

User's Manual

LG Programmable Logic Controller

MASTER-K

Instructions & Programming

LG Industrial Systems

Chapter 1 Introduction

1.1	Introductions.....	1-1
1.2	Features	1-1

1 Introductions

1.1 Introductions

The chapter 1 'Introduction' will provide brief information about the features, functions, and operation of MASTER-K series PLC.

1.2 Features

1) Features of MASTER-K series PLC are as following ;

- a) various programming device for easy programming.
- b) editing in RUN mode is available
- c) supports various open network with international standard communication protocol.
- d) realization of high processing speed with the dedicated arithmetic processor.
- e) various special function modules for PLC application fields.

2) Features of MASTER-K 200S/300S/1000S are as following;

a) The fast processing speed :

Realize the lightning processing speed at 0.2 μ s with dedicate arithmetic processor.

b) Enhanced self-diagnostic functions :

With detailed self-diagnostic error codes, the cause of error can be identified easily.

c) Debugging operation

The MASTER-K 200S/300S/1000S provides various debugging methods as following and it enable on-line debugging.

- execution with command by command
- execute with the break point designation
- execution according to the status of device
- execution with designated numbers of scan

d) Execution of various program types

The MASTER-K 200S/300S/1000S series provides time-driven interrupt, process-driven interrupt, and subroutine programs as well as normal scan program.

e) Supports the sampling trace and triggering functions

Remark The compatibility among MASTER-K series

- 1) The I/O (P) and data (D) registers may differ for each series. Please refer the memory map of each series at the chapter 2.2 before convert a program for other MASTER-K series.
- 2) Some instructions are not supported by all series. Please refer the instruction table of appendix 9.
- 3) Please backup the original program before converting the program.

Chapter 2 Functions

2.1	Performance Specifications	2-1
2.1.1	K10S1 / K10S / K30S / K60S	2-1
2.1.2	K200S / K300S / K1000S	2-2
2.2	Memory configuration map	2-3
2.2.1	K10S1	2-3
2.2.2	K10S / K30S / K60S	2-4
2.2.3	K200S / K300S / K1000S	2-5
2.3	Memory devices of MASTER-K series	2-6
2.3.1	Input / output area : P	2-6
2.3.2	Auxiliary relay : M	2-7
2.3.3	Keep relay : K	2-8
2.3.4	Link relay : L	2-8
2.3.5	Step control relay : S	2-8
2.3.6	Timer relay : T	2-9
2.3.7	Counter relay : C	2-10
2.3.8	Data register : D	2-11
2.3.9	Indirect assignment data register : #D	2-11
2.3.10	Special relay : F	2-12
2.3.11	Special M / L relay : M / L	2-12
2.3.12	Special data register : D	2-12
2.4	Parameter setting	2-13
2.4.1	Watch dog timer setting	2-13
2.4.2	Timer area setting	2-13
2.4.3	The latch (non-volatile) area setting	2-13
2.4.4	Setting the mode of CPU (RUN / STOP) when an error is occurred ..	2-14
2.4.5	Station number / Baud rate setting	2-14

2.4.6	High speed counter setting	2-15
2.4.7	Interrupt setting.....	2-15
2.4.8	The reservation of I/O number allocation	2-15
2.4.9	The output of Debug mode	2-16
2.5	Operation processing of CPU	2-17
2.5.1	The repetitive operation	2-17
2.5.2	The operation mode of CPU	2-18
2.6	Special functions of MASTER-K series	2-20
2.6.1	Interrupt functions	2-20
2.6.2	RTC (Real Time Clock) function	2-24
2.6.3	Forced I/O setting	2-27
2.6.4	Program edit in RUN mode.....	2-28
2.6.5	Self-diagnosis	2-29
2.7	Program check	2-30
2.7.1	JMP – JME.....	2-30
2.7.2	CALL , SBRT / RET	2-31
2.7.3	MCS – MCSCLR.....	2-32
2.7.4	FOR – NEXT.....	2-33
2.7.5	END / RET	2-34
2.7.6	Dual coil	2-34
2.8	Error handling.....	2-35
2.8.1	RUN / STOP at operation error.....	2-35
2.8.2	Error flag (F110 / F115).....	2-35
2.8.3	LED indication.....	2-36
2.8.4	Error code list.....	2-37

2 Functions

2.1 Performance Specifications

2.1.1 K10S1 / K10S / K30S / K60S

Items		K10S1	K10S	K30S	K60S
Program control method		Cyclic execution of stored program			
I/O control method		Indirect (Refresh) method			
Numbers of Instructions	Basic	30			
	Application	226			
Processing speed		3.2 ~ 7.6 □/step	1.2 □/step		
Program capacity		800 steps	2,048 steps		
P (I/O relay)		P0000 ~ P001F (32 points)	P0000 ~ P005F (96 points)		
M (Auxiliary relay)		M0000 ~ M015F (256 points)	M0000 ~ M031F (512 points)		
K (Keep relay)		K0000 ~ K007F (128 points)	K0000 ~ K015F (256 points)		
L (Link relay)		L0000 ~ L007F (128 points)	L0000 ~ L015F (256 points)		
F (Special relay)		F0000 ~ F015F (256 points)	F0000 ~ K015F (256 points)		
T (Timer relay)	100ms	T000 ~ T031 (32 points)	T000 ~ T095 (96 points)		
	10ms	T032 ~ T047 (16 points)	T096 ~ T127 (32 points)		
C (Counter relay)		C000 ~ C015 (16 points)	C000 ~ C127 (128 points)		
S (Step controller)		S00.00 ~ S15.99 (16×100 steps)	S00.00~ S31.99 (32×100 steps)		
D (Data register)		D0000 ~ D0063 (64 words)	D0000 ~ D0255 (256 words)		
The range of integer		16 bit : – 32768 ~ 32767 32 bit : – 2147483648 ~ 2147483647			
Timer types		On-delay, Off-delay, Accumulation, Monostable, Retriggerable (5 types)			
Counter types		Up, Down, Up-down, Ring counter (4 types)			
Programming language		Mnemonic, Ladder diagram			
Special functions		Real time clock, High speed counter, RS-485 communication			

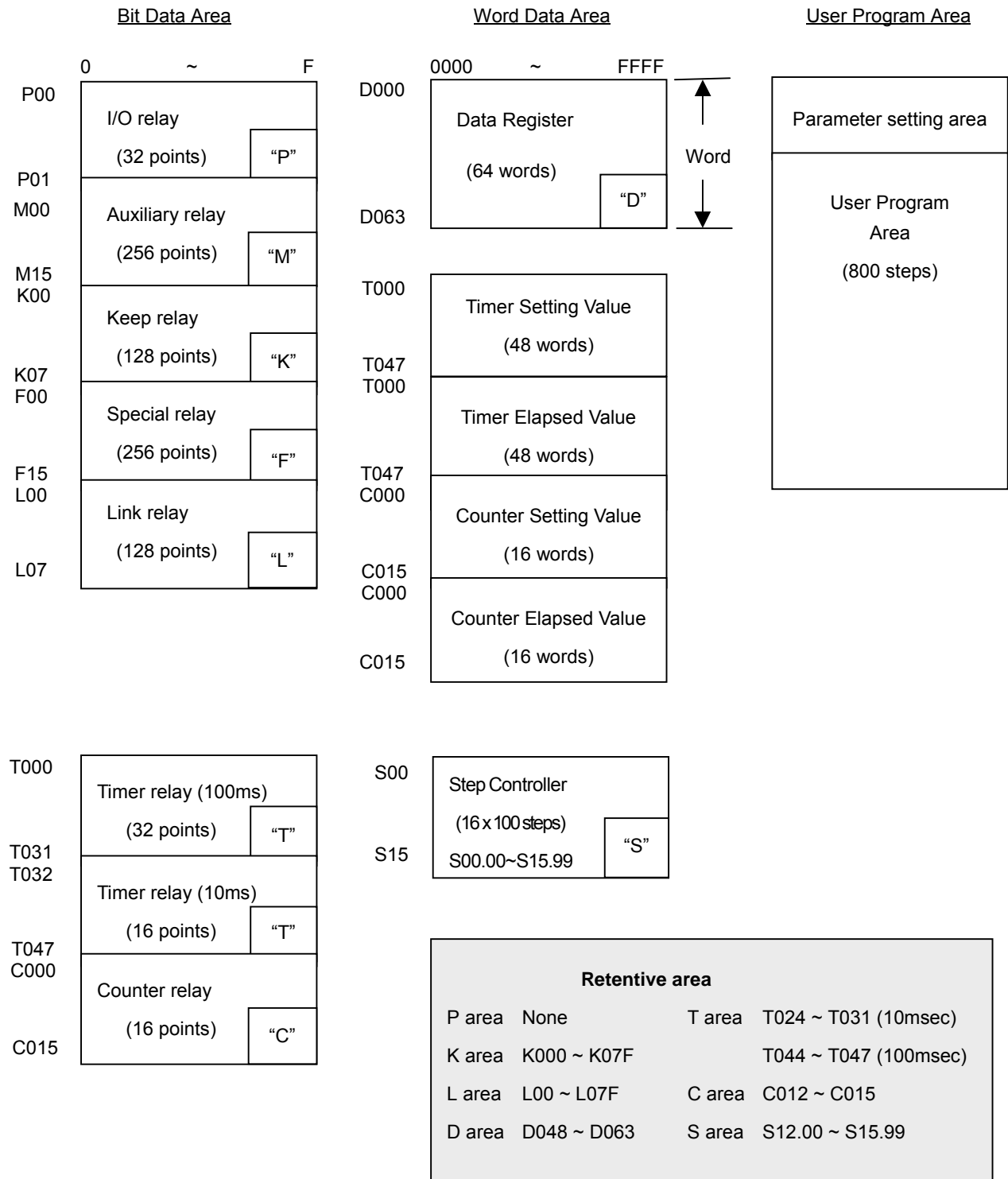
Chapter 2. Functions

2.1.2 K200S / K300S / K1000S

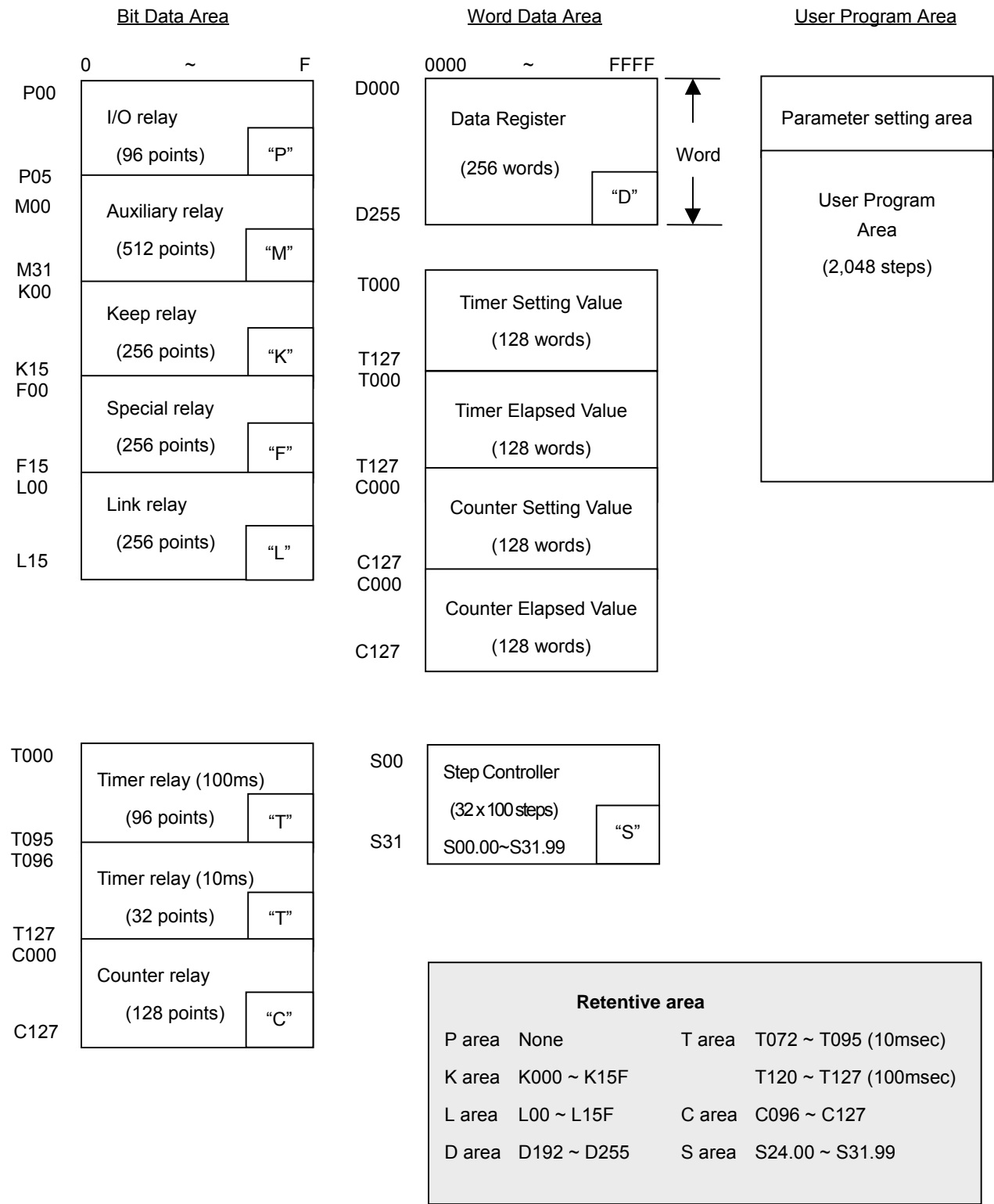
Items		K200S	K300S	K1000S
Program control method		Cyclic execution of stored program, Time-driven interrupt, Event-driven interrupt		
I/O control method		Indirect , Direct by program command		
Numbers of Instructions	Basic	30		
	Application	226	228	
Processing speed		0.5 □/step	0.2 □/step	
Program capacity		7k steps	15k steps	30k steps
P (I/O relay)		P0000 ~ P015F (256 points)	P0000 ~ P031F (512points)	P0000 ~ P063F (1,024 points)
M (Auxiliary relay)		M0000 ~ M191F (3,072 points)		
K (Keep relay)		K0000 ~ K031F (512 points)		
L (Link relay)		L0000 ~ L063F (1,024 points)		
F (Special relay)		F0000 ~ F063F (1,024 points)		
T (Timer relay)		100msec (T000 ~ T191 : 192 points), 10msec (T192 ~ T255 : 64 points) The range of 100ms and 10ms timer can be changed with parameter setting.		
C (Counter relay)		C000 ~ C255 (256 points)		
S (Step controller)		S00.00 ~ S99.99 (100×100 steps)		
D (Data register)		D0000 ~ D4999 (5,000 words)		D0000 ~ D9999 (10,000 words)
The range of integer		1. Signed instruction 16 bit : – 32768 ~ 32767 32 bit : – 2147483648 ~ 2147483647 2. Unsigned instruction 16 bit : 00000 ~ 65535 32 bit : 00000000 ~ 4295967295		
Timer types		On-delay, Off-delay, Accumulation, Monostable, Retriggerable (5 types)		
Counter types		Up, Down, Up-down, Ring counter (4 types)		
Programming language		Mnemonic, Ladder diagram		
Special functions		Real time clock, RUN mode editing, Forced I/O control		

2.2 Memory configuration map

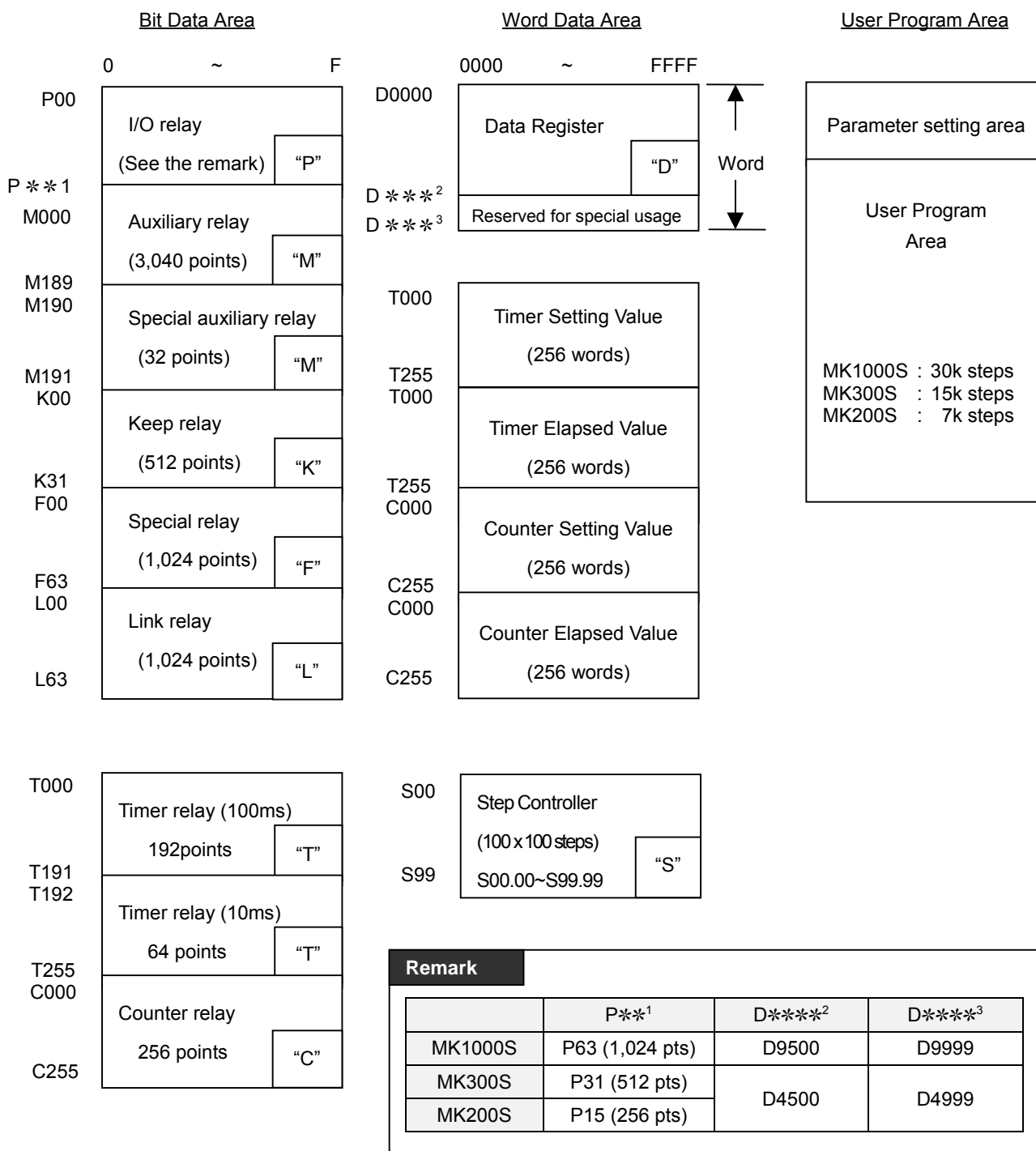
2.2.1 K10S1



2.2.2 K10S / K30S / K60S



2.2.3 K200S / K300S / K1000S



2.3 Memory devices of MASTER-K series

2.3.1 Input / output area : P

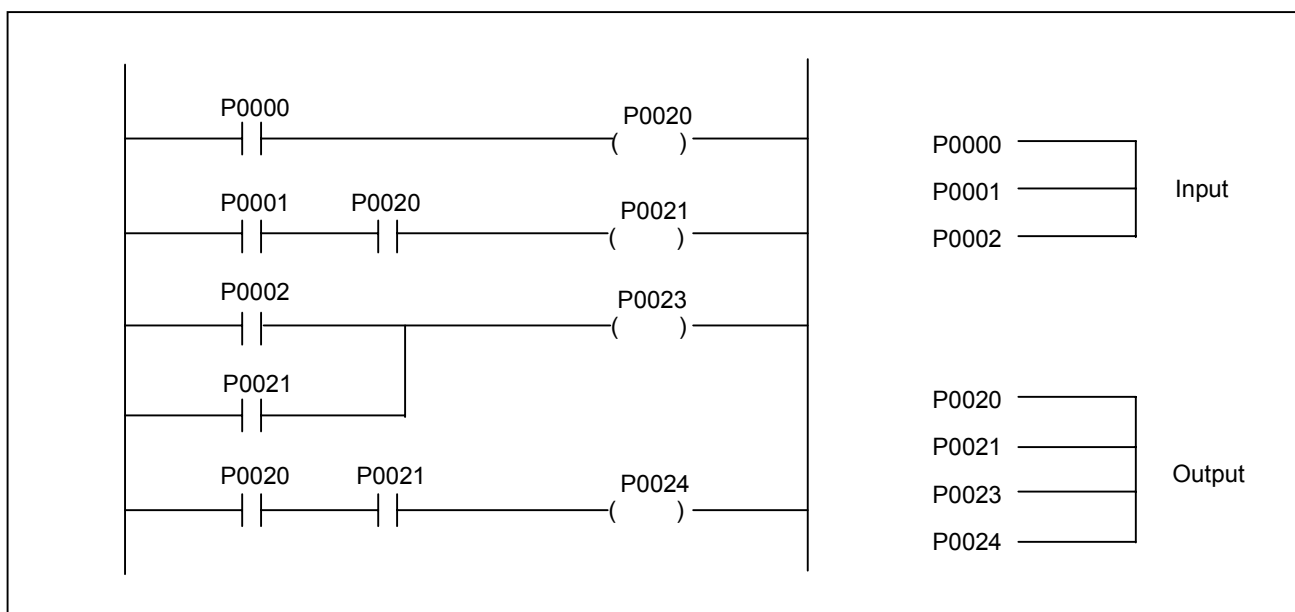
The P devices are used for data transaction between the PLC CPU and external devices.

The input devices hold ON/OFF data sent from external devices (e.g. pushbuttons, select switches, limit switches, digital switches, etc.) to input module. Input data is used by the program as contact data (NO¹ and NC contacts) and as the source data for basic and application instructions.

The output devices are used to output operation results of the program from the output module to external devices (e.g. solenoids, magnetic switches, signal lamps, digital indicators). Only NO contact type is available for output devices.

The redundant P devices that are not connected to external devices can be used in the same way with the auxiliary relay M.

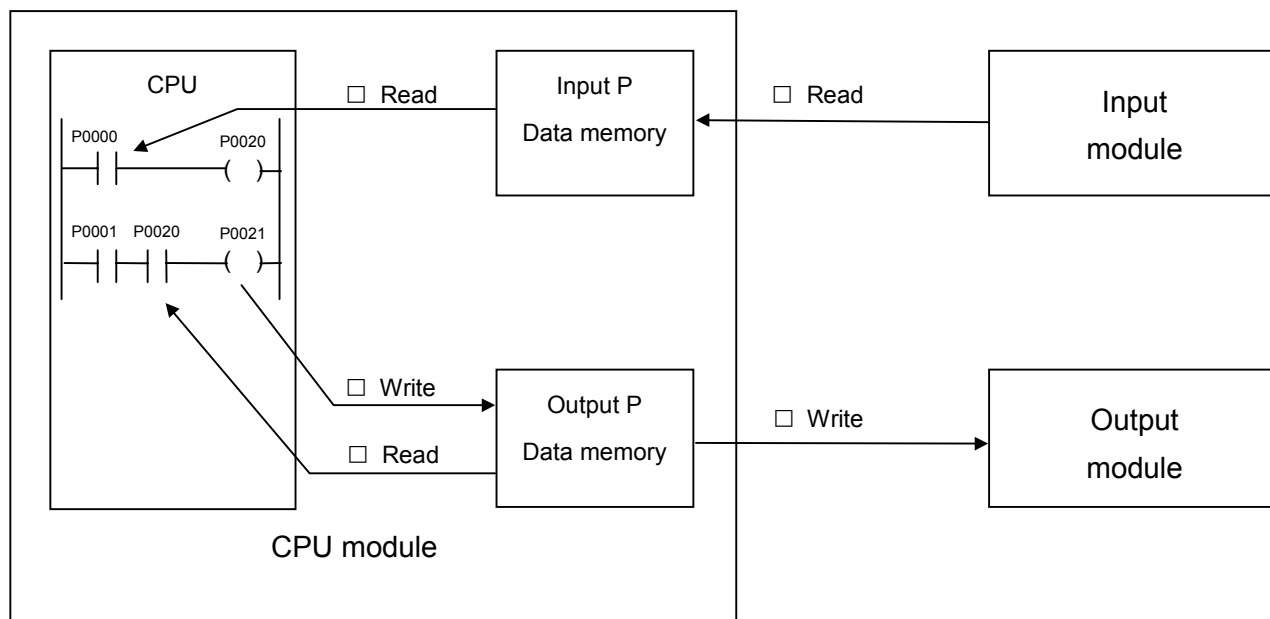
< Figure 1. The example of input/output configuration >



The input signals are stored in batch in the input data memory before execution of each scan. The data in the input data memory is used for execution of the sequence program operation. The operation results are output by each result to the output data memory. The data in the output data memory is output in batch to the output modules after execution of the END instruction. Please make sure that there is no conflict of input and output in the user program because the MASTER-K series uses a P area for input and output in common.

¹ NO : Normally Open contact, NC : Normally Closed contact

< Figure 2. Flow of input / output data in the refresh mode >



- Input refresh

Input data is read (□) in batch from the input module before execution of step 0 and stored in the input data memory.

- When an input contact command is executed :

Input data is read (□) from the input data memory and used for execution of the sequence program.

- When an output contact command is executed :

Output data is read (□) from the output data memory and used for execution of the sequence program.

- When an output OUT instruction is executed :

The operation result (□) is stored in the output data memory.

- Output refresh

Data (□) in the output data memory is output in batch to the output module after execution of the END instruction.

2.3.2 Auxiliary relay : M

The M area is internal relay used in the PLC CPU, and can not be connected directly with external devices. All M area except designated as latched area will be cleared as 0 when the PLC is switched on or turned to RUN mode. With K200S / K300S / K1000S, a user can change the latched area by parameter setting.

2.3.3 Keep relay : K

The K area functions as same as M area. However, the operation results are retained if the PLC is switched on or turned to RUN mode. The K area can be cleared by following methods;

- put the initialization routine in the sequence program.
- Run the data clear function of hand-held loader (KLD-150S)
- Run the data clear function of graphic loader (KGL-WIN)

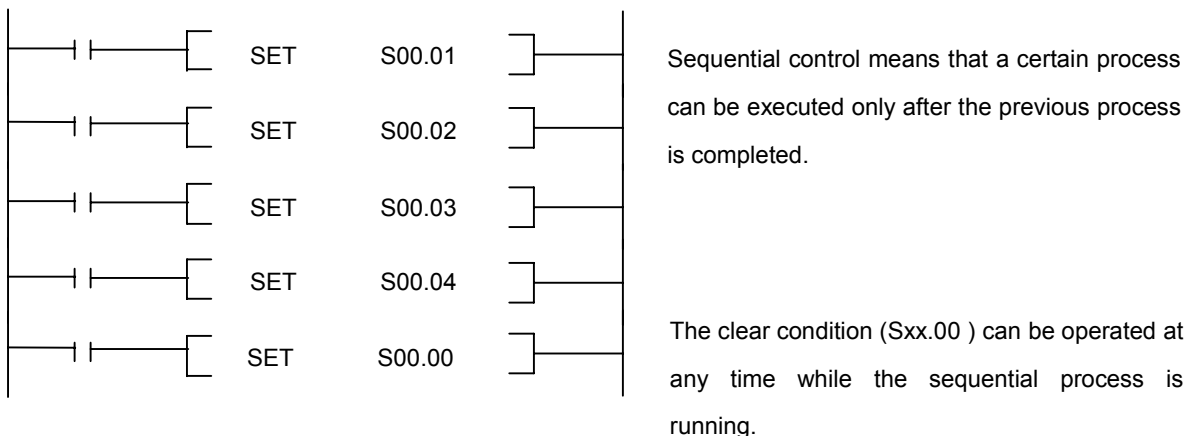
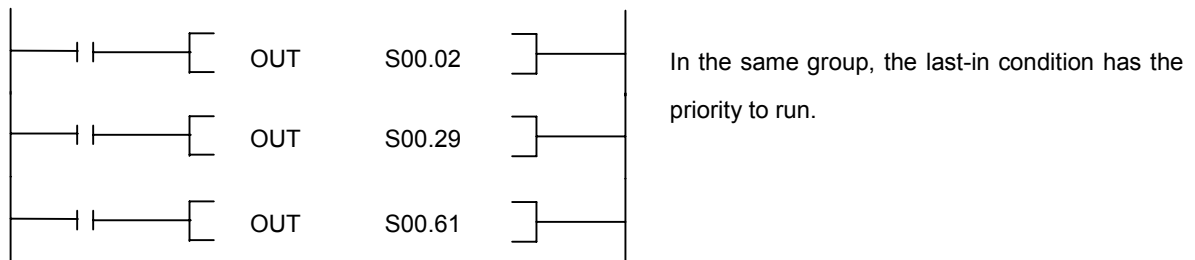
2.3.4 Link relay : L

The L area is the internal memory for use in a data or computer link system. It can be used as same as M area if no link module is mounted on the PLC system. With K200S / K300S / K1000S, it is possible to change the range of latch area by parameter setting. For the detail usage of L area, please refer the list of link relay at appendix and the computer link user's manual.

2.3.5 Step control relay : S

The S area can be used for two kinds of step control according to the instruction – OUT or SET. If the OUT instruction is used, the S area functions as last-in priority. Otherwise, it functions as sequential control. (See the chapter 4 for detailed usage.)

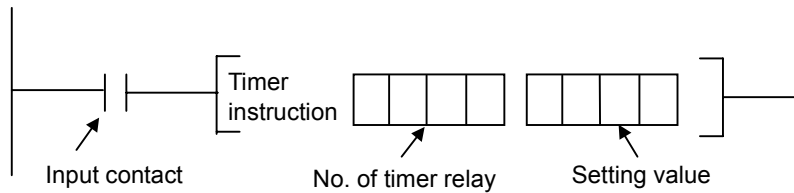
When the CPU is switched on or turned to RUN mode, the S area will be initialized as first step (Sxx.00) except the latch area designated by parameter setting.



2.3.6 Timer relay : T

MASTER-K series have 100msec and 10msec timer. The timing method is various according to the timer instructions (TON, TOFF, TMR, TMON, TRTG). The maximum timer setting value is hFFFF by hexadecimal or 65535 by decimal. The following figure shows the types and timing methods of each timer instruction.

< Figure 3. Types and timing methods of timer instructions >

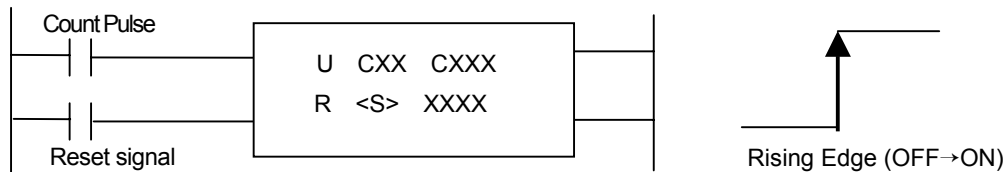


Timer instruction	Description	Timing method	Time chart
TON	ON Delay	Increment	<p>Input condition</p> <p>Timer output</p> <p>ON Delay timer</p> <p>$t = \text{setting value}$</p>
TOFF	OFF Delay	Decrement	<p>Input condition</p> <p>Timer output</p> <p>OFF Delay timer</p> <p>$t = \text{setting value}$</p>
TMR	Accumulation ON Delay	Increment	<p>Input condition</p> <p>Timer output</p> <p>Accumulation timer</p> <p>$t = t_1 + t_2$</p> <p>$t = \text{setting value}$</p>
TMON	Monostable	Decrement	<p>Input condition</p> <p>Timer output</p> <p>Monostable timer</p> <p>$t = \text{setting value}$</p>
TRTG	Retriggerable	Decrement	<p>Input condition</p> <p>Timer output</p> <p>Retriggerable timer</p> <p>$t = \text{setting value}$</p>

2.3.7 Counter relay : C

The counter counts the rising edges of pulses driving its input signal and counts once only when the input signal is switched from off to on. MASTER-K series have 4 counter instructions such as CTU, CTD, CTUD, and CTR. The maximum counter setting value is hFFFF (= 65535). The followings shows brief information for counter operation.

< Figure 4. Types and counting methods of counter instructions >



Counter instruction	Type	Counting method	Input signal	Time chart
CTU	Up Counter	Increment	1	
CTD	Down counter	Decrement	1	
CTUD	Up/Down Counter	Increment / Decrement	2	
CTR	Ring counter	Increment	1	

2.3.8 Data register : D

The D area is used to store numeric data. Each data register consists of 16 bits (1 word) which is the unit of data read and write.

The data register number designated by the double-word instruction holds the lower 16 bits and the designated data register number + 1 holds the higher 16 bits.

Example)



High 16 bits	Lower 16 bits
D51	D50
h1234	h5678

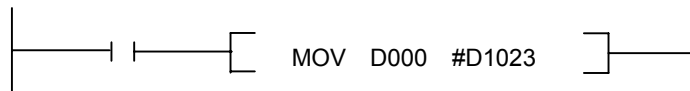
The D area except latched area assigned by parameter setting will be cleared as 0 when the CPU is switched on or turned to RUN mode.

2.3.9 Indirect assignment data register : #D

#D is used for indirect addressing of the D area. The contained value of data register assigned with '#' symbol points the real address of data register at which the result of operation is stored.

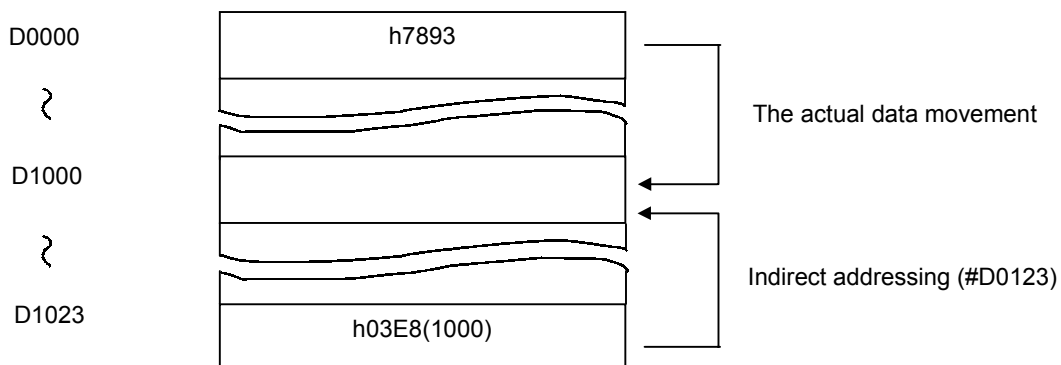
If #D is used with a double-word instruction, the lower 16 bits will stored at the data register number designated by the contained value of #D, and higher 16 bits will stored at the data register number + 1.

Example)



(No. of data register)

(Hex value)

**Remark**

If the value of data register assigned at # D exceeds the physical address range of D area, the operation error flag(F110) will be set and the relevant instruction will be ignored.

2.3.10 Special relay : F

The F area is read-only relay and user can not change the value of F area. See the F relay table at the appendix for details.

2.3.11 Special M / L relay : M / L

Some M or L relays are reserved for special usage. See the list of special relays at the appendix and be careful when use those M or L area in the program.

2.3.12 Special data register : D

Some data registers are reserved for special usage. These registers are various according to the type of CPU. See the list of special registers at the appendix and be careful when use those data register in the program.

2.4 Parameter setting

2.4.1 Watch dog timer setting

(Applicable to K200S / K300S / K1000S only)

Setting range : 10msec ~ 6000msec

Setting unit : 10msec

The default value of watch dog timer is 200msec. The watch dog timer of K10S1, K10S, K30S, and K60S is fixed as 200msec.

2.4.2 Timer area setting

(Applicable to K200S / K300S / K1000S only)

Timing unit	Setting range	Default
100ms	T000 ~ T255	T000 ~ T191
10ms	T000 ~ T255	T192 ~ T255

By setting the 100msec timer area, the 10msec timer area is automatically set as the rest of area.

2.4.3 The latch (non-volatile) area setting

(Applicable to K200S / K300S / K1000S only)

The latch area designated by parameter setting will hold the result of operation when the CPU is switched on or turned to RUN mode.

The latch area of K10S1/K10S/K30S/K60S is fixed and can not be changed. Please refer the Chapter 2.2 for memory configuration.

Device		Setting range	Default range
M		M0000~M191F	None
L		L0000~L063F	
T(100ms)		T0000~T0191 *	T144~191 *
T(10ms)		T0192~T0255 *	T240~T255 *
C		C0000~C0255	C192~C255
D	K1000S	D0000~D9999	D6000~D8999
	K300S	D0000~D4999	D3500~D4500
	K200S		
S		S00.00~S99.99	S80~S99

* The setting range of timer can be changed by 100msec / 10msec timer range setting.

2.4.4 Setting the mode of CPU (RUN / STOP) when an error is occurred

(Applicable to K200S / K300S / K1000S only)

When a non-critical error such as fuse blown or operation error, the CPU will keep running or stop operation according to the parameter setting. See the following table for details.

(K10S1/K10S/K30S/K60S is set as the default setting of K200S/300S/1000S.)

* = Default setting

Error type	Description	Mode	RUN LED	Error flag
Fuse blown	The fuse of output or hybrid module is blown	RUN * / STOP	ON * / OFF	F035
Operation Error	BCD error The result of BCD conversion is exceeds the specified range (9999 or 99999999) Over range error One or more operands exceed the specified device range.	RUN * / STOP	ON * / OFF	F110 F115

The F110 bit is updated after each instruction is completed. Therefore, even an operation error was occurred, the F110 will be cleared if the next instruction is completed normally. In other hand, the F115 will be set when an operation error occurs and keep the on status until user cleared the F115 with CLE instruction.

2.4.5 Station number / Baud rate setting

(Applicable for K10S1 / K10S / K30S / K60S / K200S)

PLC type	Protocol	Station number	Baud rate	Descriptions
K10S1	RS-485	0 ~ 31 (h00 ~ h1F)	300, 600, 1200, 2400, 4800, 9600, 19200 bps	
K10S				
K30S				
K60S				
K200S-A ²	RS-232	N/A	9600, 19200, 38400 bps	
K200S-B/C ³	RS-422	0 ~ 31		

² K3P-07AS

³ K3P-07BS / K3P-07CS

2.4.6 High speed counter setting

(Applicable for K10S1, K10S, K30S, K60S)

The block type models of MASTER-K series include the high speed counter function in the main module. When the 'HSCNT' instruction is used, the high speed counter parameters should be set with KGL-WIN or KLD-150S. Refer the 5.22.1 'HSCNT' instruction part.

2.4.7 Interrupt setting

(Applicable for K200S / K300S / K1000S)

1) The priority of interrupts setting

Each of interrupts has a priority level. If two or more interrupts occur at the same time, the CPU will process the interrupt that has higher priority. Priority levels are described by numbers, and smaller number means higher priority.

2) TDI (Time driven interrupt)

TDI is the interrupt that occurs periodically. The period of interrupt can be set with parameters by 10-msec unit. The longest period is 60 seconds.

3) PDI (Process driven interrupt)

PDI is the interrupt that occurs when an input of interrupt module was turned on.

Refer the chapter 2.6.1 for more details.

2.4.8 The reservation of I/O number allocation

(Applicable for K200S / K300S / K1000S)

The I/O number is allocated automatically by CPU module according to the actual status of module. However, user can reserve I/O number allocation with parameter setting, then it makes a user keep a sequence program in case of module fault or replacement.

- 1) User can assign the type of module (input, output, or special module) and I/O occupation number to each module.
- 2) If the reserved I/O number is larger than the I/O number of actual module, the redundant reserved I/O points are used as internal relay. Otherwise, the redundant actual I/O points are disabled.
- 3) Non-reserved slots occupy I/O points as the I/O points of actual module, and special modules occupy 16 I/O points.

4) The example of I/O reservation

a) Actual status of module mounting

Power supply module	CPU module	AC Input	DC Input	Relay Output	Special module (Analog input)	Empty slot	DC input	Relay output	Empty	TR output
		16 Pts	32 Pts	16 Pts	16 Pts		16 Pts	16 Pts		32 Pts

b) I/O address assignment according to the I/O parameter setting

No parameter setting	000 ~00F	010 ~02F	030 ~03F	040 ~04F	050 ~05F	060 ~06F	070 ~07F	080 ~08F	090 ~10F
Parameter setting	AC Input 16 Pts	DC Input 16 Pts	Relay Output 32 Pts	Special AD 16 Pts	Empty 16 Pts	Empty 0 Pts	Relay Output 16 Pts	Empty 0 Pts	TR Output 16 Pts
	000 ~00F	*1) 010 ~01F	*2) 020 ~03F	040 ~04F	050 ~05F	*3)	060 ~06F		070 ~07F

*1) The upper 16 pts of module is disabled.

*2) The upper 16 pts (P030 ~ P03F) are used as internal relay.

*3) Since the slot is reserved as empty, the input module is disabled.

2.4.9 The output during debugging

(Applicable for K200S / K300S / K1000S)

With this parameter setting, user can decide that the result of operation will be output to external device or not when the CPU is in DEBUG mode.

2.4.10 The slot No. for external interrupt

(Applicable for K200S only)

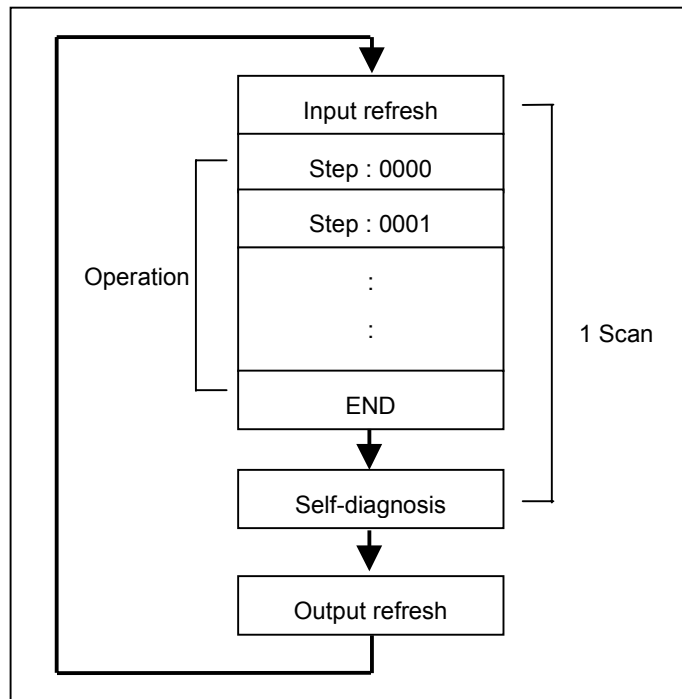
The K200S series has no interrupt module. Therefore, user can assign an input module as interrupt input module and input signals applied to this module will be handled as external interrupt input.

2.5 Operation processing of CPU

2.5.1 The repetitive operation

The repetitive operation method repeats execution of a series of operations. The CPU repeats the operation processing as follows.

Fig. 2-3 Operation processing of the CPU



The CPU refreshes input data, then executes the sequence program stored in the internal memory, beginning with step 0 to the END instruction. After executing the END instruction, the CPU performs self-diagnosis and refreshes output data, and then returns to input refresh.

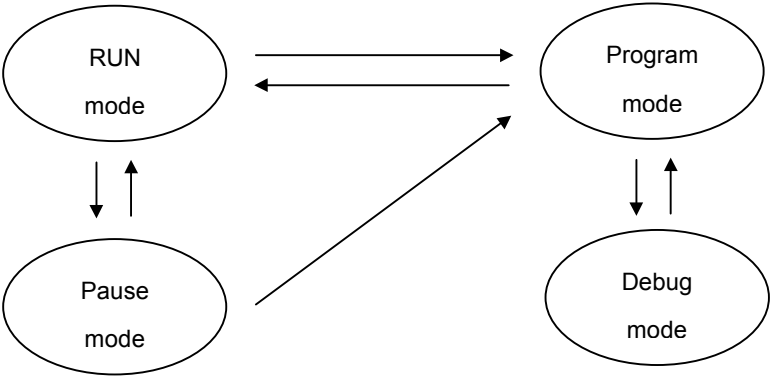
Remark

Scan : The series of steps from step 0 to the next step 0 is called a scan. Therefore, a scan time of the CPU is calculated as a total of the processing time of the sequence program (step 0 to END) and the internal processing time (self-diagnosis and I/O refresh) of the CPU.

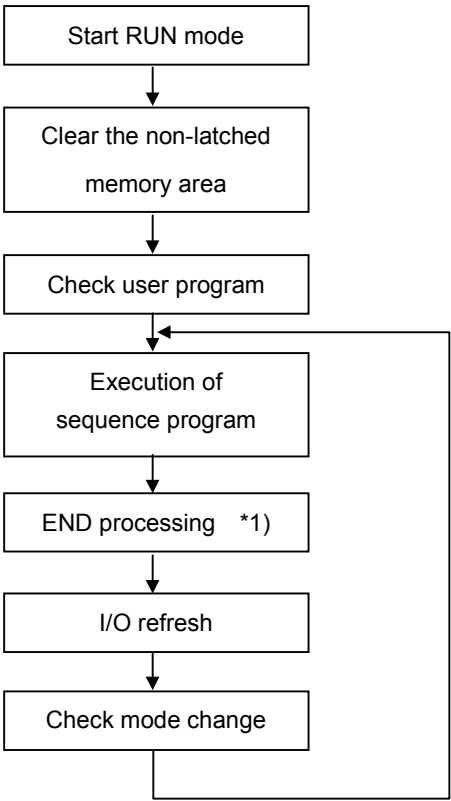
2.5.2 The operation mode of CPU

MASTER-K series has 4 operation modes as shown below. The arrow indicates that mode change is available.

<Figure 2-4 Operation modes of MASTER-K series>

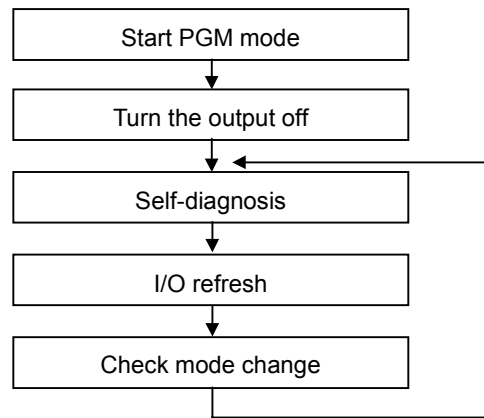


1) The flow of RUN mode



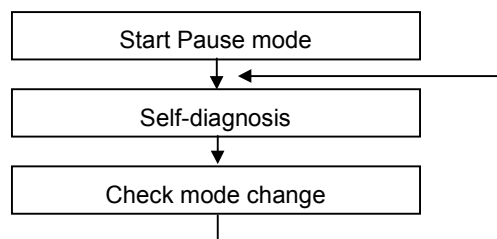
*1) END processing : Self-diagnosis, Timer / Counter update

2) The flow of Program (PGM) mode



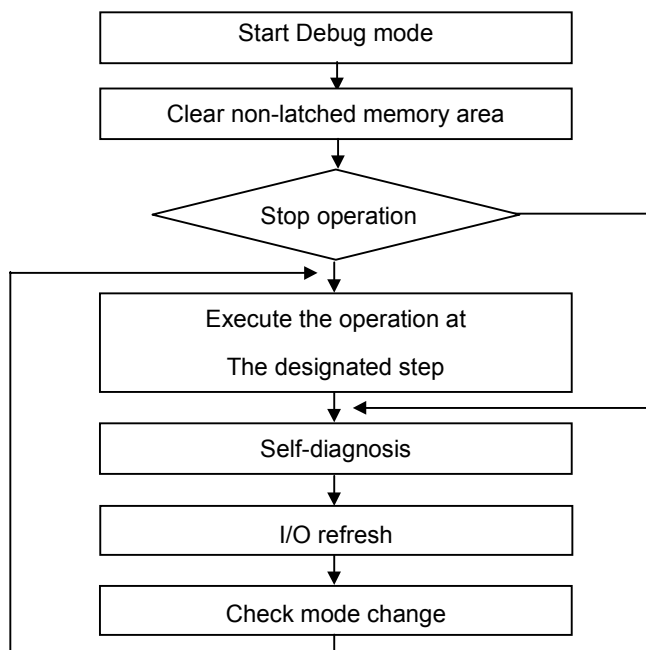
- Program read / write / monitor can be performed in program mode. External wiring check is also possible with the forced I/O on/off function.

3) The flow of Pause mode



- Stops the operation of CPU, but keep the status of output and internal memory.

4) The flow of Debug mode



2.6 Special functions of MASTER-K series

2.6.1 Interrupt functions

(Applicable for K200S / K300S / K1000S)

When an interrupt occurs, the CPU module will stop the current operation and execute the corresponding interrupt routine. After finish the interrupt routine, the CPU resume the sequence program from the stopped step.

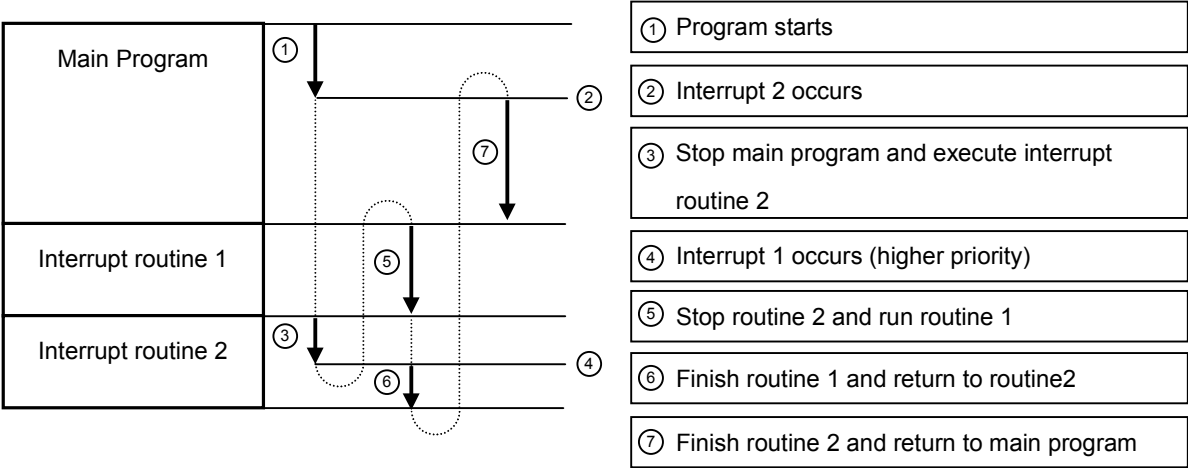
MASTER-K series provides 2 types of interrupt. The TDI (Time driven interrupt) occurs with the constant period, and PDI (Process driven interrupt) occurs with the status of external input.

Before to use interrupt function in sequence program, the parameter setting should be done properly. (See 2.4.6 for parameter setting) Then the corresponding interrupt routine should be written after END instruction. (Refer chapter 4 for details) If interrupt routines are not matched with parameter settings, an error occurs and the operation of CPU will be stopped.

To execute an interrupt routine, use the EI instruction to enable the corresponding interrupt. The interrupt routine is not executed if an interrupt factor occurs before execution of an EI instruction. Once an interrupt is enabled with EI instruction, it keeps the enabled status until DI instruction is executed to disable the interrupt. When a CPU is turned to RUN mode, all interrupts are disabled by default.

When multiple interrupt factors occur simultaneously, interrupt routines are executed according to the priority given to the each interrupts. If an interrupt factor that has higher priority occurs while other interrupt that has lower priority are executing, the interrupt routine of lower priority will be stopped and the interrupt of higher priority will be executed first. The following figure shows how a CPU handle multiple interrupts.

<Figure 2. 5 The execution order of multiple interrupts>



1) Parameter setting

K200S			K300S			K1000S		
Priority	Type	Period	Priority	Type	Period	Priority	Type	Period
0	TDI0	10msec	0	TDI0	10msec	0	TDI0	10msec
1	TDI2	25msec	1	TDI2	25msec	1	TDI2	25msec
2	TDI5	100msec	2	TDI5	100msec	2	TDI5	100msec
:	:		:			:		
:	:		:			:		
7	INT7		:			:		
			13	INT7		:		
						29	INT15	

Remark

- a) Period is the interval of time driven interrupt occurring. It is variable from 10msec to 60000msec (60sec) by 100msec unit.
- b) The priority is also used as the number of interrupt. To enable/disable the TDI5 interrupt with priority level 2, for example, use EI/DI instruction as 'EI 5'/'DI5'.

2) TDI (Time driven interrupt)

TDI occurs periodically with the constant interval assigned in parameter setting. The interrupt routine of TDI starts with the TDINT instruction and ends with the IRET instruction.

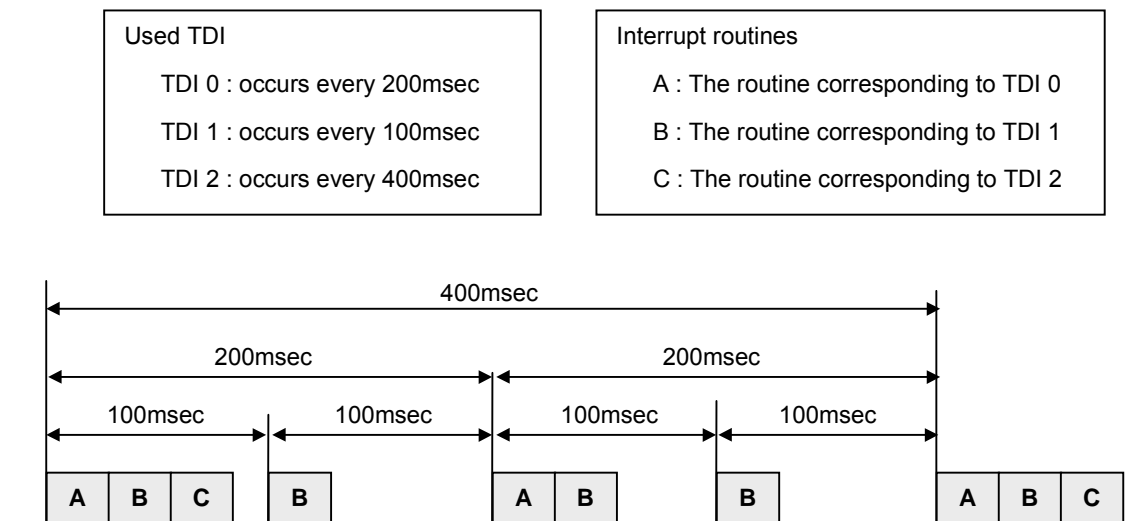
When multiple interrupt factors occur simultaneously, interrupt routines are executed according to the priority given to the each interrupt. If an interrupt factor has higher priority occurs while other interrupt has lower priority are executing, the interrupt routine of lower priority will be stopped and the interrupt of higher priority will be executed first. Otherwise, two interrupts are executed consequently.

The maximum numbers of TDI for K200S / 300S / 1000S are shown as following table. See the 2.4.5 for details of parameter setting.

PLC type	Available TDI
K200S	TDINT 0 ~ 7
K300S	TDINT 0 ~ 13
K1000S	TDINT 0 ~ 29

The following figure shows an example of TDI execution.

<Figure 2.6 The example of execution of TDI>

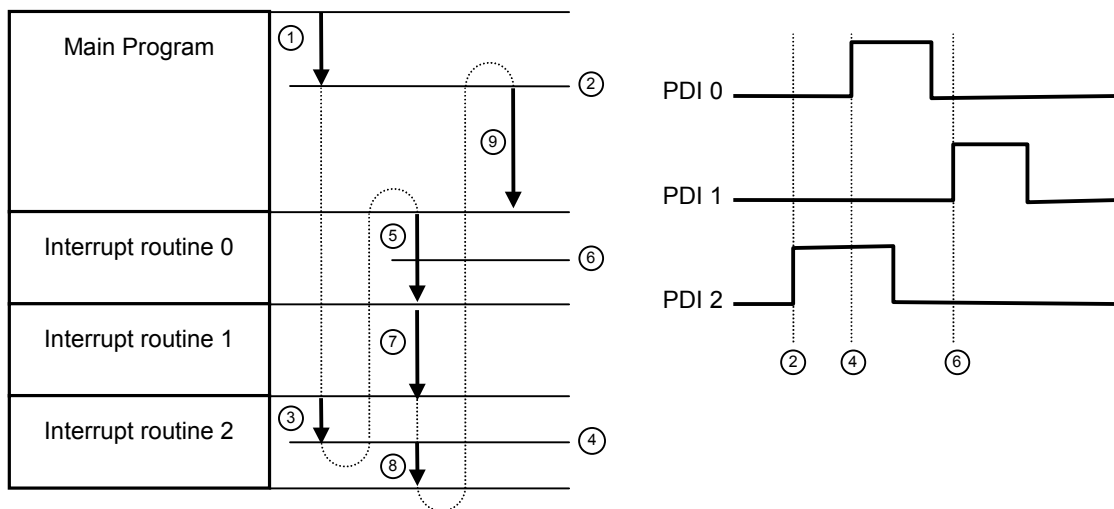


3) PDI (Process driven interrupt)

PDI occurs when the input status of interrupt module is changed from OFF to ON or from ON to OFF. (Select by DIP switch setting) Since K200S does not have interrupt module, PDI will occur when the input assigned as interrupt input by parameter setting is changed from OFF to ON.

The execution order of multiple interrupts is similar as TDI. The following figure shows an example of execution order of multiple PDI.

<Figure 2-7 The execution order of multiple PDI>



- | |
|---|
| ① Program starts |
| ② Interrupt 2 occurs |
| ③ Stop main program and run PDI routine 2 |
| ④ Interrupt 0 occurs (higher priority) |
| ⑤ Stop routine 2 and execute routine 0 |
| ⑥ Interrupt 1 occurs (lower priority) |
| ⑦ Finish routine 0 and execute routine 1 |
| ⑧ Finish routine 1 and resume routine 2 |
| ⑨ Finish routine 2 and back to main program |

2.6.2 RTC (Real Time Clock) function

Since the RTC function is optional function, not all MASTER-K series support this function. Please refer the Catalog and CPU manual for applicable models.

Clock operation by the RTC function is continued with a battery or super capacitor when the CPU is powered off.

1) Clock data

Clock data is the data comprised of year, month, day, hour, minute, second, and date.

Data name	Description	
Year	The lower 2 digits of the Christian Era	
Month	1 to 12	
Day	1 to 31 (A leap year is distinguished automatically)	
Hour	0 to 23 (24 hours)	
Minute	0 to 59	
Second	0 to 59	
Date	0	Sunday
	1	Monday
	2	Tuesday
	3	Wednesday
	4	Thursday
	5	Friday
	6	Saturday

2) Precision

Max. 1.728 second per day (general temperature)

3) K10S / K30S / K60S

a) Read RTC data

RTC data is stored as following table.

Memory Area (Word)	Description		Example data (BCD format)
	Higher byte	Lower byte	
L012	Year	-	h98xx
L013	Day	Month	h2212
L014	Hour	Date	h1902
L015	Second	Minute	h4637

Example : 1998. 12. 22. 19:37:46, Tuesday

b) Write RTC data

There is two ways to write new RTC data to the CPU.

The first one is using a handy loader (KLD-150S) or graphic loader (KGL-WIN). For detailed information, refer the user's manual of KLD-150S or KGL-WIN.

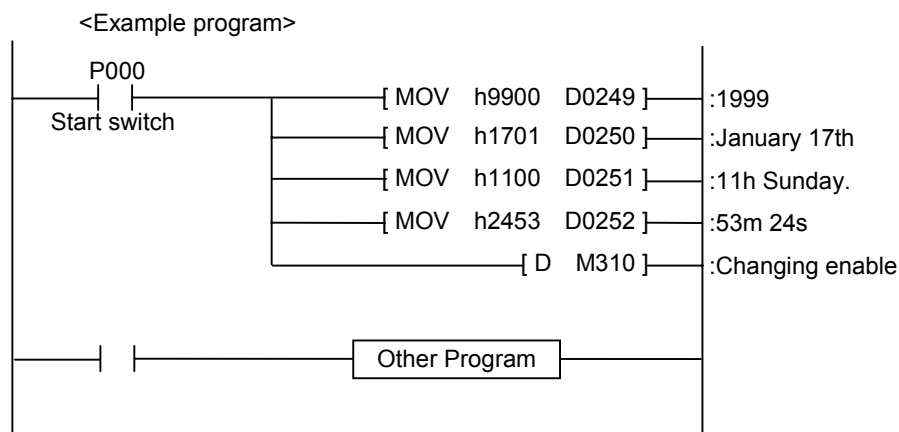
The second one is write sequence program. By switching a special bit on, user can replace the current RTC data with the preset data stored in a specified memory area. The followings are the memory address of preset data and an example program.

RTC preset data is stored as following table.

Memory Area (Word)	Description		Example data (BCD format)
	Higher byte	Lower byte	
D249	Year	-	h99xx
D250	Day	Month	h1701
D251	Hour	Date	h1100
D252	Second	Minute	h2453

Example : 1999. 1. 17. 11:53:24, Sunday

M310 (RTC data change bit) : When the M310 bit is switched on, the new data in D249 ~ D252 will be moved to L12 ~ L15. After data is moved, M310 has to be switched off immediately because current data will be updated every scan while M310 is on.

**Remark**

- The RTC data has not been set by factory default. Before using RTC function, write a correct RTC data to the CPU module.
- If unreasonable RTC data is written to the CPU, the RTC operation can not be executed normally.

Example : 13 (month) 32 (day)

4) K200S / K300S / K1000S

To read / write RTC data of K200S / K300S / K1000S is similar as K10S / K30S / K60S. The only difference is memory address of current / preset RTC data. See the following table.

The current RTC data

Memory Area (Word)	Description		Data (BCD format)
	Higher byte	Lower byte	
F053	Year	Month	h9812
F054	Day	Hour	h2219
F055	Minute	Second	h3746
F056	-	Date	hxx02

Example : 1998. 12. 22. 19:37:46, Tuesday

The preset RTC data

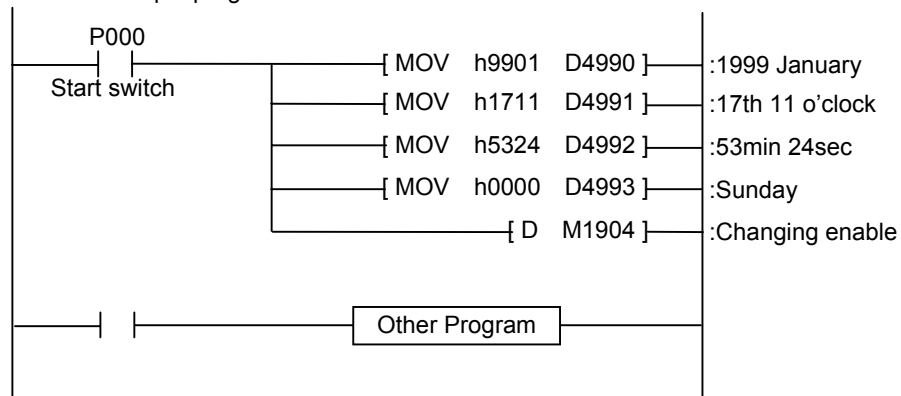
Memory Area (Word)		Description		Data (BCD format)
K200S / K300S	K1000S	Higher byte	Lower byte	
D4990	D9990	Year	Month	h9901
D4991	D9991	Day	Hour	h1711
D4992	D9992	Minute	Second	h5324
D4993	D9993	-	Date	hxx00

Example : 1999. 1. 17. 11:53:24, Sunday

M1904 : RTC data change bit

When the M1904 bit is switched on, the new data in D4990 ~ D4993 (D9990 ~ D9993) will be moved to F53 ~ F56. After data is moved, M1904 has to be switched off immediately because current data will be updated every scan while M1904 is on.

<Example program for K200S / K300S>



2.6.3 Forced I/O setting

(Applicable for K200S / K300S / K1000S)

It is possible to output a designated data regardless of the result of operation. This function is useful to check operation of the output modules and wiring between the output modules and external devices.

	K200S	K300S	K1000S
Forced I/O request bit	M1910		
The forced I/O address	D4700 ~		D9700 ~
The forced I/O data	D4800 ~		D9800 ~

Example 1) Output h8721 to the P10 word by force (K200S / K300S)

- a) Write the forced I/O data (h8721) to the corresponding data word. P10 is matched to the D4810 word.

<D4810 word>

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

- b) Write the forced I/O address (All bit = hFFFF) to the corresponding address word. Write hFFFF to the D4710.

<D4710 word> (0 = disable forced I/O, 1 = enable forced I/O)

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

- c) Switch on the forced I/O request bit (M1910).

- d) Output of P10 word

(P : The previous result of operation)

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



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

Example 2) Switch On/Off the last bit of P07 word (K1000S)

- a) Write the forced I/O data (h0001) to the corresponding data word. P10 is matched to the D9807 word.

<D9807 word>

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

- b) Write the forced I/O address (last bit = h0001) to the corresponding address word. Write h0001 to the D9707.

<D9707 word>

(0 = disable forced I/O, 1 = enable forced I/O)

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

- c) Switch on the forced I/O request bit (M1910).

- d) Output of P07 word

(P : The previous result of operation)

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



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

2.6.4 Program edit in RUN mode

(Applicable for K200S / K300S / K1000S)

User can insert, delete, or change instructions of program while the CPU is running. This function is useful to debugging or test-operation. Please refer the user's manual of KLD-150S or KGL-WIN for detail information.

Remark

The program edit in RUN mode can not be performed for the following instructions – JMP, JME, CALL, SBRT, FOR, and NEXT instructions. Moreover, the program that has very long scan time (longer than 2 seconds) can not be edit while the CPU is in the RUN mode.

2.6.5 Self-diagnosis

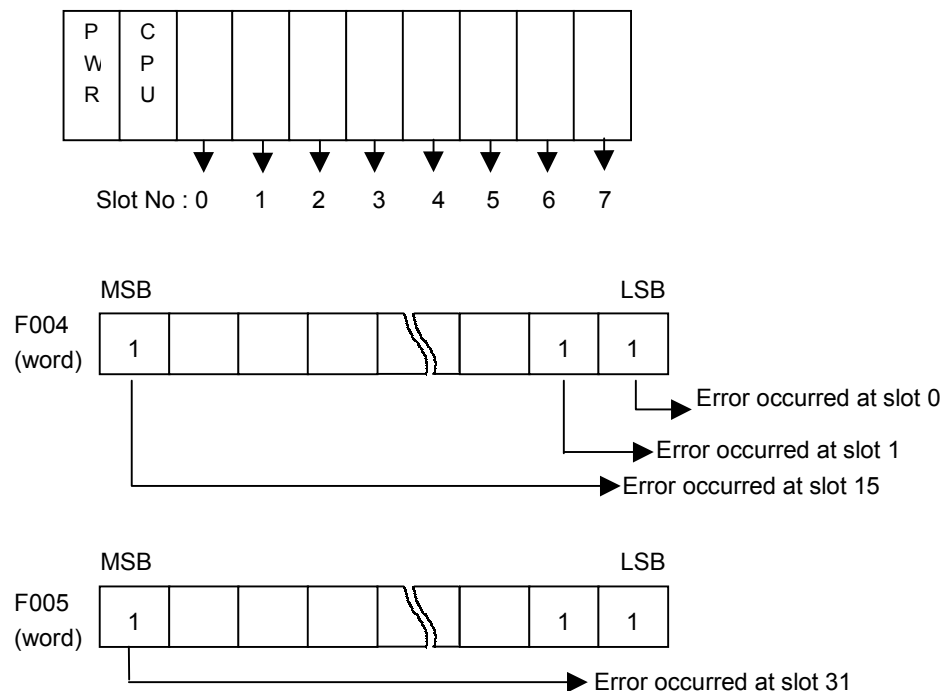
1) WDT (Watch dog timer) function

The watch dog timer is an internal timer of a PLC to detect the error of hardware and a sequence program. The default value is set as 200msec, and it is changeable with parameter setting. (K200S / K300S / K1000S only) Refer 2.4.1 for details on the parameter setting.

The CPU resets the watch dog timer before step 0 is executed (after the END processing is finished). When the END instruction has not been executed within the set value due to an error occurred in the PLC or the long scan time of a sequence program, the watch dog timer will times out. When a watch dog timer error is occurred, all outputs of the PLC are turned OFF, and the ERR LED of the CPU will flashes. (RUN LED will be turned OFF) Therefore, when use FOR ~ NEXT or CALL instruction, insert WDT instruction to reset the watch dog timer.

2) I/O module check function

If one or more I/O modules are mounted/dismounted while the PLC is powered, the corresponding bit (F0040 ~ F0050 : 32 bits) will be switched on. If a module is mounted improperly, the relevant bit will be switched on also.



3) Battery check function

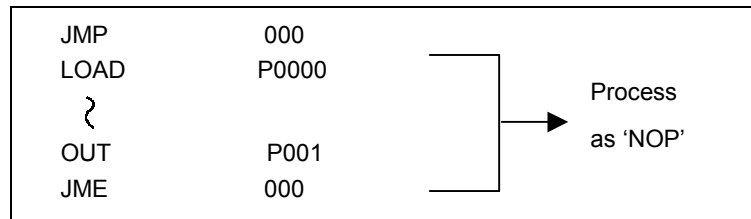
(Applicable for K200S / K300S / K1000S)

When the voltage of the battery for back-up the memory IC of CPU are lower than the minimum back-up voltage, the BAT LED of CPU module will be turned on.

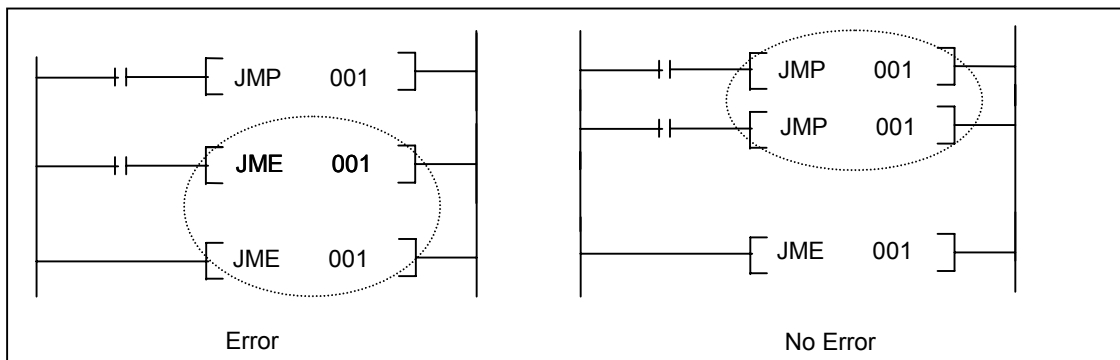
2.7 Program check

2.7.1 JMP – JME

- 1) If the input condition of JMP n instruction is turned on, the CPU skips all instructions until JME n instruction. The skipped instructions are processed as NOP instruction. Max. 128 JMP-JME can be used. (JMP 0 ~ JMP 127, JME 0 ~ JME 127)



- 2) The JMP n instruction should be matched only one JME n instruction. The duplication of JME is not permitted. However, the duplication of JMP n instructions is possible.



- 3) The JMP n instruction without corresponding JME n instruction (stand-alone JMP n) will cause program error. If only JME or JMP is inside of a loop (subroutine, FOR~NEXT block, or interrupt routine), an operation error will occur when the JMP instruction is enabled.

```

JMP 005 : Error (Stand-alone)
MOV
{
END
    
```

```

JMP 005 : Error when it is enabled
{
FOR 50
{
JME 005
NEXT
    
```

2.7.2 CALL , SBRT / RET

1) CALL n, CALLP n :

The CALL(P) instruction executes the subroutine program specified by the pointer 'n'. Multiple levels of nesting of the CALL(P) instruction are allowed.

2) SBRT / RET

SBRT instruction shows the start of subroutine program, and RET shows the end. Those two instructions should be in pairs.

LOAD	P000	
{		
SBRT	40	: Error (SBRT before END)
}		
END		
RET		: Error (Stand Alone)

LOAD	P042	
{		
CALL	30	: Error (No SBRT)
}		
END		

LOAD	P010	
{		
CALL	30	
}		
END		
SBRT	30	: Error (No RET)

2.7.3 MCS – MCSCLR

The MCS n instruction starts a master control sequence. Each MCS instructions are followed by a number (n) that shows the priority of the master control. The range of n is 0 ~ 7.

MCS	0 : High
{	↑
MCS	7 : LOW

The MCSCLR n instruction ends a master control sequence. If a MCSCLR instruction is executed, all master controls that have lower priority are cleared automatically.

MCS	0
MCS	1
{	
MCSCLR	0 : (MCS 1 is cleared automatically)
MCSCLR	1 : Error (Improper order of MCSCLR)

When use master control, it should start from the highest priority level and end from the lowest priority level. The MCS n and MCSCLR n instructions should be in pairs. Otherwise, a program error occurs.

2.7.4 FOR – NEXT

(Applicable for K200S / K300S / K1000S)

FOR and NEXT instructions should be in pairs, and each pairs should be start by FOR instructions. The maximum nesting level of FOR-NEXT block is 5.

If there is a stand-alone FOR or NEXT instruction or the nesting level exceeds 6, a program error occurs and CPU stops operation.

```

LOAD    P000
FOR      1      : No error (3 level nesting)
FOR      2
FOR      3
{
NEXT
NEXT
NEXT
}
END

```

```

LOAD    P001
{
FOR      20
{
NEXT
NEXT      : Error (Stand-alone NEXT)
}
END

```

```

LOAD    P002
FOR      20
{
END      :Error (No END instruction is permitted
{        between FOR and NEXT)
NEXT
END

```

2.7.5 END / RET

1) If there is no END in a sequence program, a program error occurs and CPU stops operation.

LOAD	P012	
	{	
JMP	10	
	{	
JME	10	
	{	

: Missing END

2) If there is no RET in a subroutine program, a program error occurs and CPU stops operation.

END		
SBRT		
	{	
LOAD	P000	
	{	
OUT	P010	
	{	

: Missing RET

2.7.6 Dual coil

If a memory device is used as an output of operation two or more times, a dual coil error occurs.

Because this is not a serious error, it does not make the CPU stop.

LOAD	P0000	
OUT	M000	
	{	
OUT	M000	: Dual coil error
SET	M000	: Dual coil error

2.8 Error handling

2.8.1 RUN / STOP at operation error

When an operation error (indirect addressing error, BCD operation error, etc) occurs, the CPU decide to continue operation or not based on parameter setting.

Refer the chapter 2.4.4 for details.

2.8.2 Error flags (F110 / F115)

If an error occurs while the CPU is running, 2 error flags (F110 and F115) are switched on.

The F110 is updated after each instruction is executed. However, the executed instruction is not related to any error (such as the LOAD instruction), it keeps the previous value. In other hand, the F115 keeps the on status after it was switched on at once. To clear the F115 flag, execute the CLE instruction. The following table shows an example of F110 and F115 operation.

Program	Error occurred?	F110	F115	Remarks
ADD D0 10 M20	No	OFF	OFF	
MOV D0 #D10	Yes	ON	ON	D10 = hFFFF
LOAD P0000	N/A	ON	ON	
INC D0	No	OFF	ON	
LOAD P0001	N/A	OFF	ON	
WAND P01 M10 #D400	Yes	ON	ON	D400 = hFF00
LOAD P0002	N/A	ON	ON	
WAND P01 M10 D300	No	OFF	ON	
CLE	N/A	OFF	OFF	Clear F115
LOAD P0003	N/A	OFF	OFF	
WAND P01 M10 D500	No	OFF	OFF	
BCD hFFFF D20	Yes	ON	ON	

2.8.3 LED indication**1) K10S1 / K10S / K30S / K60S**

LED Name	Operating Status	LED Indication
ERR	· Serious error	Flickering with 1sec period
	· Light error	
	· Program or parameter error	
RUN	· CPU is in RUN mode	Always ON
	· CPU is in Stop mode or an error occurred	Always OFF

2) K200S / K300S / K1000S

LED Name	Operating Status	LED Indication
STOP	· CPU is in Stop, Remote mode	Always ON
	· Serious error	Flickering with 200ms period
	· Light error	Flickering with 600ms period
	· Program or parameter error	Flickering with 1sec period
RUN	· CPU is in RUN mode	Always ON

2.8.4 Error code list

Error type	Message	Code (F006)	CPU	Description	Corrective action
Internal system error	System error	h0001	Stop	The operating system ROM or other H/W is defective.	Contact the nearest LG representative
OS ROM error	OS ROM error	h0002	Stop	The internal system ROM is defective.	Contact the nearest LG representative
OS RAM error	OS RAM error	h0003	Stop	The internal system RAM is defective.	Contact the nearest LG representative
Data RAM Error	Data RAM error	h0004	Stop	The RAM at which a data is stored is defective.	Contact the nearest LG representative
Program RAM error	Error	h0005	Stop	The RAM at which a program is stored is defective.	Contact the nearest LG representative
Gate array error	G/A error	h0006	Stop	The gate array of CPU is defective.	Contact the nearest LG representative
Sub rack power down error	Sub power error	h0007	Stop	The expansion rack is power off or defective.	Check the expansion rack is powered.
OS WDT time out error	OS WDT error	h0008	Stop	The CPU operation time (not the scan time) is too long.	Power off and restart the CPU. If error still occurs, contact the nearest LG representative
Shared RAM error	Common RAM error	h0009	Stop	Shared RAM interface error	Contact the nearest LG representative
Fuse break error	I/O fuse error	h000A	Run (Stop)	The fuse used in output module is blown.	Check the fuse and replace it.
Instruction code error	OP code error	h000B	Stop	CPU meets an instruction can not be decoded during executing program.	Contact the nearest LG representative
Flash Memory error	User memory error	H000C	Stop	CPU can not access the internal flash memory	Check the flash memory and replace it if necessary.
I/O slot error	I/O slot error	h0010	Stop	① Mount or dismount a module while the PLC is powered. A module is mounted improperly. ② I/O module or expansion cable is defective.	① Power off - Remount module – Power on ② Replace the I/O module or expansion cable
Maximum I/O over	Max I/O over	h0011	Stop	I/O points exceed the maximum limit points (Fmm mounting numbers over error, ...)	Replace I/O unit
Special Card I/F error	Special I/F error	h0012	Stop	Error occurred during special card interface	Contact the nearest LG representative
Fmm 0 I/F error	Fmm 0 I/F error	h0013	Stop	Fmm 0 I/F error	Contact the nearest LG representative
Fmm 1 I/F error	Fmm 1 I/F error	h0014	Stop	Fmm 1 I/F error	Contact the nearest LG representative
Fmm 2 I/F error	Fmm 2 I/F error	h0015	Stop	Fmm 2 I/F error	Contact the nearest LG representative
Fmm 3 I/F error	Fmm 3 I/F error	h0016	Stop	Fmm 3 I/F error	Contact the nearest LG representative
Parameter error	Parameter error	h0020	Stop	When the parameter is wrong or has incorrect checksum.	Change parameter setting

Error Code (Continued)

Error type	Message	Code (F006)	CPU	Description	Corrective action
I/O parameter error	I/O parameter error	h0021	Stop	When the CPU is powered on or turned to RUN mode, I/O modules are not mounted as I/O reservation of parameter setting.	Change parameter setting or re-arrange I/O modules
Maximum I/O error	I/O parameter error	h0022	Stop	I/O parameter setting value or actually mounted I/O points exceeds the maximum I/O points of CPU module.	Change parameter setting
Fmm 0 parameter error	Fmm 0 parameter error	h0023	Run	Fmm 0 parameter error	Change parameter setting
Fmm 1 parameter error	Fmm 1 parameter error	h0024	Run	Fmm 1 parameter error	Change parameter setting
Fmm 2 parameter error	Fmm 2 parameter error	h0025	Run	Fmm 2 parameter error	Change parameter setting
Fmm 3 parameter error	Fmm 3 parameter error	h0026	Run	Fmm 3 parameter error	Change parameter setting
Operation error	Operation error	h0030	Stop (Run)	<ul style="list-style-type: none"> · BCD operation error · Operand error 	Revise program
WDT error	WDT over error	h0031	Stop	The scan time exceeds the parameter setting value of watch dog timer.	Change parameter value or insert WDT instruction
Program change error	PGM Change error	h0032	Stop	An error occurred while editing program in RUN mode. (The change is not completed)	-
Program change error	PGM Change error	h0033	Run	A code error occurred while editing program in RUN mode.	-
Code check error	Code check error	h0040	Stop	There is an instruction cannot be decoded in program.	Revise program
Missing END instruction	Missing END instruction	h0041	Stop	There is no END instruction in program.	Insert END instruction at the end of program.
Missing RET error	Missing RET instruction	h0042	Stop	There is no RET instruction in subroutine.	Insert RET instruction at the end of subroutine
Missing SBRT error	Missing SBRT instruction	h0043	Stop	A subroutine is called with CALL instruction, but there is no corresponding subroutine.	Write subroutine.
JMP~JME instruction error	JMP/JME error	h0044	Stop	JMP~JME instructions are used improperly in program.	Revise program
FOR~NEXT instruction error	FOR~NEXT error	h0045	Stop	FOR~NEXT instructions are used improperly in program.	Revise program
MCS ~ MCSCLR error	MCS ~ MCSCLR error	h0046	Stop	MCS~MCSCLR instructions are used improperly in program.	Revise program
MPUSH ~ MPOP error	MPUSH ~ MPOP error	h0047	Stop	MPUSH~MPOP instruction are used improperly in program.	Revise program
Dual Coil error	Dual Coil error	h0048	Stop	A device is used as the output of operation more than one time.	Revise program
Syntax error	Syntax error	h0049	Stop	Wrong input conditions or too many LOAD instructions, etc.	Revise program
Battery error	Battery error	h0050	Run	The voltage of back-up battery is too low	Replace battery with new one.

Chapter 3 Instructions

3.1 Basic instructions	3-1
3.1.1 Contact instructions.....	3-1
3.1.2 Connection instructions	3-1
3.1.3 Inversion instruction	3-1
3.1.4 Master control instructions	3-2
3.1.5 Output instructions	3-2
3.1.6 Step controller instructions	3-2
3.1.7 END instruction	3-2
3.1.8 No operation instruction	3-3
3.1.9 Timer instructions	3-3
3.1.10 Counter instructions	3-4
3.2 Application instructions	3-5
3.2.1 Data transfer instructions	3-5
3.2.2 Conversion instructions.....	3-6
3.2.3 Compare instructions	3-6
3.2.4 Increment / Decrement instructions.....	3-9
3.2.5 Rotation instructions.....	3-9
3.2.6 Shift instructions.....	3-10
3.2.7 Exchange instructions	3-11
3.2.8 BIN arithmetic instructions	3-11
3.2.9 BCD arithmetic instructions	3-13
3.2.10 Logical operation instructions.....	3-14
3.2.11 Data processing instructions	3-15
3.2.12 System instructions	3-17
3.2.13 Branch instructions.....	3-17
3.2.14 Loop instructions	3-17

3.2.15 Flag instructions	3-18
3.2.16 Special module instructions	3-18
3.2.17 Data link instructions	3-18
3.2.18 Interrupt instructions.....	3-19
3.2.19 Sign inversion instructions	3-19
3.2.20 Bit contact instructions	3-20

3 Instructions

3.1 Basic instructions

3.1.1 Contact instructions

Mnemonic symbol	Function No.	Ladder symbol	Unit	Contents of processing	CPU	Page
LOAD	-		-	NO contact operation start	○	4- 1
LOAD NOT	-		-	NC contact operation start	○	4- 1
AND	-		-	NO contact series connection	○	4- 3
AND NOT	-		-	NC contact series connection	○	4- 3
OR	-		-	NO contact parallel connection	○	4- 4
OR NOT	-		-	NC contact parallel connection	○	4- 4

3.1.2 Connection instructions

Mnemonic symbol	Function No.	Ladder symbol	Unit	Contents of processing	CPU	Page
AND LOAD	-		-	Series connection of blocks	○	4- 6
OR LOAD	-		-	Parallel connection of blocks	○	4- 8
MPUSH	005		-	Stores the operation result	○	4- 10
MLOAD	006		-	Reads the operation result from MPUSH	○	4- 10
MPOP	007		-	Reads the operation result from MPUSH and clears the result	○	4- 10



3.1.3 Inversion instruction

Mnemonic symbol	Function No.	Ladder symbol	Unit	Contents of processing	CPU	Page
NOT	-		-	Invert the operation result	○	4- 12


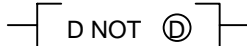



Remark

Applicable CPU type : ○ = All CPUs ; □ = K10S1 / K10S / K30S / K60S ; ★ = K200S / K300S / K1000S

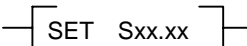
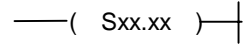
3.1.4 Master control instructions

Mnemonic symbol	Function No.	Ladder symbol	Unit	Contents of processing	CPU	Page
MCS	010		-	Start a master control	<input type="radio"/>	4– 13
MCSCLR	011		-	End a master control	<input type="radio"/>	4 – 13

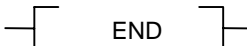
3.1.5 Output instructions

Mnemonic symbol	Function No.	Ladder symbol	Unit	Contents of processing	CPU	Page
D	017		-	Generates one scan pulse on the rising edge of input signal.	<input type="radio"/>	4– 16
D NOT	018		-	Generates one scan pulse on the falling edge of input signal.	<input type="radio"/>	4 – 18
SET	-		-	Set a device	<input type="radio"/>	4 – 19
RST	-		-	Reset a device	<input type="radio"/>	4 – 20
OUT	-		-	Output a device	<input type="radio"/>	

3.1.6 Step controller instructions

Mnemonic symbol	Function No.	Ladder symbol	Unit	Contents of processing	CPU	Page
SET S	-		-	Sequential processing control	<input type="radio"/>	4– 22
OUT S	-		-	Last-in priority control	<input type="radio"/>	4 – 24

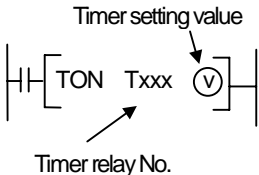
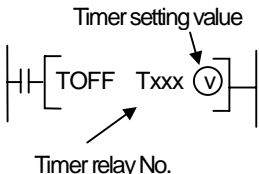
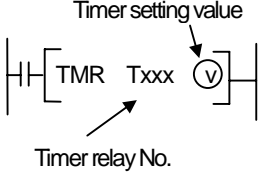
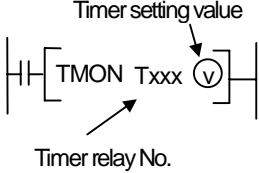
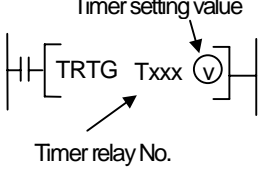
3.1.7 END instruction

Mnemonic symbol	Function No.	Ladder symbol	Unit	Contents of processing	CPU	Page
END	001		-	Ends a sequence program	<input type="radio"/>	4– 25

3.1.8 No operation instruction

Mnemonic symbol	Function No.	Ladder symbol	Unit	Contents of processing	CPU	Page
NOP	000	No ladder symbol	-	No operation (occupies 1 step)	○	4- 26

3.1.9 Timer instructions

Mnemonic symbol	Function No.	Ladder symbol	Unit	Contents of processing	CPU	Page
TON	-	 <p>Timer setting value</p> <p>Timer relay No.</p>	-	<p><On delay timer></p> <p>Input: [Pulse]</p> <p>Output: [Delay t, then pulse]</p> <p>t = setting value</p>	○	4- 27
TOFF	-	 <p>Timer setting value</p> <p>Timer relay No.</p>	-	<p><Off delay timer></p> <p>Input: [Pulse]</p> <p>Output: [Pulse, then delay t, then off]</p> <p>t = setting value</p>	○	4- 29
TMR	-	 <p>Timer setting value</p> <p>Timer relay No.</p>	-	<p><Accumulation timer></p> <p>Input: [Pulse 1, Pulse 2]</p> <p>Output: [Pulse 1, Pulse 2, then delay t]</p> <p>t = setting value (t = t1 + t2)</p>		4- 31
TMON	-	 <p>Timer setting value</p> <p>Timer relay No.</p>	-	<p><Monostable timer></p> <p>Input: [Pulse]</p> <p>Output: [Delay t, then pulse]</p> <p>t = setting value</p>		4- 33
TRTG	-	 <p>Timer setting value</p> <p>Timer relay No.</p>	-	<p><Retriggerable timer></p> <p>Input: [Pulse 1, Pulse 2, Pulse 3]</p> <p>Output: [Delay t, then pulse, then delay t, then pulse]</p> <p>t = setting value</p>		4- 35

3.1.10 Counter instructions

Mnemonic symbol	Function No.	Ladder symbol	Unit	Contents of processing	CPU	Page
CTU	-		-		○	4 – 37
CTD	-		-		○	4 – 38
CTUD	-		-		○	4 – 39
CTR	-		-		○	4 – 41

3.2 Application instructions

3.2.1 Data transfer instructions

Mnemonic symbol	Function No.	Ladder symbol	Unit	Contents of processing	CPU	Page
MOV	080		16 bits	Move data $[\text{S}] \longrightarrow [\text{D}]$	<input type="radio"/>	5-1
MOVP	081		16 bits			
DMOV	082		32 bits	Move data $[\text{S} + 1, \text{S}] \longrightarrow [\text{D} + 1, \text{D}]$	<input type="radio"/>	5-1
DMOVP	083		32 bits			
CMOV	084		16 bits	Complement data move $[\overline{\text{S}}] \longrightarrow [\text{D}]$	<input type="radio"/>	5-3
CMOVP	085		16 bits			
DCMOV	086		32 bits	Complement data move $[\overline{\text{S} + 1}, \overline{\text{S}}] \longrightarrow [\text{D} + 1, \text{D}]$	<input type="radio"/>	5-3
DCMOVP	087		32 bits			
GMOV	090		16 bits	Group move 	<input type="radio"/>	5-5
GMOVP	091		16 bits			
FMOV	092		16 bits	Filling move 	<input type="radio"/>	5-7
FMOVP	093		16 bits			
BMOV	100		n bit	Bit move	<input type="radio"/>	5-9
BMOVP	101		n bit	(See the 5-9 page for detail usage)		



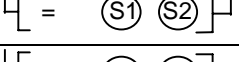


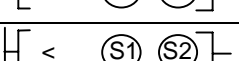




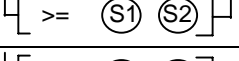


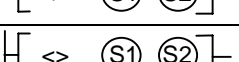
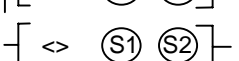



3.2.2 Conversion instructions

Mnemonic symbol	Function No.	Ladder symbol	Unit	Contents of processing	CPU	Page
BCD	060	$\text{[BCD } \textcircled{S} \textcircled{D}]$	16 bits	BCD conversion Binary $\textcircled{S} \longrightarrow \textcircled{D}$	<input type="radio"/>	5-11
BCDP	061	$\text{[BCDP } \textcircled{S} \textcircled{D}]$	16 bits	BCD conversion Binary $\textcircled{S} \longrightarrow \textcircled{D}$	<input type="radio"/>	5-11
DBCD	062	$\text{[DBCD } \textcircled{S} \textcircled{D}]$	32 bits	BCD conversion Binary $\textcircled{S} + 1, \textcircled{S} \longrightarrow \textcircled{D} + 1, \textcircled{D}$	<input type="radio"/>	5-11
DBCDP	063	$\text{[DBCDP } \textcircled{S} \textcircled{D}]$	32 bits	BCD conversion Binary $\textcircled{S} + 1, \textcircled{S} \longrightarrow \textcircled{D} + 1, \textcircled{D}$	<input type="radio"/>	5-11
BIN	064	$\text{[BIN } \textcircled{S} \textcircled{D}]$	16 bits	BIN conversion BCD $\textcircled{S} \longrightarrow \textcircled{D}$	<input type="radio"/>	5-14
BINP	065	$\text{[BINP } \textcircled{S} \textcircled{D}]$	16 bits	BIN conversion BCD $\textcircled{S} \longrightarrow \textcircled{D}$	<input type="radio"/>	5-14
DIND	066	$\text{[DBIN } \textcircled{S} \textcircled{D}]$	32 bits	BIN conversion BCD $\textcircled{S} + 1, \textcircled{S} \longrightarrow \textcircled{D} + 1, \textcircled{D}$	<input type="radio"/>	5-14
DBINP	067	$\text{[DBINP } \textcircled{S} \textcircled{D}]$	32 bits	BIN conversion BCD $\textcircled{S} + 1, \textcircled{S} \longrightarrow \textcircled{D} + 1, \textcircled{D}$	<input type="radio"/>	5-14

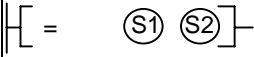

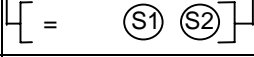
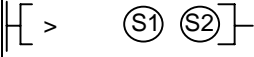
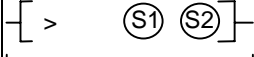
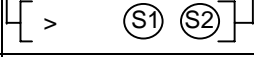
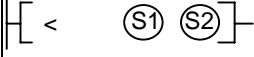
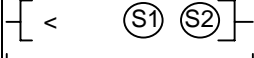
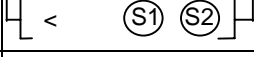

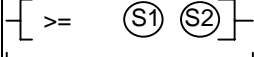
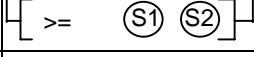
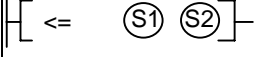
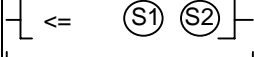
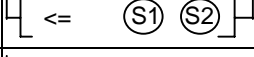
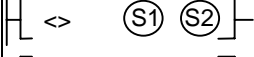
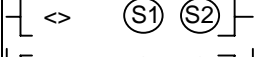
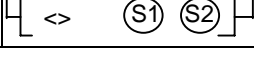
3.2.3 Compare instructions

Mnemonic symbol	Function No.	Ladder symbol	Unit	Contents of processing	CPU	Page
CMP	050	$\text{[CMP } \textcircled{S1} \textcircled{S2}]$	16 bits	Compare S1 and S2.	<input type="radio"/>	5-16
CMPP	051	$\text{[CMPP } \textcircled{S1} \textcircled{S2}]$	16 bits	Result is indicated by F120 ~ F125	<input type="radio"/>	5-16
DCMP	052	$\text{[DCMP } \textcircled{S1} \textcircled{S2}]$	32 bits	Compare [S1+1, S1] and [S2+1, S2]	<input type="radio"/>	5-16
DCMPP	053	$\text{[DCMPP } \textcircled{S1} \textcircled{S2}]$	32 bits	Result is indicated by F120 ~ F125	<input type="radio"/>	5-16
TCMP	054	$\text{[TCMP } \textcircled{S1} \textcircled{S2} \textcircled{D}]$	16 bits	Compare S1 and 16 words from S2	<input type="radio"/>	5-19
TCMPP	055	$\text{[TCMPP } \textcircled{S1} \textcircled{S2} \textcircled{D}]$	16 bits	Result(16bits) is stored at D	<input type="radio"/>	5-19
DTCMP	056	$\text{[DTCMP } \textcircled{S1} \textcircled{S2} \textcircled{D}]$	32 bits	Compare [S1+1, S1] and 32 words from S2	<input type="radio"/>	5-19
DTCMPP	057	$\text{[DTCMPP } \textcircled{S1} \textcircled{S2} \textcircled{D}]$	32 bits	Result(32 bits) is stored at [D+1, D]	<input type="radio"/>	5-19

Comparison instructions (Continued)

Mnemonic symbol	Function No.	Ladder symbol	Unit	Contents of processing	CPU	Page
LOAD=	028		16 bits	The input condition is switched on when [S1] = [S2]	★	5-21
AND=	094					5-22
OR=	188					5-23
LOAD>	038		16 bits	The input condition is switched on when [S1] > [S2] (Signed comparison)	★	5-21
AND>	096					5-22
OR>	196					5-23
LOAD<	048		16 bits	The input condition is switched on when [S1] < [S2] (Signed comparison)	★	5-21
AND<	098					5-22
OR<	198					5-23
LOAD>=	058		16 bits	The input condition is switched on when [S1] >= [S2] (Signed comparison)	★	5-21
AND>=	106					5-22
OR>=	216					5-23
LOAD<=	068		16 bits	The input condition is switched on when [S1] <= [S2] (Signed comparison)	★	5-21
AND<=	108					5-22
OR<=	218					5-23
LOAD<>	078		16 bits	The input condition is switched on when [S1] <> [S2] (Not equal)	★	5-21
AND<>	118					5-22
OR<>	228					5-23

Comparison instructions (Continued)

Mnemonic symbol	Function No.	Ladder symbol	Unit	Contents of processing	CPU	Page
LOADD=	029		32 bits	The input condition is switched on when [S1+1, S1] = [S2+1, S2]	★	5-21
ANDD=	095					5-22
ORD=	189					5-23
LOADD>	039		32 bits	The input condition is switched on when [S1+1, S1] > [S2+1, S2] (Signed comparison)	★	5-21
ANDD>	097					5-22
ORD>	197					5-23
LOADD<	049		32 bits	The input condition is switched on when [S1+1, S1] < [S2+1, S2] (Signed comparison)	★	5-21
ANDD<	099					5-22
ORD<	199					5-23
LOADD>=	059		32 bits	The input condition is switched on when [S1+1, S1] >= [S2+1, S2] (Signed comparison)	★	5-21
ANDD>=	107					5-22
ORD>=	217					5-23
LOADD<=	069		32 bits	The input condition is switched on when [S1+1, S1] <= [S2+1, S2] (Signed comparison)	★	5-21
ANDD<=	109					5-22
ORD<=	219					5-23
LOADD<>	079		32 bits	The input condition is switched on when [S1+1, S1] <> [S2+1, S2]	★	5-21
ANDD<>	119					5-22
ORD<>	229					5-23

3.2.4 Increment / Decrement instructions

Mnemonic symbol	Function No.	Ladder symbol	Unit	Contents of processing	CPU	Page
INC	020		16 bits	Increment	<input type="radio"/>	5-24
INCP	021		16 bits	$[\textcircled{D}] + 1 \longrightarrow [\textcircled{D}]$	<input type="radio"/>	5-24
DINC	022		32 bits	Increment	<input type="radio"/>	5-24
DINCP	023		32 bits	$[\textcircled{D} + 1, \textcircled{D}] + 1 \longrightarrow [\textcircled{D} + 1, \textcircled{D}]$	<input type="radio"/>	5-24
DEC	024		16 bits	Decrement	<input type="radio"/>	5-26
DECP	025		16 bits	$[\textcircled{D}] - 1 \longrightarrow [\textcircled{D}]$	<input type="radio"/>	5-26
DDEC	026		32 bits	Decrement	<input type="radio"/>	5-26
DDECP	027		32 bits	$[\textcircled{D} + 1, \textcircled{D}] - 1 \longrightarrow [\textcircled{D} + 1, \textcircled{D}]$	<input type="radio"/>	5-26

3.2.5 Rotation instructions

Mnemonic symbol	Function No.	Ladder symbol	Unit	Contents of processing	CPU	Page
ROL	020		16 bits	 1 bit rotate to left	<input type="radio"/>	5-28
ROLP	021		16 bits			
ROR	034		16 bits	 1 bit rotate to right		5-30
RORP	035		16 bits			
RCL	040		16 bits	 1 bit rotate to left include carry		5-32
RCLP	041		16 bits			
RCR	044		16 bits	 1 bit rotate to right include carry		5-34
RCRP	045		16 bits			

Rotation instructions (Continued)

Mnemonic symbol	Function No.	Ladder symbol	Unit	Contents of processing	CPU	Page
DROL	022		32 bits		○	5-28
DROL P	023		32 bits	1 bit rotate to left		
DROR	036		32 bits		○	5-30
DROR P	037		32 bits	1 bit rotate to right		
DRCL	042		32 bits		○	5-32
DRCL P	043		32 bits	1 bit rotate to left include carry		
DRCR	046		32 bits		○	5-34
DRCR P	047		32 bits	1 bit rotate to right include carry		

3.2.6 Shift instructions

Mnemonic symbol	Function No.	Ladder symbol	Unit	Contents of processing	CPU	Page
BSFT	074		S1-S2 bits		○	5-36
BSFT P	075		S1-S2 bits	1 bit shift from S1 to S2		
WSFT	070		S1-S2 words		○	5-38
WSFT P	071		S1-S2 words	1 word shift from S1 to S2		
SR			16 bits			5-40
				Bit shift (See 4. For details)		

3.2.7 Exchange instructions

Mnemonic symbol	Function No.	Ladder symbol	Unit	Contents of processing	CPU	Page
XCH	102		16 bits	$[D1] \longleftrightarrow [D2]$	<input type="radio"/>	5-42
XCHP	103		16 bits	$[D1] \longleftrightarrow [D2]$	<input type="radio"/>	5-42
DXCH	104		32 bits	$[D1+1, D1] \longleftrightarrow [D2+1, D2]$	<input type="radio"/>	5-42
DXCHP	105		32 bits	$[D1+1, D1] \longleftrightarrow [D2+1, D2]$	<input type="radio"/>	5-42

3.2.8 BIN arithmetic instructions

Mnemonic symbol	Function No.	Ladder symbol	Unit	Contents of processing	CPU	Page
ADD	110		16 bits	$[S1] + [S2] \longrightarrow [D]$	<input type="radio"/>	5-44
ADDP	111		16 bits	$[S1] + [S2] \longrightarrow [D]$	<input type="radio"/>	5-44
DADD	112		32 bits	$[S1+1, S1] + [S2+1, S2]$	<input type="radio"/>	5-44
DADDP	113		32 bits	$\longrightarrow [D+1, D]$	<input type="radio"/>	5-44
SUB	114		16 bits	$[S1] - [S2] \longrightarrow [D]$	<input type="radio"/>	5-46
SUBP	115		16 bits	$[S1] - [S2] \longrightarrow [D]$	<input type="radio"/>	5-46
DSUB	116		32 bits	$[S1+1, S1] - [S2+1, S2]$	<input type="radio"/>	5-46
DSUBP	117		32 bits	$\longrightarrow [D+1, D]$	<input type="radio"/>	5-46
MUL	120		16 bits	$[S1] \times [S2] \longrightarrow [D+1, D]$	<input type="radio"/>	5-48
MULP	121		16 bits	$[D+1] : \text{High word}, [D] : \text{Low word}$	<input type="radio"/>	5-48
DMUL	122		32 bits	$[S1+1, S1] \times [S2+1, S2]$	<input type="radio"/>	5-48
DMULP	123		32 bits	$\longrightarrow [D+3, D+2, D+1, D]$ $[D+3, D+2] = \text{Higher 2 words}$ $[D+1, D] = \text{Lower 2 words}$	<input type="radio"/>	5-48

BIN arithmetic instructions (Continued)

Mnemonic symbol	Function No.	Ladder symbol	Unit	Contents of processing	CPU	Page
DIV DIVP	124 125	$\left[\text{DIV} \begin{array}{c} \textcircled{S1} \textcircled{S2} \textcircled{D} \end{array} \right]$ $\left[\text{DIVP} \begin{array}{c} \textcircled{S1} \textcircled{S2} \textcircled{D} \end{array} \right]$	16 bits	$[S1] \div [S2] \longrightarrow [D]$ [D+1] = Remainder [D] = Quotient	○	5-50
DDIV DDIVP	126 127	$\left[\text{DDIV} \begin{array}{c} \textcircled{S1} \textcircled{S2} \textcircled{D} \end{array} \right]$ $\left[\text{DDIVP} \begin{array}{c} \textcircled{S1} \textcircled{S2} \textcircled{D} \end{array} \right]$	32 bits	$[S1+1, S1] \div [S2+1, S2]$ $\longrightarrow [D+3, D+2, D+1, D]$ [D+3, D+2] = Remainder [D+1, D] = Quotient	○	5-50
MULS MULSP	072 073	$\left[\text{MULS} \begin{array}{c} \textcircled{S1} \textcircled{S2} \textcircled{D} \end{array} \right]$ $\left[\text{MULSP} \begin{array}{c} \textcircled{S1} \textcircled{S2} \textcircled{D} \end{array} \right]$	16 bits	Signed multiplying operation $[S1] \times [S2] \longrightarrow [D+1, D]$ [D+1] : High word, [D] : Low word	○	
DMULS DMULSP	076 077	$\left[\text{DMULS} \begin{array}{c} \textcircled{S1} \textcircled{S2} \textcircled{D} \end{array} \right]$ $\left[\text{DMULSP} \begin{array}{c} \textcircled{S1} \textcircled{S2} \textcircled{D} \end{array} \right]$	32 bits	Signed multiplying operation $[S1+1, S1] \times [S2+1, S2]$ $\longrightarrow [D+3, D+2, D+1, D]$ [D+3, D+2] = Higher 2 words [D+1, D] = Lower 2 words	○	
DIVS DIVSP	088 089	$\left[\text{DIVS} \begin{array}{c} \textcircled{S1} \textcircled{S2} \textcircled{D} \end{array} \right]$ $\left[\text{DIVSP} \begin{array}{c} \textcircled{S1} \textcircled{S2} \textcircled{D} \end{array} \right]$	16 bits	Signed dividing operation $[S1] \div [S2] \longrightarrow [D]$ [D+1] = Remainder [D] = Quotient	○	
DDIVS DDIVSP	128 129	$\left[\text{DDIVS} \begin{array}{c} \textcircled{S1} \textcircled{S2} \textcircled{D} \end{array} \right]$ $\left[\text{DDIVSP} \begin{array}{c} \textcircled{S1} \textcircled{S2} \textcircled{D} \end{array} \right]$	32 bits	Signed dividing operation $[S1+1, S1] \div [S2+1, S2]$ $\longrightarrow [D+3, D+2, D+1, D]$ [D+3, D+2] = Remainder [D+1, D] = Quotient	○	

3.2.9 BCD arithmetic instructions

Mnemonic symbol	Function No.	Ladder symbol	Unit	Contents of processing	CPU	Page
ADDB	130	$\left[\text{ADDB } \textcircled{S1} \textcircled{S2} \textcircled{D} \right]$	16 bits	BCD addition $[S1] + [S2] \longrightarrow [D]$	○	5-52
ADDBP	131	$\left[\text{ADDBP } \textcircled{S1} \textcircled{S2} \textcircled{D} \right]$				
DADDB	132	$\left[\text{DADDB } \textcircled{S1} \textcircled{S2} \textcircled{D} \right]$	32 bits	BCD addition $[S1+1, S1] + [S2+1, S2] \longrightarrow [D+1, D]$	○	5-52
DADDBP	133	$\left[\text{DADDBP } \textcircled{S1} \textcircled{S2} \textcircled{D} \right]$				
SUBB	134	$\left[\text{SUBB } \textcircled{S1} \textcircled{S2} \textcircled{D} \right]$	16 bits	BCD subtraction $[S1] - [S2] \longrightarrow [D]$	○	5-54
SUBBP	135	$\left[\text{SUBBP } \textcircled{S1} \textcircled{S2} \textcircled{D} \right]$				
DSUBB	136	$\left[\text{DSUBB } \textcircled{S1} \textcircled{S2} \textcircled{D} \right]$	32 bits	BCD subtraction $[S1+1, S1] - [S2+1, S2] \longrightarrow [D+1, D]$	○	5-54
DSUBBP	137	$\left[\text{DSUBBP } \textcircled{S1} \textcircled{S2} \textcircled{D} \right]$				
MULB	140	$\left[\text{MULB } \textcircled{S1} \textcircled{S2} \textcircled{D} \right]$	16 bits	BCD multiplication $[S1] \times [S2] \longrightarrow [D+1, D]$ [D+1] : High word, [D] : Low word	○	5-56
MULBP	141	$\left[\text{MULBP } \textcircled{S1} \textcircled{S2} \textcircled{D} \right]$				
DMULB	142	$\left[\text{DMULB } \textcircled{S1} \textcircled{S2} \textcircled{D} \right]$	32 bits	BCD multiplication $[S1+1, S1] \times [S2+1, S2]$ $\longrightarrow [D+3, D+2, D+1, D]$ [D+3, D+2] = Higher 2 words [D+1, D] = Lower 2 words	○	5-56
DMULBP	143	$\left[\text{DMULBP } \textcircled{S1} \textcircled{S2} \textcircled{D} \right]$				
DIVB	144	$\left[\text{DIVB } \textcircled{S1} \textcircled{S2} \textcircled{D} \right]$	16 bits	BCD division $[S1] \div [S2] \longrightarrow [D]$ [D+1] = Remainder [D] = Quotient	○	5-58
DIVBP	145	$\left[\text{DIVBP } \textcircled{S1} \textcircled{S2} \textcircled{D} \right]$				
DDIVB	146	$\left[\text{DDIVB } \textcircled{S1} \textcircled{S2} \textcircled{D} \right]$	32 bits	BCD division $[S1+1, S1] \div [S2+1, S2]$ $\longrightarrow [D+3, D+2, D+1, D]$ [D+3, D+2] = Remainder [D+1, D] = Quotient	○	5-58
DDIVBP	147	$\left[\text{DDIVBP } \textcircled{S1} \textcircled{S2} \textcircled{D} \right]$				

3.2.10 Logical operation instructions

Mnemonic symbol	Function No.	Ladder symbol	Unit	Contents of processing	CPU	Page
WAND	130		16 bits	[S1] AND [S2] [D]	○	5-60
WANDP	131		16 bits	[S1] AND [S2] [D]	○	5-60
DWAND	132		32 bits	[S1+1,S1] AND [S2+1,S2]	○	5-60
DWANDP	133		32 bits	→ [D+1,D]	○	5-60
WOR	154		16 bits	[S1] OR [S2] → [D]	○	5-62
WORP	155		16 bits	[S1] OR [S2] → [D]	○	5-62
DWOR	156		32 bits	[S1+1,S1] OR [S2+1,S2]	○	5-62
DWORP	157		32 bits	→ [D+1,D]	○	5-62
WXOR	160		16 bits	[S1] XOR [S2] → [D]	○	5-64
WXORP	161		16 bits	[S1] XOR [S2] → [D]	○	5-64
DWXOR	162		32 bits	[S1+1,S1] XOR [S2+1,S2]	○	5-64
DWXORP	163		32 bits	→ [D+1,D]	○	5-64
WXNR	164			[S1] XNR [S2] → [D]		5-66
WXNRP	165			[S1] XNR [S2] → [D]		5-66
DWXNR	166			[S1+1,S1] XNR [S2+1,S2]		5-66
DWXNRP	167			→ [D+1,D]		5-66



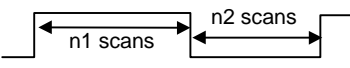


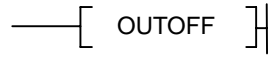

3.2.11 Data processing instructions

Mnemonic symbol	Function No.	Ladder symbol	Unit	Contents of processing	CPU	Page
SEG	174	[SEG (S) (D) Cw]	16 bits	7 Segment decode	<input type="radio"/>	5-68
SEGP	175	[SEGP (S) (D) Cw]		[S] $\xrightarrow{\text{decoding}}$ [D]		
ASC	190	[ASC (S) (D) Cw]		Converts the data of [S] into ASCII code format and stores it at [D].	<input type="radio"/>	5-71
ASCP	191	[ASCP (S) (D) Cw]				
BSUM	170	[DBIN (S) (D)]	16 bits		<input type="radio"/>	5-73
BSUMP	171	[DBINP (S) (D)]				
DBSUM	172	[DBIN (S) (D)]	32 bits		<input type="radio"/>	5-73
DBSUMP	173	[DBINP (S) (D)]				
ENCO	176	[ENCO (S) (D) n]	2 ⁿ bits		<input type="radio"/>	5-75
ENCOP	177	[ENCOP (S) (D) n]				
DECO	178	[DECO (S) (D) n]	n bits		<input type="radio"/>	5-77
DECOP	179	[DECOP (S) (D) n]				
FILR	180	[FILR (S) (D) n]	16 bits	<p>File table read</p>	<input type="radio"/>	5-79
FILRP	181	[FILRP (S) (D) n]				
DFILR	182	[DFILR (S) (D) n]	32 bits	<p>File table read</p>	<input type="radio"/>	5-79
DFILRP	183	[DFILRP (S) (D) n]				



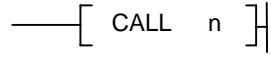

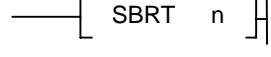

Data processing instructions (Continued)

Mnemonic symbol	Function No.	Ladder symbol	Unit	Contents of processing	CPU	Page
FILW FILWP	184 185	$\left[\text{FILW} \quad \textcircled{S} \quad \textcircled{D} \quad n \right]$ $\left[\text{FILWP} \quad \textcircled{S} \quad \textcircled{D} \quad n \right]$	16 bits	File table write 	○	5-81
DFILW DFILWP	186 187	$\left[\text{DFILW} \quad \textcircled{S} \quad \textcircled{D} \quad n \right]$ $\left[\text{DFILWP} \quad \textcircled{S} \quad \textcircled{D} \quad n \right]$	32 bits	File table write 	○	5-81
DIS DISP	194 195	$\left[\text{DIS} \quad \textcircled{S} \quad \textcircled{D} \quad n \right]$ $\left[\text{DISP} \quad \textcircled{S} \quad \textcircled{D} \quad n \right]$	16 bits		○	5-83
UNI UNIP	192 193	$\left[\text{UNI} \quad \textcircled{S} \quad \textcircled{D} \quad n \right]$ $\left[\text{UNIP} \quad \textcircled{S} \quad \textcircled{D} \quad n \right]$	32 bits		○	5-85
IORF IORFP	200 201	$\left[\text{IORF} \quad \textcircled{D1} \quad \textcircled{D2} \right]$ $\left[\text{IORFP} \quad \textcircled{D1} \quad \textcircled{D2} \right]$	16 bits	Refreshes the memory area from [D1] to [D2] ([D1] < [D2])	★	5-87




3.2.12 System instructions

Mnemonic symbol	Function No.	Ladder symbol	Unit	Contents of processing	CPU	Page
FALS	204			Stores n to specified F area	★	5-89
DUTY	205			Generate a clock pulse as below 	○	5-90
WDT	202			Clear watch dog timer	★	5-92
WDTP	203					
OUTOFF	208			Switch off all outputs	○	5-94
STOP	008			Stop the operation of CPU	★	5-95

3.2.13 Branch instructions

Mnemonic symbol	Function No.	Ladder symbol	Unit	Contents of processing	CPU	Page
JMP	012			Jump	○	5-96
JME	013			Jump end		
CALL	014			Call the subroutine	○	5-98
CALLP	015					
SBRT	016			Start of a subroutine	○	
RET	004			End of subroutine		

3.2.14 Loop instructions

Mnemonic symbol	Function No.	Ladder symbol	Unit	Contents of processing	CPU	Page
FOR	206			Executes the sequence program between FOR and NEXT n times	★	5-100
NEXT	207					
BREAK	220			Escape from FOR/NEXT loop	★	5-101

3.2.15 Flag instructions

Mnemonic symbol	Function No.	Ladder symbol	Unit	Contents of processing	CPU	Page
STC	002			Set the carry flag	○	5-102
CLC	003			Clear the carry flag		
CLE	009			Clear the error flag	★	5-103

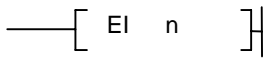

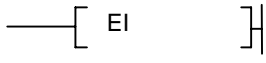
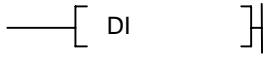


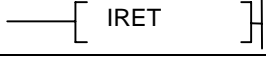
3.2.16 Special module instructions

Mnemonic symbol	Function No.	Ladder symbol	Unit	Contents of processing	CPU	Page
GET	230			Read data from shared RAM of a special module	★	5-104
GETP	231					
PUT	234			Write data to shared RAM of a special module	★	5-106
PUTP	235					

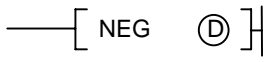



3.2.17 Data link instructions

Mnemonic symbol	Function No.	Ladder symbol	Unit	Contents of processing	CPU	Page
READ	244			Read / Write data of remote station	★	5-108
WRITE	245					5-111
RGET	232			Read / Write data of shared RAM of remote special module	★	5-113
RPUT	233					5-116
CONN	246			Establish a communication channel	★	
STATUS	247			Read the information of remote station	★	5-118







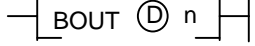
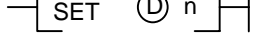
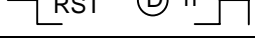
3.2.18 Interrupt instructions

Mnemonic symbol	Function No.	Ladder symbol	Unit	Contents of processing	CPU	Page
EI	236			Enable an interrupt	★	5-119
DI	239			Disable an interrupt		
EI	221			Enable all interrupts	★	5-119
DI	222			Disable all interrupts		
TDINT	226			Start of TDI routine	★	5-120
INT	227			Start of PDI routine		5-121
IRET	225			End of interrupt routine		

3.2.19 Sign inversion instructions

Mnemonic symbol	Function No.	Ladder symbol	Unit	Contents of processing	CPU	Page
NEG	240			Invert the sign of [D]	★	5-122
NEGP	241					
DNEG	242			Invert the sign of [D+1, D]	★	5-122
DNEGP	243					

3.2.20 Bit contact instructions

Mnemonic symbol	Function No.	Ladder symbol	Unit	Contents of processing	CPU	Page
BLD	248		-	NO contact operation start with the n th bit of [D]	★	5-124
BLDN	249			NC contact operation start with the n th bit of [D]	★	5-124
BAND	250			NO contact series connection with the n th bit of [D]	★	5-125
BANDN	251			NC contact series connection with the n th bit of [D]	★	5-125
BOR	252			NO contact parallel connection with the n th bit of [D]	★	5-126
BORN	253			NC contact parallel connection with the n th bit of [D]	★	5-126
BOUT	236			Output the result of operation to the n th bit of [D]	★	5-127
BSET	223			Set the n th bit of [D]	★	5-128
BRST	224			Clear the n th bit of [D]	★	5-128

Chapter 4 Basic instructions

4.1	Contact instructions	4-1
4.1.1	LOAD, LOAD NOT, OUT	4-1
4.1.2	AND, AND NOT	4-3
4.1.3	OR, OR NOT	4-4
4.2	Connection instructions	4-6
4.2.1	AND LOAD	4-6
4.2.2	OR LOAD	4-8
4.2.3	MPUSH, MLOAD, MPOP	4-10
4.3	Inversion instruction	4-12
4.3.1	NOT	4-12
4.4	Master control instructions	4-13
4.4.1	MCS, MCSCLR	4-13
4.5	Output instructions	4-16
4.5.1	OUT	4-16
4.5.2	D	4-17
4.5.3	D NOT	4-19
4.5.4	SET	4-20
4.5.5	RST	4-21
4.6	Step controller instructions	4-23
4.6.1	SET Sxx.xx	4-23
4.6.2	OUT Sxx.xx	4-25
4.7	End instructions	4-26
4.7.1	END	4-26
4.8	No operation instruction	4-27
4.8.1	NOP	4-27
4.9	Timer instructions	4-28
4.9.1	TON	4-28
4.9.2	TOFF	4-30

4.9.3	TMR	4-32
4.9.4	TMON.....	4-34
4.9.5	TRTG	4-36
4.10	Counter instructions	4-38
4.10.1	CTU.....	4-38
4.10.2	CTD.....	4-39
4.10.3	CTUD	4-40
4.10.4	CTR.....	4-42

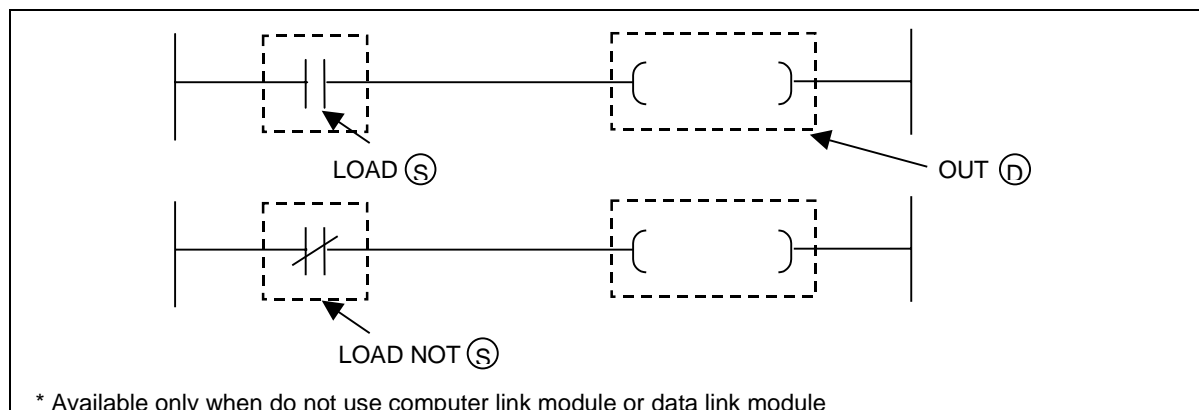
4 Basic instructions

4.1 Contact instructions

4.1.1 LOAD, LOAD NOT, OUT

LOAD	
LOAD NOT	
OUT	

Instructions		Available Device											Steps	Flag		
		M	P	K	L	F	T	C	S	D	#D	Integer		Error (F110)	Zero (F111)	Carry (F112)
LOAD	(S)	O	O	O	O	O	O	O	O				1			
LOAD NOT	(S)	O	O	O	O	O	O	O	O							
OUT	(D)	O	O	O	O*				O							



1) LOAD (S)

a) Functions

- Starts a NO contact.
- Draw the on/off data of the specified device ((S)) and use the data as an operation result.

2) LOAD NOT (S)

a) Functions

- Starts a NC contact.
- Draw the on/off data of the specified device ((S)) and use the data as an operation result.

3) OUT (D)

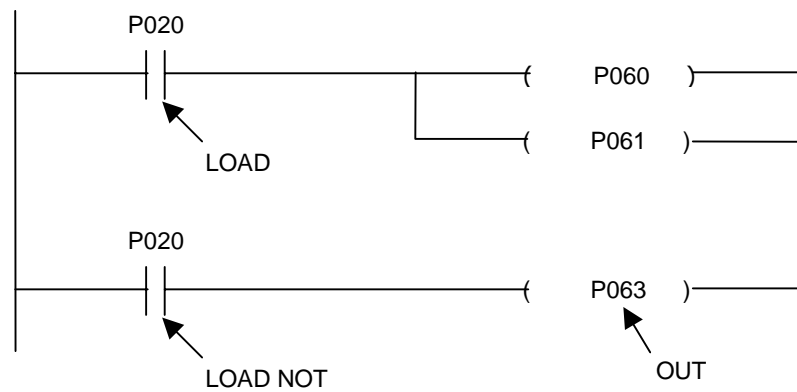
a) Functions

- Outputs the operation result to the specified device ((D)).
- Several OUT instructions can be used in parallel with one operation result.

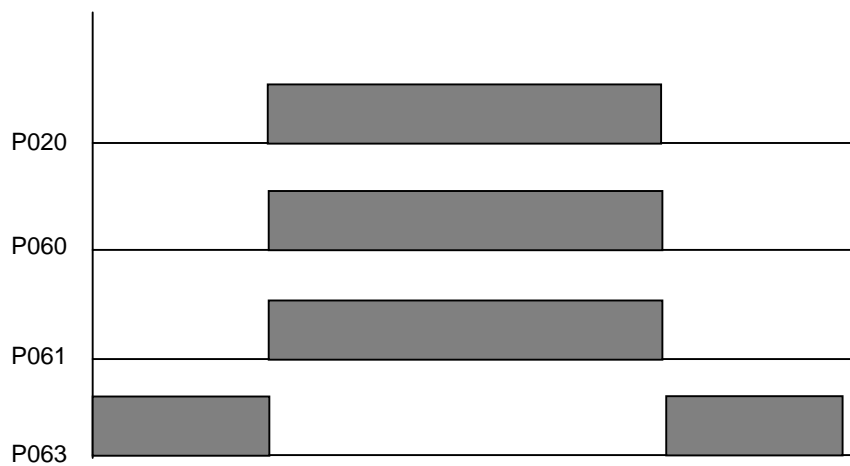
4) Program example

- When the input condition (P020) is switched on, the P060 and P061 will be switched on and the P062 will be switched off.

[Program]



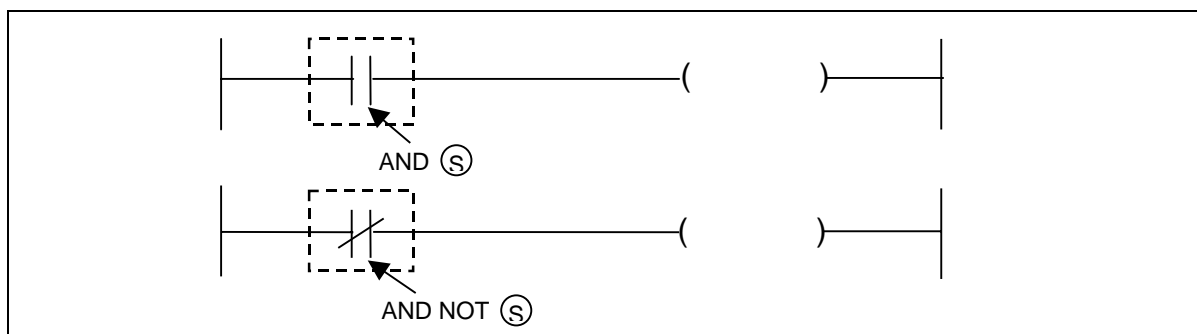
[Time chart]



4.1.2 AND, AND NOT

AND	
AND NOT	

Instructions		Available Device											Steps	Flag		
		M	P	K	L	F	T	C	S	D	#D	Integer		Error (F110)	Zero (F111)	Carry (F112)
AND AND NOT	(S)	O	O	O	O	O	O	O	O				1			



1) AND

a) Functions

- The NO contact series connection
- Read the on/off data of the specified device ((S)), performs the AND operation of that data and the previous operation result, and use it as a new operation result.

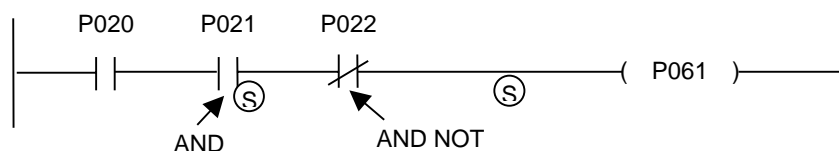
2) AND NOT

a) Functions

- The NC contact series connection
- Read the on/off data of the specified device ((S)), performs the AND operation of that data and the previous operation result, and use it as a new operation result.

3) Program example

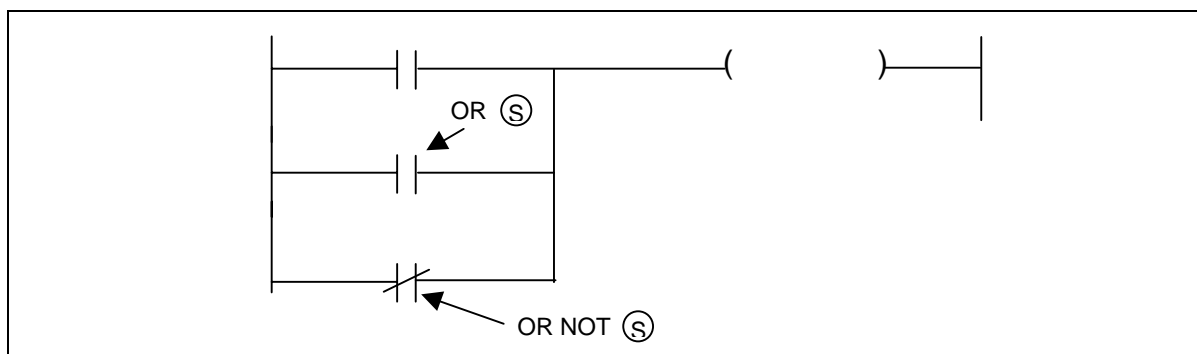
The P061 contact will be switched on when the P020 and P021 are on and the P022 is off.



4.1.3 OR, OR NOT

OR	
OR NOT	

Instructions		Available Device											Steps	Flag		
		M	P	K	L	F	T	C	S	D	#D	Integer		Error (F110)	Zero (F111)	Carry (F112)
OR OR NOT	(S)	O	O	O	O	O	O	O	O				1			



1) OR

a) Functions

- The NO contact parallel connection
- Draw the on/off data of the specified device ((S)), performs the OR operation of that data and the previous operation result, and use it as a new operation result.

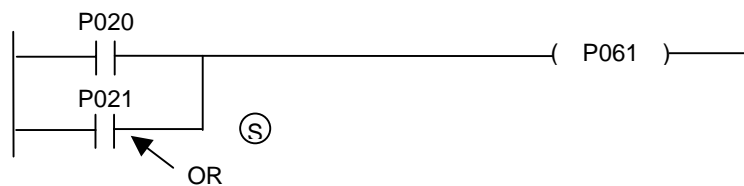
2) OR NOT

a) Functions

- The NC contact parallel connection
- Draw the on/off data of the specified device ((S)), performs the OR operation of that data and the previous operation result, and use it as a new operation result.

3) Program example

The P061 contact will be switched on when one of P020 and P021 is on.

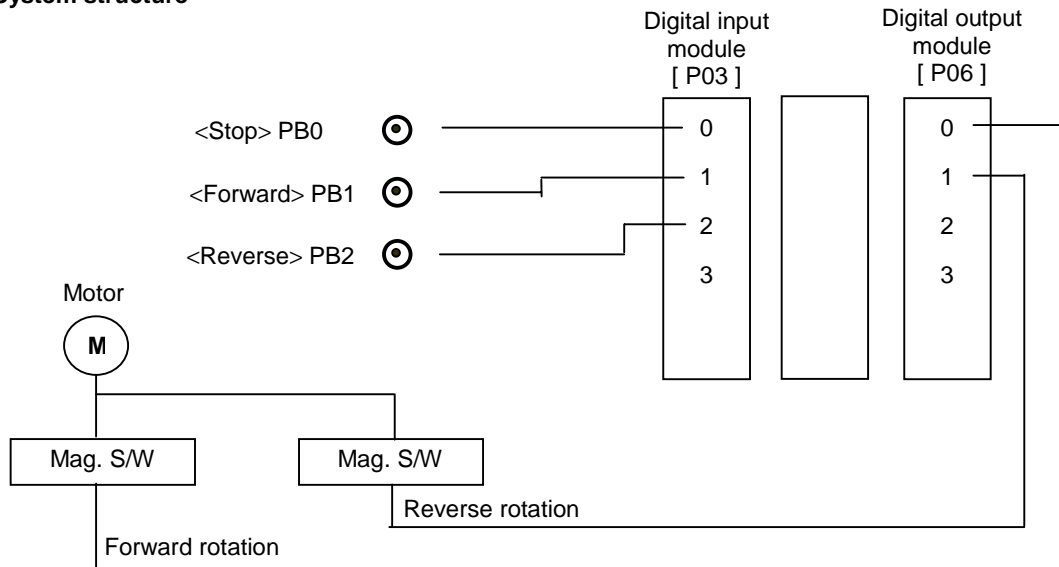


The motor operation (Example of LOAD, AND, OR, OUT instructions)

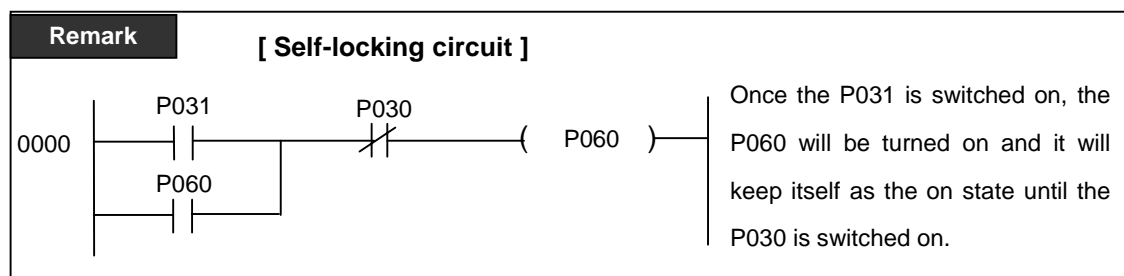
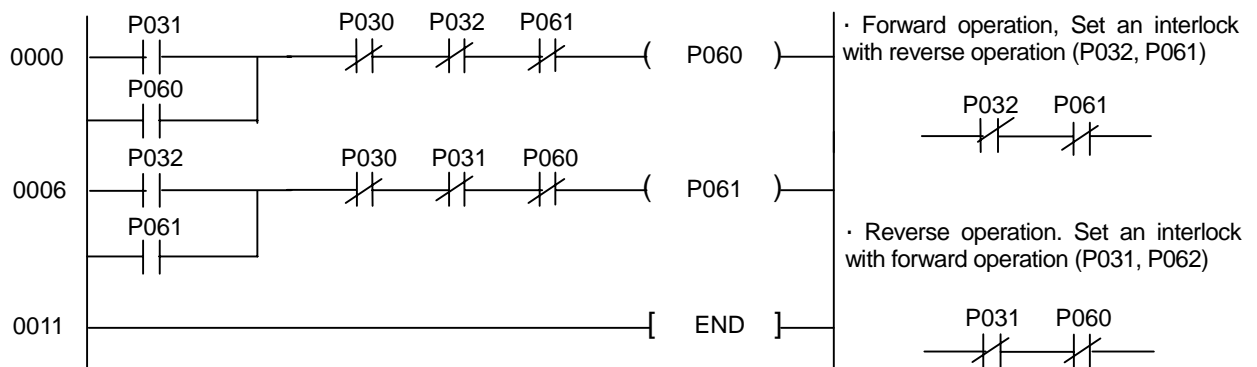
1. Operation

There are three push-button switches - PB0, PB1, and PB2. When PB1 is pushed, a motor will start to rotate with a forward (clockwise) direction. It will start to rotate with a reverse (counterclockwise) direction when the PB2 is pushed. The PB0 is emergency stop switch and the motor will stop operation when the PB0 is pushed.

2. System structure



3. Program

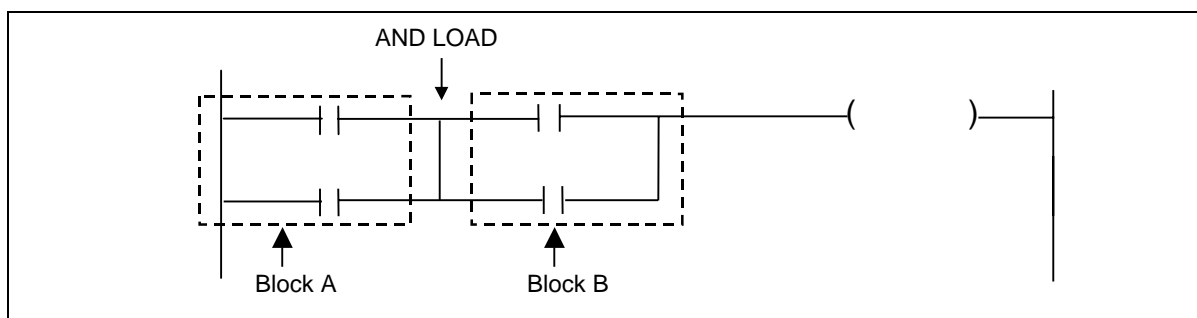


4.2 Connection instructions

4.2.1 AND LOAD

AND LOAD	
----------	--

Instructions	Available Device											Steps	Flag		
	M	P	K	L	F	T	C	S	D	#D	Integer		Error (F110)	Zero (F111)	Carry (F112)
AND LOAD												1			

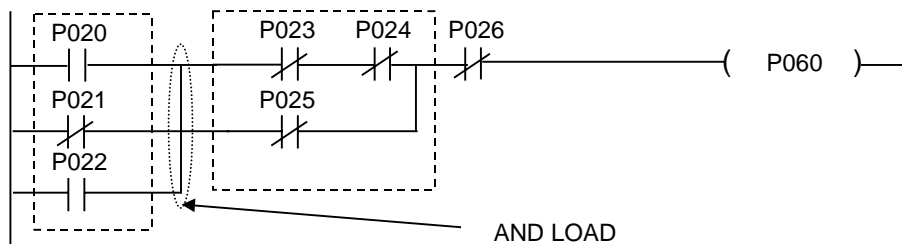


1) Functions

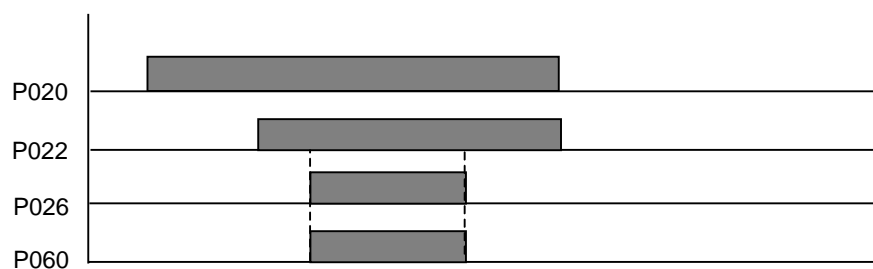
- Performs the AND operation of block A and block B, and uses it as an operation result.
- The AND LOAD instruction can be written consecutively up to 7 times.

2) Program example

[Program]



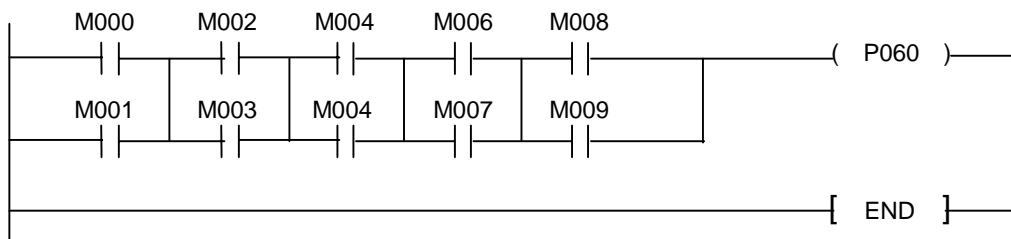
[Time chart]



[NOTE] The consecutive use of the AND LOAD instruction

There are two methods to connect several blocks in serial. See the following example.

[Ladder program]



[Mnemonic program]

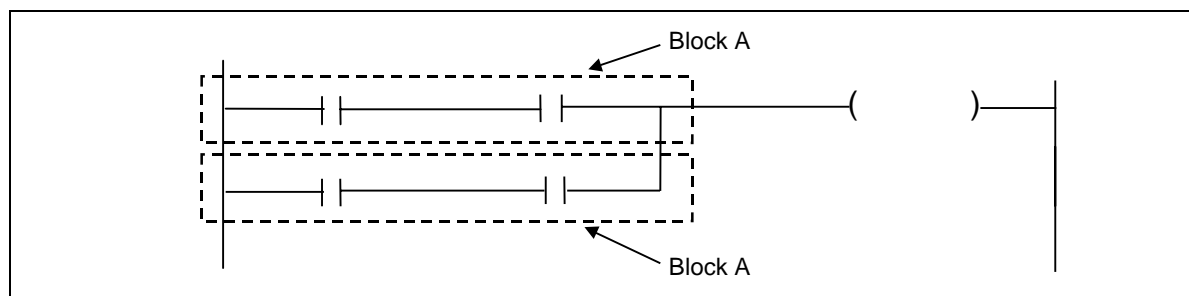
[A] Don't use AND LOAD instructions consecutively		[B] Use AND LOAD instructions consecutively	
LOAD	M000	LOAD	M000
OR	M001	OR	M001
LOAD	M002	LOAD	M002
OR	M003	OR	M003
AND LOAD		LOAD	M004
LOAD	M004	OR	M005
OR	M005	LOAD	M006
AND LOAD		OR	M007
LOAD	M006	LOAD	M008
OR	M007	OR	M009
AND LOAD		AND LOAD	
LOAD	M008	AND LOAD	
OR	M009	AND LOAD	
AND LOAD		OUT	P060
OUT	P060	END	
END			

The AND LOAD instruction can be used consecutively up to 7 times (8 block). When connect more than 9 blocks in serial, write a mnemonic program such as the example [A]. If you use KGL-WIN software and write program in ladder mode, the KGL-WIN software will convert the ladder program into mnemonic program [A] automatically.

4.2.2 OR LOAD

OR LOAD	
---------	--

Instructions	Available Device											Steps	Flag		
	M	P	K	L	F	T	C	S	D	#D	Integer		Error (F110)	Zero (F111)	Carry (F112)
OR LOAD												1			

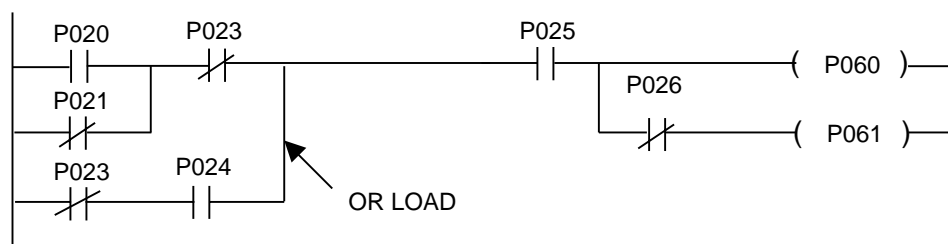


1) Functions

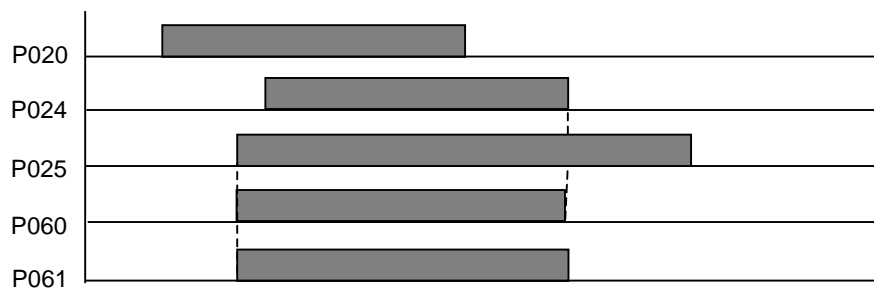
- Performs the OR operation of block A and block B, and uses it as an operation result.
- The AND LOAD instruction can be written consecutively up to 7 times.

2) Program example

[Program]



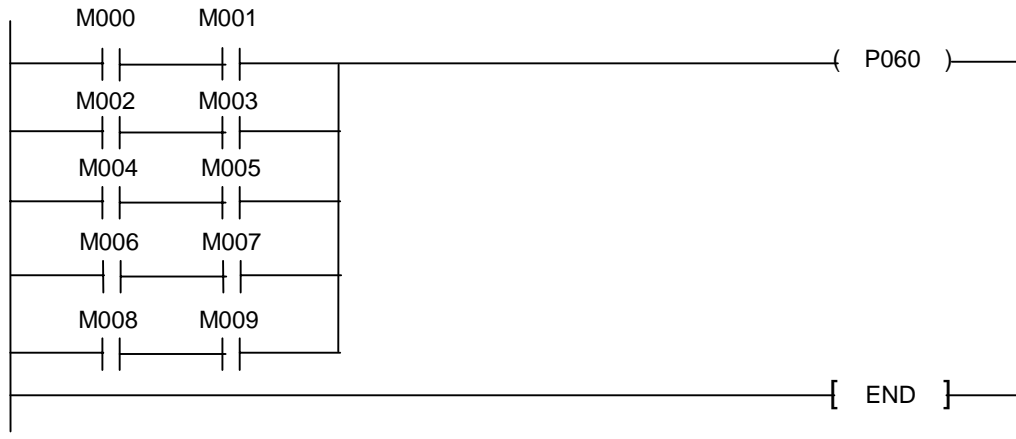
[Time chart]



[NOTE] The consecutive use of the OR LOAD instruction

There are two methods to connect several blocks in parallel. See the following example.

[Ladder program]



[Mnemonic program]

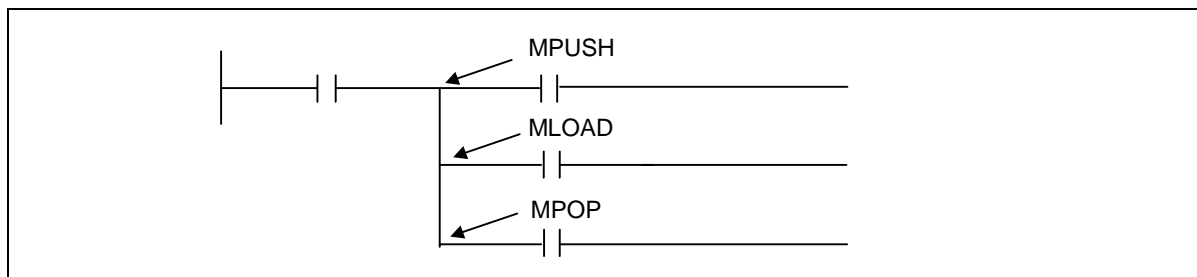
[A] Don't use OR LOAD instructions consecutively		[B] Use OR LOAD instructions consecutively	
LOAD	M000	LOAD	M000
AND	M001	AND	M001
LOAD	M002	LOAD	M002
AND	M003	AND	M003
OR LOAD		LOAD	M004
LOAD	M004	AND	M005
AND	M005	LOAD	M006
OR LOAD		AND	M007
LOAD	M006	LOAD	M008
AND	M007	AND	M009
OR LOAD		OR LOAD	
LOAD	M008	OR LOAD	
AND	M009	OR LOAD	
OR LOAD		OR LOAD	
OUT	P060	OUT	P060
END		END	

The OR LOAD instruction can be used consecutively up to 7 times (8 block). When connect more than 9 blocks in parallel, write a mnemonic program such as the example [A]. If you use KGL-WIN software and write program in ladder mode, the KGL-WIN software will convert the ladder program into mnemonic program [A] automatically.

4.2.3 MPUSH, MLOAD, MPOP

MPUSH	FUN (005) MPUSH
MLOAD	FUN (006) MLOAD
MPOP	FUN (007) MPOP

Instructions	Available Device											Steps	Flag		
	M	P	K	L	F	T	C	S	D	#D	Integer		Error (F110)	Zero (F111)	Carry (F112)
MPUSH MLOAD MPOP												1			

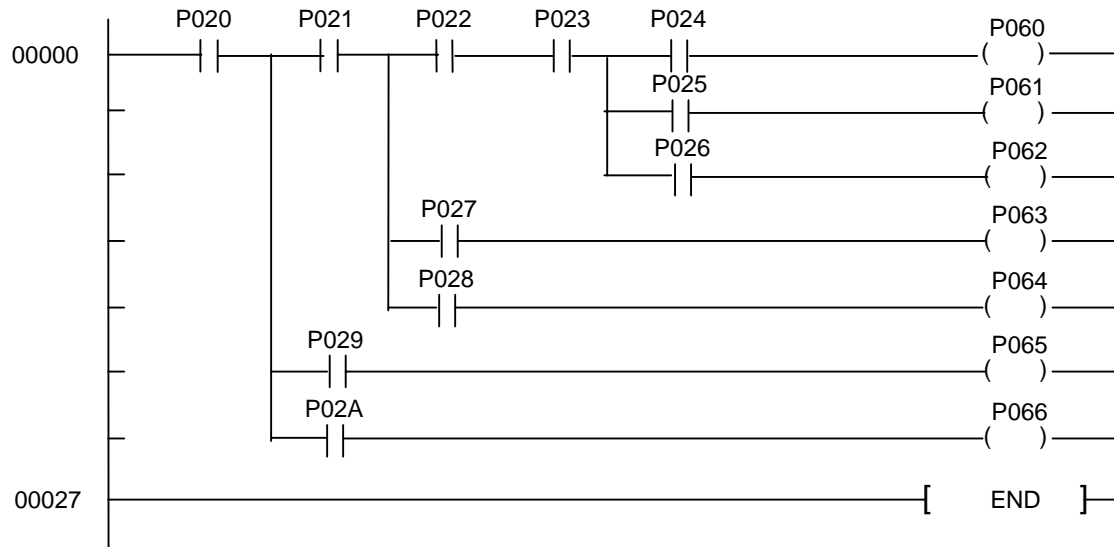


1) Functions

- MPUSH : Stores the operation result (On/Off) immediately preceding the MPUSH instruction.
- MLOAD : Reads the operation result stored by the MPUSH instruction and resume the operation with that operation result, starting at the next step.
- MPOP : Reads the operation result stored by the MPUSH instruction and resumes the operation with that operation result, starting at the next step. Then clears the operation result stored by the MPUSH instruction.
- The MPUSH instruction can be used up to 8 times consecutively. If a MLOAD instruction is used in between, 1 is reduced from the number of used MPS instructions.

2) Program example

[Ladder program]



[Mnemonic program]

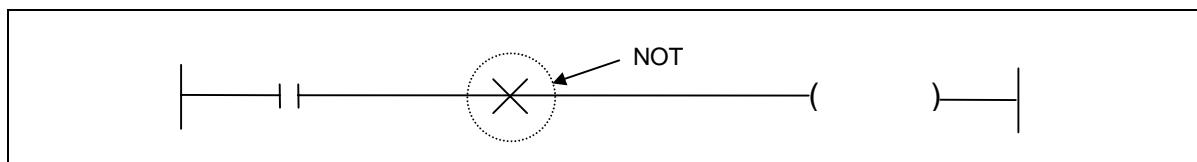
STEP	INSTRUCTION	
0000	LOAD	P020
0001	MPUSH	
0002	AND	P021
0003	MPUSH	
0004	AND	P022
0005	AND	P023
0006	MPUSH	
0007	AND	P024
0008	OUT	P061
0009	MLOAD	
0010	AND	P025
0011	OUT	P061
0012	MPOP	
0013	AND	P026
0014	OUT	P061
0015	MLOAD	
0016	AND	P027
0017	OUT	P063
0018	MPOP	
0019	AND	P028
0020	OUT	P064
0021	MLOAD	
0022	AND	P029
0023	OUT	P065
0024	MPOP	
0025	AND	P02A
0026	OUT	P066
0027	END	
0028	NOP	
0029	NOP	
0030	NOP	

4.3 Inversion instruction

4.3.1 NOT

NOT	
-----	--

Instructions	Available Device											Steps	Flag		
	M	P	K	L	F	T	C	S	D	#D	Integer		Error (F110)	Zero (F111)	Carry (F112)
NOT												1			



1) Functions

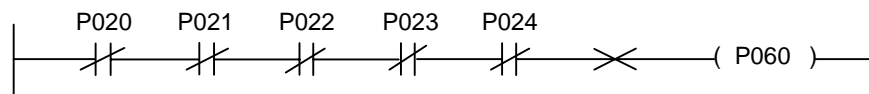
- Inverts the operation result before the NOT instruction.

Before NOT instruction	After NOT instruction
NC contact	NO contact
NO contact	NC contact
Serial (AND) connection	Parallel (OR) connection
Parallel (OR) connection	Serial (AND) connection

2) Program example

The following two programs perform same operation.

Program A



Program B

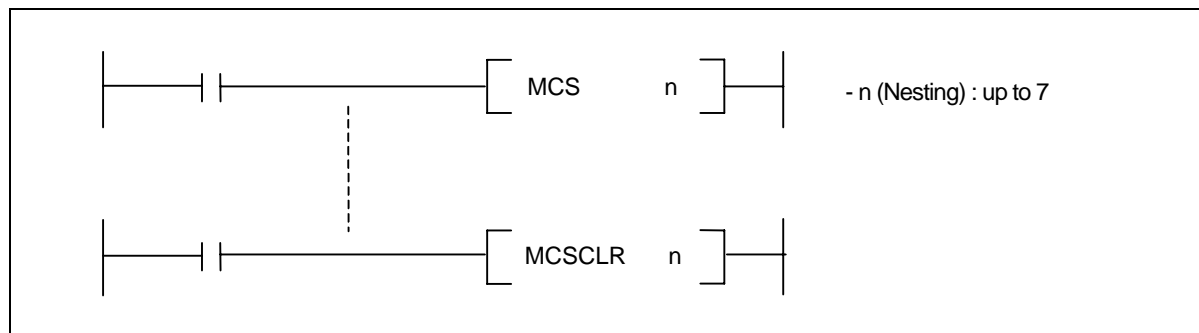


4.4 Master control instructions

4.4.1 MCS, MCSCLR

MCS	FUN (010) MCS
MCSCLR	FUN (011) MCSCLR

Instructions	Available Device											Steps	Flag		
	M	P	K	L	F	T	C	S	D	#D	Integer		Error (F110)	Zero (F111)	Carry (F112)
MCS MCSCLR											O	1			

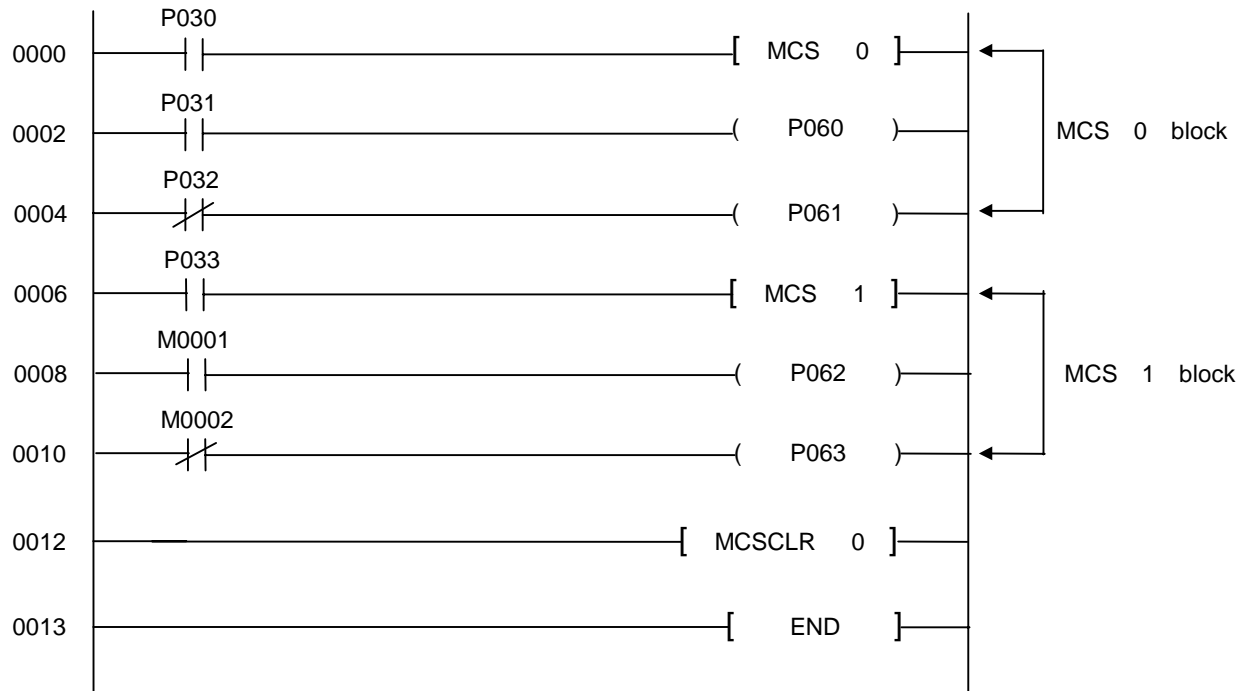


1) Functions

- When the On/Off command of MCS instruction is switched on, the sequence program between the MCS instruction and the MCSCLR instruction that has same n of the MCS instruction is executed.
- Each MCS instructions are followed by a number (n) that shows the priority of master control. 0 is the highest priority, and 7 is lowest priority. The MCS instruction should be used in order of priority level.
- The MCSCLR instruction shows the end of master control. When a MCSCLR n instruction is executed, all master control that has lower priority than 'n' are ended automatically.

2) Program example

Use 2 master control blocks (MCS 0 and MCS 1), and they are cleared with MCSCLR 0 instruction. The MCS 1 block is cleared automatically.



Remark

- Scanning between the MCS and MCSCLR instructions is executed even when the On/Off command for the MCS instruction is off. Therefore, scan time does not become shorter.
- When On/Off command for the MCS instruction is off, the operation result of MCS to MCSCLR is as indicated below.

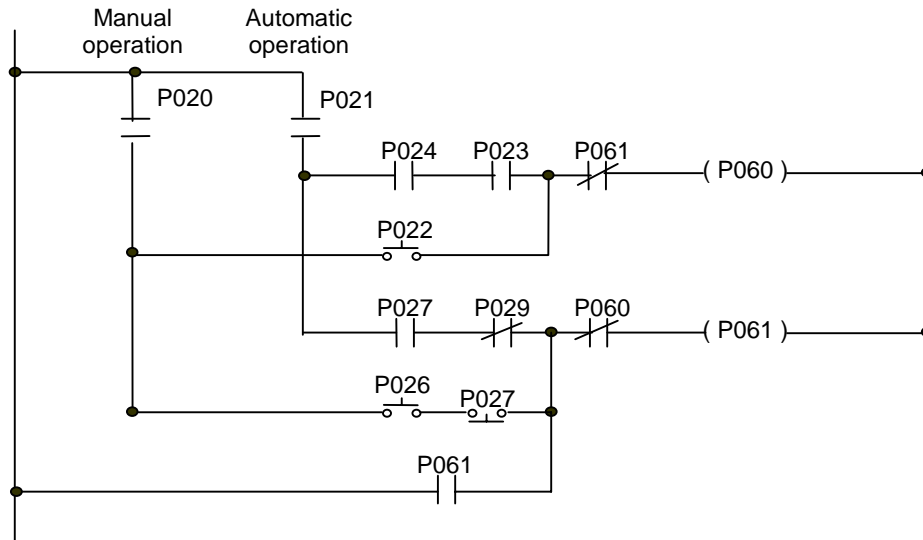
TIMER	The timer output contact turns off and the current value becomes 0.
COUNTER	The counter output contact turns off but the current value holds the present value.
OUT	All turns off
SET, RST	Hold present value

- If an instruction which does not need a contact instruction immediately before it (FOR, NEXT, EI, DI, etc.) is contained in the MCS ~ MCSCLR block, the CPU executes the instruction regardless of the status of the On/Off command for the MCS instruction.

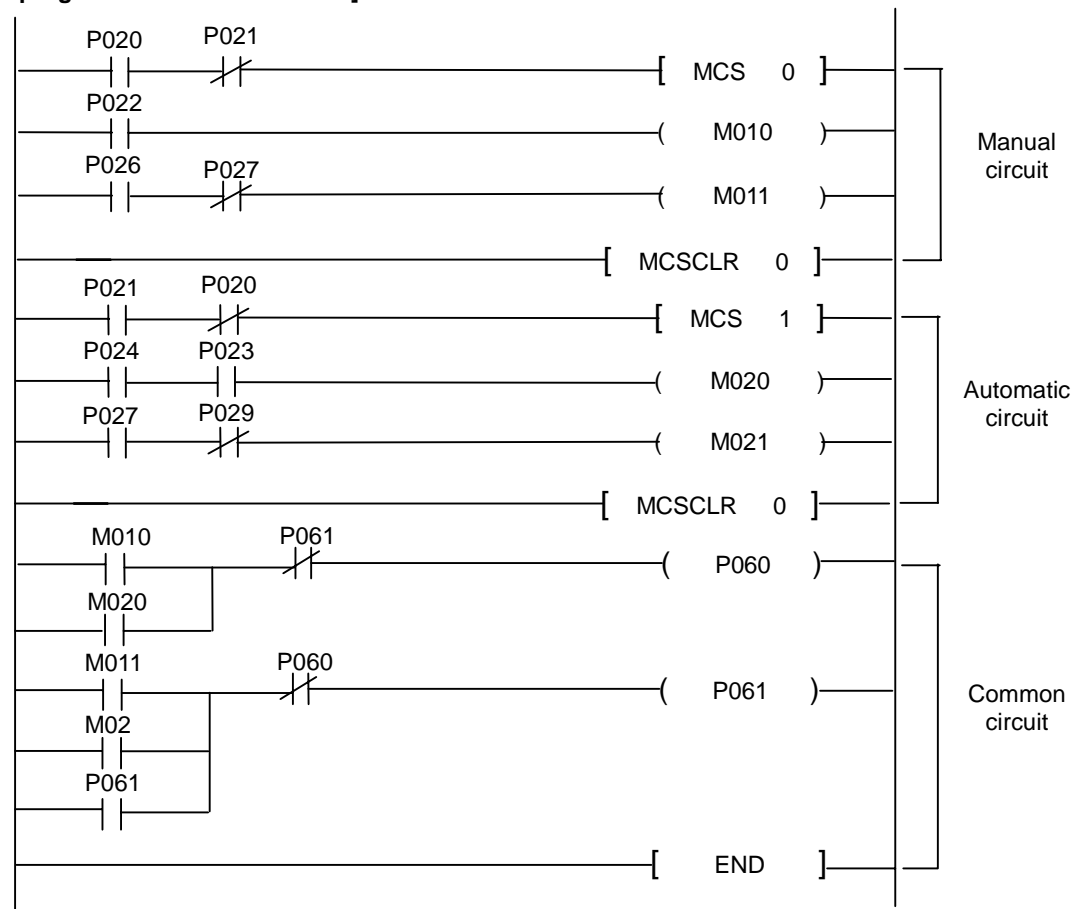
The circuit with common line (Example of MCS, MCSCLR instructions)

The below relay circuit can not be programmed into PLC program directly. Therefore, it should be programmed with master control. (MCS and MCSCLR instructions)

[Relay circuit]



[PLC program with master control]

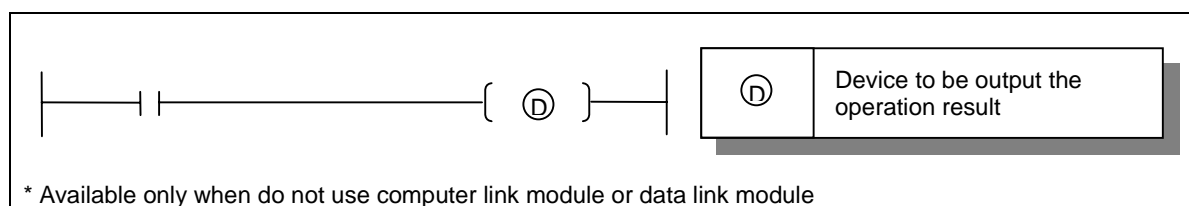


4.5 Output instructions

4.5.1 OUT

OUT	
-----	--

Instructions		Available Device											Steps	Flag		
		M	P	K	L	F	T	C	S	D	#D	Integer		Error (F110)	Zero (F111)	Carry (F112)
OUT	ⓓ	O	O	O	O*								2			

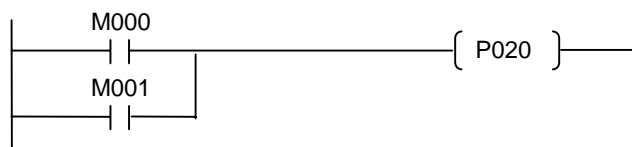


1) Functions

- Outputs the operation result to the bit device specified as [D].

2) Program example

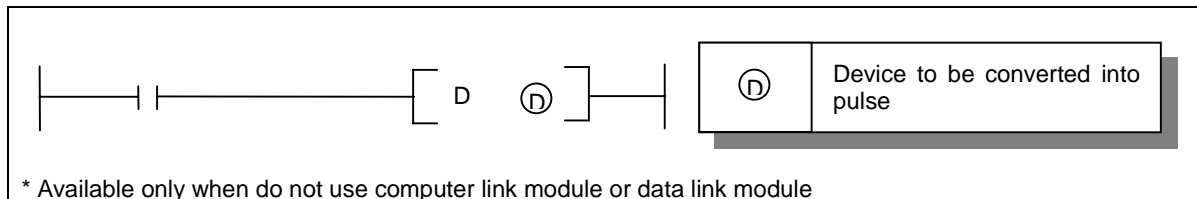
- Program that turn the P020 on when the M000 or M001 is turn on.



4.5.2 D

D	FUN (017) D
---	-------------

Instructions		Available Device										Steps	Flag			
		M	P	K	L	F	T	C	S	D	#D		Integer	Error (F110)	Zero (F111)	Carry (F112)
D	ⓓ	O	O	O	O*								2			



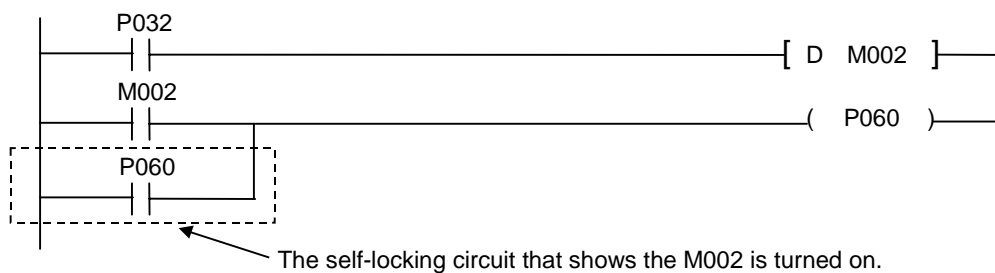
1) Functions

- The D instruction turns on the specified device for one scan when the input condition of D instruction is turned on.
- Be careful when use a P area as (D)

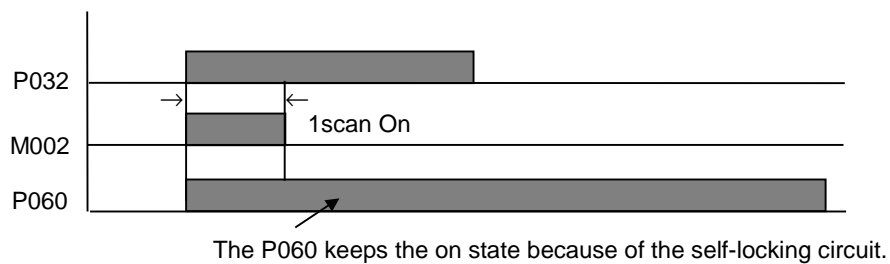
2) Program example

When the P032 is switched on, the M002 is turned on for one scan.

[Program]



[Time chart]

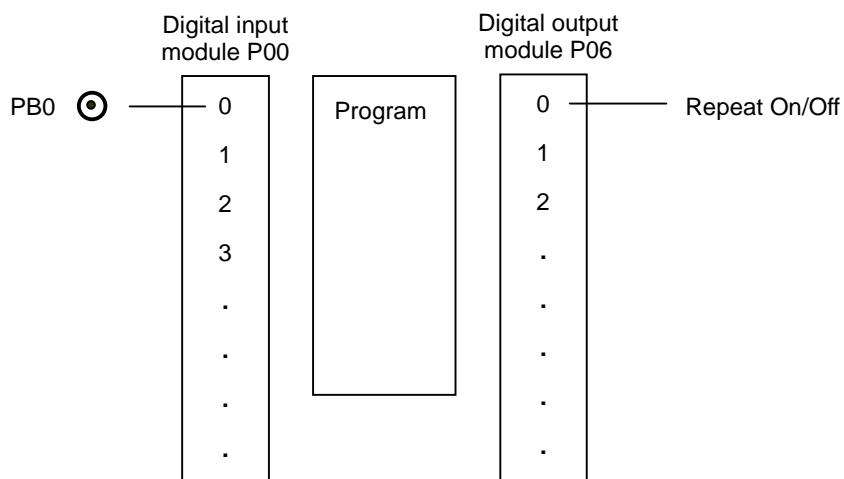


The on / off toggle control (Example of D instructions)

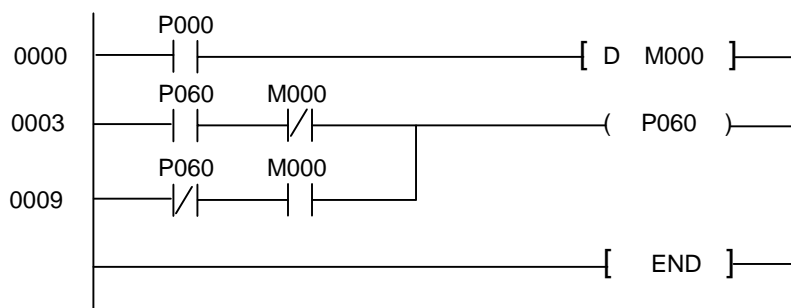
1. Operation

When the push-button PB0 is pushed, the P060 is switched on. It is switched off when the PB0 is pushed again. The P060 will repeat on / off whenever the PB0 is pushed.

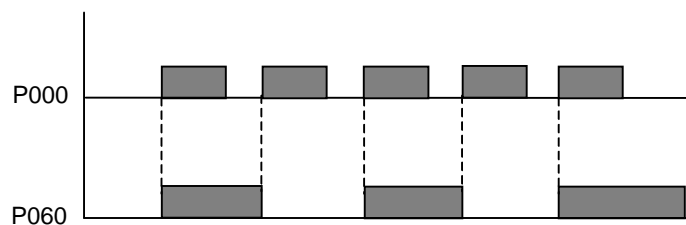
2. System structure



3. Program



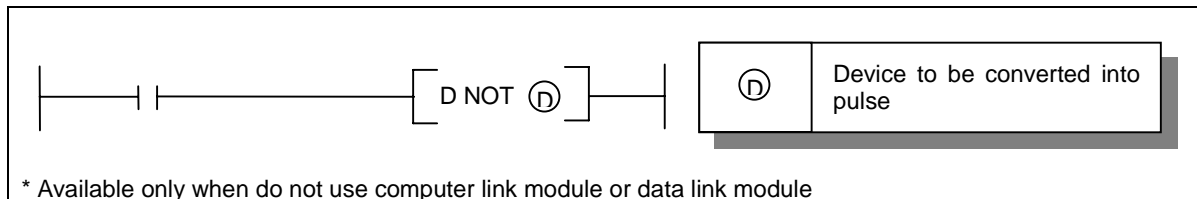
4. Time chart



4.5.3 D NOT

D NOT	FUN (018) D NOT
-------	-----------------

Instructions		Available Device										Steps	Flag			
		M	P	K	L	F	T	C	S	D	#D		Integer	Error (F110)	Zero (F111)	Carry (F112)
D NOT	ⓓ	O	O	O	O*								2			



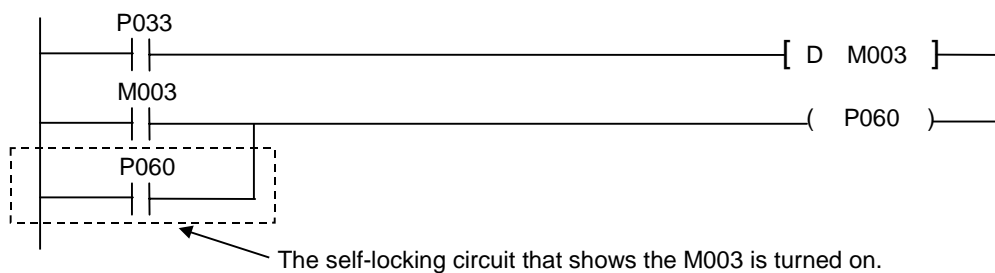
1) Functions

- The D instruction turns on the specified device for one scan when the input condition of D instruction is turned on.
- Be careful when use a P area as (D).

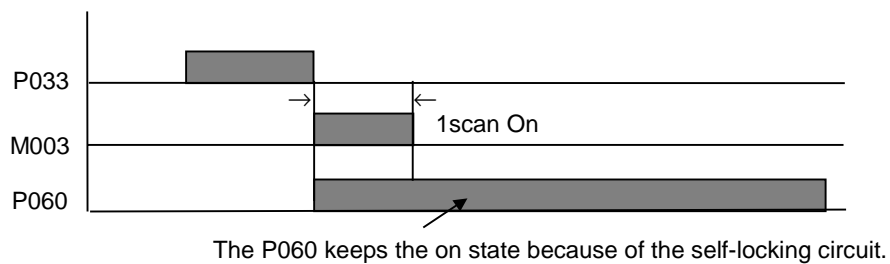
2) Program example

When the P033 is switched off, the M003 is turned on for one scan.

[Program]



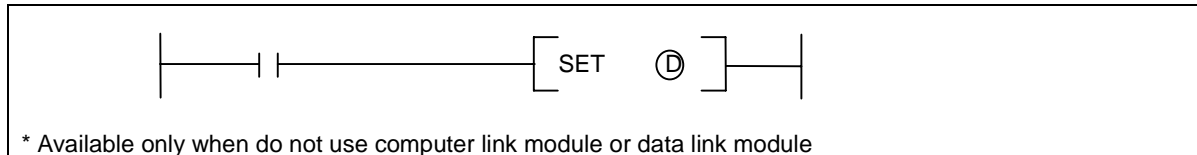
[Time chart]



4.5.4 SET

SET	
-----	--

Instructions		Available Device										Steps	Flag			
		M	P	K	L	F	T	C	S	D	#D		Integer	Error (F110)	Zero (F111)	Carry (F112)
SET	Ⓓ	O	O	O	O*				O				1			



1) Functions

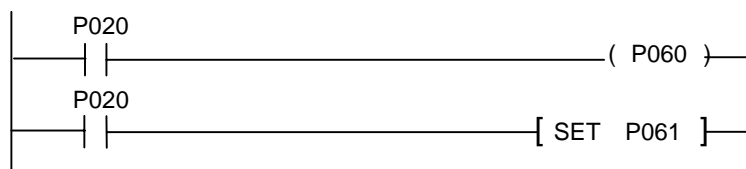
- When the input condition of SET instruction turns on, the specified device is switched on.
- The turned-on device remains on even if the input condition of SET instruction is turned off. The device can be switched off by the RST instruction.

Ⓓ

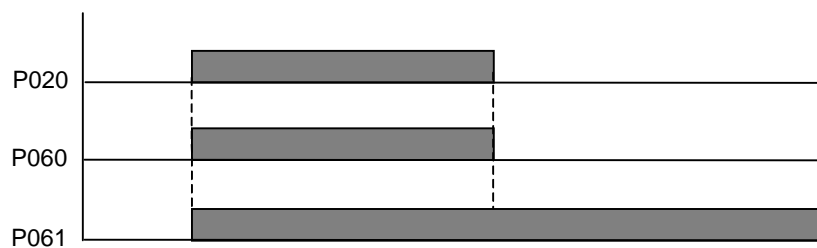
2) Program example

When the input condition P020 turns on, the P060 and P061 is turned on by OUT and SET instruction.

[Program]



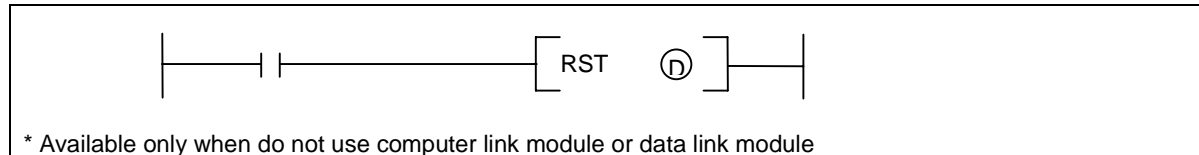
[Time chart]



4.5.5 RST

RST	
-----	--

Instructions		Available Device										Steps	Flag			
		M	P	K	L	F	T	C	S	D	#D		Integer	Error (F110)	Zero (F111)	Carry (F112)
RST	ⓓ	O	O	O	O*		O						1			



1) Functions

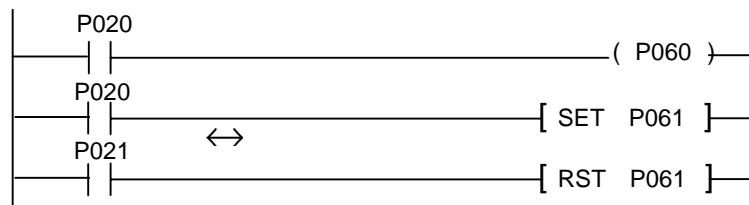
- When the input condition of RST instruction turns on, the specified device is changed as described

Device	Status
M, P, K, L	The specified bit is turned off.
T	The timer output is turned off and current value is cleared as 0

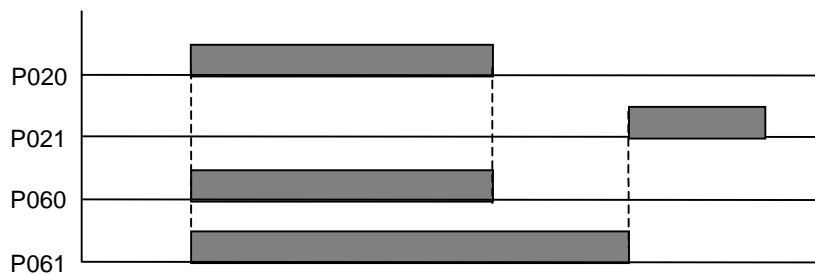
2) Program example

Set the P061 with the P020 and reset the P061 with the P021.

[Program]



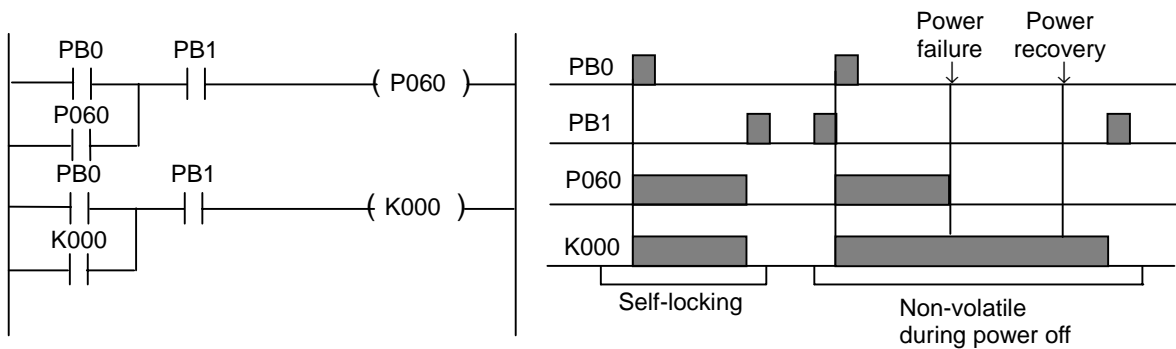
[Time chart]



The countermeasure of power failure (The difference of P area and K area)

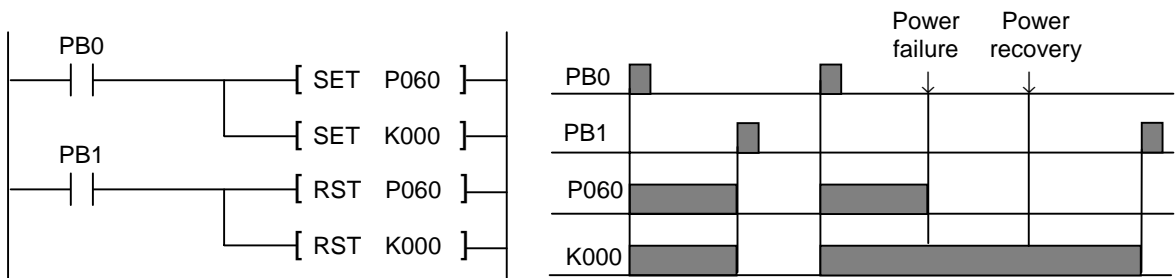
1. The difference of I/O relay (P) and keep relay (K) with the OUT instruction

The following program shows the difference of P area and K area. Both of P060 and K000 have self-locking circuit, and the operation of two contact is same. However, when the power is switched off and turned on again (power failure), operations of P and K are different as shown below.



2. The difference of I/O relay (P) and keep relay (K) with the SET/RST instruction

The SET instruction makes turn a specified device on and keep the on status until the RST instruction is executed. However, the P and K area operate differently when a power failure occurred.

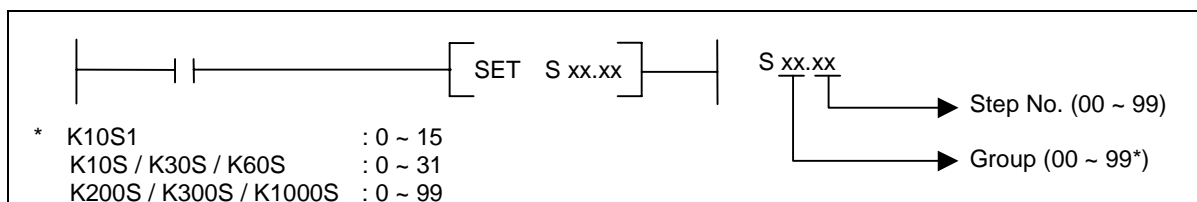


4.6 Step controller instructions

4.6.1 SET Sxx.xx

SET S	
-------	--

Instructions		Available Device											Steps	Flag		
		M	P	K	L	F	T	C	S	D	#D	Integer		Error (F110)	Zero (F111)	Carry (F112)
SET S	ⓓ								O				2			



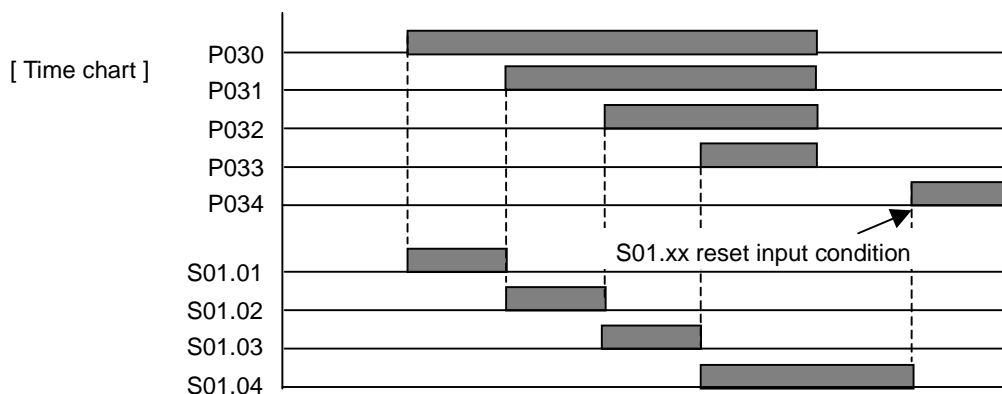
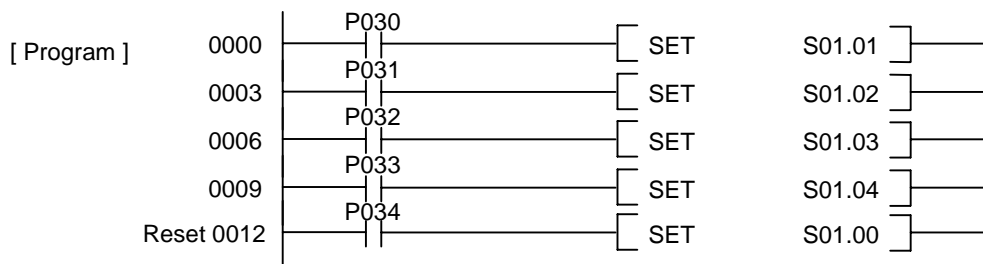
1) Functions

- The Sxx.xx contact will turn on when the previous step of same group and the input condition is on.
- Once a Sxx.xx is switched on, it keeps on state until the next step turns on or the step controller group is initialized. (The Sxx.00 is switched on)
- Even if multiple input condition turn on, only one step controller is switched on.
- The Sxx.00 is initialization step and the Sxx.xx group will be initialized by switching on the Sxx.00.

When the CPU is turned to RUN mode, the Sxx.00 is set by default.

2) Program example

A sequential control by using S01.xx group

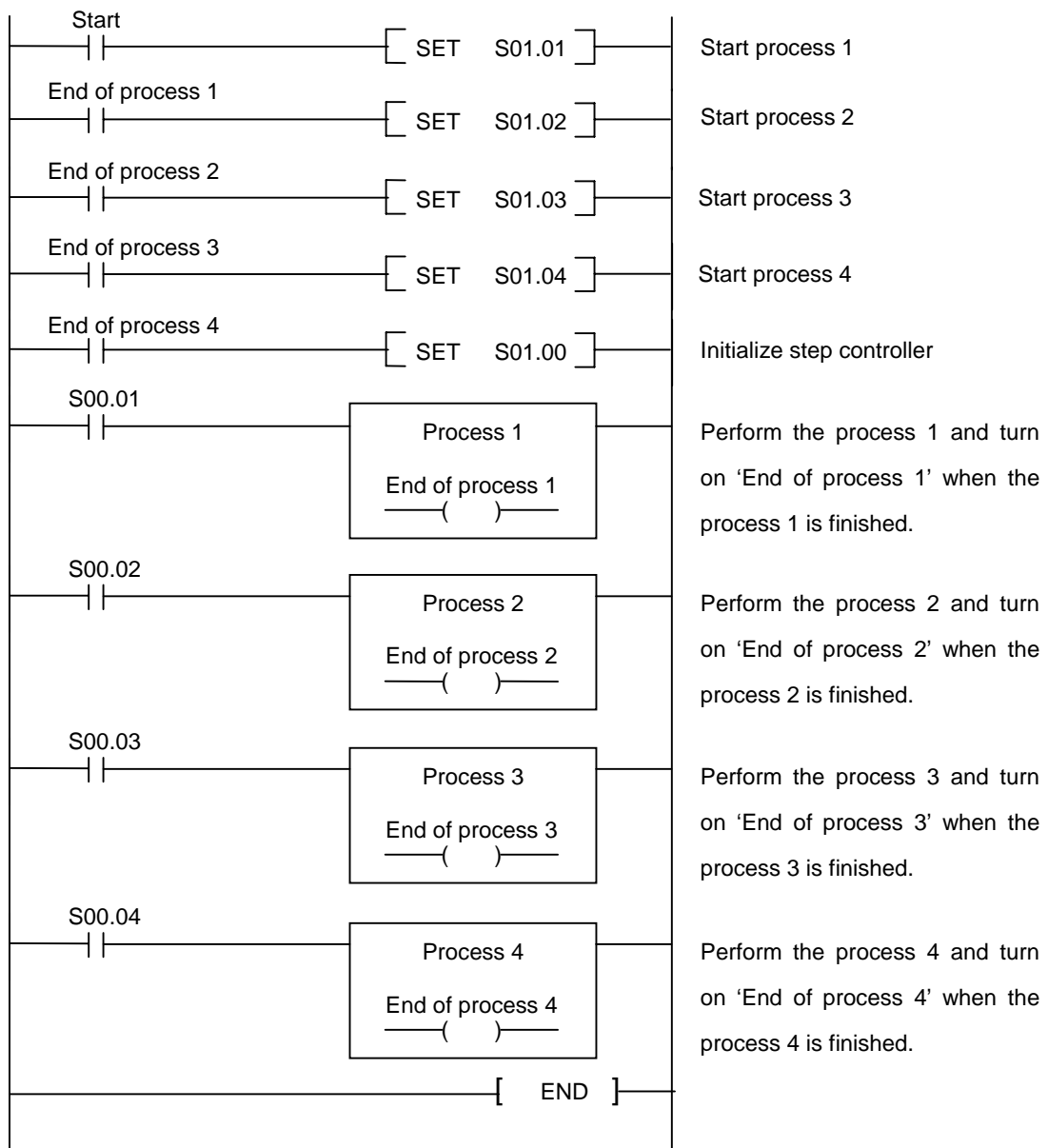


The sequential control (example of SET Sxx.xx instruction)

1. Operation

This program shows briefly an example of sequential control by using SET Sxx.xx instruction. In this example, there are 4 processes and each process is performed in sequence. The process 2 starts after the process 1 ended, and process 3 starts after the process 2 finished. When the process 4 is completed, the process 1 will start again.

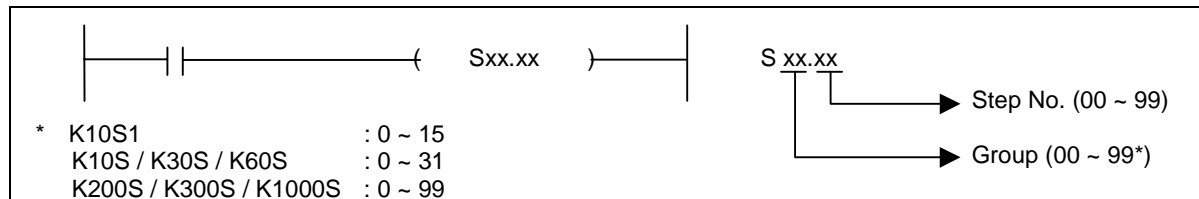
2. Program



4.6.2 OUT Sxx.xx

OUT S	
-------	--

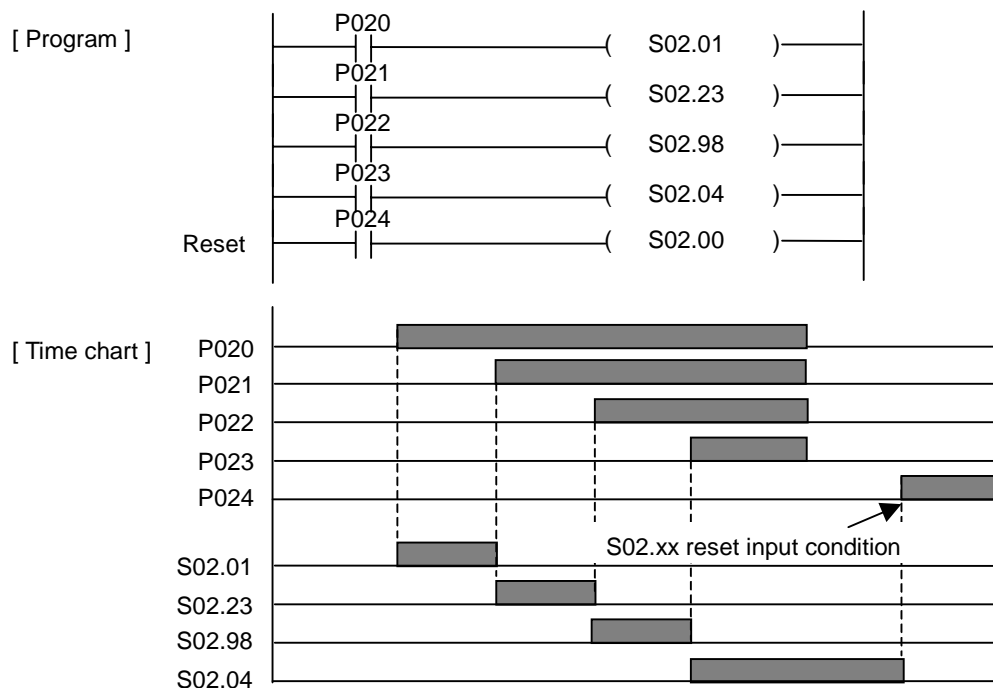
Instructions		Available Device											Steps	Flag		
		M	P	K	L	F	T	C	S	D	#D	Integer		Error (F110)	Zero (F111)	Carry (F112)
OUT S	ⓓ								O				2			



1) Function

- Last-in priority control
- When the input condition turns on, the specified step controller is switched on and keep the on status until other step controller of same group turns on.
- Only one step controller turns on even if multiple input conditions turn on. At this time, the last turned on step controller has the highest priority.
- The Sxx.00 is initialization step and the Sxx.xx group will be initialized by switching on the Sxx.00. When the CPU is turned to RUN mode, the Sxx.00 is set by default.

2) Program example

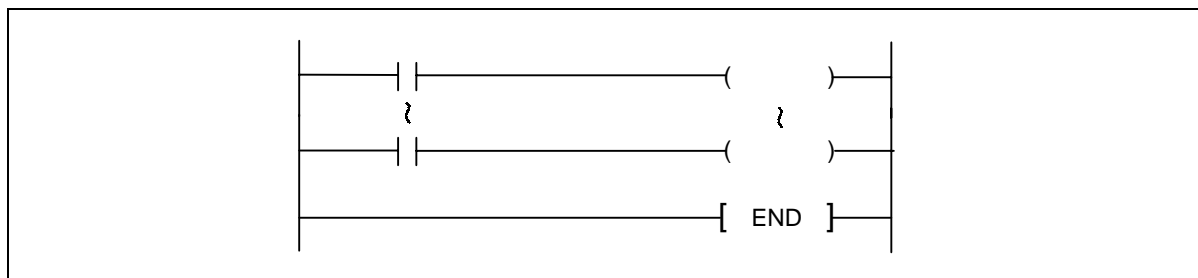


4.7 End instructions

4.7.1 END

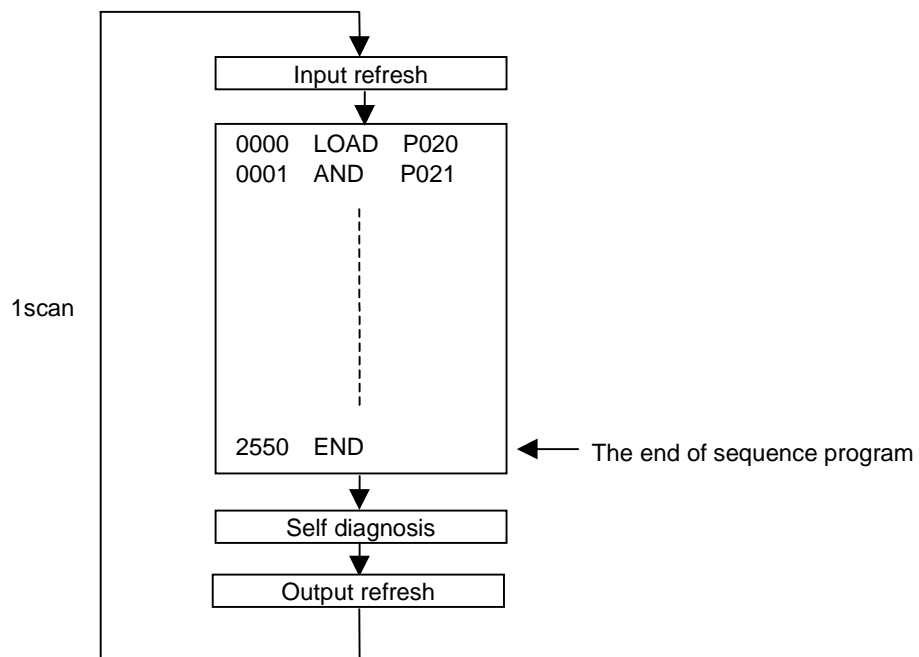
END	FUN (001) END
-----	---------------

Instructions	Available Device											Steps	Flag		
	M	P	K	L	F	T	C	S	D	#D	Integer		Error (F110)	Zero (F111)	Carry (F112)
END												1			



1) Functions

- The END instruction indicates the end of sequence program. When the CPU meet the END instruction, it stops the execution of sequence program and perform the END process.
- All instructions after END instruction except subroutines and interrupt routine are ignored and not executed.
- If there is no END instruction, the program error will occur.



4.8 No operation instruction

4.8.1 NOP

NOP	FUN (000) NOP
-----	---------------

Instructions	Available Device											Steps	Flag		
	M	P	K	L	F	T	C	S	D	#D	Integer		Error (F110)	Zero (F111)	Carry (F112)
NOP												1			

No ladder symbol (Only available in mnemonic mode)

1) Functions

- This is a no operation instruction and has no effect on the previous operation result.
- The NOP instruction is used in the following cases ;
 - a) To provide space for debugging of sequence programs.
 - b) To delete an instruction without changing the number of steps.
 - c) To delete an instruction temporarily.

2) Program example

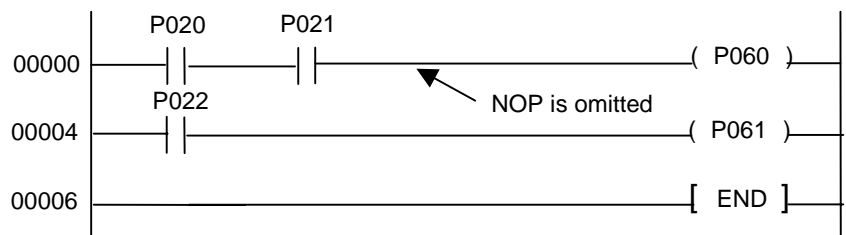
[Mnemonic program]

```

0000 LOAD P020
0001 AND P021
0002 NOP
0003 OUT P060
0004 LOAD P022
0005 OUT P061
0006 END
    
```



[Ladder program]

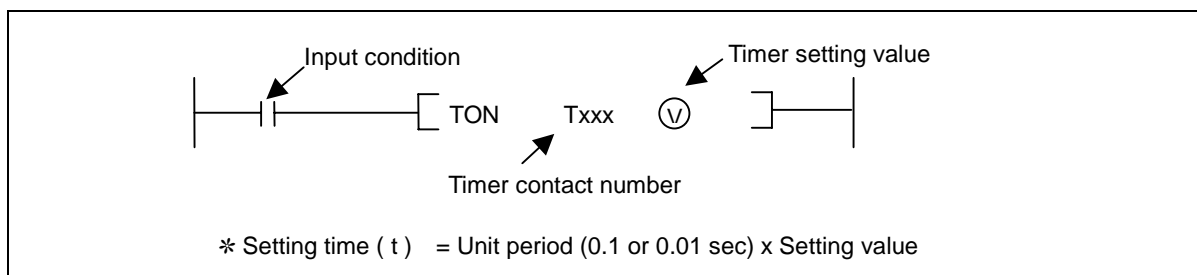


4.9 Timer instructions

4.9.1 TON

TON	On-delay timer
-----	----------------

Instructions		Available Device											Steps	Flag		
		M	P	K	L	F	T	C	S	D	#D	Integer		Error (F110)	Zero (F111)	Carry (F112)
TON	Txxx						O						3			
	(V)									O		O				

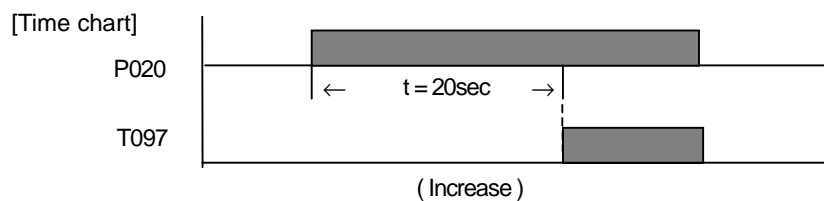
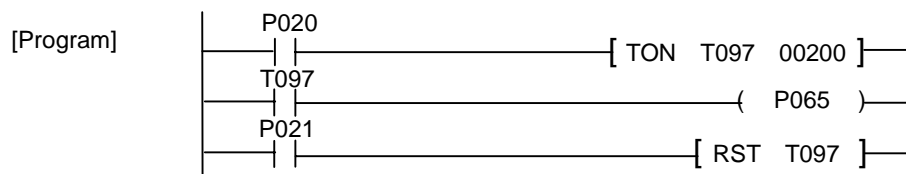


1) Functions

- A timer consists of timer contact, current value, and setting value.
- The current value will start to increase when the input condition turns on. It will increase by 1 at every 0.1 or 0.01 sec until it reaches to the setting value or input condition turns off.
- The timer contact will be switched on when the current value reaches to the setting value.
- The timer contact and current value is cleared when the input condition turns off or RST instruction is executed.

2) Program example

The T097 (0.01 sec timer) will turn on 20 seconds later until the P020 is switched on.

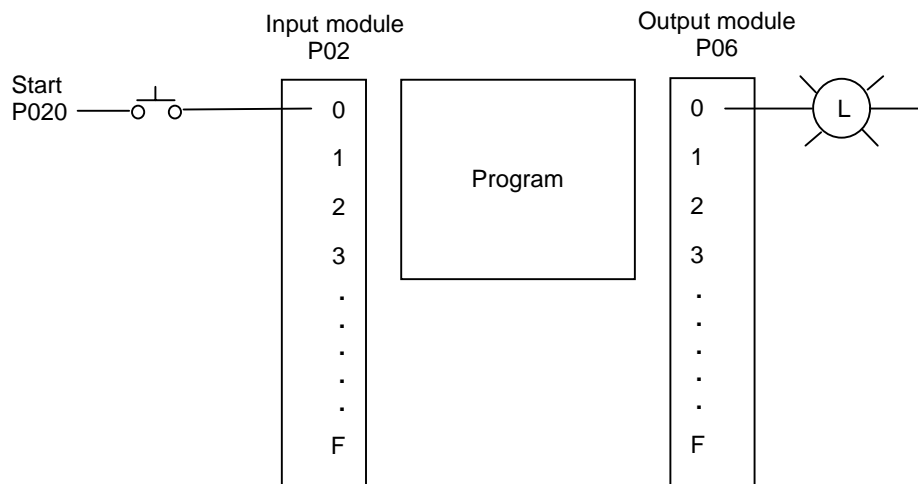


A flickering lamp (example of TON instruction)

1. Operation

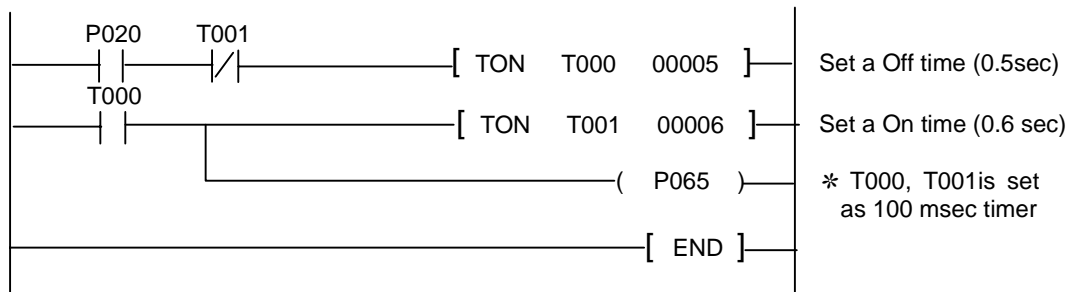
By using two timers, a lamp flickers periodically while the P020 is on.

2. System diagram

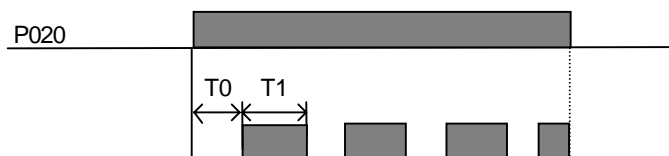


3. Program

[Ladder program]



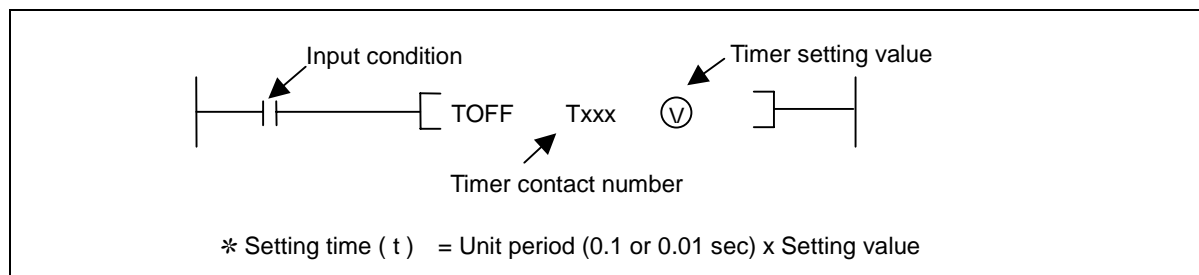
[Time chart]



4.9.2 TOFF

TOFF	Off-delay timer
------	-----------------

Instructions		Available Device										Steps	Flag		
		M	P	K	L	F	T	C	S	D	#D	Integer	Error (F110)	Zero (F111)	Carry (F112)
TOFF	Txxx						O					3			
	(V)									O		O			

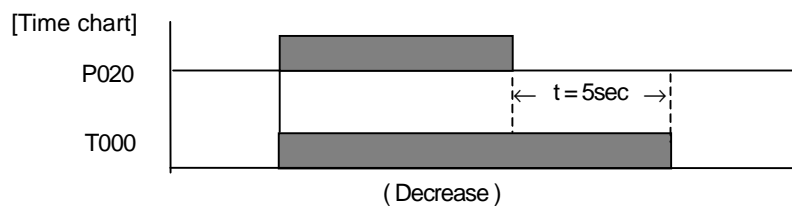
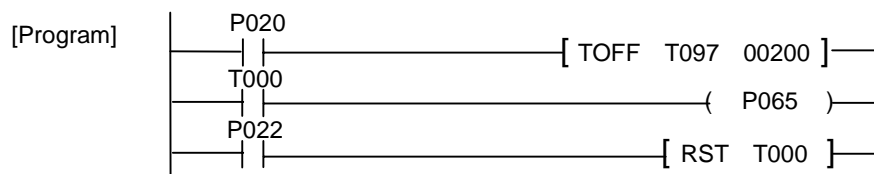


1) Functions

- A timer consists of timer contact, current value, and setting value.
- When the input condition turns on, the current value will be set as the setting value and the timer contact will turn on.
- When the input condition turns off, the current value will decrease by 1 at every 0.1 or 0.01 sec until it reaches to 0 or input condition turns off.
- The timer contact will be switched off when the current value reaches to 0.
- When the input condition turns off or RST instruction is executed, the timer contact will turn off and the current value will be cleared as 0.

2) Program example

The T000 (0.1 sec timer) will turn off 5 seconds later until the P020 is switched off.



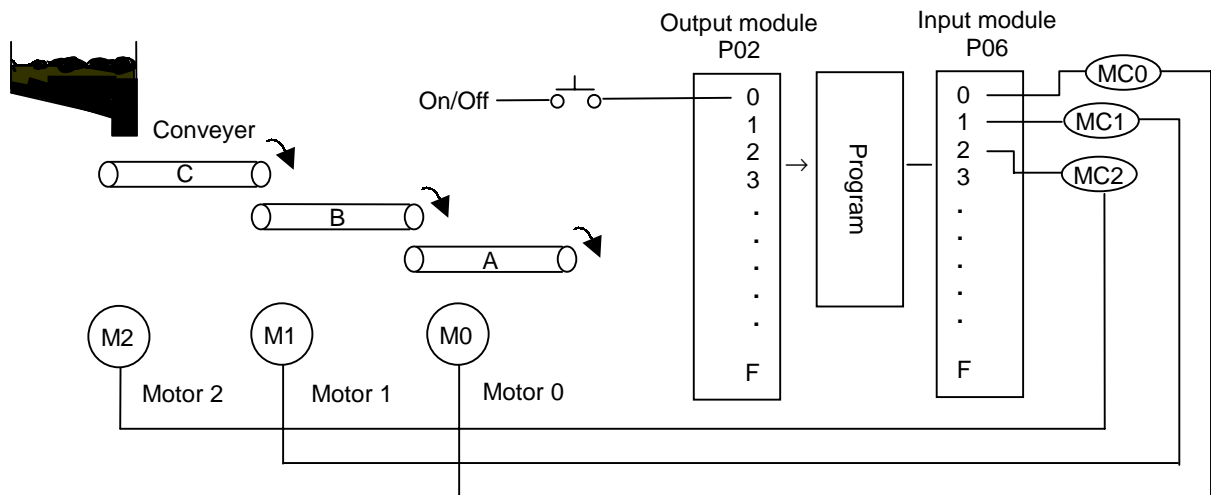
A conveyor control (example of TOFF instruction)

1. Operation

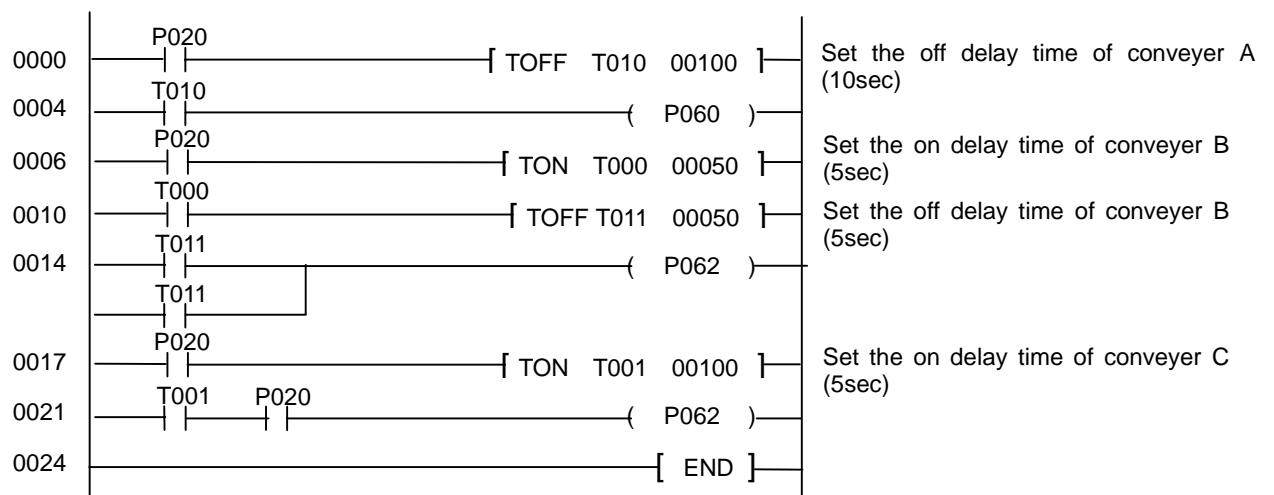
Operate three conveyers (A, B, C) in sequence by using TOFF timers.

(Start : A – B – C, Stop : C – B – A)

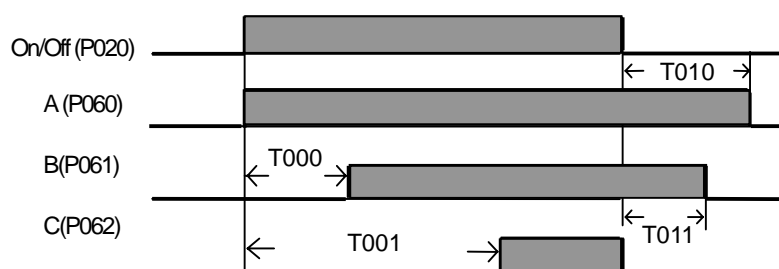
2. System diagram



3. Program



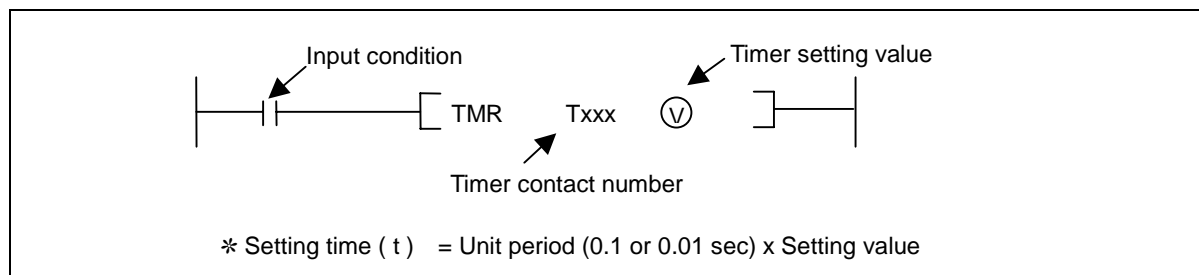
[Time chart]



4.9.3 TMR

TMR	Integrating timer
-----	-------------------

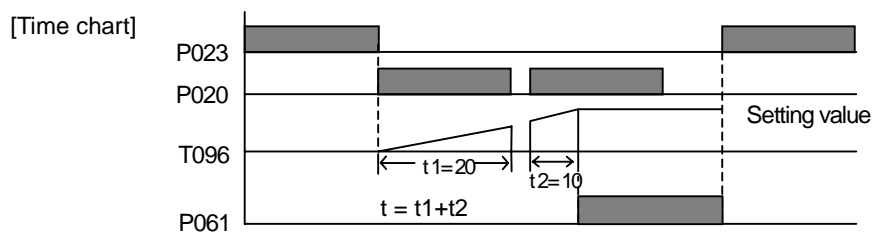
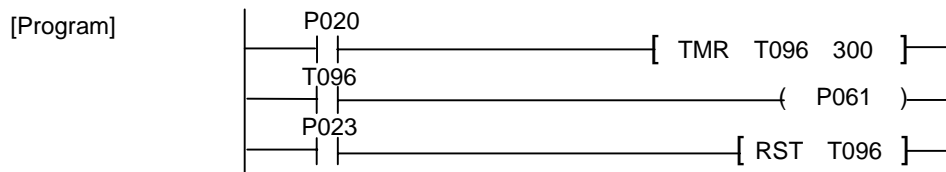
Instructions		Available Device											Steps	Flag		
		M	P	K	L	F	T	C	S	D	#D	Integer		Error (F110)	Zero (F111)	Carry (F112)
TMR	Txxx						O						3			
	(V)									O		O				



1) Functions

- The current value will increase by 1 while the input condition is on.
- When the current value reached to the setting value, the timer contact turns on.
- Even if the input condition is off, the current value is not cleared. If uses a timer of retentive data area, the timer will keep the current value while the CPU is powered off.
- When the RST instruction is executed, the timer contact and current value will be cleared as 0.

2) Program example

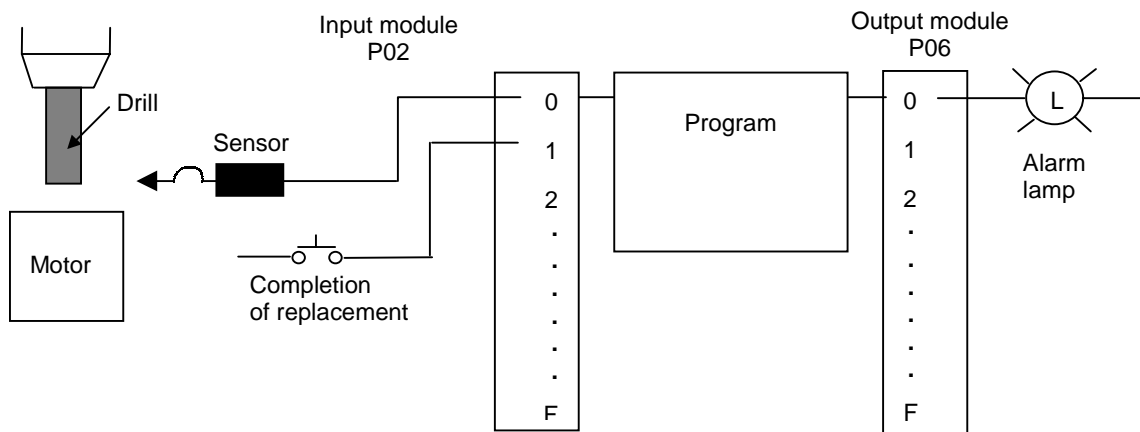


An alarm of drill replacement (example of TMR instruction)

1. Operation

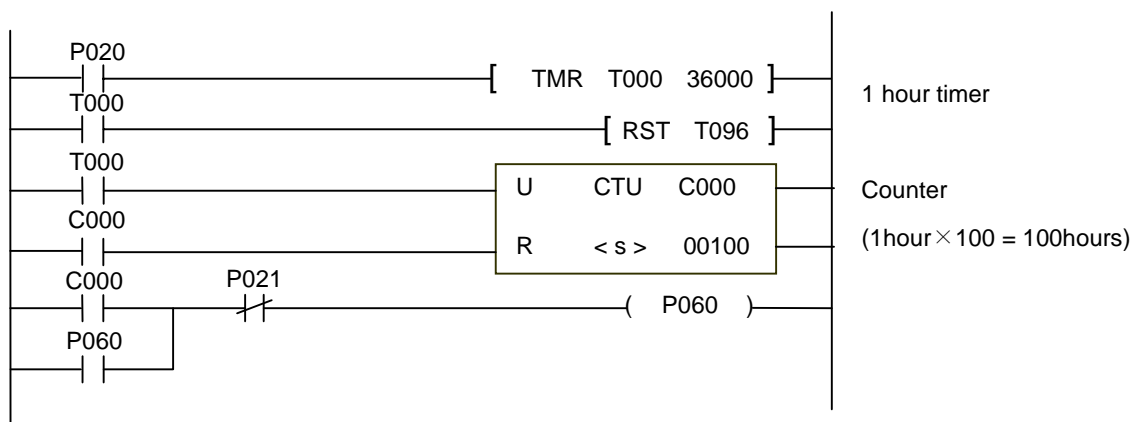
The total usage time of the drill of a machining center is counted by PLC. If the total usage time exceeds the lifetime of drill (100 hours), the PLC outputs an alarm signal to notice that a replacement of drill is required.

2. System diagram



I/O	Description
P020	Detect of drill down
P021	Replacement completion
P060	Turn on an alarm lamp
T000	Timer for the lifetime of drill

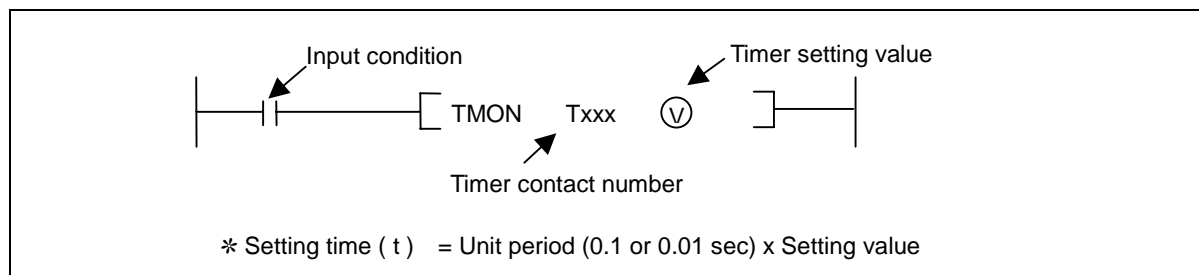
3. Program



4.9.4 TMON

TMON	Monostable timer
------	------------------

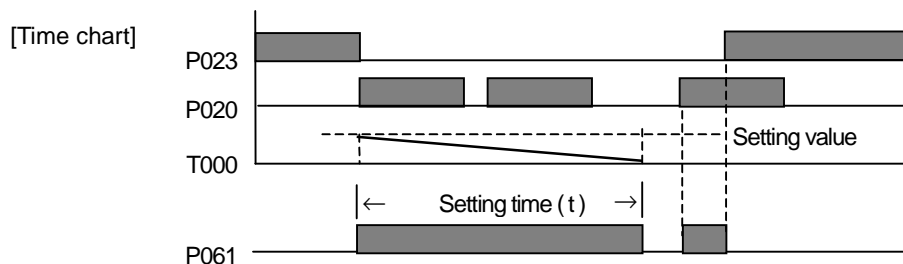
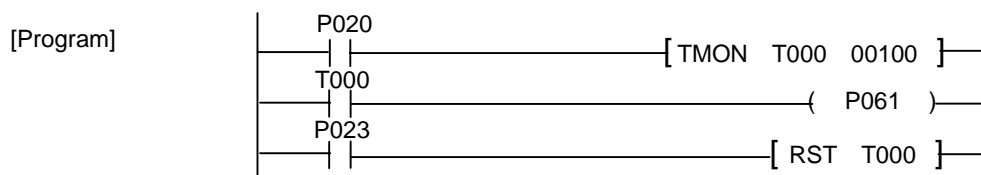
Instructions		Available Device											Steps	Flag		
		M	P	K	L	F	T	C	S	D	#D	Integer		Error (F110)	Zero (F111)	Carry (F112)
TMON	Txxx						O						3			
	(V)									O		O				



1) Functions

- When the input condition turns on, the current value will be set as the setting value and starts to decrease. The timer contact turns on when the input condition turns on.
- When the input condition turns off, the current value will decrease by 1 at every 0.1 or 0.01 sec until it reaches to 0 and the timer contact will be switched off when the current value reaches to 0.
- While a timer is operating, on/off changed of input condition is ignored.
- When the RST instruction is executed, the timer contact will turn off and the current value will be cleared as 0.

2) program example

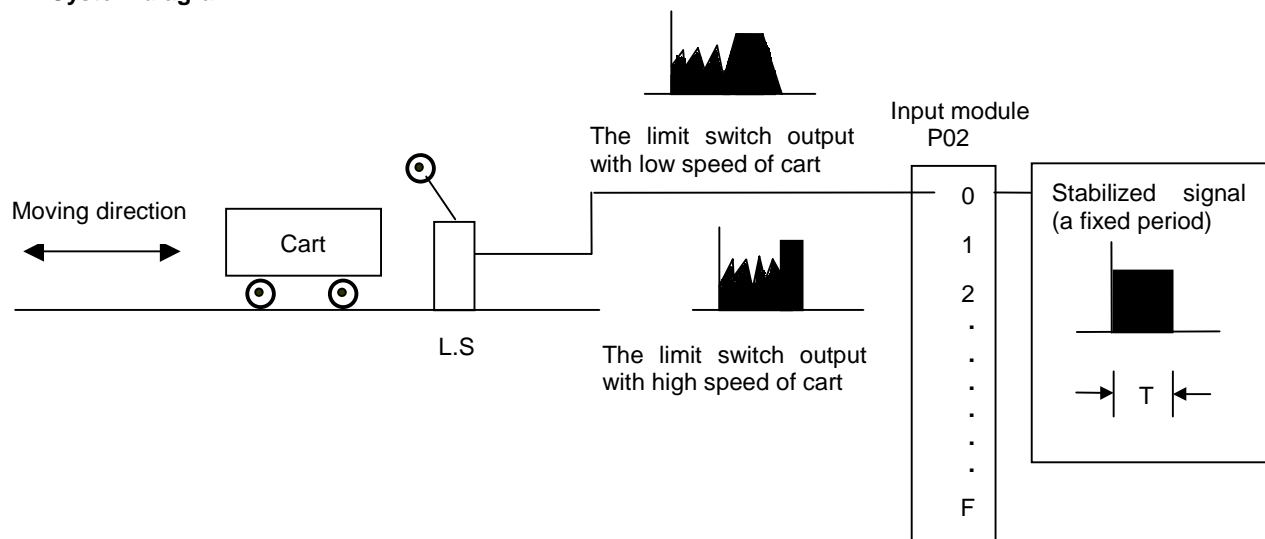


A chattering prevention circuit (example of TMON instruction)

1. Operation

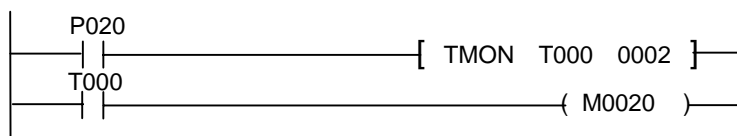
A input signal from limit switch has chattering noise. By using TMON instruction, a stabilized signal can be obtained from a noisy input signal.

2. System diagram



I/O	Description
P020	Limit switch output
M0020	Stabilized signal of P020
T000	Chattering prevention timer

3. Program

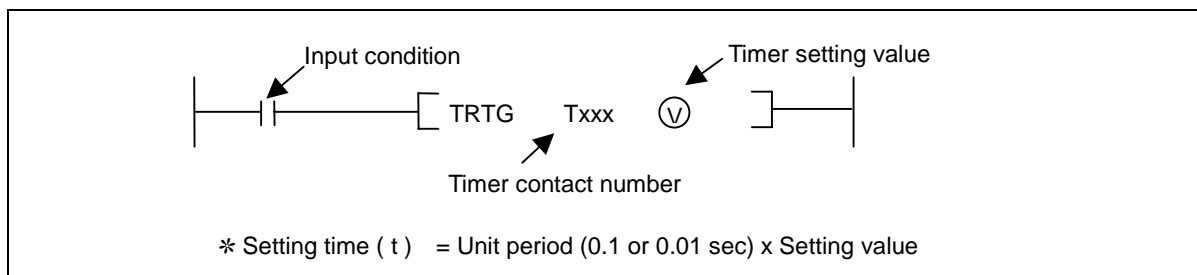


Even if P020 trembles after P020 turns on in a moment, M0020 keeps its on state for 0.2 second.

4.9.5 TRTG

TRTG	Retrigerrable timer
------	---------------------

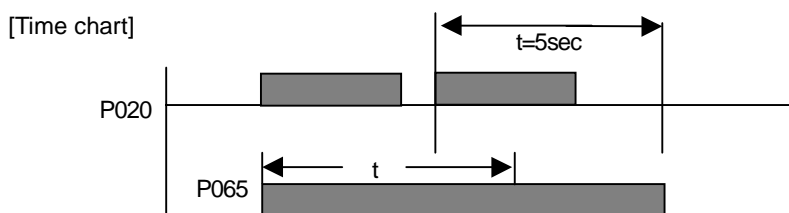
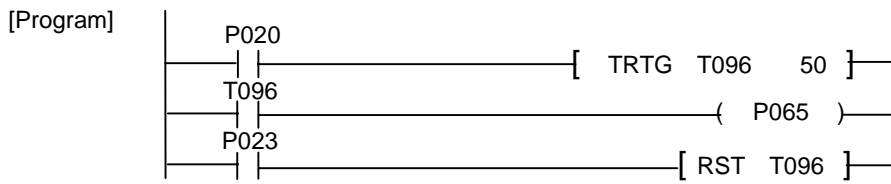
Instructions		Available Device											Steps	Flag		
		M	P	K	L	F	T	C	S	D	#D	Integer		Error (F110)	Zero (F111)	Carry (F112)
TRTG	Txxx						O						3			
	(V)									O		O				



1) Functions

- When the input condition turns on, the current value will be set as the setting value and starts to decrease. The timer contact turns on when the input condition is switched on.
- The current value will decrease by 1 at every 0.1 or 0.01sec until it reaches to 0 and the timer contact will be switched off when the current value reaches to 0.
- If the input condition turns on again during timer operation, the current value will reset as the setting value and re-start to decreasing from the setting value.
- When the RST instruction is executed, the timer contact will turn off and the current value will be cleared as 0.

2) Program example

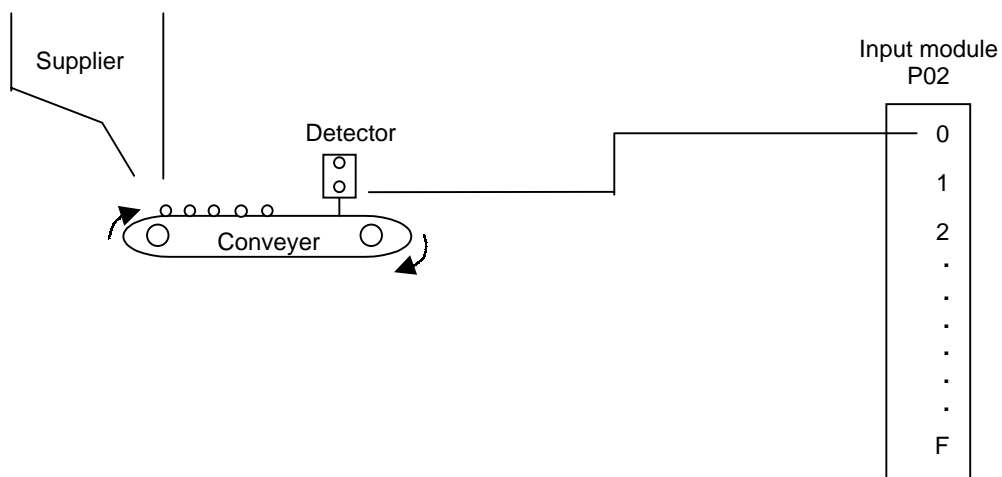


The fault of conveyer detecting circuit (example of TRTG instruction)

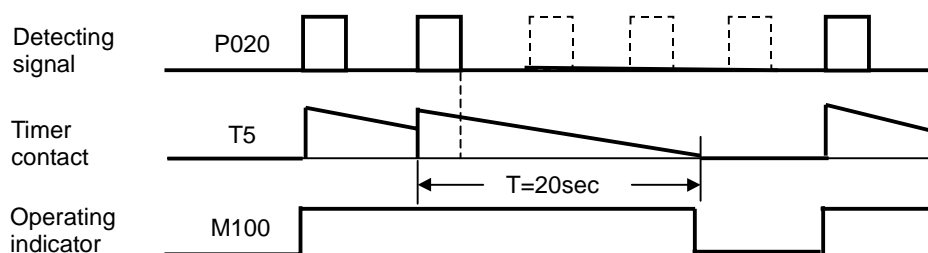
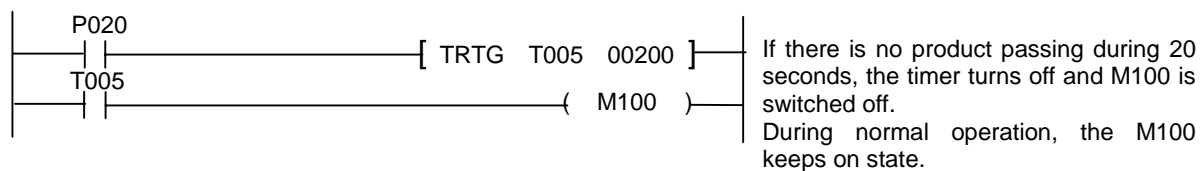
1. Operation

Detect the fault of conveyer by check that a product is passed within a specified period or not.

2. System diagram



3. Program

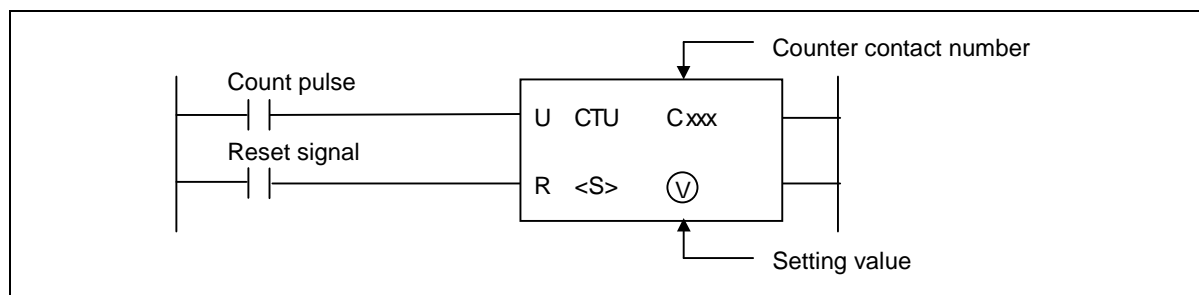


4.10 Counter instructions

4.10.1 CTU

CTU	Up counter
-----	------------

Instructions		Available Device										Steps	Flag			
		M	P	K	L	F	T	C	S	D	#D		Integer	Error (F110)	Zero (F111)	Carry (F112)
CTU	Cxxx							O					3			
	\bigcirc									O		O				

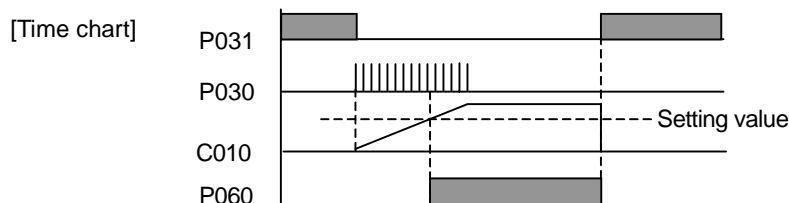
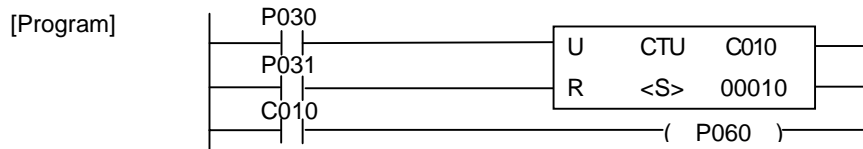


1) Functions

- Whenever a rising edged is detected at the count pulse input, the current value is increased by 1
- The initial current value is 0 and when the current value is reached to the setting value, the counter contact turns on.
- After the counter contact turns on, the current value keeps increasing until its maximum value. (65535)
- When the reset signal is switched on, the counter contact and current value is cleared as 0.

2) Program example

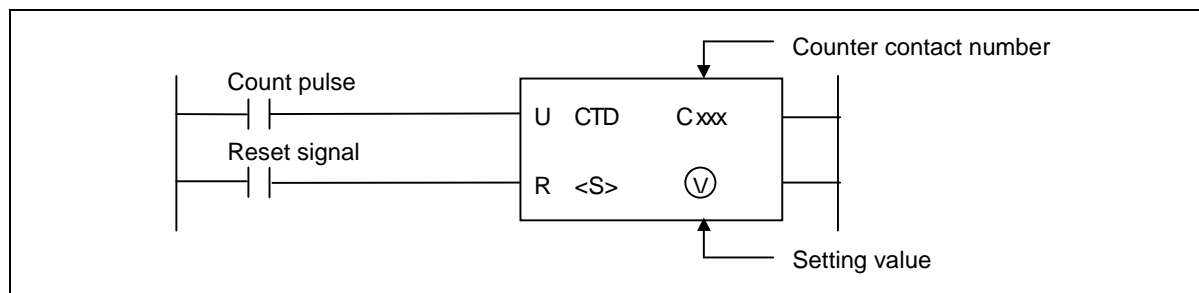
- Whenever the P030 is changed from off to on, the current value of C010 is increased by 1.
- The P031 is reset condition.



4.10.2 CTD

CTD	Down counter
-----	--------------

Instructions		Available Device											Steps	Flag		
		M	P	K	L	F	T	C	S	D	#D	Integer		Error (F110)	Zero (F111)	Carry (F112)
CTD	Cxxx							O					3			
	(V)									O		O				

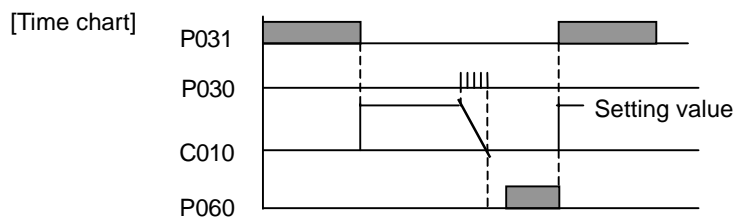
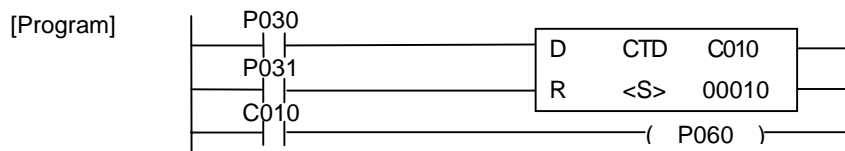


1) Functions

- Whenever the rising edge is detected from counter pulse input, the current value is decreased by 1.
- The initial current value is the setting value, and when the current value reached to 0, the counter contact is switched on.
- When the reset signal turns on, the counter contact is switched off and the current value is reset as the setting value.

2) Program example

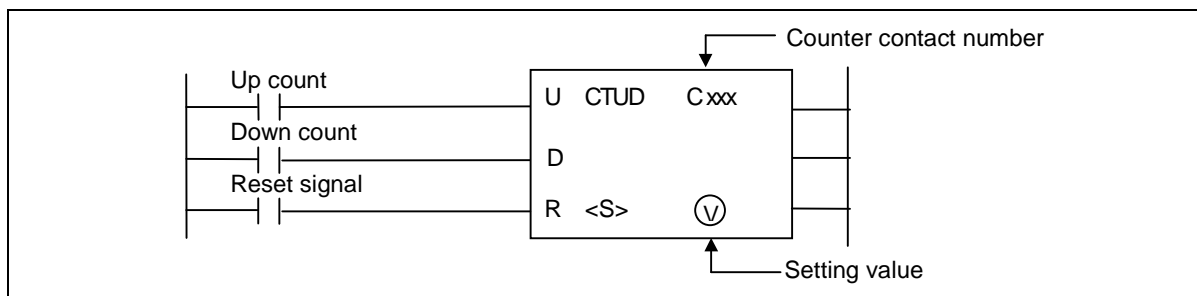
- Whenever the P030 is changed from off to on, the current value of C010 is decreased by 1.
- The P031 is reset condition.



4.10.3 CTUD

CTUD	Up-down counter
------	-----------------

Instructions		Available Device											Steps	Flag		
		M	P	K	L	F	T	C	S	D	#D	Integer		Error (F110)	Zero (F111)	Carry (F112)
CTUD	Cxxx							O					3			
	(V)									O		O				

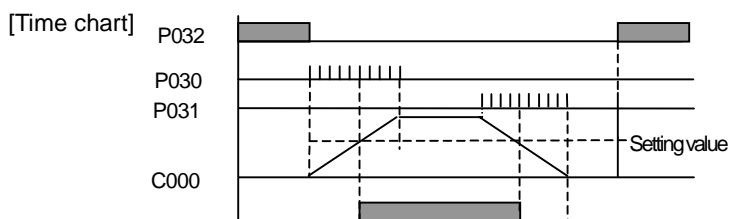
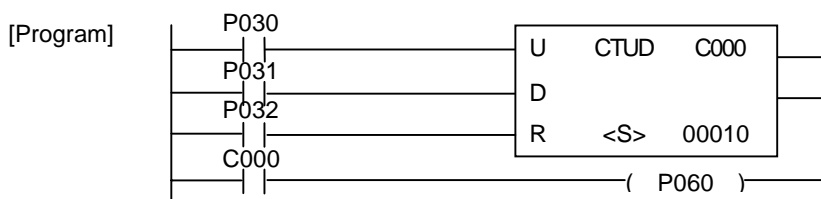


1) Functions

- Whenever a rising edged is detected from up count input, the current value is increased by 1.
- The current value is decreased by 1 whenever a rising edge is detected at the down count input.
- The initial current value is 0.
- The counter contact turns on when the current value is same or greater than the setting value.
- When the reset signal turns on, the counter contact and current value is cleared as 0.

2) Program example

- The P030 is up count input, and the P031 is down count input.
- The P032 is reset signal.

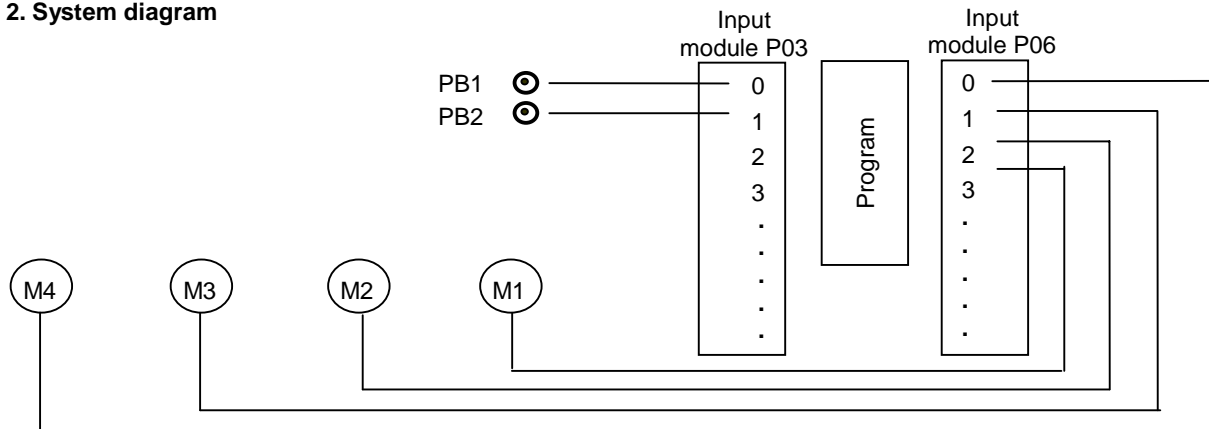


A control circuit for motor operation (example of CTUD instruction)

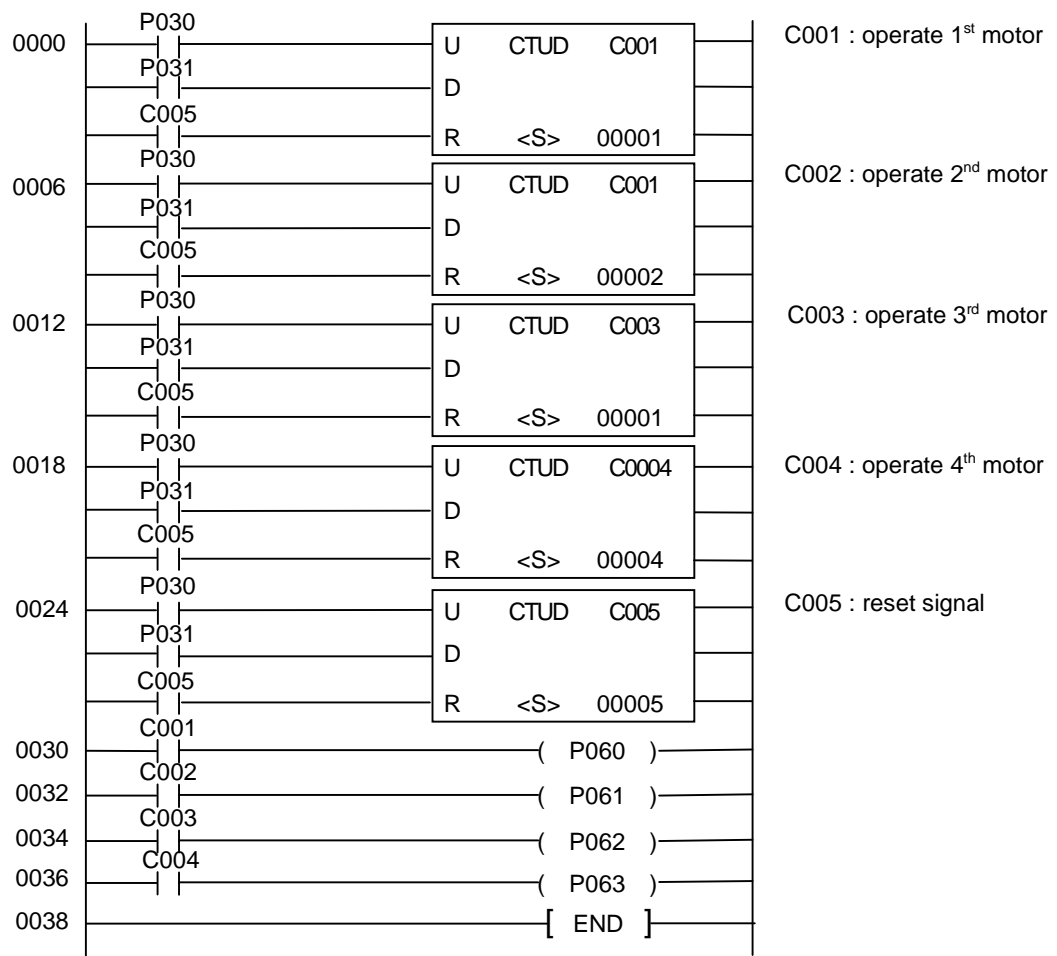
1. Operation

There are 4 motors controlled by PLC. Whenever the push-button PB1 is pressed, the numbers of operating motor is increased by 1. The PB2 decreases the numbers of operating motor whenever it is pressed. If the PB1 is pushed when 4 motors are operating, all motors will stop their operation.

2. System diagram



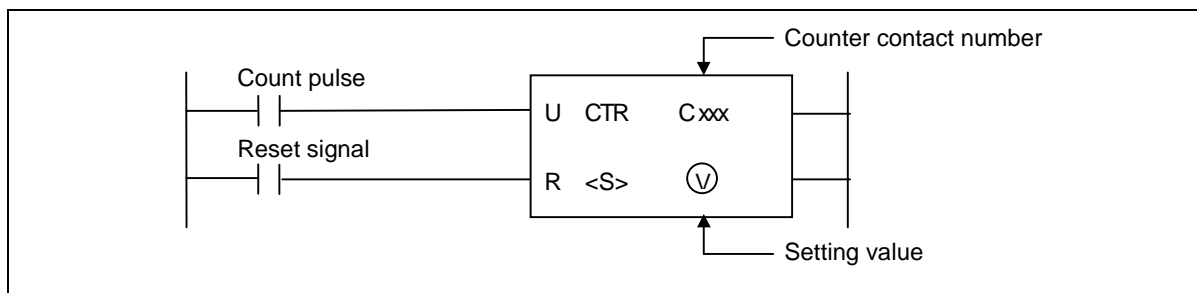
3. Program



4.10.4 CTR

CTR	Ring counter
-----	--------------

Instructions		Available Device										Steps	Flag			
		M	P	K	L	F	T	C	S	D	#D		Integer	Error (F110)	Zero (F111)	Carry (F112)
CTR	Cxxx							O					3			
	Ⓥ									O		O				



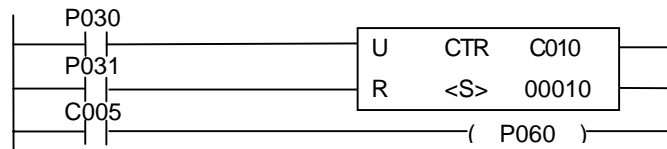
1) Functions

- Whenever a rising edge is detected at the count pulse input, the current value is increased by 1.
- If the current value is reached to the setting value, the counter contact is switched on. Then the counter contact and current value will be cleared as 0 when the next rising edge is applied to the count pulse input.
- When the reset signal turns on, the counter contact and current value will be cleared as 0.

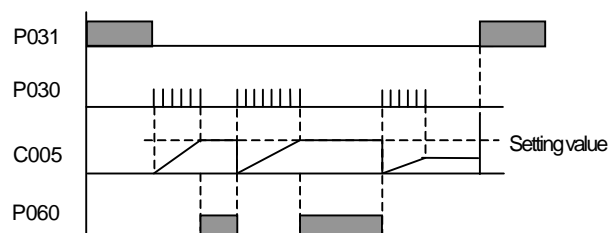
2) Program example

- The P030 is count pulse input and when the current value is same as the setting value, the counter contact is switched on.
- When the P030 is switched on 11th time, the counter contact (P060) is off and the current value is cleared as 0.

[Program]



[Time chart]



Chapter 5 Application instructions

5.1 Data transfer instructions.....	5-1
5.1.1 MOV, MOVP, DMOV, DMOVP	5-1
5.1.2 CMOV, CMOVP, DCMOV, DCMOVP	5-3
5.1.3 GMOV, GMOV	5-5
5.1.4 FMOV, FMOV	5-7
5.1.5 BMOV, BMOV	5-9
5.2 Conversion instructions	5-11
5.2.1 BCD, BCDP, DBCD, DBCDP	5-11
5.2.2 BIN, BINP, DBIN, DBINP	5-14
5.3 Comparison instructions	5-16
5.3.1 CMP, CMPP, DCMP, DCMPP	5-16
5.3.2 TCMP, TCMPP, DTCMP, DTCMPP	5-19
LD (=, >, <, >=, <=, <>)	5-21
5.3.4 AND (=, >, <, >=, <=, <>)	5-22
5.3.5 OR (=, >, <, >=, <=, <>)	5-23
5.4 Increment/decrement operations	5-24
5.4.1 INC, INCP, DINC, DINCP	5-24
5.4.2 DEC, DECP, DDEC, DDECP	5-26
5.5 Rotation instructions	5-28
5.5.1 ROL, ROLP, DROL, DROL	5-28
5.5.2 ROR, RORP, DROR, DROR	5-30
5.5.3 RCL, RCLP, DRCL, DRCL	5-32
5.5.4 RCR, RCRP, DRCR, DRCR	5-34
5.6 Shift instructions	5-36
5.6.1 BSFT, BSFTP	5-36
5.6.2 WSFT, WSFTP	5-38
5.6.3 SR	5-40
5.7 Exchange instructions	5-42

5.7.1	XCHG, XCHGP, DXCHG, DXCHGP	5-42
5.8	BIN arithmetic instructions	5-44
5.8.1	ADD, ADDP, DADD, DADDP	5-44
5.8.2	SUB, SUBP, DSUB, DSUBP	5-46
5.8.3	MUL, MULP, DMUL, DMULP	5-48
5.8.4	MUL, MULP, DMUL, DMULP	5-50
5.8.5	DIV, DIVP, DDIV, DDIVP	5-52
5.8.6	DIVS, DIVSP, DDIVS, DDIVSP	5-54
5.9	BCD arithmetic instructions	5-56
5.9.1	ADDB, ADDBP, DADDB, DADDBP	5-56
5.9.2	SUBB, SUBBP, DSUBB, DSUBBP	5-58
5.9.3	MULB, MULBP, DMULB, DMULBP	5-60
5.9.4	DIVB, DIVBP, DDIVB, DDIVBP	5-62
5.10	Logical arithmetic instructions	5-64
5.10.1	WAND, WANDP, DWAND, DWANDP	5-64
5.10.2	WOR, WORP, DWOR, DWORP	5-66
5.10.3	WXOR, WXORP, DWXOR, DWXORP	5-68
5.10.4	WXNR, WXNRP, DWXNR, DWXNRP	5-70
5.11	Data processing instructions	5-72
5.11.1	SEG, SEGP	5-72
5.11.2	ASC, ASCP	5-75
5.11.3	BSUM, BSUMP, DBSUM, DBSUMP	5-77
5.11.4	ENCO, ENCOF	5-79
5.11.5	DECO, DECOP	5-81
5.11.6	FILR, FILRP, DFILR, DFILRP	5-83
5.11.7	FILW, FILWP, DFILW, DFILWP	5-85
5.11.8	DIS, DISP	5-87
5.11.9	UNI, UNIP	5-89
5.11.10	IORF, IORFP	5-91
5.12	System instructions	5-93
5.12.1	FALS	5-93
5.12.2	DUTY	5-94
5.12.3	WDT, WDTP	5-96
5.12.4	OUTOFF	5-98
5.12.5	STOP	5-99
5.13	Branch instructions	5-100
5.13.1	JMP, JME	5-100
5.13.2	CALL, CALLP, SBRT, RET	5-102

5.14	Loop instructions.....	5-104
5.14.1	FOR, NEXT	5-104
5.14.2	BREAK	5-105
5.15	Flag instructions.....	5-106
5.15.1	STC, CLC	5-106
5.15.2	CLE	5-107
5.16	Special module instructions	5-108
5.16.1	GET, GETP	5-108
5.16.2	PUT, PUTP	5-110
5.17	Data link instructions	5-112
5.17.1	READ.....	5-112
5.17.2	WRITE.....	5-115
5.17.3	RGET.....	5-117
5.17.4	RPUT.....	5-120
5.17.5	STATUS.....	5-122
5.18	Inturrupt instructions	5-123
5.18.1	EI, DI	5-123
5.18.2	TDINT, IRET	5-124
5.18.3	INT, IRET	5-125
5.19	Sign inversion instruction	5-126
5.19.1	NEG, NEGP, DNEG, DNEGP	5-126
5.20	Bit contact instructions	5-128
5.20.1	BLD, BLDN.....	5-128
5.20.2	BAND, BANDN	5-129
5.20.3	BAND, BANDN	5-130
5.20.4	BOUT.....	5-131
5.20.5	BSET, BRST	5-132
5.21	Computer link module instructions	5-133
5.21.1	SND.....	5-133
5.21.2	RCV.....	5-134
5.22	High speed counter instructions	5-135
5.22.1	HSCNT	5-135
5.22.2	HSC	5-137
5.23	RS-485 communication instructions	5-139
5.23.1	RECV.....	5-139
5.23.2	SEND.....	5-141

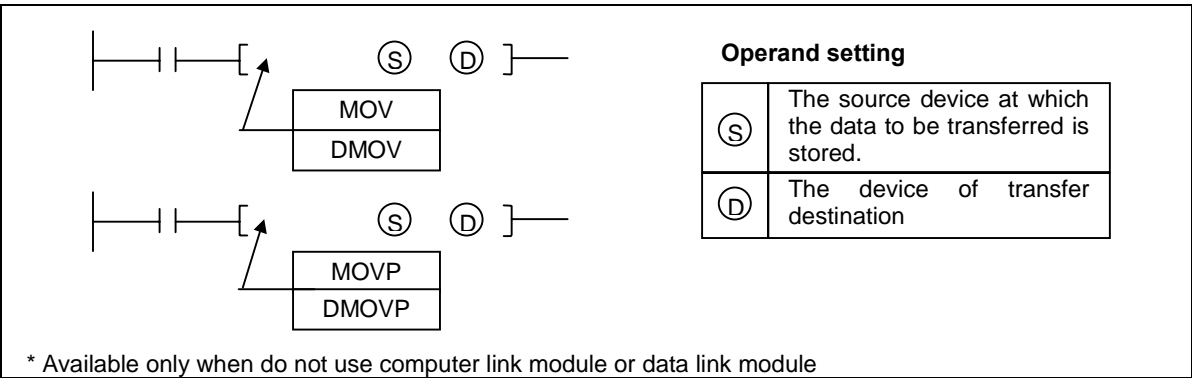
5 Application Instructions

5.1 Data transfer instructions

5.1.1 MOV, MOVP, DMOV, DMOVP

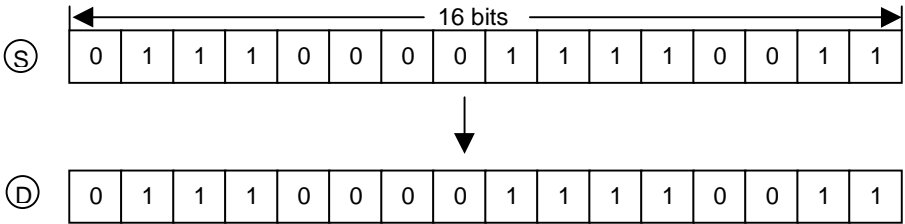
MOV (Move)	FUN(80) MOV FUN(82) DMOV FUN(81) MOVP FUN(83) DMOCP	Applicable CPU	All CPUs
---------------	--	-------------------	----------

Instructions		Available Device											Steps	Flag		
		M	P	K	L	F	T	C	S	D	#D	Integer		Error (F110)	Zero (F111)	Carry (F112)
MOV(P)	Ⓢ	O	O	O	O*	O	O	O		O	O	O	5/7	O		
DMOV(P)	ⓓ	O	O	O	O*		O	O		O	O					

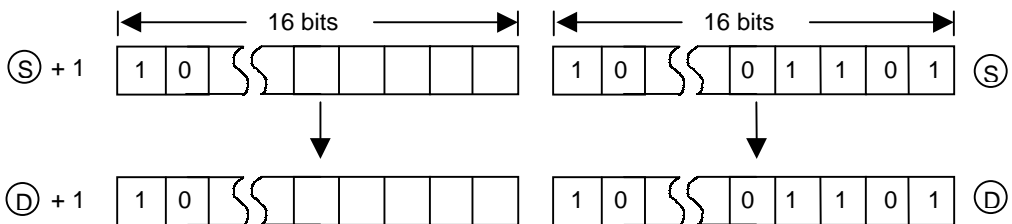


1) Functions

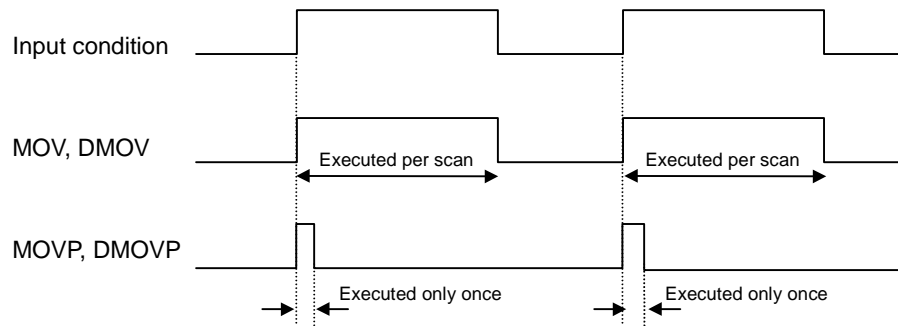
- MOV(P) : Transfer the 16-bits data of the device specified at [S] to the device specified at [D].



- DMOV(P) : Transfer the 32-bits data of the device specified at [S+1, S] to the device specified at [D+1, D].

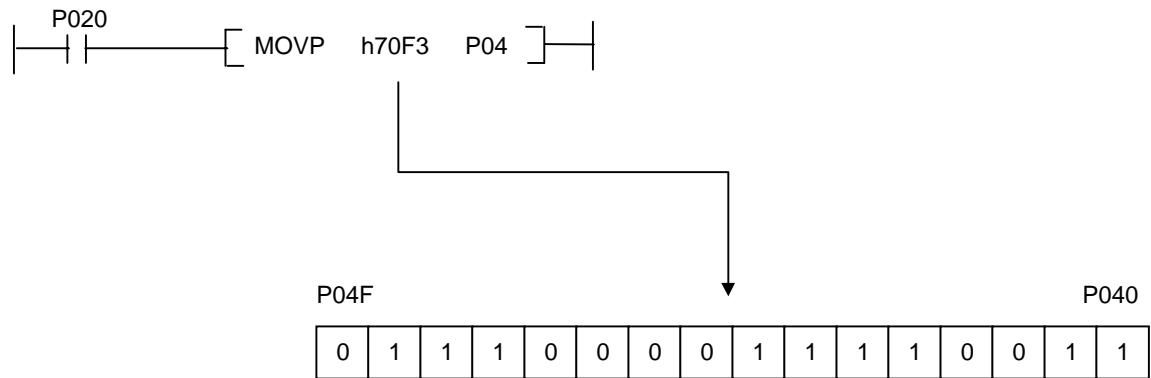


- Execution condition



2) Program example

Whenever a rising edge is detected at the P020, the 'h00F3' is moved to the P04 word.



5.1.2 CMOV, CMOVP, DCMOV, DCMOVP

CMOV (Complement move)	FUN(84) CMOV FUN(86) DCMOV FUN(85) CMOVP FUN(87) DCMOCP	Applicable CPU	All CPUs
---------------------------	--	-------------------	----------

Instructions		Available Device											Steps	Flag		
		M	P	K	L	F	T	C	S	D	#D	Integer		Error (F110)	Zero (F111)	Carry (F112)
CMOV(P)	Ⓢ	O	O	O	O*	O	O	O		O	O	O	5/7	O		
DCMOV(P)	ⓓ	O	O	O	O*		O	O		O	O					

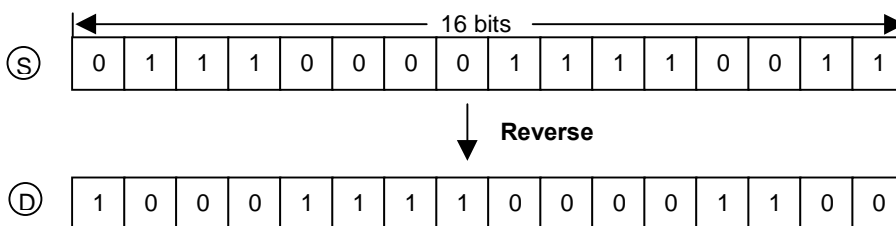
Operand setting

Ⓢ	The source device at which the data to be transferred is stored.
ⓓ	The device at which the reversed result is stored

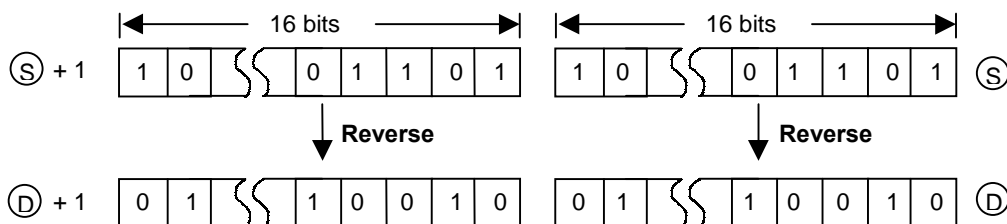
* Available only when do not use computer link module or data link module

1) Functions

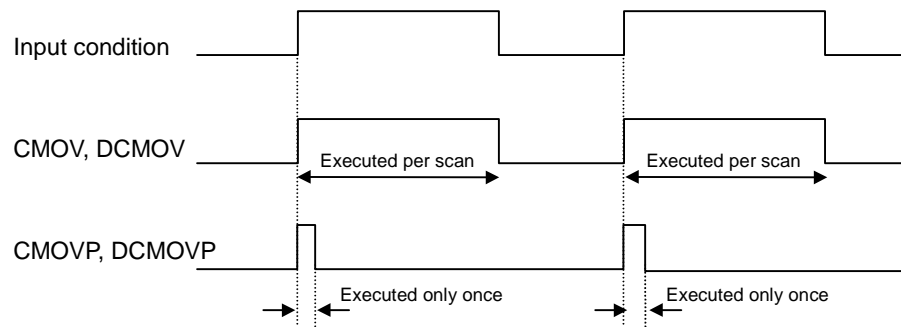
- CMOV(P) : Reverses the 16-bits data of [S] per bit and transfers the result to [D].



- DCMOV(P) : Reverses the 32-bits data of [S+1, S] per bit and transfers the result to [D+1, D].

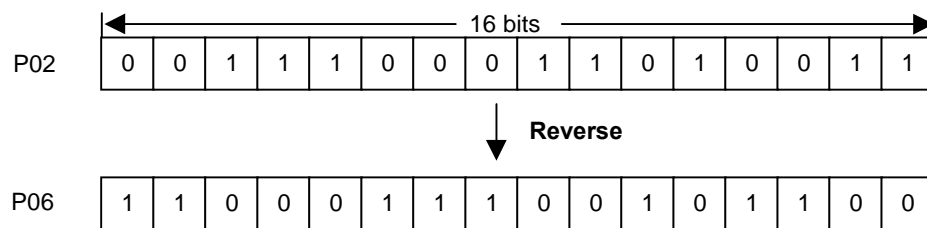


- Execution conditions



2) Program example

- While the M020 is on, reverse the data of P02 word and transfer the result to P06 word.



5.1.3 GMOV, GMOV P

GMOV	FUN(90) GMOV	Applicable CPU	All CPUs
(Group move)	FUN(91) GMOV P		

Instructions		Available Device											Steps	Flag		
		M	P	K	L	F	T	C	S	D	#D	Integer		Error (F110)	Zero (F111)	Carry (F112)
GMOV	Ⓢ	O	O	O	O*	O	O	O		O	O		7	O		
GMOV P	ⓓ	O	O	O	O*		O	O		O	O					
	n									O		O				

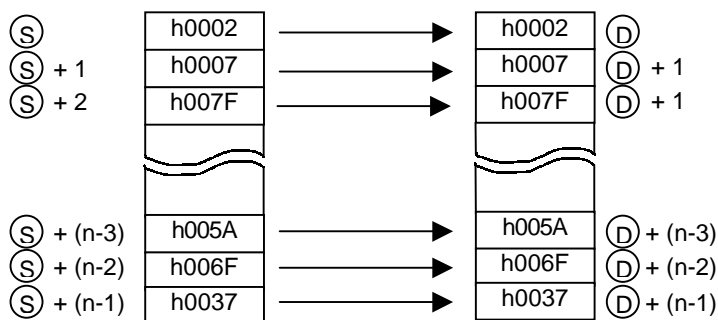
Operand setting

Ⓢ	The start address of source data area
ⓓ	The start address of destination area will store transferred data
n	Numbers of transferred words

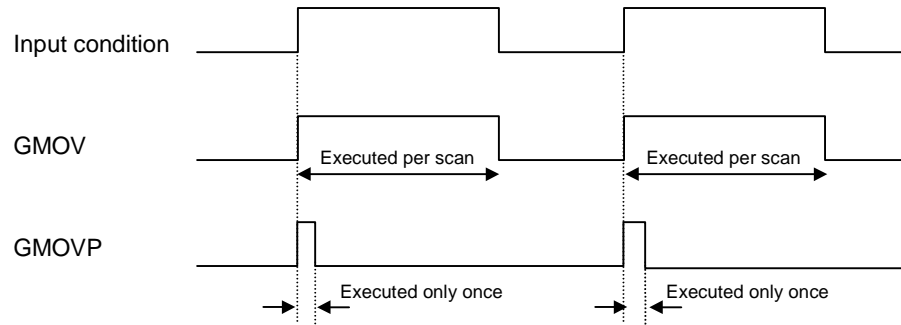
* Available only when do not use computer link module or data link module

1) Functions

- Transfers the content of 'n' words, which begin with the device specified at [S], in blocks to 'n' words which begin with the device specified at [D].

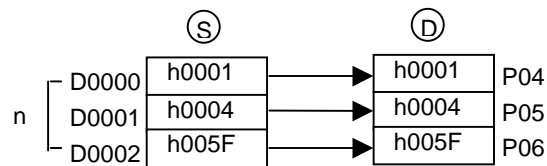


- Execution conditions



2) Program example

- While the P020 is on, move the data of D000, D001, and D002 to P04, P05, and P06 area.



5.1.4 FMOV, FMOVP

FMOV (File move)	FUN(92) FMOV FUN(93) FMOVP	Applicable CPU	All CPUs
---------------------	-------------------------------	-------------------	----------

Instructions		Available Device											Steps	Flag		
		M	P	K	L	F	T	C	S	D	#D	Integer		Error (F110)	Zero (F111)	Carry (F112)
FMOV FMOVP	Ⓢ	O	O	O	O*	O	O	O		O	O		7	O		
	Ⓓ	O	O	O	O*		O	O		O	O					
	n									O		O				

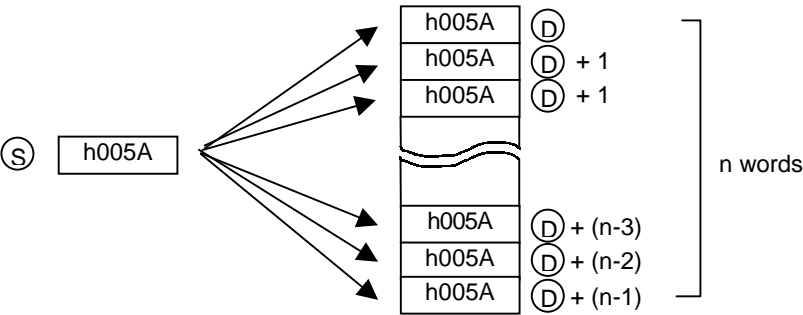
Operand setting

Ⓢ	The device at which source data area is stored
Ⓓ	The start address of destination area will store transferred data
n	Numbers of transferred words

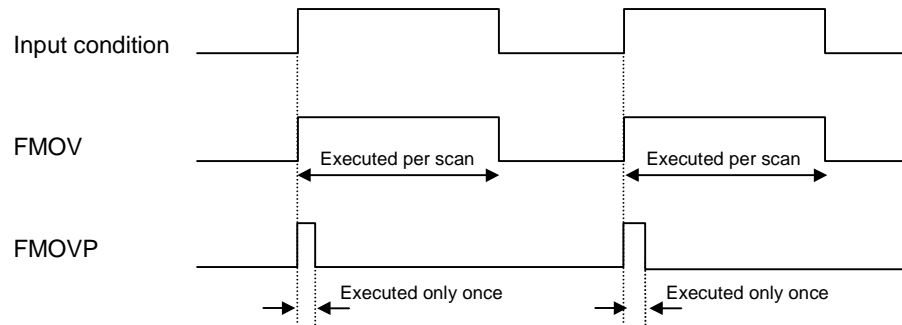
* Available only when do not use computer link module or data link module

1) Functions

- Transfers the content of device specified at [S] in blocks to 'n' points which begin with the device specified at [D].

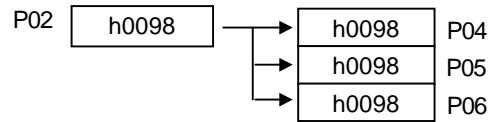


- Execution condition



2) Program example

- Whenever a rising edge is detected at P030, transfer the content of P02 word to the block of P04, P05, and P06.



5.1.5 BMOV, BMOVP

BMOV (Bit move)	FUN(100) BMOV FUN(101) BMOVP	Applicable CPU	All CPUs
--------------------	---------------------------------	-------------------	----------

Instructions		Available Device											Steps	Flag		
		M	P	K	L	F	T	C	S	D	#D	Integer		Error (F110)	Zero (F111)	Carry (F112)
BMOV	Ⓢ	O	O	O	O*	O	O	O		O	O		7	O		
BMOVP	ⓓ	O	O	O	O*		O	O		O	O					
	Cw											O				

The diagram illustrates the operand setting for BMOV and BMOVP. For BMOV, the source device [S] and destination device [D] are specified. For BMOVP, the source device [S], destination device [D], and the transfer bit count [Cw] are specified. Arrows point from the instruction boxes to the corresponding operand fields in the ladder logic diagrams.

Operand setting

Ⓢ	The device at which source data is stored
ⓓ	The device which will store transferred data
Cw	Information for start bit and numbers of transferred bits

* Available only when do not use computer link module or data link module

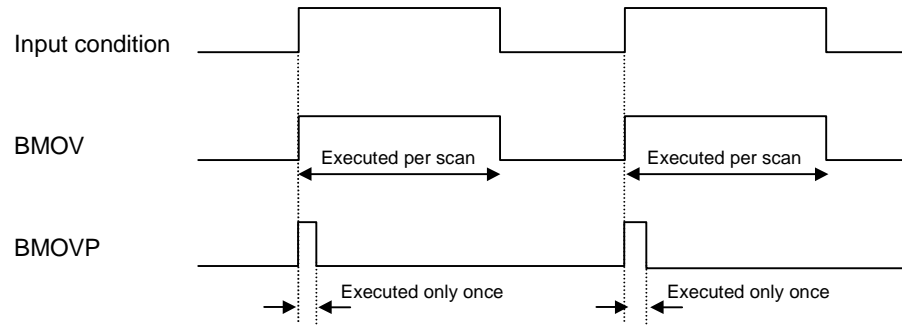
1) Functions

- The format of 'Cw'

h	s	d	z	z
---	---	---	---	---

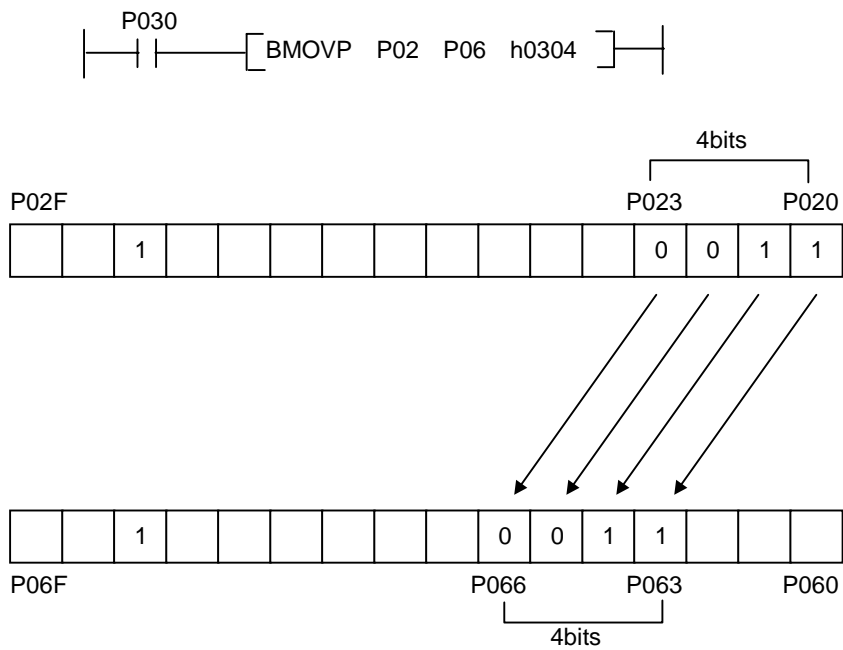
- s : The start bit of [S].
 - d : The start bit of [D]
 - zz : Numbers of transferred bits. (Hexadecimal)
- Transfers the content of 'zz' bits from the 's' bit of the device specified at [S] to the 'zz' bits from 'd' bit of the device specified at [D].
 - The maximum value of the 'zz' is h10 (= 16). If the value of 'zz' is 0 or over h10, the instruction will be ignored. (The error flag F110 is set when the 'zz' is over h10.)

- Execution condition



2) Program example

- Whenever a rising edge is detected at the P030, transfer 4 bits from the P020 bit to the 4 bits from the P063 bit.

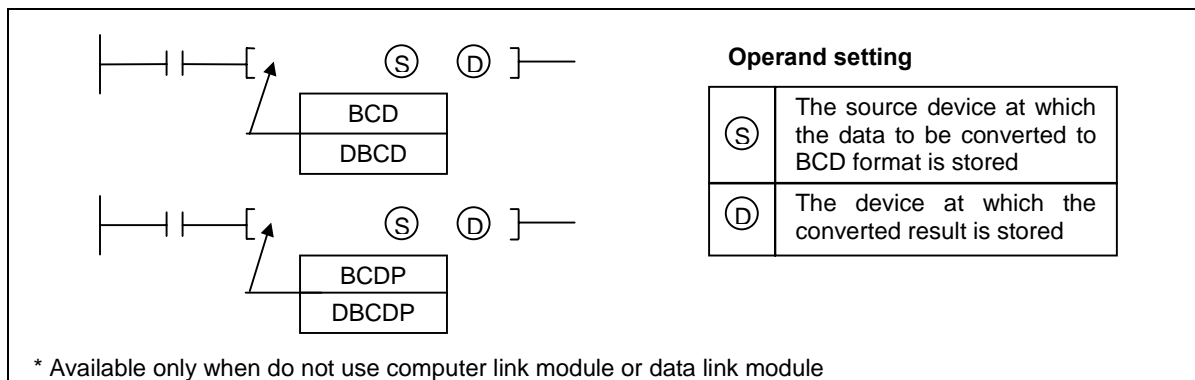


5.2 Conversion instructions

5.2.1 BCD, BCDP, DBCD, DBCDP

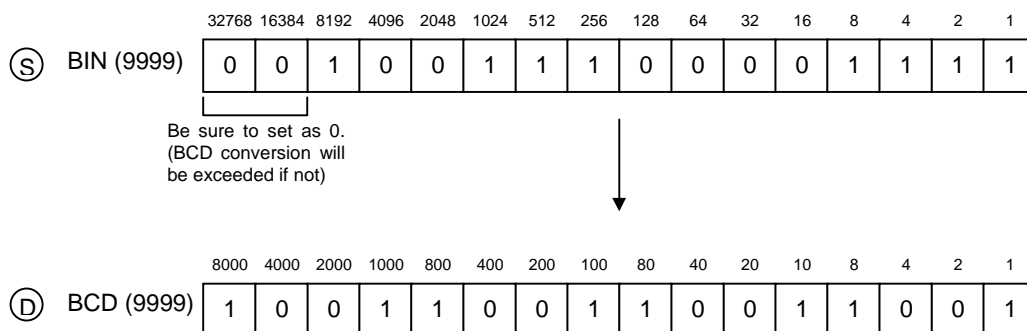
BCD (Binary coded decimal)	FUN(60) BCD FUN(62) DBCD FUN(61) BCDP FUN(63) DBCDP	Applicable CPU	All CPUs
-------------------------------	--	----------------	----------

Instructions		Available Device											Steps	Flag		
		M	P	K	L	F	T	C	S	D	#D	Integer		Error (F110)	Zero (F111)	Carry (F112)
BCD(P)	Ⓢ	O	O	O	O*	O	O	O		O	O		5	O		
DBCD(P)	ⓓ	O	O	O	O*		O	O		O	O					



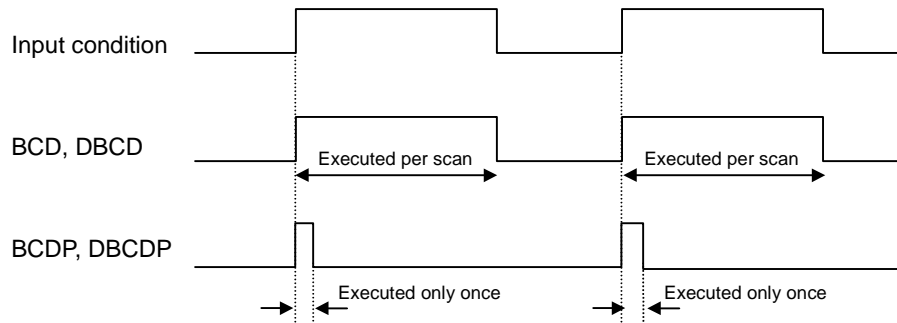
1) Functions

- BCD : Converts binary data (0 to 9999) of the device specified at [S] into BCD format and transfers the result to the device specified at [D].



- DBCD : Converts binary data (0 to 99999999) of the device specified at [S] into BCD format and transfers the result to the device specified at [D].

- Execution conditions



- Operation Error

In the following cases, operation error occurs and the error flag (F110) turns on.

- When BCD(P) instruction is used

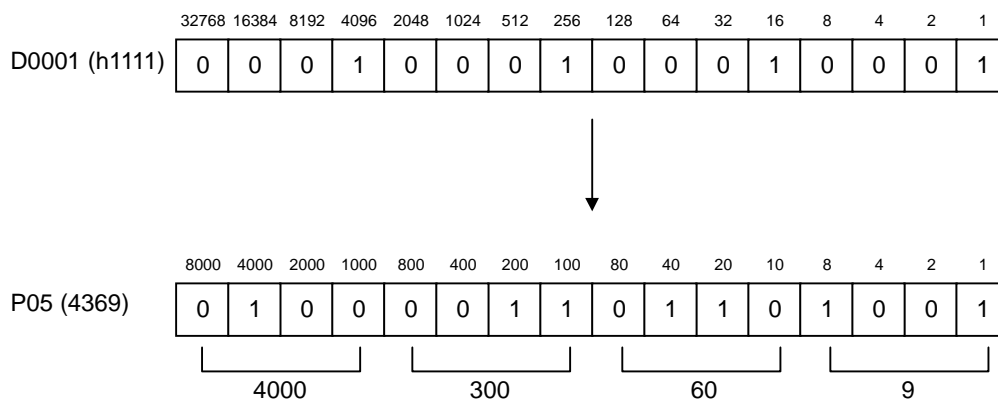
The data of source [S] is outside the range of 0 to 9999

- When DBCD(P) instruction is used

The data of source [S] is outside the range of 0 to 99999999

2) Program example

- While P020 is on, convert the binary data of D0001 and transfer the result to the P05 word.

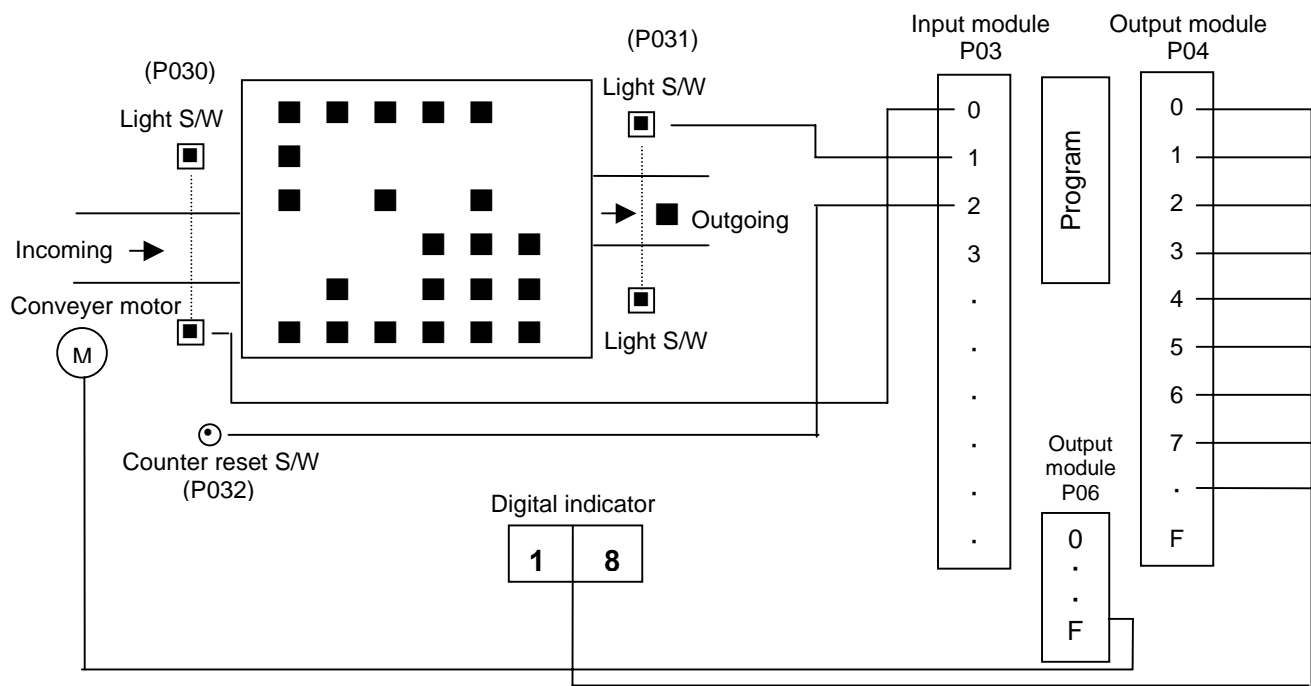


Displaying the current value of counter
(example of BCD, BMOV instructions)

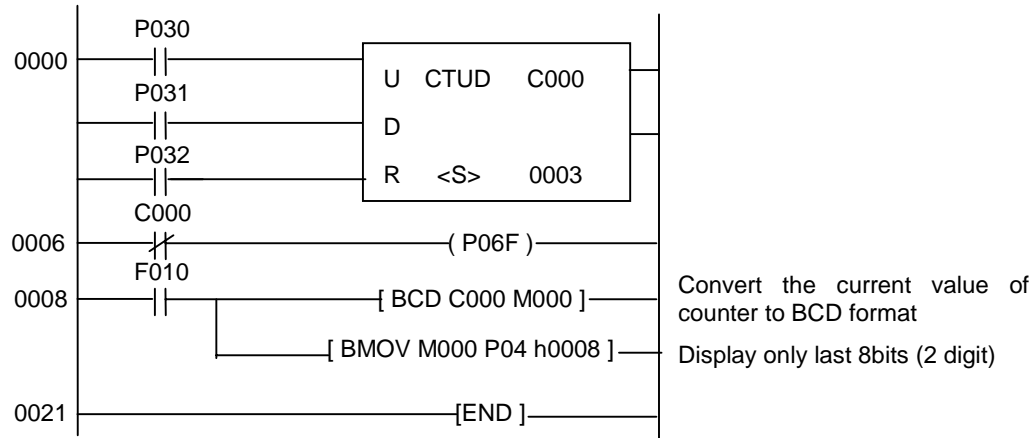
1. Operation

There is a warehouse and numbers of product incoming / outgoing are counted by light switches. The current stock of inside of warehouse is displayed by a digital indicator. When the stock of inside of warehouse reaches to 30, the incoming conveyer will be stopped.

2. System diagram



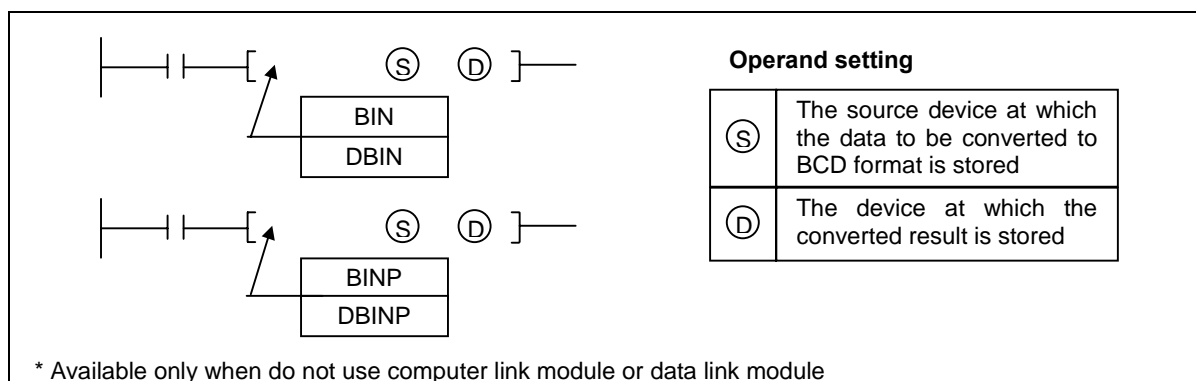
3. Program



5.2.2 BIN, BINP, DBIN, DBINP

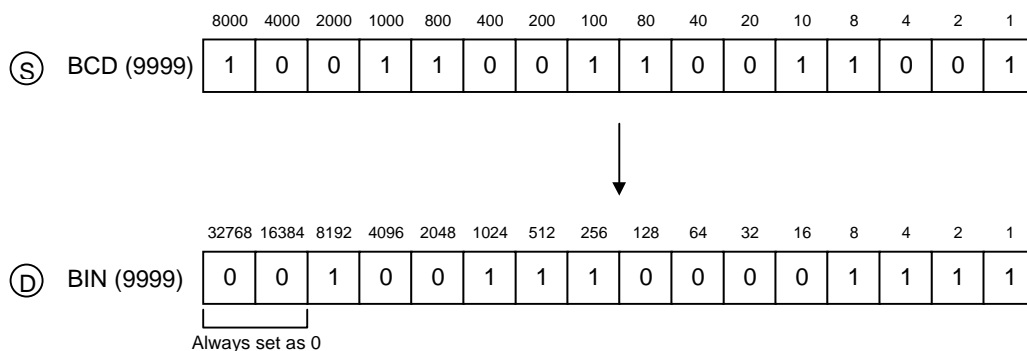
BIN (Binary)	FUN(64) BIN	FUN(66) DBIN	Applicable CPU	All CPUs
	FUN(65) BINP	FUN(67) DBINP		

Instructions		Available Device											Steps	Flag		
		M	P	K	L	F	T	C	S	D	#D	Integer		Error (F110)	Zero (F111)	Carry (F112)
BIN(P)	Ⓢ	O	O	O	O*	O	O	O		O	O		5	O		
DBIN(P)	ⓓ	O	O	O	O*		O	O		O	O					



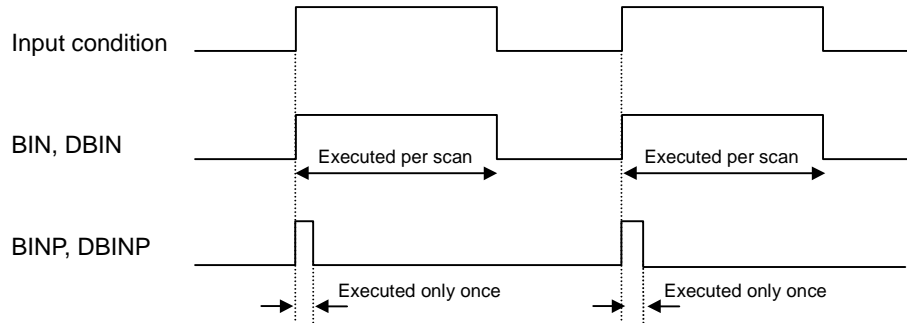
1) Functions

- BIN : Converts BCD data (0 to 9999) of the device specified at [S] into binary format and transfers the result to the device specified at [D].



- DBIN : Converts BCD data (0 to 99999999) of the device specified at [S] into binary format and transfers the result to the device specified at [D].

- Execution conditions



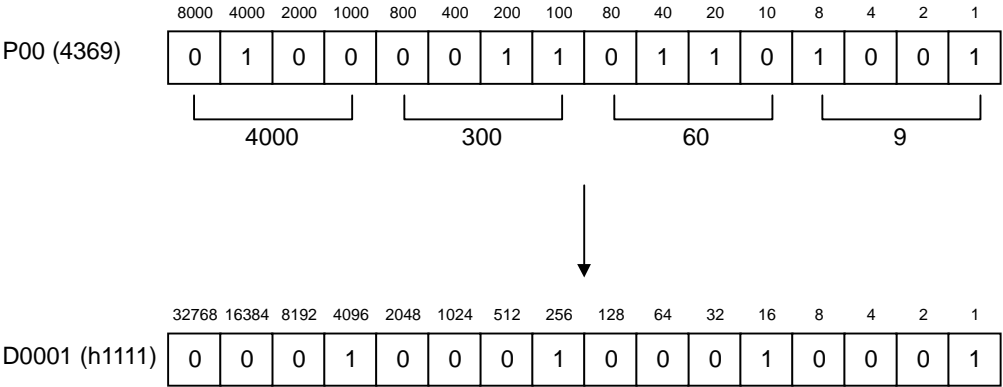
- Operation Error

In the following cases, operation error occurs and the error flag (F110) turns on.

- a) Each digit (4 bits) of source [S] is outside the range of 0 to 9
(Example : [S] = h78A5)

2) Program example

- While P020 is on, convert the BCD data of P00 word and transfer the result to the D0001.

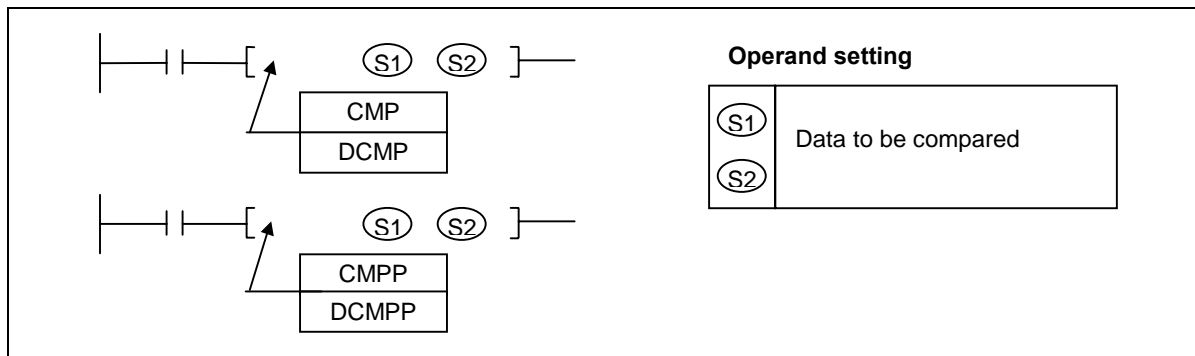


5.3 Comparison instructions

5.3.1 CMP, CMPP, DCMPP, DCMPP

CMP (Compare)	FUN(50) CMP FUN(51) CMPP	FUN(52) DCMPP FUN(53) DCMPP	Applicable CPU	All CPUs
------------------	-----------------------------	--------------------------------	-------------------	----------

Instructions		Available Device											Steps	Flag		
		M	P	K	L	F	T	C	S	D	#D	Integer		Error (F110)	Zero (F111)	Carry (F112)
CMP(P)	(S1)	O	O	O	O	O	O	O		O	O	O	5 / 9	O		
DCMP(P)	(S2)	O	O	O	O	O	O	O		O	O	O				



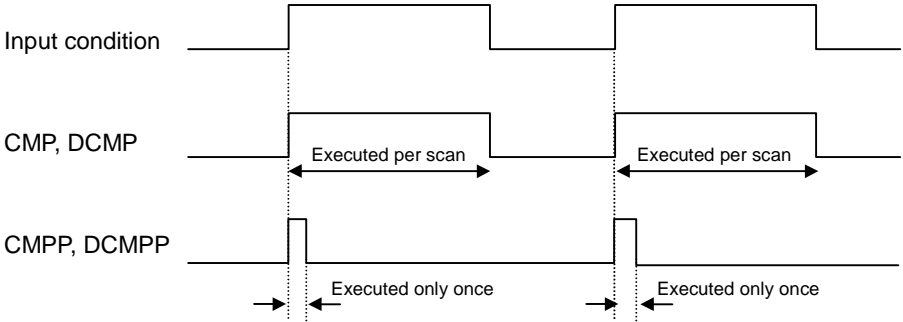
1) Functions

- Compares contents of two devices specified at [S1] and [S2].
- After comparing, set corresponding flag among F120 ~ F125.

Flag	F120	F121	F122	F123	F124	F125
	<	≤	=	>	≥	≠
(S1) > (S2)	0	0	0	1	1	1
(S1) < (S2)	1	1	0	0	0	1
(S1) = (S2)	0	1	1	0	1	0

- Above flags indicate the result of CMP instruction executed at last.
- The error flag (F110) is set when [S1] or [S2] specified as #D format is over device range. The instruction at which error occurred is not executed.

- Execution conditions



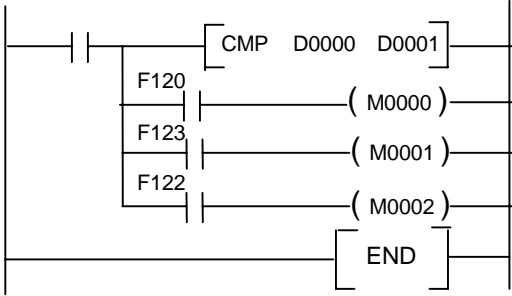
2) Program example

- While P020 is on, compare contents of D000 and D001 and set flags according to the result.

(D0000) 0 0 0 } { 1 0 0 0 (h0008)

(D0001) 0 0 1 } { 0 0 0 1 (h2001)

[Program]



[Flag setting]

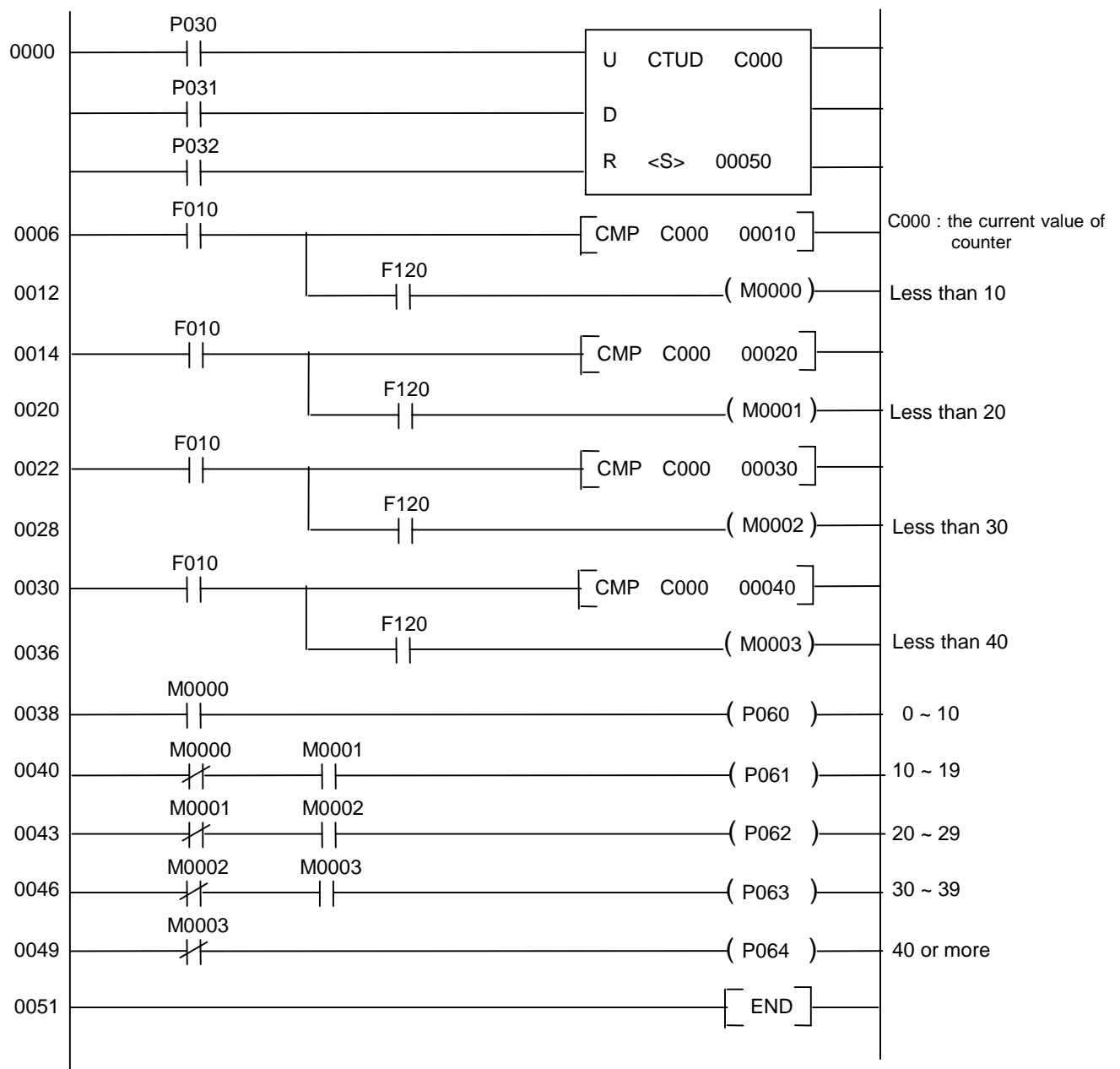
Flag	F120	F121	F122	F123	F124	F125
	<	≤	=	>	≥	≠
Result	1	1	0	0	0	1

Comparing circuit (example of CMP instruction)

1. Operation

There is a up-down counter C000. P030 is up-count input, and P031 is down-count input. If the current value of timer is 0 ~ 9, P060 turns on. If the current value is 10 ~ 19, P061 will be on. P062 will turn on when 20 ~ 29, P063 will turn on when 30 ~ 39, and P064 will be on when the current value is 40 or larger.

2. Program



5.3.2 TCMP, TCMPP, DTCMP, DTCMPP

TCMP (Table compare)	FUN(54) TCMP FUN(56) DTCMP FUN(55) TCMPP FUN(57) DTCMPP	Applicable CPU	All CPUs
-------------------------	--	-------------------	----------

Instructions		Available Device											Steps	Flag		
		M	P	K	L	F	T	C	S	D	#D	Integer		Error (F110)	Zero (F111)	Carry (F112)
TCMP(P) DTCMP(P)	(S1)	O	O	O	O	O	O	O		O	O	O	7 / 9	O	O	
	(S2)	O	O	O	O	O	O	O		O	O					
	(D)	O	O	O	O*		O	O		O	O					

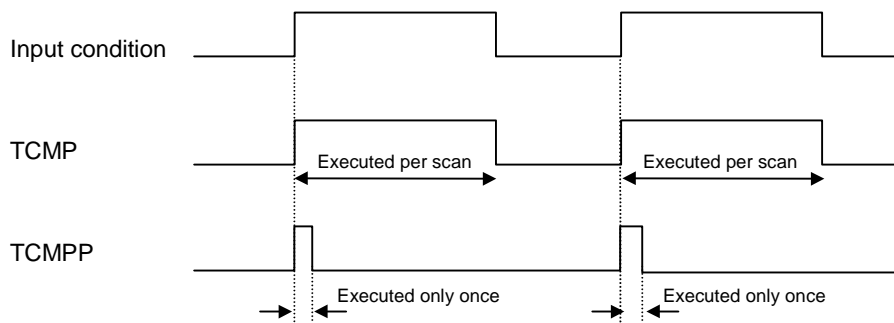
Operand setting

(S1)	Data to be compared
(S2)	The start address of block to be compared with (S1)
(D)	The device at which the comparison result is stored

* Available only when do not use computer link module or data link module

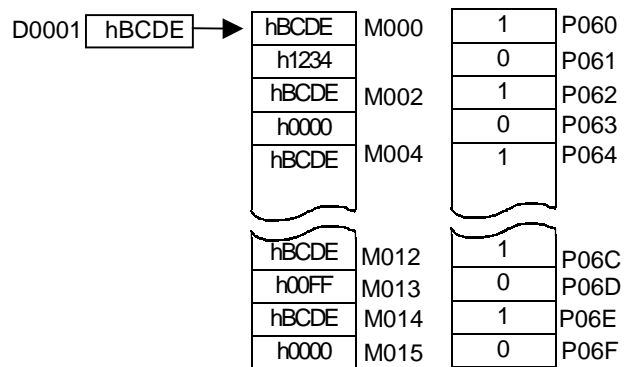
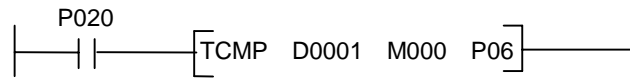
1) Functions

- Compares content of the device specified at [S1] with each contents of 16 words from the device specified at [S2].
- The comparison result (If two words are same, 1 is output. If not, 0 is output) consists of 16 bits, and they are stored at the device specified at [D].
- If all comparison result are 0, then the zero flag (F111) is switched on. ([D] = 0)
- Execution conditions



2) Program example

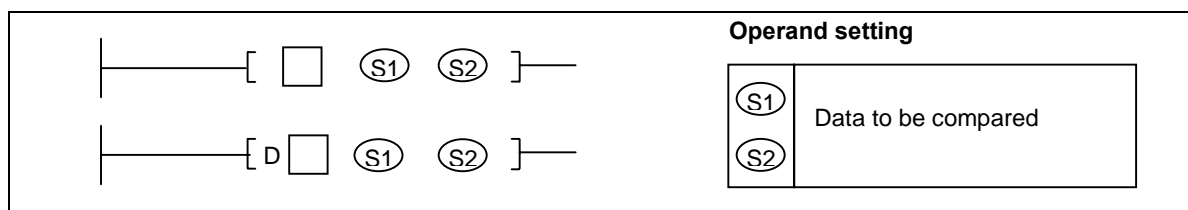
- While P020 is on, compare the content of D0001 with 16 words from M00 (M00 ~ M15) and output the comparison result to P06 word (P060 ~ P06F).



5.3.3 LD (=, >, <, >=, <=, <>)

LD <input type="checkbox"/> (Start NO contact with comparison result)	FUN(28) LD=	FUN(29) LDD=	Applicable CPU	K200S K300S K1000S
	FUN(38) LD>	FUN(39) LDD>		
	FUN(48) LD<	FUN(49) LDD<		
	FUN(58) LD>=	FUN(59) LDD>=		
	FUN(68) LD<=	FUN(69) LDD<=		
	FUN(78) LD<>	FUN(79) LDD<>		

Instructions		Available Device											Steps	Flag		
		M	P	K	L	F	T	C	S	D	#D	Integer		Error (F110)	Zero (F111)	Carry (F112)
LD <input type="checkbox"/>	(S1)	O	O	O	O	O	O	O		O	O	O	5 / 9	O		
LDD <input type="checkbox"/>	(S2)	O	O	O	O	O	O	O		O	O	O				



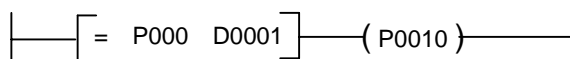
1) Functions

- Compares contents of [S1] and [S2], then operates as NO contact according to the comparison operation result. The comparison is executed as signed operation.
- The comparison operation result is as shown below :

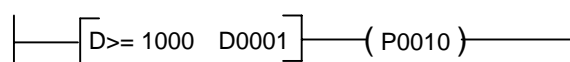
Instruction symbol in <input type="checkbox"/>	Condition	Comparison operation result	Instruction symbol in <input type="checkbox"/>	Condition	Comparison operation result
=	$S_1 = S_2$	On	=	$S_1 \neq S_2$	Off
<=	$S_1 \leq S_2$		<=	$S_1 > S_2$	
>=	$S_1 \geq S_2$		>=	$S_1 < S_2$	
<>	$S_1 \neq S_2$		<>	$S_1 = S_2$	
<	$S_1 < S_2$		<	$S_1 \geq S_2$	
>	$S_1 > S_2$		>	$S_1 \leq S_2$	

2) Program example

- a) Compare P0000 ~ P000F and D0001. If their values are equal, P0010 bit will be switched on.



- b) Compare 1000 and contents of D0001 and D0002 (32 bits). If the contents of D0001 and D0002 is less than 1000, P0010 will turn on.



5.3.4 AND (=, >, <, >=, <=, <>)

AND <input type="checkbox"/> (Serial NO contact with comparison result)	FUN(94) AND=	FUN(95) ANDD=	Applicable CPU K200S K300S K1000S
	FUN(96) AND>	FUN(97) ANDD>	
	FUN(98) AND<	FUN(99) ANDD<	
	FUN(106) AND>=	FUN(107) ANDD>=	
	FUN(108) AND<=	FUN(109) ANDD<=	
	FUN(118) AND<>	FUN(119) ANDD<>	

Instructions		Available Device											Steps	Flag		
		M	P	K	L	F	T	C	S	D	#D	Integer		Error (F110)	Zero (F111)	Carry (F112)
AND <input type="checkbox"/>	(S1)	O	O	O	O	O	O	O		O	O	O	5 / 9	O		
ANDD <input type="checkbox"/>	(S2)	O	O	O	O	O	O	O		O	O	O				

<div> <div>— [<input type="checkbox"/> (S1) (S2)] —</div> <div>— [D <input type="checkbox"/> (S1) (S2)] —</div> </div>		Operand setting <div> <div>(S1)</div> <div>(S2)</div> </div> <div>Data to be compared</div>
--	--	---

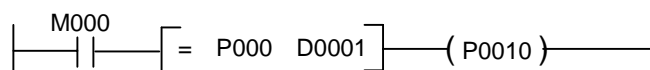
1) Functions

- Compares contents of [S1] and [S2], then operates as a serial NO contact according to the comparison operation result. The comparison is executed as signed operation.
- The comparison operation result is as shown below :

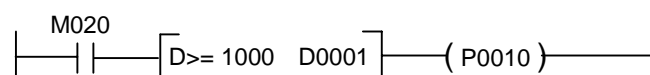
Instruction symbol in <input type="checkbox"/>	Condition	Comparison operation result	Instruction symbol in <input type="checkbox"/>	Condition	Comparison operation result
=	$S_1 = S_2$	On	=	$S_1 \neq S_2$	Off
<=	$S_1 \leq S_2$		<=	$S_1 > S_2$	
>=	$S_1 \geq S_2$		>=	$S_1 < S_2$	
<>	$S_1 \neq S_2$		<>	$S_1 = S_2$	
<	$S_1 < S_2$		<	$S_1 \geq S_2$	
>	$S_1 > S_2$		>	$S_1 \leq S_2$	

2) Program example

- a) Compare P0000 ~ P000F and D0001. If their values are equal and M000 is on, P0010 bit will be switched on.



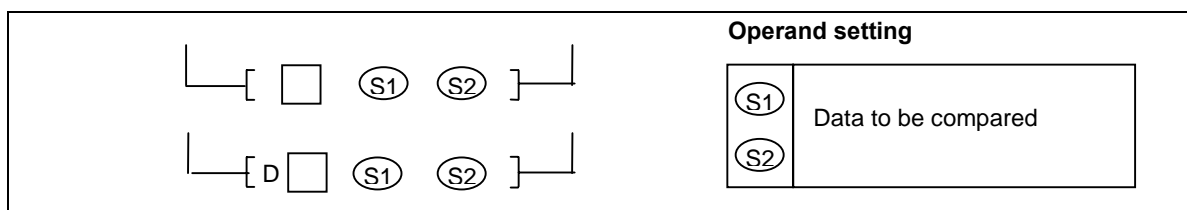
- b) Compare 1000 and contents of D0001 and D0002 (32 bits). If the content of D0001 and D0002 is less than 1000 and M020 is on, P0010 will turn on.



5.3.5 OR (=, >, <, >=, <=, <>)

OR <input type="checkbox"/> (Parallel NO contact with comparison result)	FUN(188) OR= FUN(189) ORD=	Applicable CPU	K200S K300S K1000S
	FUN(196) OR> FUN(197) ORD>		
	FUN(198) OR< FUN(199) ORD<		
	FUN(216) OR>= FUN(217) ORD>=		
	FUN(218) OR<= FUN(219) ORD<=		
	FUN(228) OR<> FUN(229) ORD<>		

Instructions		Available Device											Steps	Flag		
		M	P	K	L	F	T	C	S	D	#D	Integer		Error (F110)	Zero (F111)	Carry (F112)
OR <input type="checkbox"/>	(S1)	O	O	O	O	O	O	O		O	O	O	5 / 9	O		
ORD <input type="checkbox"/>	(S2)	O	O	O	O	O	O	O		O	O	O				



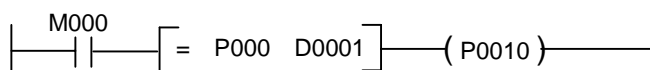
1) Functions

- Compares contents of [S1] and [S2], then operates as a parallel NO contact according to the comparison operation result. The comparison is executed as signed operation.
- The comparison operation result is as shown below :

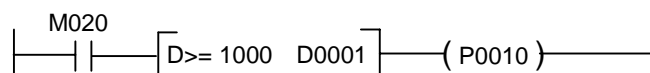
Instruction symbol in <input type="checkbox"/>	Condition	Comparison operation result	Instruction symbol in <input type="checkbox"/>	Condition	Comparison operation result
=	$S_1 = S_2$	On	=	$S_1 \neq S_2$	Off
<=	$S_1 \leq S_2$		<=	$S_1 > S_2$	
>=	$S_1 \geq S_2$		>=	$S_1 < S_2$	
<>	$S_1 \neq S_2$		<>	$S_1 = S_2$	
<	$S_1 < S_2$		<	$S_1 \geq S_2$	
>	$S_1 > S_2$		>	$S_1 \leq S_2$	

2) Program example

- a) Compare P0000 ~ P000F and D0001. If their values are equal or M000 is on, P0010 bit will be switched on.



- b) Compare 1000 and contents of D0001 and D0002 (32 bits). If the content of D0001 and D0002 is less than 1000 or M020 is on, P0010 will turn on.

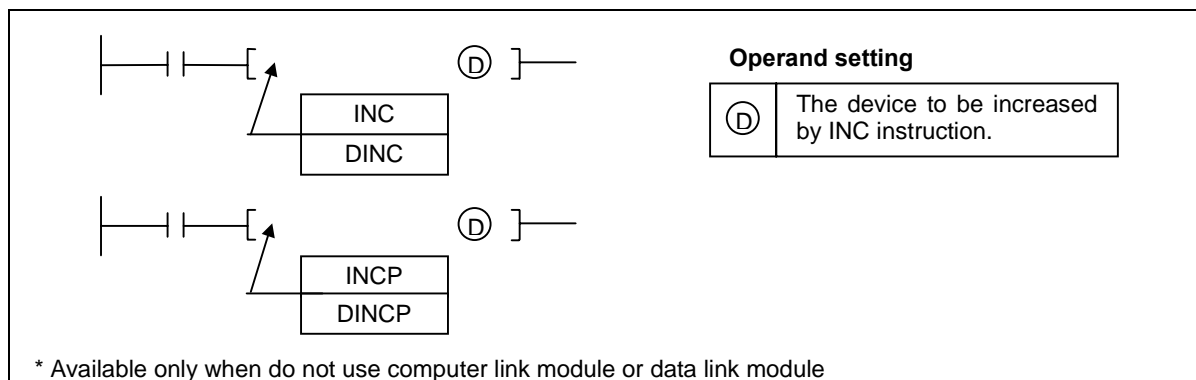


5.4 Increment/decrement operations

5.4.1 INC, INCP, DINC, DINCP

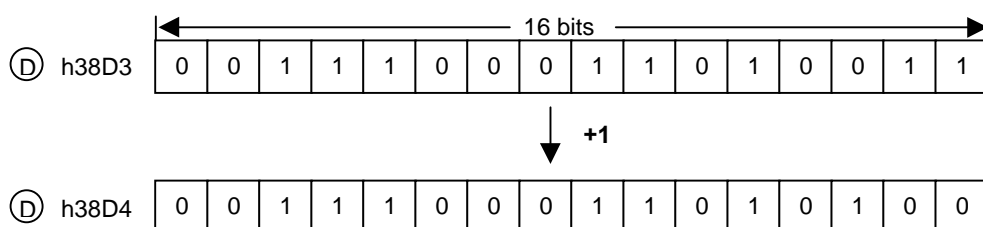
INC (Increment)	FUN(20) INC FUN(21) INCP	FUN(22) DINC FUN(23) DINCP	Applicable CPU	All CPUs
--------------------	-----------------------------	-------------------------------	-------------------	----------

Instructions		Available Device										Steps	Flag		
		M	P	K	L	F	T	C	S	D	#D		Error (F110)	Zero (F111)	Carry (F112)
INC(P)	Ⓓ	O	O	O	O*		O	O		O	O	3	O	O	O
DINC(P)															

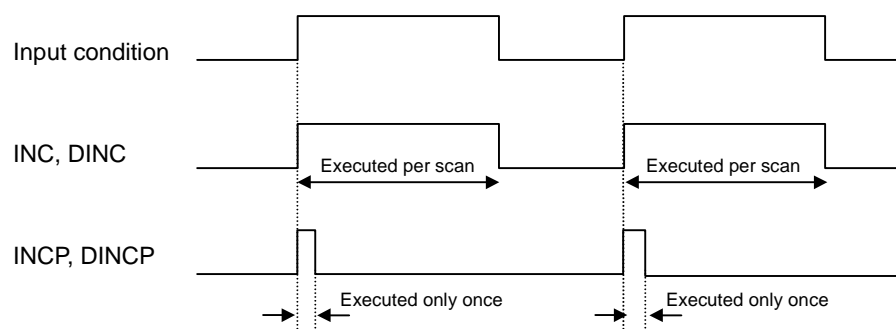


1) Functions

- INC(P) : Performs the addition of 1 to the device (16-bits data) specified at [D].
- DINC(P) : Performs the addition of 1 to the device (32-bits data) [D+1, D].
- If the INC(P) or DINC(P) is executed when the content of device is hFFFF or hFFFFFFFF, the content of device will be 0. At the same time, the zero flag (F111) and the carry flag (F112) are set.
- If the device specified by #D is out of the range, the operation error occurs and the error flag (F110) will be set.

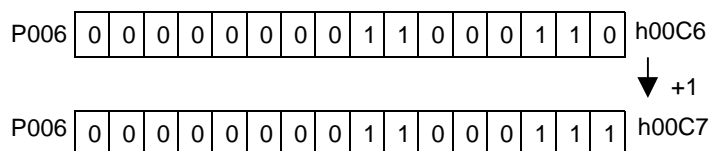
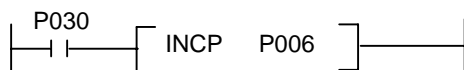


- Execution conditions



2) Program example

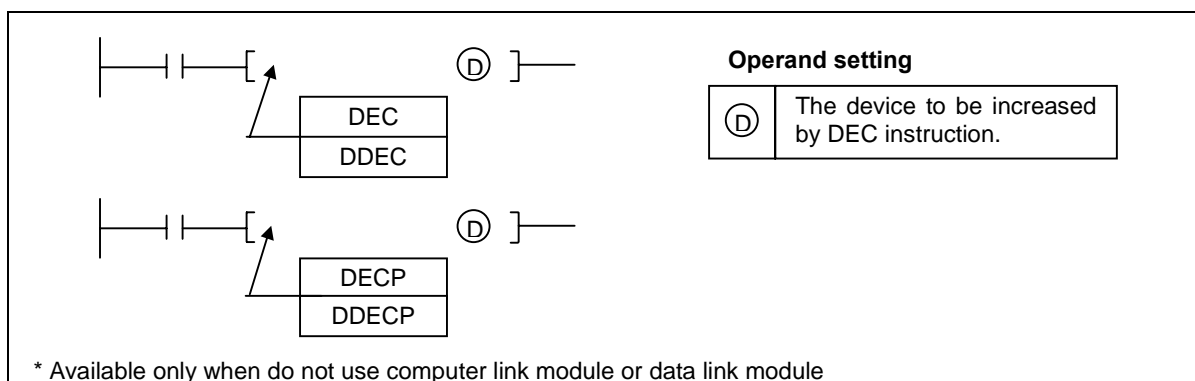
- Whenever a rising edge is detected at P030, the content of P06 word will be increased by 1.



5.4.2 DEC, DECP, DDEC, DDECP

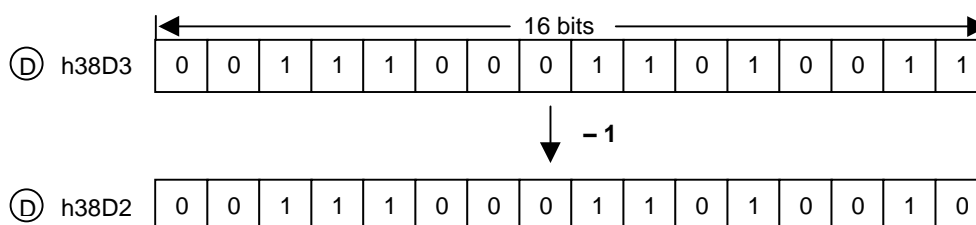
DEC	FUN(24)	DEC	FUN(26)	DDEC	Applicable CPU	All CPUs
(Decrement)	FUN(25)	DECP	FUN(27)	DDECP		

Instructions		Available Device										Steps	Flag			
		M	P	K	L	F	T	C	S	D	#D		Integer	Error (F110)	Zero (F111)	Carry (F112)
DEC(P) DDEC(P)	Ⓓ	○	○	○	○*		○	○		○	○		3	○	○	○

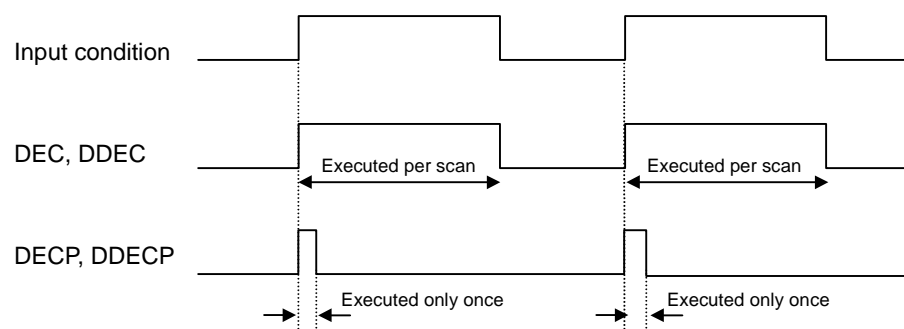


1) Functions

- DEC(P) : Performs the subtraction of 1 to the device (16-bits data) specified at [D].
- DDEC(P) : Performs the subtraction of 1 to the device (32-bits data) [D+1, D].
- If the DEC(P) or DDEC(P) is executed when the content of device is 0, the content of device will be hFFFF or hFFFFFFFF and the carry flag (F112) is set.
- The zero flag will be set when the content of device is 0.
- If the device specified by #D is out of the range, the operation error occurs and the error flag (F110) will be set.

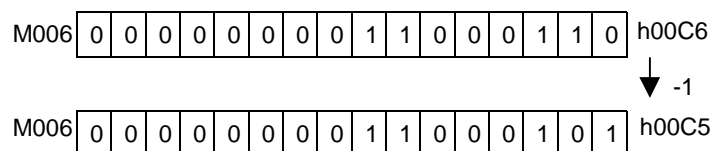


- Execution conditions



2) Program example

- Whenever a rising edge is detected at P020, the content of M006 word will be decreased by 1.

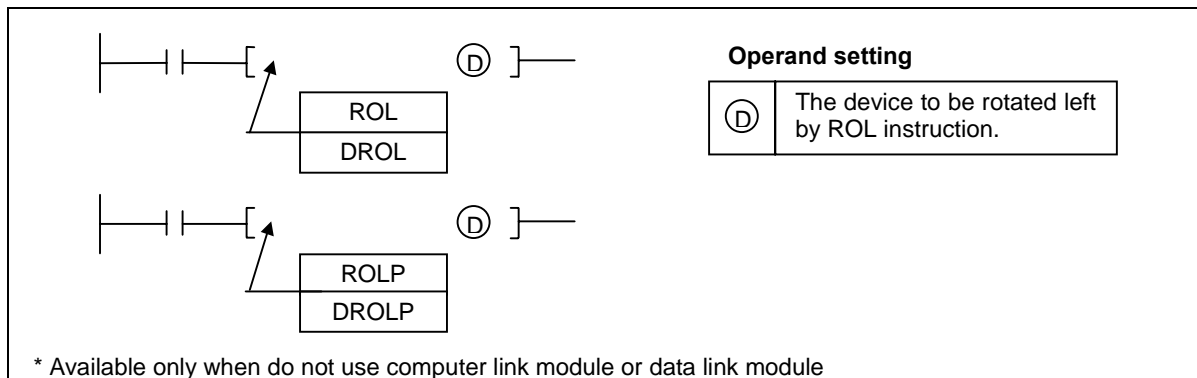


5.5 Rotation instructions

5.5.1 ROL, ROLP, DROL, DROL P

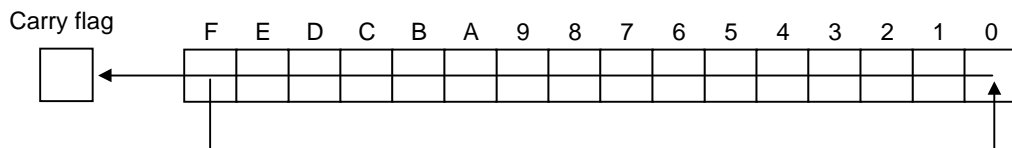
ROL (Rotate left)	FUN(30) ROL FUN(32) DROL FUN(31) ROLP FUN(33) DROL P	Applicable CPU	All CPUs
----------------------	---	-------------------	----------

Instructions		Available Device										Steps	Flag			
		M	P	K	L	F	T	C	S	D	#D		Integer	Error (F110)	Zero (F111)	Carry (F112)
ROL(P)	Ⓓ	O	O	O	O*		O	O		O	O		3	O		O
DROL(P)																

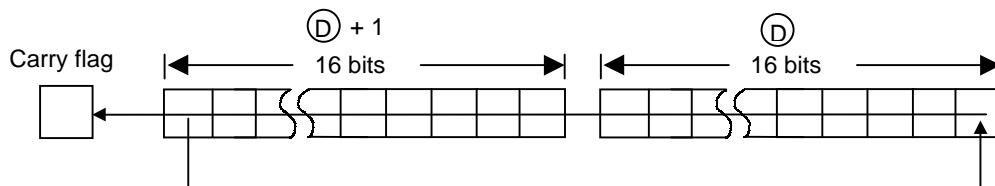


1) Functions

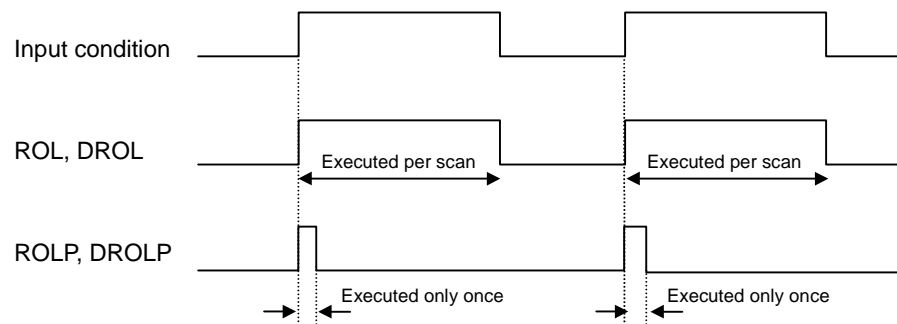
- ROL(P) : Rotates 16 bits of the device specified at [D] in left direction.
- The MSB will be transferred to the LSB and carry flag (F112)



- DROL(P) : Rotates 32-bits of the device specified as [D+1, D] in left direction.
- The MSB of [D+1] will be transferred to the LSB of [D] and carry flag.



- Execution conditions

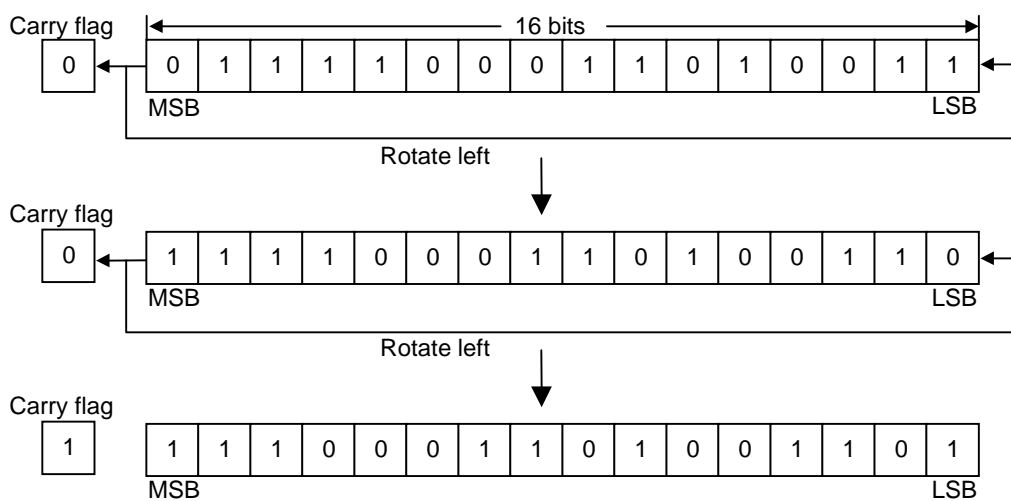


2) Program example

- Whenever a rising edge is detected at P030, 16-bits of D0000 word will be rotated with left direction.



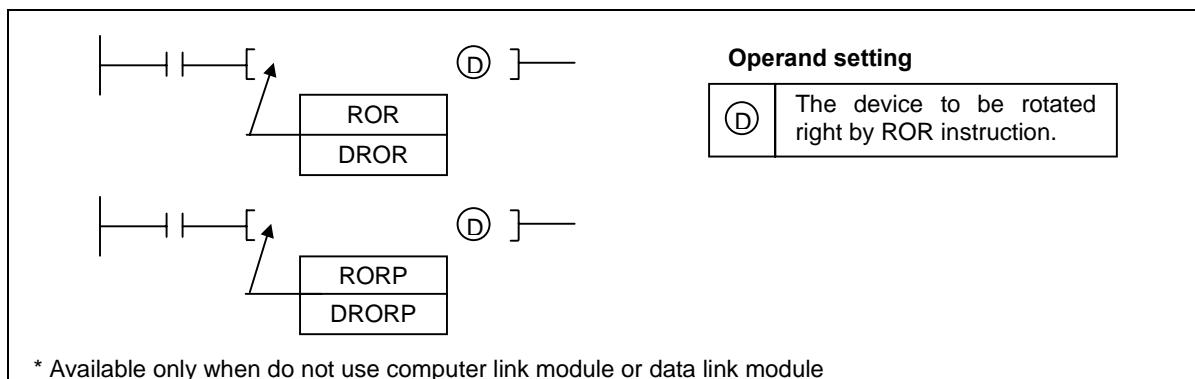
D0000 = h78D3



5.5.2 ROR, RORP, DROR, DRORP

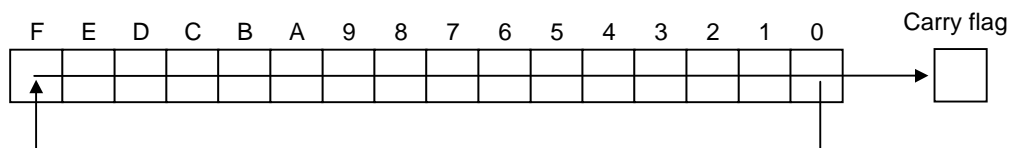
ROR (Rotate right)	FUN(34) ROR FUN(36) DROR FUN(35) RORP FUN(37) DRORP	Applicable CPU	All CPUs
-----------------------	--	-------------------	----------

Instructions		Available Device										Steps	Flag			
		M	P	K	L	F	T	C	S	D	#D		Integer	Error (F110)	Zero (F111)	Carry (F112)
ROR(P) DROR(P)	Ⓓ	○	○	○	○*		○	○		○	○		3	○		○

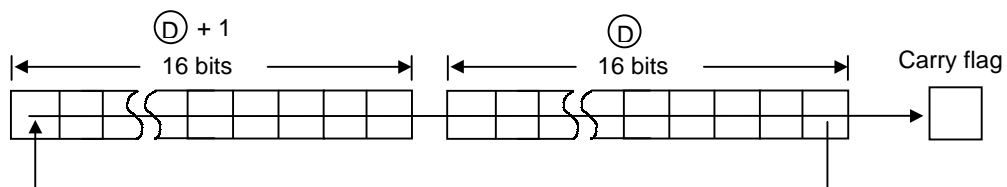


1) Functions

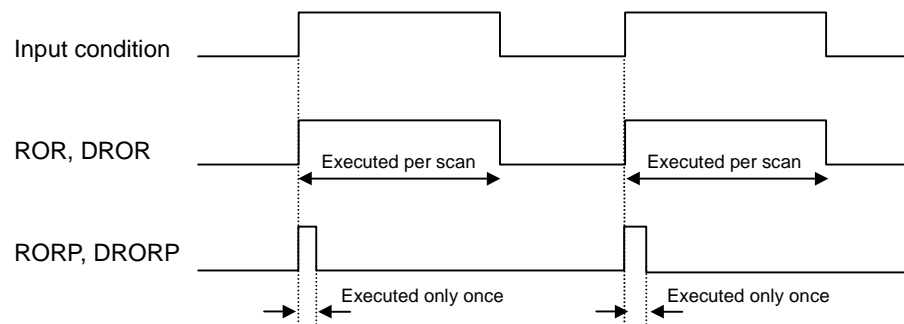
- ROR(P) : Rotates 16 bits of the device specified at [D] in right direction.
- The LSB will be transferred to the MSB and carry flag (F112)



- DROR(P) : Rotates 32-bits of the device specified as [D+1, D] in right direction.
- The LSB of [D] will be transferred to the MSB of [D+1] and carry flag.



- Execution conditions

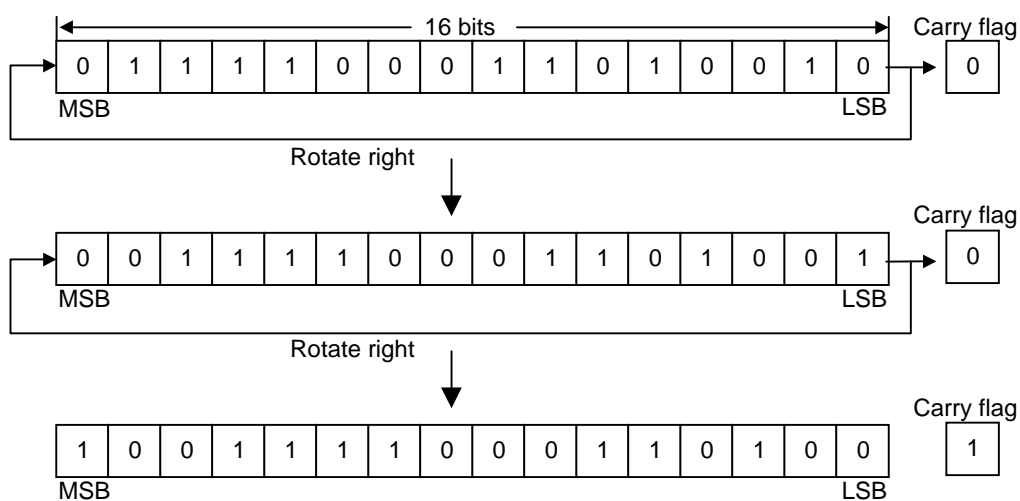


2) Program example

- Whenever a rising edge is detected at P030, 16-bits of D0000 word will be rotated with right direction.



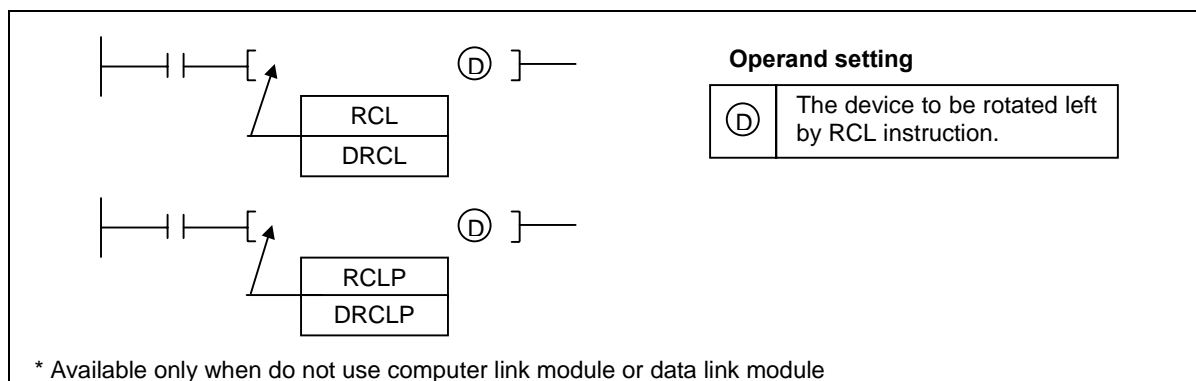
D0000 = h78D2



5.5.3 RCL, RCLP, DRCL, DRCLP

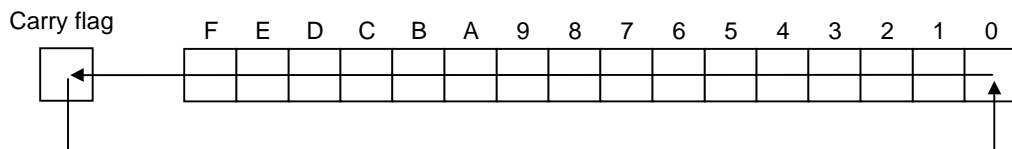
RCL (Rotate left include carry flag)	FUN(40) RCL	FUN(42) DRCL	Applicable CPU	All CPUs
	FUN(41) RCLP	FUN(43) DRCLP		

Instructions		Available Device											Steps	Flag		
		M	P	K	L	F	T	C	S	D	#D	Integer		Error (F110)	Zero (F111)	Carry (F112)
RCL(P)	Ⓓ	O	O	O	O*		O	O		O	O		3	O		O
DRCL(P)																

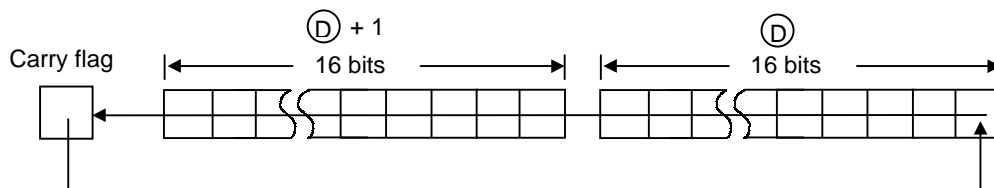


1) Functions

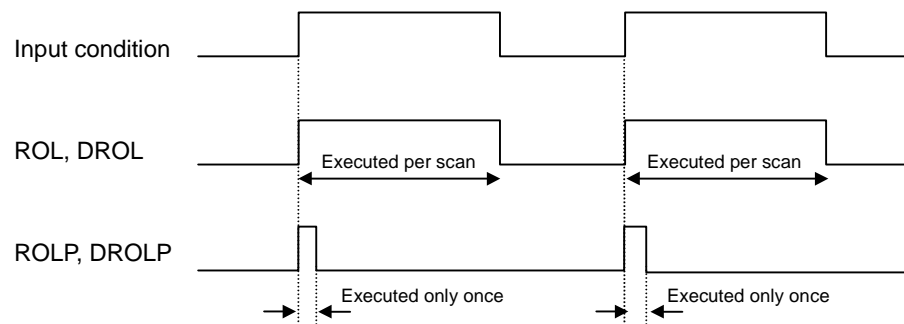
- RCL(P) : Rotates 16 bits of the device specified at [D] and carry flag (F112) in left direction.
- The MSB will be transferred to the carry flag (F112) and the carry flag will be transferred to the LSB.



- DRCL(P) : Rotates 32-bits of the device specified as [D+1, D] and carry flag in left direction.
- The MSB of [D+1] will be transferred to the carry flag (F112) and the carry flag will be transferred to the LSB of [D].

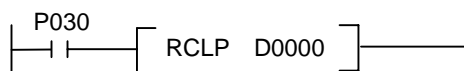


- Execution conditions

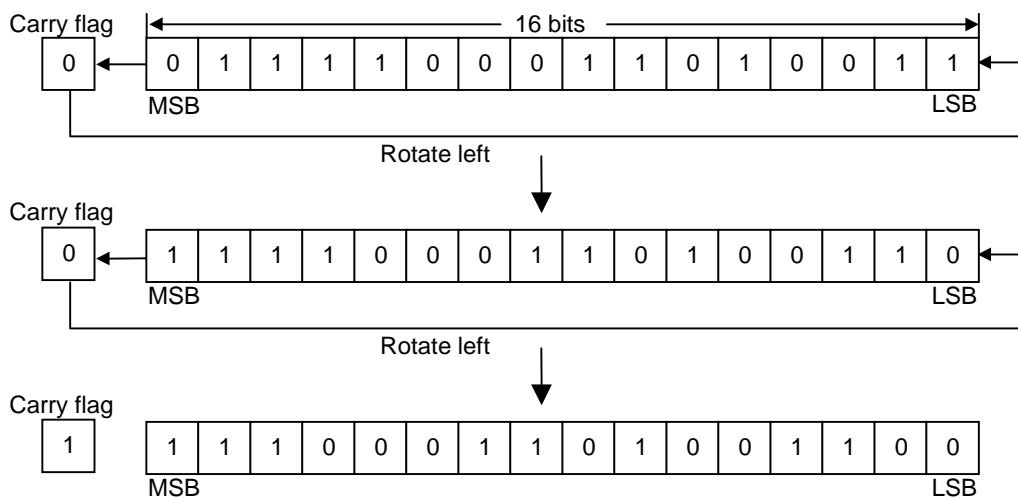


2) Program example

- Whenever a rising edge is detected at P030, 16-bits of D0000 word and carry flag will be rotated with left direction.



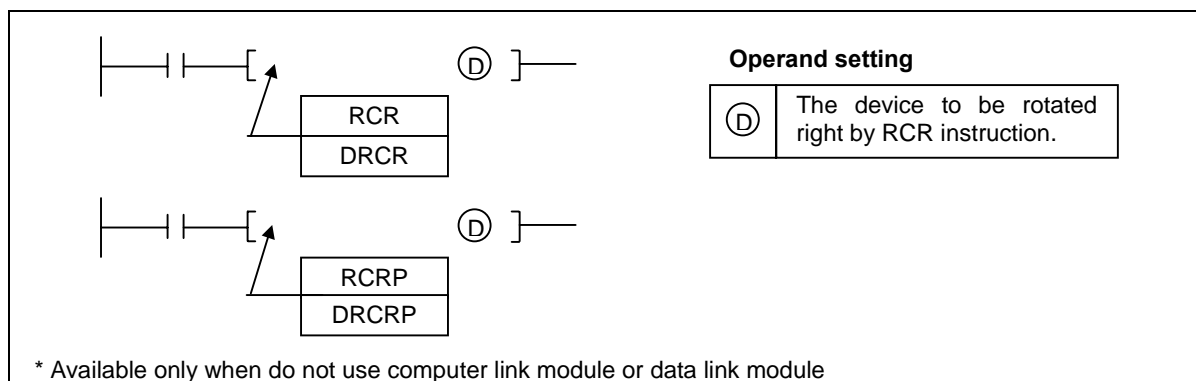
D0000 = h78D3



5.5.4 RCR, RCRP, DRCR, DRCRP

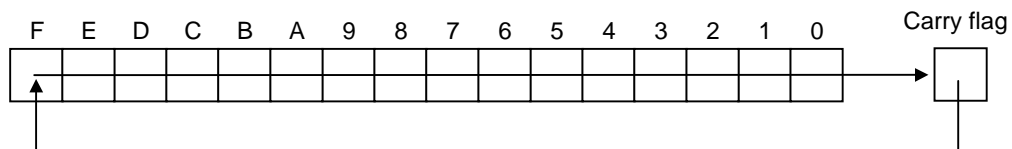
RCR (Rotate right include carry flag)	FUN(44) RCR FUN(46) DRCR FUN(45) RCRP FUN(47) DRCRP	Applicable CPU	All CPUs
---	--	-------------------	----------

Instructions		Available Device											Steps	Flag		
		M	P	K	L	F	T	C	S	D	#D	Integer		Error (F110)	Zero (F111)	Carry (F112)
RCR(P) DRCR(P)	Ⓓ	O	O	O	O*		O	O		O	O		3	O		O

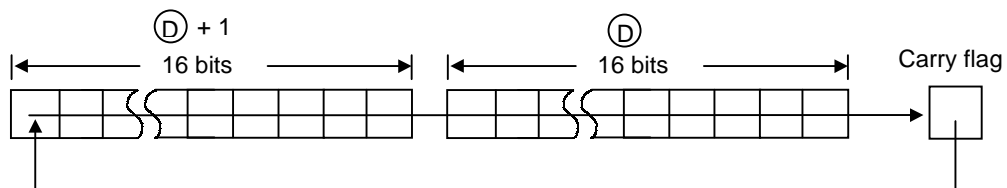


1) Functions

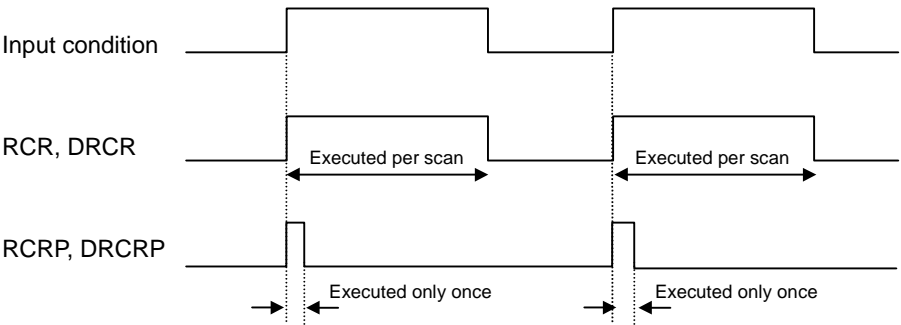
- RCR(P) : Rotates 16 bits of the device specified at [D] and the carry flag in right direction.
- The LSB will be transferred to the carry flag (F112) and the carry flag will be transferred to the MSB.



- DRCR(P) : Rotates 32-bits of the device specified as [D+1, D] and the carry flag in right direction.
- The LSB of [D] will be transferred to the carry flag and the carry flag will be transferred to the MSB of [D+1].



- Execution conditions

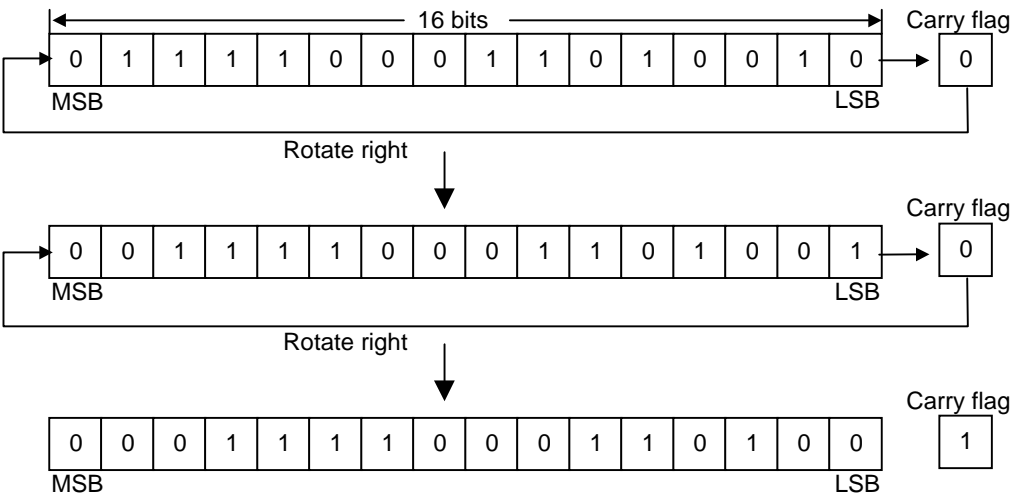


2) Program example

- Whenever a rising edge is detected at P030, 16-bits of D0000 word and carry flag will be rotated with right direction.



D0000 = h78D2

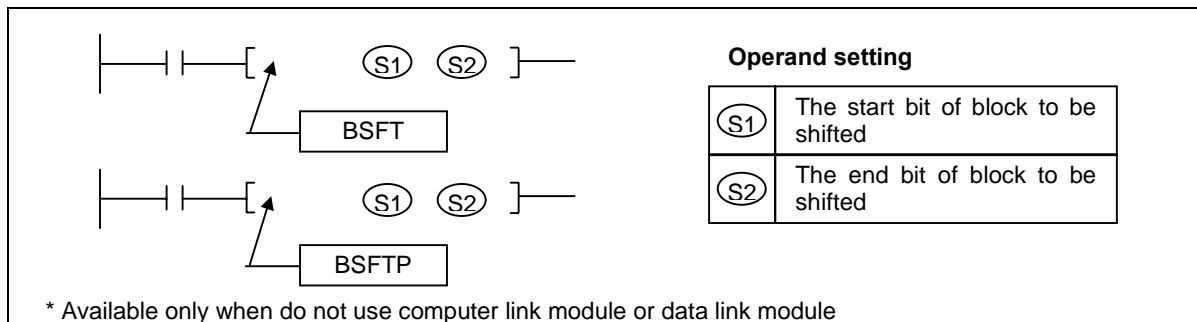


5.6 Shift instructions

5.6.1 BSFT, BSFTP

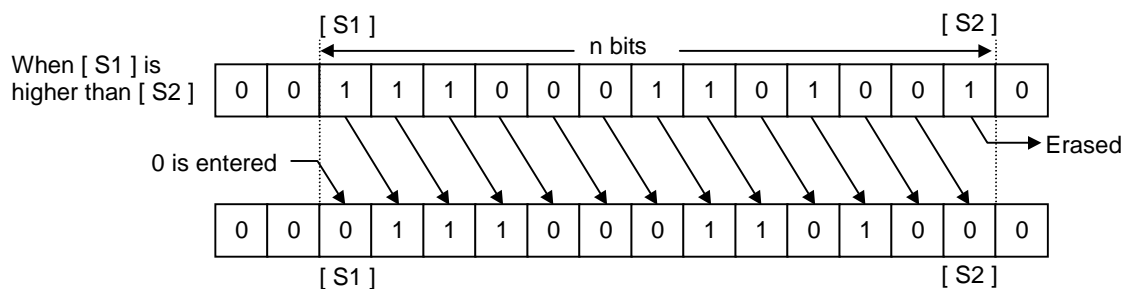
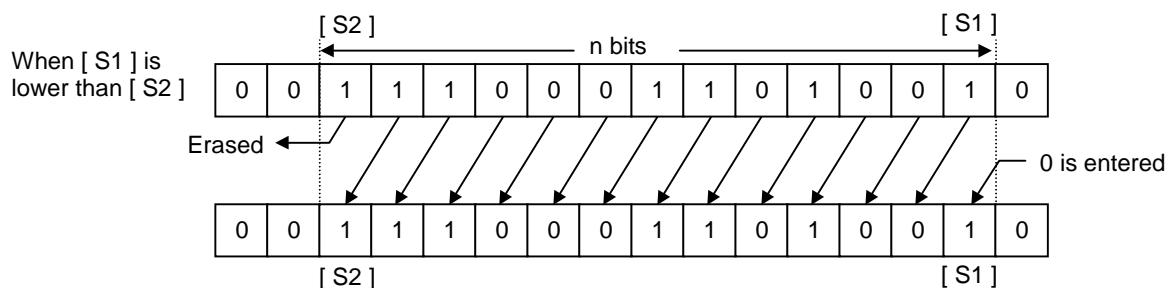
BSFT (Bit shift)	FUN(74) BSFT FUN(75) BSFTP	Applicable CPU	All CPUs
---------------------	-------------------------------	-------------------	----------

Instructions		Available Device										Steps	Flag		
		M	P	K	L	F	T	C	S	D	#D	Integer	Error (F110)	Zero (F111)	Carry (F112)
BSFT(P)	(S1)	O	O	O	O*								5	O	
	(S2)	O	O	O	O*										

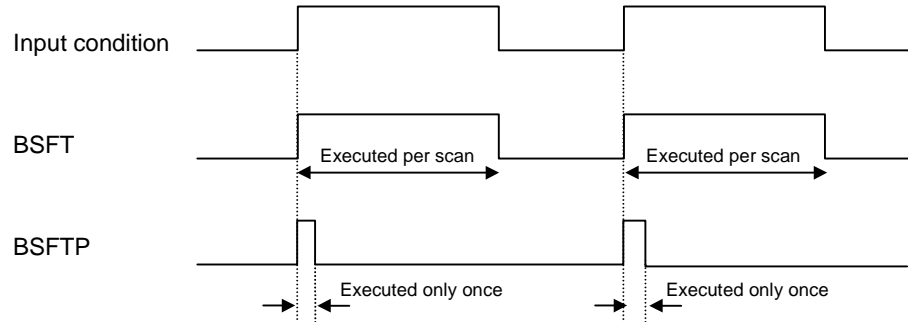


1) Functions

- Shifts the block specified as [S1] ~ [S2] by 1 bit.
- The direction of shift is from [S1] to [S2]. Therefore, if [S1] is lower than [S2], the block is shifted in left direction. Otherwise, the block is shifted in right direction.

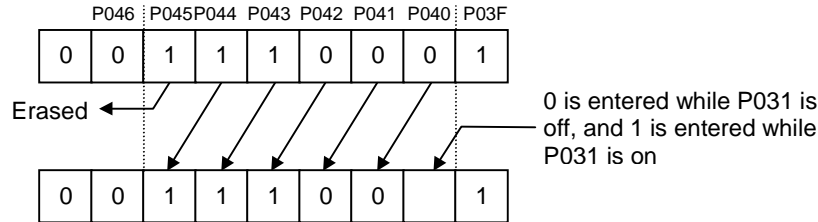
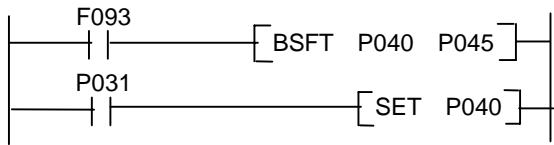


- Execution conditions



2) Program example

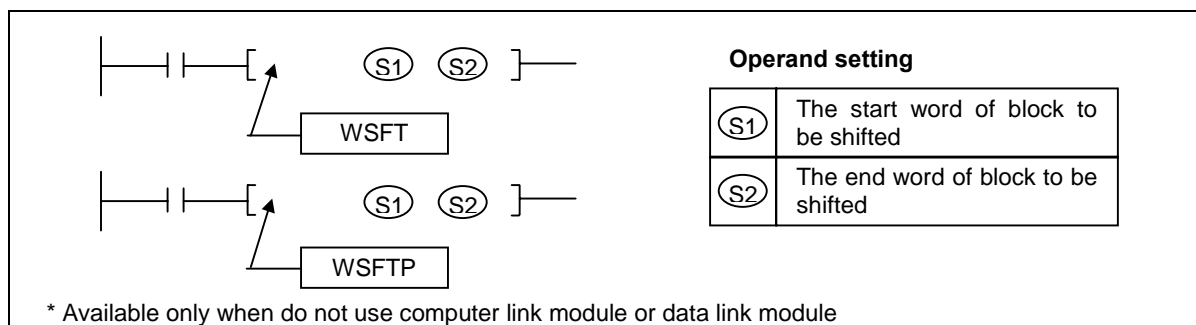
- At every 1 second, the block from P040 to P045 is shifted in left direction by 1 bit. The 1 second clock flag (F093) is used for input condition. P040 is set as 1 when the P031 is on.



5.6.2 WSFT, WSFTP

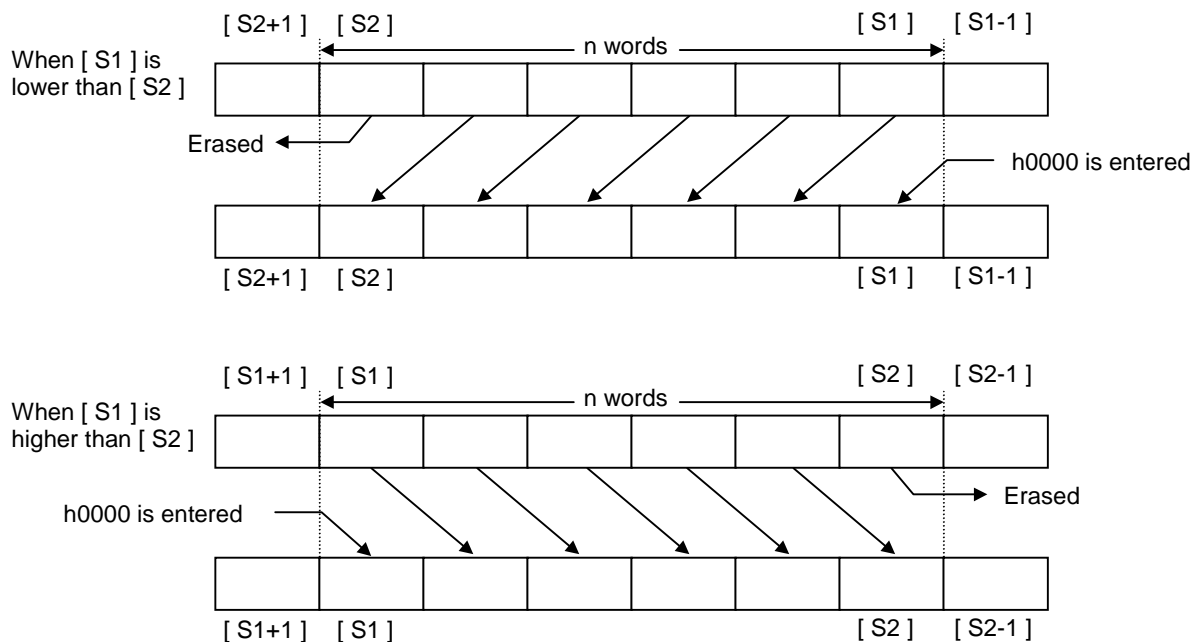
WSFT (Word shift)	FUN(70) WSFT FUN(71) WSFTP	Applicable CPU	All CPUs
----------------------	-------------------------------	-------------------	----------

Instructions		Available Device											Steps	Flag		
		M	P	K	L	F	T	C	S	D	#D	Integer		Error (F110)	Zero (F111)	Carry (F112)
WSFT(P)	(S1)	O	O	O	O*		O	O		O	O		5	O		
	(S2)	O	O	O	O*		O	O		O	O					

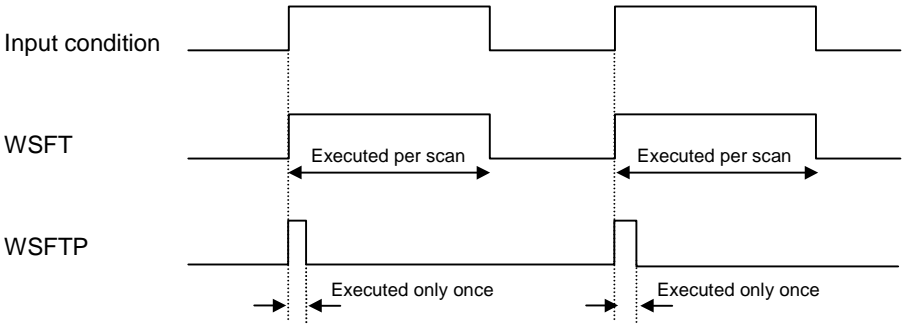


1) Functions

- Shifts the block specified as [S1] ~ [S2] by 1 word.
- The direction of shift is from [S1] to [S2]. Therefore, if [S1] is lower than [S2], the block is shifted in left direction. Otherwise, the block is shifted in right direction.

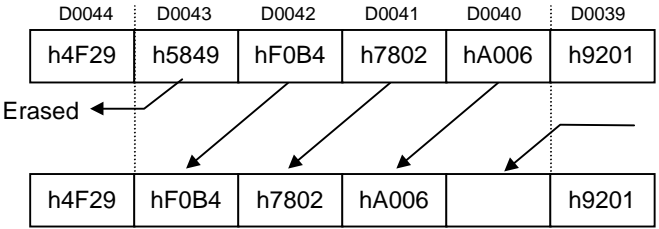
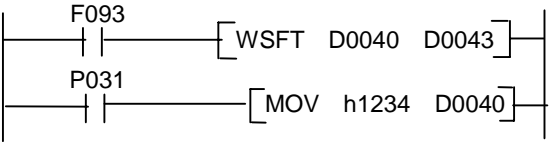


- Execution conditions



2) Program example

- At every 1 second, the block from D0040 to D0043 is shifted in left direction by 1 word. The 1 second clock flag (F093) is used for input condition. D0040 is set as h1234 when the P031 is on.

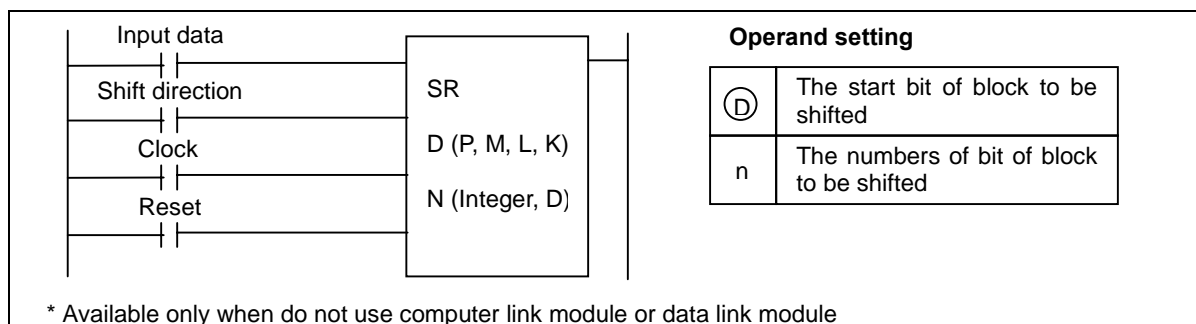


h0000 is entered while P031 is off, and h1234 is entered while P031 is on

5.6.3 SR

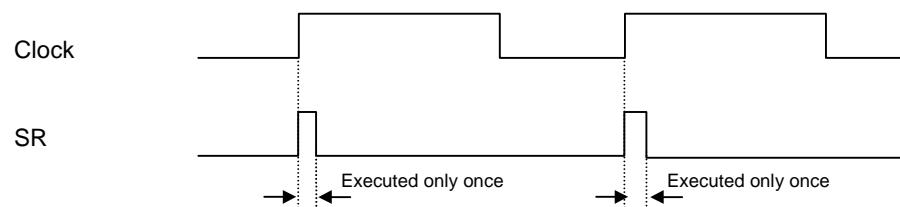
SR (Shift register)	FUN(237) SR	Applicable CPU	K200S K300S K1000S
------------------------	-------------	-------------------	--------------------------

Instructions		Available Device										Steps	Flag		
		M	P	K	L	F	T	C	S	D	#D	Integer	Error (F110)	Zero (F111)	Carry (F112)
BSFT(P)	Ⓓ	O	O	O	O*								5	O	
	n									O		O			



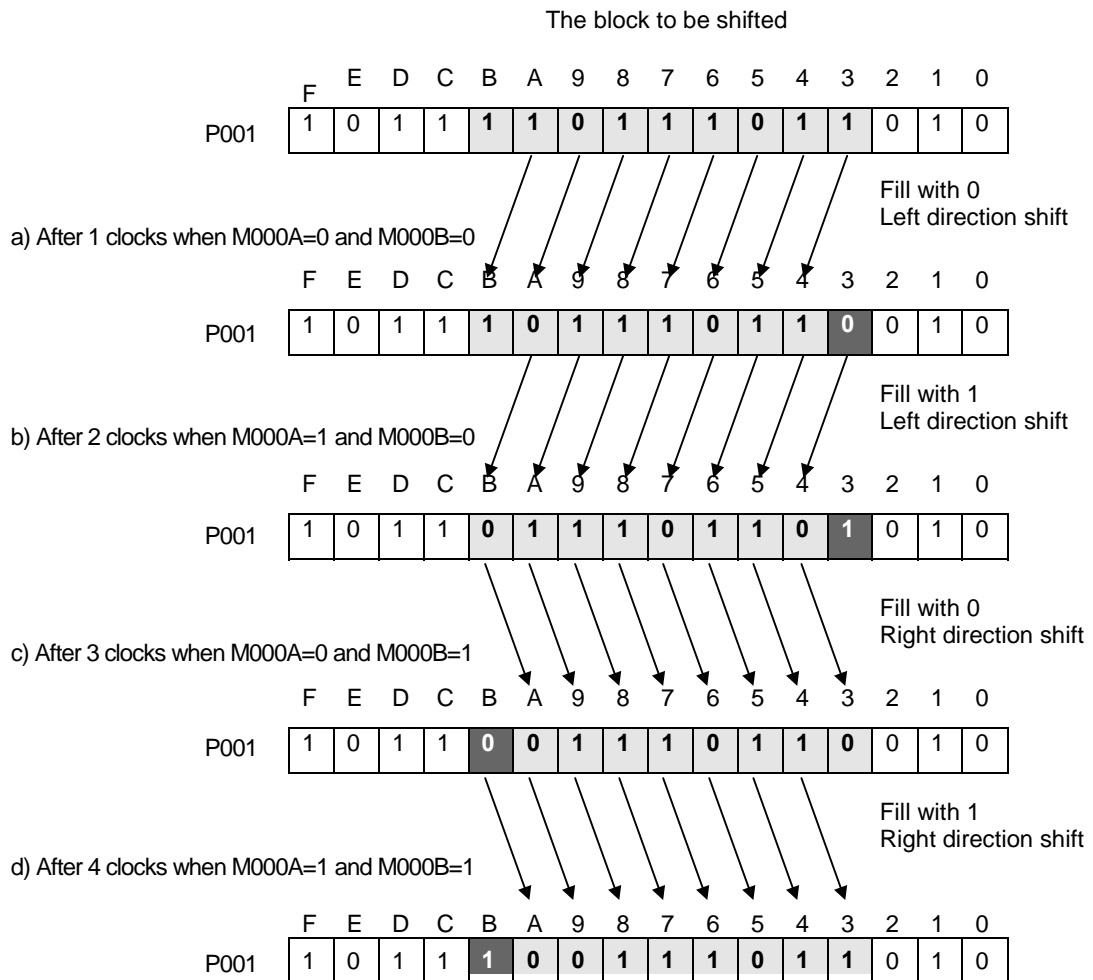
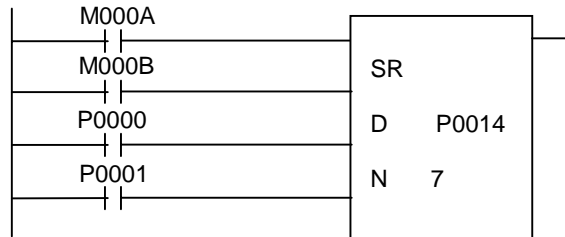
1) Functions

- Whenever a rising edge is detected at the clock input, shifts the block from the bit specified at [D] to the bit [D+n] by 1 bit.
- At the start bit of shift operation, 0 is entered when the input data is off, and 1 is entered when the input data is on. The start bit of shift operation is various according to the direction of shift operation (left direction shift : LSB, right direction shift : MSB).
- The shift direction indicates the direction of shift operation. If the shift direction is off, it means a left direction shift. Otherwise, it means a right direction shift.
- When the reset signal is switched on, all bit from [D] to [D+n] is cleared as 0.
- Execution condition



2) Program example

- Shifts the block from P0013 ~ P001B (9 bits) with the configuration as following :
- Input data bit : M000A
- Shift direction but : M000B
- Clock : P0000
- Reset signal : P0001

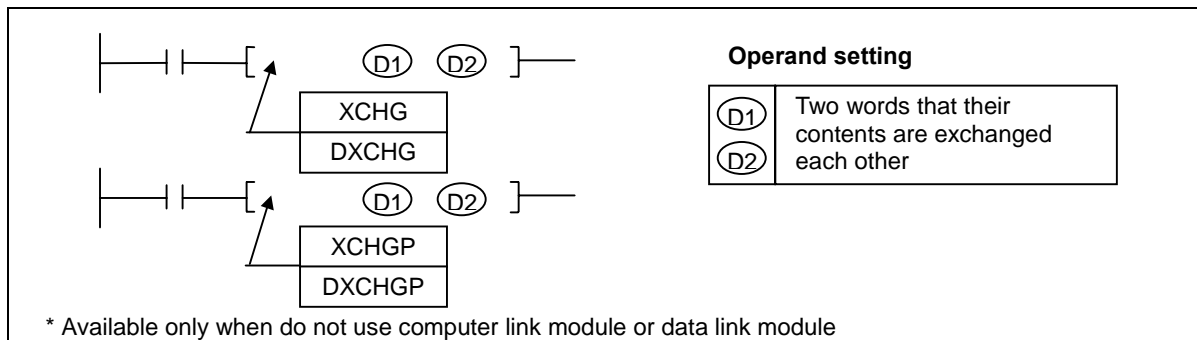


5.7 Exchange instructions

5.7.1 XCHG, XCHGP, DXCHG, DXCHGP

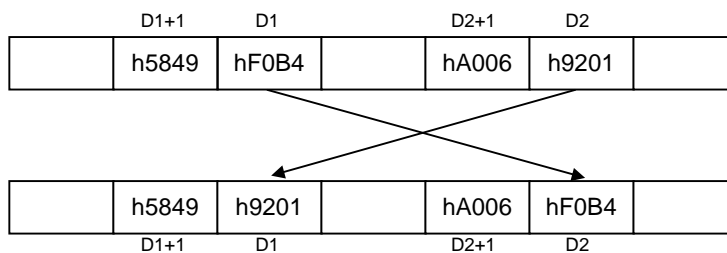
XCHG (Word exchange)	FUN(102) XCHG FUN(104) DXCHG FUN(103) XCHGP FUN(105) DXCHGP	Applicable CPU	All CPUs
-------------------------	--	-------------------	----------

Instructions		Available Device											Steps	Flag		
		M	P	K	L	F	T	C	S	D	#D	Integer		Error (F110)	Zero (F111)	Carry (F112)
XCHG(P)	(D1)	O	O	O	O*		O	O		O	O		5	O		
DXCHG(P)	(D2)	O	O	O	O*		O	O		O	O					

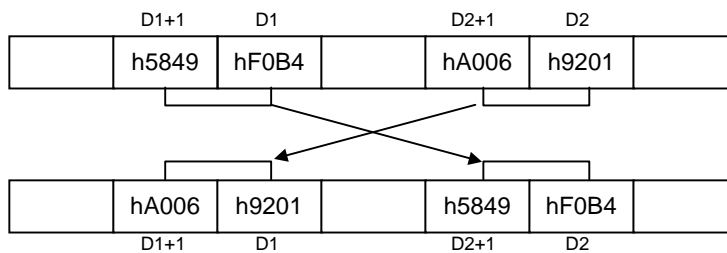


1) Functions

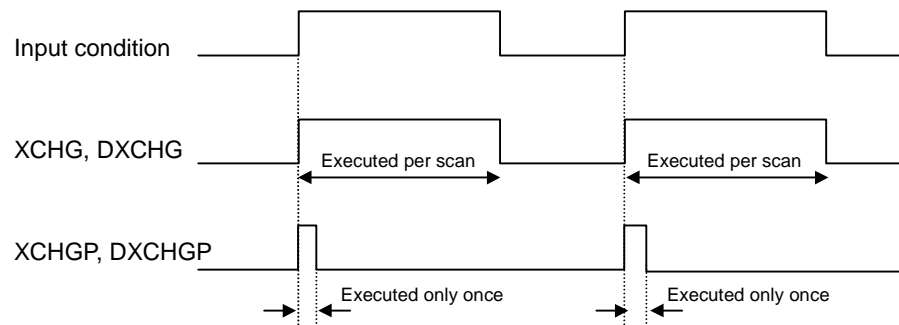
- XCHG(P) : Exchanges 16-bits contents of two devices specified at [D1] and [D2].



- DXCHG(P) : Exchanges 32-bits contents of two devices specified as [D1+1, D1] and [D2+1, D2].

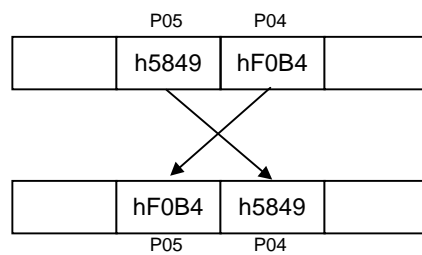
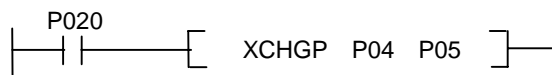


- Execution condition



2) Program example

- While P020 is on, exchange contents of P04 and P05 words each other.



5.8 BIN arithmetic instructions

5.8.1 ADD, ADDP, DADD, DADDP

ADD (Binary addition)	FUN(110) ADD FUN(112) DADD FUN(111) ADDP FUN(113) DADDP	Applicable CPU	All CPUs
--------------------------	--	-------------------	----------

Instructions		Available Device											Steps	Flag		
		M	P	K	L	F	T	C	S	D	#D	Integer		Error (F110)	Zero (F111)	Carry (F112)
ADD(P) DADD(P)	(S1)	O	O	O	O	O	O	O		O	O	O	7/9/11	O	O	O
	(S2)	O	O	O	O	O	O	O		O	O	O				
	(D)	O	O	O	O*		O	O		O	O					

Operand setting

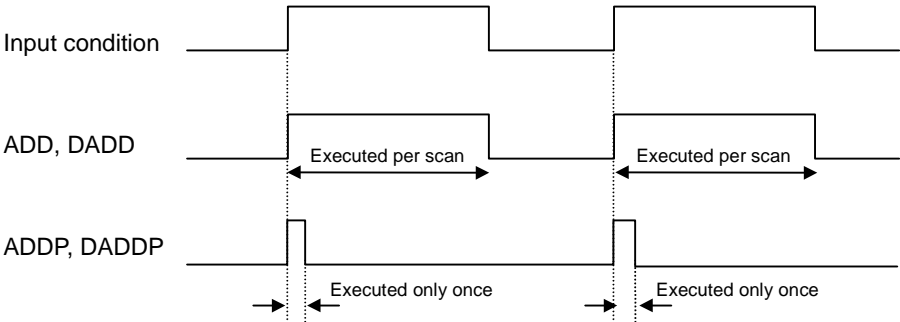
(S1)	The device storing augend
(S2)	The device storing addend
(D)	The device at which the addition result is stored

* Available only when do not use computer link module or data link module

1) Functions

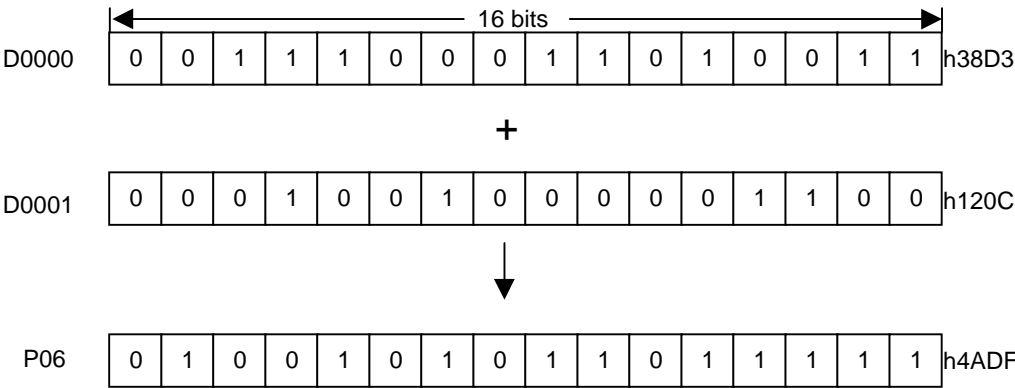
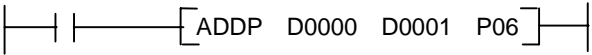
- ADD(P) : Performs the addition of 16-bits BIN data specified at [S1] and [S2]. The addition result is stored at the device specified at [D].
- DADD(P) : Performs the addition of 32-bits BIN data specified at [S1+1, S1] and [S2+1, S2]. The addition result is stored at the device specified at [D1+1, D1].
- When the addition result is over hFFFF(ADD / ADDP) or hFFFFFFFF(DADD / DADDP), the carry flag (F112) is set.
- When the addition result is 0, the zero flag is set.
- If indirect address specified by #D format is out of device range, the operation error occur and the error flag (F110) is set.

- Execution condition



2) Program example

- When a rising edge is detected at P020, add contents of D0000 and D0001 and store the addition result to P06 word.



5.8.2 SUB, SUBP, DSUB, DSUBP

SUB (Binary subtraction)	FUN(114) SUB FUN(116) DSUB FUN(115) SUBP FUN(117) DSUBP	Applicable CPU	All CPUs
-----------------------------	--	-------------------	----------

Instructions		Available Device											Steps	Flag		
		M	P	K	L	F	T	C	S	D	#D	Integer		Error (F110)	Zero (F111)	Carry (F112)
SUB(P) DSUB(P)	(S1)	O	O	O	O	O	O	O		O	O	O	7/9/11	O	O	O
	(S2)	O	O	O	O	O	O	O		O	O	O				
	(D)	O	O	O	O*		O	O		O	O					

Operand setting

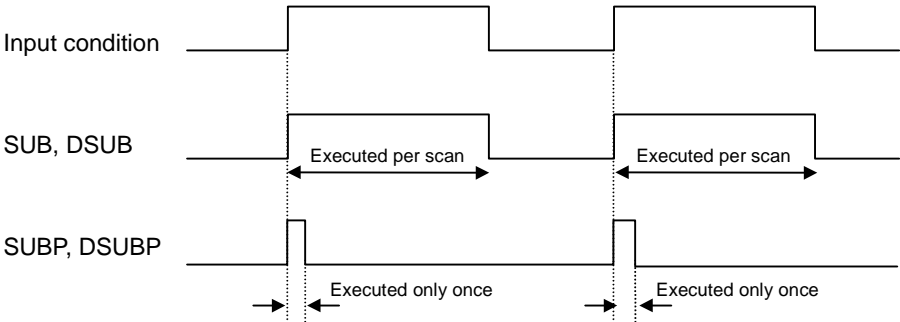
(S1)	The device storing minuend
(S2)	The device storing subtrahend
(D)	The device at which the subtraction result is stored

* Available only when do not use computer link module or data link module

1) Functions

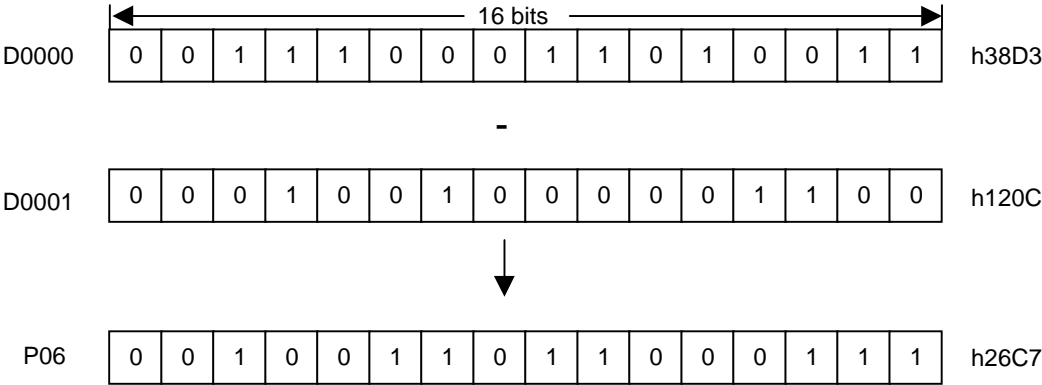
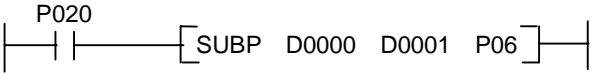
- SUB(P) : Performs the subtraction of 16-bits BIN data specified at [S1] and [S2]. The subtraction result is stored at the device specified at [D].
- DSUB(P) : Performs the subtraction of 32-bits BIN data specified at [S1+1, S1] and [S2+1, S2]. The subtraction result is stored at the device specified at [D1+1, D1].
- When the minuend is less than subtrahend, the LSB will underflow, and the carry flag (F112) will be set.
- When the subtraction result is 0, the zero flag is set.
- If indirect address specified by #D format is out of device range, the operation error occurs and the error flag (F110) is set.

- Execution condition



2) Program example

- When a rising edge is detected at P020, subtract contents of D0000 and D0001 and store the addition result to P06 word.



5.8.3 MUL, MULP, DMUL, DMULP

MUL (Binary multiply)	FUN(120) MUL FUN(122) DMUL FUN(121) MULP FUN(123) DMULP	Applicable CPU	All CPUs
--------------------------	--	-------------------	----------

Instructions	Available Device											Steps	Flag		
		M	P	K	L	F	T	C	S	D	#D	Integer	Error (F110)	Zero (F111)	Carry (F112)
MUL(P)	(S1)	O	O	O	O	O	O	O		O	O	O	7/9/11	O	O
	(S2)	O	O	O	O	O	O	O		O	O	O			
DMUL(P)	(D)	O	O	O	O*		O	O		O	O				

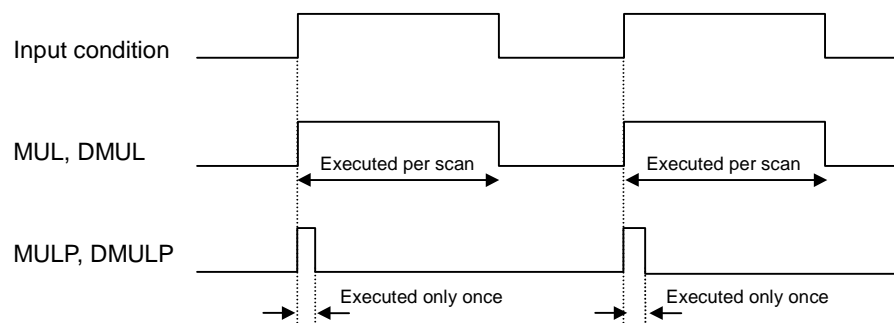
Operand setting

(S1)	The device storing multiplicand
(S2)	The device storing multiplier
(D)	The device at which the multiplication result is stored

* Available only when do not use computer link module or data link module

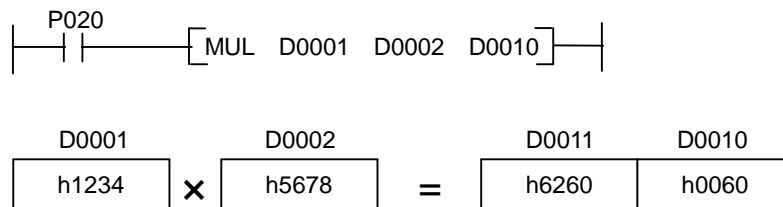
1) Functions

- MUL(P) : Performs the multiplication of BIN data specified as [S1] and the BIN data specified as [S2], and stores the multiplication result into the device specified as [D+1, D].
- DMUL(P) : Performs the multiplication of BIN data specified as [S1+1, S] and the BIN data specified as [S2+1, S2], and stores the multiplication result into the device specified as [D+3, D+2, D+1, D].
- If the multiplication result is zero, the zero flag will be set.
- If indirect address specified by #D format is out of device range, the operation error occurs and the error flag (F110) is set.
- Execution conditions

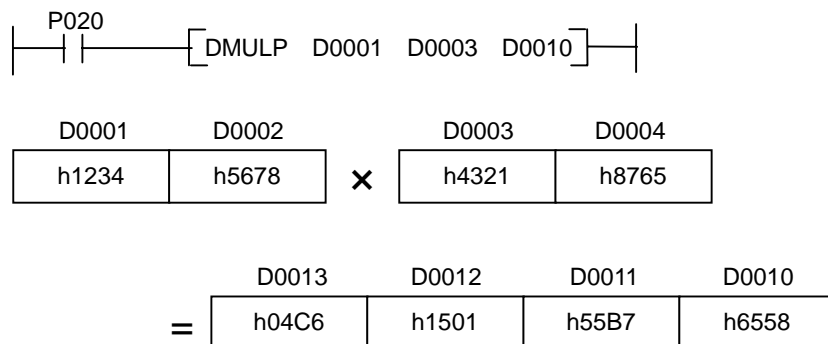


2) Program example

- Program which stores the multiplication result of D0001 and D0002 at D0010, D0011 while P020 is on.



- Program which stores the multiplication result of D0001, D0002 and D0003,D0004 at D0010 ~ D0013 when P020 is switched on.



5.8.4 MULS, MULSP, DMULS, DMULSP

MULS (Signed binary multiply)	FUN(072) MULS FUN(076) DMULS	Applicable CPU	K200S
	FUN(073) MULSP FUN(077) DMULSP		K300S K1000S

Instructions		Available Device											Steps	Flag		
		M	P	K	L	F	T	C	S	D	#D	Integer		Error (F110)	Zero (F111)	Carry (F112)
MULS(P) DMULS(P)	(S1)	O	O	O	O	O	O	O		O	O	O	7/9/11	O	O	
	(S2)	O	O	O	O	O	O	O		O	O	O				
	(D)	O	O	O	O*		O	O		O	O					

Operand setting

(S1)	The device storing multiplicand
(S2)	The device storing multiplier
(D)	The device at which the multiplication result is stored

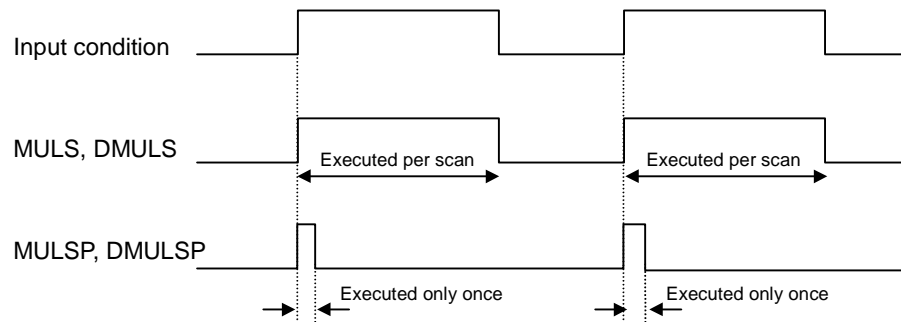
* Available only when do not use computer link module or data link module

1) Functions

- MULS(P) : Performs the multiplication of the signed BIN data specified as [S1] and the signed BIN data specified as [S2], and stores the multiplication result into the device specified as [D+1, D].
- DMULS(P) : Performs the multiplication of signed BIN data specified as [S1+1, S] and the signed BIN data specified as [S2+1, S2], and stores the multiplication result into the device specified as [D+3, D+2, D+1, D].
- If the multiplication result is zero, the zero flag will be set.
- If indirect address specified by #D format is out of device range, the operation error occurs and the error flag (F110) is set.
- The sign of multiplication result is as following table

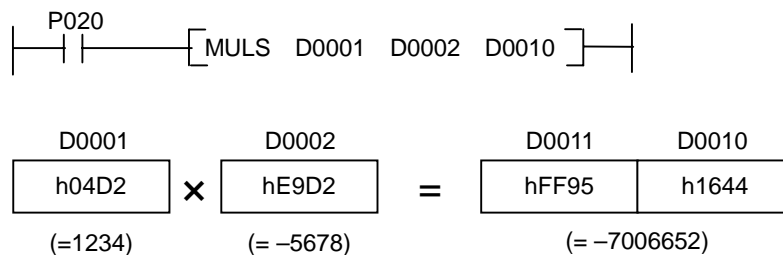
(S1)	(S2)	(D)
+ (positive)	+ (positive)	+ (positive)
+ (positive)	- (negative)	- (negative)
- (negative)	+ (positive)	- (negative)
- (negative)	- (negative)	+ (positive)

- Execution conditions

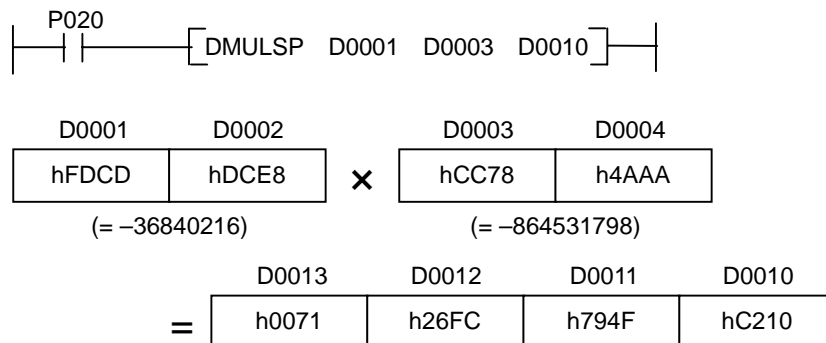


2) Program example

- Program which stores the multiplication result of D0001 and D0002 at D0010, D0011 while P020 is on.



- Program which stores the multiplication result of D0001, D0002 and D0003,D0004 at D0010 ~ D0013 when P020 is switched on.



5.8.5 DIV, DIVP, DDIV, DDIVP

DIV	FUN(124) DIV	FUN(126) DDIV	Applicable CPU	All CPUs
(Binary divide)	FUN(125) DIVP	FUN(127) DDIVP		

Instructions	Available Device											Steps	Flag		
		M	P	K	L	F	T	C	S	D	#D	Integer	Error (F110)	Zero (F111)	Carry (F112)
DIV(P)	(S1)	O	O	O	O	O	O	O		O	O	O	7/9/11	O	O
DDIV(P)	(S2)	O	O	O	O	O	O	O		O	O	O			
	(D)	O	O	O	O*		O	O		O	O				

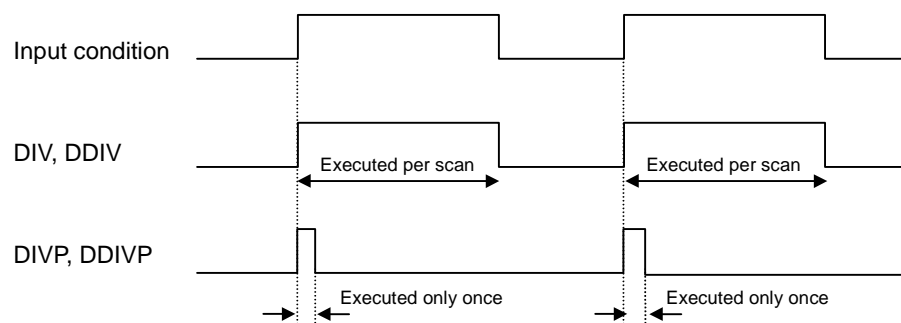
Operand setting

(S1)	The device storing dividend
(S2)	The device storing divider
(D)	The device at which the division result is stored

* Available only when do not use computer link module or data link module

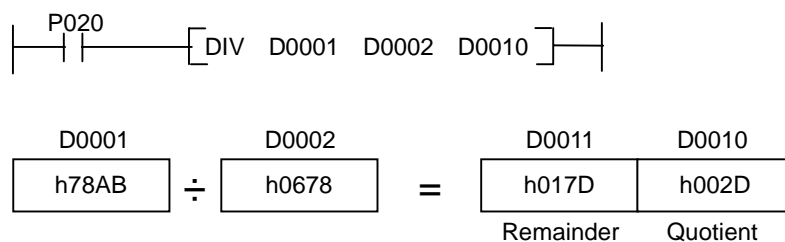
1) Functions

- DIV(P) : Performs the division of BIN data specified as [S1] and the BIN data specified as [S2], and stores the division result into the device specified as [D+1, D]. The quotient is stored at [D], and the remainder is stored at [D+1].
- DDIV(P) : Performs the division of BIN data specified as [S1+1, S] and the BIN data specified as [S2+1, S2], and stores the division result into the device specified as [D+3, D+2, D+1, D]. The quotient is stored at [D+1, D], and the remainder is stored at [D+3, D+2].
- If the quotient is zero, the zero flag will be set.
- If indirect address specified by #D format is out of device range or the content of divider is 0, the operation error occurs and the error flag (F110) is set.
- Execution conditions

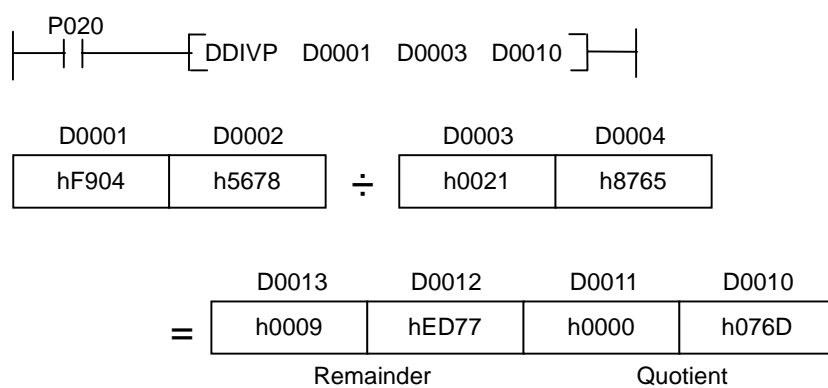


2) Program example

- Program which stores the division result of D0001 and D0002 at D0010, D0011 while P020 is on.



- Program which stores the division result of D0001, D0002 and D0003,D0004 at D0010 ~ D0013 when P020 is switched on.



5.8.6 DIVS, DIVSP, DDIVS, DDIVSP

DIVS (Signed binary divide)	FUN(124) DIV	FUN(126) DDIV	Applicable CPU	K200S
	FUN(125) DIVP	FUN(127) DDIVP		K300S K1000S

Instructions		Available Device											Steps	Flag		
		M	P	K	L	F	T	C	S	D	#D	Integer		Error (F110)	Zero (F111)	Carry (F112)
DIVS(P) DDIVS(P)	(S1)	O	O	O	O	O	O	O		O	O	O	7/9/11	O	O	
	(S2)	O	O	O	O	O	O	O		O	O	O				
	(D)	O	O	O	O*		O	O		O	O					

Operand setting

(S1)	The device storing dividend
(S2)	The device storing divider
(D)	The device at which the division result is stored

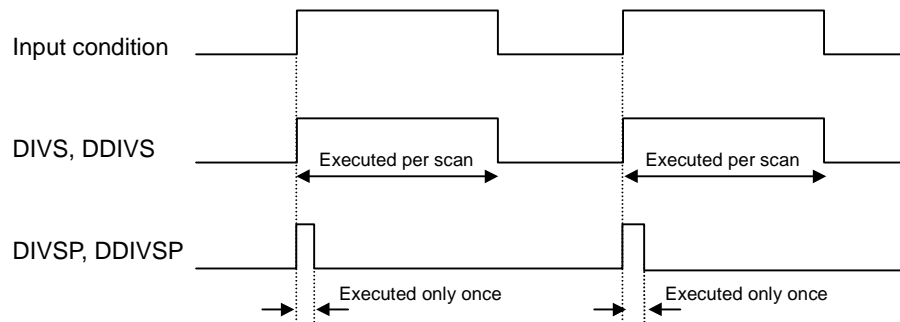
* Available only when do not use computer link module or data link module

1) Functions

- DIVS(P) : Performs the division of the signed BIN data specified as [S1] and the signed BIN data specified as [S2], and stores the division result into the device specified as [D+1, D]. The quotient is stored at [D], and the remainder is stored at [D+1].
- DDIVS(P) : Performs the division of the signed BIN data specified as [S1+1, S] and the signed BIN data specified as [S2+1, S2], and stores the division result into the device specified as [D+3, D+2, D+1, D]. The quotient is stored at [D+1, D], and the remainder is stored at [D+3, D+2].
- If the quotient is zero, the zero flag will be set.
- If indirect address specified by #D format is out of device range or the content of divider is 0, the operation error occurs and the error flag (F110) is set.
- The sign of quotient and remainder is as following table;

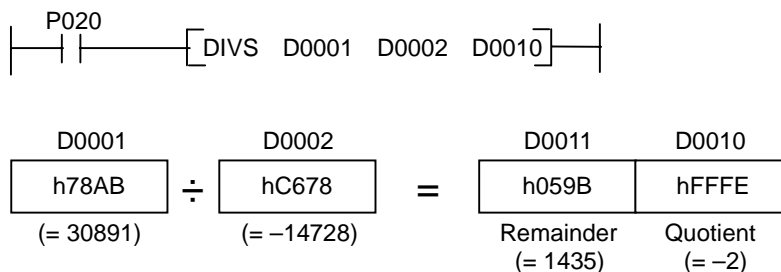
(S1)	(S2)	Quotient	Remainder
+ (positive)	+ (positive)	+ (positive)	+ (positive)
+ (positive)	- (negative)	- (negative)	+ (positive)
- (negative)	+ (positive)	- (negative)	- (negative)
- (negative)	- (negative)	+ (positive)	- (negative)

- Execution conditions

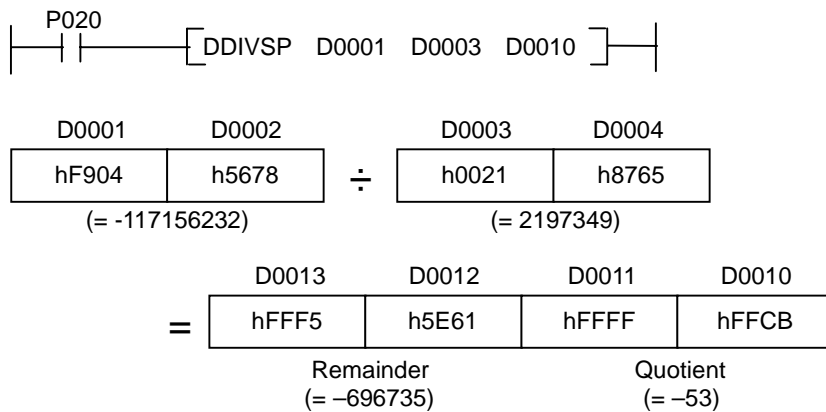


2) Program example

- Program which stores the division result of D0001 and D0002 at D0010, D0011 while P020 is on.



- Program which stores the division result of D0001, D0002 and D0003, D0004 at D0010 ~ D0013 when P020 is switched on.



5.9 BCD arithmetic instructions

5.9.1 ADDB, ADDBP, DADDB, DADDBP

ADDB (BCD addition)	FUN(130) ADDB FUN(132) DADDB FUN(131) ADDBP FUN(133) DADDBP	Applicable CPU	All CPUs
------------------------	--	-------------------	----------

Instructions		Available Device											Steps	Flag		
		M	P	K	L	F	T	C	S	D	#D	Integer		Error (F110)	Zero (F111)	Carry (F112)
ADDB(P) DADDB(P)	(S1)	O	O	O	O	O	O	O		O	O	O	7/9/11	O	O	O
	(S2)	O	O	O	O	O	O	O		O	O	O				
	(D)	O	O	O	O*		O	O		O	O					

The diagram illustrates the operand settings for ADDB and DADDB instructions. For ADDB, the operands are (S1), (S2), and (D). For DADDB, the operands are (S1), (S2), and (D). The diagram shows the instruction format with the operands in parentheses and the instruction name in a box.

Operand setting

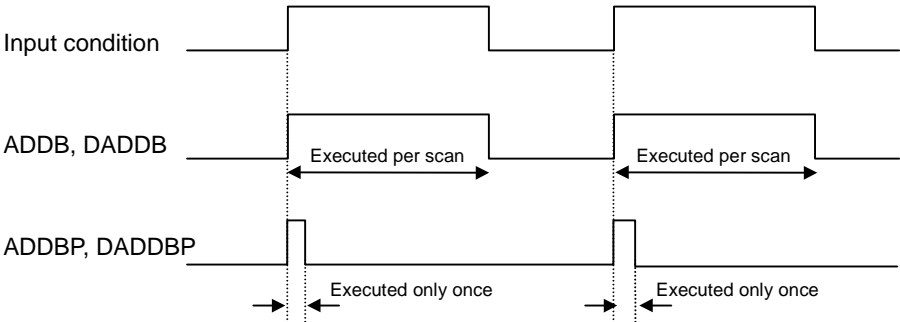
(S1)	The device storing augend
(S2)	The device storing addend
(D)	The device at which the addition result is stored

* Available only when do not use computer link module or data link module

1) Functions

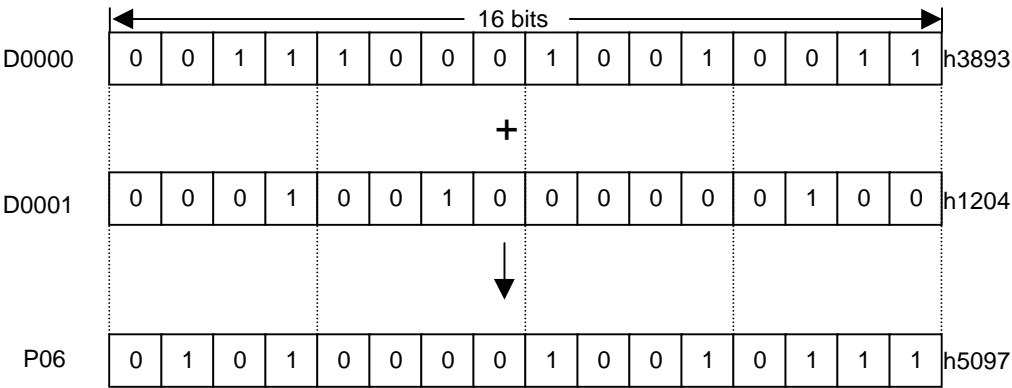
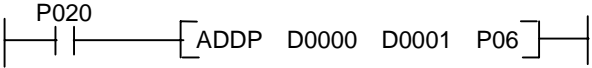
- ADDB(P) : Performs the addition of 16-bits BCD data specified at [S1] and [S2]. The addition result is stored at the device specified at [D].
- DADDB(P) : Performs the addition of 32-bits BIN data specified at [S1+1, S1] and [S2+1, S2]. The addition result is stored at the device specified at [D1+1, D1].
- When the addition result is over h9999(ADD / ADDP) or h99999999(DADD / DADDP), the carry flag (F112) is set.
- When the addition result is 0, the zero flag is set.
- If indirect address specified by #D format is out of device range or contents of [S1] and [S2] are invalid BCD format (out of 0 ~ 9), the operation error occur and the error flag (F110) is set.

- Execution condition



2) Program example

- When a rising edge is detected at P020, add BCD data of D0000 and D0001 and store the addition result to P06 word.



5.9.2 SUBB, SUBBP, DSUBB, DSUBBP

SUBB (BCD subtraction)	FUN(134) SUBB FUN(136) DSUBB FUN(135) SUBBP FUN(137) DSUBBP	Applicable CPU	All CPUs
---------------------------	--	-------------------	----------

Instructions		Available Device											Steps	Flag		
		M	P	K	L	F	T	C	S	D	#D	Integer		Error (F110)	Zero (F111)	Carry (F112)
SUBB(P) DSUBB(P)	(S1)	O	O	O	O	O	O	O		O	O	O	7/9/11	O	O	O
	(S2)	O	O	O	O	O	O	O		O	O	O				
	(D)	O	O	O	O*		O	O		O	O					

Operand setting

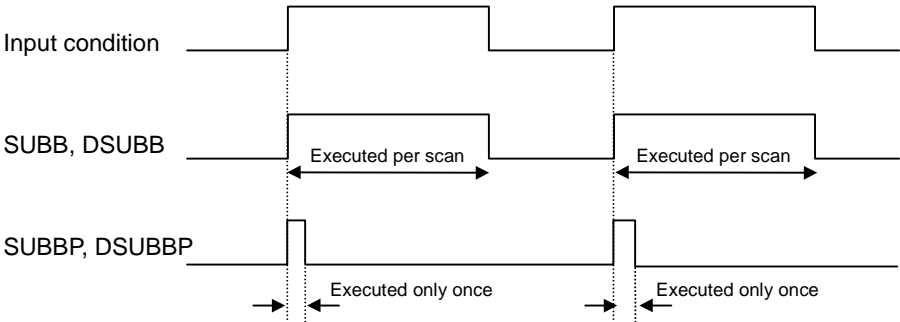
(S1)	The device storing minuend
(S2)	The device storing subtrahend
(D)	The device at which the subtraction result is stored

* Available only when do not use computer link module or data link module

1) Functions

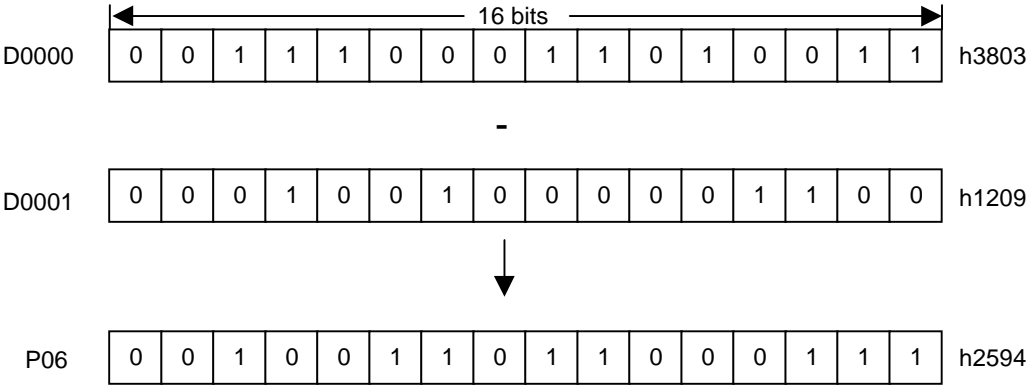
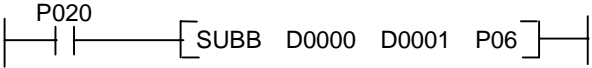
- SUBB(P) : Performs the subtraction of 16-bits BCD data specified at [S1] and [S2]. The subtraction result is stored at the device specified at [D].
- DSUBB(P) : Performs the subtraction of 32-bits BCD data specified at [S1+1, S1] and [S2+1, S2]. The subtraction result is stored at the device specified at [D1+1, D1].
- When the minuend is less than subtrahend, the LSB will underflow, and the carry flag (F112) will be set.
- When the subtraction result is 0, the zero flag is set.
- If indirect address specified by #D format is out of device range or contents of [S1] and [S2] are invalid BCD format (out of 0 ~ 9), the operation error occurs and the error flag (F110) is set.

- Execution condition



2) Program example

- When a rising edge is detected at P020, subtract contents of D0000 and D0001 and store the addition result to P06 word.



5.9.3 MULB, MULBP, DMULB, DMULBP

MULB (BCD multiply)	FUN(140) MULB FUN(142) DMULB FUN(141) MULBP FUN(143) DMULBP	Applicable CPU	All CPUs
------------------------	--	-------------------	----------

Instructions		Available Device											Steps	Flag		
		M	P	K	L	F	T	C	S	D	#D	Integer		Error (F110)	Zero (F111)	Carry (F112)
MULB(P) DMULB(P)	(S1)	O	O	O	O	O	O	O		O	O	O	7/9/11	O	O	
	(S2)	O	O	O	O	O	O	O		O	O	O				
	(D)	O	O	O	O*		O	O		O	O					

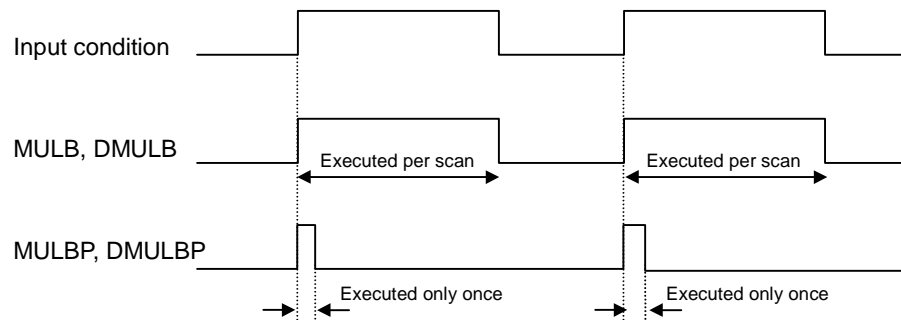
Operand setting

(S1)	The device storing multiplicand
(S2)	The device storing multiplier
(D)	The device at which the multiplication result is stored

* Available only when do not use computer link module or data link module

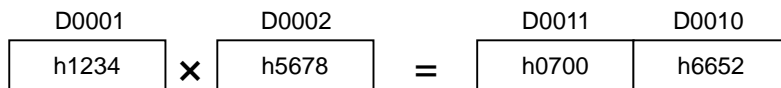
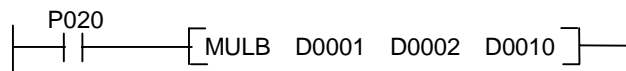
1) Functions

- MULB(P) : Performs the multiplication of BCD data specified as [S1] and the BCD data specified as [S2], and stores the multiplication result into the device specified as [D+1, D].
- DMULB(P) : Performs the multiplication of BCD data specified as [S1+1, S] and the BCD data specified as [S2+1, S2], and stores the multiplication result into the device specified as [D+3, D+2, D+1, D].
- If the multiplication result is zero, the zero flag will be set.
- If indirect address specified by #D format is out of device range or contents of [S1] and [S2] is invalid BCD format (out of 0 ~ 9), the operation error occurs and the error flag (F110) is set.
- Execution conditions

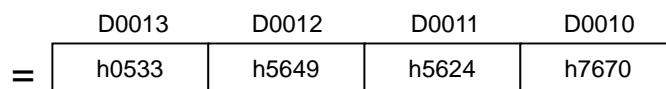


2) Program example

- Program which stores the multiplication result of D0001 and D0002 at D0010, D0011 while P020 is on.



- Program which stores the multiplication result of D0001, D0002 and D0003,D0004 at D0010 ~ D0013 when P020 is switched on.



5.9.4 DIVB, DIVBP, DDIVB, DDIVBP

DIVB (BCD divide)	FUN(144) DIVB FUN(146) DDIVB FUN(145) DIVBP FUN(147) DDIVBP	Applicable CPU	All CPUs
----------------------	--	-------------------	----------

Instructions		Available Device											Steps	Flag		
		M	P	K	L	F	T	C	S	D	#D	Integer		Error (F110)	Zero (F111)	Carry (F112)
DIVB(P) DDIVB(P)	(S1)	O	O	O	O	O	O	O		O	O	O	7/9/11	O	O	
	(S2)	O	O	O	O	O	O	O		O	O	O				
	(D)	O	O	O	O*		O	O		O	O					

Operand setting

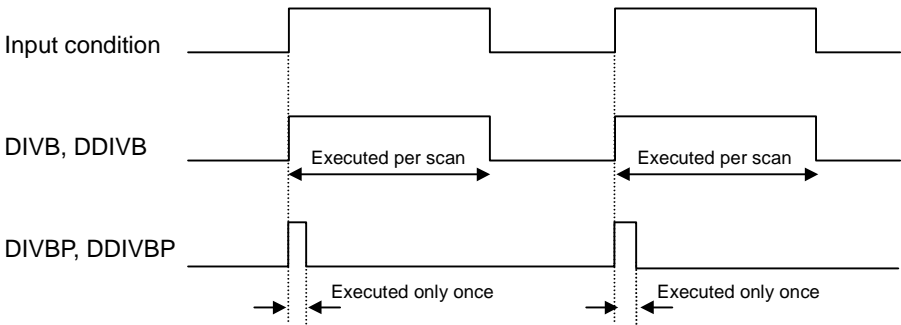
(S1)	The device storing dividend
(S2)	The device storing divider
(D)	The device at which the division result is stored

* Available only when do not use computer link module or data link module

1) Functions

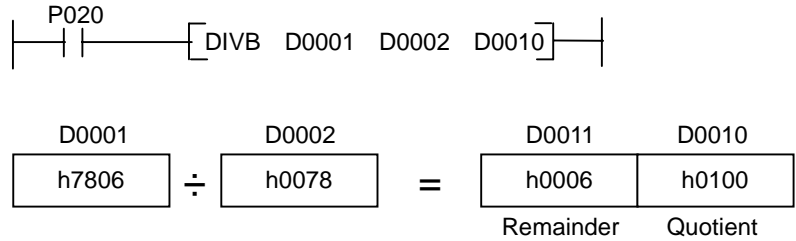
- DIVB(P) : Performs the division of BCD data specified as [S1] and the BCD data specified as [S2], and stores the division result into the device specified as [D+1, D]. The quotient is stored at [D], and the remainder is stored at [D+1].
- DDIVB(P) : Performs the multiplication of BCD data specified as [S1+1, S] and the BCD data specified as [S2+1, S2], and stores the multiplication result into the device specified as [D+3, D+2, D+1, D]. The quotient is stored at [D+1, D], and the remainder is stored at [D+3, D+2].
- If the quotient is zero, the zero flag will be set.
- If indirect address specified by #D format is out of device range or the content of divider is 0 or contents of [S1] and [S2] is invalid BCD format (out of 0 ~ 9), the operation error occurs and the error flag (F110) is set.

- Execution conditions

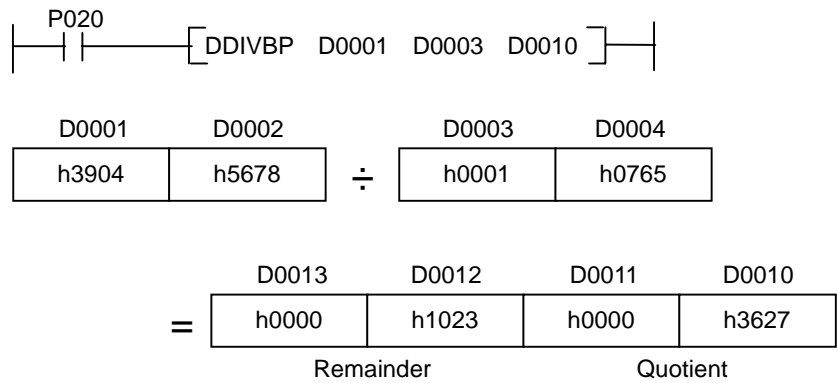


2) Program example

- Program which stores the division result of D0001 and D0002 at D0010, D0011 while P020 is on.



- Program which stores the division result of D0001, D0002 and D0003,D0004 at D0010 ~ D0013 when P020 is switched on.



5.10 Logical arithmetic instructions

5.10.1 WAND, WANDP, DWAND, DWANDP

WAND (Word AND)	FUN(150) WAND FUN(152) DWAND FUN(151) WANDP FUN(153) DWANDP	Applicable CPU	All CPUs
--------------------	--	-------------------	----------

Instructions		Available Device											Steps	Flag		
		M	P	K	L	F	T	C	S	D	#D	Integer		Error (F110)	Zero (F111)	Carry (F112)
WAND(P) DWAND(P)	(S1)	O	O	O	O	O	O	O		O	O	O	7/9/11	O	O	
	(S2)	O	O	O	O	O	O	O		O	O	O				
	(D)	O	O	O	O*		O	O		O	O					

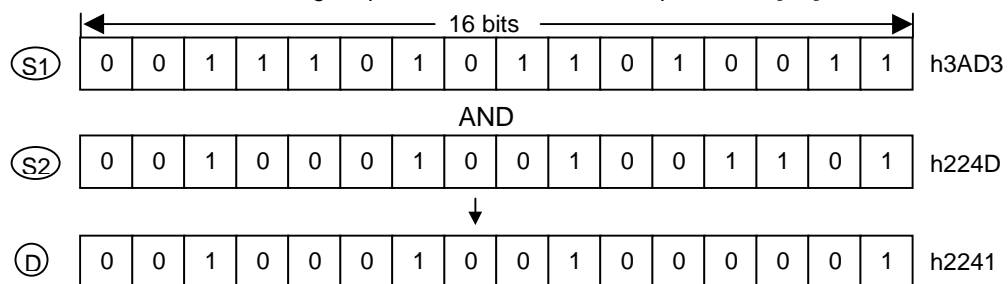
Operand setting

(S1)	Data for which logical product will be performed
(S2)	
(D)	The device at which the result of logical product is stored

* Available only when do not use computer link module or data link module

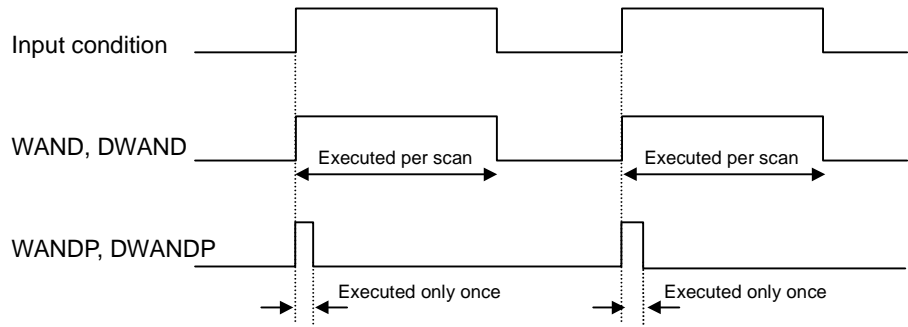
1) Functions

- WAND(P) : Performs the logical product of the 16-bit data of device specified at [S1] and [S2] per bit. Then stores the result of logical production into the device specified at [D].



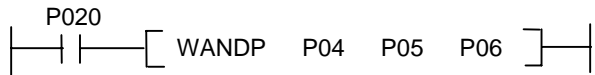
- DWAND(P) : Performs the logical product of the 32-bit data of device specified as [S1+1, S1] and [S2+1, S2] per bit, and stores the result into the device specified at [D+1, D].
- If the result of logical product is 0, the zero flag (F111) is set.
- If the indirect address specified by #D format is out of device range, the operation error occurs and the error flag (F110) is set.

- Execution conditions



2) Program example

- Program which performs the logical production of the contents of P04 and P05 words, then stores the result to the P06 word when the P020 is switched on.

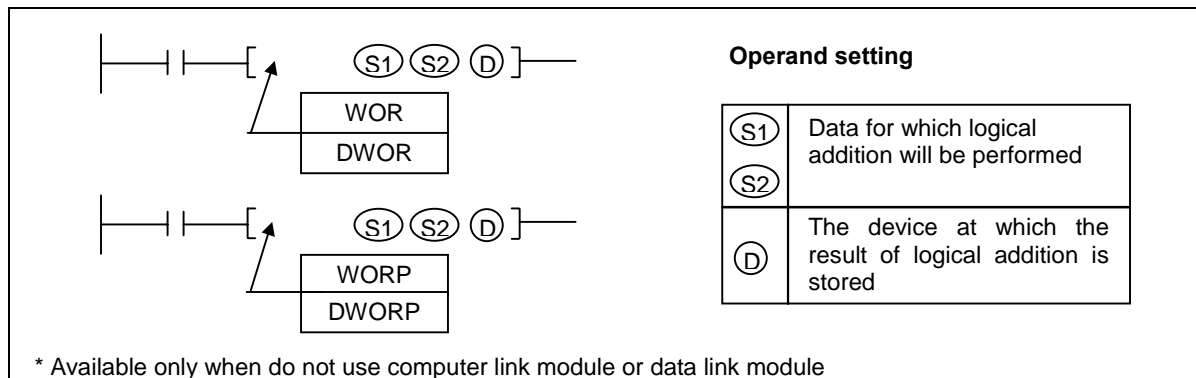


	16 bits																
P04	0	0	1	1	1	0	1	0	1	1	0	1	0	0	1	1	h3AD3
	AND																
P05	0	0	1	0	0	0	1	0	0	1	0	0	1	1	0	1	h224D
	↓																
P06	0	0	1	0	0	0	1	0	0	1	0	0	0	0	0	1	h2241

5.10.2 WOR, WOPR, DWOR, DWORP

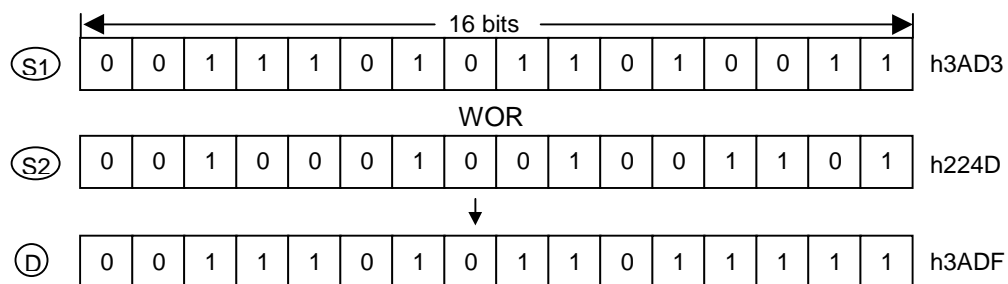
WOR (Word OR)	FUN(154) WOR FUN(156) DWOR FUN(155) WOPR FUN(157) DWORP	Applicable CPU	All CPUs
------------------	--	-------------------	----------

Instructions		Available Device										Steps	Flag		
		M	P	K	L	F	T	C	S	D	#D	Integer	Error (F110)	Zero (F111)	Carry (F112)
WOR(P) DWOR(P)	(S1)	O	O	O	O	O	O	O		O	O	O	7/9/11	O	O
	(S2)	O	O	O	O	O	O	O		O	O	O			
	(D)	O	O	O	O*		O	O		O	O				



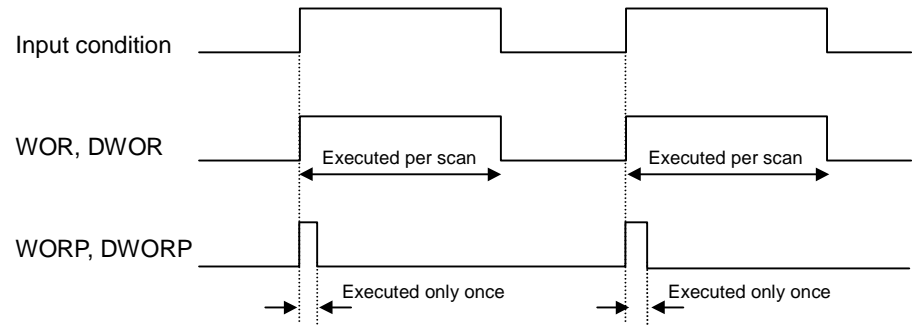
1) Functions

- WOR(P) : Performs the logical addition of the 16-bit data of device specified at [S1] and [S2] per bit. Then stores the result of logical addition into the device specified at [D].



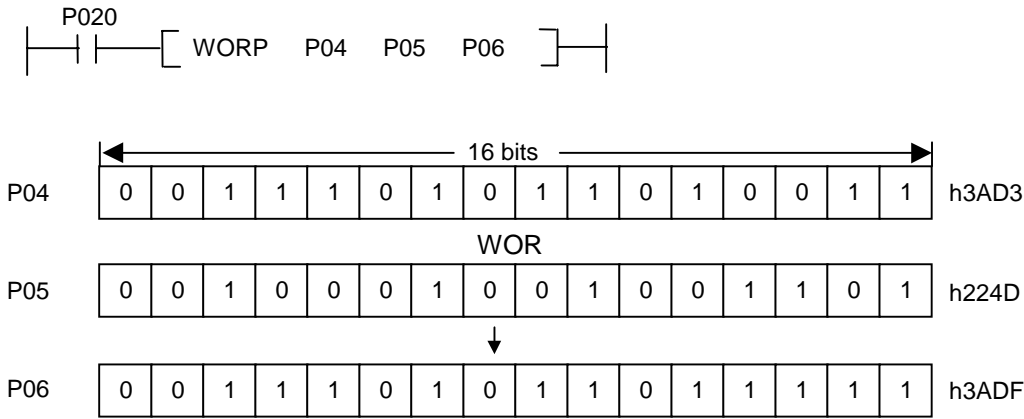
- DWOR(P) : Performs the logical addition of the 32-bit data of device specified as [S1+1, S1] and [S2+1, S2] per bit, and stores the result into the device specified at [D+1, D].
- If the result of logical addition is 0, the zero flag (F111) is set.
- If the indirect address specified by #D format is out of device range, the operation error occurs and the error flag (F110) is set.

- Execution conditions



2) Program example

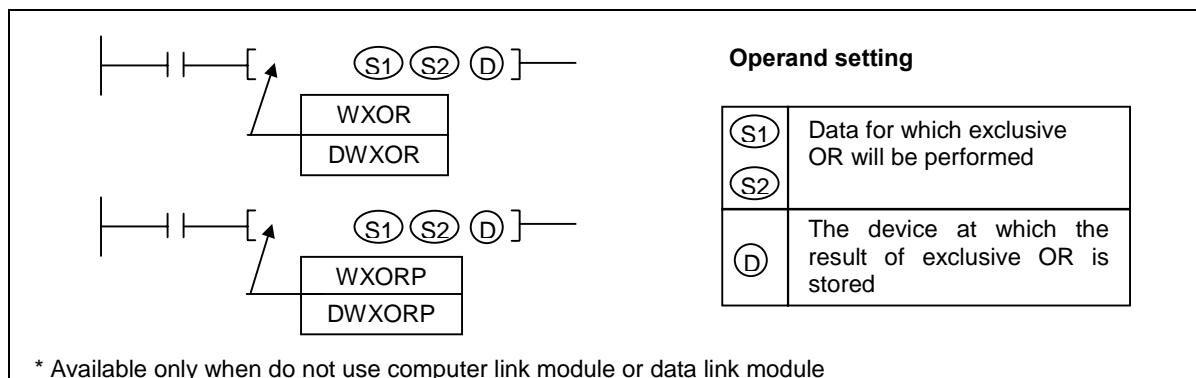
- Program that performs the logical addition of the contents of P04 and P05 words, then stores the result to the P06 word when the P020 is switched on.



5.10.3 WXOR, WXORP, DWXOR, DWXORP

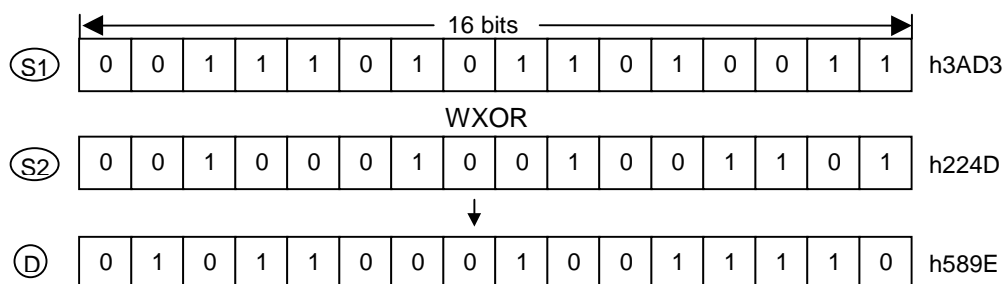
WXOR (Word exclusive OR)	FUN(160) WXOR FUN(162) DWXOR FUN(161) WXORP FUN(163) DWXORP	Applicable CPU	All CPUs
-----------------------------	--	-------------------	----------

Instructions		Available Device											Steps	Flag		
		M	P	K	L	F	T	C	S	D	#D	Integer		Error (F110)	Zero (F111)	Carry (F112)
WXOR(P) DWXOR(P)	(S1)	O	O	O	O	O	O	O		O	O	O	7/9/11	O	O	
	(S2)	O	O	O	O	O	O	O		O	O	O				
	(D)	O	O	O	O*		O	O		O	O					



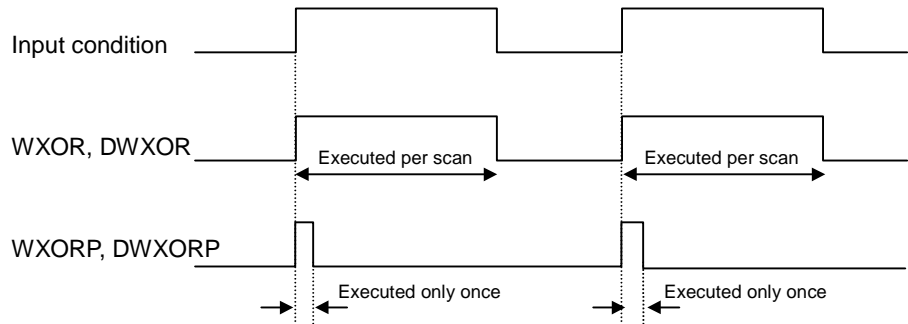
1) Functions

- WXOR(P) : Performs the exclusive OR of the 16-bit data of device specified at [S1] and [S2] per bit. Then stores the result of logical production into the device specified at [D].



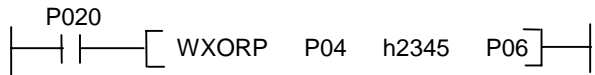
- DWXOR(P) : Performs the exclusive OR of the 32-bit data of device specified as [S1+1, S1] and [S2+1, S2] per bit, and stores the result into the device specified at [D+1, D].
- If the result of exclusive OR is 0, the zero flag (F111) is set.
- If the indirect address specified by #D format is out of device range, the operation error occurs and the error flag (F110) is set.

- Execution conditions



2) Program example

- Program that performs the exclusive OR of the contents of P04 and h2345, then stores the result to the P06 word when the P020 is switched on.



	16 bits																
P04	0	0	1	1	1	0	1	0	1	1	0	1	0	0	1	1	h3AD3
WXOR																	
h2345	0	0	1	0	0	0	1	1	0	1	0	0	0	1	0	1	h2345
↓																	
P06	0	0	0	1	1	0	0	1	1	0	0	1	0	1	1	0	h1996

5.10.4 WXNR, WXNRP, DWXNR, DWXNRP

WXOR (Word exclusive NOR)	FUN(164) WXOR FUN(166) DWXOR FUN(165) WXORP FUN(167) DWXORP	Applicable CPU	All CPUs
------------------------------	--	----------------	----------

Instructions		Available Device											Steps	Flag		
		M	P	K	L	F	T	C	S	D	#D	Integer		Error (F110)	Zero (F111)	Carry (F112)
WXNR(P) DWXNR(P)	(S1)	O	O	O	O	O	O	O		O	O	O	7/9/11	O	O	
	(S2)	O	O	O	O	O	O	O		O	O	O				
	(D)	O	O	O	O*		O	O		O	O					

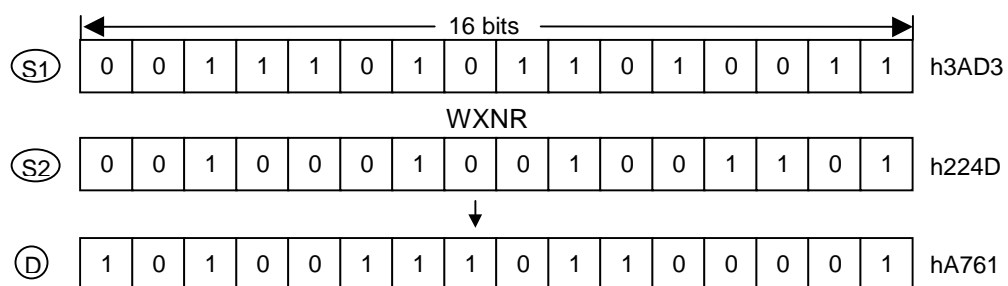
Operand setting

(S1)	Data for which exclusive NOR will be performed
(S2)	
(D)	The device at which the result of exclusive NOR is stored

* Available only when do not use computer link module or data link module

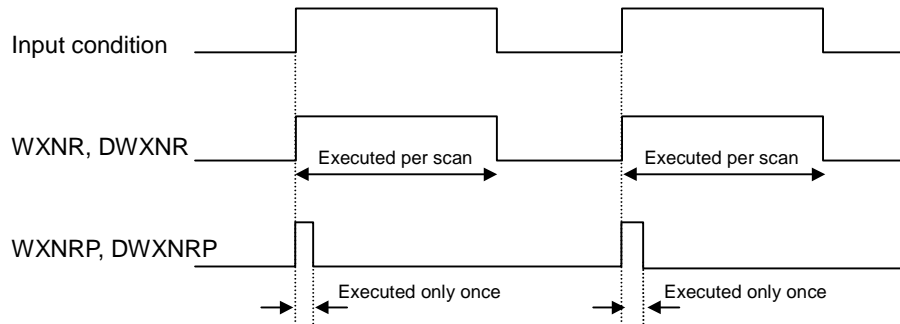
1) Functions

- WXNR(P) : Performs the exclusive NOR of the 16-bit data of device specified at [S1] and [S2] per bit. Then stores the result of logical production into the device specified at [D].



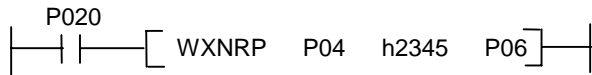
- DWXNR(P) : Performs the exclusive NOR of the 32-bit data of device specified as [S1+1, S1] and [S2+1, S2] per bit, and stores the result into the device specified at [D+1, D].
- If the result of exclusive NOR is 0, the zero flag (F111) is set.
- If the indirect address specified by #D format is out of device range, the operation error occurs and the error flag (F110) is set.

- Execution conditions



2) Program example

- Program that performs the exclusive NOR of the contents of P04 and h2345, then stores the result to the P06 word when the P020 is switched on.



	16 bits																
P04	0	0	1	1	1	0	1	0	1	1	0	1	0	0	1	1	h3AD3
WXOR																	
h2345	0	0	1	0	0	0	1	1	0	1	0	0	0	1	0	1	h2345
↓																	
P06	1	1	1	0	0	1	1	0	0	1	1	0	1	0	0	1	hE669

5.11 Data processing instructions

5.11.1 SEG, SEGP

SEG (7 segment)	FUN(174) SEG FUN(175) SEGP	Applicable CPU	All CPUs
--------------------	-------------------------------	-------------------	----------

Instructions		Available Device											Steps	Flag		
		M	P	K	L	F	T	C	S	D	#D	Integer		Error (F110)	Zero (F111)	Carry (F112)
SEG	Ⓢ	O	O	O	O	O	O	O		O	O		7	O		
SEGP	Ⓓ	O	O	O	O*		O	O		O	O					
	Cw											O				

Operand setting

Ⓢ	The device at which source data is stored
Ⓓ	The device which will store 7 segment display data
Cw	Information for start bit and numbers of transferred bits

* Available only when do not use computer link module or data link module

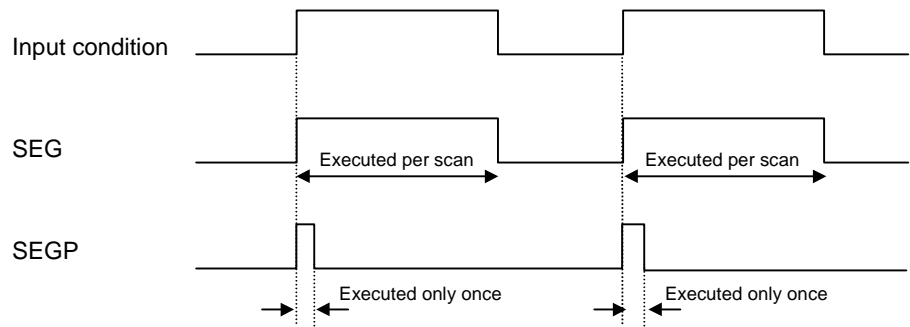
1) Functions

- The format of 'Cw'

h	s	d	x	z
---	---	---	---	---

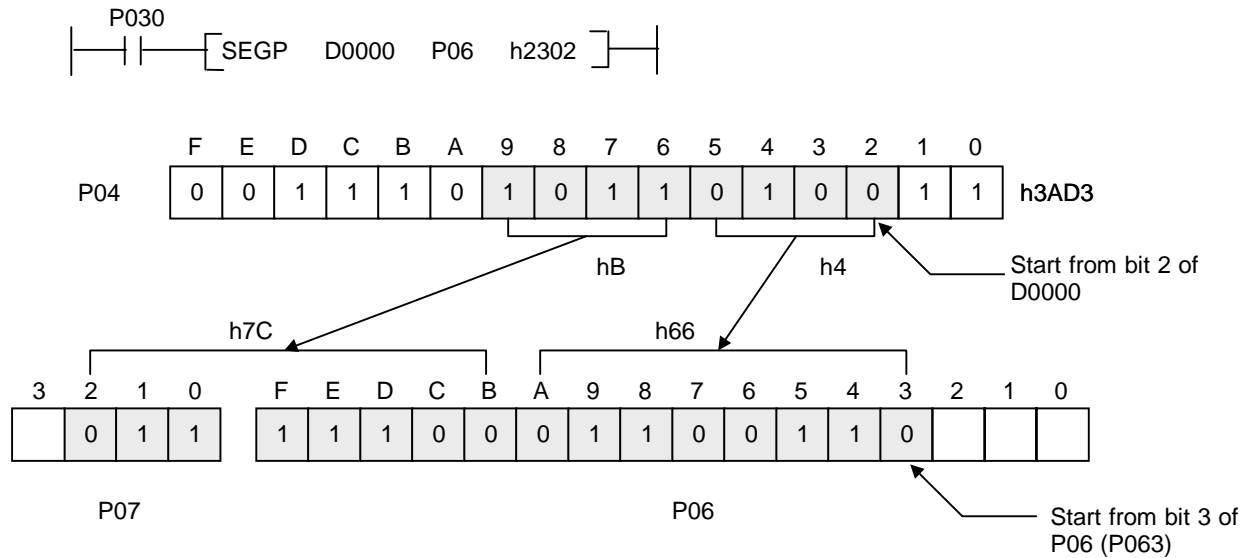
- s : The start bit of [S].
 - d : The start bit of [D]
 - x : Don't care
 - x : Numbers of decoded nibbles. (range : 0 ~ F)
- Decodes the data of $z \times 4$ bits block that start from the s^{th} bit of device specified at [S] into 7 segment display data and stores the result to the $z \times 8$ bits block that starts from the s^{th} bit of device [D].

- Execution conditions

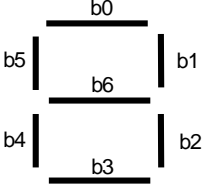


2) Program example

- Program that decodes 8-bits from the bit 2 of D0000 into 7 segment display format, and stores the result to 16-bits from the bit 3 of P06 when P030 is switched on.



3) 7 segment display data

S		Configuration of 7 segment	D								Displayed data
Hex	Binary		b7	b6	b5	b4	b3	b2	b1	b0	
0	0000		0	0	1	1	1	1	1	1	0
1	0001		0	0	0	0	0	1	1	0	1
2	0010		0	1	0	1	1	0	1	1	2
3	0011		0	1	0	0	1	1	1	1	3
4	0100		0	1	1	0	0	1	1	0	4
5	0101		0	1	1	0	1	1	0	1	5
6	0110		0	1	1	1	1	1	0	1	6
7	0111		0	0	1	0	0	1	1	1	7
8	1000		0	1	1	1	1	1	1	1	8
9	1001		0	1	1	0	1	1	1	1	9
A	1010		0	1	1	1	0	1	1	1	A
B	1011		0	1	1	1	1	1	0	0	B
C	1100		0	0	1	1	1	0	0	1	C
D	1101		0	1	0	1	1	1	1	0	D
E	1110		0	1	1	1	1	0	0	1	E
F	1111		0	1	1	1	0	0	0	1	F

5.11.2 ASC, ASCP

ASC (ASCII code)	FUN(190) ASC FUN(191) ASCP	Applicable CPU	All CPUs
---------------------	-------------------------------	-------------------	----------

Instructions		Available Device											Steps	Flag		
		M	P	K	L	F	T	C	S	D	#D	Integer		Error (F110)	Zero (F111)	Carry (F112)
ASC ASCP	Ⓢ	O	O	O	O	O	O	O		O	O		7	O		
	Ⓓ	O	O	O	O*		O	O		O	O					
	Cw											O				

Operand setting

Ⓢ	The device at which source data is stored
Ⓓ	The device which will store ASCII data
Cw	Information for start bit and numbers of transferred bits

* Available only when do not use computer link module or data link module

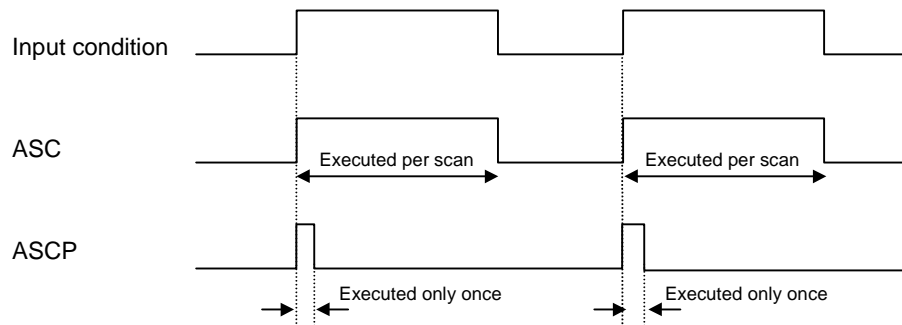
1) Functions

- The format of 'Cw'

h	s	d	x	z
---	---	---	---	---

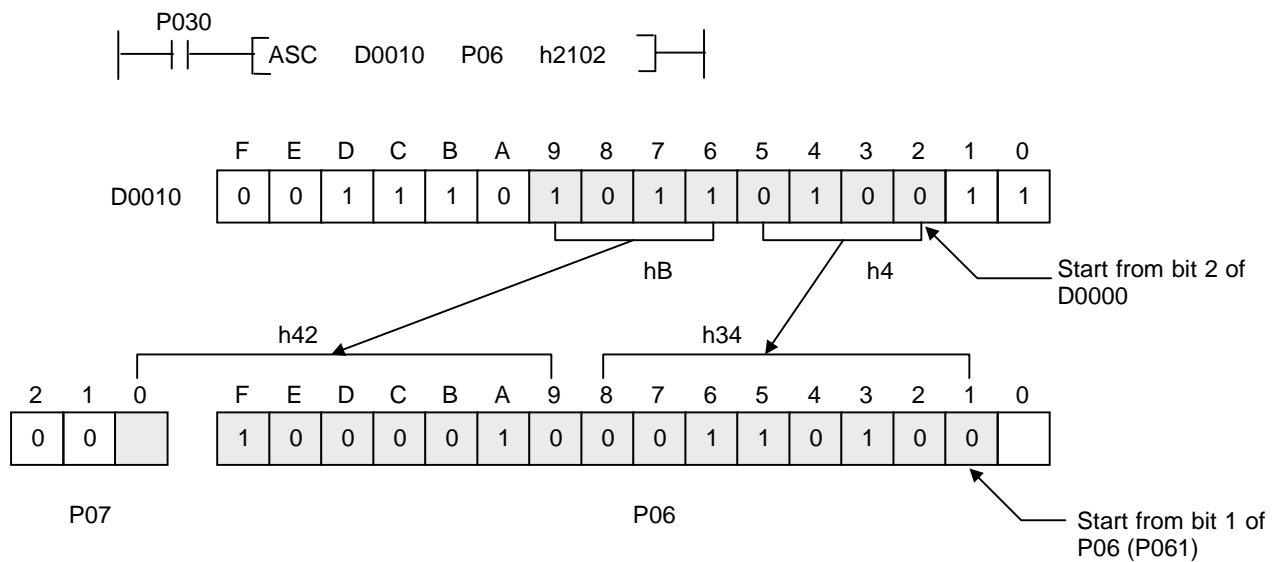
- e) s : The start bit of [S].
- f) d : The start bit of [D]
- g) x : Don't care
- h) x : Numbers of decoded nibbles. (range : 0 ~ F)
- Converts the data of $z \times 4$ bits block that start from the s^{th} bit of device specified at [S] into ASCII code and stores the result to the $z \times 8$ bits block that starts from the s^{th} bit of device [D].

- Execution conditions



2) Program example

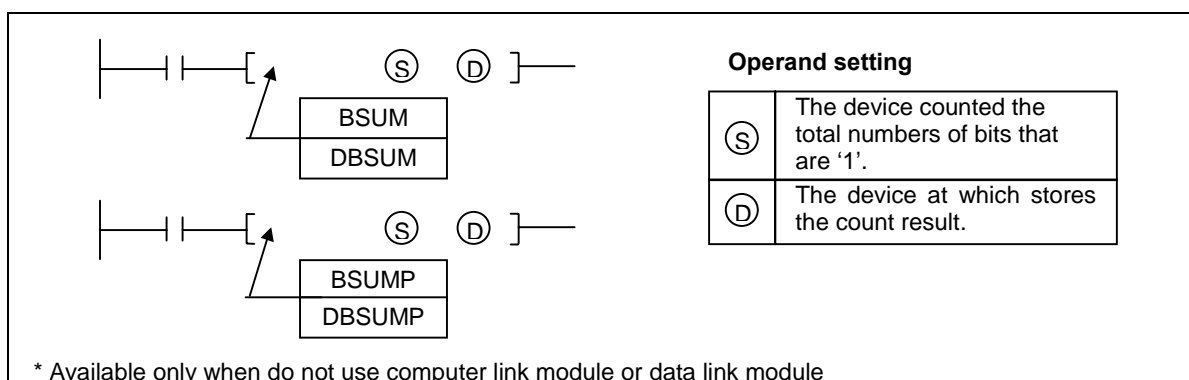
- Program that converts 8-bits from the bit 2 of D0010 into ASCII code data, and stores the result to 16-bits from the bit 1 of P06 while P030 is on.



5.11.3 BSUM, BSUMP, DBSUM, DBSUMP

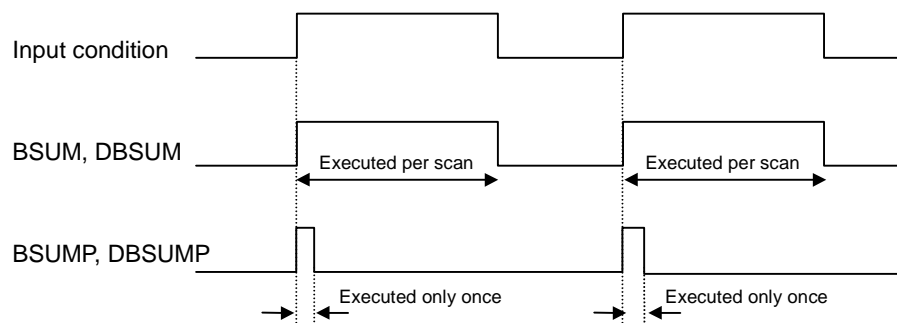
BSUM (Bit summary)	FUN(170) BSUM FUN(172) DBSUM FUN(171) BSUMP FUN(173) DBSUMP	Applicable CPU	All CPUs
-----------------------	--	-------------------	----------

Instructions		Available Device											Steps	Flag		
		M	P	K	L	F	T	C	S	D	#D	Integer		Error (F110)	Zero (F111)	Carry (F112)
BSUM(P)	Ⓢ	O	O	O	O	O	O	O		O	O	O	5	O	O	
DBSUM(P)	ⓓ	O	O	O	O*		O	O		O	O					



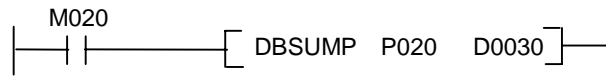
1) Functions

- BSUM(P) : Counts the numbers of '1' in the device specified as [S], then stores the result into the device specified as [D] in hexadecimal format.
- DBSUM(P) : Counts the numbers of '1' in the device specified as [S+1, S], then stores the result into the device specified as [D] in hexadecimal format.
- If the count result is 0, the zero flag is set.
- If the indirect address specified by #D format is out of device range, the operation error occurs and the error flag (F110) is set.
- Execution conditions



2) Program example

- Program that count the numbers of '1' in P020 and P021, then store the count result into D0030 when M020 is switched on.



	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0	
P020	0	0	1	1	1	0	1	0	1	1	0	1	0	0	1	1	The numbers of 1 = 9

	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0	
P021	1	0	1	0	0	0	0	1	1	0	1	1	0	1	1	0	The numbers of 1 = 8



	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0	
D0030	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	Total numbers of 1 = 17 (h0011)

5.11.4 ENCO, ENCOP

ENCO (Encode)	FUN(176) ENCO FUN(177) ENCOP	Applicable CPU	All CPUs
------------------	---------------------------------	-------------------	----------

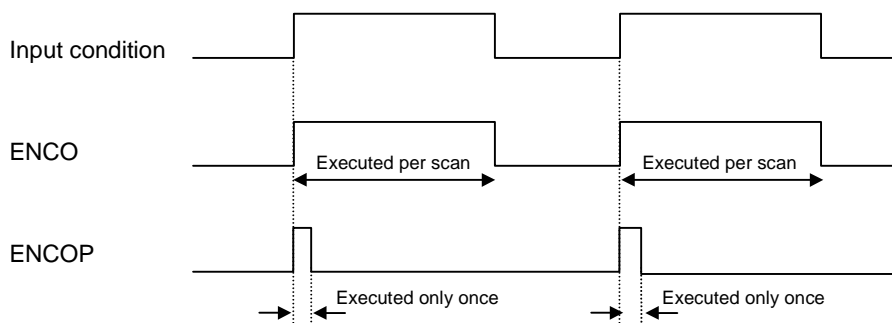
Instructions		Available Device											Steps	Flag		
		M	P	K	L	F	T	C	S	D	#D	Integer		Error (F110)	Zero (F111)	Carry (F112)
ENCO	Ⓢ	O	O	O	O	O	O	O		O	O	O	7	O		
ENCOP	ⓓ	O	O	O	O*		O	O		O	O					
	n									O		O				

Operand setting	
Ⓢ	The start address of source data area
ⓓ	The start address of destination area will store encoding result
n	Effective bit length 2^n (1 ~ 8)

* Available only when do not use computer link module or data link module

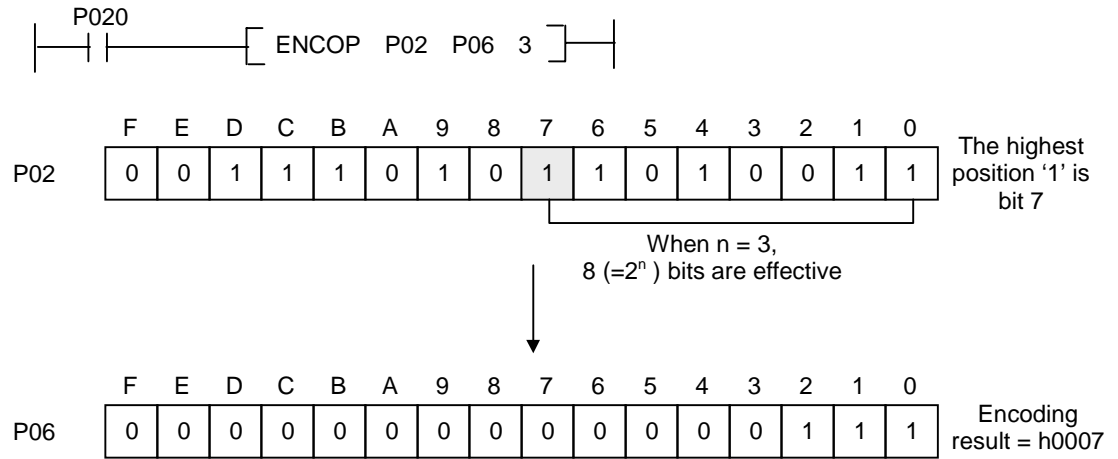
1) Functions

- Encodes the data of 2^n bits, which begin the bit 0 of device specified as [S], and stores the result to the device specified as [D].
- For 'n', 1 ~ 8 can be specified. If the value of n is out of this range, no processing is performed and the contents of [D] is not changed.
- When multiple bits are 1, processing is performed for the most significant bit. If the value of n is 0, the zero flag (F111) will be set.
- When the value of n is larger than 4, the source data area is expanded like [S+1], [S+2], ...
When n=8, the length of source data is 256 bits. ([S+15, S+14, ..., S+1, S])
- Execution conditions



2) Program example

- Program that encode the 8-bits (bit 0 ~ bit 7) of P06 word and stores the encoding result to the P06 word when P020 is switched on.



- Program that encode the current value of counter C000 and stores the encoding result at the P05 word. The counter C000 is increased at every 1 sec.

5.11.5 DECO, DECOP

DECO (Decode)	FUN(178) DECO FUN(179) DECOP	Applicable CPU	All CPUs
------------------	---------------------------------	-------------------	----------

Instructions		Available Device											Steps	Flag		
		M	P	K	L	F	T	C	S	D	#D	Integer		Error (F110)	Zero (F111)	Carry (F112)
DECO	Ⓢ	O	O	O	O	O	O	O		O	O	O	7	O		
DECOP	ⓓ	O	O	O	O*		O	O		O	O					
	n									O		O				

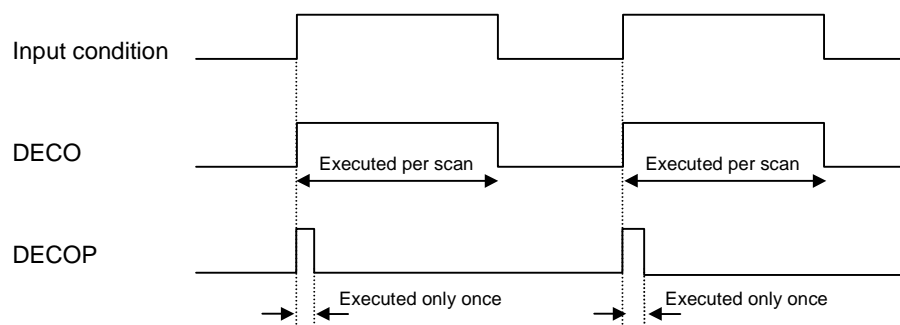
Operand setting

Ⓢ	The start address of source data area
ⓓ	The start address of destination area will store decoding result
n	Effective bit length 2^n (1 ~ 8)

* Available only when do not use computer link module or data link module

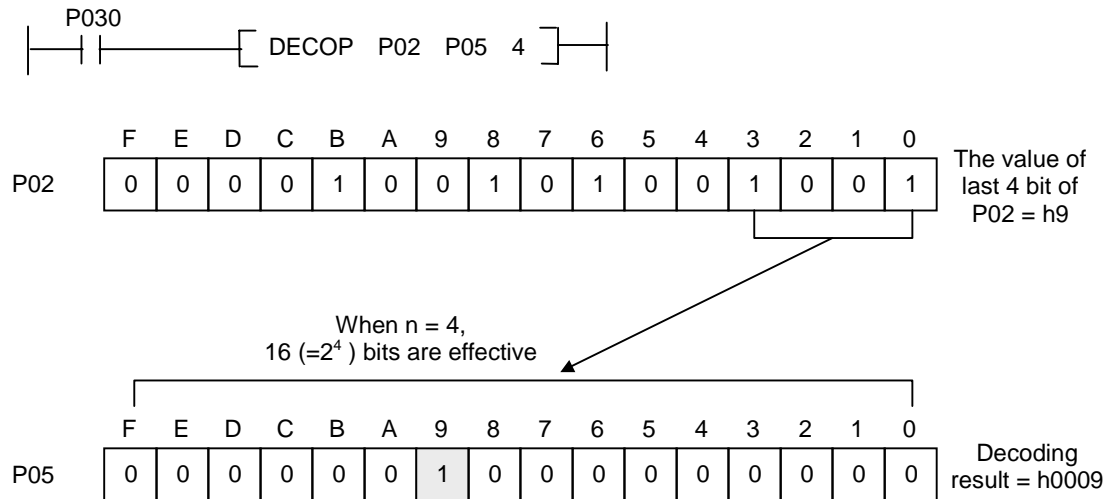
1) Functions

- Decodes the data of lower n bits of the device specified as [S], and stores the decoding result to the block of 2^n bits that start from the bit 0 of the device specified as [D].
- For 'n', 1 ~ 8 can be specified. If the value of n is 0, no processing is performed and the contents of [D] is not changed. If the value of n is over 8, the error flag (F110) is set and no processing is performed.
- When the value of n is larger than 4, the source data area is expanded like [D+1], [D+2], ... When n=8, the length of decoding result data is 256 bits. ([D+15, D+14, ..., D+1, D])
- Execution conditions

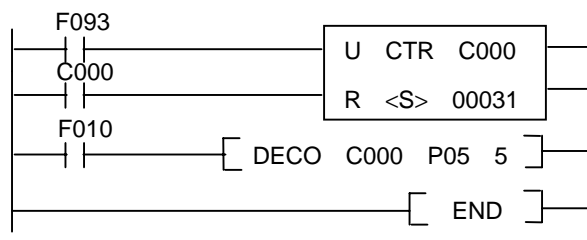


2) Program example

- Program that decode lower 4 bits of p02 word and stores the decoding result to P05 word when P030 is switched on.



- Program that decodes the current value of counter C000 and stores the decoding result to P05 and P06 word. The current value of counter is increased at every 1 second and when the current value reaches to 31, the counter C000 is reset.



5.11.6 FILR, FILRP, DFILR, DFILRP

FILR (File table read)	FUN(180) FILR FUN(182) DFILR FUN(181) FILRP FUN(183) DFILRP	Applicable CPU	All CPUs
---------------------------	--	-------------------	----------

Instructions	Available Device											Steps	Flag		
		M	P	K	L	F	T	C	S	D	#D	Integer	Error (F110)	Zero (F111)	Carry (F112)
FILR(P)	Ⓢ	O	O	O	O	O	O	O		O	O	O	7	O	
DFILR(P)	Ⓓ	O	O	O	O*		O	O		O	O				
	n									O		O			

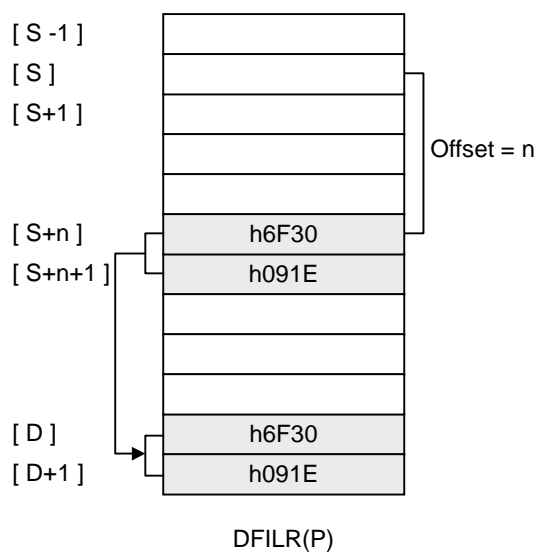
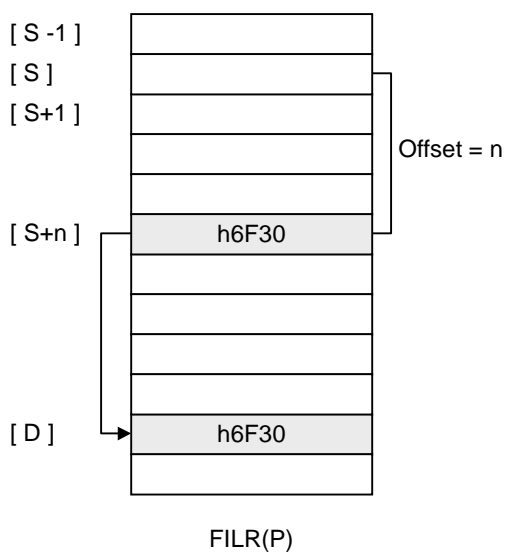
Operand setting

Ⓢ	The origin address of the source data word
Ⓓ	The destination word at which the contents of [S+n] word
n	The offset

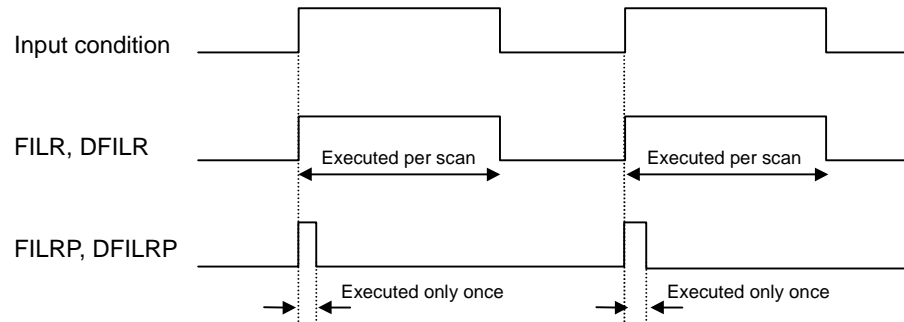
* Available only when do not use computer link module or data link module

1) Functions

- FILR(P) : Transfers the content of [S+n] word to the device specified as [D]
- DFILR(P) : Transfers the contents of [S+n+1, S+n] to the device specified as [D+1, D]
- When the [S+n] is over the range of corresponding device area, the error flag is set and no processing is performed.

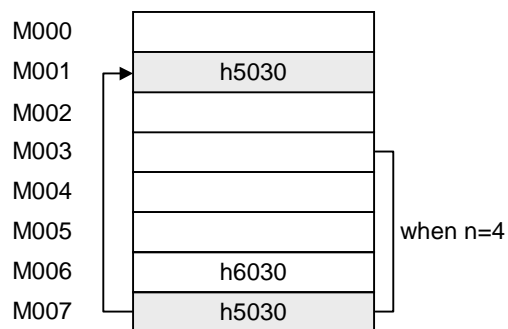
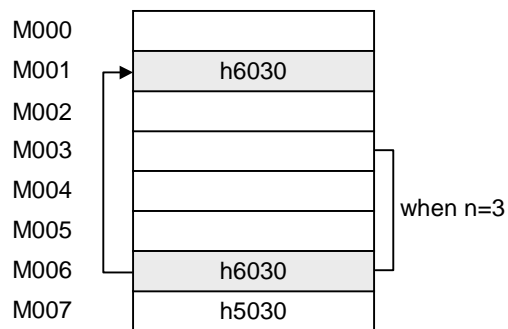
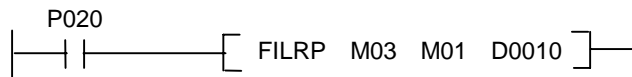


- Execution conditions



2) Program example

- Program that transfer the content of the M03+n word to M01 word when P020 is switched on. The n is stored at D0010 word.



5.11.7 FILW, FILWP, DFILW, DFILWP

FILW (File table write)	FUN(184) FILW FUN(186) DFILW FUN(185) FILWP FUN(187) DFILWP	Applicable CPU	All CPUs
----------------------------	--	-------------------	----------

Instructions	Available Device											Steps	Flag		
	M	P	K	L	F	T	C	S	D	#D	Integer		Error (F110)	Zero (F111)	Carry (F112)
FILW(P)	Ⓓ	O	O	O	O*		O	O		O	O	7	O		
DFILW(P)	Ⓔ	O	O	O	O	O	O			O	O				
	n									O	O				

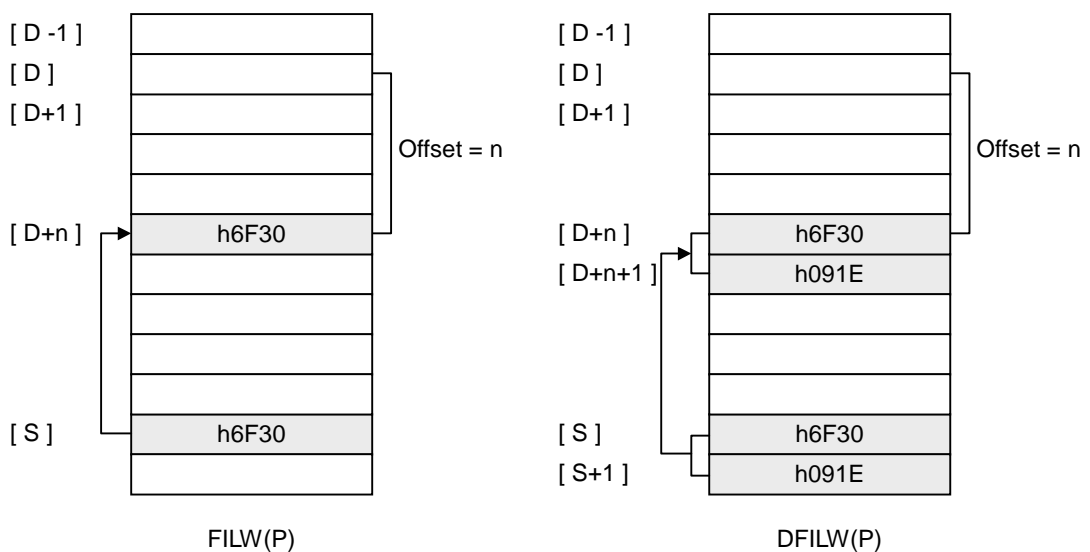
Operand setting

Ⓓ	The origin address of the destination.
Ⓔ	The source data or device at which the source data is stored
n	The offset

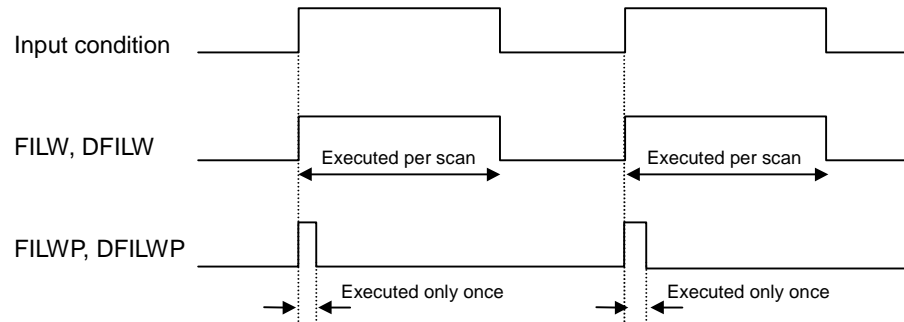
* Available only when do not use computer link module or data link module

1) Functions

- FILW(P) : Transfers the content of [S] word to the device specified as [D+n]
- DFILW(P) : Transfers the contents of [S+1, S] to the device specified as [D+n+1, D+n]
- When the [D+n] is over the range of corresponding device area, the error flag is set and no processing is performed.

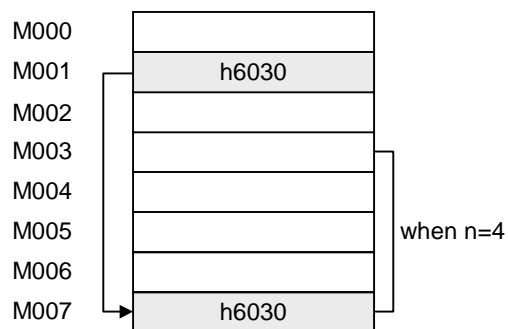
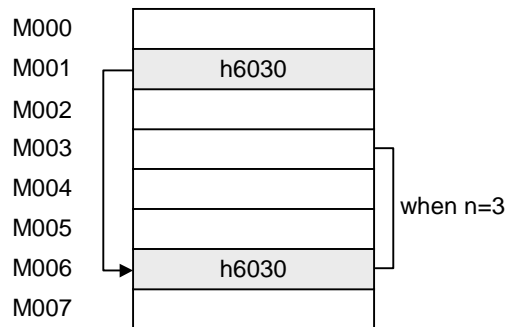


- Execution conditions



2) Program example

- Program that transfer the content of the M01 word to M03+n word when P020 is switched on. The n is stored at D0010 word.



5.11.8 DIS, DISP

DIS	FUN(194) DIS	Applicable CPU	All CPUs
(Data dissociation)	FUN(195) DISP		

Instructions		Available Device											Steps	Flag		
		M	P	K	L	F	T	C	S	D	#D	Integer		Error (F110)	Zero (F111)	Carry (F112)
DIS(P)	Ⓢ	O	O	O	O	O	O	O		O	O		7	O		
	Ⓓ	O	O	O	O*		O	O		O	O					
	n									O		O				

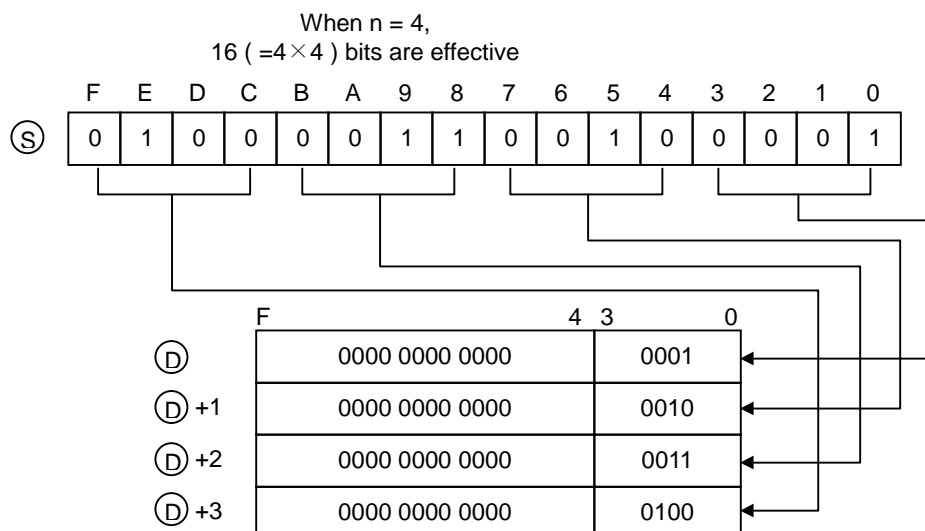
Operand setting

Ⓢ	The source device
Ⓓ	The start address of destination devices.
n	The number of nibble to be dissociated (1 ~ 4)

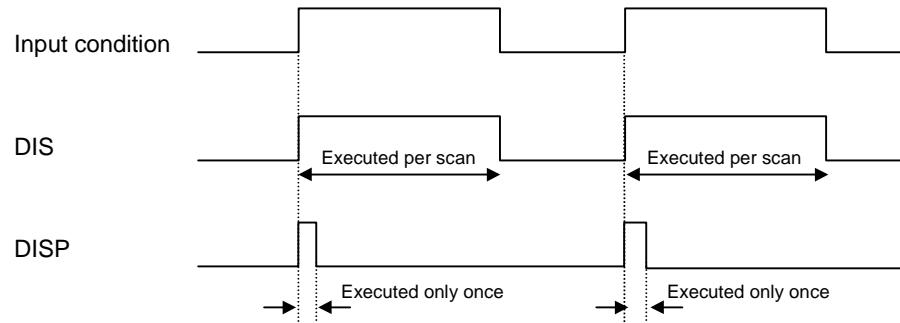
* Available only when do not use computer link module or data link module

1) Functions

- Transfers n nibbles that start from the bit 0 of device specified as [S] into the lower 4 bits of the block specified as [D+n-1] ~ [D].
- The higher 12 bits (bit 4 ~ bit F) of the block specified as [D+n-1] ~ [D] are cleared as 0.
- When n=0, no processing is performed.
- When n > 4, the error flag is set and no processing is performed.

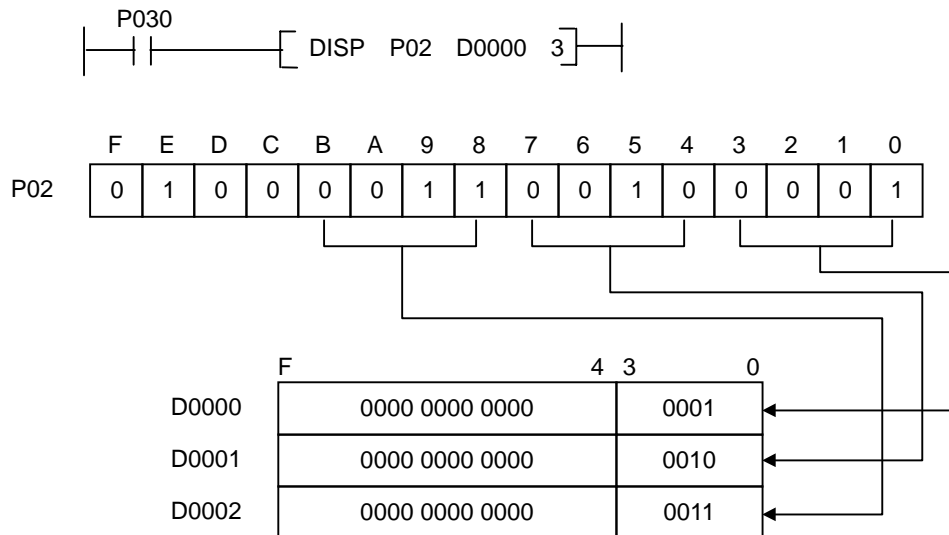


- Execution conditions



2) Program examples

- Program that dissociate the contents of lower 3 nibbles of P02 word to lower 4 bits of D0000 ~ D0003 when P030 is switched on.



5.11.9 UNI, UNIP

UNI	FUN(192) UNI	Applicable CPU	All CPUs
(Data association)	FUN(193) UNIP		

Instructions		Available Device											Steps	Flag		
		M	P	K	L	F	T	C	S	D	#D	Integer		Error (F110)	Zero (F111)	Carry (F112)
UNI(P)	Ⓢ	O	O	O	O	O	O	O		O	O		7	O		
	Ⓓ	O	O	O	O*		O	O		O	O					
	n									O		O				

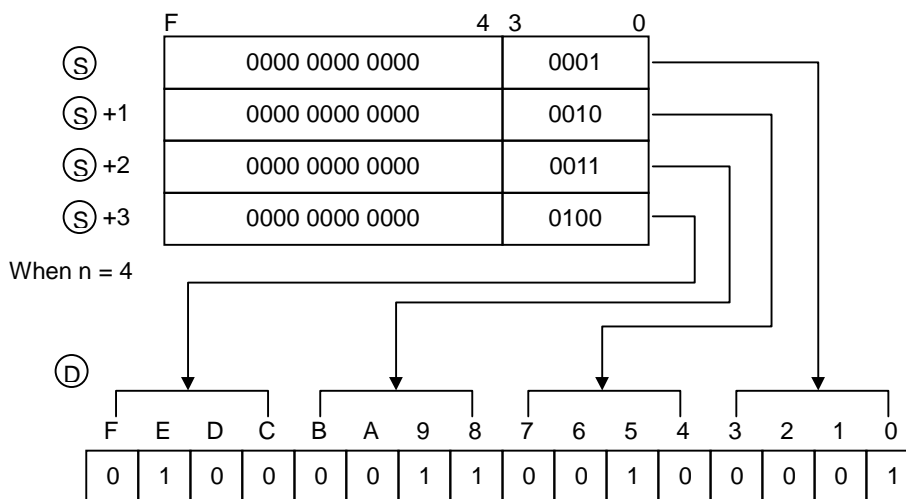
Operand setting

Ⓢ	The start address of source devices.
Ⓓ	The destination device
n	The number of nibble to be associated (1 ~ 4)

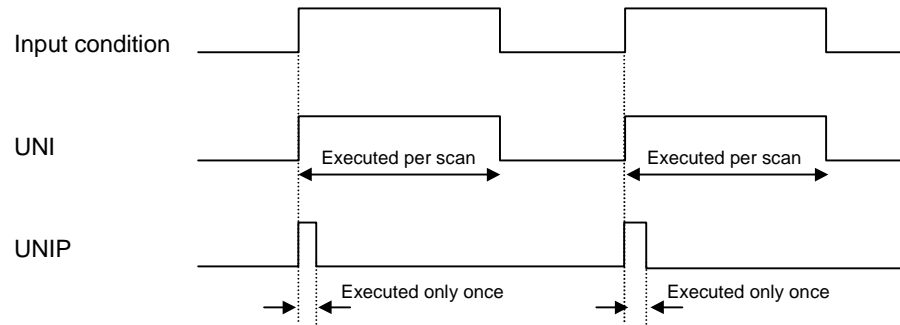
* Available only when do not use computer link module or data link module

1) Functions

- Transfers lowest 4 bits of the block specified as [S+n-1] ~ [S] into the lower n nibbles of the device specified as [D].
- The higher bits (bit 2ⁿ ~ bit F) of the device specified as [D] are cleared as 0.
- When n=0, no processing is performed.
- When n > 4, the error flag is set and no processing is performed.

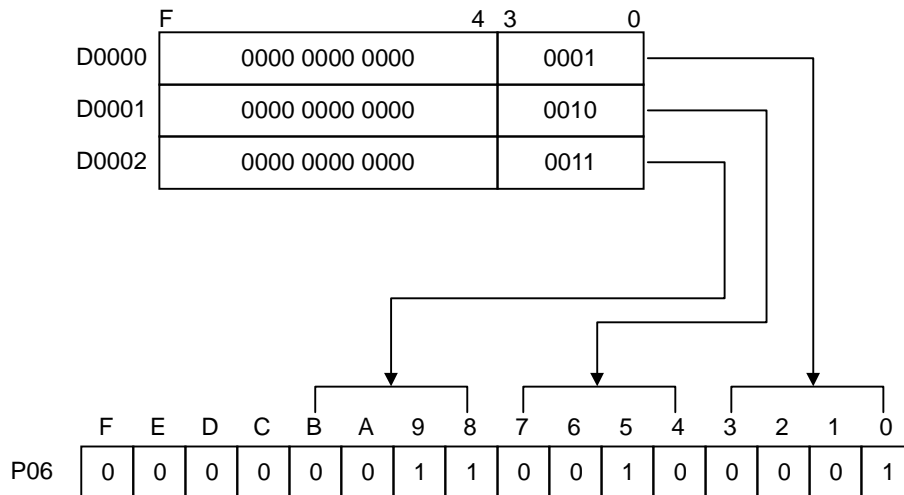


- Execution conditions



2) Program examples

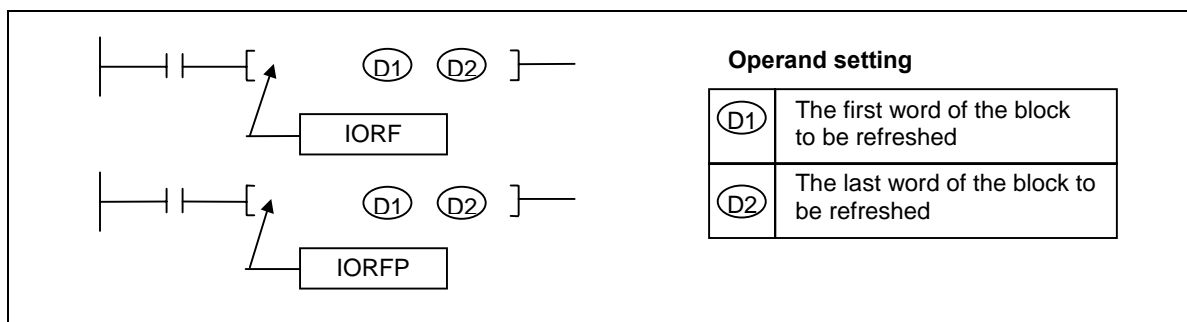
- Program that associate the content of lower 4 bits of D0000 ~ D0003 to lower 3 nibbles of P02 word when P030 is switched on.



5.11.10 IORF, IORFP

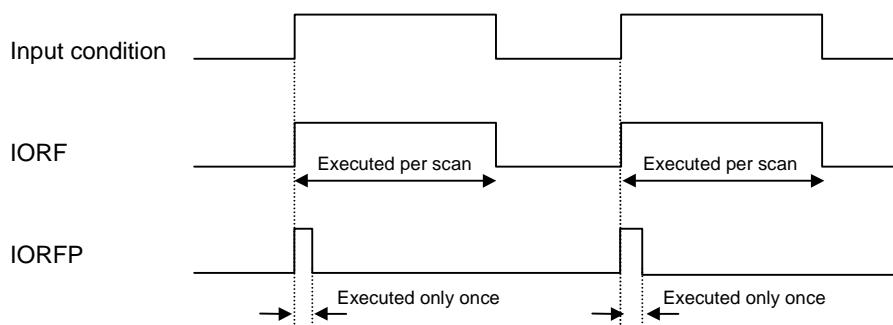
IORF (I/O refresh)	FUN(200) IORF	Applicable CPU	K200S
	FUN(201) IORFP		K300S K1000S

Instructions		Available Device										Steps	Flag		
		M	P	K	L	F	T	C	S	D	#D	Integer	Error (F110)	Zero (F111)	Carry (F112)
IORF(P)	(D1)		O										5	O	
	(D2)		O												



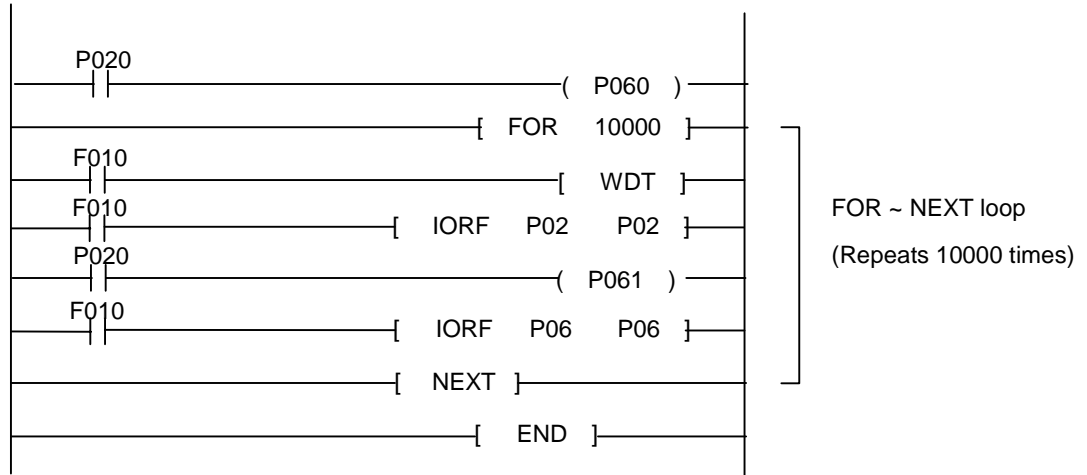
1) Functions

- Refresh I/O data of the block specified as [D1] ~ [D2].
- The [D1] should be lower word than [D2]. If the [D1] is higher than [D2], the error flag is set and no processing is performed.
- This instruction is useful when read latest input data or output the data of P area to external device immediately in a sequence program that has a long scan time.
- Execution conditions



2) Program example

- During FOR ~ NEXT loop execution, the P060 is keep the initial status, but the P061 is switched on/off according to the status change of the P020.




5.12 System instructions

5.12.1 FALS

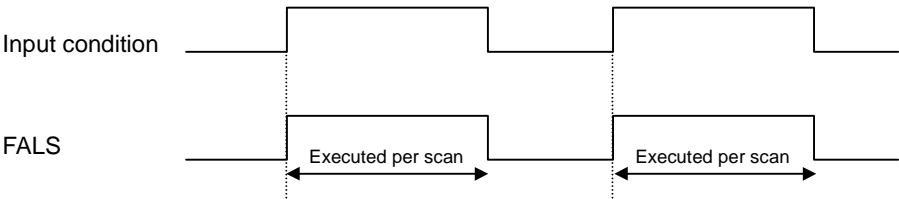
FALS (Error display)	FUN(204) FALS	Applicable CPU	K200S K300S K1000S
-------------------------	---------------	-------------------	--------------------------

Instructions		Available Device										Steps	Flag		
		M	P	K	L	F	T	C	S	D	#D		Integer	Error (F110)	Zero (F111)
FALS	n										O	3			

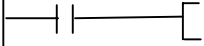
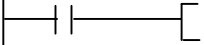
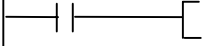
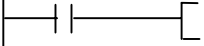
		Operand setting	
n	The error code to be stored at the F area		

1) Functions

- When the input condition is switched on, stores a number specified as 'n' to the F14 word (F140 ~ F14F) and set the FALS flag (F038).
- Once the F14 is set by a FALS instruction, it keeps the value until it is cleared by executing 'FALS 0000' instruction. Even if other FALS instruction is executed, the value of F14 word is not changed.
- Execution condition




2) Program example

P030		FALS	h 1 2 3 4
P031		FALS	h A A A A
P032		FALS	h B B B B
P033		FALS	h 0 0 0 0

5.12.2 DUTY

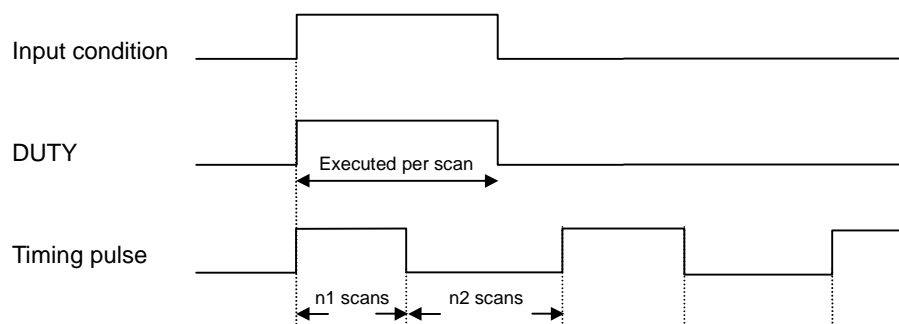
DUTY (User defined pulse)	FUN(205) DUTY	Applicable CPU	All CPUs
------------------------------	---------------	----------------	----------

Instructions		Available Device											Steps	Flag		
		M	P	K	L	F	T	C	S	D	#D	Integer		Error (F110)	Zero (F111)	Carry (F112)
DUTY	Ⓓ					O							7			
	n1											O				
	n2											O				

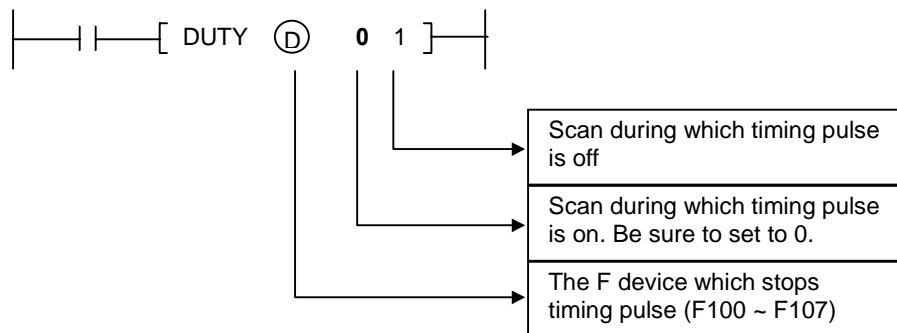
		Operand setting	
Ⓓ	The contact of F device to which a pulse is output	n1	Number of scans during which the pulse is on
n2	Number of scans during which the pulse is off	n2	Number of scans during which the pulse is off

1) Functions

- Generates an user defined timing clock specified as [D] to On at the scan count specified as 'n1' and to OFF at the scan count specified as 'n2'.
- At the initial status (when the timing pulse is off), the timing pulse is off.
- When 'n1' =0, the timing pulse is always off.
- When 'n1' >0 and 'n2' =0, the timing pulse is always on.

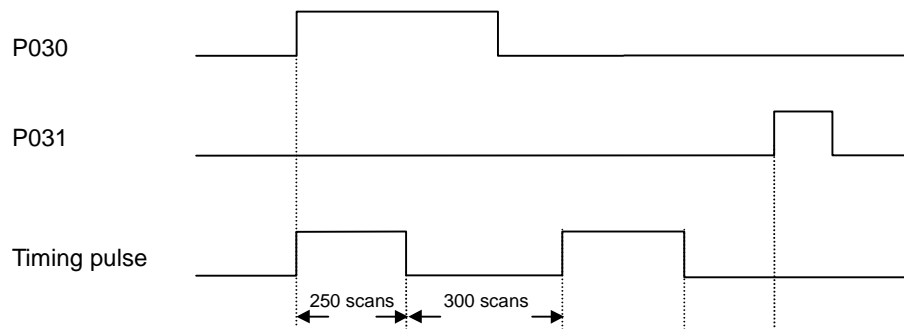
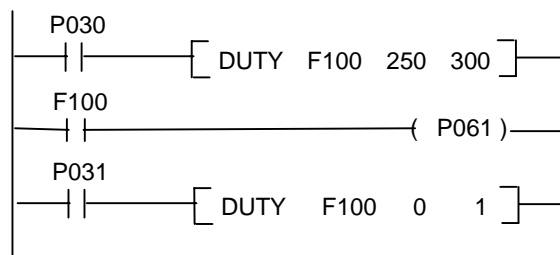


- Even if the timing pulse input turns off, the timing pulse by the DUTY instruction does not turn off. Therefore, to stop the timing pulse, execute the another DUTY instruction as shown below.



2) Program example

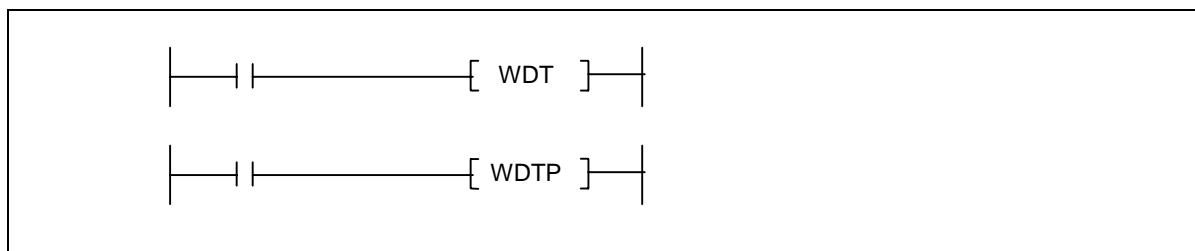
- Program that generates a timing pulse of 250 scans on, and 300 scans off and output it to F100 contact when P030 switched on. When P031 is switched on, the timing pulse is stopped.



5.12.3 WDT, WDTP

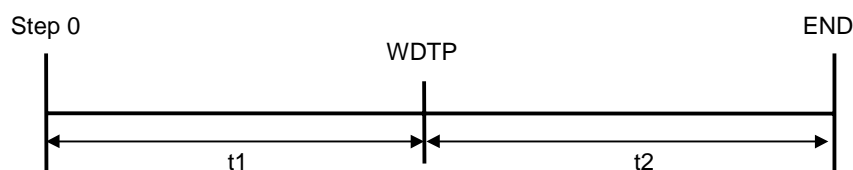
WDT (Watch dog timer clear)	FUN(202) WDT FUN(203) WDTP	Applicable CPU	K200S K300S K1000S
--------------------------------	-------------------------------	----------------	--------------------------

Instructions	Available Device											Steps	Flag		
	M	P	K	L	F	T	C	S	D	#D	Integer		Error (F110)	Zero (F111)	Carry (F112)
WDT WDTP												1			



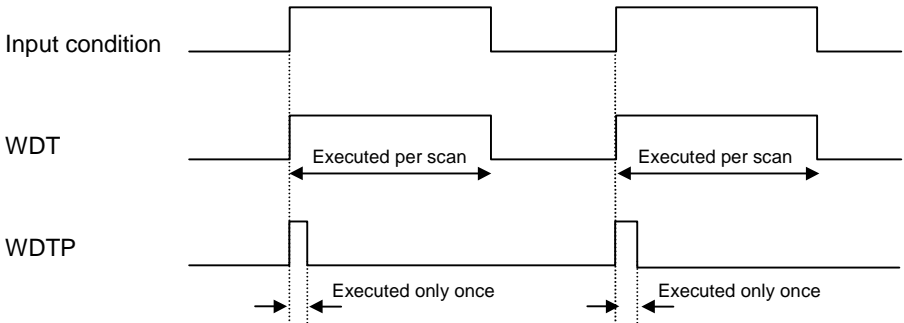
1) Functions

- Resets the watch dog timer in a sequence program
- Used when the period of scan time (from step 0 to END in the sequence program) exceeds the set value of watch dog timer depending on conditions. If the scan time exceeds the set value of watch dog timer at every scan, change the set value of watch dog timer by the parameter setting
- Set the set value of the watch dog timer so that 't1' from step 0 to WDT(P) instruction and 't2' from the WDT(P) to END instruction do not exceed the set value. (See the diagram below)



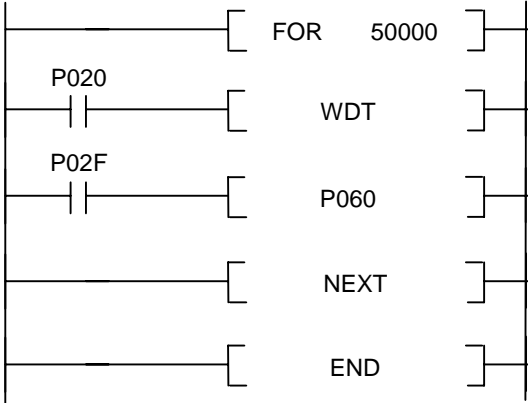
- The WDT(P) instruction can be user two or more times during one scan. However, please be careful when use WDT(P) instruction because if an error occurs, the outputs cannot be turned off immediately.
- Values of scan time stored in special resisters (F device) are not cleared though the WDT(P) instruction is executed.

- Execution conditions



2) Program example

- The program that has a long scan time because of FOR ~ NEXT loop.

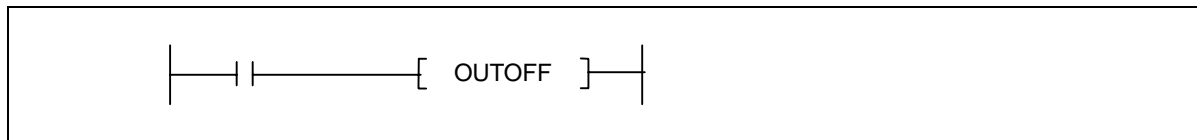


When P020 is on, the current value of WDT is reset and no WDT error occurs. Otherwise, a WDT error occurs and program is stopped.

5.12.4 OUTOFF

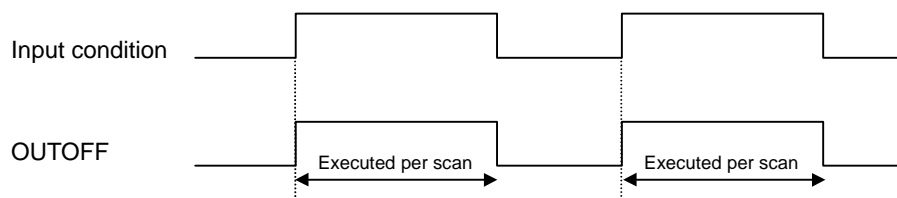
OUTOFF (All output off)	FUN(208) OUTOFF	Applicable CPU	All CPUs
----------------------------	-----------------	-------------------	----------

Instructions	Available Device											Steps	Flag		
	M	P	K	L	F	T	C	S	D	#D	Integer		Error (F110)	Zero (F111)	Carry (F112)
OUTOFF												1			



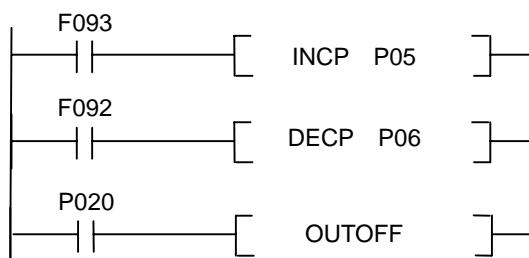
1) Functions

- Stops to output the operation result of P area to the external device and turn on the OUTOFF flag (F113) when an input condition is turns on. However, the P device is updated according to the operation result.
- When the input condition switched off, the CPU restarts to output operation result of P area to external devices.
- Useful for test operation of a PLC system.
- Execution conditions



2) Program example

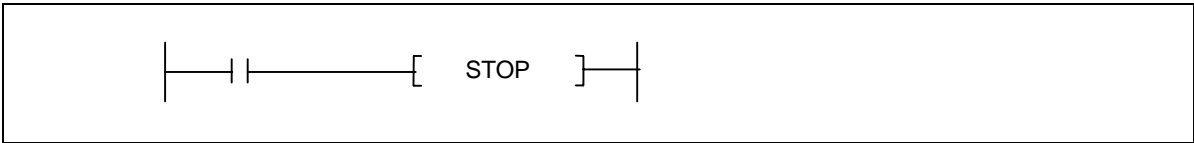
- Program that stop to output to external devices while P020 is on.



5.12.5 STOP

STOP (Stop execution of program)	FUN(008) STOP	Applicable CPU	K200S K300S K1000S
-------------------------------------	---------------	----------------	--------------------------

Instructions	Available Device											Steps	Flag		
	M	P	K	L	F	T	C	S	D	#D	Integer		Error (F110)	Zero (F111)	Carry (F112)
STOP												1			

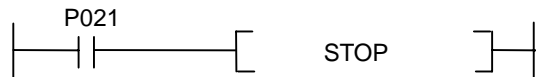


1) Functions

- When an input condition is switched on, stops execution of sequence program and change mode to STOP mode after finishing the current scan.
- To resume the operation of CPU after the execution of STOP instruction, change the mode of CPU to the STOP mode and move it to the RUN mode again by loader or mode switch.

2) Program example

- Program that stops operation when P021 turns on.

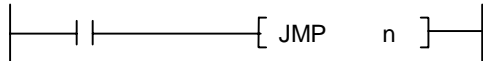
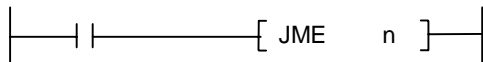


5.13 Branch instructions

5.13.1 JMP, JME

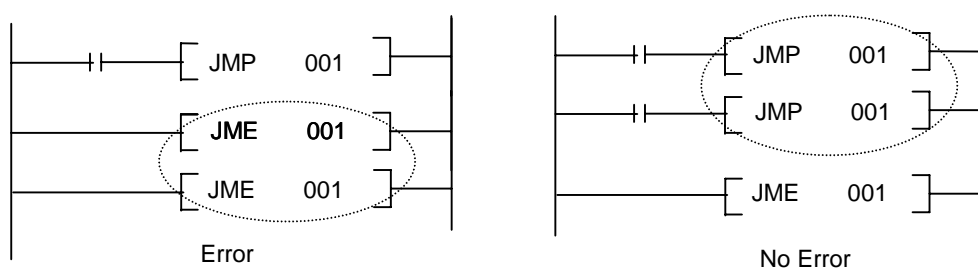
JMP (Jump)	FUN(012) JMP FUN(013) JME	Applicable CPU	All CPUs
---------------	------------------------------	-------------------	----------

Instructions		Available Device										Steps	Flag			
		M	P	K	L	F	T	C	S	D	#D		Integer	Error (F110)	Zero (F111)	Carry (F112)
JMP	n											O	1			
JME																

		Operand setting <table border="1" data-bbox="943 878 1355 1014"><tr><td rowspan="4">n</td><td>K10S1/K10S/K30S/K60S</td></tr><tr><td>: 0 ~ 63</td></tr><tr><td>K200S/K300S/K1000S</td></tr><tr><td>: 0 ~ 127</td></tr></table>		n	K10S1/K10S/K30S/K60S	: 0 ~ 63	K200S/K300S/K1000S	: 0 ~ 127
n	K10S1/K10S/K30S/K60S							
	: 0 ~ 63							
	K200S/K300S/K1000S							
	: 0 ~ 127							
								

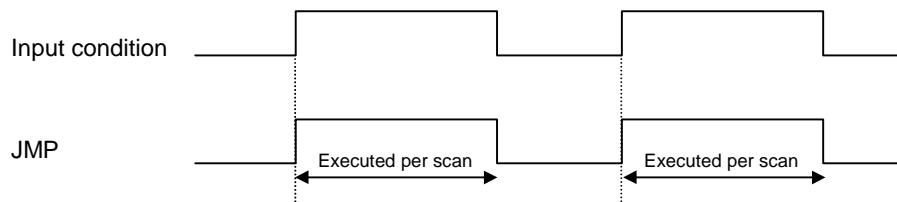
1) Functions

- When the 'JMP n' instruction is executed by turn on of input condition, the CPU jumps to the JME instruction that has same 'n' and instructions between 'JMP n' and 'JME n' are not executed.
- The 'JMP n' instruction should be matched only one 'JME n' instruction. The duplication of 'JME n' is not permitted. However, the duplication of 'JMP n' instruction is possible.



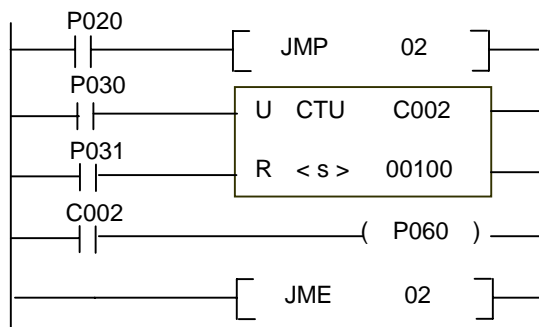
- A 'JMP n' instruction without corresponding 'JME n' instruction (stand-alone 'JMP n') will cause program error. If only a JME or JMP is inside of a loop (subroutine, FOR ~ NEXT, or interrupt routine), an operation error will occur when the JMP instruction is enabled. (Refer 2.7.1 for details)

- Execution conditions



2) Program example

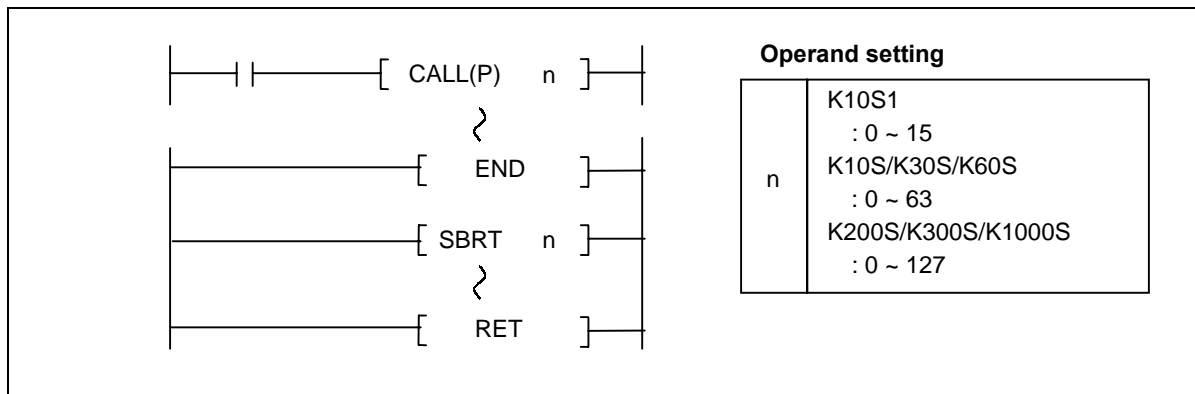
- Program that skips the ring counter operation between 'JMP 2' and 'JME 2' when P020 is on.



5.13.2 CALL, CALLP, SBRT, RET

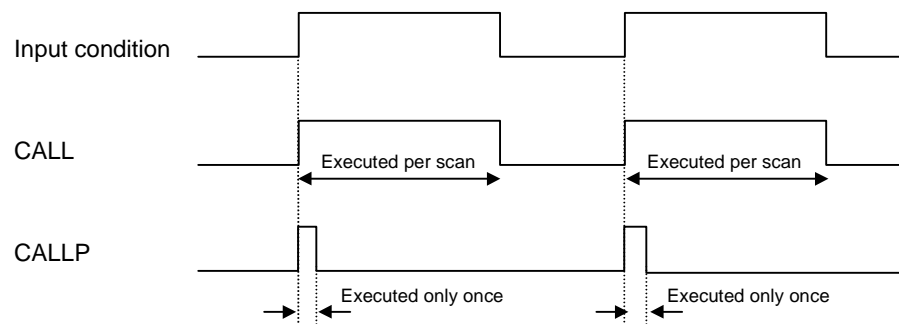
CALL / SBRT (Subroutine)	FUN(014) CALL FUN(015) CALLP FUN(016) SBRT FUN(004) RET	Applicable CPU	All CPUs
-----------------------------	--	-------------------	----------

Instructions		Available Device										Steps	Flag			
		M	P	K	L	F	T	C	S	D	#D		Integer	Error (F110)	Zero (F111)	Carry (F112)
CALL(P) SBRT	n											O	1			
RET																

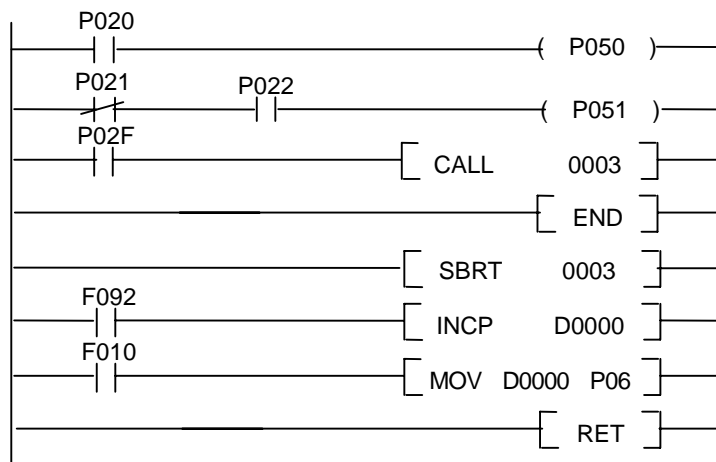


1) Functions

- When the input condition turns on, stops the sequence program and executes the corresponding subroutine program specified by pointer 'n'. After finishing execution of subroutine, resume to execute sequence program from the next step of 'CALL n' instruction.
- Multiple levels of nesting of the CALL(P) instruction are allowed as much as 64.
- The range of pointer 'n' is various according to the type of CPU. (See the above picture)
- If a 'CALL(P) n' instruction has no corresponding 'SBRT' instruction that has same pointer 'n' with the 'CALL(P) n' instruction, an instruction error occurs.
- Execution conditions



2) Program example

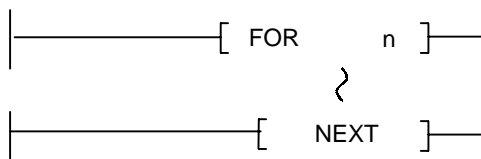


5.14 Loop instructions

5.14.1 FOR, NEXT

FOR / NEXT (Subroutine)	FUN(206) FOR FUN(207) NEXT	Applicable CPU	K200S K300S K1000S
----------------------------	-------------------------------	-------------------	--------------------------

Instructions		Available Device										Steps	Flag			
		M	P	K	L	F	T	C	S	D	#D		Integer	Error (F110)	Zero (F111)	Carry (F112)
FOR	n											O	3			
NEXT													1			

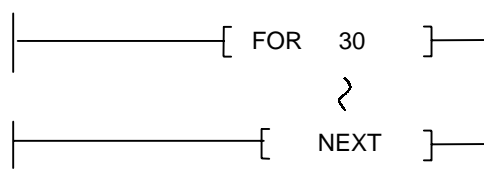
	Operand setting <table><tr><td>n</td><td>Range : 0 ~ 32767</td></tr></table>	n	Range : 0 ~ 32767
n	Range : 0 ~ 32767		

1) Functions

- The FOR instruction is unconditionally executed. The CPU repeats the FOR ~ NEXT block n times and then performs the processing of the next step of the NEXT instruction.
- The range of 'n' is 0 ~ 32767. If the value of 'n' is out of range, an instruction error occurs.
- Up to five levels of the nesting of FOR is allowed.
- In the following cases, operation error occurs;
 - a) After the execution of FOR instruction, the END instruction has been executed before the NEXT instruction is executed.
 - b) The NEXT instruction has been executed before the FOR instruction is executed.
 - c) The number of the FOR instructions is different from that of the NEXT instructions.
 - d) The JMP instruction is executed to exit from the FOR ~ NEXT block or to enter into the FOR ~ NEXT block.

2) Program example

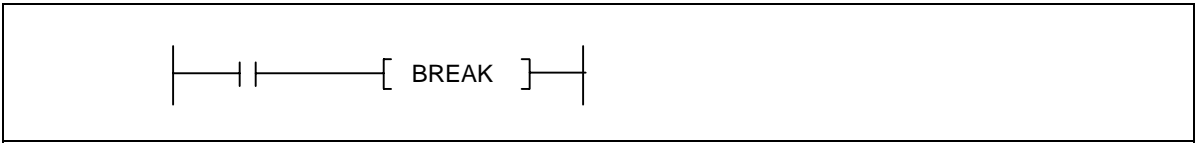
- Program the repeats 30 times the FOR ~ NEXT block.



5.14.2 BREAK

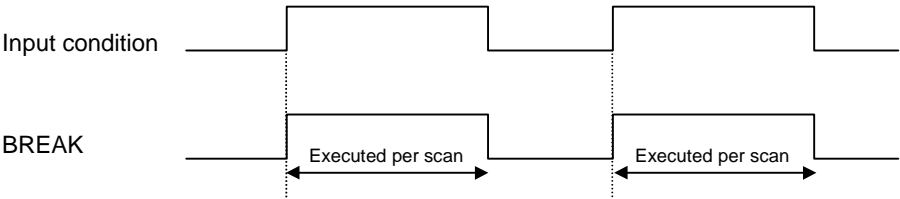
BREAK (Escape from FOR ~ NEXT block)	FUN(220) BREAK	Applicable CPU	K200S K300S K1000S
---	----------------	----------------	--------------------------

Instructions	Available Device											Steps	Flag		
	M	P	K	L	F	T	C	S	D	#D	Integer		Error (F110)	Zero (F111)	Carry (F112)
BREAK												3			

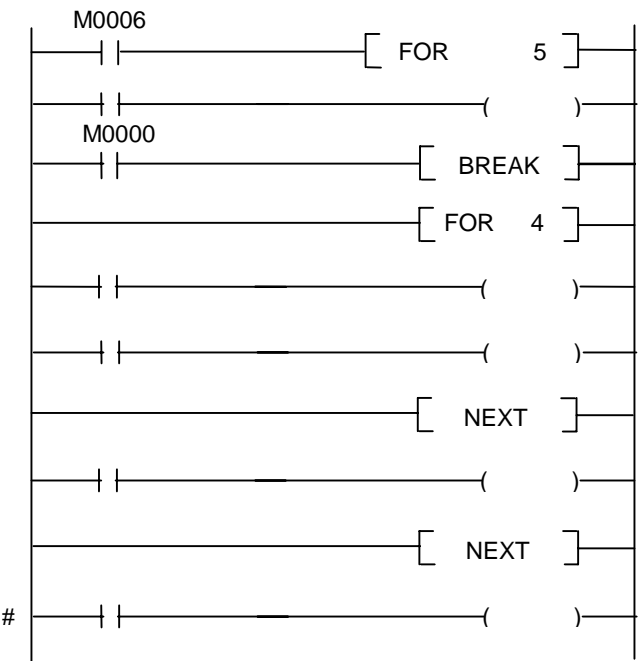


1) Function

- When the input condition is on, exits immediately from the current FOR ~ NEXT loop and go to the next step of the NEXT instruction.
- Execution condition



2) Program example



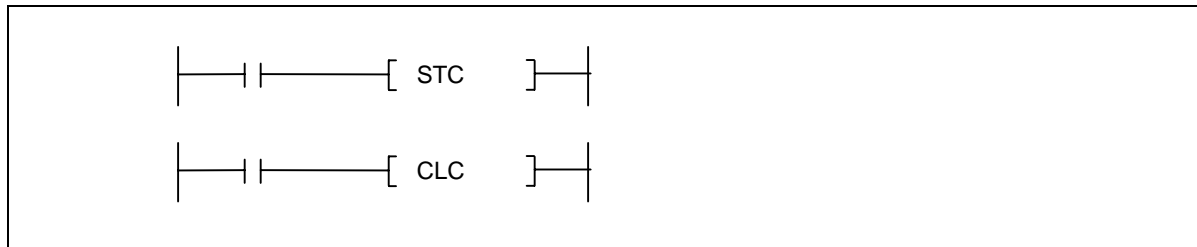
When M000 turns on, exits immediately from FOR ~ NEXT loop and go to the step #.

5.15 Flag instructions

5.15.1 STC, CLC

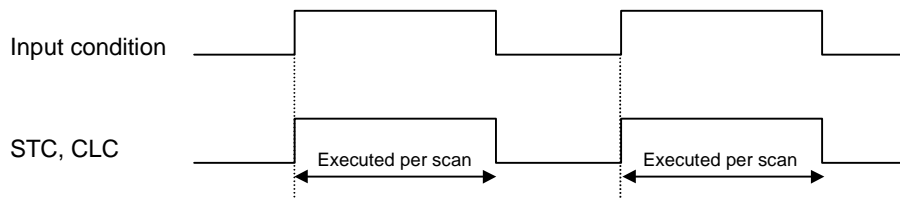
STC, CLC (Set / Reset the carry flag)	FUN(002) STC FUN(003) CLC	Applicable CPU	All CPUs
--	------------------------------	----------------	----------

Instructions	Available Device											Steps	Flag		
	M	P	K	L	F	T	C	S	D	#D	Integer		Error (F110)	Zero (F111)	Carry (F112)
STC CLC												1			O



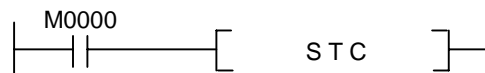
1) Functions

- STC : Turns the carry flag (F112) on when the input condition is switched on.
- CLC : Turns the carry flag (F112) off when the input condition is switched on.
- Execution conditions

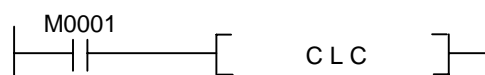


2) Program example

- Program that set the carry flag (F112) when M0000 is on.



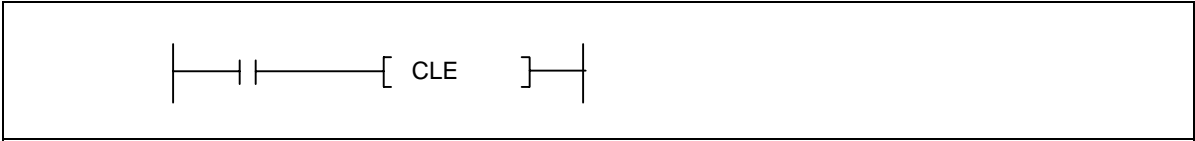
- Program that reset the carry flag (F112) when M0001 is on.



5.15.2 CLE

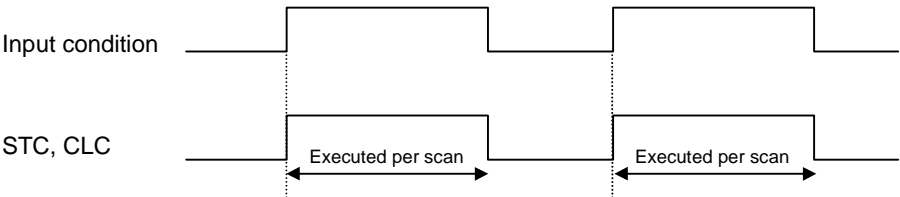
CLE (Reset the latched error flag)	FUN(009) CLE	Applicable CPU	K200S K300S K1000S
---	--------------	----------------	--------------------------

Instructions	Available Device											Steps	Flag		
	M	P	K	L	F	T	C	S	D	#D	Integer		Error (F110)	Zero (F111)	Carry (F112)
CLE												1			



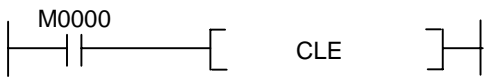
1) Functions

- Reset the latched error flag (F115) when the input condition turns on.
(See the 2.8.2 for detail information of F115 flag)
- Execution condition



2) Program example

- Program that resets the latched error flag (F115) when M0000 turns on.



5.16 Special module instructions

5.16.1 GET, GETP

GET, GETP (Read data from special module)	FUN(230) GET FUN(231) GETP	Applicable CPU	K200S K300S K1000S
--	-------------------------------	----------------	--------------------------

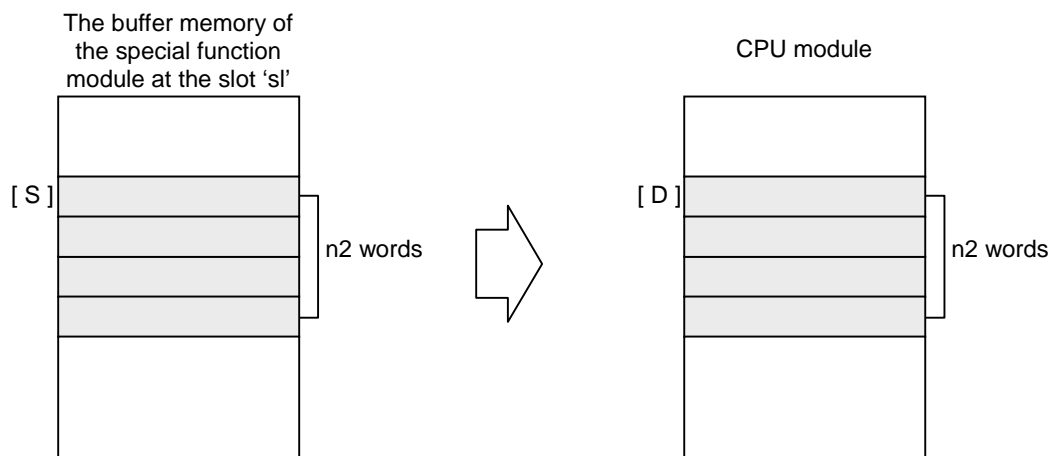
Instructions		Available Device											Steps	Flag		
		M	P	K	L	F	T	C	S	D	#D	Integer		Error (F110)	Zero (F111)	Carry (F112)
GET(P)	sl											O	9	O		
	Ⓢ											O				
	Ⓓ	O	O	O	O*		O	O		O	O					
	n2											O				

		Operand setting	
		sl	Slot number of special function module is mounted
		Ⓢ	Start address of data to be read
		Ⓓ	Start address of data to be stored
		n2	Number of word to be read

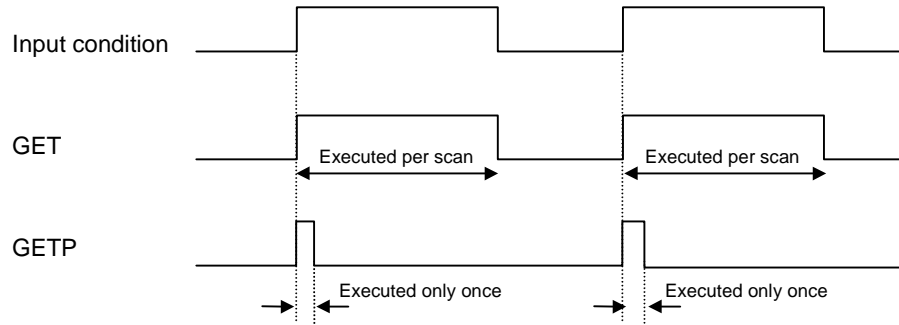
* Available only when do not use computer link module or data link module

1) Functions

- Reads the data of 'n2' words, which start at the address specified as [S] of buffer memory inside the special module specified at 'sl', and stores the data into devices which begin with the device specified at [D].

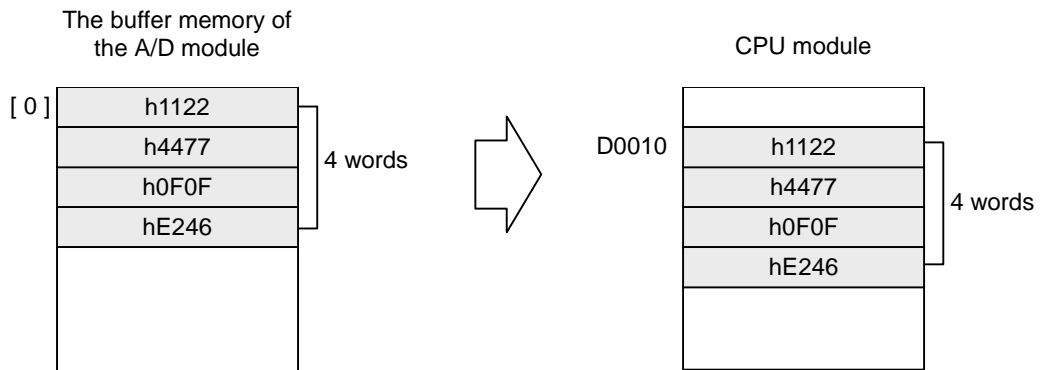
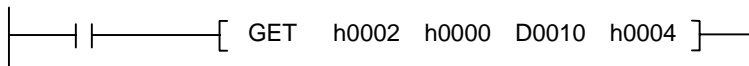
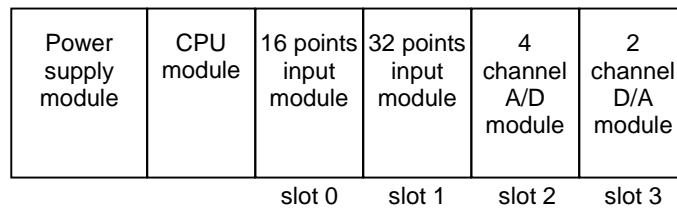


- In the following cases, operation error occurs;
 - a) The slot number specified as 'sl' is not a special function module
 - b) The value of 'n2' is over 512, or [D+n2] exceeds the specified device range.
- Execution conditions



2) Program example

- Program that reads 4 words from the address 0 of buffer memory of A/D module, and stores them to the 4 words from D0010 of CPU module.



5.16.2 PUT, PUTP

PUT, PUTP (Write data to special function module)	FUN(234) PUT FUN(235) PUTP	Applicable CPU	K200S K300S K1000S
--	-------------------------------	----------------	--------------------------

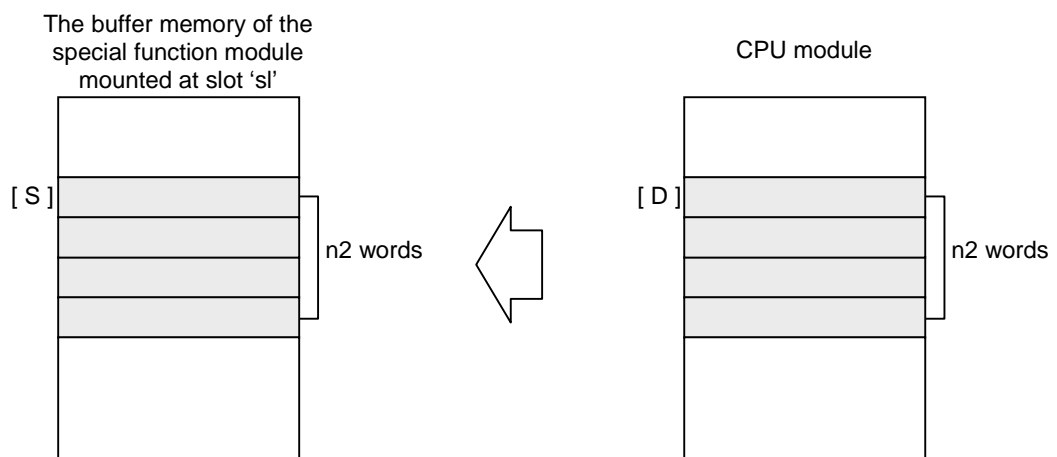
Instructions		Available Device										Steps	Flag		
		M	P	K	L	F	T	C	S	D	#D	Integer	Error (F110)	Zero (F111)	Carry (F112)
PUT(P)	sl											O	9	O	
	Ⓓ											O			
	Ⓔ	O	O	O	O*		O	O		O	O				
	n2											O			

		Operand setting	
		n1	Slot number of special function module is mounted
		Ⓓ	Start address of data to be stored
		Ⓔ	Start address of source data
		n2	Numbers of word to be written

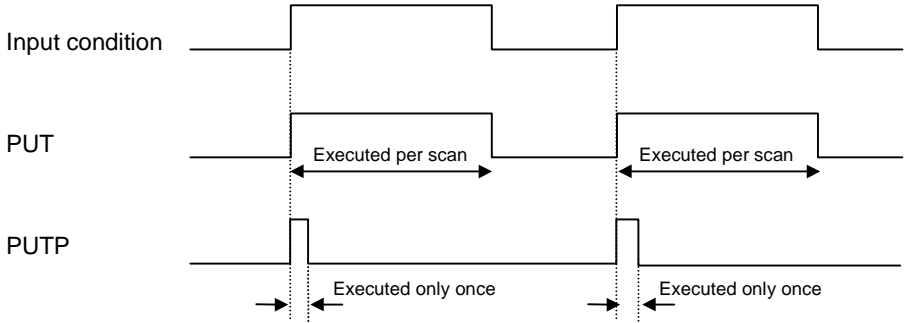
* Available only when do not use computer link module or data link module

1) Functions

- Writes the data of 'n2' words, which start at the address specified as [S] of CPU, and transfer the data into the block starting at the address specified as [S] of buffer memory inside the special function module mounted at the slot specified as 'sl'

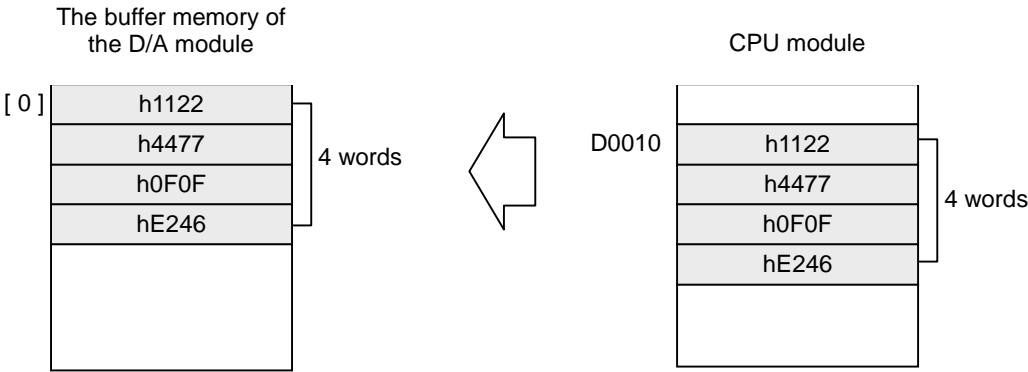
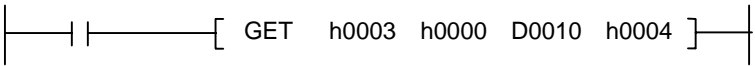
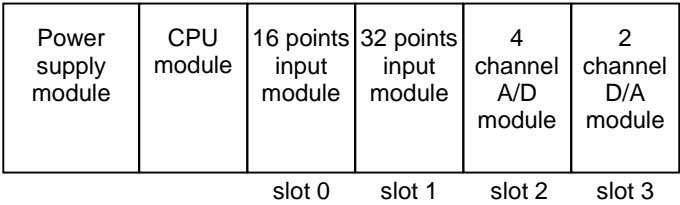


- In the following cases, operation error occurs;
 - a) The slot number specified as 'sl' is not a special function module
 - b) The value of 'n2' is over 512, or [D+n2] is exceeds the specified device range.
- Execution conditions



2) Program example

- Program that write 4 words from D0010 of CPU module, and stores them to the 4 words from the address 0 of buffer memory of D/A module.



5.17 Data link instructions

5.17.1 READ

READ (Read data from remote station)	FUN(244) READ	Applicable CPU	K200S K300S K1000S
--	---------------	----------------	--------------------------

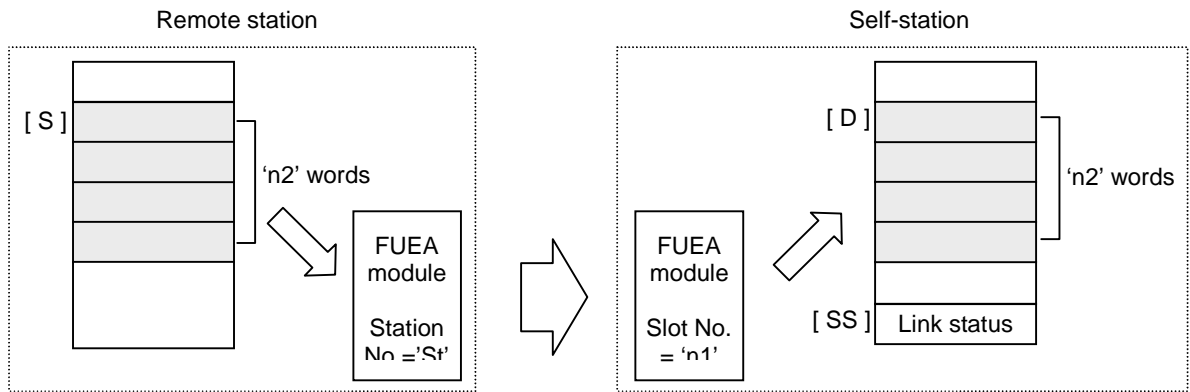
Instructions		Available Device											Steps	Flag		
		M	P	K	L	F	T	C	S	D	#D	Integer		Error (F110)	Zero (F111)	Carry (F112)
READ	sl											O	13	O		
	St	O	O	O	O	O	O	O		O	O	O				
	Ⓓ	O	O	O	O*		O	O		O	O					
	Ⓔ	O	O	O	O	O	O	O		O	O					
	n2									O		O				
	SS	O	O	O	O*		O	O		O	O					

Operand setting	
sl	Slot number of FUEA module is mounted
St	Station number of remote station to be read data
Ⓓ	Start address of master station at which read data is stored
Ⓔ	Start address of remote station at which data to be read
n2	Numbers of word to be read
SS	Device at which the link status is stored

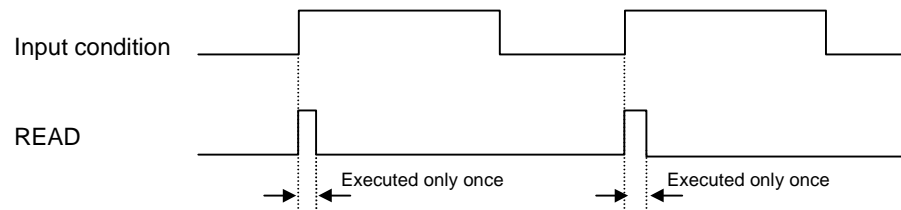
* Available only when do not use computer link module or data link module

1) Functions

- Reads 'n2' words which begin with the address [S] of the remote station that has station number 'St' through the FUEA module mounted at the slot 'sl', and store the read data to the block which begin with the address [D] of the master station. The link status is stored at the address 'SS' of the master station.

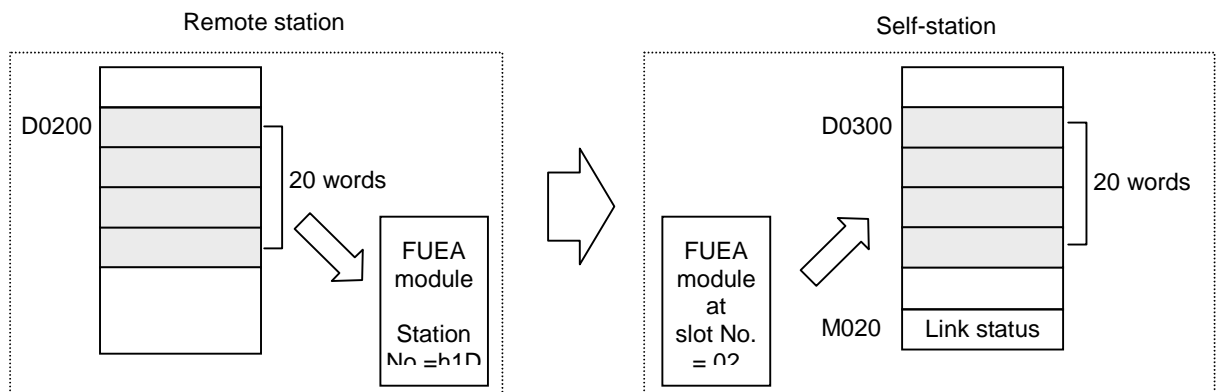
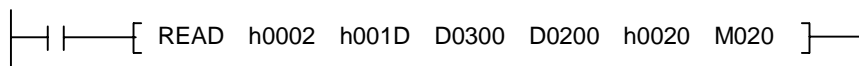


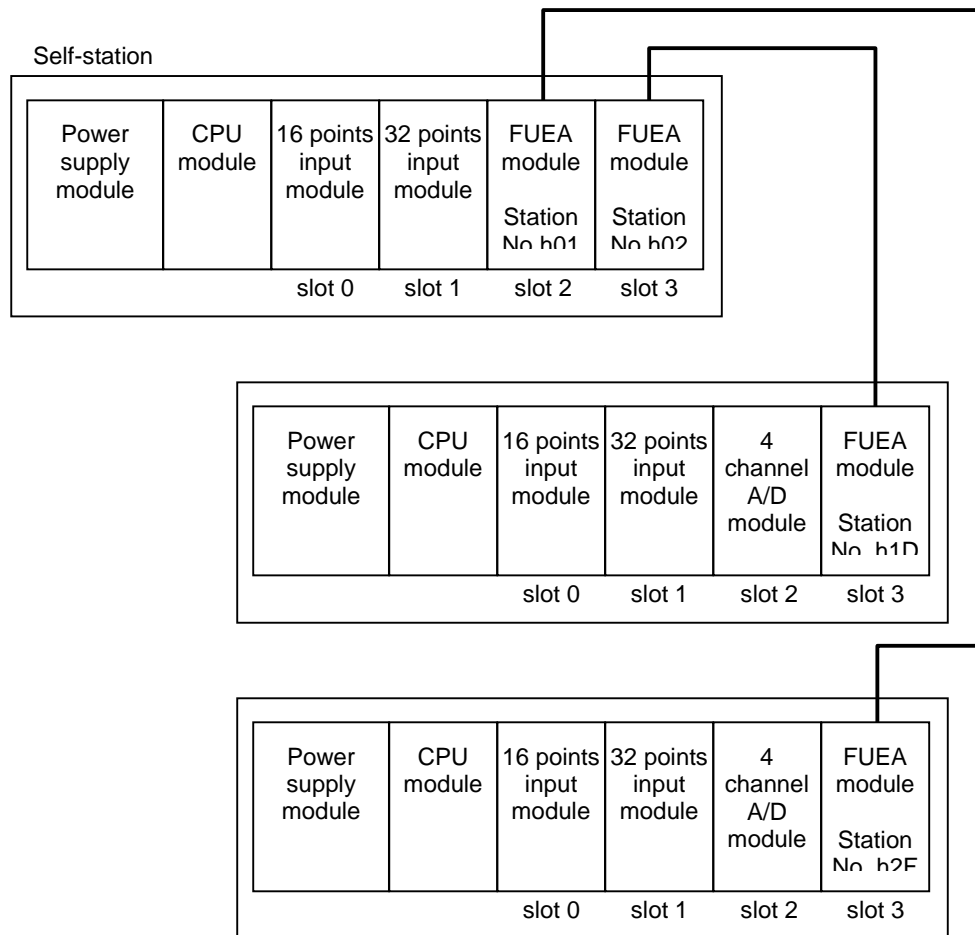
- An instruction error occurs when the address [S+n2] or [D+n2] is out of the range of specified device.
- Execution conditions



2) Program example

- Program that read 20 words which begin with D0200 of remote station (Station No. of FUEA module = h1D) through the FUEA module of slot number 2, and store the read data to the block which begin with D0300. The link status is stored at the M020 word.

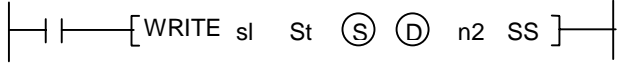


System configuration

5.17.2 WRITE

WRITE (Write data to remote station)	FUN(245) WRITE	Applicable CPU	K200S K300S K1000S
---	----------------	----------------	--------------------------

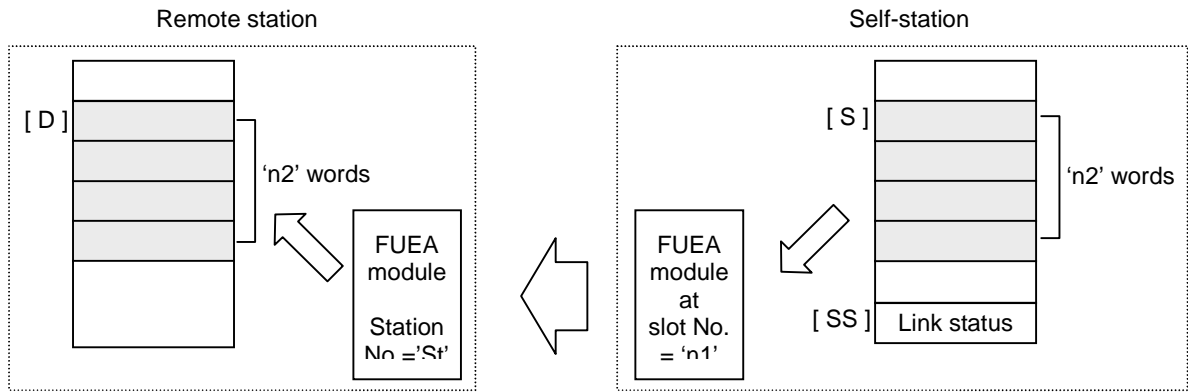
Instructions		Available Device											Steps	Flag		
		M	P	K	L	F	T	C	S	D	#D	Integer		Error (F110)	Zero (F111)	Carry (F112)
WRITE	sl											O	13	O		
	St	O	O	O	O	O	O	O		O	O	O				
	Ⓢ	O	O	O	O*		O	O		O	O					
	Ⓓ	O	O	O	O	O	O	O		O	O					
	n2									O		O				
	SS	O	O	O	O*		O	O		O	O					

		Operand setting	
		sl	Slot number of FUEA module is mounted
		St	Station number of remote station to be read data
		Ⓢ	Start address of master station at which read data is stored
		Ⓓ	Start address of remote station at which data to be read
		n2	Numbers of word to be read
		SS	Device at which the link status is stored

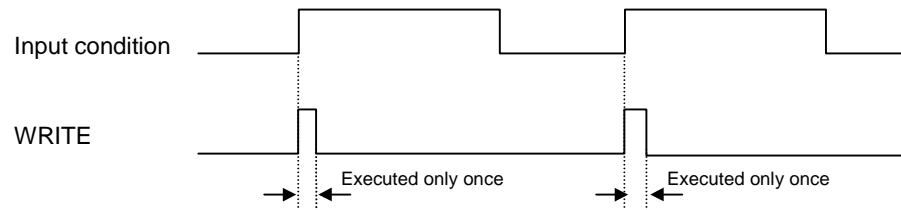
* Available only when do not use computer link module or data link module

1) Functions

- Reads 'n2' words which begin with the address [S], and write the read data to the block which begin with the address [D] of the remote station that has a station number specified as 'St' through the FUEA module mounted at the slot 'sl' of self station. The link status is stored at the address 'SS' of the master station.



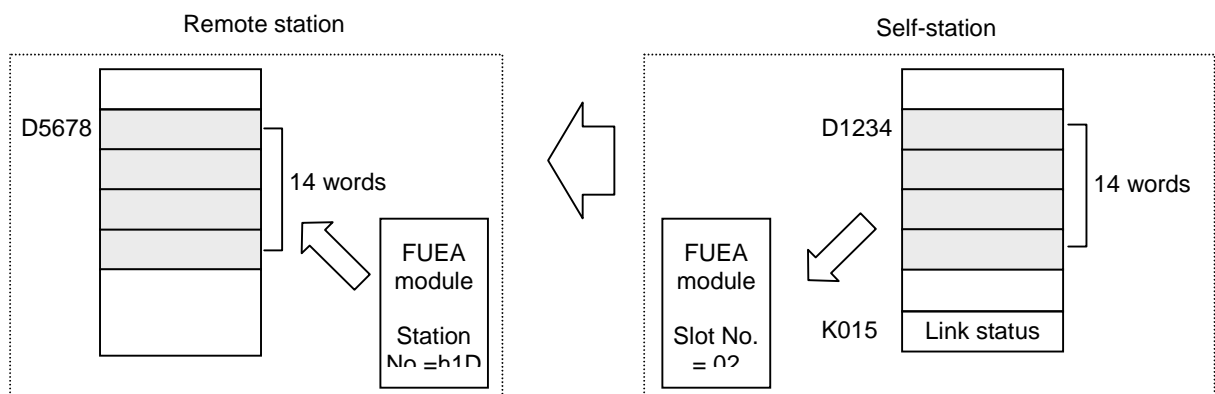
- An instruction error occurs when the address [S+n2] or [D+n2] is out of the range of specified device.
- Execution conditions



2) Program example

- Program that read 14 words which begin with D1234 of self station, and write the read data to the block which begin with D5678 of remote station (Station No. of FUEA module = h2F) through the FUEA module at slot number 3 of self station. The link status is stored at the K015 word.
(System configuration is same as the example of READ instruction.)

┌───┐ ┌───┐ [WRITE h0003 h002F D1234 D5678 h0014 K015] ───┐



5.17.3 RGET

RGET (Read data from special function module of remote station)	FUN(232) RGET	Applicable CPU	K200S K300S K1000S
--	---------------	----------------	--------------------------

Instructions		Available Device											Steps	Flag		
		M	P	K	L	F	T	C	S	D	#D	Integer		Error (F110)	Zero (F111)	Carry (F112)
RGET	sl											O	13	O		
	St											O				
	Ⓓ	O	O	O	O*		O	O		O	O					
	Ⓔ											O				
	n2									O		O				
	SS	O	O	O	O*		O	O		O	O					

The diagram shows the RGET instruction format enclosed in brackets. The components are: RGET, a space, 'sl', a space, 'St', a space, a circled 'D', a space, a circled 'S', a space, 'n2', a space, and 'SS'.

The configuration of 'sl'

AB	CD	Lower 8 bits (CD) : Slot No. of FUEA Higher 8 bits (AB) : Type of special function module
----	----	--

The ID code of special function module

	AD	DAI	DAV	TC	RTD
K1000S	00h	01h	02h	03h	04h
K300S	80h	81h	82h	83h	84h
K200S	80h	81h	82h	—	—

The configuration of 'St'

AB	CD	Lower 8 bits (CD) : Station No. of remote station Higher 8 bits (AB) : Slot No. of special function module
----	----	---

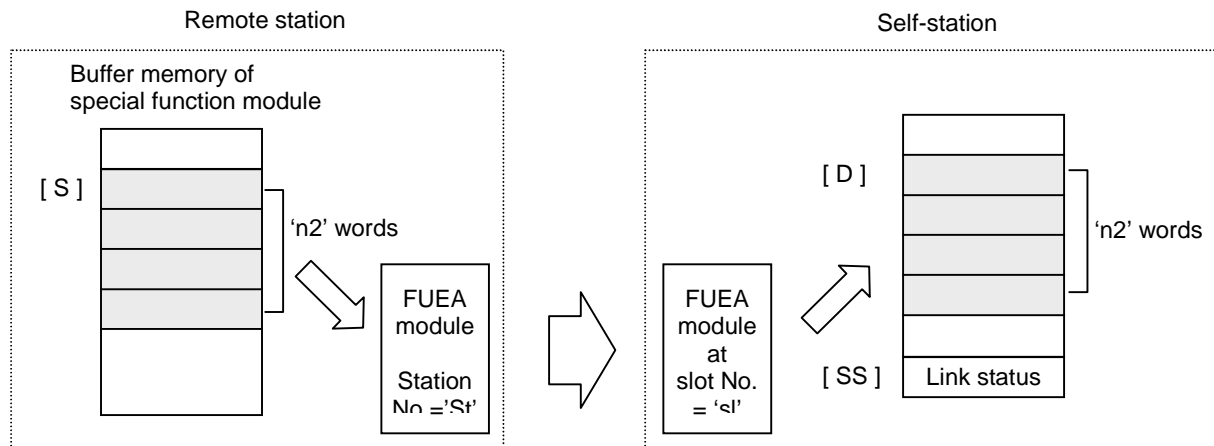
Operand setting

sl	Slot number of FUEA module is mounted & Type of special function module to be read
St	Station number of remote station to be read data & Slot number of special function module
(D)	Start address of self station at which read data is stored
(S)	Start address of remote station at which data to be read
n2	Numbers of word to be read
SS	Device at which the link status is stored

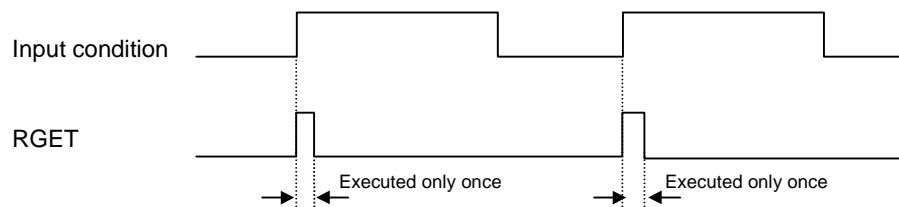
* Available only when do not use computer link module or data link module

1) Functions

- Reads the data of 'n2' words, which start at the address specified as [S] of buffer memory inside the special function module of remote station (station number & slot number is specified as 'St') through the FUEA module, and stores the data into devices which begin with the device specified as [D]. Then, stores the link status into the device specified as [SS] of self station.

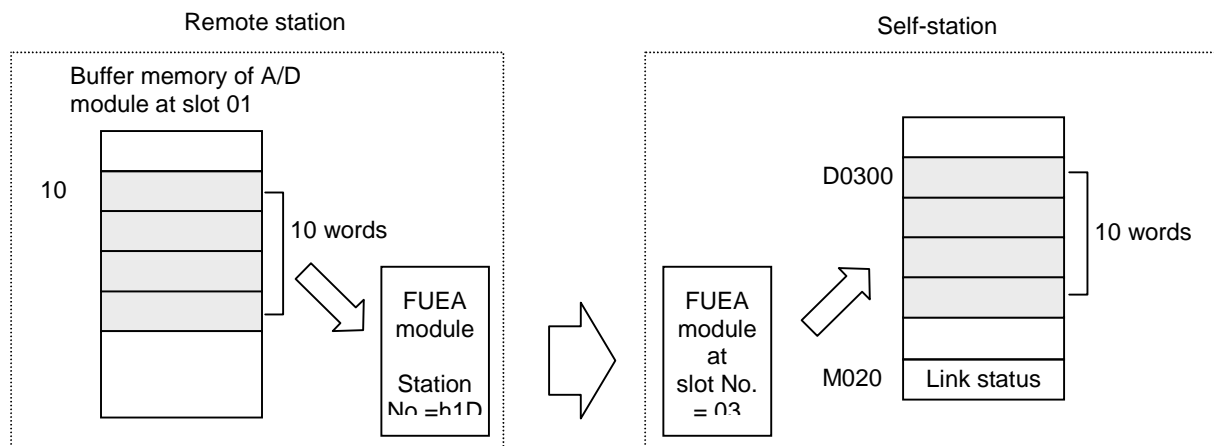


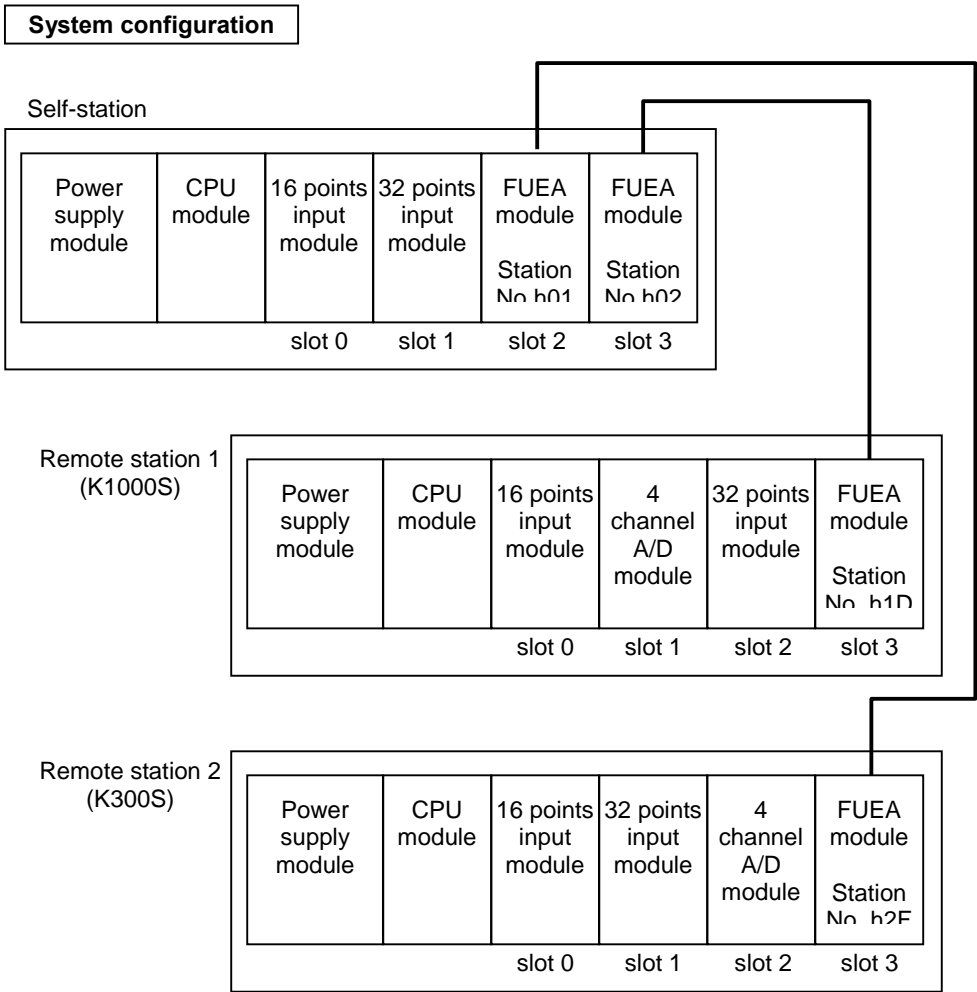
- An instruction error occurs when the address $[S+n2]$ or $[D+n2]$ is out of the range of specified device.
- Execution conditions



2) Program example

- Program that reads 10 words, through the FUEA module mounted at the slot 03, from the address 10 of the buffer memory of the K1000S A/D module mounted at the slot 01 of remote station h1D, and stores the read data to the 10 words which begin with D0300 of self station. The link status is stored at M020 word of self station.





5.17.4 RPUT

RPUT (Write data to special function module of remote station)	FUN(233) RPUT	Applicable CPU	K200S K300S K1000S
---	---------------	----------------	--------------------------

Instructions		Available Device											Steps	Flag		
		M	P	K	L	F	T	C	S	D	#D	Integer		Error (F110)	Zero (F111)	Carry (F112)
RPUT	sl											O	13	O		
	St											O				
	Ⓢ											O				
	Ⓓ	O	O	O	O*		O	O		O	O					
	n2									O		O				
	SS	O	O	O	O*		O	O		O	O					

The diagram shows the RPUT instruction format in square brackets. The fields are: RPUT, sl, St, (S), (D), n2, and SS. The (S) and (D) fields are enclosed in circles.

The configuration of 'sl'

AB	CD	Lower 8 bits (CD) : Slot No. of FUEA Higher 8 bits (AB) : Type of special function module
----	----	--

The ID code of special function module

	AD	DAI	DAV	TC	RTD
K1000S	00h	01h	02h	03h	04h
K300S	80h	81h	82h	83h	84h
K200S	80h	81h	82h	—	—

The configuration of 'St'

AB	CD	Lower 8 bits (CD) : Station No. of remote station Higher 8 bits (AB) : Slot No. of special function module
----	----	---

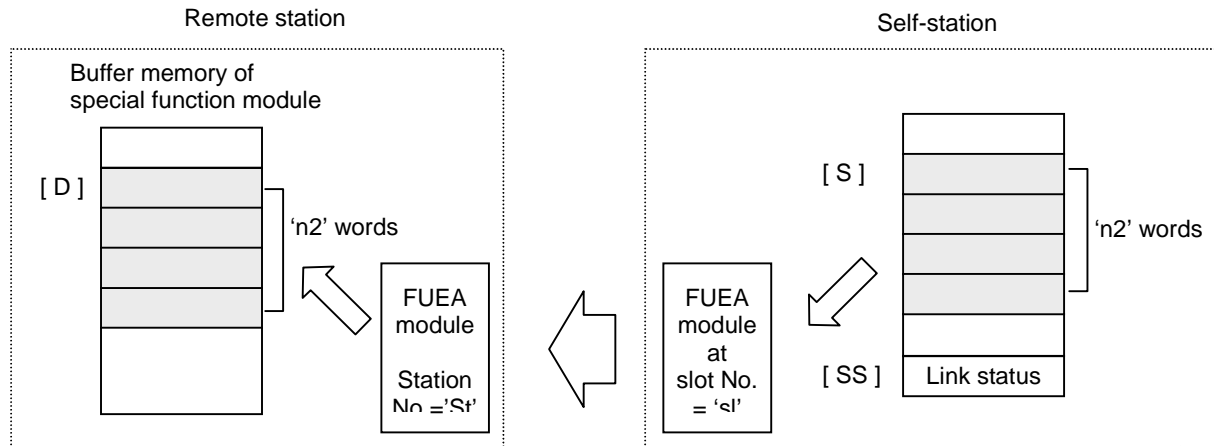
Operand setting

sl	Slot number of FUEA module is mounted & Type of special function module to be read
St	Station number of remote station to be read data & Slot number of special function module
(S)	Start address of self station at which source data is stored
(D)	Start address of remote station at which data to be written
n2	Numbers of word to be written
SS	Device at which the link status is stored

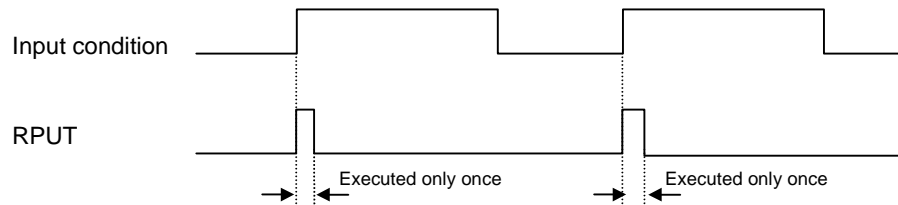
* Available only when do not use computer link module or data link module

1) Functions

- Reads the data of 'n2' words, which start at the device specified as [D], and write the data to the block which begin with the address specified as [S] of buffer memory inside the special function module of remote station (station number & slot number is specified as 'St') through the FUEA module. Then, stores the link status into the device specified as [SS] of self station.



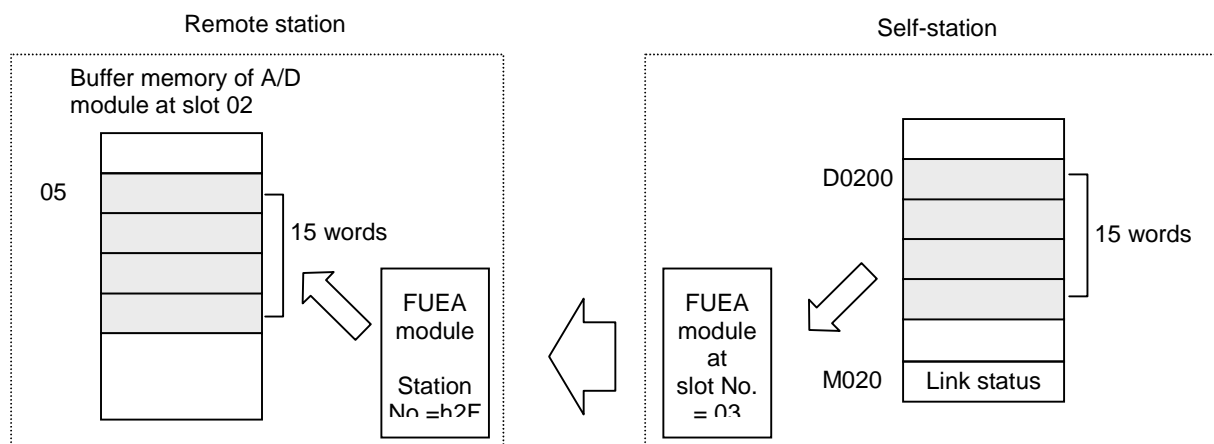
- An instruction error occurs when the address **[S+n2]** or **[D+n2]** is out of the range of specified device.
- Execution conditions



2) Program example

- Program that reads 15 words from **D0200** of self station, then write the data to the 15 words block which begin with the address **5** of buffer memory of the **K300S A/D** module mounted at the slot **02** of remote station **h2F** through the **FUEA** module at the slot **02**. The link status is stored at **M020** word of self station.

┌───┐ ┌───┐ [RPUT h8002 h022F D0200 h0005 h0015 M020] ───┐



(System configuration is same as that of the example of RGET instruction – p115)

5.17.5 STATUS

STATUS (Read the link information of FUEA module of remote station)	FUN(247) STATUS	Applicable CPU	K200S K300S K1000S
--	-----------------	----------------	--------------------------

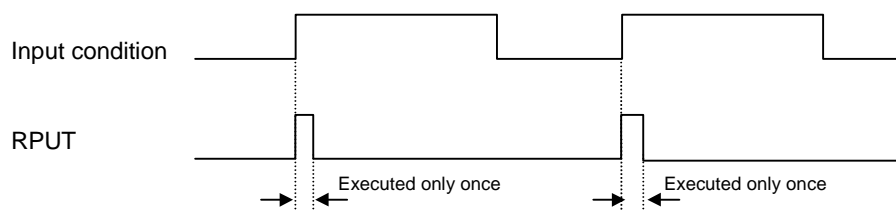
Instructions		Available Device											Steps	Flag		
		M	P	K	L	F	T	C	S	D	#D	Integer		Error (F110)	Zero (F111)	Carry (F112)
STATUS	sl											O	9	O		
	St	O	O	O	O	O	O	O		O	O					
	Ⓓ	O	O	O	O*		O	O		O	O					
	SS	O	O	O	O*		O	O		O	O					

		Operand setting	
		sl	Slot number of FUEA module is mounted & Type of special function module to be read
		St	Station number of remote station to be read data & Slot number of special function module
		Ⓓ	Start address of remote station at which data to be written
		SS	Device at which the link status is stored

* Available only when do not use computer link module or data link module

1) Functions

- Read the link information (10 words) from the remote station of which station number is 'St' through the FUEA module mounted at the slot number 'sl', and stores the information data to the block begin with the device specified as [D]. The link status is stored into the device [SS].
- Execution conditions



2) Program example

- Program that reads the information of remote station (station number is stored at D0000) through the FUEA module of slot 07, and stores the data from D1234. The link status is stored into K015 word.

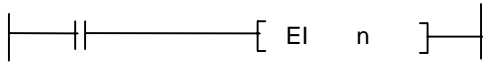
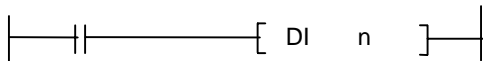


5.18 Interrupt instructions

5.18.1 EI, DI

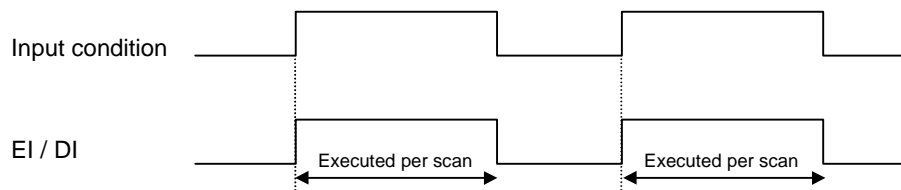
EI / DI (Enable / Disable interrupt)	FUN(238) EI FUN(239) DI	Applicable CPU	K200S K300S K1000S
---	----------------------------	-------------------	--------------------------

Instructions		Available Device										Steps	Flag			
		M	P	K	L	F	T	C	S	D	#D		Integer	Error (F110)	Zero (F111)	Carry (F112)
EI	n											O	1			
DI																

	Operand setting <table><tr><td rowspan="3">n</td><td>0 ~ 7 (K200S)</td></tr><tr><td>0 ~ 13 (K300S)</td></tr><tr><td>0 ~ 29 (K1000S)</td></tr></table>	n	0 ~ 7 (K200S)	0 ~ 13 (K300S)	0 ~ 29 (K1000S)
n			0 ~ 7 (K200S)		
	0 ~ 13 (K300S)				
	0 ~ 29 (K1000S)				
					

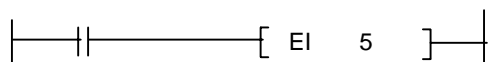
1) Functions

- EI : Enables the interrupt (TDI or PDI) specified as 'n' when input condition turns on. If 'n' is not specified, all interrupts are enabled.
- DI : Disables the interrupt (TDI or PDI) specified as 'n' when input condition turns on. If 'n' is not specified, all interrupts are disabled.
- The 'n' is assigned to each interrupt by parameter setting.
- Execution conditions

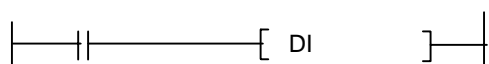


2) Program example

- Program that enable the interrupt 5



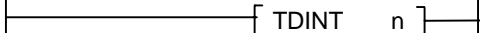
- Program that disable all interrupts



5.18.2 TDINT, IRET

TDINT / IRET (Time driven interrupt)	FUN(226) TDINT FUN(225) IRET	Applicable CPU	K200S K300S K1000S
---	---------------------------------	-------------------	--------------------------

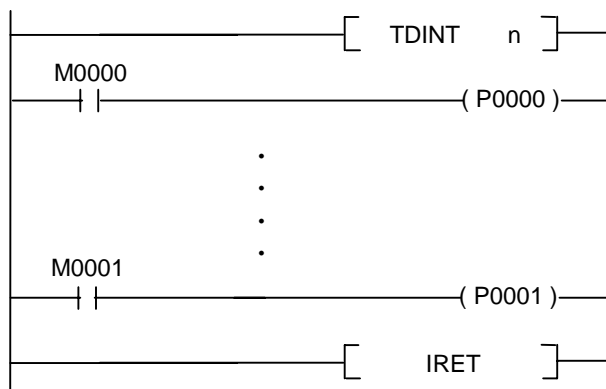
Instructions		Available Device										Steps	Flag			
		M	P	K	L	F	T	C	S	D	#D		Integer	Error (F110)	Zero (F111)	Carry (F112)
TDINT	n											O	1			

		Operand setting	
n	0 ~ 5	(K200S)	
	0 ~ 7	(K300S)	
	0 ~ 15	(K1000S)	

1) Functions

- TDINT : Indicates the begin of the time driven interrupt routine.
- IRET : Indicates the end of the interrupt routine.
- Instructions between the TDINT 'n' and the next IRET are executed only when the corresponding time driven interrupt occurs and is enabled by the EI instruction.
- The interval of interrupt can be set as 60msec ~ 60000msec (unit : 10msec) with parameter setting.
- The execution time of interrupt routine should be less than the interval of interrupt.
- The TDINT n instruction has to be placed after the END instruction.
- The TDINT and IRET instructions are executed unconditionally.

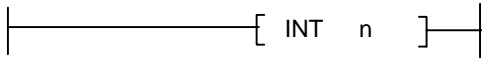
2) Program example



5.18.3 INT, IRET

INT / IRET (Process driven interrupt)	FUN(227) INT FUN(225) IRET	Applicable CPU	K200S K300S K1000S
--	-------------------------------	----------------	--------------------------

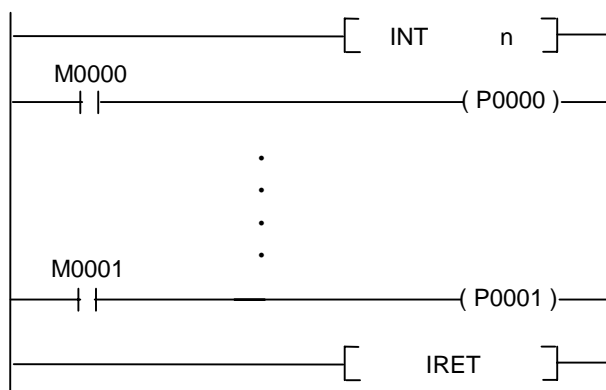
Instructions		Available Device										Steps	Flag		
		M	P	K	L	F	T	C	S	D	#D Integer		Error (F110)	Zero (F111)	Carry (F112)
INT	n										O	1			

		Operand setting	
n		0 ~ 5	(K200S)
		0 ~ 7	(K300S)
		0 ~ 15	(K1000S)

1) Functions

- INT : Indicates the begin of the process driven interrupt routine.
- IRET : Indicates the end of the interrupt routine.
- Instructions between the INT 'n' and the next IRET are executed only when the corresponding time driven interrupt occurs and is enabled by the EI instruction.
- To use process driven interrupts, the interrupt module is required and general input module can not be used for interrupt input. However, K200S can use general input module for interrupt input by parameter setting. (Refer 2.4 'Parameter setting' for details)
- The INT n instruction has to be placed after the END instruction.
- The INT and IRET instructions are executed unconditionally.

2) Program example



5.19 Sign inversion instruction

5.19.1 NEG, NEGP, DNEG, DNEGP

NEG (Sign inverse)	FUN(240) NEG FUN(242) DNEG FUN(241) NEGP FUN(243) DNEGP	Applicable CPU	K200S K300S K1000S
-----------------------	--	-------------------	--------------------------

Instructions		Available Device											Steps	Flag		
		M	P	K	L	F	T	C	S	D	#D	Integer		Error (F110)	Zero (F111)	Carry (F112)
NEG(P)	Ⓓ	O	O	O	O*		O	O		O	O		1	O		
DNEG(P)																

Operand setting

Ⓓ The device which stores the data to be inverted.

* Available only when do not use computer link module or data link module

1) Functions

- NEG(P) : Reverses the sign of the 16 bits data of device specified as [D] and stores the result in the device specified as [D].
- DNEG(P) : Reversees the sign of the 32 bits data of device specified as [D+1, D] and stores the result in the device specified as [D+1, D].
- Used to reverse the positive sign to the negative sign or vice versa.

Ⓓ

Before execution

← 16bits →

0	0	1	1	1	0	1	0	1	1	0	1	0	0	1	1
---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

h3AD3
= 15059

Sign inversion
(2's complement)

1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

–

0	0	1	1	1	0	1	0	1	1	0	1	0	0	1	1
---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

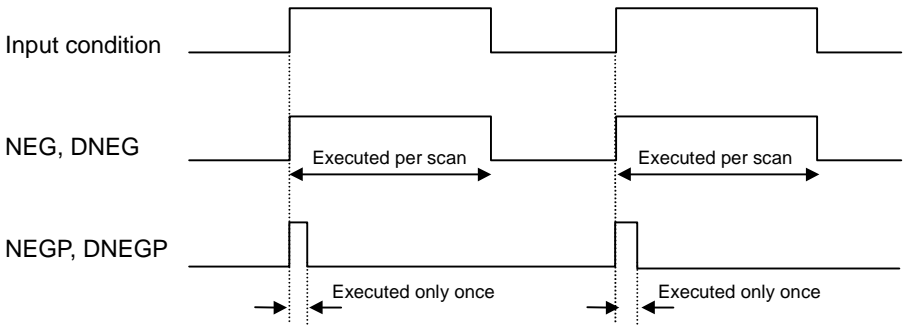
Ⓓ

After execution

1	1	0	0	0	1	0	1	0	0	1	0	1	1	0	1
---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

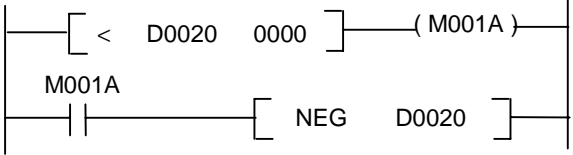
hC52D
= –15059

- Execution conditions



2) Program example

- Program that get an absolute value of D0000 when the value of D0000 is negative.





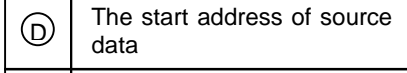
D0000	1	0	1	0	0	0	1	1	0	1	0	0	0	1	0	1	-23739
D0000	0	1	0	1	1	1	0	0	1	0	1	1	1	0	1	1	23739

5.20 Bit contact instructions

5.20.1 BLD, BLDN

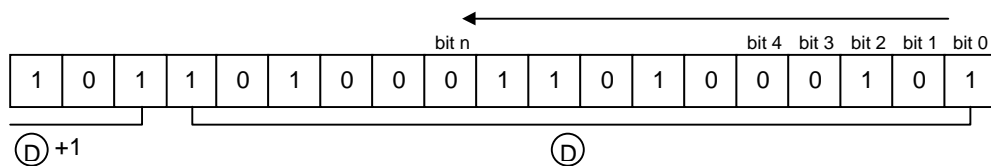
BLD (Bit load)	FUN(248) BLD FUN(249) BLDN	Applicable CPU	K200S K300S K1000S
-------------------	-------------------------------	-------------------	--------------------------

Instructions		Available Device											Steps	Flag		
		M	P	K	L	F	T	C	S	D	#D	Integer		Error (F110)	Zero (F111)	Carry (F112)
BLD	Ⓓ									O	O		5	O		
BLDN	n									O						

		Operand setting	
			

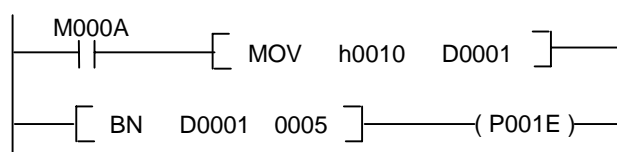
1) Functions

- BLD : Start a NO contact. Draw the on/off status of the n^{th} bit from the bit 0 of [D] and use the data as an operation result.
- BLDN : Start a NC contact. Draw the on/off status of the n^{th} bit from the bit 0 of [D] and use the data as an operation result.



2) Program example

- Program that turns P01E on when the 5th bit of the D0001 word is on.



5.20.2 BAND, BANDN

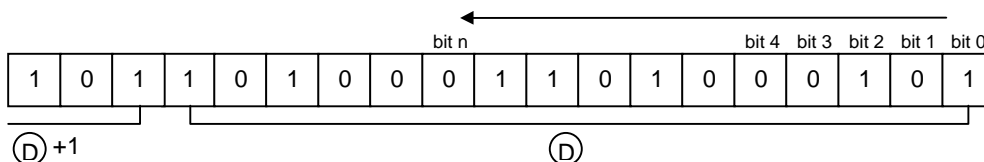
BAND (Bit AND)	FUN(250) BAND	Applicable CPU	K200S
	FUN(251) BANDN		K300S K1000S

Instructions		Available Device											Steps	Flag		
		M	P	K	L	F	T	C	S	D	#D	Integer		Error (F110)	Zero (F111)	Carry (F112)
BAND	Ⓓ									O	O		5	O		
BANDN	n									O						

Operand setting	
— [B Ⓓ n] —	Ⓓ The start address of source data
— [BN Ⓓ n] —	n Offset from the bit 0 to the destination bit

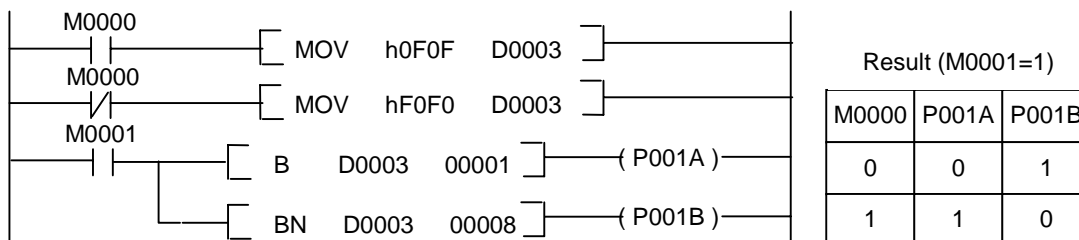
1) Functions

- BAND : A series connection of a NO contact. Reads the on/off status of the nth bit from the bit 0 of [D] and performs the AND operation with previous result, and use it as an new operation result.
- BANDN : A series connection of a NC contact. Reads the on/off status of the nth bit from the bit 0 of [D] and performs the AND operation with previous result, and use it as an new operation result..



2) Program example

- Program that turns on P001A and P001B according to the status of bit 1 and bit 8 of the D0003 word.



5.20.3 BAND, BANDN

BOR (Bit OR)	FUN(252) BOR	Applicable CPU	K200S
	FUN(253) BORN		K300S K1000S

Instructions		Available Device											Steps	Flag		
		M	P	K	L	F	T	C	S	D	#D	Integer		Error (F110)	Zero (F111)	Carry (F112)
BOR	Ⓓ									O	O		5	O		
BORN	n									O						

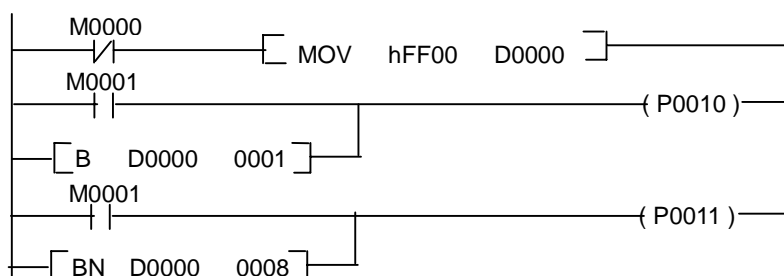
<div> <div> <div>┌</div> <div>└</div> </div> <div> <div>B</div> <div>Ⓓ</div> <div>n</div> </div> <div> <div>┌</div> <div>└</div> </div> </div> <div> <div>┌</div> <div>└</div> </div> <div> <div>BN</div> <div>Ⓓ</div> <div>n</div> </div> <div> <div>┌</div> <div>└</div> </div>		Operand setting	
Ⓓ		The start address of source data	
n		Offset from the bit 0 to the destination bit	

1) Functions

- BOR : A parallel connection of a NO contact. Reads the on/off status of the nth bit from the bit 0 of [D] and performs the OR operation with previous result, and use it as an new operation result.
- BORN : A parallel connection of a NC contact. Reads the on/off status of the nth bit from the bit 0 of [D] and performs the OR operation with previous result, and use it as an new operation result.

2) Program example

- Program that turns on P0010 and P0011 according to the on/off status of bit 1 and bit 8 of the D0000 word.



5.20.4 BOUT

BOUT (Bit output)	FUN(236) BOUT	Applicable CPU	K200S
			K300S
			K1000S

Instructions		Available Device											Steps	Flag		
		M	P	K	L	F	T	C	S	D	#D	Integer		Error (F110)	Zero (F111)	Carry (F112)
BOUT	Ⓓ									O	O		5	O		
	n									O						

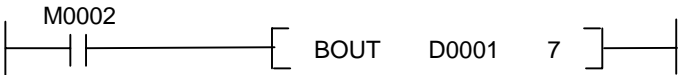
		Operand setting	
Ⓓ	The start address of source data	n	Offset from the bit 0 to the destination bit

1) Function

- Outputs the current operation result to the the nth bit from the bit 0 of [D].

2) Program example

- Program that turns on the bit 2 of D0001 when M0002 is on.



5.20.5 BSET, BRST

BSET / BRST (Bit set / reset)	FUN(232) BSET FUN(224) BRST	Applicable CPU	K200S K300S K1000S
----------------------------------	--------------------------------	-------------------	--------------------------

Instructions		Available Device											Steps	Flag		
		M	P	K	L	F	T	C	S	D	#D	Integer		Error (F110)	Zero (F111)	Carry (F112)
BSET	Ⓓ									O	O		5	O		
BRST	n									O						

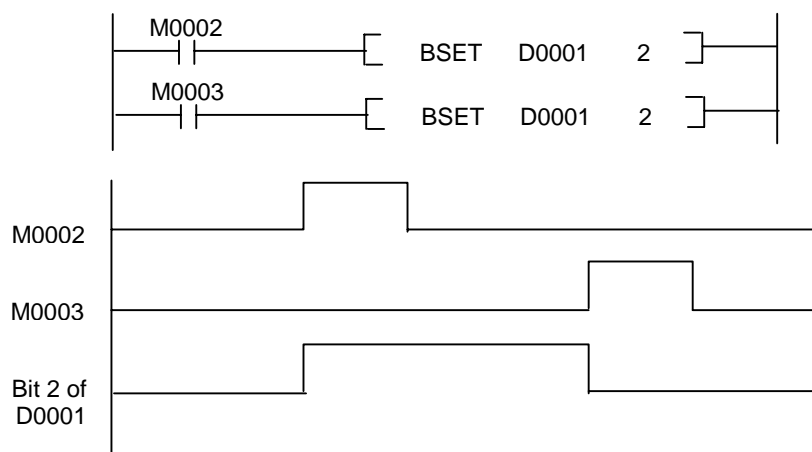
 		Operand setting	
Ⓓ	The start address of source data		
n	Offset from the bit 0 to the destination bit		

1) Functions

- BSET : When the input condition of BSET instruction turns on, the n^{th} bit from the bit 0 of [D] is switched on. The bit remains on state even if the input condition of BSET instruction is turned off. It can be switched off by the BRST instruction.
- BRST : When the input condition of BRST instruction turns on, the n^{th} bit from the bit 0 of [D] is switched off.

2) Program example

- Program that set the bit 2 of D0001 when M0002 is on, and reset the bit 2 of D0001 when M0003 is on.



5.21 Computer link module instructions

5.21.1 SND

SND (Send data and frame name to Cnet module)	FUN(169) SND	Applicable CPU	K200S K300S K1000S
--	--------------	----------------	--------------------------

Instructions		Available Device											Steps	Flag		
		M	P	K	L	F	T	C	S	D	#D	Integer		Error (F110)	Zero (F111)	Carry (F112)
SND	sl											O	11	O		
	Fnam	O	O	O			O	O		O	O					
	Ⓢ	O	O	O			O	O		O	O					
	n									O		O				
	SS	O	O	O			O	O		O	O					

Operand setting	
sl	Slot number of Cnet module is mounted & Type of channel (RS232C or RS422)
Fnam	The name of frame (8 words)
Ⓢ	Start address of device that stores source data to be sent
n	Numbers of byte to be sent
SS	Device at which the link status is stored

The configuration of 'sl'

AB	CD	Lower 8 bits (CD) : Slot No. of Cnet module
		Higher 8 bits (AB) : Type of channel h00 : RS232C h01 : RS422

1) Functions

- Sends 'n' bytes which begin with the device specified as [S] to the Cnet module that mounted on the slot 'sl'. The name of frame is stored as ASCII format into 8 words which begin with the device [Fnam]. The link status is stored at the device specified as [SS].
- The maximum size of data block to be sent is 256 bytes.

2) Program example

- Program that send 10 words from D1234 and frame name (8 words from D0000) to the Cnet module at slot 3, channel 0 (RS232C). The link status is stored at K015 word.

┌──┴──┐ [SND h0003 D0000 D1234 h0010 K015] ─┴──┐

5.21.2 RCV

RCV (Receive data and frame name from Cnet module)	FUN(168) RCV	Applicable CPU	K200S K300S K1000S
--	--------------	-------------------	--------------------------

Instructions		Available Device											Steps	Flag		
		M	P	K	L	F	T	C	S	D	#D	Integer		Error (F110)	Zero (F111)	Carry (F112)
RCV	sl											O	11	O		
	Fnam	O	O	O			O	O		O	O					
	Ⓓ	O	O	O			O	O		O	O					
	n									O		O				
	SS	O	O	O			O	O		O	O					

The configuration of 'sl'		Operand setting	
<p>Lower 8 bits (CD) : Slot No. of Cnet module Higher 8 bits (AB) : Type of channel h00 : RS232C h01 : RS422</p>		sl	Slot number of Cnet module is mounted & Type of channel (RS232C or RS422)
		Fnam	The name of frame (8 words)
		Ⓓ	Start address of device that stores source data to be sent
		n	Numbers of byte to be read
		SS	Device at which the link status is stored

1) Functions

- Receives 'n' bytes and frame name from the Cnet module that mounted on the slot 'sl', then stores the data from the device specified as [D], and frame name as ASCII format into 8 words which begin with the device [Fnam]. The link status is stored at the device specified as [SS].
- The maximum size of data block to be sent is 256 bytes.

2) Program example

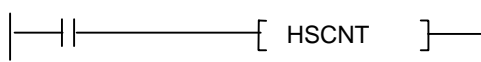
- Program that receive 20 words and frame name from Cnet module at slot 5, channel 1 (RS232C), and stores them to the block which begin with D0200 and D0100. The link status is stored at K016 word.

5.22 High speed counter instructions

5.22.1 HSCNT

HSCNT (Enable high speed counter)	FUN(210) HSCNT	Applicable CPU	K10S1 / K10S K30S / K60S
--------------------------------------	----------------	----------------	-----------------------------

Instructions	Available Device											Steps	Flag		
	M	P	K	L	F	T	C	S	D	#D	Integer		Error (F110)	Zero (F111)	Carry (F112)
HSCNT												1			



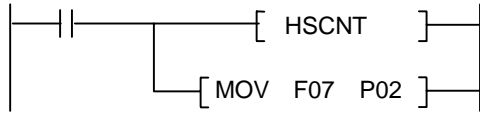
1) Functions

- Enable a high speed counter when the input condition turns on.
- After the high speed counter is enabled, it operates according to the parameter setting.
- When the input condition is switched off, the high speed counter is reset.
- The HSCNT instruction can not be used with HSC instruction in a sequence program simultaneously.
- The specification of high speed counter

Items	Contents	
Counting speed	1 point	
Phase	1 phase	
Count speed	8kpps	
Count range	0 ~ hFFFF (16 bits)	
Parameter setting	Level	Max. 20 level can be set up. (0 ~ 19)
	Data	Setting data (16 bits)
	SET	Bits to be switched on (8 bits)
	RESET	Bits to be switched off (8 bits)
Current value	F140 ~ F14F (16 bits)	
Setting value	F150 ~ F15F (16 bits)	
Output device	F070 ~ F077 (8 bits)	

2) Program example

- Program that output the high speed counter output to P002 word.



< Parameter setting with KGL-WIN>

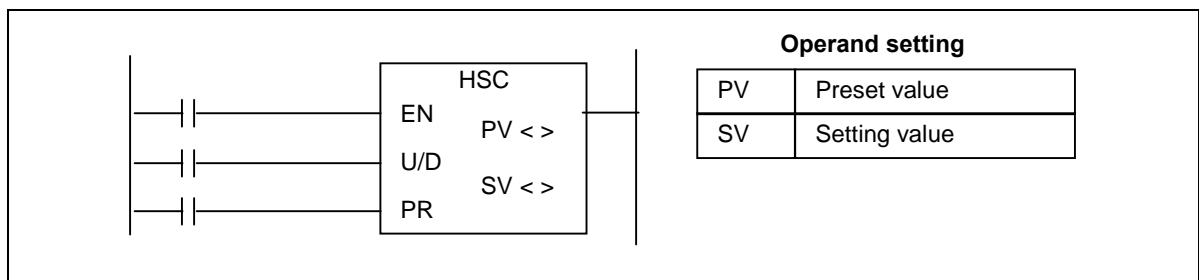


- When the input condition turns on, the current value is stored to F14 and setting value of step 0 is stored to F15.
- When the current value reaches to setting value #0, F070 ~ F077 is set / reset according to the parameter setting and F15 is updated as setting value of step 1.
- When the current value reaches to the setting value of last step (step 5 at the this example), F15 is updated as setting value of step 0 and current value (F14) is cleared as 0.
- If the input condition turns off, current value and HSC output (F070 ~ F077) is cleared as 0.

5.2.2.2 HSC

HSC (High speed counter)	FUN(215) HSC	Applicable CPU	K10S1 / K10S K30S / K60S
-----------------------------	--------------	----------------	-----------------------------

Instructions		Available Device											Steps	Flag		
		M	P	K	L	F	T	C	S	D	#D	Integer		Error (F110)	Zero (F111)	Carry (F112)
HSC	PV	O	O	O	O	O	O	O		O	O	O	7/9/11			
	SV	O	O	O	O	O	O	O		O	O	O				

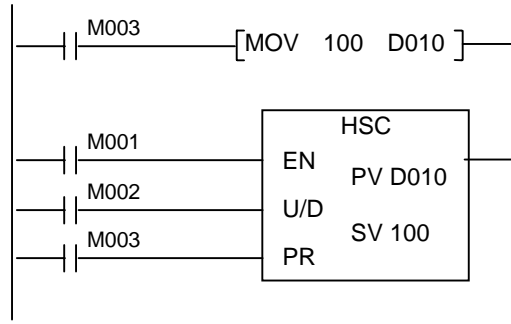


1) Functions

- HSC instruction can not be used with HSCNT instruction in a program. Only one of them can be used in a sequence program.
- 32-bits, up / down high speed counter. (HSCNT : 16-bits, up-counter)
- If the current value same or greater than SV, the HSC output bit (F070) turns on.
- The current value can not be changed by user.
- The current value is stored at F14 (lower word) and F15 (higher word).
- When the HSC instruction is used, the high speed counter parameter setting is ignored.
- Explanation of operands
 - a) EN : High speed counter enable contact
 - b) U/D : operates as up counter when U/D is 0, and down counter when U/D is 1.
 - c) PR : If the PR input turns on, the current value is changed as preset value (PV).
 - d) PV : Preset value. If the PV is specified as device [A], the PV is contents of [A+1, A].
 - e) SV : Setting value. If the SV is specified as device [A], the SV is contents of [A+1, A].

2) Program example

- M1 : HSC reset, M2 : U/D input (0 = up, 1 = down), M3 : Change current value as PV
- If the current value is same or greater than SV, the F070 bit turns on.



5.23 RS-485 communication instructions

5.23.1 RECV

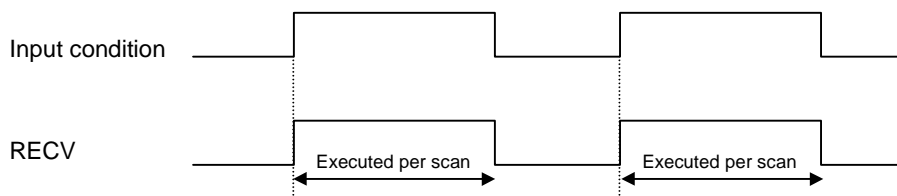
RECV (Receive data)	FUN(158) RECV	Applicable CPU	K10S1 / K10S K30S / K60S
------------------------	---------------	-------------------	-----------------------------

Instructions		Available Device										Steps	Flag		
		M	P	K	L	F	T	C	S	D	#D	Integer	Error (F110)	Zero (F111)	Carry (F112)
RECV	St	O	O	O	O	O	O	O		O	O	O	9	O	
	(D)	O	O	O	O	O	O	O		O	O				
	(S)	O	O	O	O		O	O		O					
	n	O	O	O	O	O	O	O		O	O	O			

Operand setting	
St	Station number of slave station to be read
(D)	The start address of device of master station at which the received data is stored
(S)	Start address of device of slave station that stores source data to be sent
n	Numbers of word to be read (n : h00 ~ h1F)

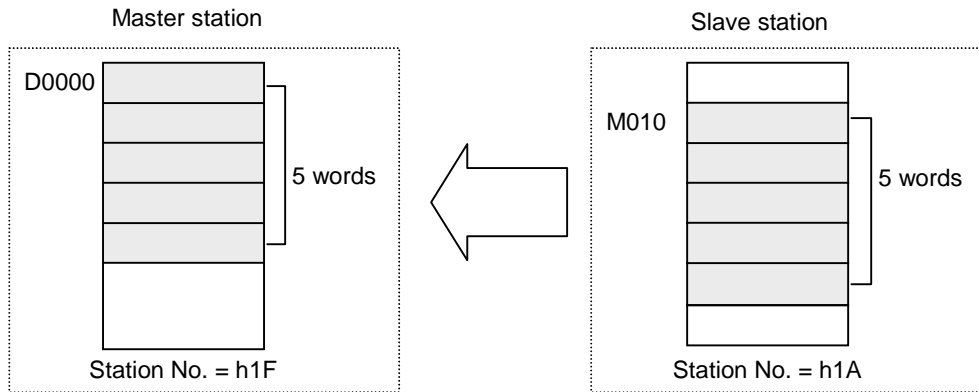
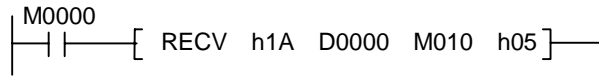
1) Functions

- Read 'n' words from the device specified as [S] of slave station (Station number = 'st'), and stores the read data into the block which begin with the device specified as [D] of master station.
- RECV instruction can be used with master station (station number = h1F) only.
- Execution condition



2) Program example

- Program that read 5 words from M010 of the slave station (station number = h1A), and stores the data to D0000 ~ D0004 of the master station while the M0000 turns on.



5.23.2 SEND

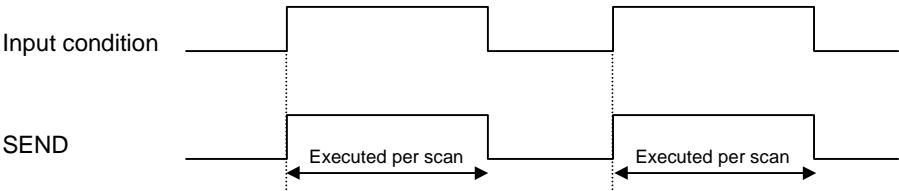
SEND (Send data)	FUN(159) SEND	Applicable CPU	K10S1 / K10S K30S / K60S
---------------------	---------------	-------------------	-----------------------------

Instructions		Available Device											Steps	Flag		
		M	P	K	L	F	T	C	S	D	#D	Integer		Error (F110)	Zero (F111)	Carry (F112)
SEND	St	O	O	O	O	O	O	O		O	O	O	9	O		
	Ⓢ	O	O	O	O	O	O	O		O	O					
	Ⓓ	O	O	O	O		O	O		O						
	n	O	O	O	O	O	O	O		O	O	O				

		Operand setting	
		St	Station number of slave station to which data to be written
		Ⓢ	The start address of device of master station at which the source data is stored
		Ⓓ	Start address of device of slave station at which the sent data is stored
		n	Numbers of word to be read (n : h00 ~ h1F)

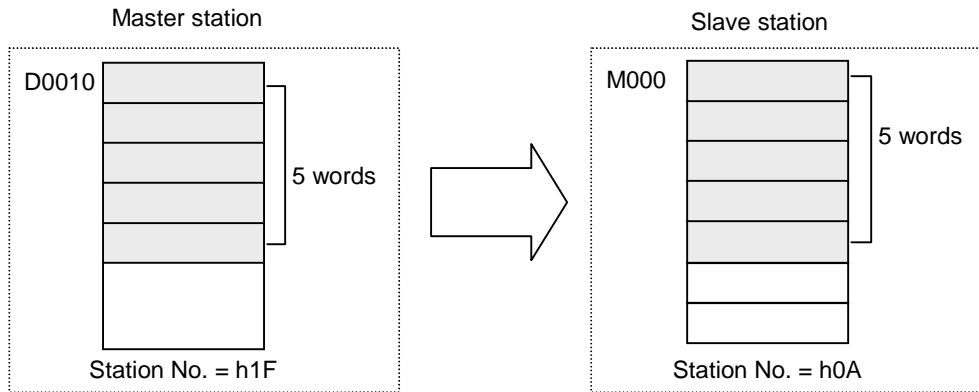
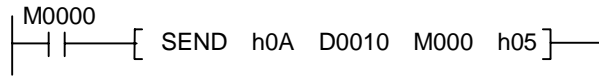
1) Functions

- Sends 'n' words from the device specified as [S] of master station, and stores the read data into the block which begin with the device specified as [D] of slave station (Station number = 'st')
- SEND instruction can be used with master station (station number = h1F) only.
- Execution condition



2) Program example

- Program that send 5 words from D0010 of the master station, and stores the data to M0000 ~ M0004 of the slave station (station number = h0A) while the M0000 turns on.



Appendix

- A.1 Memory configuration..... 1
- A.2 Special relay 3
- A.2 Instruction list..... 12

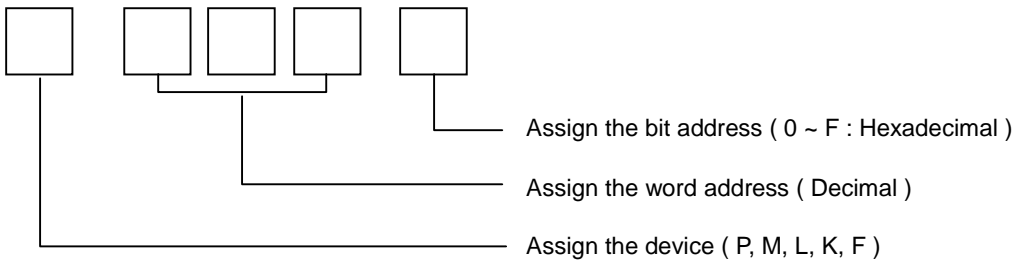
Appendix

A.1 Memory configuration

A.1.1 Bit memory device


The bit memory device is the memory area that can be read / write by bit. The P, M, L, K, F areas are bit memory devices. However, the bit memory device can be used as word device area.

< The notation of bit memory device >



< The memory structure of bit memory device >

	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0
000																
001																
002																
003																
nnn																

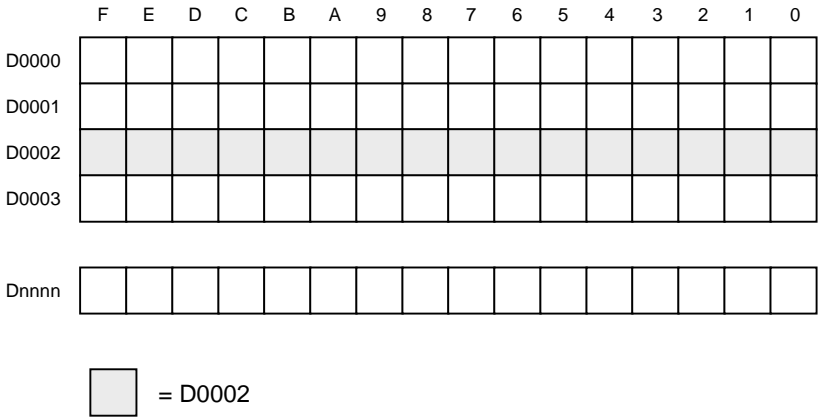
 = P002B

A.1.2 Bit / Word memory device (timer & counter)

The timer and counter memory area consist of 3 parts - the output bit, current value word, and setting value word. When the T or C device is used as a operand of bit instruction, the instruction takes effect to the output bit of timer or counter. If the T or C device is used as an operand of word instruction, the current value word is effected by the instruction. The setting value can not be changed by user.

A.1.3 Word memory device

The D device used by word. Therefore, the D device can not be used as an operand of bit instruction such as LOAD, OUT, etc. To control the D device by bit, use special instructions such as BLD, BAND, BOR, etc.

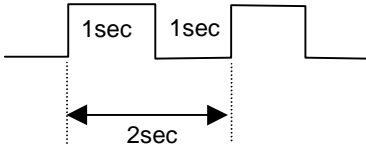
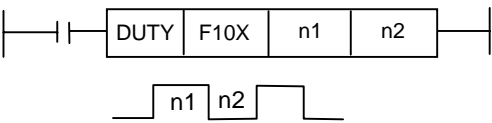


A.2 Special relay

A.2.1 K10S1 / K10S / K30S / K60S

1) F device

Relay	Name	Description
F000	Run flag	Set while PLC is on RUN mode
F001	PGM flag	Set while PLC is on PGM mode
F002	Pause flag	Set while PLC is on Pause mode
F007	EPROM mode	Set when PLC is on EPROM run mode.
F010	Always on	Used as a dummy relay or initialization in user programs
F011	Always off	
F012	1 scan On	On during the first scan after PGM→RUN mode
F013	1 scan Off	Off during the first scan after PGM→RUN mode
F014	Turnover per Each scan	Repeat set/reset during PLC is on RUN mode
F020 ~ F02F	Communication Error information	<ul style="list-style-type: none"> ● Replated to SEND, RECV instructions only ● Upper byte:The station No. where error occurred Lower byte:error code ● The error code of time out error:h20 ● No error:h000
F030	Serious error	Set in case of internal ROM error, 24V fail error
F031	Light error	Set in case of WDT error, program error, I/O combination error, missing END/RET error
F03A	RTC data error flag	Set when an error is detected in RTC data
F040 ~ F045	I/O combination error	Set in case of attachment/detachment of I/O unit During operation, or improper connection
F050 ~ F05F	Error code	<ul style="list-style-type: none"> ● h0000:No error ● h0023:Code error ● h0014:I/O error ● h0024:Missing END error ● h0021:Parameter error ● h0025:Missing RET error
F060 ~ F06F	The step No. Where error occurred	<ul style="list-style-type: none"> ● The step No. where program error occurred is stored ● In case of branch instruction error, the destination step No. is stored.
F070 ~ F077	HSC register	High speed counter area
F080 ~ F08F	PLC model	<ul style="list-style-type: none"> ● K10S:h0031 ● K60S:h0036 ● K30S:h0033 ● K100S:h0035 Upper byte:PLC station No. Lower byte:PLC model

Relay	Name	Description
F090	20msec period clock	<p>These relays repeat On/Off with fixed time interval, and are generated in RUN mode only.</p> <p style="text-align: center;">F094</p> 
F091	100msec period clock	
F092	200msec period clock	
F093	1sec period clock	
F094	2sec period clock	
F095	10sec period clock	
F096	20sec period clock	
F097	1minute period clock	
F100 ~ F107	User defined clock F100:clock0 ~ F107:clock7	<p>These relays repeat On/Off based on a scan time. (Initial state=Off)</p> 
F110	Arithmetic error flag	Set when an arithmetic error occurred during operation
F111	Zero flag	Set when the result value is zero
F112	Carry flag	Set when Carry or Borrow occurs as a result of operation
F11A	On sending flag	<p>These relays indicate the communication status When DIN, DOUT instruction are used.</p>
F11C	On receiving flag	
F11E	Receive completion Flag	
F11F	Communication error flag	<ul style="list-style-type: none"> ● DIN, DOUT: Set when time-out error occurred ● SEND, RECV: Set when time-out error occurs or NAK message is detected.
F120	<	<p>These relays are set according to the result of Compare instructions (CMP, CMPP, DCMPP, DCMPP)</p>
F121	≤	
F122	=	
F123	>	
F124	≥	
F125	≠	
F130 ~ F135	I/O status	Each relays show whether relevant I/O modules Are attached or not.
F140 ~ F14F	HSC present value	<p>HSCNT : The present value of high speed counter is stored.</p> <p>HSC : The low word of present value of high speed counter is stored.</p>
F150 ~ F15F	HSC preset value	<p>HSCNT : The preset value of high speed counter is stored.</p> <p>HSC : The high word of present value of high speed counter is stored.</p>


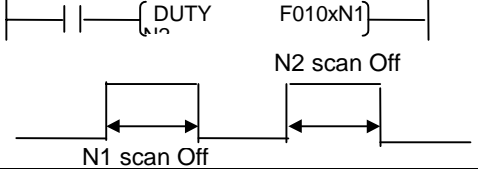
2) Other special relays

Area	Description		Remarks
M310	RTC User Write Enable		When M310 is On, the Data of RTC is changed As the data of D249~D252
L12~L15	RTC Data		
D240	Analog Unit #1	Data of A/D Ch.0 Input	K30S-A / K60S-A type Only
D241		Data of A/D Ch.1 Input	
D242		Data of D/A Output	
D243	Analog Unit #1	Data of A/D Ch.0 Input	
D244		Data of A/D Ch.1 Input	
D245		Data of D/A Output	
D247	High Speed Counter Mode Set Area		
D248	Time out Value of RS485 Communications		O/S V1.5 or later
D249~D252	RTC User Write Data Area		MK-S Series Ver1.3 or Later Used when M310 Point is on Same format As L12~L15
D253	Current Scan Time		
D254	Minimum Scan Time		
D255	Maximum Scan Time		

A.2.2 K200S / K300S / K1000S

1) F relay

Contact	Keyword	Function	Description
F0000		RUN mode	Turns on when the CPU in the RUN mode.
F0001		Program mode	Turns on when the CPU in the Program mode
F0002		Pause mode	Turns on when the CPU in the Pause mode
F0003		Debug mode	Turns on when the CPU in the Debug mode
F0006		Remote mode	Turns on when the CPU in the Remote mode
F0007		User memory installation	Turns on when a user memory is installed.
F0008 and F0009		Unused	
F000A		User memory operation	Turns on when a user memory is being operated
F000B to F000E		Unused	
F000F	p_STOP	Execution of the STOP instruction	Turns on when the STOP instruction is being operated.
F0010	p_ON	Always On	Always On
F0011	p_OFF	Always Off	Always Off
F0012	p_1ON	1 Scan On	1 Scan On
F0013	p_1OFF	1 Scan Off	1 Scan Off
F0014	p_STOG	Scan toggle	Scan toggle
F0015 to F001F		Unused	
F0020		1 step run	Turns on when the 1 step run is operated in the Debug mode.
F0021		Breakpoint run	Turns on when the breakpoint run is operated in the Debug mode.
F0022		Scan run	Turns on when the scan run is operated in the Debug mode.
F0023		Coincident junction value run	Turns on when the coincident junction run is operated in the Debug mode.
F0024		Coincident word value run	Turns on when the coincident word run is operated in the Debug mode.
F0025 to F002F		Unused	
F0030	s_HEAVY_ER	Fatal error	Turns on when a fatal error has occurred.
F0031	s_LIGHT_ER	Ordinary error	Turns on when an ordinary error has occurred.
F0032	s_WD_ER	WDT Error	Turns on when a watch dog timer error has occurred.
F0033	s_IO_TYER	I/O combination error	Turns on when an I/O error has occurred. (When one or more bit(s) of F0040 to F005F turns on)
F0034	s_BAT_ER	Battery voltage error	Turns on when the battery voltage has fallen below the defined value.
F0035	s_FUSE_ER	Fuse error	Turns on when a fuse of output modules has been disconnected.
F0036 to F0038		Unused	
F0039	s_D_BCK_OK	Normal backup operation	Turns on when the data backup is normal.
F003A	s_RTC_ER	RTC data error	Turns on when the RTC data setting error has occurred.
F003B		During program edit	Turns on during program edit while running the program.
F003C		Program edit error	Turns on when a program edit error has occurred while running the program.
F003D to F003F		Unused	

Junction	Keyword	Function	Description
F0040 to F005F	S_IO_TYER (0 to 31)	I/O error	When the reserved I/O module (set by the parameter) differs from the real loaded I/O module or a I/O module has been mounted or dismounted, the corresponding bit turns on.
F0060 to F006F		Storing error code	Stores the system error code, (See Section 2.9)
F0070 to F008F	S_FUSE_ER	Storing the disconnection state of fuses	When a fuse has disconnected in an output module, the corresponding bit to the slot turns on.
F0090	_p_T20MS	20-ms cycle clock	<p>Turning On/Off is repeated with a constant cycle.</p> 
F0091	_p_T100MS	100-ms cycle clock	
F0092	_p_T200MS	200-ms cycle clock	
F0093	_p_T1S	1-sec cycle clock	
F0094	_p_T2S	2-sec cycle clock	
F0095	_p_T10S	10-sec cycle clock	
F0096	_p_T10S	20-sec cycle clock	
F0097	_p_T60S	60-sec cycle clock	
F0098 to F009F		Unused	
F0100		User clock 0	<p>Turning On/Off is repeated as many times as the scan specified by Duty instruction.</p> 
F0101		User clock 1	
F0102		User clock 2	
F0103		User clock 3	
F0104		User clock 4	
F0105		User clock 5	
F0106		User clock 6	
F0107		User clock 7	
F0108 to F010f		Unused	
F0110	_p_ERR	Operation error flag	Turns on when an operation error has occurred.
F0111	_p_ZERO	Zero flag	Turns on when the operation result is "0".
F0112	_p_CARRY	Carry flag	Turns on when a carry occurs due to the operation.
F0113	_p_OUT_OFF	All outputs off	Turns on when an output instruction is executed.
F0114		Common RAM R/W error	Turns on when a memory access error of the special module has occurred.
F0115		Operation error flag (Latch)	Turns on when an operation error has occurred.(Latch)
F0116 to F011F		Unused	
F0120	_p_LT	LT flag	Turns on if $S_1 < S_2$ when using the CMP instruction.
F0121	_p_LTE	LTE flag	Turns on if $S_1 \leq S_2$ when using the CMP instruction.
F0122	_p_EQU	EQU flag	Turns on if $S_1 = S_2$ when using the CMP instruction.
F0123	_p_GT	GT flag	Turns on if $S_1 > S_2$ when using the CMP instruction.
F0124	_p_GTE	GTE flag	Turns on if $S_1 \geq S_2$ when using the CMP instruction.
F0125	_p_NEQ	NEQ flag	Turns on if $S_1 \neq S_2$ when using the CMP instruction.

Junction	Keyword	Function	Description
F0126 to F012F		Unused	
F0130 to F013F	_S_AC_F_CNT	AC Down Count	Stores AC down counting value.
F0140 to F014F	_S_FALS_N	FALS No.	The error code generated by FALS instruction is stored to this flag.
F0150 to F015F		PUT/GET error flag	When a common RAM access error of special modules has occurred an output module, the corresponding bit to the slot turns on.
F0160 to F049F		Unused	
F0500 to F050F	S_SCAN_MAX	Maximum scan time	Stores the maximum scan time.
F0510 to F051F	S_SCAN_MIN	Minimum scan time	Stores the minimum scan time.
F0520 to F052F	S_SCAN_AVG	Present scan time	Stores the present scan time.
F0530 to F053F		Clock data (year/month)	Clock data (year/month)
F0540 to F054F		Clock data (day/hour)	Clock data (day/hour)
F0550 to F055F		Clock data (minute/second)	Clock data (minute/second)
F0560 to F056F		Clock data (day of the week)	Clock data (day of the week)
F0570 to F058F		Unused	
F0590 to F059F	S_CODE_ER_N	Storing error step	Stores the error step of the program.
F0600 to F060F	S_ERR-TYP	Storing FMM step	If a FMM related error has occurred, its occurrence information is stored.
F0610 to F063F		Unused	

Appendix

High speed link flag list

x : K1000S = 9, K300S / K200S = 4, n = 0 ~ 7 (Slot No.)

Keyword	Type	Address	Name	Description
_CnSTNOL _CnSTNOH	Dword	Dx500 Dx502	Communications module station No.	Indicates the number which is set on communications module station switch. Enet/Mnet : MAC station No. marked on the front of communication module. Fne/Fdnett : Station switch No. marked on the front of communications module.
_CnTXECN T	Word	Dx504	Communications frame sending error	Increments by one whenever sending error of communications frame occurs. Connection condition of network is evaluated by this value.
_CnRXECN T	Word	Dx505	Communications frame receiving error	Increments by one whenever receiving error of communications frame occurs. Connection condition of network is evaluated by this value.
_CnSVCFC NT	Word	Dx506	Communications service processing error	Increments by one whenever communications service fails. Connection condition of network and overall communication quantity and program stability can be evaluated by this value.
_CnSCANM X	Word	Dx507	Maximum communications scan time (unit : 1 ms)	Indicates the maximum time that is spent until every station connected to network has the token at least one time and sends a sending frame.
_CnSCANA V	Word	Dx508	Average communications scan time (unit : 1 ms)	Indicates the average time that is spent until every station connected to network has the token at least one time and sends a sending frame.
_CnSCANM N	Word	Dx509	Minimum communications scan time (unit : 1 ms)	Indicates the minimum time that is spent until every station connected to network has the token at least one time and sends a sending frame.
_CnLINF	Word	Dx510	Communications module system information	Indicates operation state of communications module with a word.
_CnCRDER	Bit	Dx510.0	System error (error = 1)	Indicates communications module hardware or system O/S error.
_CnSVBSY	Bit	Dx510.1	Insufficient common RAM (Insufficient = 1)	Indicates that service cannot be offered due to insufficient common RAM.
_CnIFERR	Bit	Dx510.2	Interface error (error = 1)	Indicates that interface with communications modules has been stopped.
_CnINRING	Bit	Dx510.3	In-ring (IN_RING = 1)	Indicates that the communications module can communicates with other station or not.
_CnLNKMO D	Bit	Dx510.4	Operation mode (RUN=1)	Indicates that operation mode of communications module is in the normal operation mode or test mode.
_CnVERNO	Word	Dx680.	Version No. of communications module	O/S version No. of communications module
_FSMn_ST _NO	Word	Dx690	Numbers of remote I/O stations. (Write is enabled)	Sets the remote I/O station number to the upper 8 bits. (See REMARK given in the following page)
_fsmn_RES ET	Bit	Dx690.0	Remote I/O station S/W reset	Initializes special modules and I/O modules in the remote station defined by the FSMn_st_no.
_fsmn_IO_ RESET	Bit	Dx690.1	Remote I/O station digital output reset	Clears the output of I/O modules in the remote station defined by the FSMn_st_no.
_fsmn_IO_ RESET	Bit	Dx690.2	Initialize the high speed link information of remote I/O station	If a momentary power failure occurs in the remote I/O station, the operation mode bit of high speed link information turns off and link trouble has the value 1. If the bit is turned on to clear that bit, the operation mode bit turns on and link trouble is cleared with 0.

Slot No. & Flag List

Slot No.	D area address	Remark
1	Dx511 to Dx521	<p>The address of the flag which is loaded onto the slot n is calculated as shown below.</p> <p>* Address of D area = Address shown in the [TABLE1] + 11 × n, where n = 1 to 7</p> <p>Example) Address for the average communications scan time of the communications module loaded on the slot 6. → Dx508 + 11 × 6 = Dx574</p>
2	Dx522 to Dx532	
3	Dx533 to Dx543	
4	Dx544 to Dx554	
5	Dx555 to Dx565	
6	Dx566 to Dx576	
7	Dx577 to Dx587	

Detailed High Speed Link Information Flag List (when m=0)

Keyword	Type	Bit Address	Name	Description
_HSmRLINK	Bit	Dx600.0	High speed link normal run information(RUN_LINK)	<p>Indicates that all stations are normally operating complying with the parameter set in the high speed link. This flag turns on under the following conditions.</p> <ol style="list-style-type: none"> 1. All stations set in the parameter are in the RUN mode and have no error, and 2. All blocks set in the parameter normally communicate, and 3. The parameters set in all stations, which are set in the parameter, normally communicate. <p>Once this flag is turned on, it maintains that state as long as link disable does not make that state stopped.</p>
_HSmLTRBL	Bit	Dx600.1	High speed link trouble abnormal run information	<p>This flag turns on when, under the condition that _HSmRLINK is turned on, communications of the stations and data blocks set in the parameter is under the following conditions.</p> <ol style="list-style-type: none"> 1. A station set in the parameter is not in the RUN mode, or 2. A station set in the parameter has an error, or 3. The communications of data blocks set in the parameter does not normally operate. <p>This flag turns on if the above conditions 1), 2) and 3) occur. If normal conditions are restored, it will turn off again.</p>
_HSmSTATE[k] (k = 0 to 63)	Bit Array	Dx601.0 to Dx604.15	Overall communications state information of K Data Block set by the high link parameter	Indicates overall communications state of every blocks of the parameters set. _HSmSTATE[k] = _HSmMOD[k] & _HSmTRX[k] & _HSmERR[k]
_HSmMOD[k] (k = 0 to 63)	Bit Array	Dx605.0 to Dx608.15	K Data Block setting stations mode information. (RUN = 1, others = 0)	Indicates the operation modes of stations set the K data block of parameters.
_HSmTRX[k] (k = 0 to 63)	Bit Array	Dx609.0 to Dx612.15	K Data Block communications state information (Normal = 1, abnormal = 0)	Indicates whether communications of the K data block of parameters are normally operating as set.
_HSmERR[k] (k = 0 to 63)	Bit Array	Dx613.0 to Dx616.15	K Data Block setting stations state information. (Normal = 1, abnormal = 0)	Indicates whether the stations set in the K data block of parameters have an error.

Detailed High Speed Link Information Flag List (when m= 1 to 3)

High Speed Link Type	D area Address	Remark
High Speed Link 2 (m=1)	Dx620 to Dx633	Compared to the D area addresses shown in the [TABLE 3], where m = 0, they are calculated as shown below where m = 1 to 3. * Address of D area = Address shown in the [TABLE3] + 11 × m, where n = 1 to 3
High Speed Link 3 (m=2)	Dx640 to Dx653	
High Speed Link 4 (m=3)	Dx660 to Dx673	

Slave System Flag List

Keyword	Name	Start Address (hexadecimal)	Data Type	Size	Remark
_CPU_Type	Remote CPU Type	h0000	Word	2 Byte	
_VER_NUM	O/S Version Number	h0002	Word	2 Byte	
_SYS_STATE	System State	h0004	Word	2 Byte	
_FSMTXECNT	TX Error Count	h0006	Word	2 Byte	
_FSMRXECNT	RS Error Count	h0008	Word	2 Byte	
_FSMSVCFcnt	Service Fail Count	h000A	Word	2 Byte	
_FSMScanMX	Maximum Scan Time	h000C	Word	2 Byte	
_FSMScanAV	Average Scan Time	h000E	Word	2 Byte	
_FSMScanMI	Minimum Scan Time	h0010	Word	2 Byte	
_MONTHSTNO	Mother Station No.	h0012	Word	2 Byte	
_FSMVRcnt	Variable RD Count	h0014	Word	2 Byte	
_FSMVWCNT	Variable WR Count	h0016	Word	2 Byte	
_FSMHSTXCNT	HS-Link TX Count	h0018	Word	2 Byte	
_FSMHSRXCNT	HS-Link RX Count	h001A	Word	2 Byte	
_AC_Fail_CNT	Power Fail Counter	h001C	Word	2 Byte	
_IO_TYER_N	Module Setting Error	h0020	Word	2 Byte	
_CNF_ER b0 : CPU_ER b1 : IO_TYER b2 : IO_DEER b3 : FUSE_ER b4 : IO_RWER b5 : IP_IFER b6 : PWR_ERR	Representative Flag CPU H/W defect Module Setting Error Mounting/Dismounting Error Fuse Blown Error I/O Access Error I/P Access Error Sub Power Error	h001E	Word Bit 0 Bit 1 Bit 2 Bit 3 Bit 4 Bit 5 Bit 6	2 Byte	Bit information (b7 to b15 : reserved)
_IO_DEER_N	Mounting/Dismounting Error	h0022	Word	2 Byte	
_FUSE_ER_N	Fuse Error	h0024	Word	2 Byte	
_IO_PWER_N	I/O Error	h0026	Word	2 Byte	
_IP_IFER_N	Special Module Error	h0028	Word	2 Byte	
_KGL_CNF b0 : local connection b1 : remote connection	KGL Connection Error	h002A	Byte	1 Byte	
_E_DATA_OPTION	Emergency Data Output type	h002B	Byte	1 Byte	0 : Output Latch 1 : User-Defined

A.3 Instruction list

Function No.	0	1	2	3	4	5	6	7	8	9
00x	NOP	END	STC	CLC	RET	MPUSH	MLOAD	MPOP	STOP	CLE
01x	MCS	MCSCLR	JMP	JME	CALL	CALLP	SBRT	D	DNOT	
02x	INC	INCP	DINC	DINCP	DEC	DECP	DDEC	DDECP	LD= ●	LDD= ●
03x	ROL	ROLP	DROL	DROLP	ROR	RORP	DRDR	DRDRP	LD> ●	LDD> ●
04x	RCL	CMPP	DRCL	DRCLP	RCR	RCRP	DRCR	DRCRP	LD< ●	LDD< ●
05x	CMP	BCDP	DCMP	DCMPP	TCMP	TCMPP	DTCMP	DTCMPP	LD>= ●	LDD>= ●
06x	BCD	WSFTP	DBCD	DBCDP	BIN	BINP	DBIN	DBINP	LD<= ●	LDD<= ●
07x	WSFT	MOVP	MULS ●	MULSP ●	BSFT	BSFTP	DMULS ●	DMULSP ●	LD<> ●	LDD<> ●
08x	MOV	GMOVP	DMOV	DMOVP	CMOV	CMOVP	DCMOV	DCMOV	DIVS ●	DIVSP ●
09x	GMOV	BMOVP	FOMV	FOMVP	AND= ●	ANDD= ●	AND> ●	ANDD> ●	AND< ●	ANDD< ●
10X	BMOV	ADDP	XCHG	XCHGP	DXCHG	DXCHGP	AND>= ●	ANDD>= ●	ANDD>= ●	ANDD>= ●
11X	ADD	ADDP	DADD	DADDP	SUB	SUBP	DSUB	DSUBP	AND<> ●	ANDD<> ●
12X	MUL	MULP	DMUL	DMULP	DIV	DIVP	DDIV	DDIVP	DDIVS	DDIVSP
13X	ADDB	ADDBP	DADDB	DADDBP	SUBB	SUBBP	DSUBB	DSUBBP		
14X	MULB	MULBP	DMULB	DMULBP	DIVB	DIVP	DDIVB	DDIVBP		
15X	WAND	WANDP	DWAND	DWAND	WOR	WORP	DWIR	DWORP	RECV ■	SEND ■
16X	WXOR	WXORP	DWXOR	DWXOR	WXNR	WXNRP	DWXNR	DWXNR	RCV ●	SND ●
17X	BSUM	BSUMP	DBUSM	DBUSMP	SEG	SEGP	ENCO	ENCOP	DECO	DECOP
18X	BSUM	FILRP	DFILR	DFILRP	FILW	FILWP	DFILW	DFILWP	OR= ●	ORD= ●
19X	ASC	ASCP	UNI	DSI	DIS	DISP	OR> ●	ORD> ●	OR< ●	ORD< ●
20X	IORF ●	IORFP ●	WDT ●	WDTP ●	FALS ●	DUTY	FOR ●	NEXT ●	OUTOFF	.
21X	HSCNT ■	DIN	DINP	DOUT	DOUTP	HSC ■	OR>= ●	ORD>= ●	OR<= ●	ORD<= ●
22X	BREAK ●	EI ●	DI ●	BSET ●	BRST ●	IRET ●	TDINT ●	INT ●	OR<> ●	ORD<> ●
23X	GET ●	GETP ●	RGET ●	RPUT ●	PUT ●	PUTP ●	BOU ●	SR ●	EI ●	DI ●
24X	NEG ●	NEGP ●	DNeg ●	DNegP ●	READ ●	WRITE ●	CONN ●	STATUS ●	BLD ●	BLDN ●
25X	BAND ●	BANDN ●	BOR ●	BORN ●						

● : Available with K1000S, K300S, K200S series only

■ : Available with K10S, K10S1, K30S, K60S series only