit 1 : When receiving frame data is failed, bit 1 is always ON.

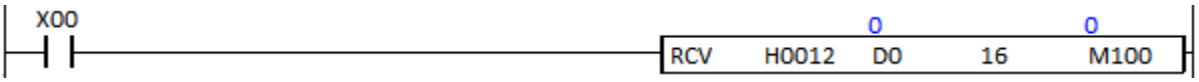
c. Bit 2 ~ 7 : OFF (Not used)

d. Bit 8 ~ F : Error Code (0 = No Error)



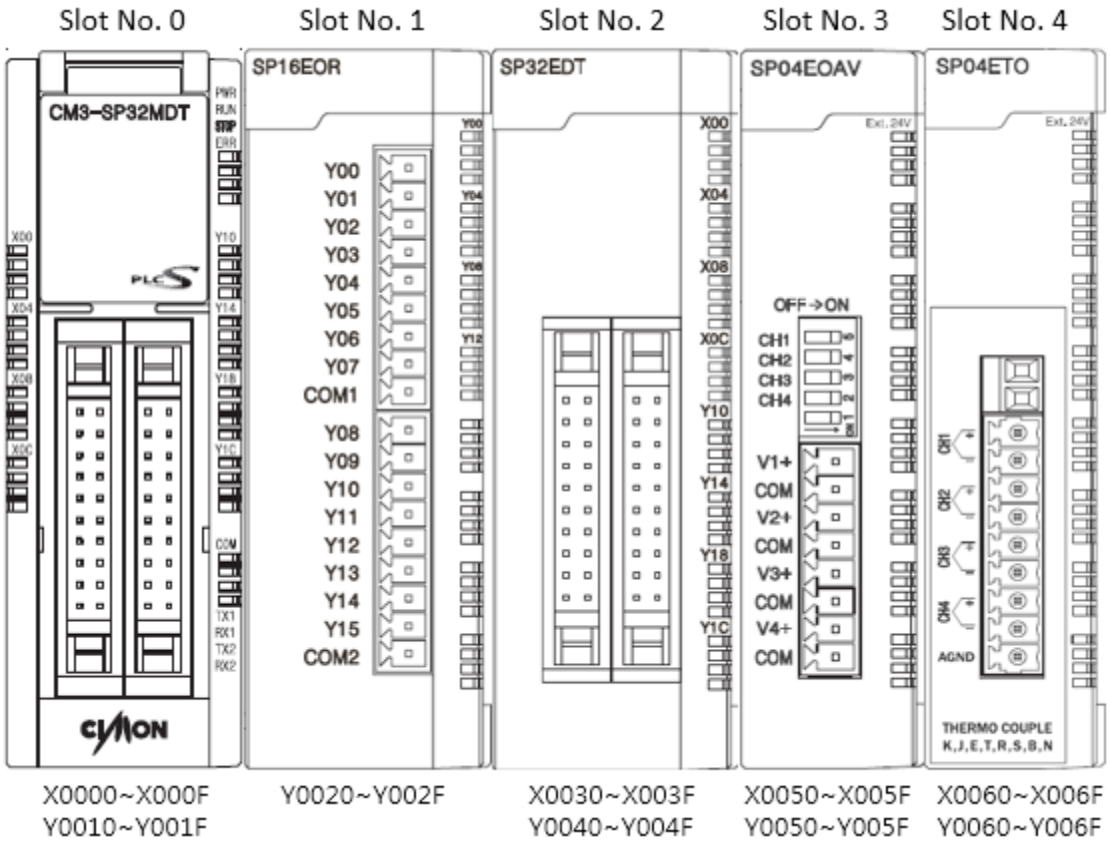
Example)

• When X00 is ON, the RCV instruction receives D0~D7 (8word) from the Master station to the slot number 2 module of local base through channel 2. The communication result is saved in M100.

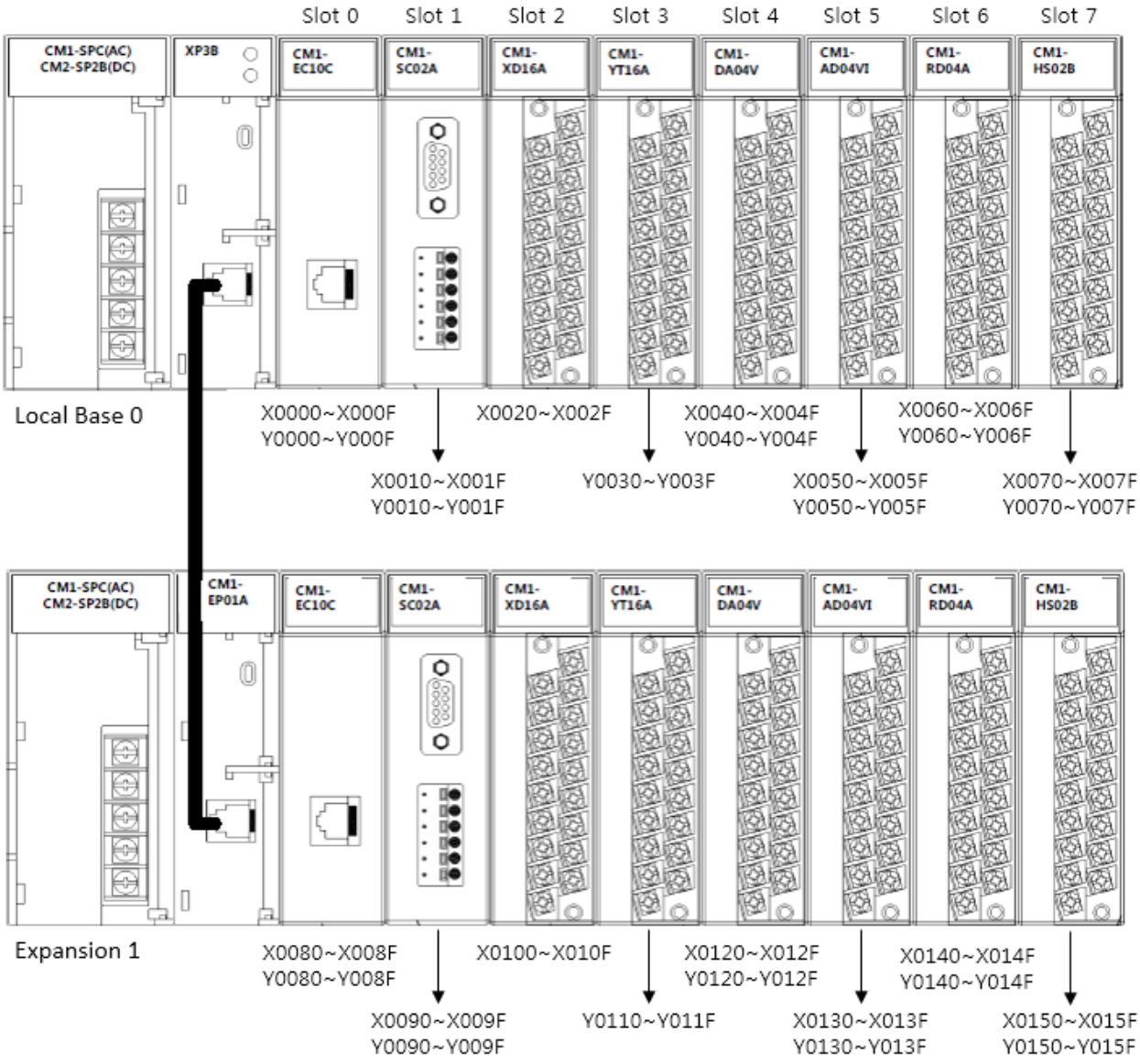


• Slot number and address assignment for CM1 and CM3 series.

1) Slot number of CM3 Series

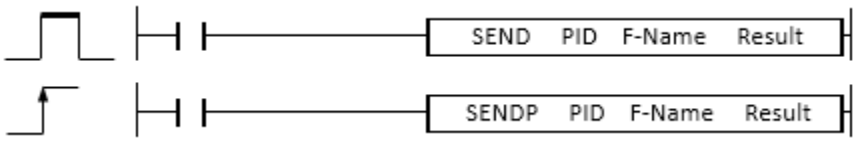


2) Slot number of CM1 Series



2.15.3. SEND, SENDP

SEND instruction sends frame data of Master station to Slave station.

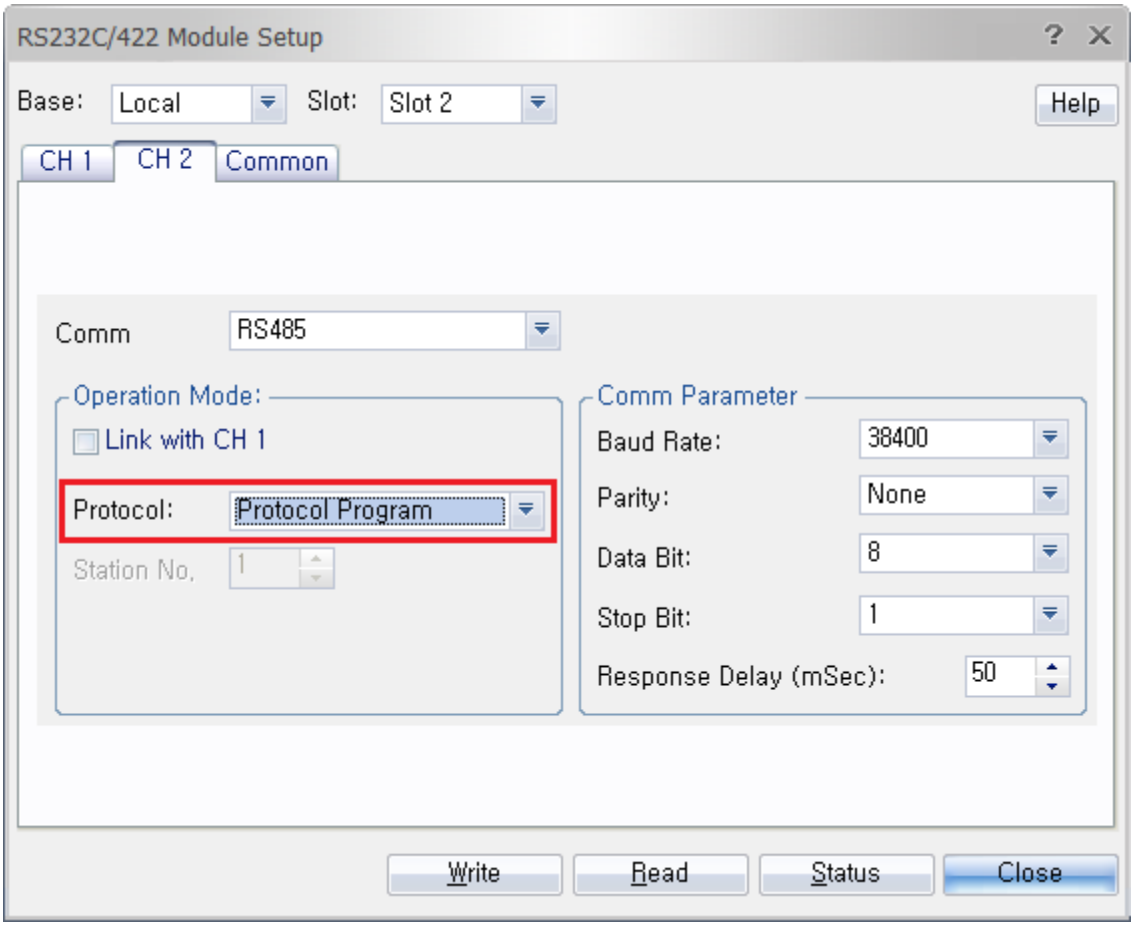


PID	Program ID number
F-Name	Frame number with communication type
Result	Result of sending frame data

Instruction		Device address													No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
SEND SENDP	PID	o	o	o	o	o	o	o	o	-	o	o	o	o	4	o	-	-
	F-Name	o	o	o	o	o	-	-	-	-	o	o	o	o				
	Result	o	o	o	o	o	-	-	-	-	o	o	o	-				

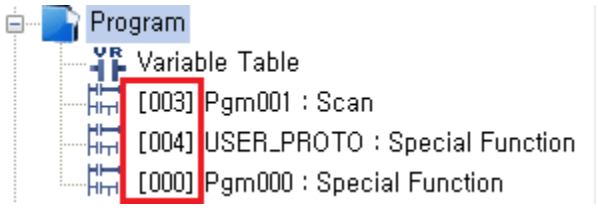
In order to use SEND instruction, the protocol type of communication module should be matched with special program.

Ex) If Serial Protocol program is used in CICON, the protocol type must be Protocol Program.



Format : SEND PID (Program ID number) F-name(frame number with communication type) Result

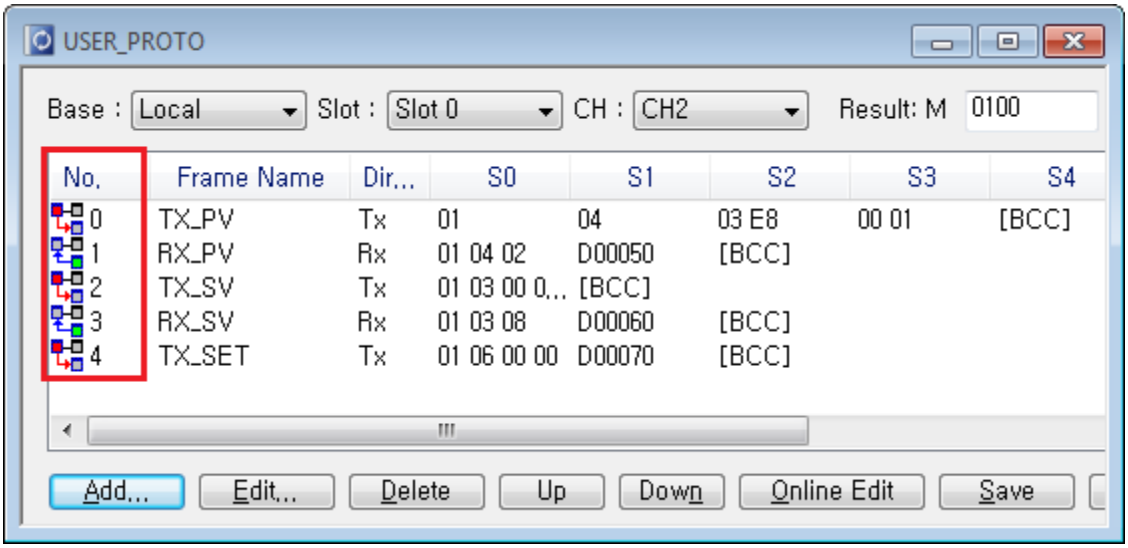
- PID : ID number of communication special program

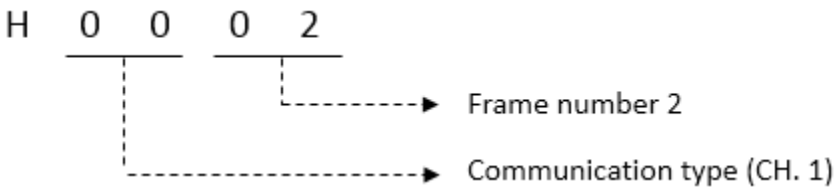


- F-Name

a. Bit8 ~ F : Communication type (CH1 : 0, CH2 : 1, Ethernet : host number)

b. Bit 0 ~ 7 : Frame number





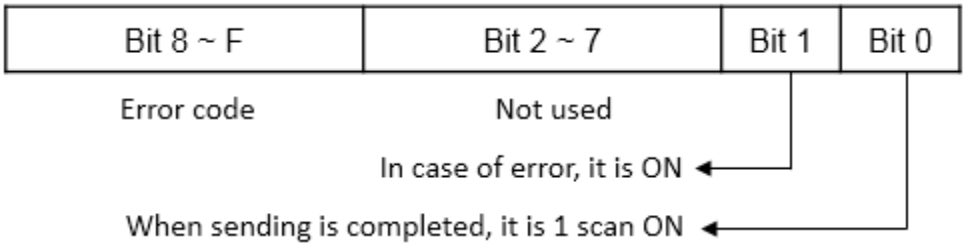
- Result : Device address which will have a communication result of sending frame data.

a. Bit 0 : When sending frame data is completed, bit 0 is 1 scan ON.

b. Bit 1 : When sending frame data is failed, bit 1 is always ON.

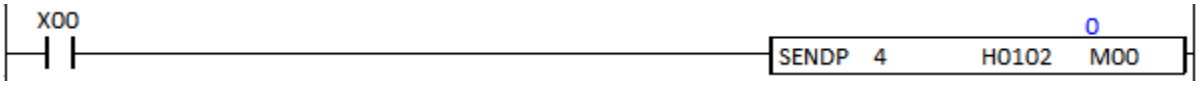
c. Bit 2 ~ 7 : OFF (Not used)

d. Bit 8 ~ F : Error Code (0 = No Error)



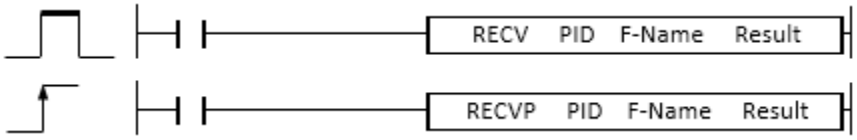
Example)

• When X00 is ON, the SEND instruction sends the frame number 2 of program number 4 to Slave station by channel 2. The communication result is saved in M00.



2.15.4. RECV, RECVP

RECV instruction receives the frame data from Master station to Slave station.

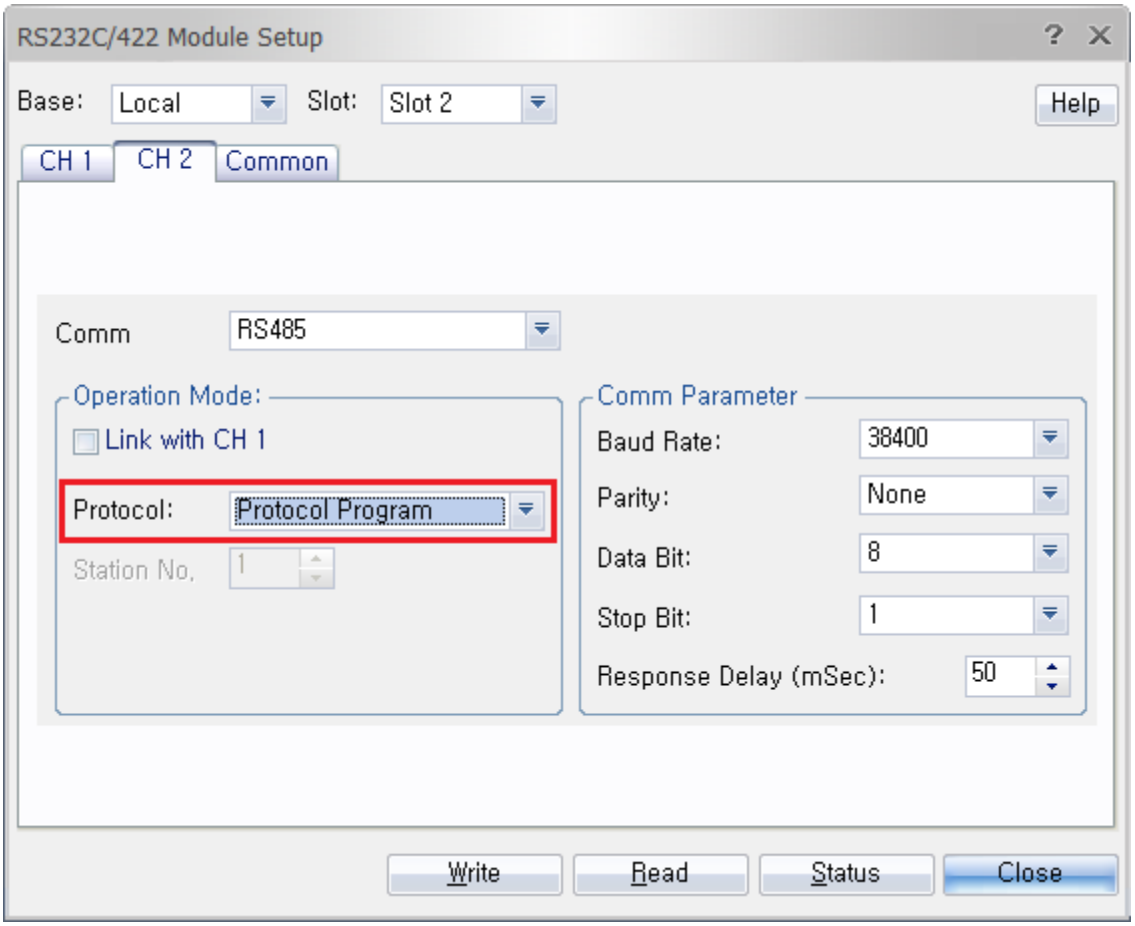


PID	Program ID number
F-Name	Frame number with communication type
Result	Result of receiving frame data

Instruction		Device address													No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
RECV RECVP	PID	o	o	o	o	o	o	o	o	-	o	o	o	o	4	o	-	-
	F-name	o	o	o	o	o	-	-	-	-	o	o	o	o				
	Result	o	o	o	o	o	-	-	-	-	o	o	o	-				

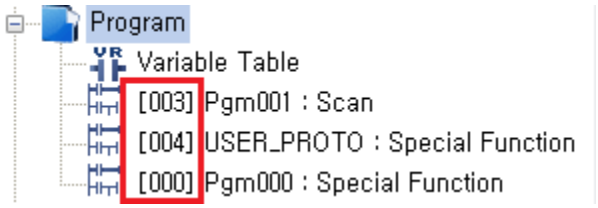
In order to use RECV instruction, the protocol type of communication module should be matched with special program.

Ex) If Serial Protocol program is used in CICON, the protocol type must be Protocol Program.



Format : RECV PID (Program ID number) F-name(frame number with communication type) Result

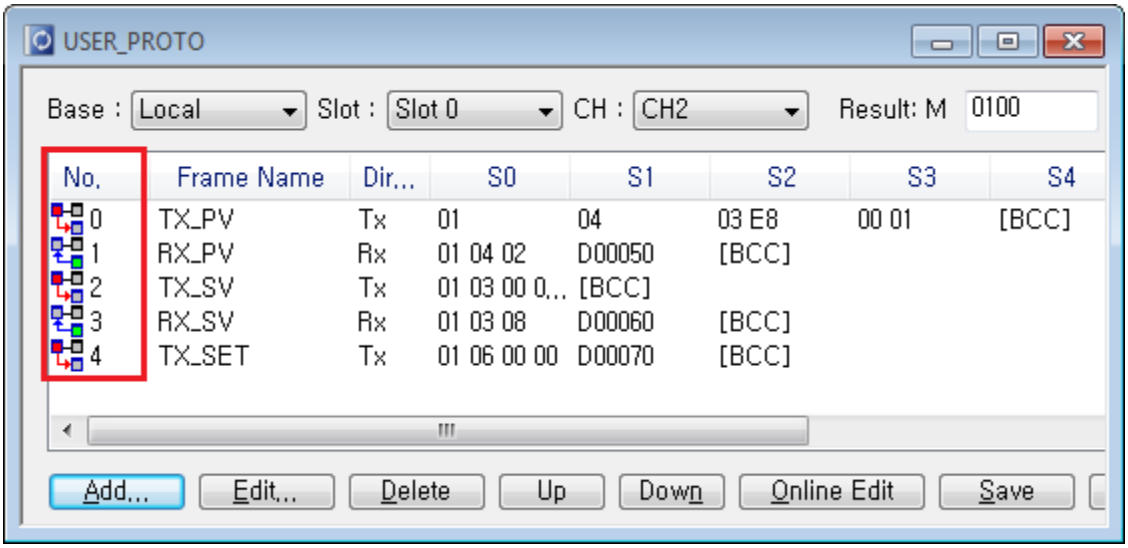
- PID : ID number of communication special program

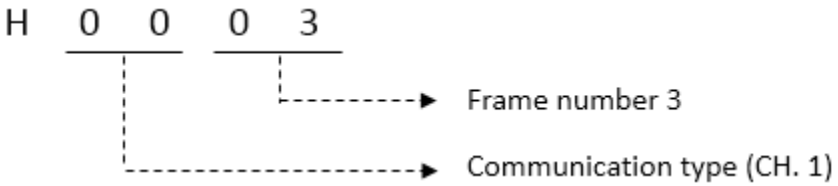


- F-Name

a. Bit8 ~ F : Communication type (CH1 : 0, CH2 : 1, Ethernet : host number)

b. Bit 0 ~ 7 : Frame number





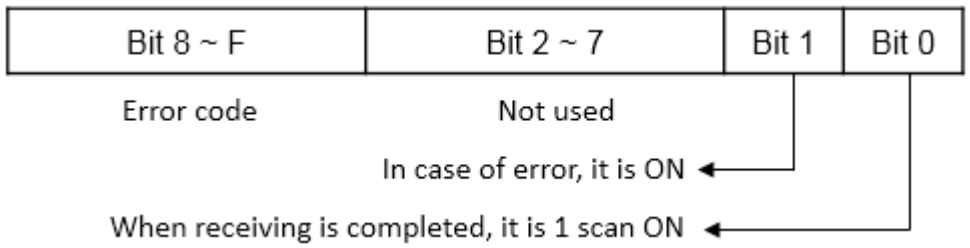
- Result : Device address which will have a communication result of receiving frame data.

a. Bit 0 : When receiving frame data is completed, bit 0 is 1 scan ON.

b. Bit 1 : When receiving frame data is failed, bit 1 is always ON.

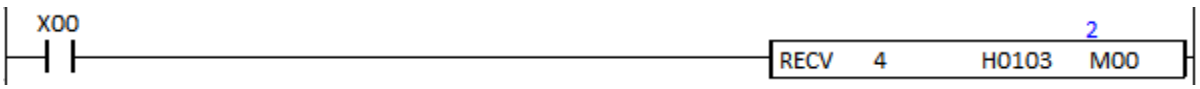
c. Bit 2 ~ 7 : OFF (Not used)

d. Bit 8 ~ F : Error Code (0 = No Error)



Example)

• When X00 is ON, the RECV instruction receives the frame data from the Master station to the frame number 3 of program number 4 by Channel 2. The communication result is saved in M00.

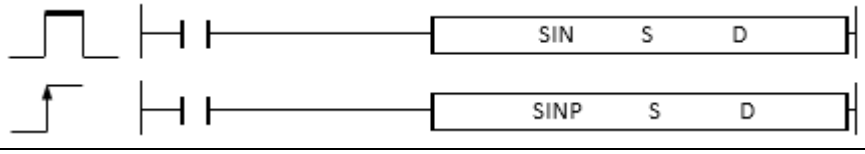


2.16. Real Number Operation Instruction

2.16.1. SIN, SINP (Sine)

SIN instruction computes the sine of source and saves the result in the Destination.

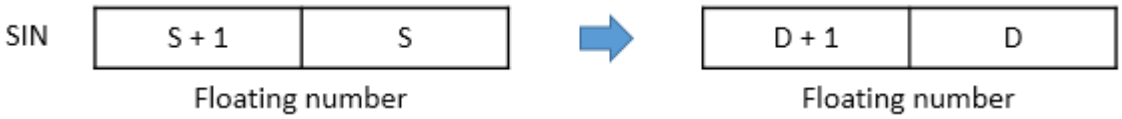
(It is supported only in XP and PLC-S CPU series)



S	Radian source
D	Destination

Instruction		Device address													No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
SIN SINP	S	o	o	o	o	o	-	-	-	-	o	o	o	-	3	o	-	-
	D	o	-	o	o	o	-	-	-	-	o	o	o	-				

When enabled, the SIN instruction computes the sine of Radian value and saves the result in Destination.



- The degree of source is set to Radian (degree x π/180)

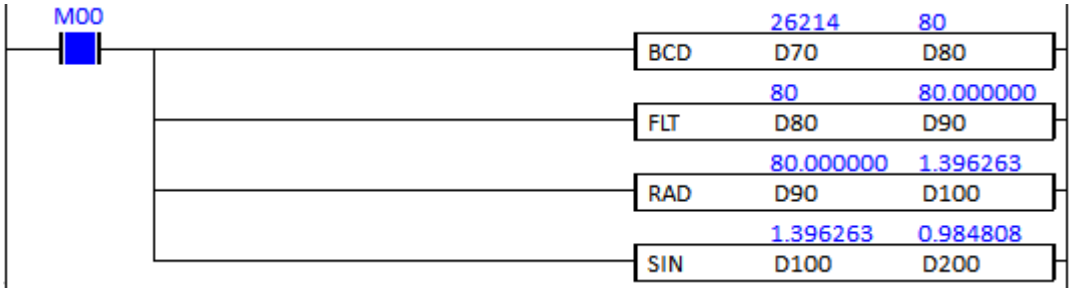
(Please refer to RAD or DEG instruction for more details)

Example)

• If M00 is ON, the degree value D70 is saved in D80.

The RAD instruction converts float value D90 to Radian value and saves its value to D100.

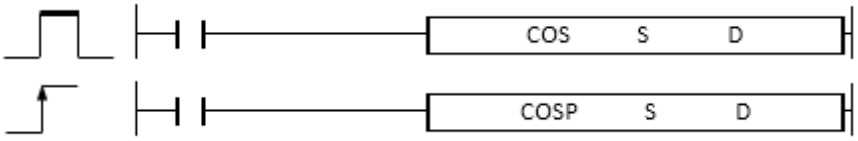
The SIN instruction computes the sine of Radian value D100 and saves its value to D200.



2.16.2. COS, COSP (Cosine)

COS instruction computes the cosine of source and saves the result in the Destination.

(It is supported only in XP and PLC-S CPU series)



S	Radian source
D	Destination

Instruction		Device address													No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
COS COSP	S	o	o	o	o	o	-	-	-	-	o	o	o	-	3	o	-	-
	D	o	-	o	o	o	-	-	-	-	o	o	o	-				

When enabled, the COS instruction computes the cosine of Radian value and saves the result in Destination.

- The degree of source is set to Radian (degree x π/180)

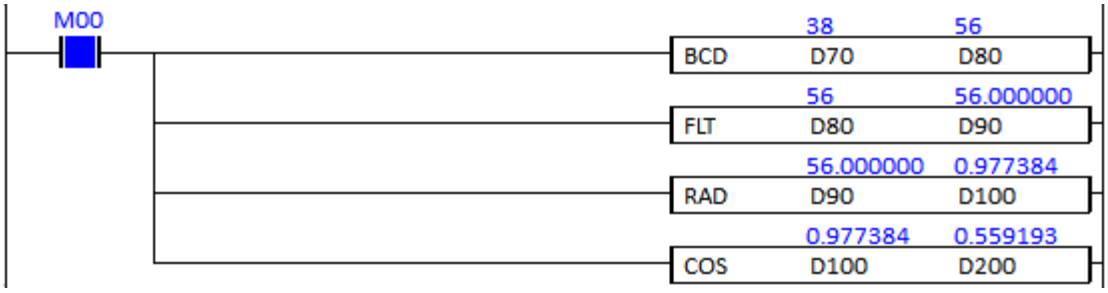
(Please refer to RAD or DEG instruction for more details)

Example)

• If M00 is ON, the degree value D70 is saved in D80.

The RAD instruction converts float value D90 to the Radian value and saves its value to D100.

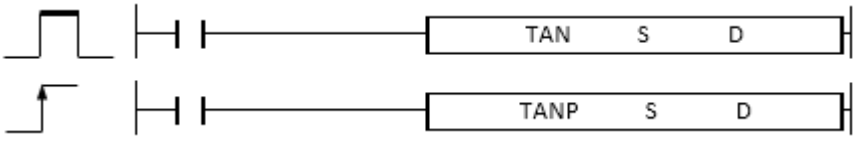
The COS instruction calculates the cosine of Radian value D100 and saves its value to D200.



2.16.3. TAN, TANP (Tangent)

TAN instruction computes the tangent of source and saves the result in the Destination.

(It is supported only in XP and PLC-S CPU series)



S	Radian source
D	Destination

Instruction		Device address													No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
TAN TANP	S	o	o	o	o	o	-	-	-	-	o	o	o	-	3	o	-	-
	D	o	-	o	o	o	-	-	-	-	o	o	o	-				

When enabled, the TAN instruction computes the tangent of Radian value and saves the result in Destination.

- The degree of source is set to Radian (degree x π/180)

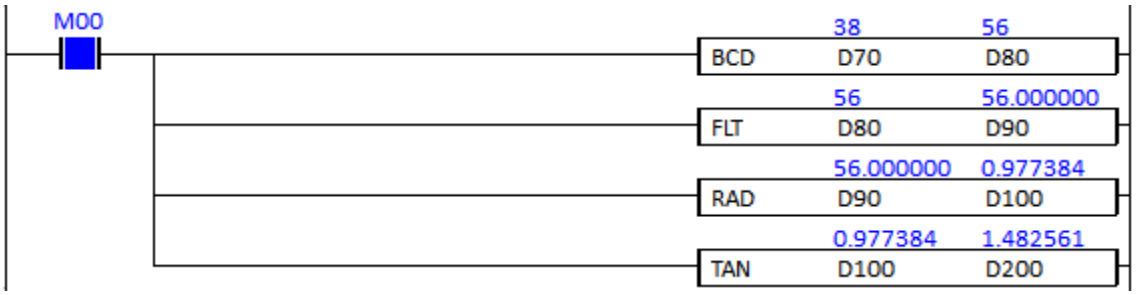
(Please refer to RAD or DEG instruction for more details)

Example)

• If M00 is ON, the degree value D70 is saved in D80.

The RAD instruction converts float value D90 to the Radian value and save its values to D100.

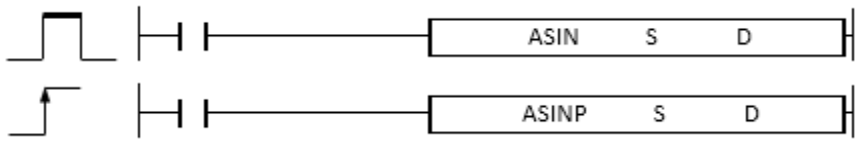
The TAN instruction calculates the tangent of Radian value D100 and save its values to D200.



2.16.4. ASIN, ASINP (Arc Sine)

ASIN instruction computes the arc sine of source and saves the Radian result in the Destination.

(It is supported only in XP and PLC-S CPU series)



S	Source
D	Destination in radians

Instruction		Device address													No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
ASIN ASINP	S	o	o	o	o	o	-	-	-	-	o	o	o	-	3	o	-	-
	D	o	-	o	o	o	-	-	-	-	o	o	o	-				

When enabled, the ASIN instruction computes the arc sine of value and saves the result in Destination (in radians).

- The range of source : -1.0 ~ 1.0

- The result is saved in the Destination (in radians)

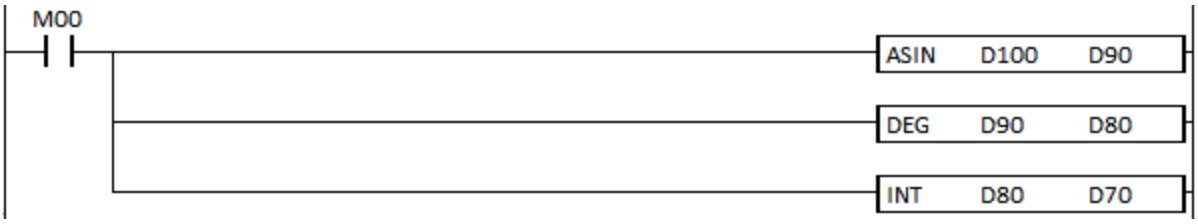
(Please refer to RAD or DEG instruction for more details)

Example)

• If M00 is ON, the ASIN calculates the arc sine of D100 and saves the result in D90 (Radian value).

The DEG instruction converts Radian value D90 to float value and saves its value to D80.

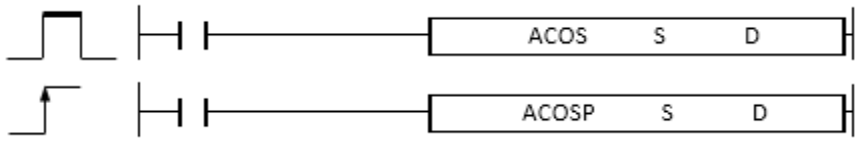
The INT instruction converts float value D80 to BCD value and saves its value to D70.



2.16.5. ACOS, ACOSP (Arc Cosine)

ACOS instruction computes the arc cosine of source and saves the Radian result in the Destination.

(It is supported only in XP and PLC-S CPU series)



S	Source
D	Destination in radians

Instruction		Device address													No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
ACOS ACOSP	S	o	o	o	o	o	-	-	-	-	o	o	o	-	3	o	-	-
	D	o	-	o	o	o	-	-	-	-	o	o	o	-				

When enabled, the ACOS instruction computes the arc cosine of source and saves the result in Destination (in radians).

- The range of source : -1.0 ~ 1.0

- The result is saved in the Destination (in radians)

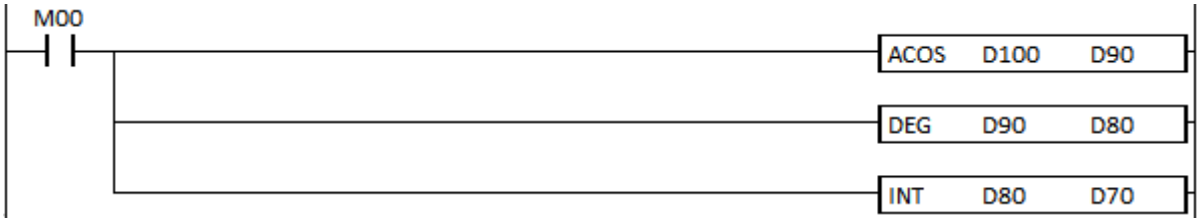
(Please refer to RAD or DEG instruction for more details)

Example)

• If M00 is ON, the ACOS calculates the arc cosine of D100 and saves the result in D90 (Radian value).

The DEG instruction converts Radian value D90 to float value and saves its value to D80.

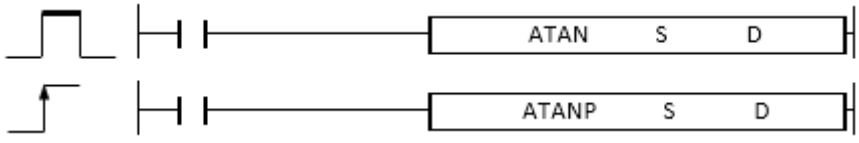
The INT instruction converts float value D80 to BCD value and saves its value to D70.



2.16.6. ATAN, ATANP (Arc Tangent)

ATAN instruction computes the arc tangent of source and saves the Radian result in the Destination.

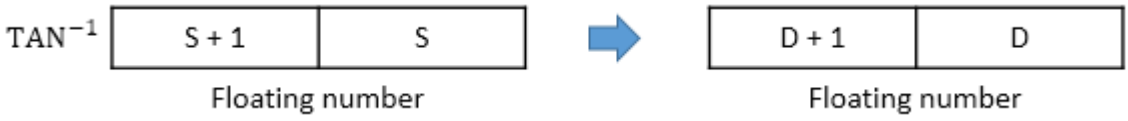
(It is supported only in XP and PLC-S CPU series)



S	Source
D	Destination in radians

Instruction		Device address													No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
ATAN ATANP	S	o	o	o	o	o	-	-	-	-	o	o	o	-	3	o	-	-
	D	o	-	o	o	o	-	-	-	-	o	o	o	-				

When enabled, the ATAN instruction computes the arc tangent of source and saves the result in Destination (in radians).



- The range of source : -1.0 ~ 1.0

- The result is saved in the Destination (in radians)

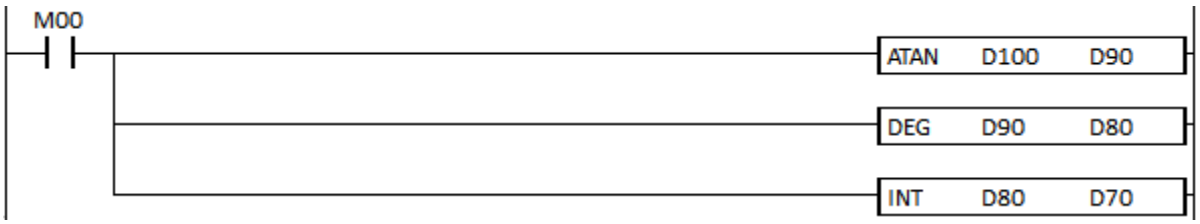
(Please refer to RAD or DEG instruction for more details)

Example)

• If M00 is ON, the ATAN calculates the arc tangent of D100 and saves the result in D90 (Radian value).

The DEG instruction converts Radian value D90 to float value and saves its value to D80.

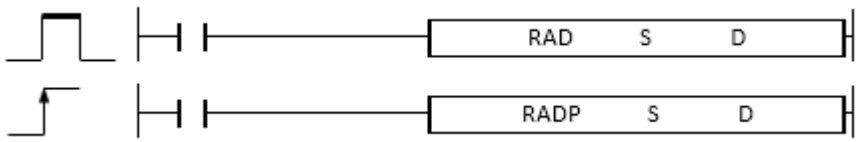
The INT instruction converts float value D80 to BCD value and saves its value to D70.



2.16.7. RAD, RADP (Radian)

RAD instruction converts source to the Radian value and saves the result to the Destination.

(It is supported only in XP and PLC-S CPU series)



S	Source in degree
D	Destination in radians

Instruction		Device address													No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
RAD RADP	S	o	o	o	o	o	-	-	-	-	o	o	o	-	3	o	-	-
	D	o	-	o	o	o	-	-	-	-	o	o	o	-				

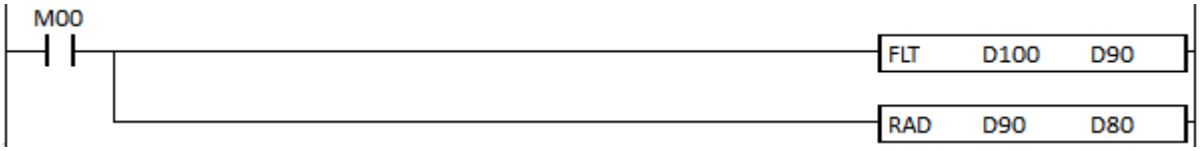
When enabled, the RAD instruction computes the source and saves the result in Destination (in radians).

- The Radian formula : degree value x (π/180)

Example)

• If M00 is ON, the FLT instruction converts D100 to float value and saves its result to D90.

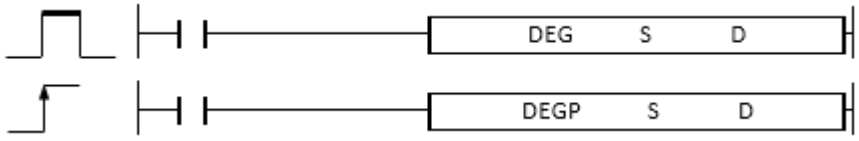
The RAD instruction converts D90 to radian value and saves its result in D80.



2.16.8. DEG, DEGP (Degrees)

DEG instruction converts Radian value to degrees and saves the result to the Destination.

(It is supported only in XP and PLC-S CPU series)



S	Radian source
D	Destination in degrees

Instruction		Device address													No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
DEG DEGP	S	o	o	o	o	o	-	-	-	-	o	o	o	-	3	o	-	-
	D	o	-	o	o	o	-	-	-	-	o	o	o	-				

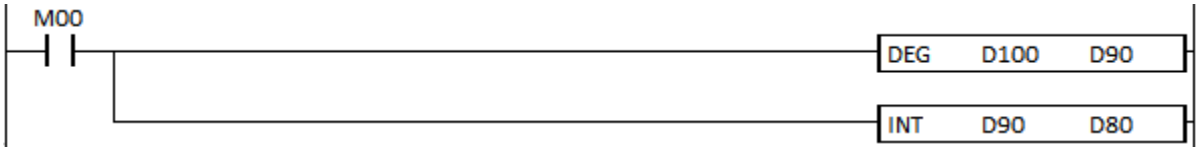
When enabled, the DEG instruction computes the radian source and saves the result in Destination (in degrees).

- The Degree formula : Radian value x (180/π)

Example)

• If M00 is ON, the DEG instruction converts D100 (Radian value) to Degree value and saves its result to D90.

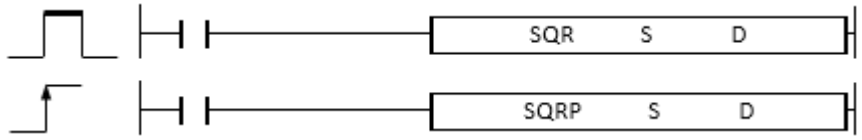
The INT instruction converts D90 (Float value) to integer value and saves its result in D80.



2.16.9. SQR, SQRP (Square root)

SQR instruction computes the square root of source and saves the result to the Destination.

(It is supported only in XP and PLC-S CPU series)



S	Source
D	Destination

Instruction		Device address													No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
SQR SQRP	S	o	o	o	o	o	-	-	-	-	o	o	o	-	3	o	-	-
	D	o	-	o	o	o	-	-	-	-	o	o	o	-				

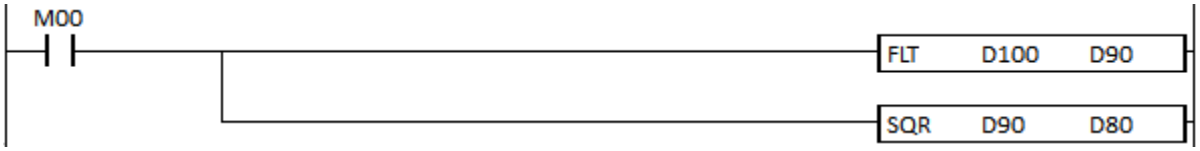
When enabled, the SQR instruction calculates the square root of source and saves the result in Destination.

- The source must be greater than zero

Example)

• If M00 is ON, the FLT instruction converts D100 to float value and saves its result to D90.

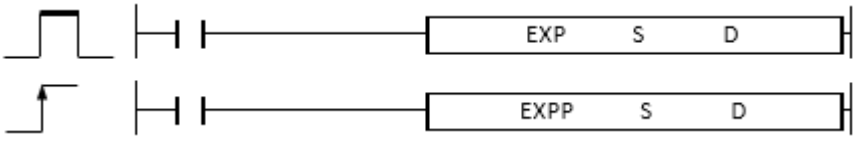
The SQR instruction calculates the square root of D90 and saves its result in D80.



2.16.10. EXP, EXPP (Natural Logarithms)

EXP instruction computes the natural log of the source and saves the result to the Destination.

(It is supported only in XP and PLC-S CPU series)



S	Source
D	Destination

Instruction		Device address													No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
EXP EXPP	S	o	o	o	o	o	-	-	-	-	o	o	o	-	3	o	-	-
	D	o	-	o	o	o	-	-	-	-	o	o	o	-				

When enabled, the EXP instruction calculates the natural log of source and saves the result in Destination.

- The natural log (e) is 2.71828.

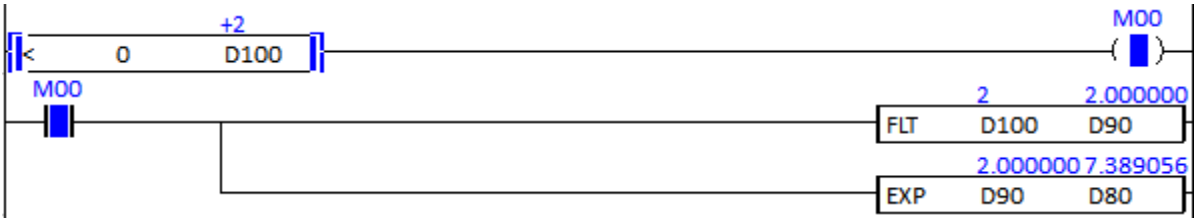
- The source must be greater than zero.

Example)

• If D100 is greater than 0, M00 will be turned ON.

If M00 is ON, the FLT instruction converts D100 to float value and saves its result to D90.

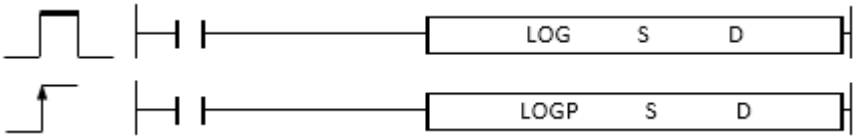
The EXP instruction calculates the natural log of D90 and saves its result in D80.



2.16.11. LOG, LOGP

LOG instruction computes the log base 10 of the source and saves the result to the Destination.

(It is supported only in XP and PLC-S CPU series)



S	Source
D	Destination

Instruction		Device address													No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
LOG LOGP	S	o	o	o	o	o	-	-	-	-	o	o	o	-	3	o	-	-
	D	o	-	o	o	o	-	-	-	-	o	o	o	-				

When enabled, the LOG instruction calculates the log base 10 of source and saves the result in Destination.

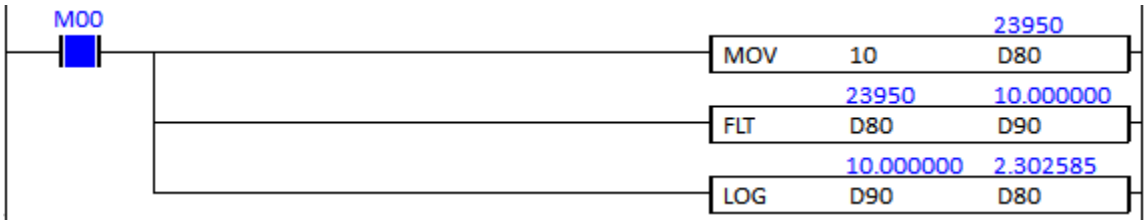
- The source must be greater than zero.

Example)

• If M00 is ON, MOV instruction copies 10 to D80.

The FLT instruction converts D80 to float value and saves its result to D90.

The LOG instruction calculates the log of D90 and saves its result in D80.

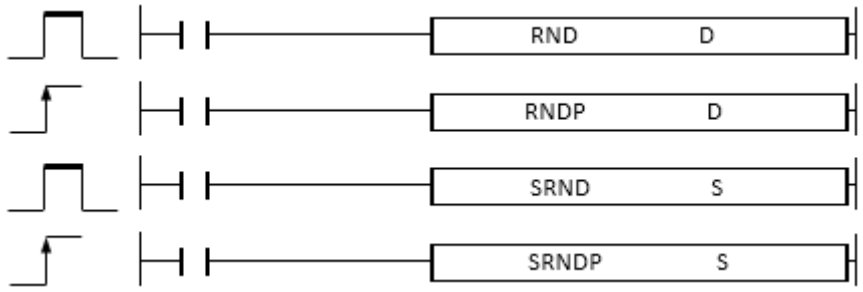


2.16.12. RND, RNDP, SRND, SRNDP

RND instruction produces a random sequence of numbers to Destination.

SRND instruction changes the seed of the Source (random sequence of numbers).

(It is supported only in XP and PLC-S CPU series)



S	Source
D	Destination

Instruction		Device address													No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
RND(P) SRND(P)	S	o	-	o	o	o	-	o	o	-	o	o	o	o	2	o	-	-
	D	o	-	o	o	o	-	o	o	-	o	o	o	o				

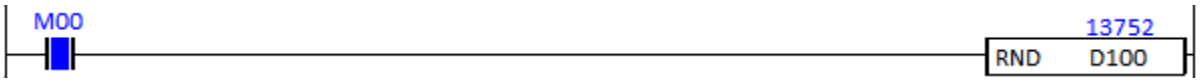
1) RND, RNDP

Format: RND D(Destination)

When enabled, RND instruction generates a random sequence of numbers to Destination.

Example)

• If M00 is ON, the RND instruction produces a random sequence of numbers to the D100.



2) SRND, SRNDP

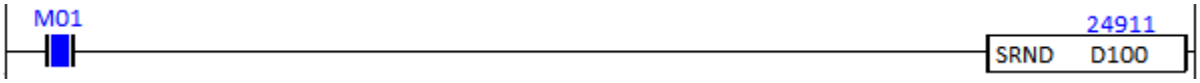
Format: SRND S(Source)

When enabled, SRND instruction sets its arguments as the seed for a new sequence of random integers to be returned by RND instruction.

SRND instruction will seed the random number generator to prevent random numbers from being the same every time the program is executed and to allow more pseudo randomness.

Example)

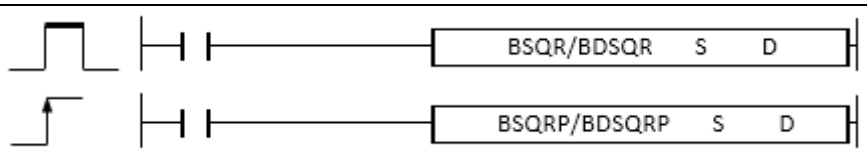
• If M01 is ON, the SRND instruction changes the seed of Source that is generated by RND instruction.



2.16.13. BSQR, BSQRP, BDSQR, BDSQRP

BSQR instruction computes the square root of source (BCD) and saves the result to the Destination.

(It is supported only in XP and PLC-S CPU series)



S	Source
D	Destination

Instruction		Device address													No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
BSQR(P) BDSQR(P)	S	o	o	o	o	o	-	-	-	-	o	o	o	o	3	o	-	-
	D	o	-	o	o	o	-	-	-	-	o	o	o	o				

1) BSQR. BSQRP

When enabled, the BSQR instruction calculates the square root of BCD source and saves the result in Destination.

- The source, D and D+1 are the BCD value. (0~9999)

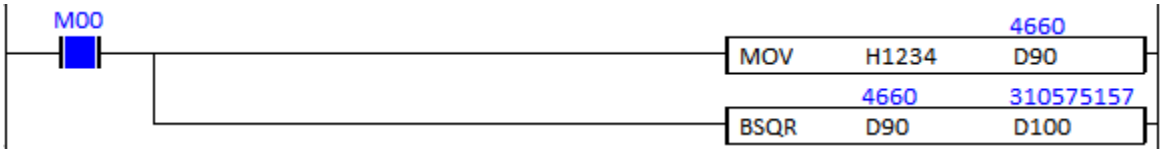
- The fraction (decimal point) is rounded off the numbers to five decimal places. Therefore, the 4 decimal place has tolerance of 1.

Example)

• If M00 is ON, the MOV instruction copies the BCD value H1234 to the D90.

The BSQR instruction calculates the square root of D90 and saves its result in D100.

The integer part is stored in the D100 and the decimal point is stored in the D101.



2) BDSQR. BDSQRP

When enabled, the BDSQR instruction calculates the square root of BCD source and saves the result in Destination.

- The S, S+1, D and D+1 are the BCD value. (0~99999999)

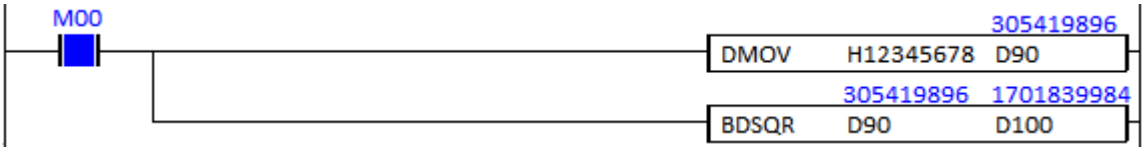
- The fraction (decimal point) is rounded off the numbers to five decimal places. Therefore, the 4 decimal place has tolerance of 1.

Example)

• If M00 is ON, the MOV instruction copies the BCD value H12345678 to the D90 and D91.

The BDSQR instruction calculates the square root of D90 and D91 and saves its result in D100 and D101.

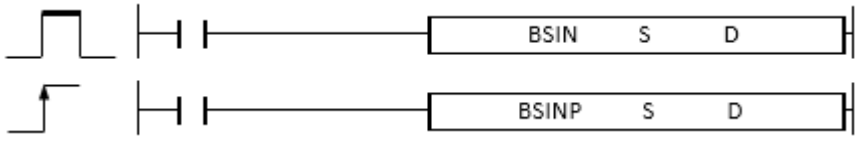
The integer part is stored in the D100 and the decimal point is stored in the D101.



2.16.14. BSIN, BSINP

BSIN instruction computes the sine of source (BCD) and saves the result in the Destination.

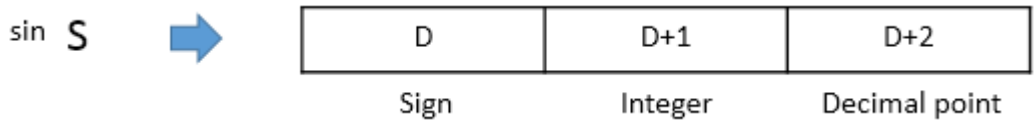
(It is supported only in XP and PLC-S CPU series)



S	BCD Source
D	Destination in BCD

Instruction		Device address													No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
BSIN BSINP	S	o	o	o	o	o	-	-	-	-	o	o	o	o	3	o	-	-
	D	o	-	o	o	o	-	-	-	-	o	o	o	o				

When enabled, the BSIN instruction computes the sine of BCD value and saves the result in Destination.



- The range of Source : 0~360(degree unit)

- D sets to 0 when its value is positive number(+). D sets to 1 when its value is negative number(-).

- D+1 and D+2 are BCD value and their range is from -1.0000 to 1.0000.

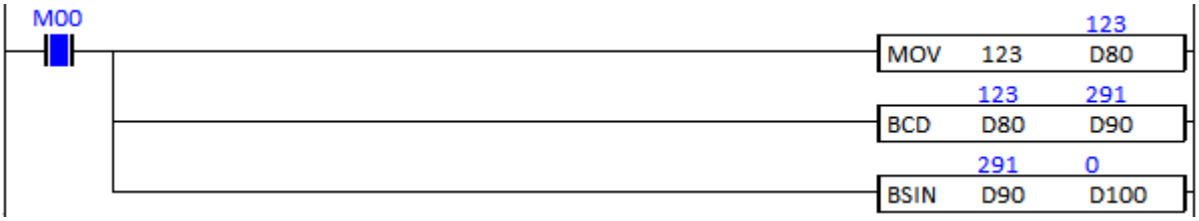
- The fraction (decimal point) is rounded off the numbers to five decimal places. Therefore, the 4 decimal place has tolerance of 1.

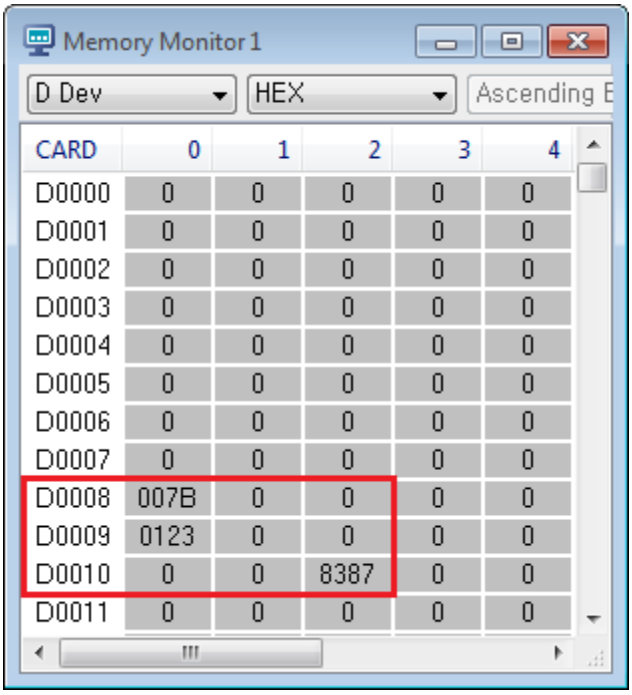
Example)

• If M00 is ON, the MOV instruction copies degree value 123 to the D80.

The BCD instruction converts 123 to BCD value and saves its value to D90.

The BSIN instruction computes the sine of BCD value (D90) and saves its result to D100 (sign), D101 (integer part) and D102 (decimal point).



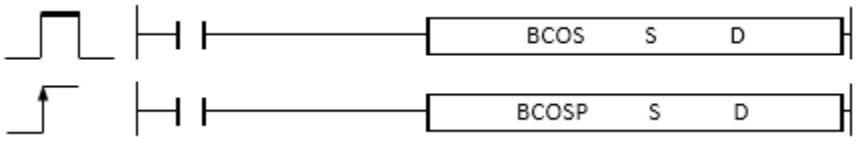


CARD	0	1	2	3	4
D0000	0	0	0	0	0
D0001	0	0	0	0	0
D0002	0	0	0	0	0
D0003	0	0	0	0	0
D0004	0	0	0	0	0
D0005	0	0	0	0	0
D0006	0	0	0	0	0
D0007	0	0	0	0	0
D0008	007B	0	0	0	0
D0009	0123	0	0	0	0
D0010	0	0	8387	0	0
D0011	0	0	0	0	0

2.16.15. BCOS, BCOSP

BCOS instruction computes the cosine of source (BCD) and saves the result in the Destination.

(It is supported only in XP and PLC-S CPU series)



S	BCD Source
D	Destination in BCD

Instruction		Device address													No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
BCOS BCOSP	S	o	o	o	o	o	-	-	-	-	o	o	o	o	3	o	-	-
	D	o	-	o	o	o	-	-	-	-	o	o	o	o				

When enabled, the BCOS instruction computes the cosine of the BCD value and saves the result in Destination.

- The range of Source : 0~360(degree unit)

- D sets to 0 when its value is positive number(+). D sets to 1 when its value is negative number(-).

- D+1 and D+2 are BCD value and their range is from -1.0000 to 1.0000.

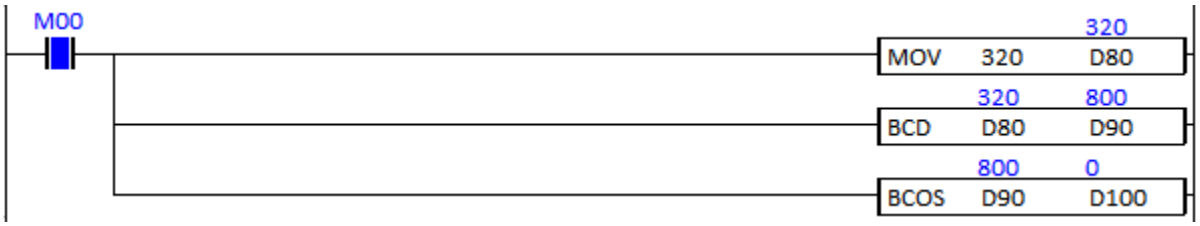
- The fraction (decimal point) is rounded off the numbers to five decimal places. Therefore, the 4 decimal place has tolerance of 1.

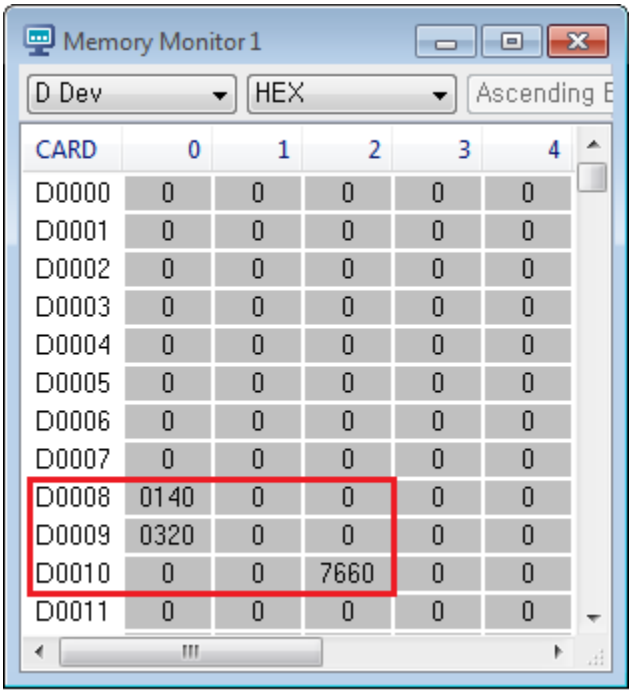
Example)

• If M00 is ON, the MOV instruction copies degree value 320 to the D80.

The BCD instruction converts 320 to BCD value and saves its value to D90.

The BCOS instruction computes the cosine of BCD value (D90) and saves its result to D100 (sign), D101 (integer part) and D102 (decimal point).

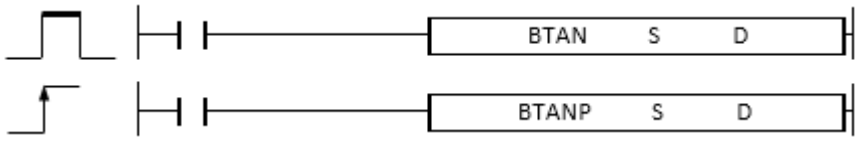




2.16.16. BTAN, BTANP

BTAN instruction computes the tangent of source (BCD) and saves the result in the Destination.

(It is supported only in XP and PLC-S CPU series)



S	BCD Source
D	Destination in BCD

Instruction		Device address													No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
BTAN BTANP	S	o	o	o	o	o	-	-	-	-	o	o	o	o	3	o	-	-
	D	o	-	o	o	o	-	-	-	-	o	o	o	o				

When enabled, the BTAN instruction computes the tangent of the BCD value and saves the result in Destination.

- The range of Source : 0~360(degree unit)

- D sets to 0 when its value is positive number(+). D sets to 1 when its value is negative number(-).

- D+1 and D+2 are BCD value and their range is from -52.2900 to 27.2900.

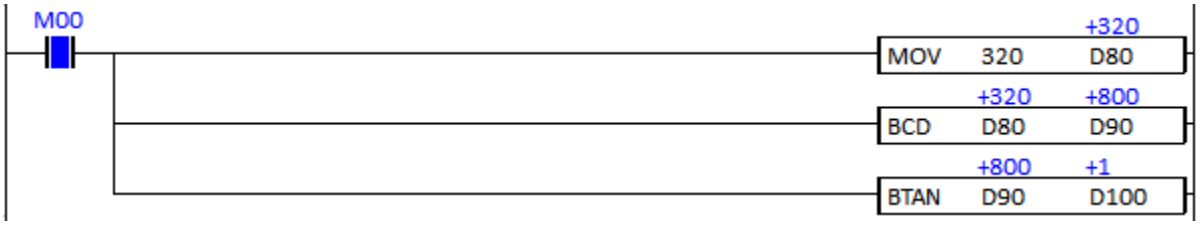
- The fraction (decimal point) is rounded off the numbers to five decimal places. Therefore, the 4 decimal place has tolerance of 1.

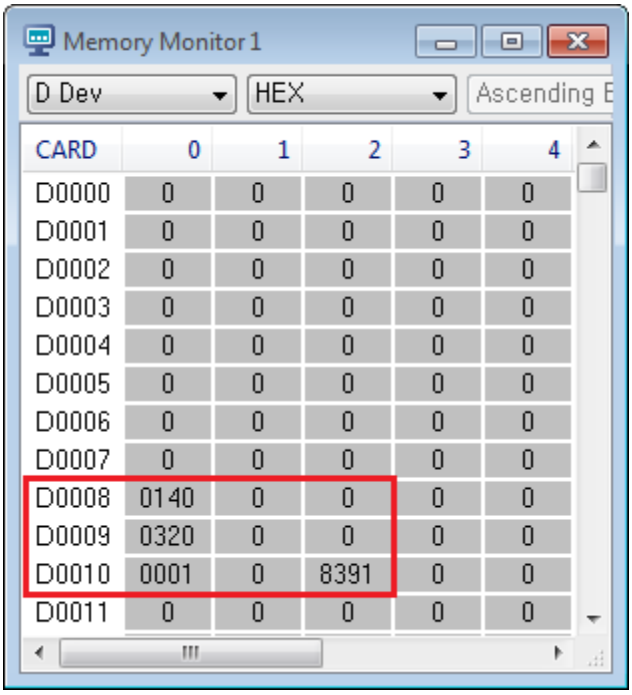
Example)

• If M00 is ON, the MOV instruction copies degree value 320 to the D80.

The BCD instruction converts 320 to BCD value and saves its value to D90.

The BTAN instruction computes the tangent of BCD value (D90) and saves its result to D100 (sign), D101 (integer part) and D102 (decimal point).

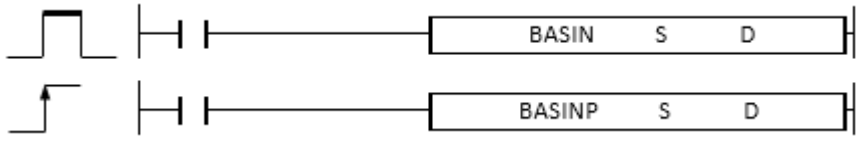




2.16.17. BASIN, BASINP

BASIN instruction computes the arc sine of source (BCD) and saves the result in the Destination.

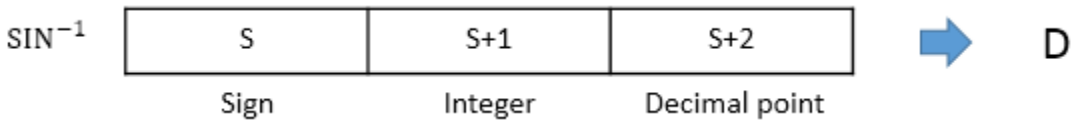
(It is supported only in XP and PLC-S CPU series)



S	BCD Source
D	Destination in BCD

Instruction		Device address													No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
BASIN BASINP	S	o	o	o	o	o	-	-	-	-	o	o	o	o	3	o	-	-
	D	o	-	o	o	o	-	-	-	-	o	o	o	o				

When enabled, the BASIN instruction computes the arc sine of the BCD value and saves the result in Destination.



- S sets to 0 when its value is positive number(+). D sets to 1 when its value is negative number(-).

- S+1 (Integer part) and S+2 (Decimal point) are the BCD value and their range is from 0 to 1.0000.

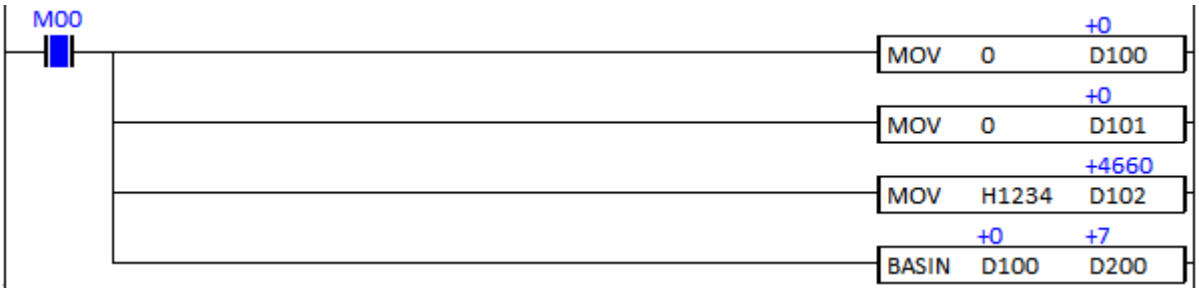
- D is the BCD value and its range is 0-90° and 270-360°.

- The fraction (decimal point) is rounded off the numbers to five decimal places. Therefore, the 4 decimal place has tolerance of 1.

Example)

• If M00 is ON, the MOV instruction copies H1234 to the D102.

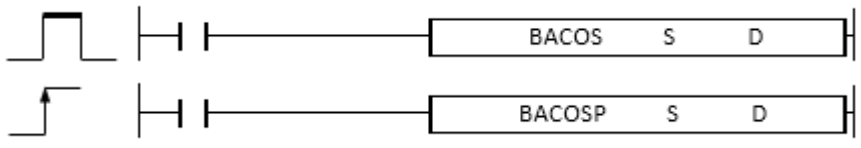
The BASIN calculates the arc sine of +0.1234 (D100, D101 and D102) and saves the result in the D200 (in BCD).



2.16.18. BACOS, BACOSP

BACOS instruction computes the arc cosine of source (BCD) and saves the result in the Destination.

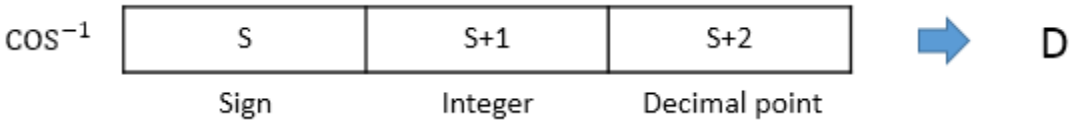
(It is supported only in XP and PLC-S CPU series)



S	BCD Source
D	Destination in BCD

Instruction		Device address													No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
BACOS BACOSP	S	o	o	o	o	o	-	-	-	-	o	o	o	o	3	o	-	-
	D	o	-	o	o	o	-	-	-	-	o	o	o	o				

When enabled, the BACOS instruction computes the arc cosine of the BCD value and saves the result in Destination.



- S sets to 0 when its value is positive number(+). D sets to 1 when its value is negative number(-).

- S+1 (Integer part) and S+2 (Decimal point) are the BCD value and their range is from 0 to 1.0000.

- D is the BCD value and its range is 0 - 180°.

- The fraction (decimal point) is rounded off the numbers to five decimal places. Therefore, the 4 decimal place has tolerance of 1.

Example)

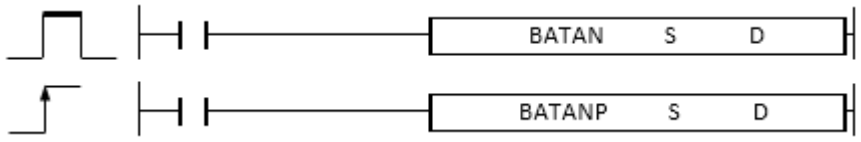
• If M00 is ON, the MOV instruction copies H1234 to the D102.

The BACOS calculates the arc cosine of +0.1234 (D100, D101 and D102) and saves the result in the D200 (in BCD).

2.16.19. BATAN, BATANP

BATAN instruction computes the arc tangent of source (BCD) and saves the result in the Destination.

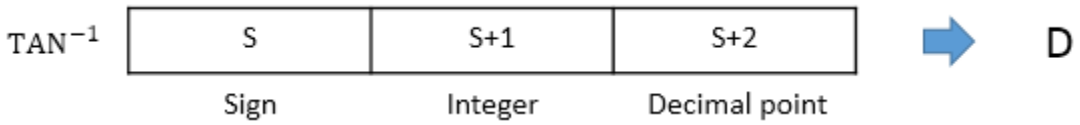
(It is supported only in XP and PLC-S CPU series)



S	BCD Source
D	Destination in BCD

Instruction		Device address												No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
BATAN BATANP	S	o	o	o	o	o	-	-	-	-	o	o	o	o	3	o	-	-
	D	o	-	o	o	o	-	-	-	-	o	o	o	o				

When enabled, the BATAN instruction computes the arc tangent of the BCD value and saves the result in Destination.



- S sets to 0 when its value is positive number(+). D sets to 1 when its value is negative number(-).

- S+1 (Integer part) and S+2 (Decimal point) are the BCD value and their range is from 0 to 9999.9999.

- D is the BCD value and its range is 0 - 90° and 270 - 360°

- The fraction (decimal point) is rounded off the numbers to five decimal places. Therefore, the 4 decimal place has tolerance of 1.

Example)

• If M00 is ON, the MOV instruction copies 1 to D100, H4321 to D101 and H1234 to D102.

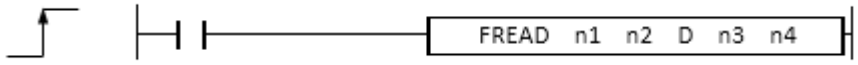
The BATAN calculates the arc tangent of -4321.1234 (D100, D101 and D102) and saves the result in the D200 (in BCD).

2.17. Special Function Instruction

2.17.1. FREAD

FREAD instruction copies a specified system memory of the special module and saves the result in the Destination.

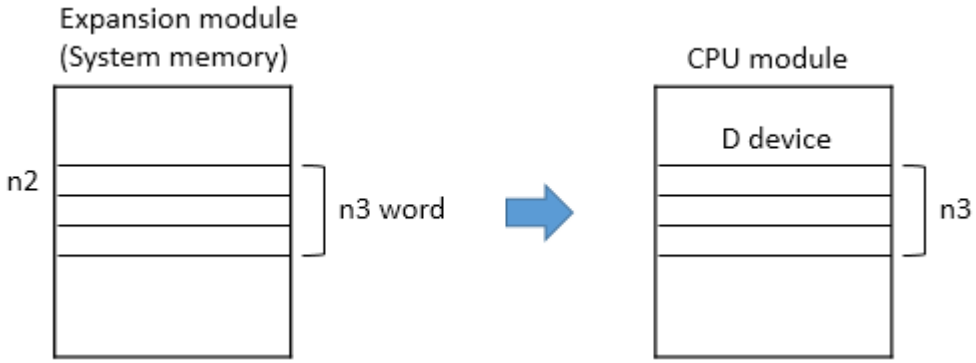
(It is supported only in CM1-PS02A special module)



n1	Slot number of special module
n2	Starting system memory to be read
D	Destination
n3	Number of system memory to be read
n4	Result of operation

Instruction		Device address													No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
FREAD	n1	o	o	o	o	o	o	o	o	-	o	o	o	o	6	o	-	-
	n2	o	o	o	o	o	o	o	o	-	o	o	o	o				
	D	o	-	o	o	o	-	-	-	-	o	o	o	-				
	n3	o	o	o	o	o	o	o	o	-	o	o	o	o				
	n4	o	-	o	o	o	-	-	-	-	o	o	o	-				

When enabled, the FREAD copies specified number(n3) of system memory(n2) from the special module(n1: slot number of expansion module) to the Destination (D device address) and saves the result of instruction operation in the n4.



- n1 : HEX that shows Base and Slot number of expansion module.

Ex) H + (Base number) + (Slot number)

Base number	Slot number	n1
Local Base	Slot number 5.	H0005
Expansion Base number 1.	Slot number 3.	H0103
Expansion Base number 10.	Slot number 7.	H0A07
Expansion Base number 14.	Slot number 12.	H0E0C
Expansion Base number 16.	Slot number 10.	H100A

* Please refer to below the Base and Slot number system for CM1 series.

- n2 : Offset that is starting address of system memory to be read.

(Please refer to System memory of positioning module)

- n3 : Size (number of system memory to be read)

The range of n3 is from 0 to 32,767.

If n3 is 0, instruction will not work and if n3 is out of range, it occurs an error.

- n4 : the result of instruction operation is stored in n4.

Bit No.	Description
0	If instruction is running (1), If instruction is completed (0)
1	Error
2 – 7	Not used
8 - 15	Error Code 00H : No error 01H : Maximum 32 data can be read and write at the same time for a scan 02H : Data size is out of range 03H : Wrong Offset

* The error flag is turned ON if :

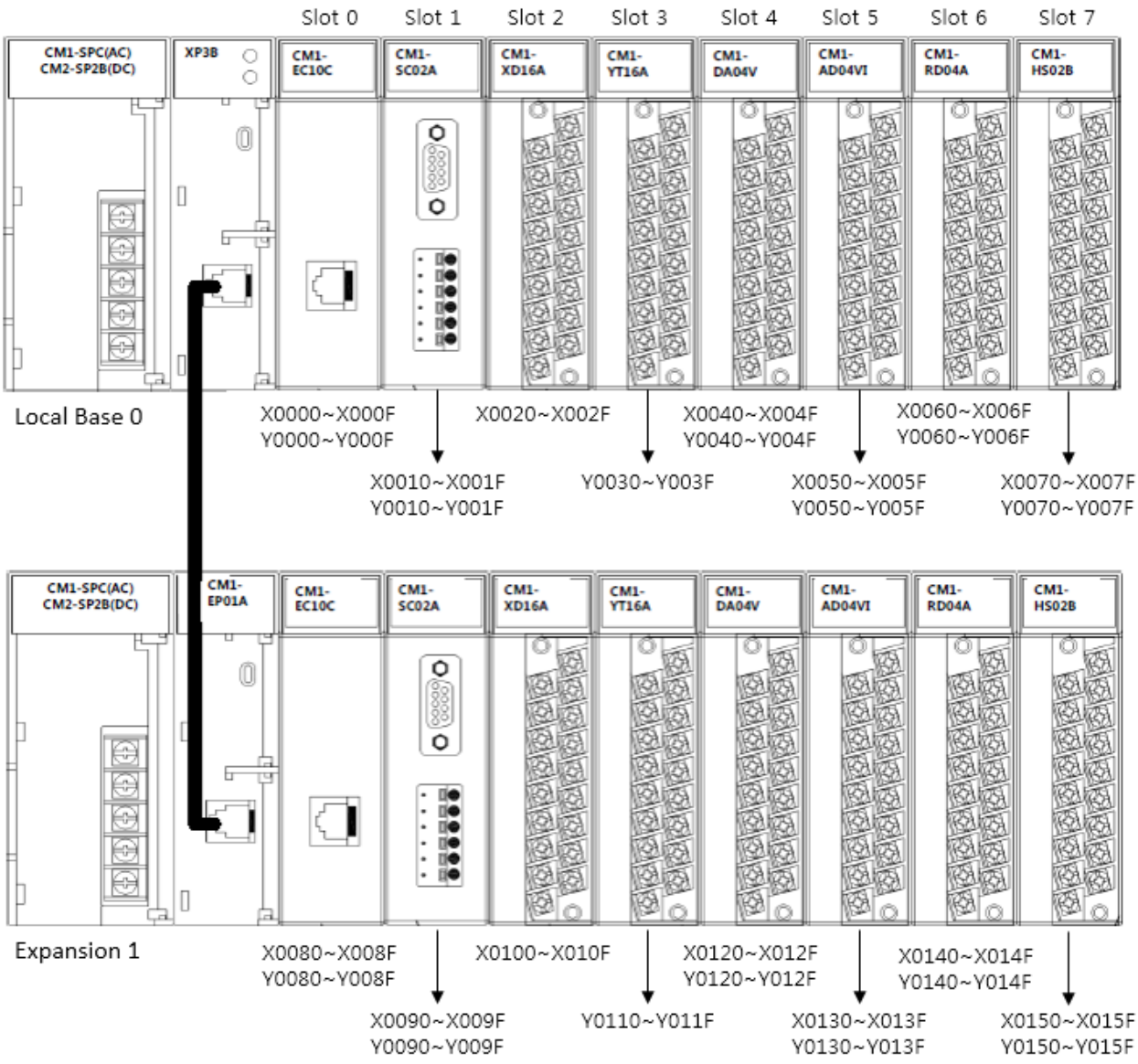
a. CPU cannot access expansion module.

b. n1 is not special module (CM1-PS02A)

c. n3 exceeds Destination.

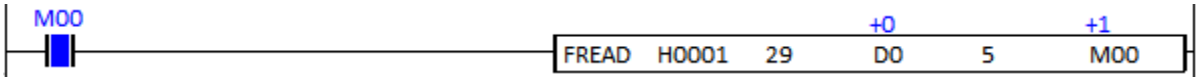
d. n3 is out of range (0 – 32,767)

• Base and Slot number for CM1



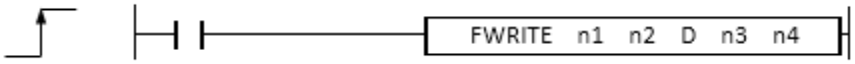
Example)

• If M00 is ON, the FREAD instruction copies 5words of system memory 29 from positioning module that is installed at slot no. 1 in the local base and saves its value to D0~D4 in sequence. The result of operation is stored in M00.



2.17.2. FWRITE

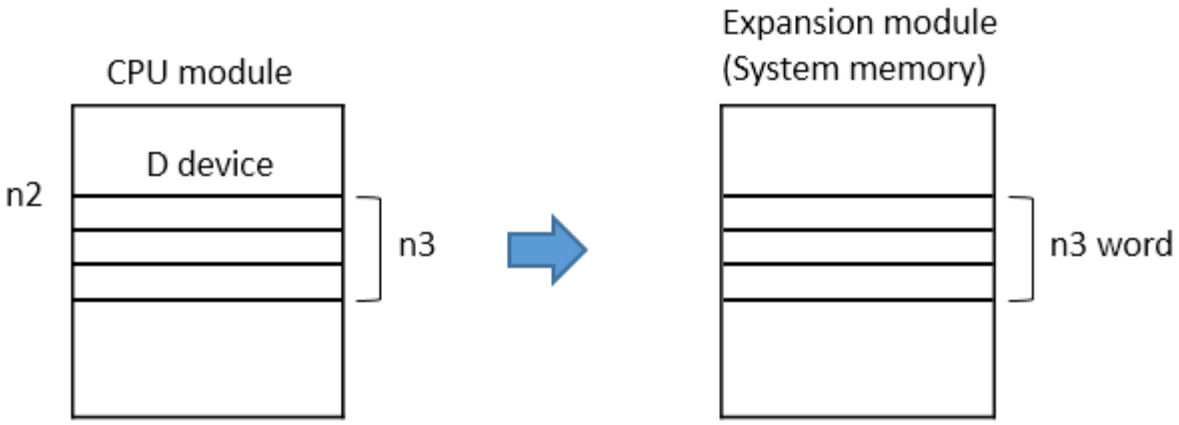
FWRITE instruction copies a specified device data of CPU module to the system memory of special module. (It is supported only in CM1-PS02A special module)



n1	Slot number of special module
n2	Starting system memory to be written
D	Starting device address to copy
n3	Number of system memory to copy
n4	Result of operation

Instruction		Device address													No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
FWRITE	n1	o	o	o	o	o	o	o	o	-	o	o	o	o	6	o	-	-
	n2	o	o	o	o	o	o	o	o	-	o	o	o	o				
	D	o	-	o	o	o	-	-	-	-	o	o	o	-				
	n3	o	o	o	o	o	o	o	o	-	o	o	o	o				
	n4	o	-	o	o	o	-	-	-	-	o	o	o	-				

When enabled, the FWRITE copies specified device address (n3 of D) from the CPU module to the system memory (n2) of special module that is installed in n1 and saves the result of instruction operation in the n4.



- n1 : HEX that shows Base and Slot number of special module.

Ex) H + (Base number) + (Slot number)

Base number	Slot number	n1
Local Base	Slot number 5.	H0005
Expansion Base number 1.	Slot number 3.	H0103
Expansion Base number 10.	Slot number 7.	H0A07
Expansion Base number 14.	Slot number 12.	H0E0C
Expansion Base number 16.	Slot number 10.	H100A

* Please refer to below the Base and Slot number system for CM1 series.

- n2 : Offset that is starting address of system memory to be written.

(Please refer to System memory of positioning module)

- n3 : Size (number of system memory to write)

The range of n3 is from 0 to 32,767.

If n3 is 0, instruction will not work and if n3 is out of range, it occurs an error.

- n4 : the result of instruction operation is stored in n4.

Bit No.	Description
0	If instruction is running (1), If instruction is completed (0)
1	Error
2 – 7	Not used
8 - 15	Error Code 00H : No error 01H : Maximum 32 data can be read and write at the same time for a scan 02H : Data size is out of range 03H : Wrong Offset

* The error flag is turned ON if :

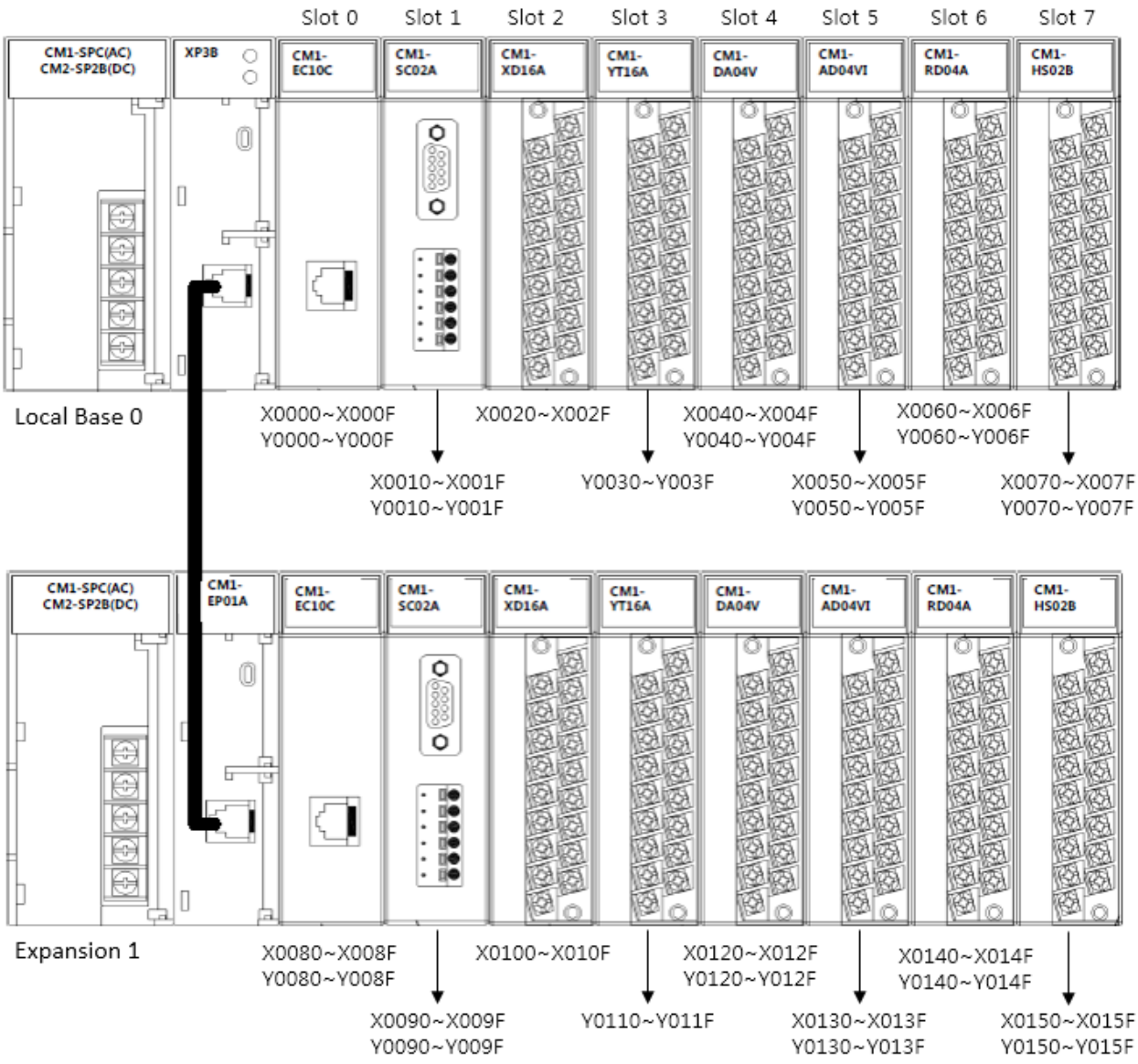
a. CPU cannot access expansion module.

b. n1 is not special module (CM1-PS02A)

c. n3 exceeds Destination.

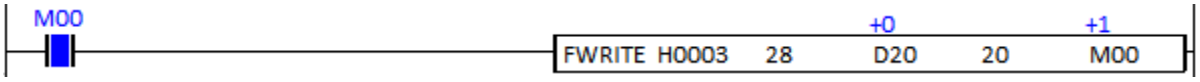
d. n3 is out of range (0 – 32,767)

• Base and Slot number for CM1



Example)

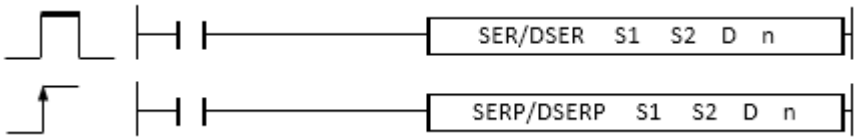
• If M00 is ON, the FWRITE instruction copies 20words of D20 (from D20 to D39) and saves to the system memory 28 (from 28 to 47) of positioning module that is installed at slot no. 3 in the local base. The result of operation is stored in M00.



2.18. Data Search Instruction

2.18.1. SER, SERP, DSER, DSERP

SER instruction searches the source value in the specific address and saves the result in the Destination.



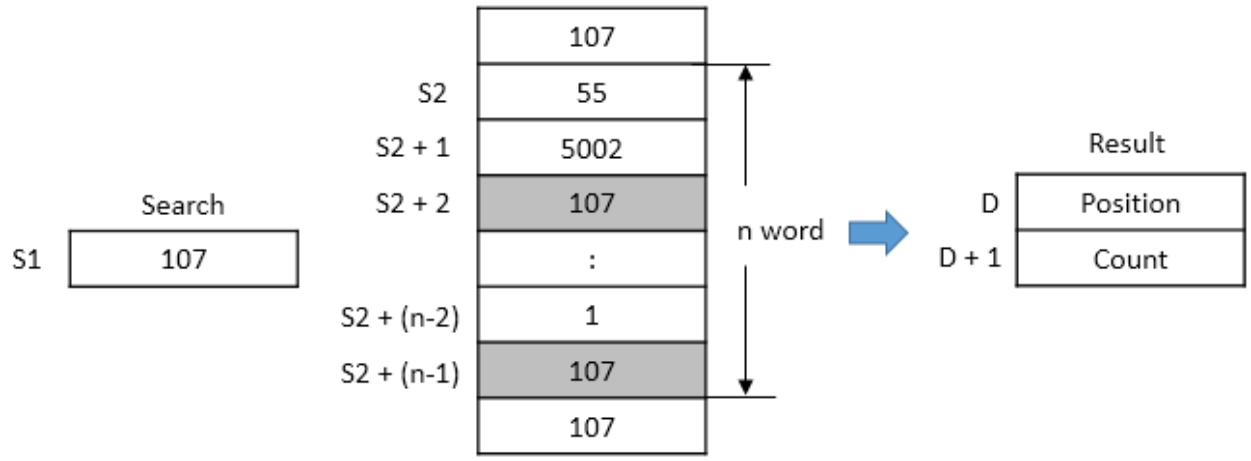
S1	Source value that you want to find out
S2	Starting address for searching
D	Destination
n	Range of search

Instruction		Device address													No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
SER SERO DSER DSERP	S1	o	o	o	o	o	o	o	o	-	o	o	o	o	5	o	-	-
	S2	o	o	o	o	o	o	-	-	-	o	o	o	-				
	D	o	-	o	o	o	-	-	-	-	o	o	o	-				
	n	o	o	o	o	o	o	o	o	-	o	o	o	o				

1) SER, SERP

Format : SER S1(Source value that you want to find out) S2(Starting address for searching) D(Destination) n(range of search)

When enabled, the SER instruction search S1 value in the (n)number of S2 and save the result in D and D+1.



- D : It saves position number. The first 107(S1) is stored in the third of S2 so that the result is 3.

- D+1 : It saves total number of 107(S1) in n. There are 2 107 in n so that the result is 2.

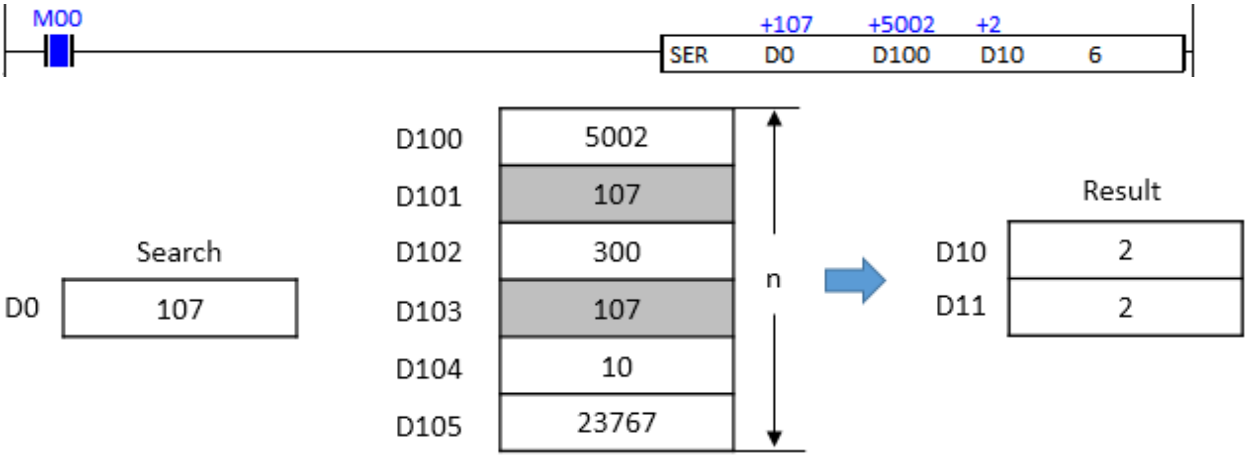
• Error

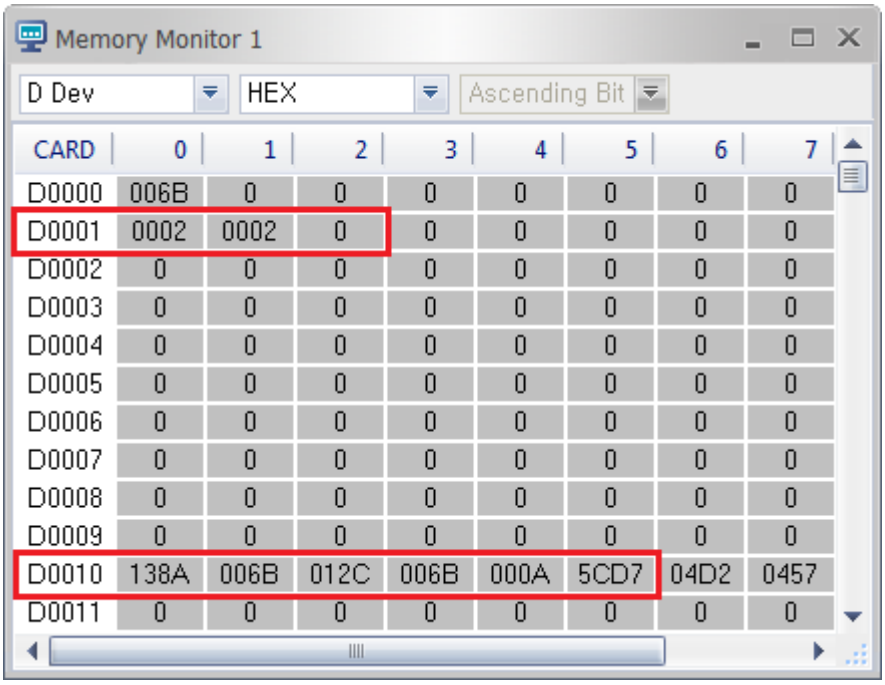
- If n is 0 or negative(-) value, the D holds the value and Error flag(F0110) will be set.

- If the search range(n) is out of device address range, the Error flag(F0110) will be set and error code (F0050) will be H0403.

Example)

• If M00 is ON, the SER instruction searches 107(D0) from D100 to D105 and saves the result in D10 and D11. Since the first 107 is stored in D101(second of D100), 2 is saved in the D10. There are two 107 in the D100 ~ D105 so 2 is saved in the D11.



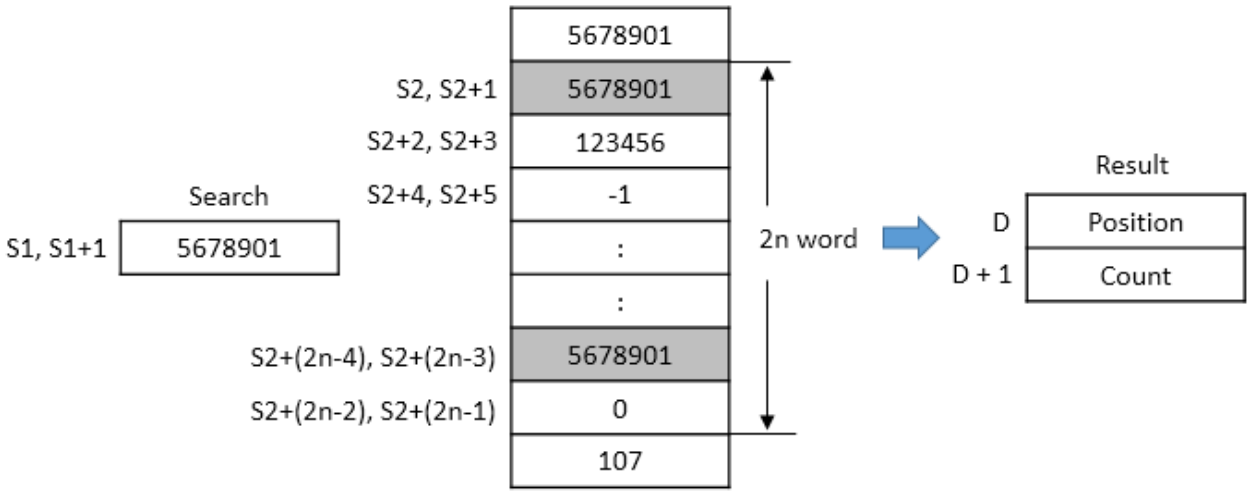


CARD	0	1	2	3	4	5	6	7
D0000	006B	0	0	0	0	0	0	0
D0001	0002	0002	0	0	0	0	0	0
D0002	0	0	0	0	0	0	0	0
D0003	0	0	0	0	0	0	0	0
D0004	0	0	0	0	0	0	0	0
D0005	0	0	0	0	0	0	0	0
D0006	0	0	0	0	0	0	0	0
D0007	0	0	0	0	0	0	0	0
D0008	0	0	0	0	0	0	0	0
D0009	0	0	0	0	0	0	0	0
D0010	138A	006B	012C	006B	000A	5CD7	04D2	0457
D0011	0	0	0	0	0	0	0	0

2) DSER, DSERP

Format : DSER S1(Double word value that you want to find out) S2(Starting address for searching) D(Destination) n(range of search)

When enabled, the DSER instruction search S1 (DWORD value) in the (n)number of S2 and save the result in D and D+1.



- D : It saves position number. The first 5678901(S2, S1+1) is stored in the first of S2 so that the result is 1.

- D+1 : It saves total number of 5678901 in 2n. There are 2 5678901 in 2n so that the result is 2.

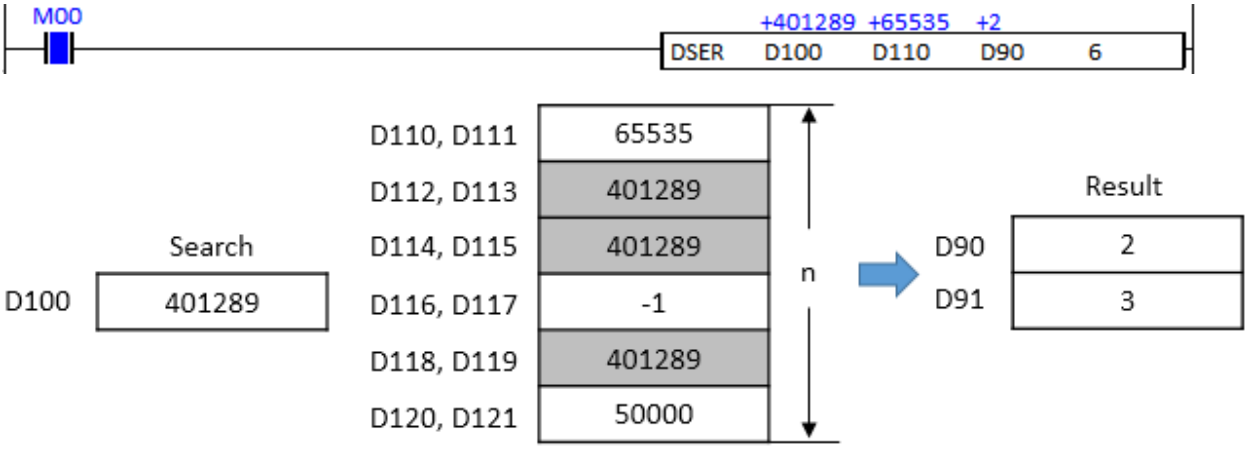
• Error

- If n is 0 or negative(-) value, the D holds the value and Error flag(F0110) will be set.

- If the search range(n) is out of device address range, the Error flag(F0110) will be set and error code (F0050) will be H0403.

Example)

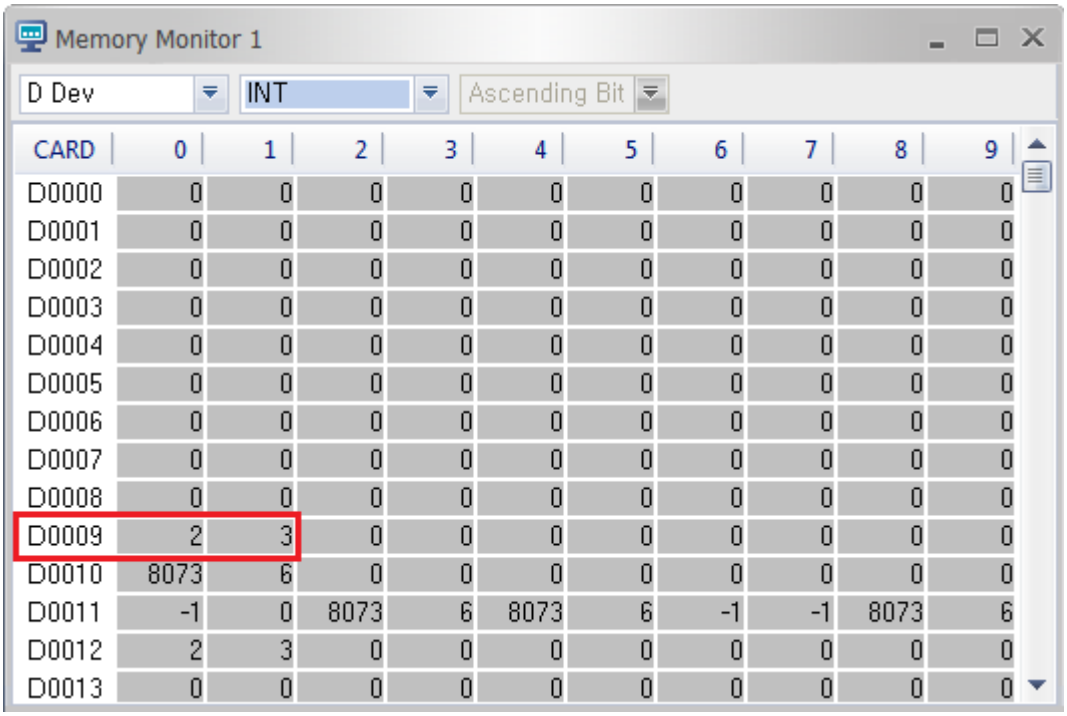
• If M00 is ON, the DSER instruction searches 401289(D100) from D110 to D121 and saves the result in D90 and D91. Since the first 401289 is stored in D112 and D113(second of D110), 2 is saved in the D90. There are three 401289 in the D110 ~ D121 so 3 is saved in the D91.



Memory Monitor 1

D Dev | DWORD | Ascending Bit

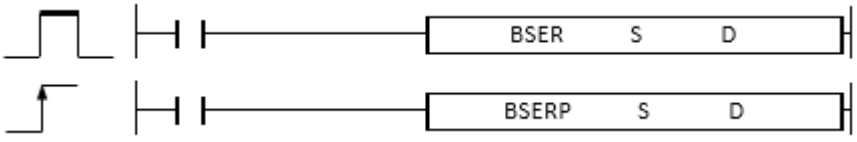
CARD	0	1	2	3	4	5	6	7	8	9
D0000	0	0	0	0	0	0	0	0	0	0
D0001	0	0	0	0	0	0	0	0	0	0
D0002	0	0	0	0	0	0	0	0	0	0
D0003	0	0	0	0	0	0	0	0	0	0
D0004	0	0	0	0	0	0	0	0	0	0
D0005	0	0	0	0	0	0	0	0	0	0
D0006	0	0	0	0	0	0	0	0	0	0
D0007	0	0	0	0	0	0	0	0	0	0
D0008	0	0	0	0	0	0	0	0	0	131072
D0009	196610	3	0	0	0	0	0	0	0	529072128
D0010	401289	6	0	0	0	0	0	0	0	-65536
D0011	65535	529072128	401289	529072134	401289	-65530	-1	529137663	401289	131078
D0012	196610	3	0	0	0	0	0	0	0	0
D0013	0	0	0	0	0	0	0	0	0	0



CARD	0	1	2	3	4	5	6	7	8	9
D0000	0	0	0	0	0	0	0	0	0	0
D0001	0	0	0	0	0	0	0	0	0	0
D0002	0	0	0	0	0	0	0	0	0	0
D0003	0	0	0	0	0	0	0	0	0	0
D0004	0	0	0	0	0	0	0	0	0	0
D0005	0	0	0	0	0	0	0	0	0	0
D0006	0	0	0	0	0	0	0	0	0	0
D0007	0	0	0	0	0	0	0	0	0	0
D0008	0	0	0	0	0	0	0	0	0	0
D0009	2	3	0	0	0	0	0	0	0	0
D0010	8073	6	0	0	0	0	0	0	0	0
D0011	-1	0	8073	6	8073	6	-1	-1	8073	6
D0012	2	3	0	0	0	0	0	0	0	0
D0013	0	0	0	0	0	0	0	0	0	0

2.18.2. BSER, BSERP

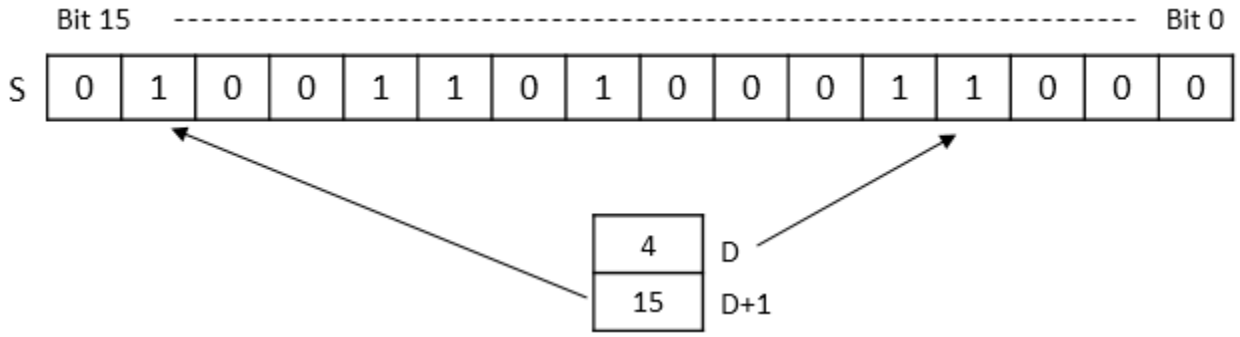
BSER instruction searches the value 1 in the source and saves the lowest bit to D and the highest bit to D+1.



S	Source
D	Destination

Instruction		Device address													No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
BSER BSERP	S	o	o	o	o	o	o	o	o	-	o	o	o	o	3	o	-	-
	D	o	-	o	o	o	-	o	o	-	o	o	o	-				

When enabled, the BSER instruction searches the value 1 and saves the position number of the lowest bit to the D and the highest bit to the D+1.



- D : The lowest bit position(4) is stored in D.

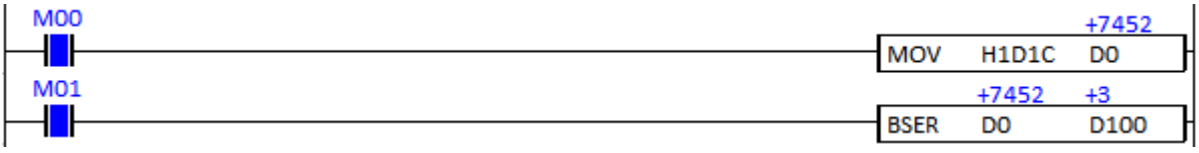
- D+1 : The highest bit position(15) is stored in D+1.

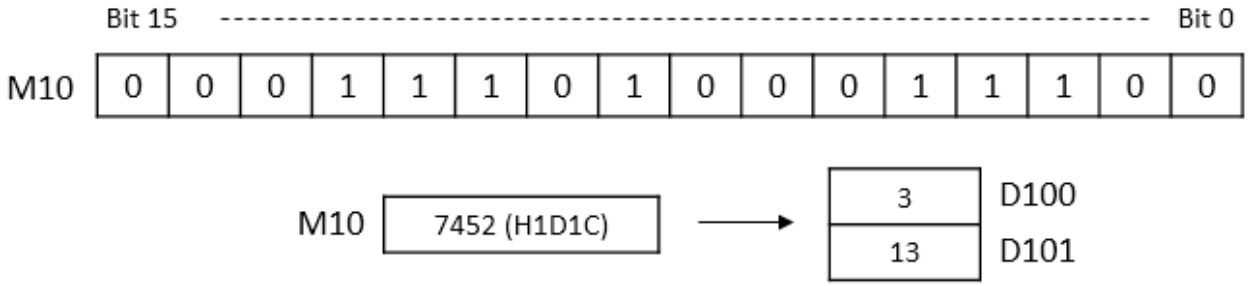
- If there is no value 1 in the source, D and D+1 will have 0 value.

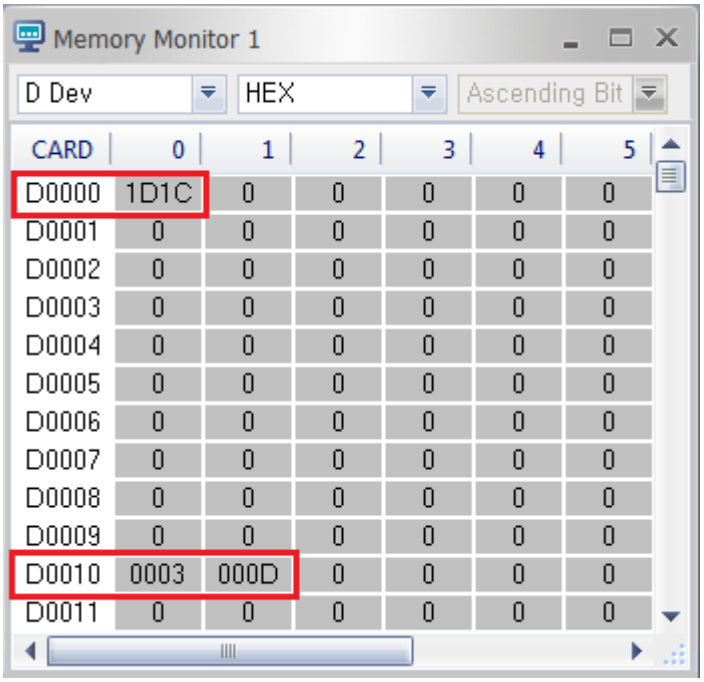
- If There is only 1 bit which has value 1, D and D+1 will have the same value.

Example)

• If M01 is ON, the BSER instruction searches value 1 in the source (H1D1C). The lowest bit position 3 is stored in D100 and the highest bit position 13 is stored in D101.







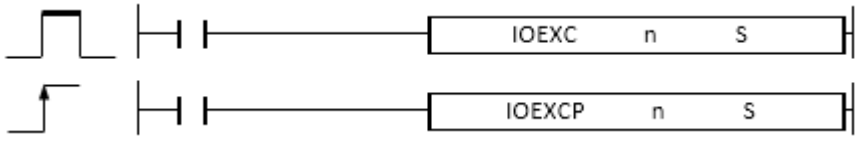
CARD	0	1	2	3	4	5
D0000	1D1C	0	0	0	0	0
D0001	0	0	0	0	0	0
D0002	0	0	0	0	0	0
D0003	0	0	0	0	0	0
D0004	0	0	0	0	0	0
D0005	0	0	0	0	0	0
D0006	0	0	0	0	0	0
D0007	0	0	0	0	0	0
D0008	0	0	0	0	0	0
D0009	0	0	0	0	0	0
D0010	0003	000D	0	0	0	0
D0011	0	0	0	0	0	0

2.19. Module Replacement Instruction

2.19.1. IOEXC, IOEXCP

IOEXC instruction is used to replace I/O module when CPU is running (Online status).

(It is supported only in XP B type CPU series)



n	Slot number of module to be replaced
S	Set 1 or Reset 0

Instruction		Device address													No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
IOEXC IOEXCP	n	o	o	o	o	o	o	o	o	-	o	o	o	o	3	o	-	-
	S	o	o	o	o	o	o	o	o	-	o	o	o	o				

Notice

- This instruction works with only IO modules. In case of special and communication module replacement, you need to reset and download parameters or special programs to the modules manually.

- In order to download special programs or parameters to special and communication module automatically, click Online and choose Enable/Disable Module in CICON.

- In case of error, take module off and put it in Base again.

- If the IOEXCP is disabled before module replacement, the system should be dead.

Format : IOEXC n S

- n : HEX that shows Base and Slot number of expansion module.

Ex) H + (Base number) + (Slot number)

Base number	Slot number	n1
Local Base	Slot number 5.	H0005
Expansion Base number 1.	Slot number 3.	H0103
Expansion Base number 10.	Slot number 7.	H0A07
Expansion Base number 14.	Slot number 12.	H0E0C
Expansion Base number 16.	Slot number 10.	H100A

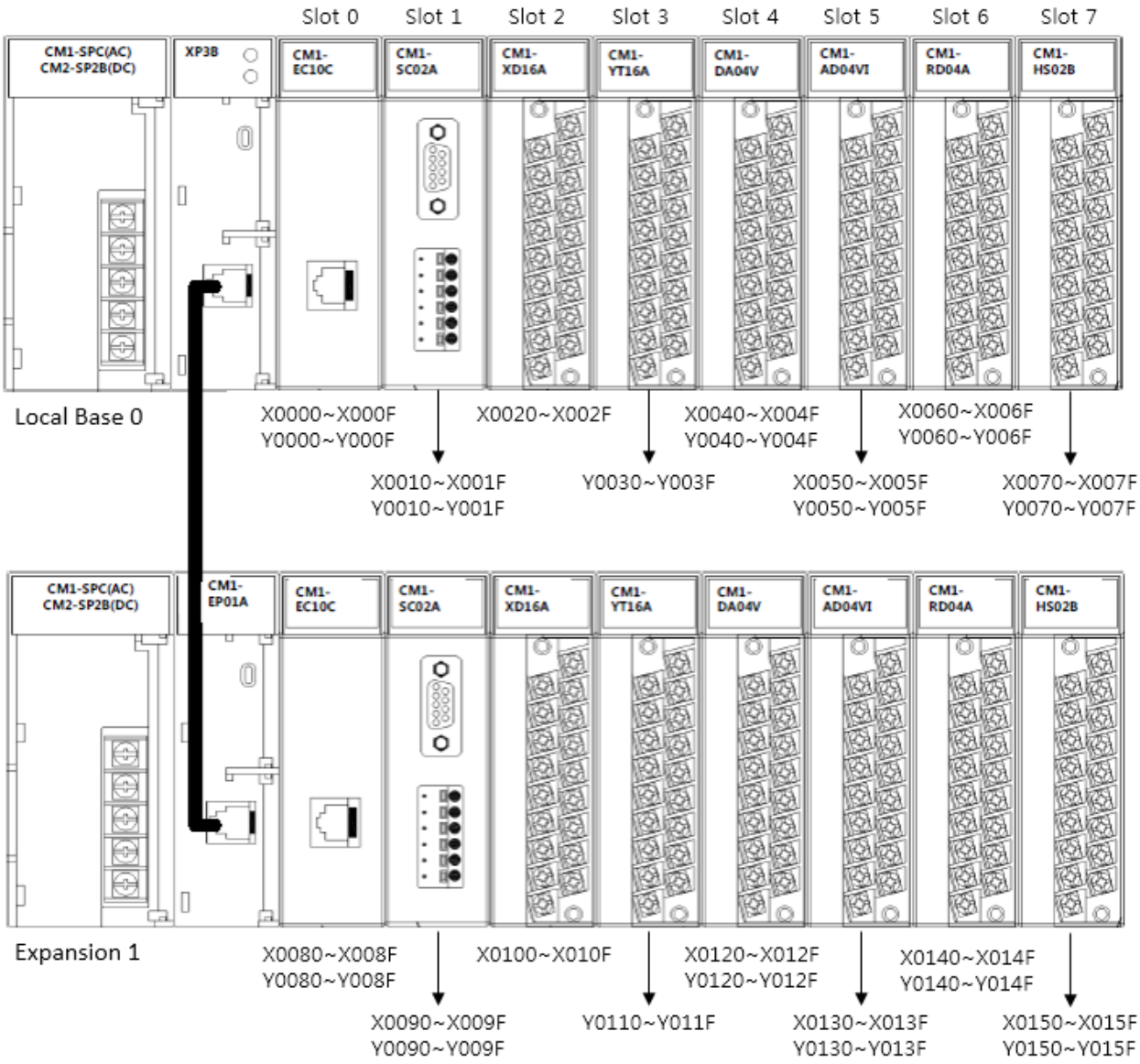
* Please refer to below the Base and Slot number system for CM1 series.

- S : Set (1) or Reset (0)

Set 1 : Start I/O module replacement

Reset 0 : Stop I/O module replacement

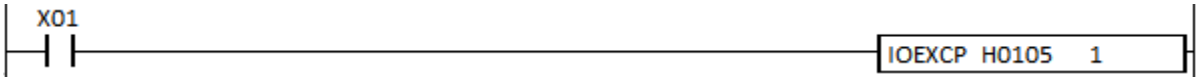
• Base and Slot number for CM1



Example) Module replacement

• The module is installed at the Base number 1 and slot number 5.

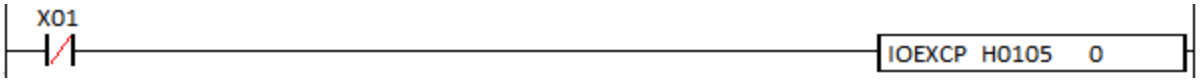
Step 1) If X01 is ON, the IOEXCP instruction sets 1 to have module replaced.



Step 2) Take module off and replace module

Step 3) If there is no error, move Step 4. If there is an error, go back to Step 2.

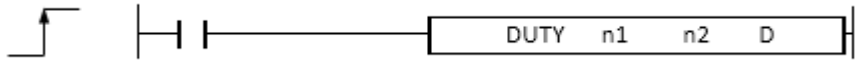
Step 4) Turn Off X01 and complete replacement.



2.20. Other Instructions

2.20.1. DUTY

DUTY instruction is used to replace I/O module when CPU is running (Online status).



n1	Scan for On status
n2	Scan for OFF status
D	Clock device address

Instruction		Device address													No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
DUTY	n1	o	o	o	o	o	o	o	o	-	o	o	o	o	4	o	-	-
	n2	o	o	o	o	o	o	o	o	-	o	o	o	o				
	D	-	-	-	-	-	o	-	-	-	-	-	-	-				

When enabled, the DUTY instruction turns D (clock device) On in n1's scan time and Off in n2's scan time.

The range of D is from F0100 to F0107.

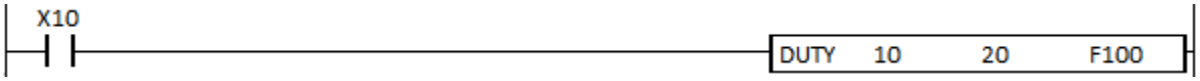
Error)

- If the D is out of range, F110 (Error flag) is set.

- If n1 or n1 is 0 or minus (-) value, F110 (Error flag) is set.

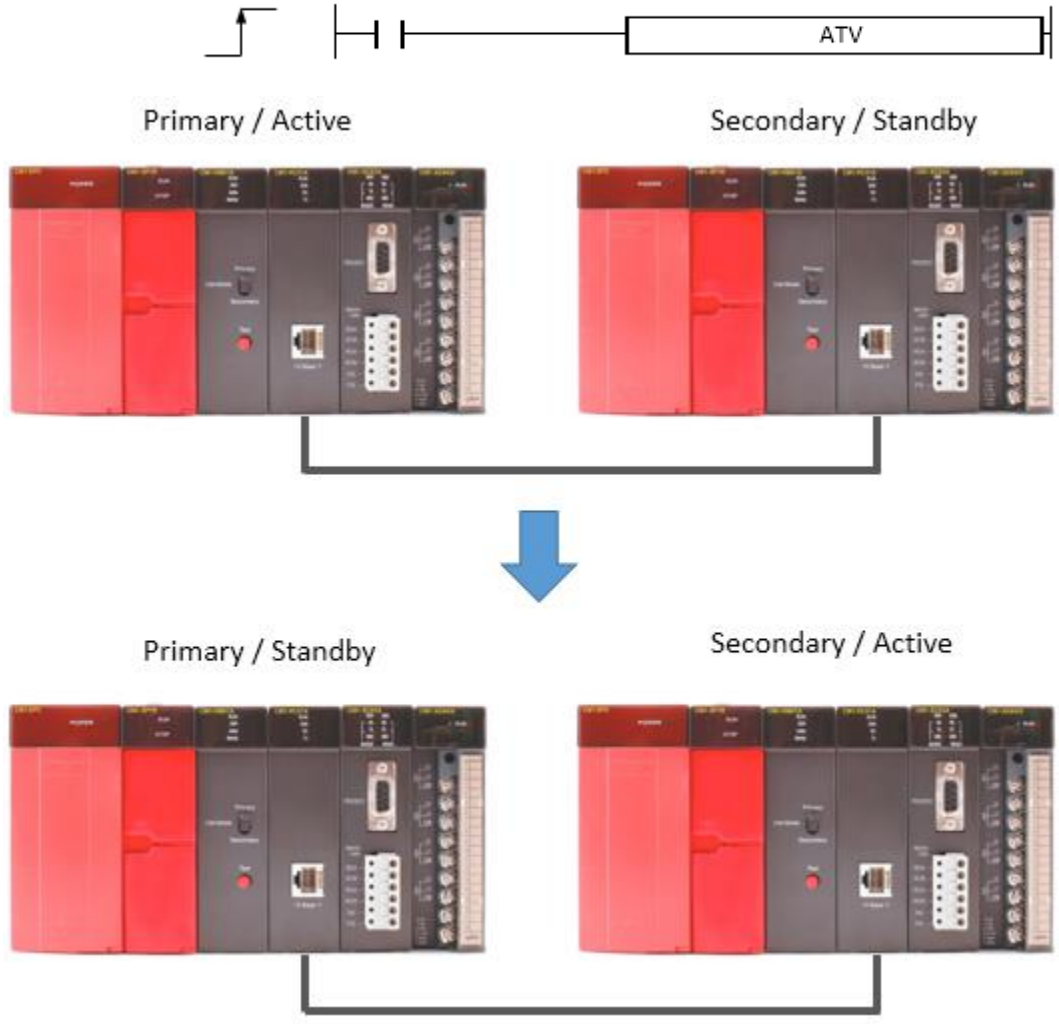
Example)

• If X10 is ON, the DUTY instruction turns ON F100 in 10sane and turns Off F100 in 20scan.



2.20.2. ATV (Redundancy Switch Instruction)

ATV instruction switches over Active mode and Standby mode between Primary and Secondary CPU in the redundancy.

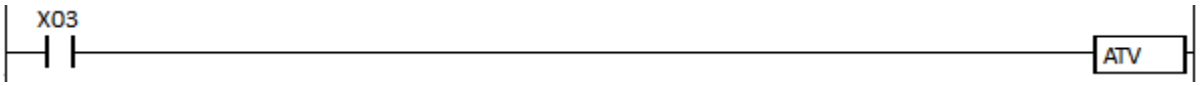


Error)

- If the Standby CPU is not ok, Error Flag (F0110) will be set.

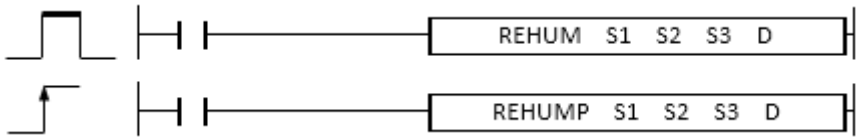
Example)

• If X03 is ON, the ATV instruction switches over CPU mode (Active to Standby or Standby to Active) in the redundancy.



2.20.3. REHUM, REHUMP

REHUM instruction operates the relative humidity and the dew point data with reference to S1 (Air Temperature), S2 (Wet Bulb Temperature), S3 (Pressure) and store the result in D

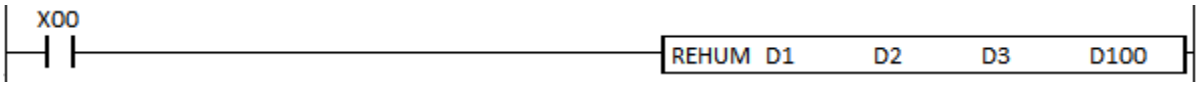


S1	The air temperature
S2	The wet bulb temperature
S3	The pressure
D	Destination

Instruction		Device address													No. of Steps	Flag		
		M	X	Y	K	L	F	T	C	S	Z	D	@D	Int.		Error	Zero	Carry
REHUM REHUMP	S1	o	o	o	o	o	o	-	-	-	o	o	o	o	5	o	-	-
	S2	o	o	o	o	o	o	-	-	-	o	o	o	o				
	S3	o	o	o	o	o	o	-	-	-	o	o	o	o				
	D	o	-	o	o	o	-	-	-	-	o	o	-	-				

Example)

• If X00 is ON, the REHUM instruction calculates the relative humidity and the dew point data with reference to D1 (Air Temperature), D2 (Wet Bulb Temperature), D3 (Pressure) and then save the result in D100.



CIMON CO.,LTD.

H.Q Address : Zip Code 13503 / 5F KDT BLDG, #48, Beolmal-ro, Bundang-gu, Seongnam-si, GyeongGi-do, Korea
Tel: +82-31-778-3071

USA : Zip Code 90010 / 3699 Wilshire Blvd, Suite 1250 Los Angeles CA 90010
Tel : 213-384-8703
Website : www.cimon.com