hpmont

HC10 Series
Intelligent Controller

Programming Manual

HC00 Series
Intelligent Controller

V1.2 2022.01

FOREWORD

Thank you for using the HC10 Series Intelligent Controller developed by Shenzhen Hpmont Technology Co., Ltd.

HC10 Intelligent Controller has rich instructions, strong high-speed signal processing ability and fast calculation speed. Its allowable user program capacity can reach 16k steps without external storage device.

The controller has a variety of communication interfaces (RS485, RS422, CAN), supporting a variety of communication protocols. Moreover, it is convenient for online and networking control together with inverters, touch screens and other equipment. Some models have 2 analog inputs and 2 analog outputs, switchable voltage/current, easy to connect to various analog signal sensors; With up to 4 pulse inputs and 4 pulse outputs, both of which support up to 100K, convenient for positioning control of the motor.

The controller provides a variety of programming languages. Users can choose programming methods such as ladder diagrams, instruction lists, and SFC sequential function charts. It provides strict user program security functions, which is convenient for users to control the intellectual property rights of process control.

Before using, please read this user manual carefully. At the same time, please fully understand the safety precautions of the product before using the product.

Note:

- **Preserve this Manual for future use.**
- **If you need the User Manual due to damage, loss or other reasons, please contact the regional distributor of our company or directly contact our company Technical Service Center.**
- **If you still have some problems during use, please contact our company Technical Service Center.**
- **Due to product upgrade or specification change, and for improving convenience and accuracy of this manual, this manual's contents may be modified.**
- **Email address: marketing@hpmont.com**

Version and Revision Records

Time: 2022/01 **Version: V1.2**	
Modified Chapter	**Modified Content**
	• V1.2 version released

CONTENTS

Chapter 1 Outline

In this chapter, the basic functions of HC10 intelligent controller are described.

In the basic functions, including the characteristics of the intelligent controller and the typical function introduction, parameters, memory operation, etc. required for the user to effectively use the functions of the intelligent controller, please read it before designing the program.

1.1 Programming Language

1.1.1 Programming Language Types

Instruction List Programming

The instruction list programming mode is the way to input the sequence instruction through the instruction languages such as "LD", "AND", and "OUT".

This method is the basic input form in the sequence program.

An example of a list display is shown below:

Step	Instruction	Soft Component Number
0000	LD	X000
0001	OR	Y005
0002	ANI	X002
0003	OUT	Y005
…	…	…

Ladder Editing

The ladder programming is to draw the sequence ladder figure on the programming software by using sequence symbol and soft components number. Since the sequence loop is realized by contact symbol and coil symbol, the content of the program is easier to understand.

The operation monitoring of the intelligent controller can be performed even in the state of the ladder figure.

SFC (STL <Step Ladder Programming>)

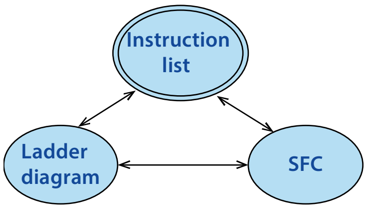
SFC (Sequence Function Figure) program is a way to design a sequence according to the mechanical action flow.

Interchangeability between SFC programs and other programs: Instruction list programs and ladder programs that can be converted to each other. If compiled according to certain rules, they can be converted to SFC figure in reverse.

1.1.2 Interchangeability of Programs

The sequence program created by the above three methods is saved to the program memory of the intelligent controller by the instruction (contents of the instruction list programming).

Programs compiled using various input methods as shown in the following figure can be converted and then displayed and edited.



1.2 Action and Overview of Soft Components

Soft Components		Instruction
1	Input (X)・output (Y) relay	• In each basic unit, the number of the input relay and output relay in octal is assigned according to X000 ~ X007, X010 ~ X017..., Y000 ~ Y007, Y010 ~ Y017... The number of the expansion unit and the expansion module is also the serial number of each of the hexadecimal numbers of X and Y in the order of connection from the basic unit.
2	Auxiliary relay (M)	• The relay inside the intelligent controller is an auxiliary relay. Unlike the input/output relay, it is not able to read the external input or directly drive the external load. • A relay that can hold the ON/OFF status even if the power of the intelligent controller is turned off.
3	Status (S)	• A relay used as a step ladder figure to indicate the engineering number. • When not used as a project number, it is the same as an auxiliary relay and can be programmed as a general contact/coil. • Can be used as a signal alarm for diagnosing external faults.
4	Timer (T)	• The timer accumulates the 1ms, 10ms, 100ms and other clock pulses inside the intelligent controller. When the accumulated result reaches the set value, the output contact action. According to the basic clock pulse, the timer can measure 0.001 ~ 3276.7 seconds. • T192 ~ T199 are timers specific to subroutines and interrupt subroutines.
5	Counter (C)	The counters are of the following types. They can be used separately depending on the purpose and use. 1. For counter (hold) • The counter is used by the internal signal of the intelligent controller, and its response speed is constant below 10 kHz. • 16-bit counter: For counting up, counting range 1 ~ 32,767. • 32-bit counter: Up/down count, count range -2,147,483,648 ~ +2,147,483,647. 2. For high speed counter • The high speed counter has nothing to do with the operation of the intelligent controller. • 32-bit counter: Up/down count, count range -2,147,483,648 ~ +2,147,483,647. • Single-phase single count, single-phase double count, and two-phase double count are assigned in specific input relays.
6	Data register (D)	The data register is the soft component that holds the data. The data registers of the intelligent controller are all 16 bits (the most significant bit is positive and negative), and the combination of 2 registers can handle the value of 32 bits (the most significant bit is positive and negative).
7	Index register (V, Z)	In the register, there are two registers, V and Z, which are called indexing (modification). By adding V and Z to other soft components, you can access the value of the address after the soft component is offset. The offset is V, Z. That is, after using V and Z to modify the soft components, access number is the current soft component number +V□ or + the soft component og the value of Z□. • When V0, Z0 = 5, D100V0 = D105, C20Z0 = C25.
8	Pointer (P) (I)	In the pointer, it is divided into two types: Branch and interrupt. • Branch pointer (P) is the object destination for specifying the CJ (FN 00) conditional branch and the CALL (FN 01) subroutine call. • Interrupt pointer (I) is an interrupt subroutine for specifying input interrupt, timer interrupt, or counter interrupt.
9	Constant (K) (H) (E)	Among the various values used in the intelligent controller, K represents decimal number, H represents hexadecimal number, and E represents real number (floating point number). These can be set value and current value of the timer and counter, or the operand of the application instruction.

1.3 Memory Operation and Outage Maintenance

The operation of the data memory, bit soft components memory and in-program memory of the HC10 intelligent controller is shown in the following table.

Types of Program Memory					
Project		**Power OFF**	**Power OFF→ON**	**STOP→RUN**	**RUN→STOP**
Parameter, sequence program		Not change			
Types of Word Soft Components					
Project		**Power OFF**	**Power OFF→ON**	**STOP→RUN**	**RUN→STOP**
Data register (D)	For general	Clear		Not change	Clear
				When M8033 = ON, it does not change	
	For power outage	Not change			
	For special use	Clear	Initial value setting	Not change	
Index register (V, Z)	V, Z	Clear		Not change	
Timer current value register (T)	100ms	Clear		Not change	Clear
				When M8033 = ON, it does not change	
	10ms	Clear		Not change	Clear
				When M8033 = ON, it does not change	
	Accumulated 100ms	Clear		Not change	Clear
				When M8033 = ON, it does not change	
	Accumulated 1ms	Clear		Not change	Clear
				When M8033 = ON, it does not change	
Counter current value register (C)	For general	Clear		Not change	Clear
				When M8033 = ON, it does not change	
	For power outage	Not change			
	For high speed	Not change			
Clock data	Current value	Keep timing			
1): Some of the devices are restored to their initial values when STOP→RUN.					
Types of Bit Soft Components Memory					
Project		**Power OFF**	**Power OFF→ON**	**STOP→RUN**	**RUN→STOP**
Contact image area (X, Y, M, S)	Input relay (X)	Clear		Not change	
	Output relay (Y)	Clear		Not change	Clear
				When M8033 = ON, it does not change	
	General auxiliary relay (M)	Clear		Not change	Clear
				When M8033 = ON, it does not change	
	Auxiliary relay for power failure maintenance (M)	Not change			
	Special auxiliary relay (M)	Clear	Initial value setting	Not change	
	General state (S)	Not change			
	Power failure maintenance state (S)	Not change			
	Signal alarm (S)	Not change			
Timer contact timing coil (T)	100ms	Clear		Not change	Clear
				When M8033 = ON, it does not change	
	10ms	Clear		Not change	Clear
				When M8033 = ON, it does not change	
	Accumulated 100ms	Clear		Not change	Clear
				When M8033 = ON, it does not change	
	Accumulated 1ms	Clear		Not change	Clear
				When M8033 = ON, it does not change	

Project		Power OFF	Power OFF→ON	STOP→RUN	RUN→STOP
Counter contact Counting coil Reset coil (C)	For general	Clear		Not change	Clear
				When M8033 = ON, it does not change	
	For power outage	Not change			
	For high speed	Not change			
1): Some of the devices are restored to their initial values during STOP→RUN.					

1.4 Data Types

In HC10 intelligent controller, depending on the different usage and purpose, there are six values available. The effects and functions are as follows.

1. DEC: DECIMAL NUMBER

Set value of timer and counter (K constant).

Auxiliary relay (M), timer (T), counter (C), status, etc. (soft component number).

The numerical value in the operand of the application instruction and the specification of the instruction action (K constant).

2. HEX: HEXADECIMAL NUMBER

The numerical value in the operand of the application instruction and the specification of the instruction action (H constant).

3. BIN: BINARY NUMBER

The numerical designation of the timer, counter or data register is performed according to decimal and hexadecimal numbers, but within the intelligent controller, these values are processed in binary numbers.

In addition, when monitoring these soft components on the peripheral device, it will be automatically converted to decimal number and displayed, or it can be switched to hexadecimal.

Inside the intelligent controller, the negative number is represented by the complement code. For details, please refer to the description of the NEG (FN 29) instruction.

4. OCT: OCTAL NUMBER

In HC10 intelligent controller, the soft component numbers of the input relay and output relay are all assigned in octal number.

Since [8, 9] does not exist in the octal number, press [0 ~ 7, 10 ~ 17 ... 70 ~ 77, 100 ~ 107] ascending order.

5. BCD: BINARY CODE DECIMAL

Use a 4-bit binary number to represent the 10 digits from 0 to 9 in a 1-digit decimal number.

Suitable for BCD output type digital switch and seven-segment display control.

6. Real Numbers (Floating Point Data)

HC10 intelligent controller has a floating-point arithmetic function that can perform high-precision operations.

Use binary floating point numbers (real numbers) for floating point operations and use decimal floating point numbers (real numbers) for monitoring.

Chapter 2 Use and Function of Soft Components

In this chapter, the use and function of the various soft components used in the intelligent controller and built-in input and output relays, auxiliary relays, status, counters, data registers, etc. are explained.

2.1 Soft Component Number Lists

The number of soft components is shown in the table below.

Soft Component	Content			Reference
Input and Output Relay				
Input relay	X000 ~ X367	248 point	The number of soft components is octal number Input and output totals 496 points	2.2
Output relay	Y000 ~ Y367	248 point		2.3
Auxiliary Relay				
For general [variable]	M0 ~ M499	500 point	The hold/non-hold setting can be changed by parameters	2.4
For maintenace [variable]	M500 ~ M1023	524 point		
For maintenace [fixed]	M1024 ~ M7679	6656 point		
For special	M8000 ~ M8511	512 point		
Status				
Initialization state (for general [variable])	S0 ~ S9	10 point	The hold/non-hold setting can be changed by parameters	2.5
For general [variable]	S10 ~ S499	490 point		
For maintenace [variable]	S500 ~ S899	400 point		
For signal alarms (for maintenace [variable])	S900 ~ S999	100 point		
For maintenace [fixed]	S1000 ~ S4095	3096 point		
Timer (ON Delay Timer)				
100ms	T0 ~ T191	192 point	0.1 ~ 3276.7s	2.7
100ms [for subroutine, interrupt subroutine]	T192 ~ T199	8 point	0.1 ~ 3276.7s	
10ms	T200 ~ T245	46 point	0.01 ~ 327.67s	
1ms cumulative type	T246 ~ T249	4 point	0.001 ~ 32.767s	
100ms cumulative type	T250 ~ T255	6 point	0.1 ~ 3276.7s	
1ms	T256 ~ T511	256 point	0.001 ~ 32.767s	
High Speed Counter				
Single phase single count input Dual direction (32 bit)	C235, C236 C237[1], C238[1]	The hold/non-hold setting can be changed by parameters, -2,147,483,648 ~ +2,147,483,647 counter Single-phase: 100kHz (4 pcs) Two-phase: 50kHz (2 pcs)		2.8
Single-phase double count input Dual direction (32 bit)	C246, C248[1]			
Two-phase double counting input Dual direction (32 bit)	C251, C253[1]			
Data Register (Used in Pairs is 32 Bits)				
For general (16 bits) [variable]	D0 ~ D199	200 point	The hold/non-hold setting can be changed by parameters	2.8
For maintenance (16 bits) [variable]	D200 ~ D511	312 point		
For maintenance (16 bits) [fixed]	D512 ~ D4999	4488 point		
For general (16 bits) [fixed]	D5000 ~ D7999	3000 point		
For special (16 bits)	D8000 ~ D8511	512 point		
For address change (16 bits)	V0 ~ V7, Z0 ~ Z7	16 point		

Soft Component	Content			Reference
Pointer				
For JUMP, CALL branch	P0 ~ P4095	4096 point	For CJ and CALL instruction	2.12
Input interrupt Input delay interrupt	I0□□ ~ I5□□	6 point		
Timer interrupt	I6□□ ~ I8□□	3 point		
Counter interrupt	I010 ~ I060	6 point	For HSCS instruction	
Nesting				
For master control	N0 ~ N7	8 point	For MC instruction	
Constant				
Decimal number (K)	16 phase	-32,768 ~ +32,767		2.13
	32 phase	-2,147,483,648 ~ +2,147,483,647		
Hexadecimal number (H)	16 phase	0000 ~ FFFF		
	32 phase	00000000 ~ FFFFFFFF		
Real number (E)	32 phase	-1.0×2^{128} ~ -1.0×2^{-126}, 0, 1.0×2^{-126} ~ -1.0×2^{128} Can be expressed in decimal and exponential form		

2.2 Input Relay [X]

The component representing the external input signal state of the intelligent controller detects the external signal state through the X port. 0 means the external signal is open, 1 means the external signal is closed. The state of the input relay cannot be modified by the program command method, and the contact signal (normally open type, normally closed type) can be used indefinitely in the user program.

The relay signals are identified by X0, X1... X7, X10, X11 and other symbols, and the serial numbers are numbered in octal.

The controller's counter signal, external interrupt signal, pulse capture and other functions are input through X0 ~ X7 ports.

2.3 Output Relay [Y]

The soft component directly connected to the hardware port of the external user control device is logically corresponding to the physical output port of the intelligent controller. The intelligent controller transmits the component status of the Y relay to the smart each time the user program is scanned. On the hardware port of the controller, 0 means the output port is open; 1 means the output port is closed.

Y relay numbers are identified by symbols such as Y0, Y1... Y7, Y10, Y11, ... etc. The serial numbers are numbered in octal. Y relay components can be used indefinitely in the user program.

Y0 ~ Y3 can set high-speed pulse output function.

2.4 Auxiliary Relay [M]

The intermediate variables in the execution of the user program, like the auxiliary relays in the actual electronic control system, are used for the transmission of status information.

It is also possible to use a plurality of M variables as word variables, and the M variables are not directly related to the external port, but may be copied to M by a program statement, or may be associated with the outside world by copying M to Y, a M variable can be used indefinitely.

The auxiliary relay M is identified by symbols such as M0, M... M8511, and the serial number is numbered in decimal. The variable above M8000 is a system-specific variable for the interaction between the intelligent controller user program and the system state; Some M variables also have power-down save feature.

There are a large number of special auxiliary relays in the intelligent controller (see Chapter 10 Special Component Description). These special auxiliary relays have their own specific functions and can be divided into the following two categories:

- The contact-utilized special auxiliary relay automatically drives the coil for the intelligent controller system. The user program can only be read and used, such as:
  M8000: Run monitor (running during operation), often used before instructions that require a drive signal.
  M8002: Initial pulse (only momentarily turned on at the beginning of the run), often used to execute an initialization command only once.
  M8012: 100ms clock pulse, used to generate a fixed interval flip signal.
- Coil-driven special auxiliary relay, which is used to drive the coil for the user program to control the working status and execution mode of the intelligent controller, such as:
  M8033: Keep output when stopped.
  M8034: Output is completely banned.
  M8039: Constant scanning.

Please note that there are two cases where the driver is valid and the END instruction is valid. The user cannot use special auxiliary relays that have not been defined yet.

2.5 Status Relay [S]

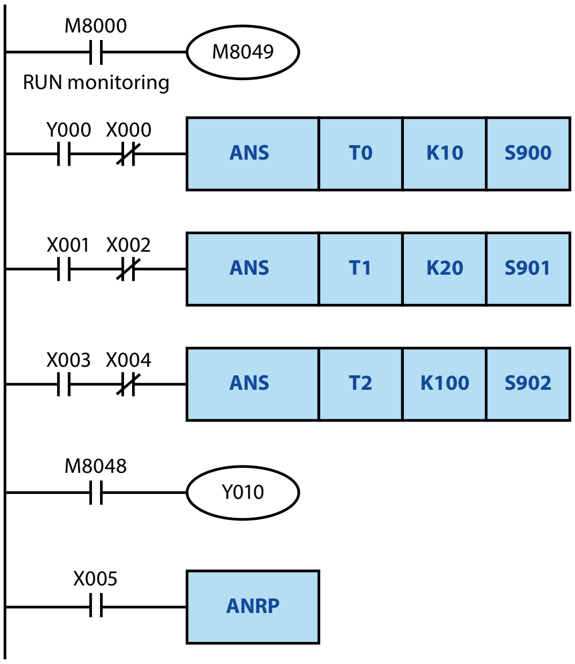
The state S variable is identified by symbols such as S0, S1... S4095, and the serial number is numbered in decimal. The partial S variable has a power-down save function.

The status relay S is used for the design and execution processing of the step program. The STL step instruction is used to control the shift of the step state S, which simplifies the programming design. If the STL programming method is not used, S can be regarded as an ordinary bit element.

In addition, S900 ~ S999 is the status of the signal alarm, and can also be used as an output for diagnosing external faults.

For example, the external fault diagnosis loop shown in the figure below is created. After monitoring the contents of the special data register D8049, the Min. number of the operating states in S900 ~ S999 is displayed.

When multiple faults occur, the next fault number can be known by eliminating the lowest numbered fault.



After the special auxiliary relay M8049 is driven, the monitoring becomes effective.

After driving forward end output Y000, if it is detected that the forward end X000 operates within 1 second, S900 operates.

If the upper limit X001 and the lower limit X002 do not operate simultaneously for more than 2 seconds, S901 operates.

In machines with a cycle time of less than 10 seconds, when input X003 in the continuous operation mode is ON, if the action switch X004 does not operate in one cycle of the machine, S902 operates.

When any of S900 ~ S999 is ON, the special auxiliary relay M8048 is activated, and the fault display output Y010 is activated.

You can use the reset button X005 to turn off the operation status caused by the external fault diagnosis program. Each time X005 is turned on, it will be reset in order from the lower number operation status.

When the special auxiliary relay M8049 is not driven, the power failure hold (hold) state is the same as the normal state and can be used in the sequence program.

2.6 Bit Soft Components Number Specification [Kn□□]

X, Y, M, and S are bit soft components and can be processed by using KnXm, KnYm, KnMm, and KnSm.

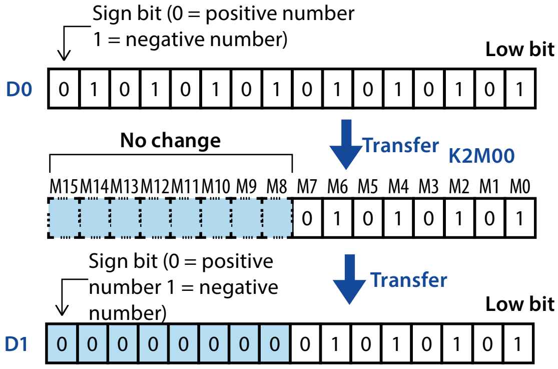
- N refers to the number of bits, one N is 4 bits. Kn□□ can be expressed in single words (n = 1 ~ 4) and double words (n = 1 ~ 8).
- M refers to the starting code of the bit software (X, Y, M, S).

For example, K2M0, refers to the 8-bit data of the combination of M0 ~ M7.

After transferring 16-bit data to K1M0 to K3M0, the upper part of the data length is not transmitted. The case of 32-bit data is the same.

In the 16-bit (or 32-bit) operation, when the bit number of K1 ~ K3 (or K ~ K7) is specified for the bit soft components, the insufficient high bit is always regarded as 0, so the positive number is always processed, as shown in the right figure.

The number of the specified bit soft components can be arbitrary as long as there is no special restriction, but it is recommended to set the lowest bit number to 0 in the case of X, Y (specify X000, X010, X020...Y000, Y010, Y020...etc.).



In the case of M, S, the most ideal is a multiple of 8, but in order to avoid confusion, it is recommended to set it to M0, M10, M20, etc.

2.7 Timer [T]

Used to complete the timing function. Each timer contains coils, contacts, and count registers.

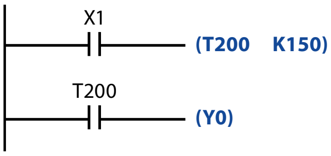
- When the timer coil is "powered" (power flow is active), the timer starts counting. If the timer value reaches the preset time value, its contact action, a contact (NO contact) is closed, b contact (NC contact) disconnected.
- If the coil is "de-energized" (the flow is invalid), the contact of the timer returns to the initial state and the timer value is automatically cleared.
- Some timers also have a cumulative feature. When the condition is broken (the flow can be invalid), the timer maintains the current state and needs to be reset with the RST instruction.

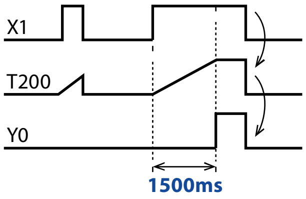
The timer T is identified by symbols such as T0, T1 ... T511, and the serial number is numbered in decimal. The timer has different timing steps, such as 1ms, 10ms, 100ms, etc., and some have power-down retention characteristics.

- There is no timer number used as a timer, and it can also be used as a data register for value storage.
- The timer accumulates the 1ms, 10ms, 100ms and other clock pulses in the intelligent controller. When the timing reaches the set value, the output contacts can only be activated when the coil command or END command is executed.
- The constant (K) in the program memory is used as the set value, and can also be indirectly specified by the contents of the data register (D). Note that the content of D must be set before starting the timer. When the count starts, the data of D changes will only take effect the next time the timing is started.
- From the start of the coil driving the timer to the contact action of the timer, the possible timing length description: The longest case is (T+T0+a).
  T is the set timing time; T0 is the program scan execution time; A is the timer's timing step. The shortest case is (T-a).
  If the timer's contact command is before the coil command, the longest timing length is (T+2T0).
- Using the b-contact of the timer, the output signal of the time-delayed, self-oscillating oscillation can be realized.
- The intelligent controller also provides special timer commands such as TTMR, STMR, etc. Please refer to the description of the corresponding instructions.

Examples:

The ordinary timer T200 is a counter with a step size of 10ms, and the actual action delay is 150 × 10ms = 1500ms, which is 1.50s. The action principle is shown in the figure on the right.





2.8 Counter [C]

The counters are identified by C0, C1, ... C255, and are sequentially numbered in decimal numbers to complete the counting function. Each counter contains a coil, a contact, and a count data value register, each time the drive signal of the counter coil is turned from OFF to ON, the counter count value is increased or decreased by 1.

If the count value reaches the preset value, its contact action, a contact (NO contact) is closed, b contact (NC contact) is open; If the timing value is cleared, the output a contact is disconnected, b contact (NC contact) is closed.

Some counters have the characteristics of power-down maintenance, accumulation, etc., and maintain the value before power-off after power-on.

The counter can be divided into a 16-bit counter and a 32-bit counter according to the length of the count data register. The 32-bit counter can also be divided into an ordinary counter and a high-speed counter according to functions. The characteristics of the 16-bit counter and the 32-bit counter are as follows. Switching, and counting range, etc. are used separately.

Project	16-bit Counter	32-bit Counter
Counting direction	Count up	Increase/decrease count can be switched
Setting value	1 ~ 32,767	-2,147,483,648 ~ +2,147,483,647
Designation of the setting value	Constant K or data register	Same as left, but the data registers need to be paired (2)
Current value change	The count value does not change after it arrives	After the count value is reached, it still changes (ring count)
Output contact	Keep the action after the count value	Hold when counting up, reset when counting down
Reset action	When the RST instruction is executed, the current value of the counter is 0, and the output contact is also reset	
Current value register	16 bits	32 bits

16-bit Counter

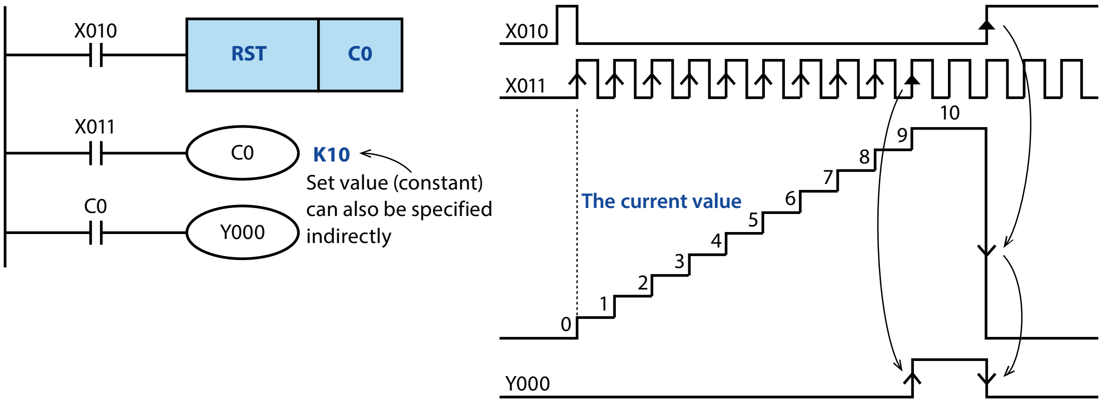
The setting value of the 16-bit binary increment counter is valid in the range of K1 ~ K32, 767 (decimal constant). The operation of K0 is the same as K1, and the output contact operation is performed at the first counting. In the case of a general counter, if the power of the intelligent controller is turned off, the count value will be cleared; However, in the case of the power failure holding counter, the count value before the power failure will be maintained, and the power can continue to count up on the previous value after the power is turned on again.

16-bit Counter Application Example

By counting input X011, the current value of the counter will increase each time the C0 coil is driven, and the output contact will be actuated when the coil command is executed for the 10th time.

Thereafter, even if the count input X011 is active, the current value of the counter does not change.

- If input reset X010 is ON, when the RST instruction is executed, the current value of the counter becomes 0, and the output contact is also reset.



As the current value of the counter, in addition to the above-mentioned constant K, it can also be specified by the data register number.

32-bit Up/Down Counter

The setting value of the 32-bit binary increment/decrement counter is valid in the range of -2,147,483,648 ~ +2,147,483,647 (decimal constant). The direction of up/down counting can be specified using the auxiliary relays M8200 to M8234.

- For C□□□, driving M8□□□ (ON) is the down counter, and not driving (OFF) is the up counter.

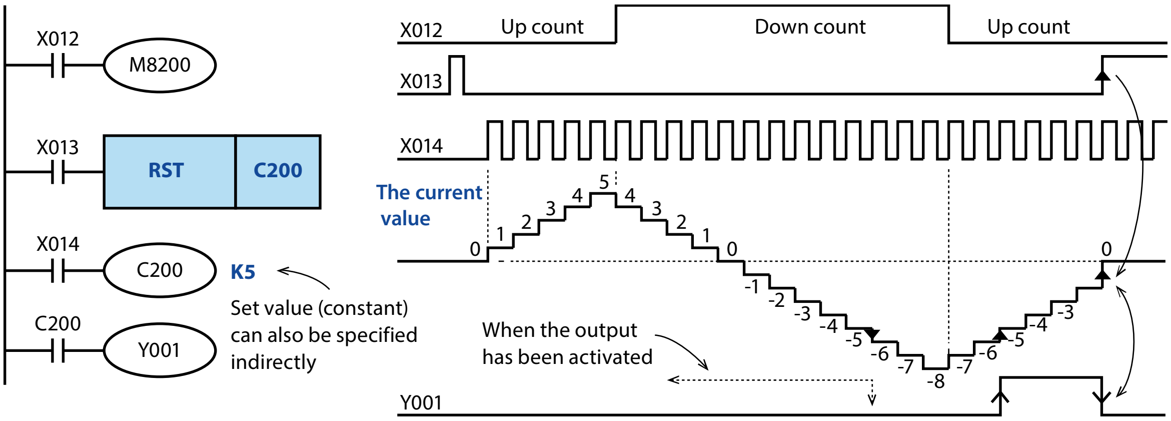
Counter Number	Switch Direction	Counter Number	Switch Direction	Counter Number	Switch Direction	Counter Number	Switch Direction
C200	M8200	C209	M8209	C218	M8218	C227	M8227
C201	M8201	C210	M8210	C219	M8219	C228	M8228
C202	M8202	C211	M8211	C220	M8220	C229	M8229
C203	M8203	C212	M8212	C221	M8221	C230	M8230
C204	M8204	C213	M8213	C222	M8222	C231	M8231
C205	M8205	C214	M8214	C223	M8223	C232	M8232
C206	M8206	C215	M8215	C224	M8224	C233	M8233
C207	M8207	C216	M8216	C225	M8225	C234	M8234
C208	M8208	C217	M8217	C226	M8226		

According to the constant K or the content of the data register D, the setting value can use positive and negative values.

32-bit Calculator Example

When using the count input X014 to drive the C200 coil, it can count up or down.

When the current value of the counter is increased from "-6" to "-5", the output contact is reset when it is reduced from "-5" to "-6".

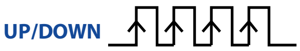
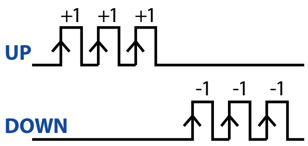
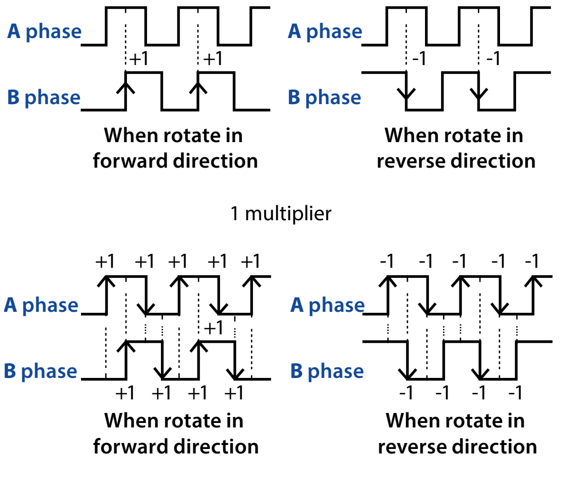


- The increase or decrease of the current value is independent of the action of the output contact. If it is incremented from 2,147,483,647, it becomes -2,147,483,648. Similarly, if it starts counting down from -2,147,483,648, it becomes 2,147,483,647 (the action like this is called ring count).
- If the reset input X013 is ON, the RST instruction is executed, and the current value of the counter becomes 0, and the output contact is also reset.
- In the case of power failure maintenance, the current value of the counter and the action and reset state of the output contact will be maintained by power failure.
- A 32-bit counter can also be used as a 32-bit data register. However, a 32-bit counter cannot be a target soft component in a 16-bit application instruction.
- When a data exceeding the set value is written to the current value register using the DMOV instruction, etc., when there is a next count input, the counter continues to count and the contact does not change.

High Speed Counter

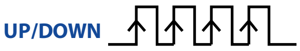
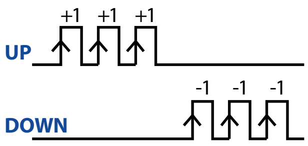
The high-speed counter 32-bit counter number C246 ~ C250 is a high-speed counter, the high-speed counter is used to measure the special counter corresponding to the high-speed pulse signal received by the X terminal, independent of the scan cycle.

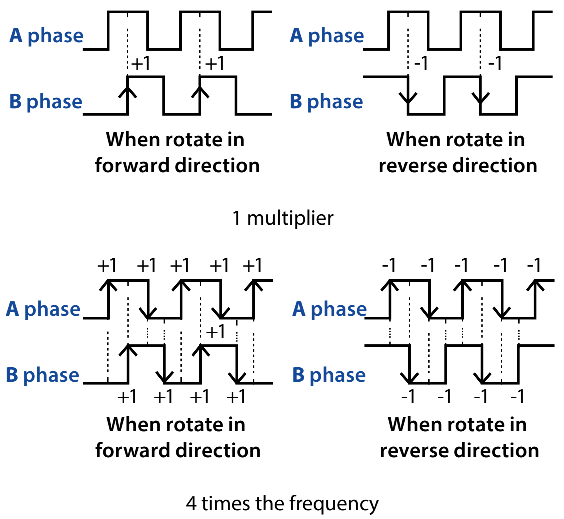
The high-speed counters supported by HC10 are shown in the following table.

Counter Number	X Terminal	Counter Type	Input Signal Form	Counting Direction
C235	X0	Single phase single count input		Increase or decrease countimg by 8235 ~ M8238. • ON: Count down • OFF: Count up
C236	X1			
C237[1]	X2			
C238[1]	X3			
C246	X0 UP X1 DOWN	Single phase double count input		X0/X2 is incremented, and X1/X3 is counted down. The counting direction is displayed by M8246/ M8248. • ON: Count down • OFF: Count up
C248[1]	X2 UP X3 DOWN			
C251	X0 A phase X1 B phase	Two-phase double counting input	 A phase +1 +1 B phase When rotate in forward direction A phase -1 -1 B phase When rotate in reverse direction 1 multiplier +1 +1 +1 +1 +1 A phase +1 B phase +1 +1 +1 +1 When rotate in forward direction -1 -1 -1 -1 -1 A phase B phase -1 -1 -1 -1 When rotate in reverse direction 4 times the frequency	According to the input state change of phase A/ phase B, it automatically increments or counts down. The counting direction is displayed by M8251/M8253. • ON: Count down • OFF: Count up M8198/M8199 is used to switch 1x/4x count. • ON: 4 times the frequency • OFF: 1 multiplier
C253[1]	X2 A phase X3 B phase			

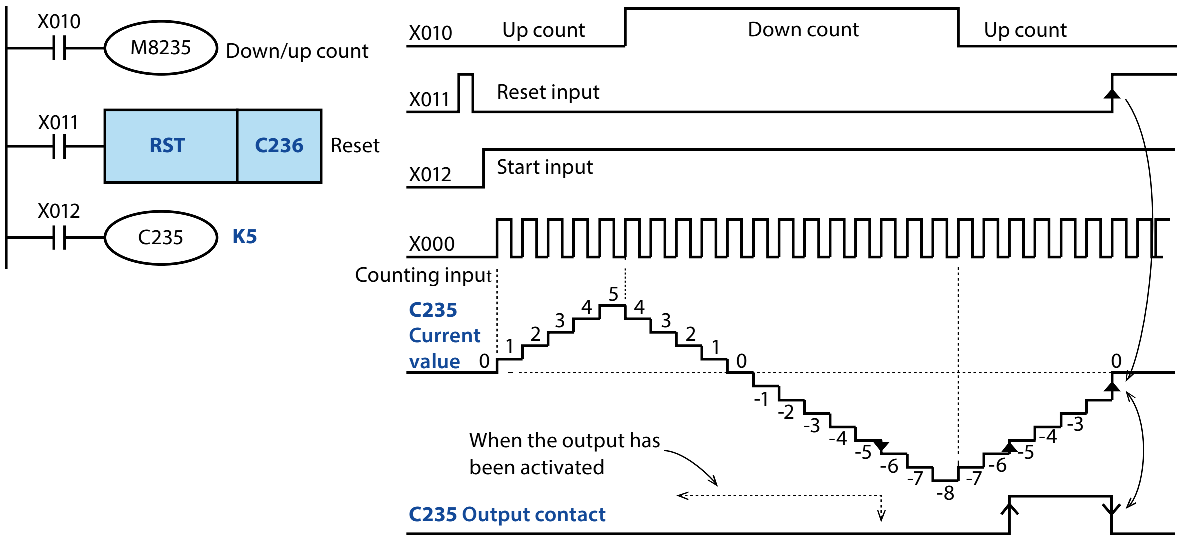
(1): HC10-M0808L4-C3 and etc. do not have this high-speed counter.

The high-speed counters supported by HC10-M0808R-C3-AB are shown in the following table.

Counter Number	X Terminal	Counter Type	Input Signal Form	Counting Direction
C235	X0	Single phase single count input		Increase or decrease countimg by M8235 ~ M8238. • ON: Count down • OFF: Count up
C236	X2			
C237	X4			
C238	X6			
C246	X0 UP X2 DOWN	Single phase double count input		X0/X4 is incremented, and X2/X6 is counted down. The counting direction is displayed by M8246/ M8248. • ON: Count down • OFF: Count up
C248	X4 UP X6 DOWN			

Counter Number	X Terminal	Counter Type	Input Signal Form	Counting Direction
C251	X0 A phase X1 B phase	Two-phase double counting input	 A phase B phase +1 +1 When rotate in forward direction A phase B phase -1 -1 When rotate in reverse direction 1 multiplier +1 +1 +1 +1 +1 +1 +1 +1 +1 When rotate in forward direction -1 -1 -1 -1 -1 -1 -1 -1 -1 When rotate in reverse direction 4 times the frequency	According to the input state change of phase A/ phase B, it automatically increments or counts down. The counting direction is displayed by M8251 ~ M8254. • ON: Count down • OFF: Count up M8196 ~ M8199 are used to switch 1x/4x count. • ON: 4 times the frequency • OFF: 1 multiplier
C252	X2 A phase X3 B phase			
C253	X4 A phase X5 B phase			
C254	X6 A phase X7 B phase			

Example 1: Single-phase Single-count Input



The C235 action is as shown above. When X012 is ON, the input X000 is turned OFF→ON.

- When the X011 bit is ON, the RST instruction is executed and C235 will be reset.
- The counters C235 to C236 are changed between decrement/increment by the ON/OFF of M8235 ~ M8236.

According to the count input X000, C235 counts up or down through the terminal.

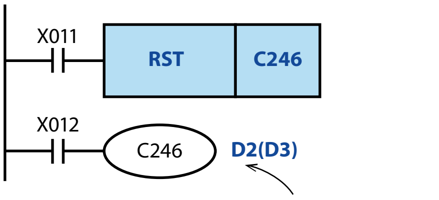
- The output contact is set when the current value is increased from "-6" to "-5". The output contact is reset when the current value is reduced from "-5" to "-6".
- The increase or decrease of the current value is independent of the action of the output contact. If it is incremented from 2,147,483,647, it becomes -2,147,483,648. Similarly, if -2,147,483,648 starts counting down and becomes 2,147,483,647 (this action becomes a ring count).
- The reset input X011 bit turns ON and the RST instruction is executed. At this time, the current value of the counter becomes 0, and the output contact is also reset.
- In the high-speed counter for power failure hold, even if the power is turned off, the current value of the counter and the action and reset state of the output contact are maintained.

Example 2: Single-phase Double-count Input

That is, a 32-bit up/down binary counter, the action of the output contact corresponding to the current value is the same as the high-speed counter of the single-phase single-count input described above.

When X012 is ON, C246 is incremented if input X000 is turned from OFF to ON. If input X001 is OFF→ON, it is counted down.

- The C246's up/down action can be monitored by the M8246's ON/OFF action.
  - ON: Count down. OFF: Count up.



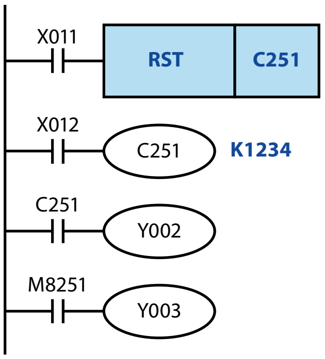
The setting value is (D3, D2)

Example 3: Two-phase Double Count Input

That is, a 32-bit up/down binary counter, the action of the output contact corresponding to the current value is the same as the high-speed counter of the single-phase single-count input described above.

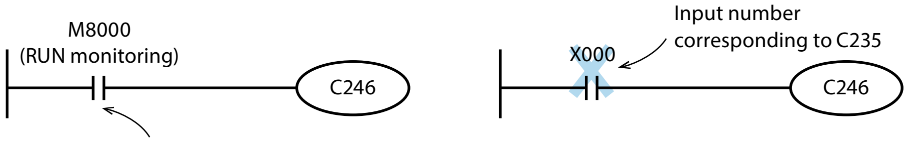
When X012 is ON, C251 counts the actions of input X000 (A phase) and X001 (B phase) through the terminal.

- When X011 is ON, the RST instruction is executed and C251 will be reset.
- When the count value reaches the set value, the Y002 condition is turned ON.
- Y003 is ON (minus) and OFF (increase) depending on the counting direction.



Note for Use

For the coil drive contacts of the high-speed counter, use a contact of the ON type at high speed.



Input number corresponding to C235

Please use the contact that is always ON when programming

High-speed counter does not count correctly when the number of input relays for counting is specified

- If the high-speed counter is operated by a device with a contact such as an analog switch, the counter may have a counting error due to the vibration of the switch. Please note.
- The input filter of the basic unit input terminal used in the high-speed counter is set by D8250 and D8251, and has no connection with the filter value set by the common terminal.
- When the input terminal is used for the high-speed counter, it can no longer be repeatedly designated as a high-speed counter, input interrupt, pulse capture, and general-purpose input functions. Do not reuse the input terminals.
- All high-speed counters, the output contact will be active in the current value = setting value.
- The count can be started/stopped by turning the output coil (OUT C***) of the high-speed counter ON/OFF, but please use this output coil for programming in the main program. If this coil is programmed in a step ladder and in a subroutine or interrupt subroutine, counting and stopping cannot be performed until these step ladders and subroutines are executed.
- When the high-speed counter is reset using the RST instruction, the high-speed counter cannot be counted until the drive of the RST instruction is turned OFF.

2.9 Data Register [D]

The data register is a soft component for storing numerical data, all of which are 16-bit data (the most significant bit is a positive or negative sign). By combining two data registers, 32-bit (the most significant sign) can be saved.

Data registers can be divided into general use, maintenance use and special use, in which D0 ~ D511 can change the scope of general use and maintenance use by setting parameters.

For General Use

When data is successfully written to the data register, the data in this register will remain unchanged as long as it is not rewritten.

When the intelligent controller changes from RUN to STOP or change from STOP to RUN, all data will be cleared.

For Maintenance Use

The data register of the power failure maintanence area still keeps the data unchanged after the intelligent controller changes from RUN to STOP or power failure.

When using the dedicated data register for power failure as general use, use the RST or ZRST instruction to set the reset ladder in the beginning of the program.

For Special Use

Special registers are used to write data for a specific purpose, or data has been written to a specific content by the system.

The data in some special registers is initialized when the smart controller is powered up.

For the number and purpose of special registers, please refer to the list of special soft components.

2.10 Bit Designation of Word Soft Components [D.b]

D (Data Register) can operate bit by bit in the way of D.b and use it as bit data.

When specifying the bit of word soft component, set it with the word soft component number and bit number.

- Word soft component: Data register or special register. Bit number: 0 ~ F (hexadecimal).
- For example: D0.0 indicates the bit data of data register D0 numbered 0, and D0.F indicates the bit data of data register D0 numbered F.

Index modification cannot be performed in the soft component number and bit number.

2.11 Index Register [V, Z]

The index register is a special register that can change the number and value of the soft component in the program by using a combination of other soft component numbers and values in the operand of the application instruction, in addition to the same method as the data register.

The index registers [V, Z] are numbered V0 ~ V7, and Z0 ~ Z7 have 16 16-bit registers.

The soft components that can be modified, the extremely modified content is as follows.

Decimal Soft Component • Value: M, S, T, C, D, R, KnM, KnS, P, K

For example, when V0 = K5, when D20V0 is executed, the execution number of the soft component number D25 (D20+5) is executed.

In addition, the constant can be modified. When K30V0 is specified, the executed instruction is the decimal value K35 (30+5).

Octal Number Soft Component: X, Y, KnX, KnY

For example, Z1 = K8, when X0Z1 is executed, the execution number of the soft component number is X10 (X0+8: octal addition). When the soft component with the soft component number is octal is indexed, the content of Z and Z will be converted into octal numbers and then added.

Therefore, assuming Z1 = K10 and X0Z1 is designated as X12, be sure to note that this is not X10.

Hexadecimal Value: H

For example, V5 = K30, when the constant H30V5 is specified, it is regarded as H4E (30H+K30).

In addition, V5 = H30, when the constant H30V5 is specified, it is regarded as H60 (30H+30H).

2.12 Pointer [P], [I]

The numbers of pointers (P) and (I) are shown in the table below (numbers are assigned in decimal numbers).

In addition, when using the input interrupt pointer, the input number assigned to the pointer cannot use the same input range [high-speed counter].

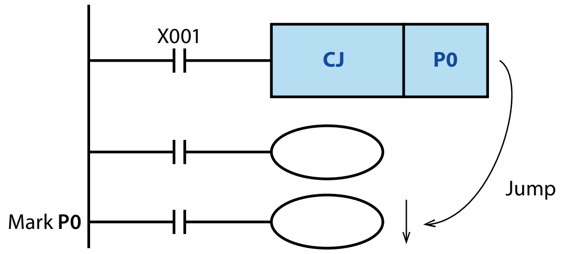
Interrupt pointer is used with application instruction IRET (FN 03) interrupt return, EI (FN 04) allow interrupt, and DI (FN 05) prohibit interrupt.

For Branch		Input Delay Interruption	For Timer Interruption	For Counter Interruption
	For END Jump			
P0 ~ P62, P64 ~ P4095 [4095 points]	P63 [1 point]	I00□ (X000) I10□ (X001) I20□ (X002) I30□ (X003) I40□ (X004) I50□ (X005) [6 points]	I6□□ I7□□ I8□□ [3 points]	I010 I020 I030 I040 I050 I060 [6 points]

Branch Pointer: 4096

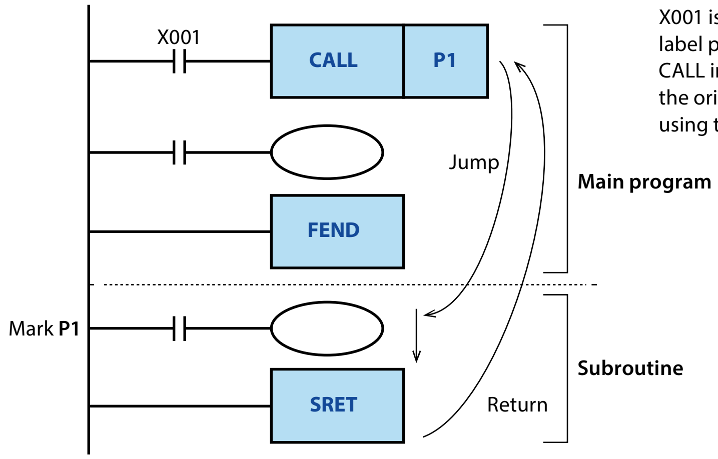
The functions and actions of the branch pointer are shown below.

CJ conditional jump



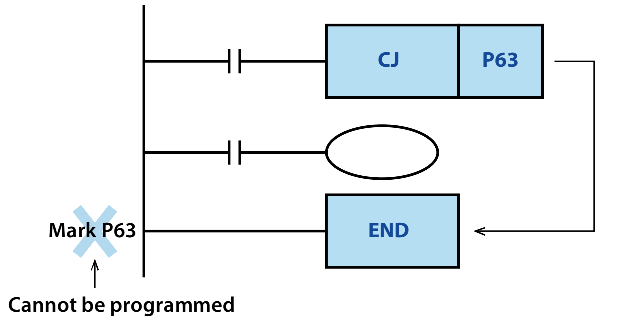
X001 is ON, it will jump to the mark position of CJ instruction and execute the following program.

CALL subroutine call



X001 is ON, the subroutine of the label position specified by the CALL instruction is executed, and the original position is returned using the SRET instruction.

The function of END jump pointer P63



P63 is a special pointer to jump to the END step when using the CJ instruction. Therefore, when P63 is programmed as a label, the program will be wrong. Please note.

In addition, these pointers are used in combination with application instructions, so please refer to the instructions for detailed instructions.

Input Interrupt (Delayed Interrupt) with Pointer: 6 Points

The input signal from a specific input number can be received without being affected by the intelligent controller's calculation cycle. The input signal is triggered to execute the interrupt subroutine.

Since the input interrupt can process signals shorter than the calculation cycle, it can be used as a priority processing or short-time pulse processing control in the sequence control process.

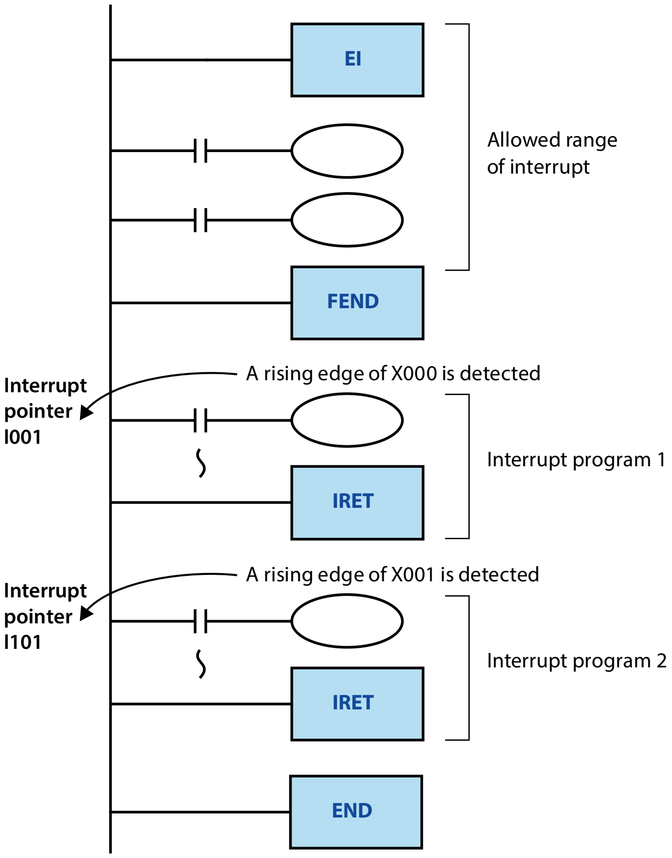
Input	Input Interrupt Pointer		Interrupt Banned Flag
	Rising Edge Interrupt	Falling Edge Interrupt	
X000	I001	I000	M8050[1)]
X001	I101	I100	M8051[1)]
X002	I201	I200	M8052[1)]
X003	I301	I300	M8053[1)]
X004	I401	I400	M8054[1)]
X005	I501	I500	M8055[1)]
1): Clear from RUN→STOP.			

Note:

Input X000 ~ X005 for high speed counter, input interrupt, pulse capture and general purpose input. Therefore, do not reuse the input terminals.

For Example

When using the input interrupt pointer [I001], since X000 is occupied, [C235, C246, C251], [input interrupt pointer I000], and [pulse capture contact M8170] cannot be used.



PLC are normally in the state where interrupts are disabled. After using the EI instruction to enable interruption, X000 or X001 is ON during the scanning procedure, executes the interrupt subroutine 1 or 2, and then returns to the main routine through the IRET instruction.

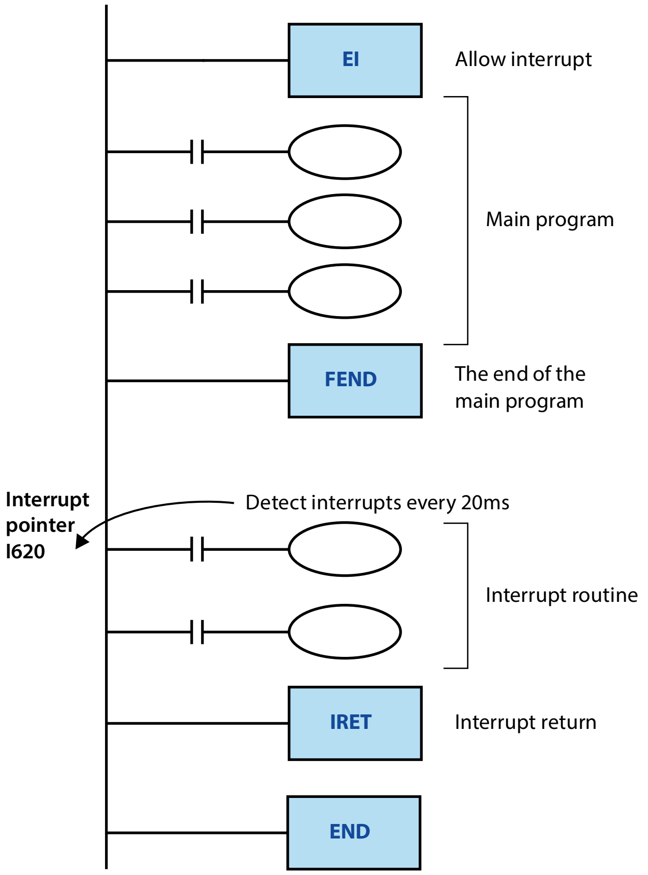
Interrupt pointer (I ***), be sure to place it as a mark after the FND instruction during programming.

Timer Interrupt Pointer: 3 Points

The interrupt subroutine is executed every specified interrupt cycle time (1 to 99ms). It is used in the control that requires cyclic interrupt processing outside the calculation cycle of the intelligent controller.

Input Number	Interrupt Period (ms)	Interrupt Banned Flag
I6□□	In the pointer name □□, enter an integer from 10 to 99. Such as: I610 = customizer interrupt every 10ms	M8056[1]
I7□□		M8057[1]
I8□□		M8058[1]
1): Clear from RUN → STOP.		

For Example



Timer interrupt is valid after EI instruction. In addition, when the disable interval of the timer interrupt is not required, it is not necessary to program DI (interrupt disable instruction).

FEND indicates the end of the main program. Interrupt subroutine must be written after FEND.

Interrupt routines are executed every 20ms. Use the IRET instruction to return to the main program.

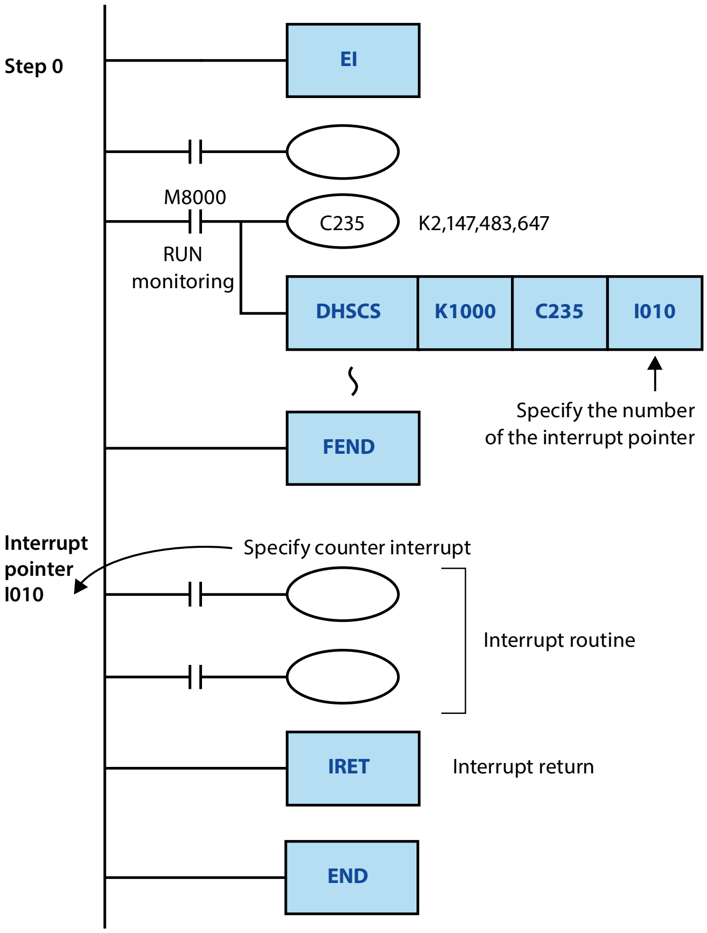
Counter Interrupt Pointer: 6 Points

The interrupt subroutine is executed according to the comparison result of the high-speed counter with the compare set instruction (DHSCS instruction).

Control for prioritizing the counting results using a high speed counter.

Pointer Number	Interrupt Banned Flag	Pointer Label	Interrupt Banned Flag
I010	M8059[1)]	I040	M8059[1)]
I020		I050	
I030		I060	
1): Clear from RUN→STOP.			

For Example



Specify the number of the interrupt pointer

Specify counter interrupt

Interrupt routine

Interrupt return

Allow interrupt after execution of EI instruction and write main program.

Drive the coil of the high-speed counter and specify the interrupt pointer in the DHSCS instruction.

When the current value of C235 changes in 999→1000/1001→1000, execute the interrupt subroutine.
Example of using interrupt program, please refer to the input interrupt above.

2.13 Constant

Constant K (Decimal)

[K] indicate the sign of the decimal integer, which is mainly used to specify the setting value of the timer and counter, or the value in the operand of the application instruction (example: K1234).

The specified range of the decimal constant is as follows.

- When using word data (16 bits): K-32768 ~ K32,767
- When using double word data (32 bits): K-2,147,483,648 ~ K2,147,483,647

Constant H (Hexadecimal)

[H] represent the sign of the hexadecimal number. It is mainly used to specify the value of the operand of the application instruction (example: H1234).

Moreover, when each digit is used in the range of 0 to 9, the status (1 or 0) of each bit is the same as the BCD code, so BCD data can be specified (for example, when H1234 specifies data in BCD, please use 0 to 9. specify the number of digits in the range of hexadecimal numbers).

The setting range of the hexadecimal constant is as follows.

- When using word data (16 bits): H0000 ~ HFFFF (H0000 ~ H9999 for BCD data)
- When using double word data (32 bits): H00000000 ~ HFFFFFFFF (H0 ~ H99, 999, 999 for BCD data)

Constant E (Real Number)

[E] represent the sign of the real number (floating point data), mainly used to specify the value of the operand of the application instruction (eg: E1.234 or E1.234 + 3).

The specified range of real numbers is -1.0×2^{128} ~ -1.0×2^{-126}, 0, 1.0×2^{-126} ~ 1.0×2^{128}.

In the sequence program, the real number can specify "normal representation" and "exponential representation".

- Normal means that the set value is specified. For example, 10.2345 is specified as E10.2345.
- Index means that the set value is specified by (num) × 10^n. For example, 1234 is specified by E1.234 + 3. [+3] of [E1.234 + 3] indicates the n-th power of 10 (+3 is 10^3).

Chapter 3 Basic Sequence Instructions

3.1 Basic Instructions

Instruction Symbol	Function	Operand Type	Operand						
			X0 ~ X377	Y0 ~ Y377	M0 ~ M7679 M8000 ~ M8511	S0 ~ S4095	T0 ~ T511	C0 ~ C255	D0 ~ D8511
Contact Instruction									
LD	Opposite	S, X, Y, M, T. C	●	●	●	●	●	●	
LDI	Reverse	S, X, Y, M, T. C							
LDP	Rising edge of the pulse	S, X, Y, M, T. C							
LDF	Falling edge of the pulse	S, X, Y, M, T. C							
AND	And	S, X, Y, M, T. C	●	●	●	●	●	●	
ANI	And reverse	S, X, Y, M, T. C							
ANDP	And rising edge of pulse	S, X, Y, M, T. C	●	●	●	●	●	●	
ANDF	And falling edge of pulse	S, X, Y, M, T. C							
OR	Or	S, X, Y, M, T. C	●	●	●	●	●	●	
ORI	Or reverse	S, X, Y, M, T. C							
ORP	Or pulse rising edge	S, X, Y, M, T. C	●	●	●	●	●	●	
ORF	Or pulse falling edge	S, X, Y, M, T. C							
Combined Instruction									
ANB	Circuit block and	No	No						
ORB	Circuit block or	No	Participating in the block operation is the computational energy flow of the last two LD (or LDI/LDP/LDF) intervals						
MPS	Store pull stack	No	No						
MRD	Store read stack	No	No						
MPP	Store push stack	No	No						
INV	Reverse	No	No						
MEP	Turn on at rising edge	No	No						
MEF	Turn on at falling edge	No	No						
Output Instruction									
OUT	Output	S, Y, M, T, C		●	●	●	●	●	
SET	Set	S, Y, M		●	●	●			
RST	Reset	S, Y, M, T, C, D		●	●	●	●	●	●
PLS	pulse	Y, M		●	●				
PLF	Pulse at falling edge	Y, M		●	●				
Master Control Instruction									
MC	Master	N0 ~ N7	N0 ~ N7						
MCR	Master reset	N0 ~ N7							
Other Instruction									
NOP	No operation	No	No						
End Instruction									
END	End	No	No						
Pointer Instruction									
P	Pointer	0 ~ 127	P0 ~ P127 • It is used to mark the beginning of the jump address in the main program, where P63 is a dedicated address pointing to END. • It is used to mark the start address of a subroutine. Each subroutine ends with SRET.						
I	Interrupt insert pointer	I101/I201/301, etc.	I00 * ~ I50 *, 6 o'clock, input interrupt pointer; I6 ** ~ I8 **, 3 o'clock, timing interrupt pointer; I010 ~ I060, 6 o'clock, counting interrupt pointer						

3.1.1 LD, LDI Instruction

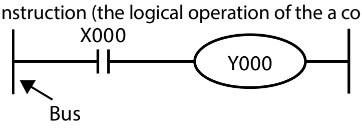
Outline

The LD and LDI instructions are the contacts connected to the bus. After being combined with the ANB instructions described later, they can also be used at the branch starting point.

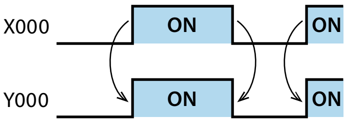
Function and Action Description

LD, LDI Instruction

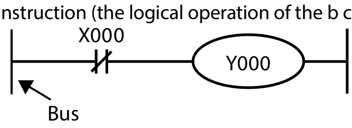
LD instruction (the logical operation of the a contact starts):



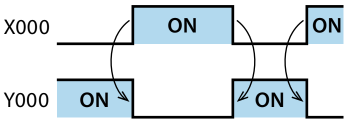
```
0000   LD    X000  ←——  Connected to
0001   OUT   Y000       the bus
```



LDI instruction (the logical operation of the b contact starts):



```
0000   LDI   X000  ←——  Connected to
0001   OUT   Y000       the bus
```

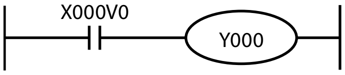


Index Modification

The soft component used in the LD and LDI instructions can be modified with the index register (V, Z).

- Status (S), special auxiliary relay (M), 32-bit counter (C), D.b cannot be modified.
- V0 ~ V7, Z0 ~ Z7 can be used in the index modification.
- When the soft component used is input (X) or output (Y), the value of the index register (V, Z) is converted to an octal number and then added.

Example: When the value of V0 is 10, the LD contact is turned ON/OFF (not turned on) by X012.



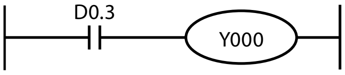
```
0000   LD    X000V0
0003   OUT   Y000
```

Bit Designation of Data Register (D)

Among the soft component used by the LD and LDI instructions, the bits of the data register (D) can be specified.

- When performing bit designation of the data register, enter “.” after the number of the data register (D), and then enter the bit number (0 ~ F).
- The data registers that can be used are only valid for 16 bits.
- Specify the bit number in the order of 0, 1, 2, ... 9, A, B ... F from the low position.

Example: In the example on the right, the third bit of D0 determines the LD contact ON (ON)/OFF (non-conducting).



```
0000   LD    D0.3
0003   OUT   Y000
```

Error

Error	
1	An operation error occurs when the index modification becomes a soft component number that does not actually exist (error code: 6706).

3.1.2 OUT Instruction

Outline

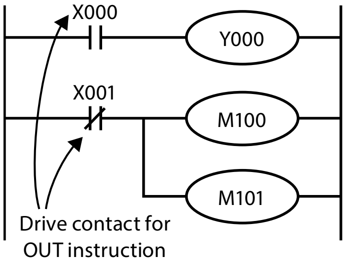
The OUT instruction is a command to coil the output relay (Y), auxiliary relay (M), state (S), timer (T), and counter (C).

Function and Action Description

OUT Instruction

When Using Bit Soft Components:

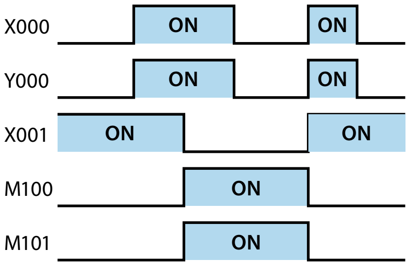
- The soft component written with the OUT instruction performs ON/OFF according to the state of the drive contact. Parallel OUT commands can be used multiple times in succession. As in the following program example, OUT M100 is followed by OUT M101.
- However, when using multiple OUT commands for the same soft component number, it will become a dual output (double coil), please note.



Drive contact for OUT instruction

```
0000  LD   X000
0001  OUT  Y000
0002  LDI  X0001
0003  OUT  M100
0004  OUT  M101
```

Automatically manag program step numbers



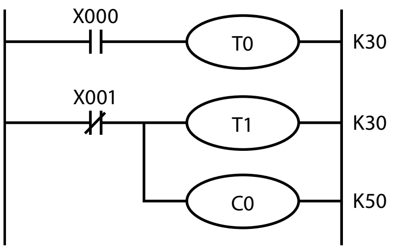
When Using Timers and Counters:

The setting value needs to be added after the OUT command for the timer's timing coil and the counter's counting coil.

The setting value can be specified directly using a decimal number (K) or indirectly using the data register (D).

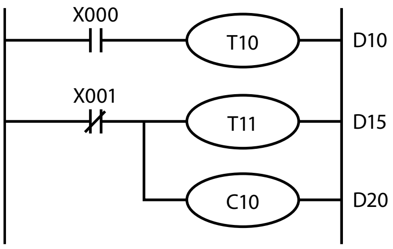
- Directly specify:
  - Set the timer and counter settings in decimal (K).
- Indirect designation:
  - The timer and counter settings can be set in the data register (D). At this time, the current value of the data register (D) is the setting value of the timer.
  - Before driving the timer and counter, the setting value must be written to the data register (D) used as the set value by MOV command, display unit, etc. in advance.

Directly specified



```
0000  LD   X000
0001  OUT  T0
      (SP) K30
0004  LDI  X001
0005  OUT  T1
      (SP) K30
0008  OUT  C0
      (SP) K50
```

Indirectly specified



```
0000  LD   X000
0001  OUT  T10
      (SP) D10
0004  LDI  X001
0005  OUT  T11
      (SP) D15
0008  OUT  C10
      (SP) D20
```

Timer, Counter Setting Range

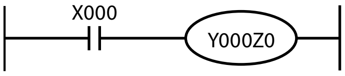
The setting range of the timer and counter setting value and the actual timer constant and the number of program steps of the OUT command (including the set value) are as shown in the table below.

Timer, Counter	Setting Range (The Value of K or the Current Value of D and R)	Actual Set Value	Steps
1ms timer	1 ~ 32,767	0.001 ~ 32.767s	3
10ms timer		0.01 ~ 327.67s	
100ms timer		0.1 ~ 3276.7s	
16-bit counter	1 ~ 32,767	Same as left	3
32-bit counter	-2,147,483,648 ~ +2,147,483,647	Same as right	5

Index Modification

The soft component used in the OUT instruction can be modified with the index register (V, Z).

- Status (S), special auxiliary relay (M), 32-bit counter (C), D.b cannot be modified.
- V0 ~ V7, Z0 ~ Z7 can be used in the index modification.
- When the soft component used is input (X) or output (Y), the value of the index register (V, Z) is converted to an octal number and then added.



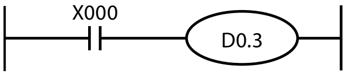
```
0000    LD     X000
0001    OUT    Y000Z0
```

Example: When the value of Z0 is 20, Y024 ON/OFF.

Bit Designation of Data Register (D)

Among the soft components used by the OUT instruction, the bit of the data register (D) can be specified.

- When performing bit designation of the data register, enter “.” after the number of the data register (D), and then enter the bit number (0 ~ F).
- The data registers that can be used are only valid for 16 bits.
- Specify the bit number in the order of 0, 1, 2, ..., 9, A, B, ... F from the low position.



```
0000    LD     X000
0001    OUT    D0.3
```

Example: In the example on the right, the bit3 (b3) of D0 is turned ON/OFF by the ON/OFF of X000.

Note

Note	
1	When special internal relays (M), timers, and counters are used, the program steps are incremented as described in "setting range of timers and counters" above.
2	Do not use the end number of the data register (D) in the 32 counter setting value.

Error

Error	
1	An operation error occurs when the index modification becomes a soft component number that does not actually exist (error code: 6706).

3.1.3 AND, ANI Instruction

Outline

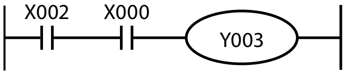
The AND and ANI commands are executed to connect one contact in series. There is no limit to the number of series contacts. This command can be used multiple times in succession.

After the OUT command, the OUT command is used for the other coils through the contacts, which is called the vertical output. As long as the order is correct, such a longitudinal output can be reused multiple times.

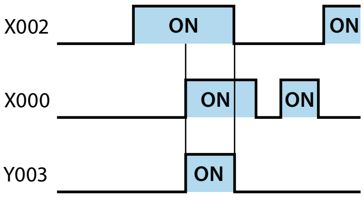
Function and Action Description

AND, ANI Instruction

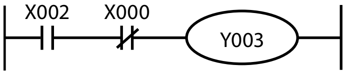
AND instruction (series a contact):



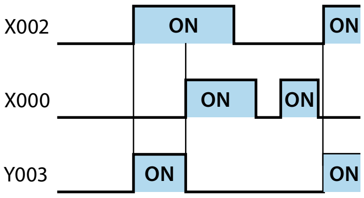
```
0000   LD    X002
0001   AND   X000  <——— Series contact
0002   OUT   Y003
```



ANI instruction (series b contact):



```
0000   LD    X002
0001   ANI   X000  <——— Series contact
0002   OUT   Y003
```

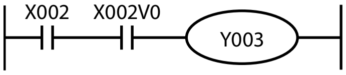


Index Modification

The soft components used in the AND and ANI instructions can be modified with the index register (V, Z).

- Status (S), special auxiliary relay (M), 32-bit counter (C), D.b cannot be modified.
- V0 ~ V7, Z0 ~ Z7 can be used in the index modification.
- When the soft component used is input (X) or output (Y), the value of the index register (V, Z) is converted to an octal number and then added.

Example: When the value of V0 is 8, the AND contact is turned ON/OFF by X012. When only X002 and X012 are ON, Y003 is turned ON.



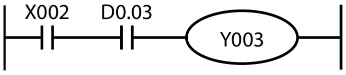
```
0000   LD    X002
0001   AND   X002V0
0004   OUT   Y003
```

Bit Designation of Data Register (D)

The bits of the data register (D) can be specified in the soft components used by the AND and ANI instructions.

- When performing bit designation of the data register, enter "." after the number of the data register (D), and then enter the bit number (0 ~ F).
- The data registers that can be used are only valid for 16 bits.
- Specify the bit number in the order of 0, 1, 2, ... 9, A, B ... F from the low position.

Example: In the example on the right, when the bit3 (b3) of D0 is ON, the AND contact is ON (on). Only when X002 and the bit3 (b3) of D0 are ON, Y003 is turned ON.



```
0000   LD    X002
0001   AND   D0.03
0004   OUT   Y003
```

Error

Error	
1	An operation error occurs when the index modification becomes a soft component number that does not actually exist (error code: 6706).

3.1.4 OR, ORI Instruction

Outline

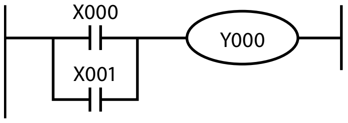
OR and ORI instructions can be used as instructions for connecting one contact in parallel. When two or more contacts are connected in series, when such a series circuit block is connected in parallel with other circuits, the ORB instruction described later is used.

OR and ORI are started from the step of this instruction and connected in parallel with the steps of the previous LD and LDI instructions. The number of parallel connections is unlimited.

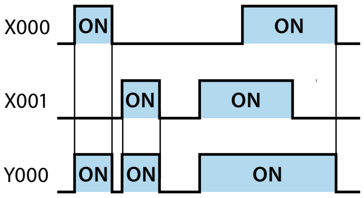
Function and Action Description

OR, ORI Instruction

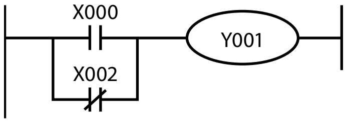
OR instruction (series a contact):



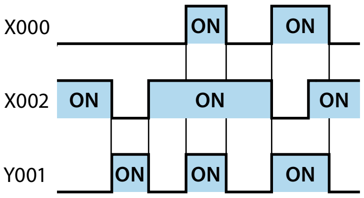
```
0000   LD    X000
0001   OR    X001
0002   OUT   Y000
```



ORI instruction (series b contact):

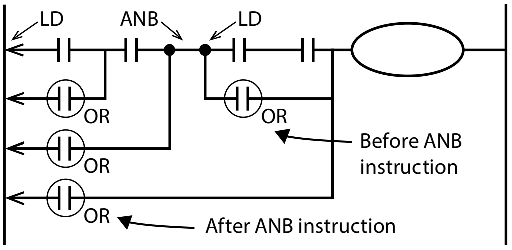


```
0000   LD    X000
0001   ORI   X002
0002   OUT   Y001
```



The relationship of the ANB instructions:

Parallel connection with OR and ORI commands is in principle connected to the previous LD and LDI points, but after the ANB instruction described later, it is connected to the previous LD and LDI points.

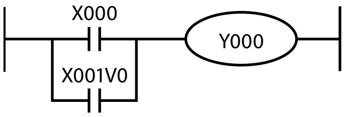


Index Modification

The soft components used in the OR and ORI instructions can be modified with the index register (V, Z).

- Status (S), special auxiliary relay (M), 32-bit counter (C), D.b cannot be modified.
- V0 ~ V7, Z0 ~ Z7 can be used in the index modification.
- When the soft component used is input (X) or output (Y), the value of the index register (V, Z) is converted to an octal number and then added.

Example: When the value of V0 is 10, the OR contact is turned ON/OFF (not turned on) by X013.



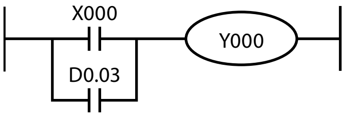
```
0000   LD    X000
0001   OR    X001V0
0004   OUT   Y000
```

Bit Designation of Data Register (D)

Among the soft components used by the OR and ORI instructions, the bits of the data register (D) can be specified.

- When performing bit designation of the data register, enter "." after the number of the data register (D), and then enter the bit number (0 ~ F).
- The data registers that can be used are only valid for 16 bits.
- Specify the bit number in the order of 0, 1, 2 ... 9, A, B ... F from the low position.

Example: In the example on the right, the bit3 (b3) of D0 determines the ON (ON)/OFF (non-conduction) of the OR contact.



```
0000   LD    X000
0001   OR    D0.03
0004   OUT   Y000
```

Error

Error	
1	An operation error occurs when the index modification becomes a soft component number that does not actually exist (error code: 6706).

3.1.5 LDP, LDF, ANDP, ANDF, ORP, ORF Instruction

Outline

The LDP, ANDP, and ORP instructions are contact instructions that detect the rising edge. When the rising edge of the specified bit soft component (from OFF to ON) is turned on, one operation cycle is turned on.

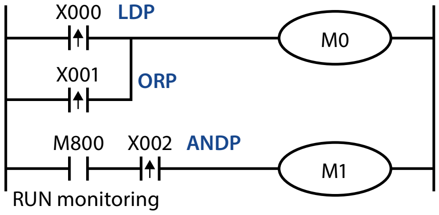
The LDF, ANDF, and ORF instructions are contact instructions that detect the falling edge. When the falling edge of the specified bit soft component (from ON to OFF) is turned on, one operation cycle is turned on.

Function and Action Description

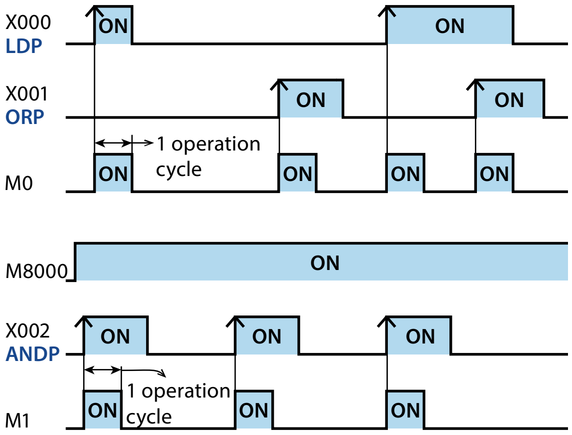
LDP, LDF, ANDP, ANDF, ORP, ORF Instruction

LDP, ANDP, ORP instructions (start of operation when detected rising edge, series connection, parallel connection):

- In the following figure, when X000 ~ X002 is changed from OFF to ON, M0 or M1 is only ON for one calculation cycle.

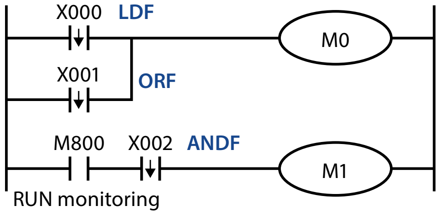


```
0000   LDP    X000
0002   ORP    X001
0004   OUT    M0
0005   LD     M8000
0006   ANDP   X002
0008   OUT    M1
```

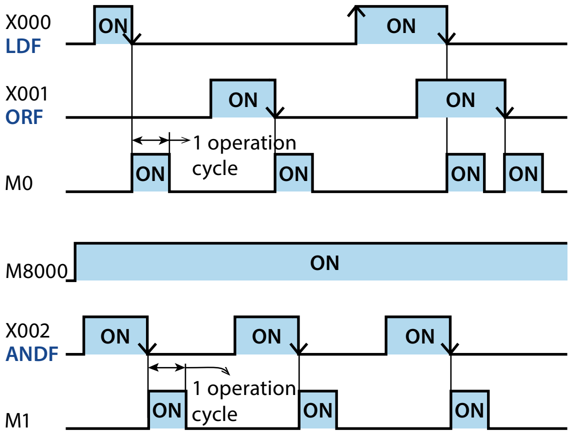


LDF, ANDF, ORF instructions (start of operation when detected falling edge, series connection, parallel connection):

- In the following figure, when X000 ~ X002 is changed from ON to OFF, M0 or M1 is maintained for only one calculation cycle.



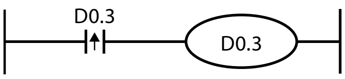
```
0000   LDF    X000
0002   ORF    X001
0004   OUT    M0
0005   LD     M8000
0006   ANDF   X002
0008   OUT    M1
```



Bit Designation of Data Register (D)

For soft components used in LDP, LDF, ANDP, ANDF, ORP, and ORF instructions, the bits of the data register (D) can be specified.

- When performing bit designation of the data register, enter “.” after the number of the data register (D), and then enter the bit number (0 ~ F).
- The data registers that can be used are only valid for 16 bits.
- Specify the bit number in the order of 0, 1, 2 ... 9, A, B ... F from the low position.



```
0000   LDP    D0.3
0003   OUT    Y000
```

Example: In the example on the right, when the bit3 (b3) of D0 is changed from OFF to ON, the LDP contact is ON/OFF.

3.1.6 ORB Instruction

Outline

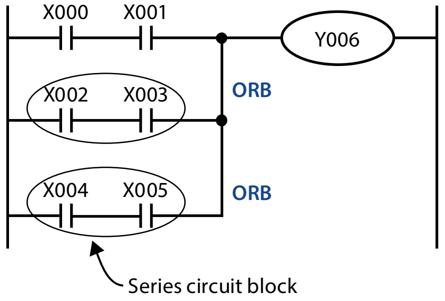
A circuit connected in series by more than two contacts is called a series circuit block.

Function and Action Description

ORB Instruction (Parallel Connection of Circuit Block)

When the series circuit block is connected in parallel, the starting point of the branch uses the LD and LDI instructions, and the end of the branch uses the ORB instruction.

- ORB instruction is the same as ANB instruction described later, and is an independent instruction without a soft component number.
- When there are multiple parallel circuits, use the ORB instruction in each circuit block to connect.



Ideal procedure				Non-ideal procedure			
0000	LD	X000		0000	LD	X000	
0001	AND	X001		0001	AND	X001	
0002	LD	X002		0002	LD	X002	
0003	AND	X003		0003	AND	X003	
0004	ORB		←	0004	LDI	X004	
0005	LDI	X004		0005	AND	X005	
0006	AND	X005		0006	ORB		←
0007	ORB		←	0007	ORB		←
0008	OUT	Y006		0008	OUT	Y006	

Note

Note	
1	There is no limit on the number of parallel circuits connected by ORB instructions.
2	ORB can be used in batches, but LD and LDI instructions can be reused up to eight times.

3.1.7 ANB Instruction

Outline

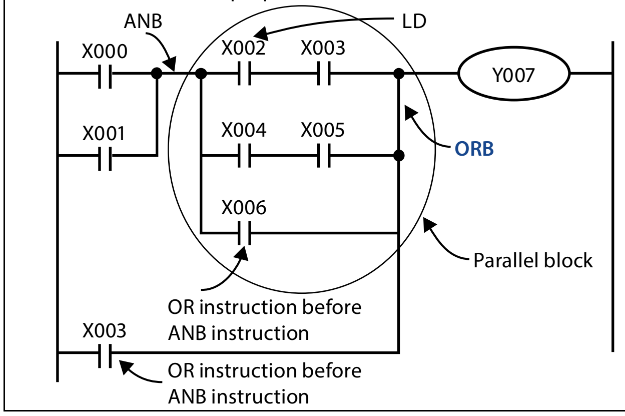
When the branch circuit (parallel circuit block) is connected in series with the previous circuit, ANB instruction is used.

Function and Action Description

ANB Instruction (Series Connection of Circuit Blocks)

The starting point of the branch uses the LD and LDI instructions. After the parallel circuit block ends, ANB instruction can be connected in series with the previous circuit.

- When there are multiple parallel circuits, use ANB instruction for each circuit block to connect.


ANB
LD
X000
X002
X003
Y007
X001
X004
X005
ORB
X006
Parallel block
OR instruction before ANB instruction
X003
OR instruction before ANB instruction

0000	LD	X000	
0001	OR	X001	
0002	LD	X002	**Branch start**
0003	AND	X003	
0004	LDI	X004	
0005	AND	X005	
0006	ORB		**End of parallel block**
0007	OR	X006	
0008	ANB		**Connected in series with the previous circuit**
0009	OR	X003	
0008	OUT	Y007	

Note

Note	
1	There is no limit on the number of ANB instructions used.
2	ANB can be used in batches, but LD and LDI instructions can be reused up to 8 times.

3.1.8 MPS, MRD, MPP Instruction

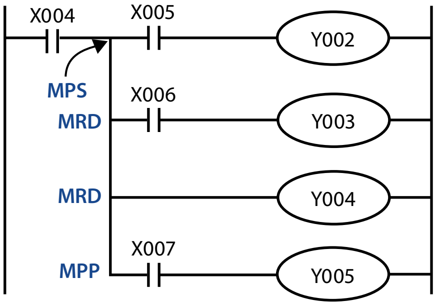
Outline

Convenient instructions for writing multiple branch output circuits.

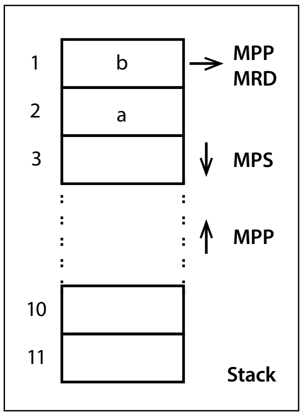
Function and Action Description

MPS, MRD, MPP Instructions (Press Stack, Read Stack, Pop Stack)

In the intelligent controller, there are 11 memories called stacks, which are used to memorize intermediate results of operations (ON or OFF).



```
0018  LD    X004
0019  MPS
0020  AND   X005
0021  OUT   Y002
0022  MRD
0023  AND   X006
0024  OUT   Y003
0025  MRD
0026  OUT   Y004
0027  MPP
0028  AND   X007
0029  OUT   Y005
0030  END
```



- After using MPS instruction to store the intermediate result of the operation, drive the output Y002.
- After reading the contents of the memory by using MRD instruction, the driver outputs Y003. MRD instruction can be programmed multiple times.
- MPP instruction is used to replace MRD instruction in the final output circuit, so that the storage content can be read out and reset at the same time.

Note

Note	
1	MPS instructions can also be reused, but the difference between the number of MPS instructions and MPP instructions is less than 11, and ultimately the number of instructions between the two needs to be the same.

3.1.9 MC, MCR Instruction

Outline

After the MC instruction is executed, the bus (LD, LDI point) moves behind the MC contact.

Using MCR instruction, it can be returned to the original bus position.

When changing the soft component numbers Y and M, MC instruction can be used multiple times. But when using the same soft component number, double coil output will occur, which is the same as OUT instruction.

Function and Action Description

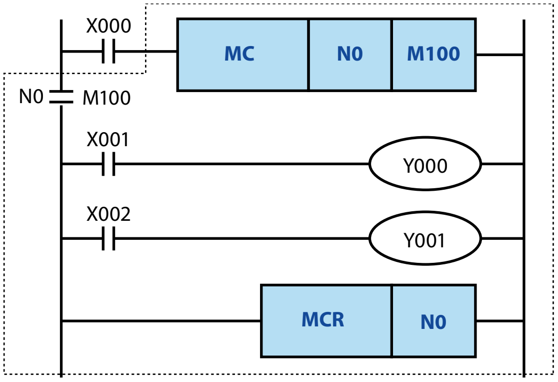
MC, MCR Instruction (Connected to the Common Contact, Disconnected to the Common Contact)

After MC instruction is executed, the bus (LD, LDI point) moves behind the MC contact.

The drive instruction connected to the bus after MC contact performs each action only when the MC command is executed, and OFF is executed when the MC instructions are not executed (the same action as when the contact is OFF).

Example: When input X000 is ON, the instruction from MC to MCR is executed, but when X000 is OFF, the actions of each drive soft component are as follows.

- Soft components converted to OFF: Timers (excluding cumulative timers), soft components driven by OUT instructions.
- Soft components that remain in state: Cumulative timers, counters, soft components driven by SET/RST instructions.



```
0000   LD    X000  ⎫
0001   MC    N0    ⎬ 3-step instruction
       SP    M100  ⎭
0004   LD    X001
0005   OUT   Y000
0006   LD    X002
0007   OUT   Y001
0008   MCR   N0     ← 2-step instruction

←  Please write the MCR NO instruction
```

Note

Note	
1	If there is no instruction (LD, LDI, etc.) following the MC instruction, there will be circuit error (error code: 6611).

3.1.10 INV Instruction

Outline

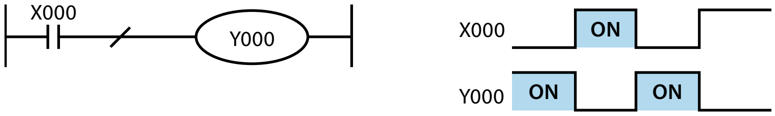
INV instruction is an instruction that reverses the result of operation before execution of INV instruction without specifying the soft component number.

Function and Action Description

INV Instruction (Inversion of Operation Result)

In the figure below, when X000 is OFF, Y000 is ON. If X000 is ON, Y000 is OFF.

INV instruction can be programmed at the same position as the series contact command (AND, ANI, ANDP, ANDF instructions). It cannot be connected to the bus as LD, LDI, LDP, LDF on the instruction list, nor can it be like OR, ORI, ORP, ORF instructions are used in parallel with the contact instructions.



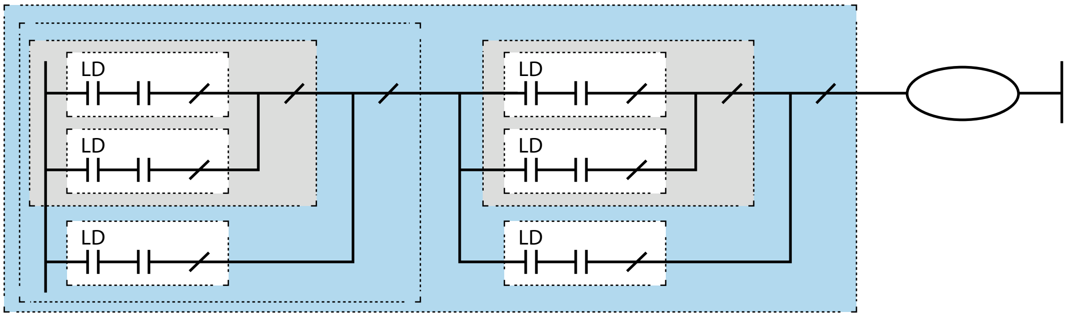
```
0000    LD     X000
0001    INV
0002    OUT    Y000
```

	Operation result before INV instruction executed	Operation result after INV instruction executed
	OFF →	ON
Instruction →	ON →	OFF

Action range of the INV instruction: When writing an INV instruction in a complex circuit containing an ORB instruction or an ANB instruction, the action range of the INV instruction is as shown in the figure below.

Function of the INV instruction: Reverses the operation result after the LD, LDI, LDP, and LDF instructions existing before the INV instruction is executed.

Therefore, as shown in the following figure, when programming in the ORB instruction or ANB instruction, the respective INV instructions are used. The block after LD, LDI, LDP, and LDF seen at the position is the object of the INV operation.



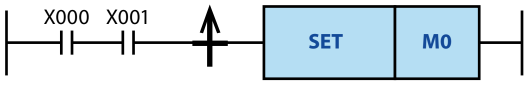
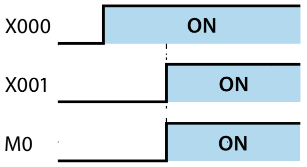
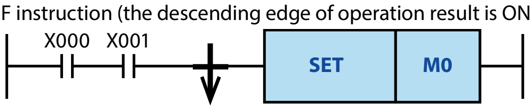
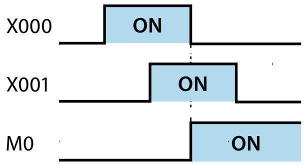
3.1.11 MEP, MEF Instruction

Outline

MEP and MEF instructions are instructions for pulsing the operation result without specifying the soft component number.

- MEP command: The result of operation up to MEP instruction changes from OFF ON to on state.
- MEF command: The result of operation up to MEF instruction changes from ON OFF to on state.
- When multiple contacts are connected in series, pulse processing can be easily realized by using MEP and MEF instructions.

Function and Action Description

<table>
<tr><th colspan="2">MEP, MFF Instruction</th></tr>
<tr><td>MEP instruction (the rising edge of the operation result is ON):

0000 LD X000
0001 AND X001
0002 MEP
0003 SET M0
 </td><td>MEF instruction (the descending edge of operation result is ON):

0000 LD X000
0001 AND X001
0002 MEF
0003 SET M0
 </td></tr>
</table>

Note

Note	
1	In subroutines and FOR ~ NEXT instructions, MEP and MEF instructions are used to pulse the contacts modified with the index, and may not operate normally.
2	MEP and MEF instructions are operated on the basis of the results of the operation up to the front of MEP/MEF instructions, so please use them in the same position as AND instruction.
3	MEP and MEF instructions cannot be used in the location of LD and OR.

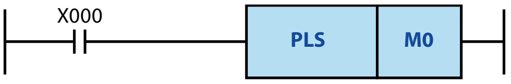
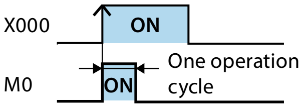
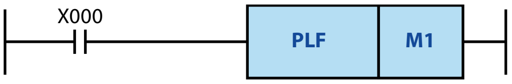
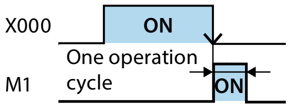
3.1.12 PLS, PLF Instruction

Outline

After using PLS instruction, the target soft component operates only in one calculation cycle after the drive input is turned ON.

After using PLF instruction, the target soft component operates only in one calculation cycle after the drive input is turned OFF.

Function and Action Description

<table>
<tr><th colspan="2">PLS, PLF Instruction</th></tr>
<tr><td>PLS instruction (differential output of rising edge):
In the figure below, when X000 changes from OFF to ON, only one operation cycle M0 is ON.

0000 LD X000
0001 PLS M0
 </td><td>PLF instruction (differential output of descending edge):
In the figure below, when X000 changes from ON to OFF, only one operation cycle M1 is ON.

0000 LD X000
0001 PLF M1
 </td></tr>
</table>

3.1.13 SET, RST Instruction

Outline

1) Bit Soft Component Setting (SET Instruction [Action Maintenance])

SET instruction is an instruction to turn ON the output relay (Y), auxiliary relay (M), status (S), and bit designation (D.b) of the word soft component when the command input is ON.

2) Bit Soft Component Reset (RST Instruction [Release Action Maintenance])

RST instruction is an instruction to reset the output relay (Y), auxiliary relay (M), status (S), timer (T), counter (C), and bit designation (Db) of the word soft component. It is possible to reset the soft component that is turned ON with SET instruction (OFF processing).

3) Current Value Clearance of the Word Soft Component (RST Directive [Current Value and Register Clearance])

RST instruction is an instruction to clear the current value data of the customizer (T), counter (C), data register (D), and index register (V), (Z).

In addition, the current value and the contact of the accumulated timers T246 ~ T255 reset can also be used using RST instruction.

Function and Action Description

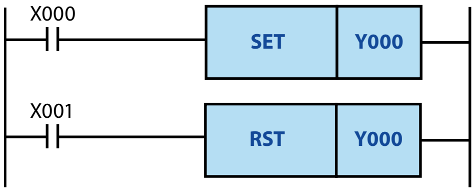
SET, RST Instruction

SET instruction is a coil drive instruction for the specified bits of the output relay (Y), auxiliary relay (M), status (S), and data register (D).

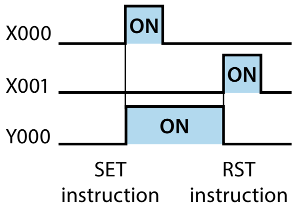
When Using Bit Software:

Parallel SET instructions can be used multiple times in succession.

In the following program example, SET Y000 is followed by RST Y000.

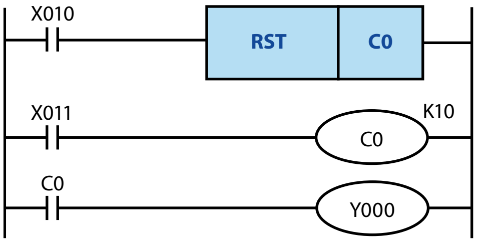


```
0000  LD   X000
0001  SET  Y000
0002  LD   X001
0003  RST  Y000
```



When Using Word Soft Components (Timers, Counters):

Use RST instruction to reset the counter and the cumulative timer.



1) Programming of internal counters

The number of times XOFF is turned off and on by C0 is counted up. When the count result reaches the set value K10, the output contact C0 is activated. If X010 is turned ON, RST instruction clears the value of the C0 counter.

2) Programming of high speed counters

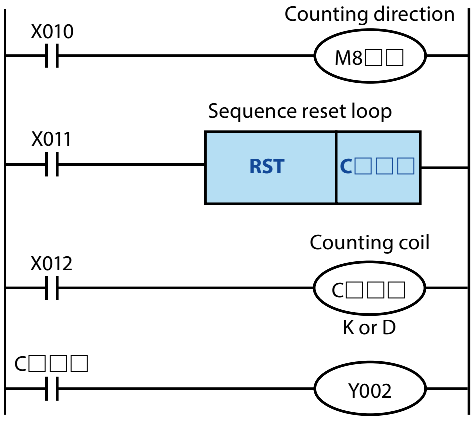
In the single-phase single-input counters of C235 and C236, special auxiliary relays M8235 and M8236 are used to specify the counting direction.

- X010: Decrease on ON, X010: Increase on OFF.

When X011 is ON, the output contact of the counter C□□□ is restored, and the current value of the counter also becomes 0.

When X012 is ON, the number of ON/OFF times of the count input X000/X001 determined by the counter number is counted.

- When the current value of the counter increases, the output contacts are positioned after the setting value (the content of K or D) and reset when passing in the reduced direction.
- For the contacts used to drive the high-speed counter count coil, use the contact that is always ON when the high-speed counter is executed.
- For input relays assigned as high-speed counters, do not drive them as counting coils, otherwise an error will be counted.

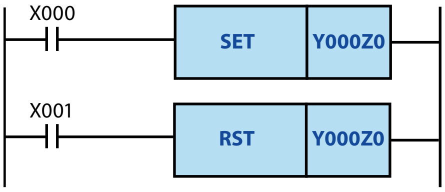


Index Modification

The soft components used in SET and RST instructions can be modified with the index register (V, Z).

- Status (S), special auxiliary relay (M), 32-bit counter (C), D.b cannot be modified.
- V0 ~ V7, Z0 ~ Z7 can be used in the index modification.
- When the soft component used is input (X) or output (Y), the value of the index register (V, Z) is converted to an octal number and then added.

Example: When the value of Z0 is 20, Y024 ON/OFF.



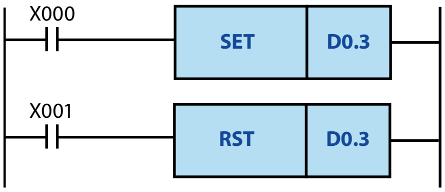
```
0000   LD    X000
0001   SET   Y000Z0
0004   LD    X001
0005   RST   Y000Z0
```

Bit Specification of Data Register (D)

In the soft component used by SET instruction and RST instruction, the bit of the data register (D) can be specified.

- When performing bit designation of the data register, enter "." after the number of the data register (D), and then enter the bit number (0 ~ F).
- The data registers that can be used are only valid for 16 bits.
- Specify the bit number in the order of 0, 1, 2 ... 9, A, B ... F from the low position.

Example: In the example on the right, after X000 is ON, the bit3 (D0.3) of D0 is ON. When X001 is ON, the bit3 (D0.3) of D0 turns OFF.



```
0000   LD    X000
0001   SET   D0.3
0004   LD    X001
0005   RST   D0.3
```

Note

Note	
1	When SET and RST instructions are executed on the output relay (Y) in the same calculation cycle, the result of the instruction near the END instruction (end of the program) is output.

Error

Error	
1	An operation error occurs when the index modification becomes a soft component number that does not actually exist (error code: 6706).

3.1.14 NOP Instruction

Outline

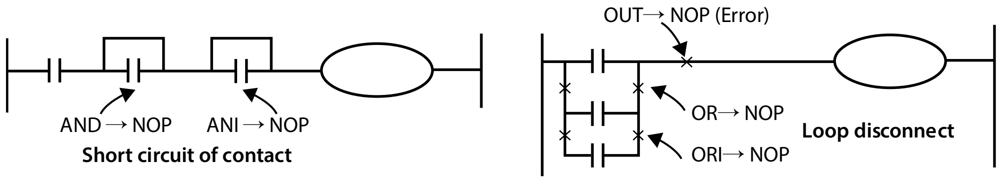
NOP instruction is a null operation instruction.

When a NOP is added between a general instruction and an instruction, the intelligent controller continues to operate regardless of its existence.

If NOP is added in the middle of the program, when the program needs to be changed or added, only a small change in the step number can be achieved, but the program is required to have a margin.

In addition, if the instructions that have been written are replaced by NOP instruction, the circuit will change+, please be careful.

Function and Action Description

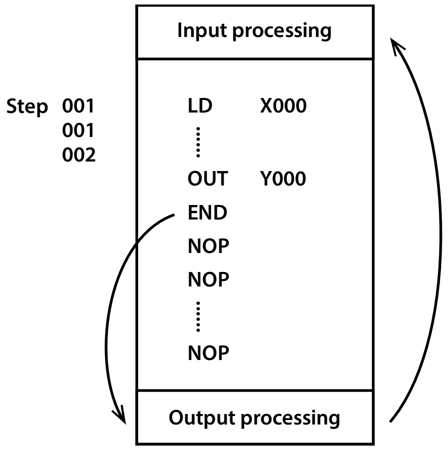
NOP Instruction (Empty Operation Instruction)
When written in the program, the intelligent controller will continue to run regardless of its existence. When changing an existing program and rewriting it to a NOP instruction, it is equivalent to the operation of deleting the instruction.  OUT→ NOP (Error) AND → NOP ANI → NOP Short circuit of contact OR→ NOP ORI→ NOP Loop disconnect

3.1.15 END Instruction

Outline

END instruction is an instruction that indicates the end of the program.

Function and Action Description

END Instructions (End of Program and Input/Output Processing and Return 0 Steps)	
The intelligent controller repeats [input processing]→[execution program]→[output processing]. If END instruction is written in the program, the remaining program steps will not be executed, and the output processing will be performed directly. When END instruction is executed, the timer is also refreshed (checking whether the operation cycle is too long).	 Input processing Step 001 001 002 LD X000 OUT Y000 END NOP NOP NOP Output processing

Note

Note	
1	Do not write END instructions in the middle of the program.

3.2 Step Sequence Control Instruction

Step ladder figure is a method of logically programming for each state according to the operation process of the controlled device, and decomposing into several states or processes, and then switching between states according to signal conditions.

STL ladder figure is used for programming. This programming method is clear with simple logic design, and is convenient for debugging and maintenance.

Step ladder figure instructions can be expressed by a ladder figure. In step ladder figure, state (S) is regarded as a control process from which input conditions and output control are programmed sequentially. The most important feature of this control is that when the process is in progress, it is not connected with the previous process, and the equipment can be controlled in a simple order of each process.

Instruction Symbol	Function	Operator Type	Operator						
			X0 ~ X377	Y0 ~ Y377	M0 ~ M3071 M8000 ~ M8511	S0 ~ S4095	T0 ~ T511	C0 ~ C255	D0 ~ D8511
STL	Program jump to subbus	S				●			
RET	Program returns to main bus	/							

Step ladder figure has corresponding programming rules, which not only contains the programming method of the ordinary ladder figure, but also have certain differences from the ordinary ladder figure programming to some extent. It is explained as follows:

- Step ladder figure starts with STL instruction (note that it is different from S in the normal ladder figure), ends with RET instruction, and the intermediate program is guided in the S state, followed by all the operation logic of the S state, including switching to the next state when the condition is satisfied.
- List of sequence instructions that can be processed in the status:

Command Status		LD/LDI/LDP/LDF, AND/ANI/ ANDP/ANDF, OR/ORI/ORF, INV, OUT, SET/RST, PLS/PLF	ANB/ORB MPS/MRD/MPP	MC/MCR
Initial/general state		Available	Available	Not available
Branch, merge state	Output processing	Available	Available	Not available
	Transfer processing	Available	Not available	Not available

- STL instruction cannot be used in interrupt programs and subroutines.

Jump instructions are not prohibited in STL instructions, but their actions are complicated and are not recommended.

See Chapter 6 for details of step sequence control instructions.

Chapter 4 Application Instructions

4.1 Program Flow

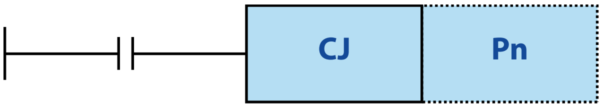
FN No.	Instruction Mark	Instruction Format	Function	Chapter	Page
00	CJ	CJ (Pn) CJP (Pn)	Conditional jump	4.1.1	40
01	CALL	CALL (Pn) CALLP (Pn)	Subroutine call	4.1.2	42
02	SRET	SRET (-)	Subroutine return	4.1.3	43
03	IRET	IRET (-)	Interrupt return	4.1.4	43
04	EI	EI (-)	Interrupt avaliable	4.1.5	44
05	DI	DI (-)	Interrupt banned	4.1.6	44
06	FEND	FEND (-)	Main program ended	4.1.7	45
07	WDT	WDT (-)	Timer	4.1.8	46
08	FOR	FOR (S)	Beginning of cycle range	4.1.9	46
09	NEXT	NEXT (-)	End of cycle range	4.1.10	47

4.1.1 FN 00 - CJ/Conditional Jump

Outline

Instructions that implement program conditional jumps.

It is possible to shorten the cycle time (scan cycle) and execute the program using the double coil.



Conditional Jump FN 00 - CJ	Instruction Mark	Execution Condition	Instruction Type	Instruction Steps
	CJ	Continuous type	16 bit	3
	CJP	Pulse type	16 bit	3

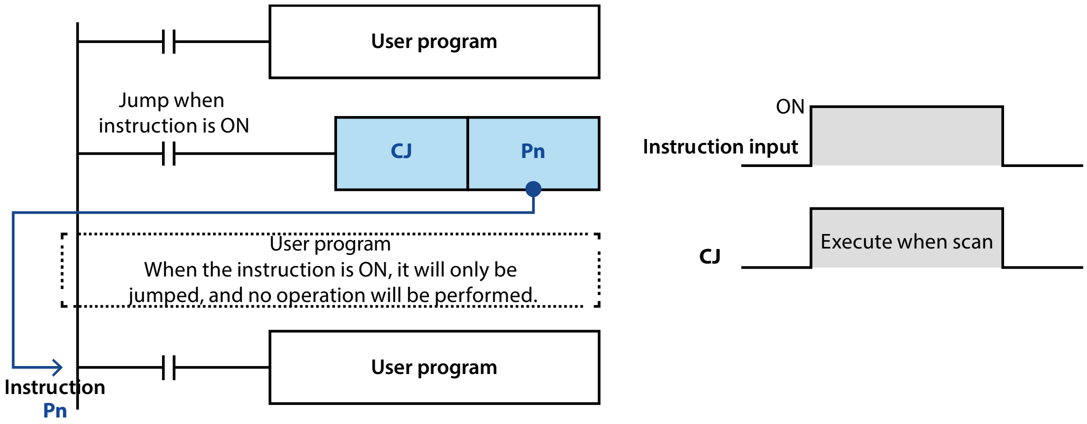
Operand	Setting Data														Instruction Type				
	The pointer number of the jump target mark number (P) (n = 0 ~ 4095, but P63 is END jump)														16 bit				
	Operand Object Soft Component																		
	Bit Soft Component							Word Soft Component								Others			
	X	Y	M	T	C	S	D.b	KnX	KnY	KnM	KnS	T	C	D	V, Z	H	K	E	P
Pn															●				●

Function and Action Description

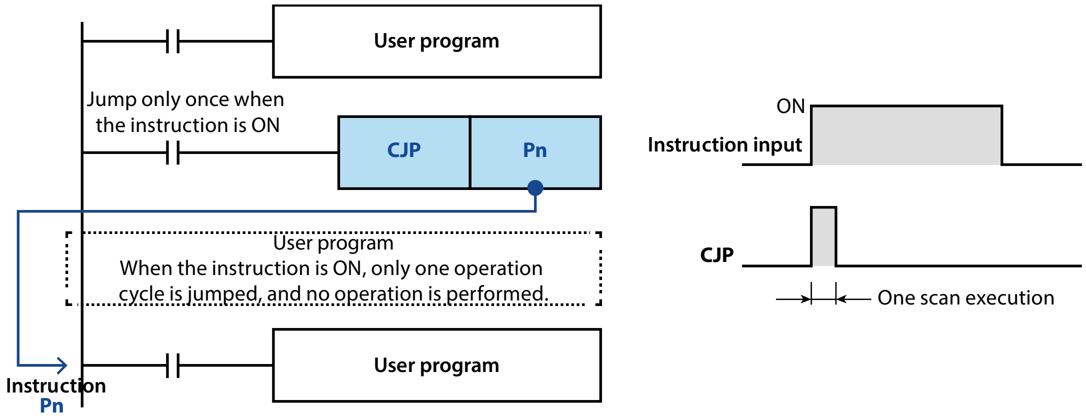
16-bit Operation (CALL, CALLP)

When the command input is ON, the program that specifies the mark (pointer number) is executed.

- When CJ instruction:


User program
Jump when instruction is ON
CJ
Pn
User program
When the instruction is ON, it will only be jumped, and no operation will be performed.
User program
Instruction
Pn
ON
Instruction input
CJ
Execute when scan

- When CJP instruction:


User program
Jump only once when the instruction is ON
CJP
Pn
User program
When the instruction is ON, only one operation cycle is jumped, and no operation is performed.
User program
Instruction
Pn
ON
Instruction input
CJP
One scan execution

Note:

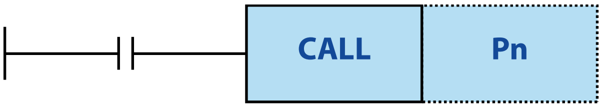
Note		Description
1	Write a mark in a position smaller than the CJ instruction step number	The marker can be written in a position smaller than the CJ instruction step number, but when the scan time exceeds 200ms (default setting), the timer error occurs, so be careful.
2	Mark (P) reuse prohibited	The mark number includes the mark for the CALL instruction described later, and an error occurs if the repeat number is used.
3	No need to enter the mark of the pointer P63	Pointer P63 indicates a jump to the END step. Do not program the P63. When programming the mark P63, the error code 6507 (mark definition error) is displayed in the intelligent controller and stops running.
4	Jump to the pointer of the subroutine	The tag used by the CALL instruction and the tag used by the CJ instruction cannot be shared. CJ does not allow jumping into subroutines or interrupt programs.

4.1.2 FN 01 - CALL/Subroutine Call

Outline

In the sequence program, instructions for calling programs that need to be processed together can reduce the number of steps in the program and design the program more efficiently.

In addition, the FEND (FN 06) and SRET (FN 02) instructions are required to write subroutines.



	Instruction Mark	Execution Condition	Instruction Type	Instruction Steps
Subroutine Call FN 01 - CALL	CALL	Continuous type	16 bit	3
	CALLP	Pulse type	16 bit	3

<table>
<tr><td rowspan="5">Operand</td><td colspan="14">Setting Data</td><td colspan="5">Instruction Type</td></tr>
<tr><td colspan="14">Pointer number of the jump target mark (P) (P0 ~ P62, P64 ~ P4095)
P63 is dedicated to CJ (FN 00) (END jump), so it cannot be used as a pointer to the CALL (FN 01) instruction.</td><td colspan="5">16 bit</td></tr>
<tr><td colspan="19">Operand Object Soft Component</td></tr>
<tr><td colspan="7">Bit Soft Component</td><td colspan="8">Word Soft Component</td><td colspan="4">Others</td></tr>
<tr><td>X</td><td>Y</td><td>M</td><td>T</td><td>C</td><td>S</td><td>D.b</td><td>KnX</td><td>KnY</td><td>KnM</td><td>KnS</td><td>T</td><td>C</td><td>D</td><td>V, Z</td><td>H</td><td>K</td><td>E</td><td>P</td></tr>
<tr><td>Pn</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td>●</td><td></td><td></td><td></td><td>●</td></tr>
</table>

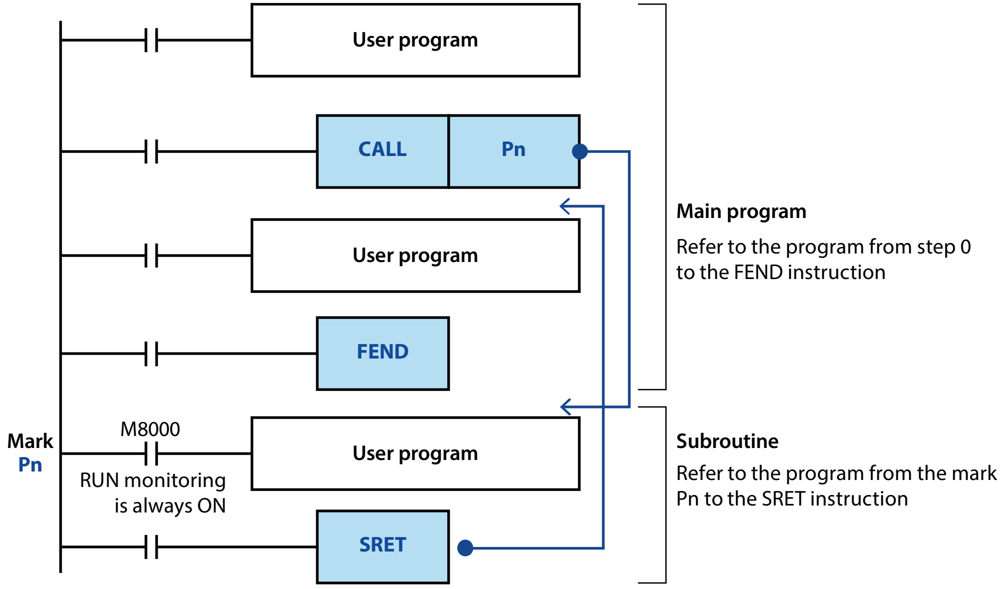
Function and Action Description

16-bit Operation (CALL, CALLP)

When the instruction input is ON, execute the CALL instruction, jump to the step of the mark Pn, and execute the subroutine of the mark Pn.

After executing SERT (FN 02), return to the next step of the CALL instruction.

- Programming with the FEND instruction at the end of the main program.
- The mark (P) for the CALL instruction, programmed after the FEND instruction.


User program
CALL
Pn
User program
FEND
Mark
Pn
M8000
RUN monitoring
is always ON
User program
SRET
Main program
Refer to the program from step 0 to the FEND instruction
Subroutine
Refer to the program from the mark Pn to the SRET instruction

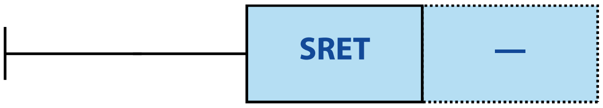
Note

Note		Description
1	Multi-level nested CALL in subroutine	The CALL instruction in the subroutine is allowed to be used up to 4 times, and as a whole, up to 5 levels of nesting are allowed.

4.1.3 FN 02 - SRET/Subroutine Return

Outline

The instruction to return from the subroutine to the main program.



Subroutine Call	Instruction Mark	Execution Condition	Instruction Type	Instruction Steps
FN 02 - SRET	SRET	Continuous type	Independent instruction	1

Operand	Setting Data															Instruction Type			
	No setting data															Independent instruction			
	Operand Object Soft Component																		
	Bit Soft Component							Word Soft Component								Others			
	X	Y	M	T	C	S	D.b	KnX	KnY	KnM	KnS	T	C	D	V, Z	H	K	E	P
—	No object soft component																		

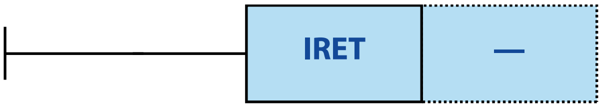
Function and Action Description

Independent Operation (SRET)
After executing the CALL instruction in the main program, jump to the subroutine, and then use the SRET instruction to return to the main program.

4.1.4 FN 03 - IRET/Interrupt Return

Outline

The instruction to return from the interrupt subroutine to the main program.



Subroutine Call	Instruction Mark	Execution Condition	Instruction Type	Instruction Steps
FN 03 - IRET	IRET	Continuous type	Independent instruction	1

Operand	Setting Data															Instruction Type			
	No setting data															Independent instruction			
	Operand Object Soft Component																		
	Bit Soft Component							Word Soft Component								Others			
	X	Y	M	T	C	S	D.b	KnX	KnY	KnM	KnS	T	C	D	V, Z	H	K	E	P
—	No object soft component																		

Function and Action Description

Independent Operation (IRET)

If an interrupt (input, timer, counter) is generated while the main program is being processed, jump to the interrupt (I) program and return to the main program using the IRET instruction.

The methods to jump to the interrupt program include the following three.

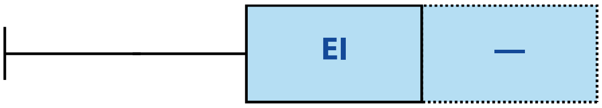
Function		Interrupt Number	Description
1	Input interrupt	100* ~ 150*	Input (X) signal ON/OFF execution interrupt processing.
2	Timer interrupt	16** ~ 18**	Interrupt processing is performed every specified time interval (fixed cycle).
3	Counter interrupt	1010 ~ 1060	Interrupt processing is performed when the high-speed counter increments.

4.1.5 FN 04 - EI/Interrupt Available

Outline

The intelligent controller usually disables the interrupt state. Using this command, the intelligent controller can be made into a state that allows interrupts.

Use the input interrupt and timer interrupt, counter interrupt function, please use this instruction.



Subroutine Call	Instruction Mark	Execution Condition	Instruction Type	Instruction Steps
FN 04 - EI	EI	Continuous type	Independent instruction	1

Operand	Setting Data														Instruction Type				
	No setting data														Independent instruction				
	Operand Object Soft Component																		
	Bit Soft Component							Word Soft Component								Others			
	X	Y	M	T	C	S	D.b	KnX	KnY	KnM	KnS	T	C	D	V, Z	H	K	E	P
—	No object soft component																		

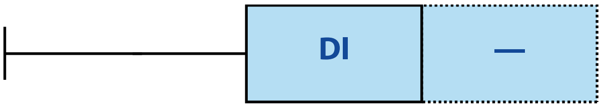
Function and Action Description

Independent Operation (EI)
The EI instruction is an independent operation that does not require an instruction (drive) contact.

4.1.6 FN 05 - DI/Interrupt Banned

Outline

Use DI (FN 05) after changing to allow interrupts, the instruction is changed again to disable the interrupt.



Subroutine Call	Instruction Mark	Execution Condition	Instruction Type	Instruction Steps
FN 05 - DI	DI	Continuous type	Independent instruction	1

Operand	Setting Data														Instruction Type				
	No setting data														Independent instruction				
	Operand Object Soft Component																		
	Bit Soft Component							Word Soft Component								Others			
	X	Y	M	T	C	S	D.b	KnX	KnY	KnM	KnS	T	C	D	V, Z	H	K	E	P
—	No object soft component																		

Function and Action Description

Independent Operation (DI)
The DI instruction is an independent instruction that does not require an instruction (drive) contact.

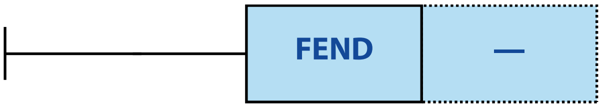
Note

Note	
1	The interrupt (request) generated after the DI will be responded to after the interrupt is restored (up to 6 groups of cache).
2	The timer interrupt is still accounting between DI and EI.
3	If there is no need to disable interrupts, only EI can be used instead of DI.

4.1.7 FN 06 - FEND/Main Program Ended

Outline

The main program ends the instruction.



Subroutine Call	Instruction Mark	Execution Condition	Instruction Type	Instruction Steps
FN 06 - FEND	FEND	Continuous type	Independent instruction	1

Operand	Setting Data														Instruction Type				
	No setting data														Independent instruction				
	Operand Object Soft Component																		
	Bit Soft Component							Word Soft Component								Others			
	X	Y	M	T	C	S	D.b	KnX	KnY	KnM	KnS	T	C	D	V, Z	H	K	E	P
—	No object soft component																		

Function and Action Description

Independent Operation (FEND)
After executing the FEND instruction, the same output processing as the END instruction, input processing, refresh of the timer, and then return to the 0-step program are executed. This instruction is required to write subroutines and interrupt programs.

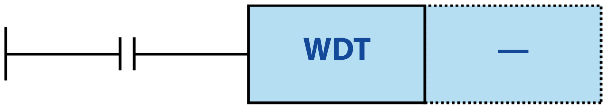
Note

Note		Description
1	Do not write FEND instructions multiple times	Please write subroutines and interrupt subroutines between the last FEND and END instructions.
2	CALL and CALLP instructions	To write a label after the FEND instruction, you must use the SRET instruction.
3	FOR instruction	After the FOR instruction is executed, an error will occur if the FEND instruction is executed before the NEXT instruction is executed.
4	When using the interrupt function (I)	The interrupt tag (pointer) must be written after the FEND instruction and the IRET instruction is required.
5	Disable CJ instructions to skip FEND execution	

4.1.8 FN 07 - WDT/Timer

Outline

The instruction to refresh the timer by the sequence program.



Subroutine Call FN 07 - WDT	Instruction Mark	Execution Condition	Instruction Type	Instruction Steps
	WDT	Continuous type	Independent instruction	1
	WDTP	Pulse type	Independent instruction	1

Operand	Setting Data																		
	No setting data														Independent instruction				
	Operand Object Soft Component																		
	Bit Soft Component							Word Soft Component								Others			
	X	Y	M	T	C	S	D.b	KnX	KnY	KnM	KnS	T	C	D	V, Z	H	K	E	P
—	No object soft component																		

Note: the first row header "Instruction Type" stands above the last columns (V, Z – P).

Function and Action Description

Independent Operation (WDT, WDTP)
If the operation cycle of the smart controller (0 ~ END or execution time of the awkward instruction) exceeds the timer time set by D8000, the smart controller will have timer failure (downtime). In the middle of a program with a long operation cycle, the watchdog timer can be refreshed by inserting a WDT instruction to avoid the timer failure.

Related Soft Component

Soft Component	Name	Content
D8000	The time of timer	The Max. can be set to 3000ms, the unit is ms (initial value: 200).

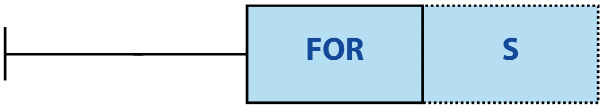
Note

Note		Description
1	Error of the timer	When there are more loop commands or more high-speed counters, the operation time will increase, resulting in the timer failure, so change the time to extend the D8000 watchdog timer near the start step.

4.1.9 FN 08 - FOR/Beginning of Cycle Range

Outline

The program from the beginning of the FOR instruction to the NEXT (FN 09) instruction is repeated for the specified number of times.



Subroutine Call	Instruction Mark	Execution Condition	Instruction Type	Instruction Steps
FN 08 - FOR	FOR	Continuous type	16 bit	3

Operand	Setting Data													Instruction Type				
	The number of repetitions between FOR ~ NEXT instructions [S = K1 ~ K32,767 (-32768 ~ 0 as 1 processing)].													16 bit				
	Operand Object Soft Component																	
	Bit Soft Component							Word Soft Component								Others			
	X	Y	M	T	C	S	D.b	KnX	KnY	KnM	KnS	T	C	D	V, Z	H	K	E	P
S								●	●	●	●	●	●	●	●	●	●		

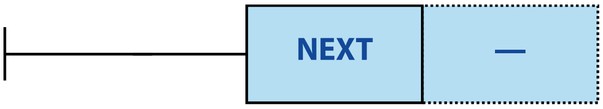
Function and Action Description

16-bit Operation (FOR)
For details, refer to the NEXT (FN 09) instruction, section 4.1.10.

4.1.10 FN 09 - NEXT/End of Cycle Range

Outline

From the FOR (FN 08) instruction to NEXT, the program between instructions is repeated a specified number of times.



Subroutine Call	Instruction Mark	Execution Condition	Instruction Type	Instruction Steps
FN 09 - NEXT	NEXT	Continuous type	Independent pointing	1

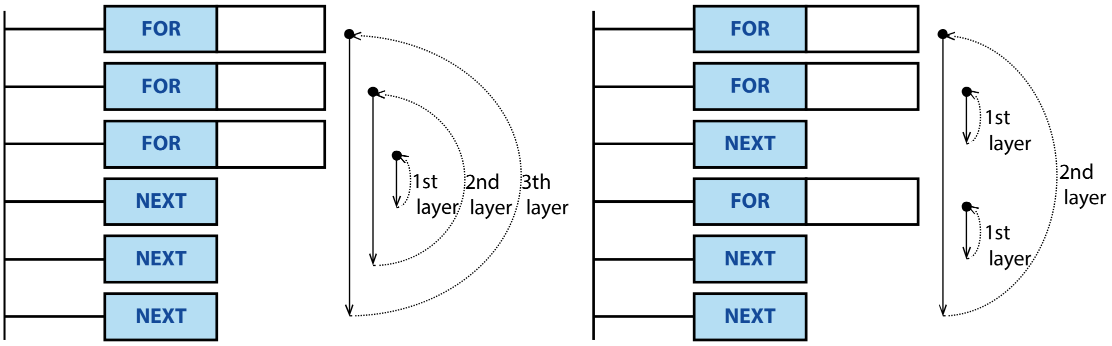
Operand	Setting Data														Instruction Type				
	No setting data														Independent instruction				
	Operand Object Soft Component																		
	Bit Soft Component							Word Soft Component								Others			
	X	Y	M	T	C	S	D.b	KnX	KnY	KnM	KnS	T	C	D	V, Z	H	K	E	P
—	No object soft component																		

Function and Action Description

Independent Operation (NEXT)
The processing between the FOR ~ NEXT instructions is repeated n times (the number of times specified in the source data). After repeating the specified number of times, the steps after the NEXT instruction are executed.

Note

Note		Description
1	Multi-layer limit	Between the FOR ~ NEXT instructions, up to 5 layers of FOR ~ NEXT instructions can be nested.



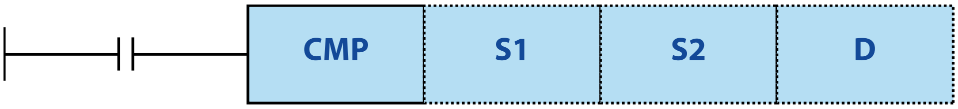
4.2 Transmission and Comparison

FN No.	Instruction Mark	Instruction Format	Function	Chapter	Page
10	CMP	CMP (S1) (S2) (D) CMPP (S1) (S2) (D) DCMP (S1) (S2) (D) DCMPP (S1) (S2) (D)	Comparison	4.2.1	50
11	ZCP	ZCP (S1) (S2) (S) (D) ZCPP (S1) (S2) (S) (D) DZCP (S1) (S2) (S) (D) DZCPP (S1) (S2) (S) (D)	Interval comparison	4.2.2	51
12	MOV	MOV (S) (D) MOVP (S) (D) DMOV (S) (D) DMOVP (S) (D)	Transmission	4.2.3	52
13	SMOV	SMOV (S) (m1) (m2) (D) (n) SMOVP (S) (m1) (m2) (D) (n)	Bit movement	4.2.4	53
14	CML	CML (S) (D) CMLP (S) (D) DCML (S) (D) DCMLP (S) (D)	Reverse transfer	4.2.5	54
15	BMOV	BMOV (S) (D) (n) BMOVP (S) (D) (n)	Batch transfer	4.2.6	55
16	FMOV	FMOV (S) (D) (n) FMOVP (S) (D) (n) DFMOV (S) (D) (n) DFMOVP (S) (D) (n)	Multicast transfer	4.2.7	56
17	XCH	XCH (D1) (D2) XCHP (D1) (D2) DXCH (D1) (D2) DXCHP (D1) (D2)	Exchange	4.2.8	57
18	BCD	BCD (S) (D) BCDP (S) (D) DBCD (S) (D) DBCDP (S) (D)	BCD conversion	4.2.9	58
19	BIN	BIN (S) (D) BINP (S) (D) DBIN (S) (D) DBINP (S) (D)	BIN conversion	4.2.10	59

4.2.1 FN 10 - CMP/Comparison

Outline

Compare the two values and output the result (large, consistent, small) to the bit soft component (3 points).



Comparison FN10 - CMP	Instruction Mark	Execution Condition	Instruction Type	Instruction Steps
	CMP	Continuous type	16 bit	7
	CMPP	Pulse type	16 bit	7
	DCMP	Continuous type	32 bit	13
	DCMPP	Pulse type	32 bit	13

Operand	Setting Data	Data Type
	S1: Data or soft component number of the comparison value	16/32 bit
	S2: Compare source data or soft component number	16/32 bit
	D: Output the starting bit soft component number of the comparison result	Bit

Operand	Operand Object Soft Component																		
	Bit Soft Component							Word Soft Component								Others			
	X	Y	M	T	C	S	D.b	KnX	KnY	KnM	KnS	T	C	D	V, Z	H	K	E	P
S1								●	●	●	●	●	●	●	●	●	●		
S2								●	●	●	●	●	●	●	●	●	●		
D		●	●			●	●								●				

Function and Action Description

16-bit Operation (CMP, CMPP)	32-bit Operation (DCMP, DCMPP)
Compare the contents of the comparison value S1 and the comparison source S2, and make one of D, D+1, D+2 ON according to the result (small, consistent, large). • Source data S1, S2 are processed as BIN (binary) values. • Compare sizes by algebra. For example: -10 < 2. • When S1 > S2, D is ON. • When S1 = S2, D+1 is ON. • When S1 < S2, D+2 is ON.	Compare the contents of the comparison value [S1+1,S1] and the comparison source [S2+1,S2], and make one of D, D+1, D+2 ON according to the result (small, consistent, large). • Source data [S1+1,S1], [S2+1,S2] are processed as BIN (binary) values. • Compare the sizes in algebraic form. For example: -125400 < 22466. • When [S1+1,S1] > [S2+1,S2], D is ON. • When [S1+1,S1] = [S2+1,S2], D+1 is ON. • When [S1+1,S1] < [S2+1,S2], D+2 is ON.

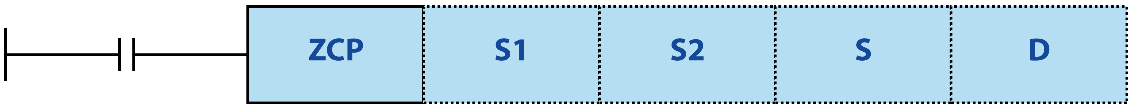
Note

Note		Description
1	Number of occupied soft components	Takes 3 points starting with the soft component specified in D. Be careful not to repeat with other soft components used in control.

4.2.2 FN 11 - ZCP/Interval Comparison

Outline

The result of comparing the comparison source with two values (up, middle, down) is output to the bit soft component (3 points).



Interval Comparison FN11 - ZCP	Instruction Mark	Execution Condition	Instruction Type	Instruction Steps
	ZCP	Continuous type	16 bit	9
	ZCPP	Pulse type	16 bit	9
	DZCP	Continuous type	32 bit	17
	DZCPP	Pulse type	32 bit	17

Operand	Setting Data													Data Type					
	S1: Data or soft component number of the lower comparison value													16/32 bit					
	S2: Data or soft component number of the comparison value on the upper side													16/32 bit					
	S: Compare source data or soft component number													16/32 bit					
	D: Output start bit soft component number of comparison result													Bit					
	Operand Object Soft Component																		
	Bit Soft Component							Word Soft Component								Others			
	X	Y	M	T	C	S	D.b	KnX	KnY	KnM	KnS	T	C	D	V, Z	H	K	E	P
S1								●	●	●	●	●	●	●	●	●	●		
S2								●	●	●	●	●	●	●	●	●	●		
S								●	●	●	●	●	●	●	●	●	●		
D		●	●			●	●								●				

Function and Action Description

16-bit Operation (ZCP, ZCPP)	32-bit Operation (DZCP, DZCPP)
Compare the content of the comparison source S with the lower comparison value S1 and the upper comparison value S2, and make one of D, D+1, D+2 ON according to the result (small, consistent, large). • Compare sizes by algebra. For example: -10 < 2 < 10. • When 1 > S, D is ON. • When S1 ≤ S ≤ S2, D+1 is ON. • When S > S2, D+2 is ON.	Compare the contents of the comparison source [S+1,S] with the lower comparison value [S1+1,S1] and the upper comparison value [S2+1,S2], and based on the result (small, intra-region, large), one of D, D+1, D+2 is ON. • Size comparisons in algebraic form. For example: -125400 < 22466 < 1015444. • When [S1+1,S1] > [S+1,S], D is ON. • When [S1+1,S1] ≤ [S+1,S] ≤ [S2+1,S2], D+1 is ON. • When [S+1,S] > [S2+1,S2], D+2 is ON.

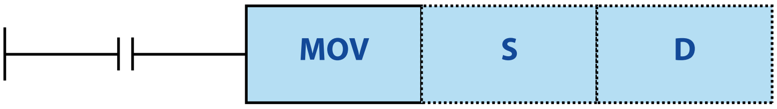
Note

Note		Description
1	Number of occupied soft components	Takes 3 points starting with the soft component specified in D. Be careful not to repeat with other soft components used in control.

4.2.3 FN 12 - MOV/Transmission

Outline

The instruction to transfer (copy) the contents of the soft component to other soft components.



	Instruction Mark	Execution Condition	Instruction Type	Instruction Steps
Interval Comparison FN12 - MOV	MOV	Continuous type	16 bit	5
	MOVP	Pulse type	16 bit	5
	DMOV	Continuous type	32 bit	9
	DMOVP	Pulse type	32 bit	9

	Setting Data	Data Type
Operand	S: Data of the transmission source or the soft component number of the saved data	16/32 bit
	D: The soft component number of the transfer destination	16/32 bit

	Operand Object Soft Component																		
	Bit Soft Component							Word Soft Component								Others			
	X	Y	M	T	C	S	D.b	KnX	KnY	KnM	KnS	T	C	D	V, Z	H	K	E	P
S								●	●	●	●	●	●	●	●	●	●		
D									●	●	●	●	●	●	●				

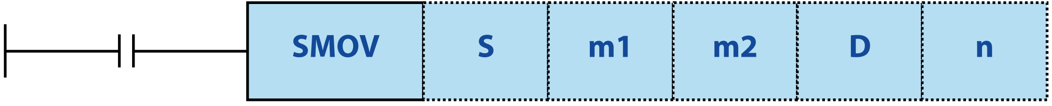
Function and Action Description

16-bit Operation (MOV, MOVP)	32-bit Operation (DMOV, DMOVP)
Transfer the content of the transfer source S to the transfer destination D. • When a constant (K) is specified in the transfer source S, it is automatically converted to BIN. When the transmission source S is designated as Kn□□: • The value is converted to BIN for transmission. Up to 16 (multiple of 4) bit soft components are transmitted. When the transfer destination D is specified as Kn□□: • Pass the low n * 4 bits of the transmitted value to D. Transfer up to 16 (multiple of 4) bit soft components.	Transfer the contents of the transfer source [S+1,S] to the transfer destination [D+1,D]. • When a constant (K) is specified in the transfer source [S+1,S], it is automatically converted to BIN. When the transmission source S is designated as Kn□□: • The value is converted to BIN for transmission. Up to 32 (multiple of 4) bit soft components are transmitted. When the transfer destination D is specified as Kn□□: • Pass the low n * 4 bits of the transmitted value to D. Transfer up to 32 (multiple of 4) bit soft components.

4.2.4 FN 13 - SMOV/Bit Movement

Outline

An instruction to perform data distribution synthesis in units of bits (4 digits).



Bit Movement FN13 - SMOV	Instruction Mark	Execution Condition	Instruction Type	Instruction Steps
	SMOV	Continuous type	16 bit	11
	SMOVP	Pulse type	16 bit	11

Operand	Setting Data	Data Type
	S: The number of the data soft component in which the bit movement is to be performed is saved	16 bit
	m1: The position of the start bit to move	16 bit
	m2: Number of bits to move	16 bit
	D: Save the soft component number of the bit movement data already	16 bit
	n: Specifies the position of the start bit of the moving target	16 bit

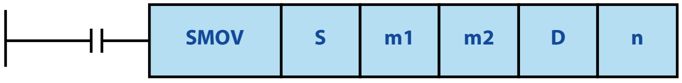
Operand Object Soft Component	Bit Soft Component							Word Soft Component								Other			
	X	Y	M	T	C	S	D.b	KnX	KnY	KnM	KnS	T	C	D	V, Z	H	K	E	P
S								●	●	●	●	●	●	●	●				
m1																●	●		
m2																●	●		
D									●	●	●	●	●	●	●				
n																●	●		

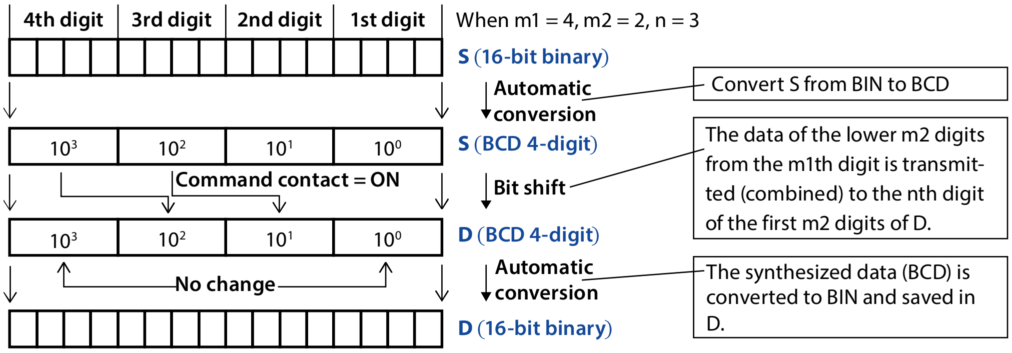
Function and Action Description

16-bit Operation (SMOV, SMOVP)

The content conversion of the transfer source S and the transfer destination D (0000 ~ 99999) is a 4-digit BCD, and the data of the low m2 digits from the m1th bit is transmitted (synthesized) to the nth digit of D. The m2 digit is then converted to BIN and saved in the transfer destination D.

- When the command input is ON, the data of the transfer source S and the number of bits except the specified transfer in the transfer destination D do not change.




4th digit
3rd digit
2nd digit
1st digit
When m1 = 4, m2 = 2, n = 3
S (16-bit binary)
Automatic conversion
Convert S from BIN to BCD
10^3
10^2
10^1
10^0
S (BCD 4-digit)
Command contact = ON
Bit shift
The data of the lower m2 digits from the m1th digit is transmitted (combined) to the nth digit of the first m2 digits of D.
10^3
10^2
10^1
10^0
D (BCD 4-digit)
No change
Automatic conversion
The synthesized data (BCD) is converted to BIN and saved in D.
D (16-bit binary)

Extensions

When the M8168 is turned ON, the BIN→BCD conversion cannot be performed when the SMOV instruction is executed.

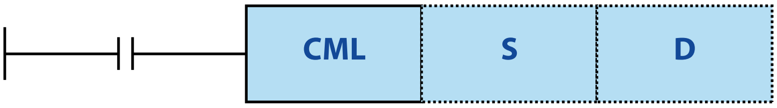
Bit movement is performed in units of 4 bits.

- M8168 can also be used for other commands, please pay attention when using.

4.2.5 FN 14 - CML/Reverse Transfer

Outline

An instruction to transfer (copy) after inverting data in bits.



Reverse Transfer FN14 - CML	Instruction Mark	Execution Condition	Instruction Type	Instruction Step
	CML	Continuous type	16 bit	5
	CMLP	Pulse type	16 bit	5
	DCML	Continuous type	32 bit	9
	DCMLP	Pulse type	32 bit	9

Operand	Setting Data														Data Type				
	S: The data to be inverted or the Word soft component number to save the data														16/32 bit				
	D: Save the target word soft component number of the data to be inverted														16/32 bit				
	Operand Object Soft Component																		
	Bit Soft Component							Word Soft Component								Others			
	X	Y	M	T	C	S	D.b	KnX	KnY	KnM	KnS	T	C	D	V, Z	H	K	E	P
S								●	●	●	●	●	●	●	●	●	●		
D									●	●	●	●	●	●	●				

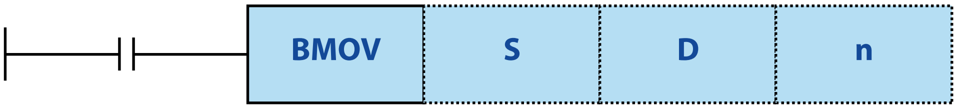
Function and Action Description

16-bit Operation (CML, CMLP)	32-bit Operation (DCML, DCMLP)
Invert the bits of the soft component specified in S (0→1, 1→0) and transfer to D. • When a constant (K) is specified in S, it is automatically converted to BIN. • You can use the output of the intelligent controller when you want to output it in a logical inversion. When the number of bits of the specified bit soft component (KnM, etc.) is included, the result is converted to a 16-bit BIN and then bitwise inverted, and the corresponding number of bits is passed to the destination operand.	Invert the bits of the soft component specified in [S+1,S] (0→1, 1→0) and transfer to [D+1,D]. • When a constant (K) is specified in [S+1,S], it is automatically converted to BIN. • You can use the output of the intelligent controller when you want to output it in a logical inversion. When the number of bits of the specified bit soft component (KnM, etc.) is included, the result is converted to a 16-bit BIN and then bitwise inverted, and the corresponding number of bits is passed to the destination operand.

4.2.6 FN 15 - BMOV/Batch Transfer

Outline

Batch transfer (copy) multiple data of a specified number of points.



	Instruction Mark	Execution Condition	Instruction Type	Instruction Step
Batch Transfer FN15 - BOV	BMOV	Continuous type	16 bit	7
	BMOVP	Pulse type	16 bit	7

	Setting Data	Data Type
Operand	S: Soft component number of the transmission source	16 bit
	D: The soft component number of the transfer destination	16 bit
	n: Number of transmission points (including file register) [n ≤ 512]	16 bit

Operand Object Soft Component	Bit Soft Component							Word Soft Component								Others			
	X	Y	M	T	C	S	D.b	KnX	KnY	KnM	KnS	T	C	D	V, Z	H	K	E	P
S								●	●	●	●	●	●	●	●				
D									●	●	●	●	●	●	●				
n														●		●	●		

Function and Action Description

16-bit Operation (BMOV, BMOVP)

The data of the n point starting from S is transmitted in batches to the D starting point n.

- The command gives an error when the soft component number range is exceeded (error No. 6706), and the transfer processing is not executed.

It can be transmitted even if the transmission number range overlaps.

- To prevent the data source from being overwritten if it is not transmitted, use the number overlap method. When the source operand address is higher than the destination operand address, it is transferred backward from the start address (lower address) when the source operand address is lower.
- When the destination operand address is transmitted from the end address (higher address).

Extended Function (Bidirectional Transfer Function)

Two-way transmission can be realized in one program by controlling the direction reversal flag M8024 of the BMOV (FN 15) instruction.

M8024 is OFF: From S to D; M8024 is ON: From D to S (M8024 is cleared when RUN→STOP).

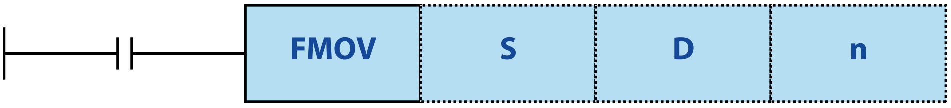
Note

Note	
1	In the case where both S and D are bit soft components specified for the number of bits, S and D are to have the same number of bits.

4.2.7 FN 16 - FMOV/Multicast Transfer

Outline

An instruction transfers the same data to multiple soft components.



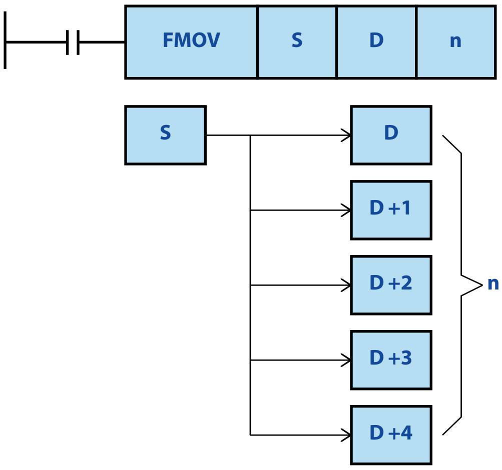
Bit Movement FN16 - FMOV	Instruction Mark	Execution Condition	Instruction Type	Instruction Step
	FMOV	Continuous type	16 bit	7
	FMOVP	Pulse type	16 bit	7
	DFMOV	Continuous type	32 bit	13
	DFMOVP	Pulse type	32 bit	13

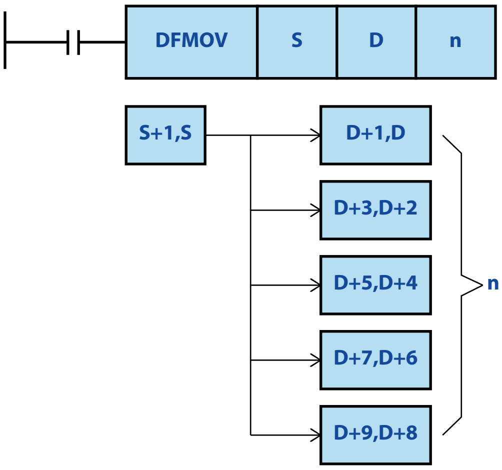
Operand	Setting Data	Data Type
	S: Data of the transmission source or the soft component number of the saved data	16/32 bit
	D: Start word soft component number of the transfer destination (the same data of the transfer source is transferred in batches)	16/32 bit
	n: Number of transmission points [K1 ≤ n ≤ K512, H1 ≤ n ≤ H1FF]	16 bit

Operand	Operand Object Soft Component																		
	Bit Soft Component							Word Soft Component								Others			
	X	Y	M	T	C	S	D.b	KnX	KnY	KnM	KnS	T	C	D	V, Z	H	K	E	P
S								●	●	●	●	●	●	●	●	●	●		
D									●	●	●	●	●	●	●				
n																●	●		

Function and Action Description

16-bit Operation (FMOV, FMOVP)	32-bit Operation (DFMOV, DFMOVP)
Transfer the contents of S to the soft component at point n starting with D. • The contents of the n-point soft components are the same. • When the number specified by n exceeds the soft component number range, the command gives an error (error No. 6706), and the transfer processing is not executed. • When a constant (K) is specified in the transfer source S, it is automatically converted to BIN.	Transfer the contents of [S+1,S] to the 32-bit soft component starting at [D+1,D]. • The contents of the 32-bit soft components at n points are the same. • When the number specified by n exceeds the soft component number range, the command gives an error (error No. 6706), and the transfer processing is not executed. • When a constant (K) is specified in the transmission source [S+1,S], it is automatically converted to BIN.

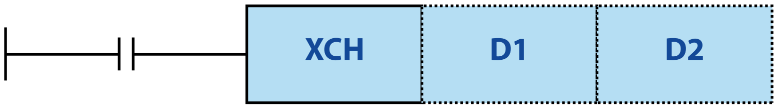




4.2.8 FN 17 - XCH/Exchange

Outline

Data exchange between two soft components.



	Instruction Mark	Execution Condition	Instruction Type	Instruction Step
Exchange **FN17 - XCH**	XCH	Continuous type	16 bit	5
	XCHP	Pulse type	16 bit	5
	DXCH	Continuous type	32 bit	9
	DXCHP	Pulse type	32 bit	9

Operand	Setting Data	Data Type
	D1: Soft component number for saving exchange data	16/32 bit
	D2: Soft component number for saving exchange data	16/32 bit

Operand	Operand Object Soft Component																		
	Bit Soft Component							Word Soft Component								Others			
	X	Y	M	T	C	S	D.b	KnX	KnY	KnM	KnS	T	C	D	V, Z	H	K	E	P
D1								●	●	●	●	●	●	●	●				
D2								●	●	●	●	●	●	●	●				

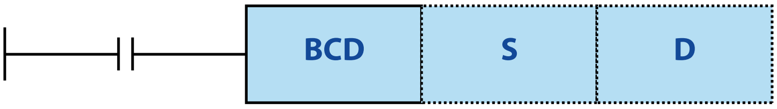
Function and Action Description

16-bit Operation (XCH, XCHP)	32-bit Operation (DXCH, DXCHP)
D1 and D2 exchange data with each other.	[D1+1,D1] and [D2+1,D2] exchange data with each other.

4.2.9 FN 18 - BCD/BCD Conversion

Outline

An instruction that is transmitted after converting a binary number (BIN) to a decimal number (BCD).



BCD Conversion FN18 - BCD	Instruction Mark	Execution Condition	Instruction Type	Instruction Steps
	BCD	Continuous type	16 bit	5
	BCDP	Pulse type	16 bit	5
	DBCD	Continuous type	32 bit	9
	DBCDP	Pulse type	32 bit	9

Operand	Setting Data	Data Type
	S: Save the conversion source (binary number) word soft component number of the data	16/32 bit
	T	16/32 bit

Operand	Operand Object Soft Component																		
	Bit Soft Component							Word Soft Component								Others			
	X	Y	M	T	C	S	D.b	KnX	KnY	KnM	KnS	T	C	D	V, Z	H	K	E	P
S								●	●	●	●	●	●	●	●				
D									●	●	●	●	●	●	●				

Function and Action Description

16-bit Operation (BCD, BCDP)

Convert BIN (binary) data of S to BCD (decimal) data and transfer it to D.

- S data can be converted to BCD (decimal number) from K0 to K9999.
- When specifying the number of digits for S and D, refer to the table below.

D	Number of Digits	Data Range
K1Y000	1 digit	0 ~ 9
K2Y000	2 digits	00 ~ 99
K3Y000	3 digits	000 ~ 999
K4Y000	4 digits	0000 ~ 9999

32-bit Operation (DBCD, DBCDP)

Convert the BIN (binary) data of [S+1,S] to BCD (decimal) data and transfer it to [D+1,D].

- The data of [S+1,S] can be converted to BCD (decimal number) of K0 ~ K99,999,999.
- When [S+1,S] and [D+1,D] specify the number of digits, refer to the table below.

[D+1,D]	Number of Digits	Data Range
K1Y000	1 digit	0 ~ 9
K2Y000	2 digits	00 ~ 99
K3Y000	3 digits	000 ~ 999
K4Y000	4 digits	0000 ~ 9999
K5Y000	5 digits	00000 ~ 99999
K6Y000	6 digits	000000 ~ 999999
K7Y000	7 digits	0,000,000 ~ 9,999,999
K8Y000	8 digits	00,000,000 ~ 99,999,999

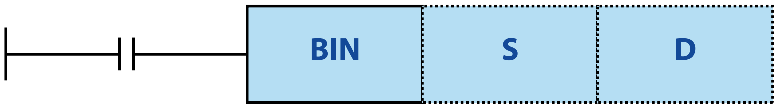
Note

Note		Description
1	About the input and output processing of BCD	Four arithmetic operations (+ - × ÷) and the addition of one, minus one instruction and other intelligent controller operations are performed in BIN (binary number). • When reading BCD (decimal) digital switch information into the intelligent controller, use BIN. • (FN 19) BCD→BIN conversion transfer instruction.

4.2.10 FN 19 - BIN/BIN Conversion

Outline

An instruction that is transmitted after converting a decimal number (BCD) to a binary number (BIN).



BIN Conversion FN20 - BIN	Instruction Mark	Execution Condition	Instruction Type	Instruction Step
	BIN	Continuous type	16 bit	5
	BINP	Pulse type	16 bit	5
	DBIN	Continuous type	32 bit	9
	DBINP	Pulse type	32 bit	9

<table>
<tr><td rowspan="6">Operand</td><td colspan="14">Setting Data</td><td colspan="5">Data Type</td></tr>
<tr><td colspan="14">S: Save conversion source (decimal number) word soft component number of data</td><td colspan="5">16/32 bit</td></tr>
<tr><td colspan="14">D: Word soft component number of conversion destination (2-digit)</td><td colspan="5">16/32 bit</td></tr>
<tr><td colspan="19">Operand Object Soft Component</td></tr>
<tr><td colspan="7">Bit Soft Component</td><td colspan="8">Word Soft Component</td><td colspan="4">Others</td></tr>
<tr><td>X</td><td>Y</td><td>M</td><td>T</td><td>C</td><td>S</td><td>D.b</td><td>KnX</td><td>KnY</td><td>KnM</td><td>KnS</td><td>T</td><td>C</td><td>D</td><td>V, Z</td><td>H</td><td>K</td><td>E</td><td>P</td></tr>
<tr><td>S</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td>●</td><td>●</td><td>●</td><td>●</td><td>●</td><td>●</td><td>●</td><td>●</td><td></td><td></td><td></td><td></td></tr>
<tr><td>D</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td>●</td><td>●</td><td>●</td><td>●</td><td>●</td><td>●</td><td>●</td><td></td><td></td><td></td><td></td></tr>
</table>

Function and Action Description

16-bit Operation (BIN, BINP)

Convert B's BCD (decimal) data to BIN (binary) data and transfer it to D.

- S data can be converted in the range of 0 ~ 9,999 (BCD).
- When specifying the number of digits for S and D, refer to the table below.

S	Number of Digits	Data Range
K1X000	1 digit	0 ~ 9
K2X000	2 digits	00 ~ 99
K3X000	3 digits	000 ~ 999
K4X000	4 digits	0000 ~ 9999

32-bit Operation (DBCD, DBCDP)

Convert BCD (decimal) data of [S+1,S] to BIN (binary) data and transfer it to [D+1,D].

- The data of [S+1,S] can be converted to a range class conversion of 0 ~ 99,999,999 (BCD).
- When [S+1,S] and [D+1,D] specify the number of digits, refer to the table below.

[D+1,D]	Number of Digits	Data Range
K1X000	1 digit	0 ~ 9
K2X000	2 digits	00 ~ 99
K3X000	3 digits	000 ~ 999
K4X000	4 digits	0000 ~ 9999
K5X000	5 digits	00000 ~ 99999
K6X000	6 digits	000000 ~ 999999
K7X000	7 digits	0,000,000 ~ 9,999,999
K8X000	8 digits	00,000,000 ~ 99,999,999

Note

Note		Description
1	About the input and output processing of BCD	Four arithmetic operations (+ - × ÷) and operations such as adding one or subtracting one instruction are performed in BIN (binary number). • When reading the digital switch information of BCD (decimal number) into the intelligent controller, use BCD→BIN conversion transfer command of BIN (FN 19).

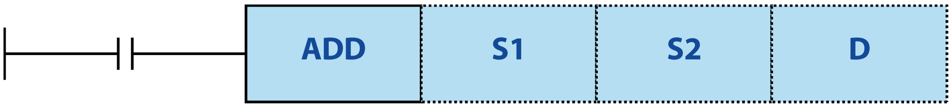
4.3 Four Logical Operations - FN 20 ~ FN 29

FN No.	Instruction Mark	Instruction Format	Function	Chapter	Page
20	ADD	ADD (S1) (S2) (D) ADDP (S1) (S2) (D) DADD (S1) (S2) (D) DADDP (S1) (S2) (D)	BIN addition	4.3.1	61
21	SUB	SUB (S1) (S2) (D) SUBP (S1) (S2) (D) DSUB (S1) (S2) (D) DSUBP (S1) (S2) (D)	BIN subtraction	4.3.2	62
22	MUL	MUL (S1) (S2) (D) MULP (S1) (S2) (D) DMUL (S1) (S2) (D) DMULP (S1) (S2) (D)	BIN multiplication	4.3.3	63
23	DIV	DIV (S1) (S2) (D) DIVP (S1) (S2) (D) DDIV (S1) (S2) (D) DDIVP (S1) (S2) (D)	BIN division	4.3.4	64
24	INC	INC (D) INCP (D) DINC (D) DINCP (D)	BIN plus one	4.3.5	65
25	DEC	DEC (D) DECP (D) DDEC (D) DDECP (D)	BIN minus one	4.3.6	66
26	WAND	WAND (S1) (S2) (D) WANDP (S1) (S2) (D) DWAND (S1) (S2) (D) DWANDP (S1) (S2) (D)	Logic AND	4.3.7	67
27	WOR	WOR (S1) (S2) (D) WORP (S1) (S2) (D) DWOR (S1) (S2) (D) DWORP (S1) (S2) (D)	Logic OR	4.3.8	68
28	WXOR	WXOR (S1) (S2) (D) WXORP (S1) (S2) (D) DWXOR (S1) (S2) (D) DWXORP (S1) (S2) (D)	Logic XOR	4.3.9	69
29	NEG	NEG (D) NEGP (D) DNEG (D) DNEGP (D)	Complement code	4.3.10	70

4.3.1 FN 20 - ADD/BIN Addition

Outline

Two values are added (A + B = C) to get the result of the instruction.



BIN Addition FN20 - ADD	Instruction Mark	Execution Condition	Instruction Type	Instruction Step
	ADD	Continuous type	16 bit	7
	ADDP	Pulse type	16 bit	7
	DADD	Continuous type	32 bit	13
	DADDP	Pulse type	32 bit	13

Operand	Setting Data	Data Type
	S1: The data of the addition operation, or the word soft component number of the saved data	16/32 bit
	S2: The data of the addition operation, or the word soft component number of the saved data	16/32 bit
	D: The word soft component number in which the result of the addition operation is saved	16/32 bit

Operand	Operand Object Soft Component																		
	Bit Soft Component							Word Soft Component								Others			
	X	Y	M	T	C	S	D.b	KnX	KnY	KnM	KnS	T	C	D	V, Z	H	K	E	P
S1								●	●	●	●	●	●	●	●	●	●		
S2								●	●	●	●	●	●	●	●	●	●		
D									●	●	●	●	●	●	●				

Function and Action Description

16-bit Operation (ADD, ADDP)	32-bit Operation (DADD, DADDP)
The contents of S1 and S2 are binary added and then transferred to D. • The highest bit of each data is a sign bit, and the data is added algebraically (eg: 5 + (-8) = -3). • When a constant (K) is specified in S1 and S2, the BIN conversion is automatically performed.	The contents of [S1+1,S1] and [S2+1,S2] are binary added and then transferred to [D+1,D]. • The highest bit of each data is the sign bit, and the data is added algebraically (eg: 5,500 + (-8,540) = -3,040). • When a constant (K) is specified in [S1+1,S1] and [S2+1,S2], BIN conversion is automatically performed.

Related Soft Component

Soft Component	Name	Content
M8020	Zero	ON: When the operation result is 0. OFF: When the operation result is other than 0.
M8021	Borrow	ON: When the operation result is less than -32,768 (16-bit operation) or -2,147,483,648 (32-bit operation), the borrow flag is activated. OFF: The operation result is not less than -32,768 (16-bit operation) or -2,147,483,648 (32-bit operation).
M8022	Carry	ON: When the operation result is greater than 32,767 (16-bit operation) or 2,147,483,647 (32-bit operation), the carry flag is activated. OFF: The operation result is not greater than 32,767 (16-bit operation) or 2,147,483,647 (32-bit operation).

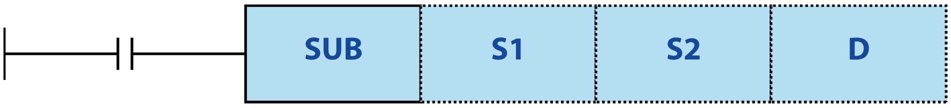
Note

Note		Description
1	When using the 32-bit operation (DADD, DADDP) instruction	In the designation of the word soft component, the soft component with the lower 16 bit side is specified, and the soft component with the consecutive number is the highest bit side. In order to not repeat the number, it is recommended to specify the soft component as an even number.
2	Designated as the same soft component in the source and destination operands	The source operand and the destination operand can also specify the same soft component number. In this case, if a continuous execution type instruction (ADD, DADD) is used, the result of the addition operation will change every operation cycle.

4.3.2 FN 21 - SUB/BIN Subtraction

Outline

Two values are subtracted (A - B = C) to get the result of the instruction.



BIN Subtraction FN21 - SUB	Instruction Mark	Execution Condition	Instruction Type	Instruction Step
	SUB	Continuous type	16 bit	7
	SUBP	Pulse type	16 bit	7
	DSUB	Continuous type	32 bit	13
	DSUBP	Pulse type	32 bit	13

Operand	Setting Data	Data Type
	S1: Data of subtraction, or word soft component number for saving data	16/32 bit
	S2: Data of subtraction, or word soft component number for saving data	16/32 bit
	D: Save the word soft component number of the subtraction result	16/32 bit

Operand	Operand Object Soft Component																		
	Bit Soft Component							Word Soft Component								Others			
	X	Y	M	T	C	S	D.b	KnX	KnY	KnM	KnS	T	C	D	V, Z	H	K	E	P
S1								●	●	●	●	●	●	●	●	●	●		
S2								●	●	●	●	●	●	●	●	●	●		
D									●	●	●	●	●	●	●				

Function and Action Description

16-bit Operation (SUB, SUBP)	32-bit Operation (DSUB, DSUBP)
The contents of S1 and S2 are binary subtracted and then transferred to D. • The most significant bit of each data is the sign bit, and the data is subtracted algebraically (eg: 5 - (-8) = 13). • When a constant (K) is specified in S1 and S2, the BIN conversion is automatically performed.	The contents of [S1+1,S1] and [S2+1,S2] are subjected to binary subtraction and then transferred to [D+1,D]. • The highest bit of each data is the sign bit, and the data is subdivided in algebraic way (eg: 5500 - (-8,540) = 14,040). • When a constant (K) is specified in [S1+1,S1] and [S2+1,S2], BIN conversion is automatically performed.

Related Soft Component

Soft Component	Name	Content
M8020	Zero	ON: When the operation result is 0. OFF: When the operation result is other than 0.
M8021	Borrow	ON: When the operation result is less than -32,768 (16-bit operation) or -2,147,483,648 (32-bit operation), the borrow flag is activated. OFF: The operation result is not less than -32,768 (16-bit operation) or -2,147,483,648 (32-bit operation).
M8022	Carry	ON: When the operation result is greater than 32,767 (16-bit operation) or 2,147,483,647 (32-bit operation), the carry flag is activated. OFF: The operation result is not greater than 32,767 (16-bit operation) or 2,147,483,647 (32-bit operation).

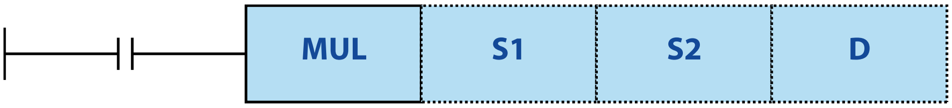
Note

Note		Description
1	When using the 32-bit operation (DSUB, DSUBP) instruction	In the designation of the word soft component, the soft component with the lower 16 bit side is specified, and the soft component with the consecutive number is the highest bit side. In order to not repeat the number, it is recommended to specify the soft component as an even number.
2	Designated as the same soft component in the source and destination operands	The source operand and the destination operand can also specify the same soft component number. In this case, if a continuous execution type instruction (SUB, DSUB) is used, the result of the addition operation will change every operation cycle.

4.3.3 FN 22 - MUL/BIN Multiplication

Outline

Two values are multiplied (A × B = C) to get the result of the instruction.



	Instruction Mark	Execution Condition	Instruction Type	Instruction Steps
BIN Multiplication FN22 - MUL	MUL	Continuous type	16 bit	7
	MULP	Pulse type	16 bit	7
	DMUL	Continuous type	32 bit	13
	DMULP	Pulse type	32 bit	13

Operand	Setting Data	Data Type
	S1: Data of the multiplication operation, or the word soft component number of the saved data	16/32 bit
	S2: Data of the multiplication operation, or the word soft component number of the saved data	16/32 bit
	D: Save the start word soft component number of the multiplication result	16/32 bit

Operand	Operand Object Soft Component																		
	Bit Soft Component							Word Soft Component								Others			
	X	Y	M	T	C	S	D.b	KnX	KnY	KnM	KnS	T	C	D	V, Z	H	K	E	P
S1								●	●	●	●	●	●	●	●	●	●		
S2								●	●	●	●	●	●	●	●	●	●		
D									●	●	●	●	●	●	●				

Function and Action Description

16-bit Operation (MUL, MULP)	32-bit Operation (DMUL, DMULP)
The contents of S1 and S2 are binary multiplied and transferred to the 32-bit (double word) of [D+1,D]. • The highest bit of each data is a sign bit, and the data is multiplied by algebra (eg: 5 × (-8) = -40). When a constant (K) is specified in S1 and S2, the BIN conversion is automatically performed. When [D+1,D] specifies the number of digits (K1 ~ 8), you can specify the number of digits from K1 to K8. • For example, when K2 is specified, only the lower 8 bits of the product (32 bits) are obtained.	The contents of [S1+1,S1] and [S2+1,S2] are binary-multiplied and transferred to 64 bits of [D+3,D+2,D+1,D] (word soft component × 4) in the middle. • The highest bit of each data is a sign bit, and the data is multiplied by algebra. (eg: 5,500 × (-8,540) = -46,970,000). When a constant (K) is specified in [S1+1,S1] and [S2+1,S2], BIN conversion is automatically performed. When the specified number of bits (K1 ~ 8) in [D+3,D+2,D+1,D], only the result of the lower 32 bits can be obtained, and the result of the upper 32 bits is not obtained. Please transmit the word to the word first. After the soft component is in, perform the operation.

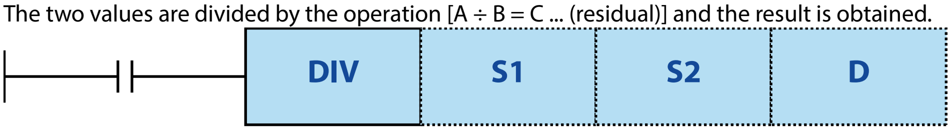
Related Soft Component

Soft Component	Name	Content
M8304	Zero	ON: When the operation result is 0. OFF: When the operation result is other than 0.

4.3.4 FN 23 - DIV/BIN Division

Outline

The two values are divided by the operation [A ÷ B = C ... (residual)] and the result is obtained.



BIN Division FN23 - DIV	Instruction Mark	Execution Condition	Instruction Type	Instruction Steps
	DIV	Continuous type	16 bit	7
	DIVP	Pulse type	16 bit	7
	DDIV	Continuous type	32 bit	13
	DDIVP	Pulse type	32 bit	13

Operand	Setting Data	Data Type
	S1: Data of the division operation, or the word soft component number (divided) of the saved data	16/32 bit
	S2: Data of division operation, or word soft component number (divisor) for saving data	16/32 bit
	D: Save the start word soft component number of the division result (quotient, remainder)	16/32 bit

Operand Object Soft Component	Bit Soft Component							Word Soft Component								Others			
	X	Y	M	T	C	S	D.b	KnX	KnY	KnM	KnS	T	C	D	V, Z	H	K	E	P
S1								●	●	●	●	●	●	●	●	●	●		
S2								●	●	●	●	●	●	●	●	●	●		
D									●	●	●	●	●	●	●				

Function and Action Description

16-bit Operation (DIV, DIVP)	32-bit Operation (DDIV, DDIVP)
The content of S1 is used as the divisor, the content of S2 is used as the divisor, the quotient is transmitted to D, and the remainder is transmitted to [D+1]. • The highest bit of each data is the sign bit, and the data is divided by algebraically. For example: (36 ÷ (-5) = -7 (quotient), 1 (remainder)). • The result of the operation (quotient, remainder) will occupy the soft component with the specified D starting to total 2 points, so please be careful not to repeat with the others control. • When a constant (K) is specified in S1 and S2, the BIN conversion is automatically performed.	The content of [S1+1,S1] is used as the divisor, the content of [S2+1,S2] is used as the divisor, the divided quotient is transmitted to [D+1,D], and the remainder is transmitted to [D+3,D+2] medium. • The highest bit of each data is the sign bit, and the data is divided by algebraically. For example: (5,500 ÷ (-540) = -10 (quotient), -100 (remainder)). • The result of the operation (quotient, remainder) will occupy the soft component with the specified D starting at 4 points, so be careful not to repeat it with other controls. • When a constant (K) is specified in [S1+1,S1] and [S2+1,S2], BIN conversion is automatically performed.

Related Soft Component

Soft Component	Name	Content
M8304	Zero	ON: When the operation result is 0. OFF: When the operation result is other than 0.
M8306	Carry	ON: When the operation result is greater than 32,767 (16-bit operation) or 2,147,483,647 (32-bit operation), the carry flag is activated. OFF: The operation result is not greater than 32,767 (16-bit operation) or 2,147,483,647 (32-bit operation).

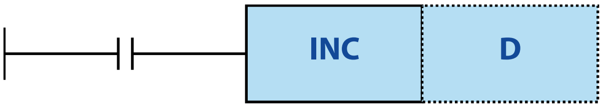
Error

Error	
1	When the divisor is 0, an operation error occurs and the instruction cannot be executed. When the operation result exceeds 32,767 (16-bit operation) or 2,147,483,647 (32-bit operation), an operation error occurs (the carry flag is also ON).

4.3.5 FN 24 - INC/BIN Plus One

Outline

Add "1" (+1 addition) to the specified soft component data.



BIN Plus One FN24 - INC	Instruction Mark	Execution Condition	Instruction Type	Instruction Step
	INC	Continuous type	16 bit	3
	INCP	Pulse type	16 bit	3
	DINC	Continuous type	32 bit	5
	DINCP	Pulse type	32 bit	5

Operand	Setting Data														Data Type				
	D: Save the word soft component number to which one data is added														16/32 bit				
	Operand Object Soft Component																		
	Bit Soft Component							Word Soft Component								Others			
	X	Y	M	T	C	S	D.b	KnX	KnY	KnM	KnS	T	C	D	V, Z	H	K	E	P
D									●	●	●	●	●	●	●				

Function and Action Description

16-bit Operation (INC, INCP)	32-bit Operation (DINC, DINCP)
After the content of D is added to an operation, it is transferred to D.	After adding the operation of [D+1,D], it is transferred to [D+1,D].

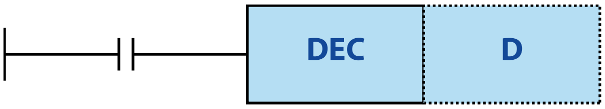
Note

Note		Description
1	Continuous execution instruction	In the continuous execution type instruction, each operation cycle performs an additional operation, so be sure to pay attention.
2	Action on the flag	16-bit operation: After adding +1 to +32,767, it becomes -32,768, but the flag bit (zero, borrow, carry) does not work. 32-bit operation: After adding 1 to +2,147,483,647, it becomes -2,147,483,648, but the flag bit (zero, borrow, carry) does not work.

4.3.6 FN 25 - DEC/BIN Minus One

Outline

The specified soft component data is decremented by "1" (-1 addition).



	Instruction Mark	Execution Condition	Instruction Type	Instruction Step
BIN Minus One **FN25 - DEC**	DEC	Continuous type	16 bit	3
	DECP	Pulse type	16 bit	3
	DDEC	Continuous type	32 bit	5
	DDECP	Pulse type	32 bit	5

<table>
<tr><td rowspan="5"><b>Operand</b></td><td colspan="14">Setting Data</td><td colspan="6">Data Type</td></tr>
<tr><td colspan="14">D: Save the word soft component number that is decremented by one data</td><td colspan="6">16/32 bit</td></tr>
<tr><td colspan="20">Operand Object Soft Component</td></tr>
<tr><td colspan="7">Bit Soft Component</td><td colspan="9">Word Soft Component</td><td colspan="4">Others</td></tr>
<tr><td>X</td><td>Y</td><td>M</td><td>T</td><td>C</td><td>S</td><td>D.b</td><td>KnX</td><td>KnY</td><td>KnM</td><td>KnS</td><td>T</td><td>C</td><td>D</td><td>V, Z</td><td>H</td><td>K</td><td>E</td><td>P</td></tr>
<tr><td>D</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td>●</td><td>●</td><td>●</td><td>●</td><td>●</td><td>●</td><td>●</td><td></td><td></td><td></td><td></td></tr>
</table>

Function and Action Description

16-bit Operation (DEC, DECP)	32-bit Operation (DDEC, DDECP)
After the content of D is decremented by one operation, it is transferred to D.	After the content of [D+1,D] is decremented by one operation, it is transferred to [D+1,D].

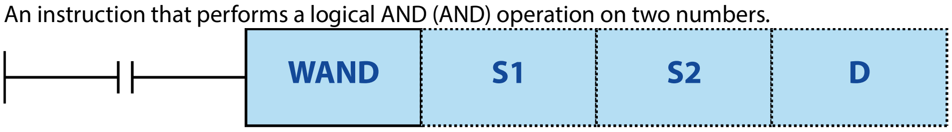
Note

Note		Description
1	Action on the flag	16-bit operation: After decrementing by 1 on -32,768, it becomes +32,767, but the flag bit (zero, borrow, carry) does not operate. 32-bit operation: After decrementing by 1 on -2,147,483,648, it becomes +2,147,483,647, but the flag bit (zero, borrow, carry) does not work.

4.3.7 FN 26 - WAND/Logic And

Outline

An instruction that performs a logical AND (AND) operation on two numbers.



Logic And FN26 - WAND	Instruction Mark	Execution Condition	Instruction Type	Instruction Steps
	WAND	Continuous type	16 bit	7
	WANDP	Pulse type	16 bit	7
	DAND	Continuous type	32 bit	13
	DANDP	Pulse type	32 bit	13

Operand	Setting Data	Data Type
	S1: Logic and data or word soft component number for saving data	16/32 bit
	S2: Logic and data or word soft component number for saving data	16/32 bit
	D: Word soft component number that holds the logic and result	16/32 bit

Operand	Operand Object Soft Component																		
	Bit Soft Component							Word Soft Component								Others			
	X	Y	M	T	C	S	D.b	KnX	KnY	KnM	KnS	T	C	D	V, Z	H	K	E	P
S1								●	●	●	●	●	●	●	●	●	●		
S2								●	●	●	●	●	●	●	●	●	●		
D									●	●	●	●	●	●	●				

Function and Action Description

16-bit Operation (WAND, WANDP)

The contents of S1 and S2 are logically ANDed in units of each, and then transferred to D.

- When the constant (K) is specified in the transfer sources S1 and S2, the BIN conversion is automatically performed.
- The logical AND operation is in bits, as shown in the following table (1∧1 = 1 0∧1 = 0 1∧0 = 0 0∧0 = 0).

	S1	S2	D WAND (FN 26) Instruction
Bit unit logic and operation	0	0	0
	1	0	0
	0	1	0
	1	1	1

32-bit Operation (DAND, DANDP)

The contents of [S1+1,S1] and [S2+1,S2] are logically ANDed in units of each, and then transferred to [D+1,D].

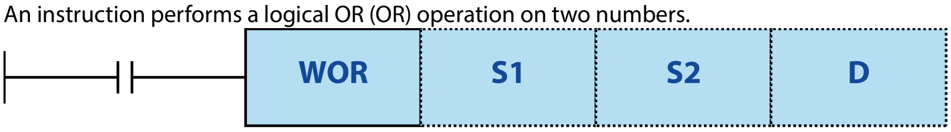
- When the constant (K) is specified in the transmission source [S1+1,S1] and [S2+1,S2], the BIN conversion is automatically performed.
- The logical AND operation is in bits, as shown in the following table (1∧1 = 1 0∧1 = 0 1∧0 = 0 0∧0 = 0).

	S1+1,S1	S2+1,S2	D+1,D DAND (FN 26) Instruction
Bit unit logic and operation	0	0	0
	1	0	0
	0	1	0
	1	1	1

4.3.8 FN 27 - WOR/Logic Or

Outline

An instruction performs a logical OR (OR) operation on two numbers.



	Instruction Mark	Execution Condition	Instruction Type	Instruction Steps
Logic Or **FN27 - WOR**	WOR	Continuous type	16 bit	7
	WORP	Pulse type	16 bit	7
	DOR	Continuous type	32 bit	13
	DORP	Pulse type	32 bit	13

	Setting Data	Data Type
Operand	S1: Logical soft component or data or word soft component number for saving data	16/32 bit
	S2: Logical soft component or data or word soft component number for saving data	16/32 bit
	D: Word soft component number to save logic or result	16/32 bit

	Operand Object Soft Component																			
	Bit Soft Component							Word Soft Component								Others				
	X	Y	M	T	C	S	D.b	KnX	KnY	KnM	KnS	T	C	D	V, Z	H	K	E	P
S1								●	●	●	●	●	●	●	●	●	●		
S2								●	●	●	●	●	●	●	●	●	●		
D									●	●	●	●	●	●	●				

Function and Action Description

16-bit Operation (WOR, WORP)

The contents of S1 and S2 are logically ORed in units of bits and transferred to D.

- When the constant (K) is specified in the transfer sources S1 and S2, the BIN conversion is automatically performed.
- The logical AND operation is in bits, as shown in the following table ($1 \vee 1 = 1$ $0 \vee 1 = 1$ $0 \vee 0 = 0$ $1 \vee 0 = 1$).

	S1	S2	D WOR (FN 27) Instruction
Bit unit logic and operation	0	0	0
	1	0	1
	0	1	1
	1	1	1

32-bit Operation (DOR, DORP)

The contents of [S1+1,S1] and [S2+1,S2] are logically ORed in units of bits and transferred to [D+1,D].

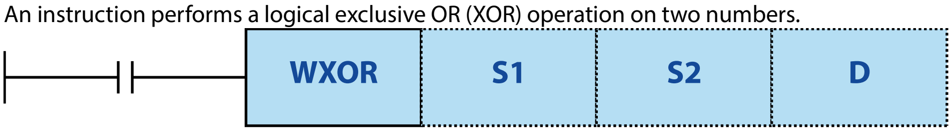
- When the constant (K) is specified in the transmission source [S1+1,S1] and [S2+1,S2], the BIN conversion is automatically performed.
- The logical OR operation is in bits, as shown in the following table ($1 \vee 1 = 1$ $0 \vee 1 = 1$ $0 \vee 0 = 0$ $1 \vee 0 = 1$).

	S1+1,S1	S2+1,S2	D+1,D WOR (FN 27) Instruction
Bit unit logic and operation	0	0	0
	1	0	1
	0	1	1
	1	1	1

4.3.9 FN 28 - WXOR/Logic XOR

Outline

An instruction performs a logical exclusive OR (XOR) operation on two numbers.



	Instruction Mark	Execution Condition	Instruction Type	Instruction Step
Logical XOR FN28 - WXOR	WXOR	Continuous type	16 bit	7
	WXORP	Pulse type	16 bit	7
	DXOR	Continuous type	32 bit	13
	DXORP	Pulse type	32 bit	13

Operand	Setting Data	Data Type
	S1: Data with logical XOR, or word soft component number for saving data	16/32 bit
	S2: Data with logical XOR, or word soft component number for saving data	16/32 bit
	D: Word soft component number that saves the logical XOR result	16/32 bit

Operand Object Soft Component	Bit Soft Component							Word Soft Component								Others			
	X	Y	M	T	C	S	D.b	KnX	KnY	KnM	KnS	T	C	D	V, Z	H	K	E	P
S1								●	●	●	●	●	●	●	●	●	●		
S2								●	●	●	●	●	●	●	●	●	●		
D									●	●	●	●	●	●	●				

Function and Action Description

16-bit Operation (WXOR, WXORP)

The contents of S1 and S2 are logically exclusive OR (XOR) in units of each, and then transferred to D.

- When the constant (K) is specified in the transfer sources S1 and S2, the BIN conversion is automatically performed.
- The logical XOR operation is in bits, as shown in the following table (1 ∀ 1 = 0 0 ∀ 0 = 0 1 ∀ 0 = 1 0 ∀ 1 = 1).

	S1	S2	D WXOR (FN 28) Instruction
Bit unit logic and operation	0	0	0
	1	0	1
	0	1	1
	1	1	0

32-bit Operation (DXOR, DXORP)

The contents of [S1+1,S1] and [S2+1,S2] are logically exclusive ORed (XOR) in units of each, and then transferred to [D+1,D].

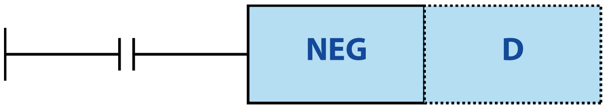
- When the constant (K) is specified in the transmission source [S1+1,S1] and [S2+1,S2], the BIN conversion is automatically performed.
- The logical OR operation is in bits, as shown in the following table (1 ∀ 1 = 0 0 ∀ 0 = 0 1 ∀ 0 = 1 0 ∀ 1 = 1).

	S1+1,S1	S2+1,S2	D+1,D WXOR (FN 28) Instruction
Bit unit logic and operation	0	0	0
	1	0	1
	0	1	1
	1	1	0

4.3.10 FN 29 - NEG/Complement Code

Outline

Find the instruction of the binary complement of the value (the value after each bit is inverted by +1).



	Instruction Mark	Execution Condition	Instruction Type	Instruction Step
Complement Code FN29 - NEG	NEG	Continuous type	16 bit	3
	NEGP	Pulse type	16 bit	3
	DNEG	Continuous type	32 bit	5
	DNEGP	Pulse type	32 bit	5

Operand	Setting Data														Data Type				
	D: The word soft component number of the data to be complemented, and the save destination soft component number (the operation result is stored in the same word soft component number)														16/32 bit				
	Operand Object Soft Component																		
	Bit Soft Component							Word Soft Component								Others			
	X	Y	M	T	C	S	D.b	KnX	KnY	KnM	KnS	T	C	D	V, Z	H	K	E	P
D									●	●	●	●	●	●	●				

Function and Action Description

16-bit Operation (NEG, NEGP)	32-bit Operation (DNEG, DNEGP)
The result of inverting each bit in the D content (0→1, 1→0) and adding one is saved to the original soft component.	The result of inverting each bit in the [D+1,D] content (0→1, 1→0) and adding one to the original soft component is saved.

Note

Note	
1	When using the continuous execution type (NEG, DNEG) instruction, each scan cycle (each calculation cycle) is executed, so be careful.

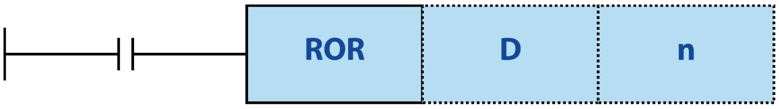
4.4 Cycles and Shift - FN 30 ~ FN 39

FN No.	Instruction Mark	Instruction Format	Function	Chapter	Page
30	ROR	ROR (D) (n) RORP (D) (n) DROR (D) (n) DRORP (D) (n)	Loop right shift	4.4.1	72
31	ROL	ROL (D) (n) ROLP (D) (n) DROL (D) (n) DROLP (D) (n)	Loop left shift	4.4.2	74
32	RCR	RCR (D) (n) RCRP (D) (n) DRCR (D) (n) DRCRP (D) (n)	Right shift of carry-in cycle	4.4.3	76
33	RCL	RCL (D) (n) RCLP (D) (n) DRCL (D) (n) DRCLP (D) (n)	Left shift of carry-in cycle	4.4.4	78
34	SFTR	SFTR (S) (D) (n1) (n2) SFTRP (S) (D) (n1) (n2)	Bit right shift	4.4.5	80
35	SFTL	SFTL (S) (D) (n1) (n2) SFTLP (S) (D) (n1) (n2)	Bit left shift	4.4.6	81
36	WSFR	WSFR (S) (D) (n1) (n2) WSFRP (S) (D) (n1) (n2)	Word right shift	4.4.7	82
37	WSFL	WSFL (S) (D) (n1) (n2) WSFLP (S) (D) (n1) (n2)	Word left shift	4.4.8	83
38	SFWR	SFWR (S) (D) (n1) SFWRP (S) (D) (n1)	Shift writing (FIFO/FIFO control)	4.4.9	84
39	SFRD	SFRD (S) (D) (n1) SFRDP (S) (D) (n1)	Shift readout (FIFO control)	4.4.10	85

4.4.1 FN 30 - ROR/Loop Right Shift

Outline

An instruction that cyclically shifts the specified number of bits of information that does not include the carry flag.



Loop Right Shift FN30 - ROR	Instruction Mark	Execution Condition	Instruction Type	Instruction Step
	ROR	Continuous type	16 bit	5
	RORP	Pulse type	16 bit	5
	DROR	Continuous type	32 bit	9
	DRORP	Pulse type	32 bit	9

Operand	Setting Data	Data Type
	D: Save the word soft component number of the right shift data	16/32 bit
	n: Number of bits of rotational movement [0 ≤ n ≤ 16 (16 bit command), 0 ≤ n ≤ 32 (32-bit command)]	16/32 bit

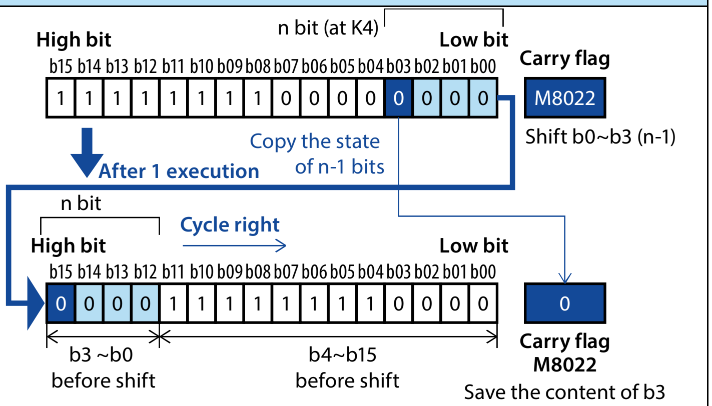
Operand Object Soft Component	Bit Soft Component						Word Soft Component									Others			
	X	Y	M	T	C	S	D.b	KnX	KnY	KnM	KnS	T	C	D	V, Z	H	K	E	P
D									●	●	●	●	●	●	●				
n														●		●	●		

Function and Action Description

16-bit Operation (ROR, RORP)

The n bit in the 16 bit of D is rotated to the right.

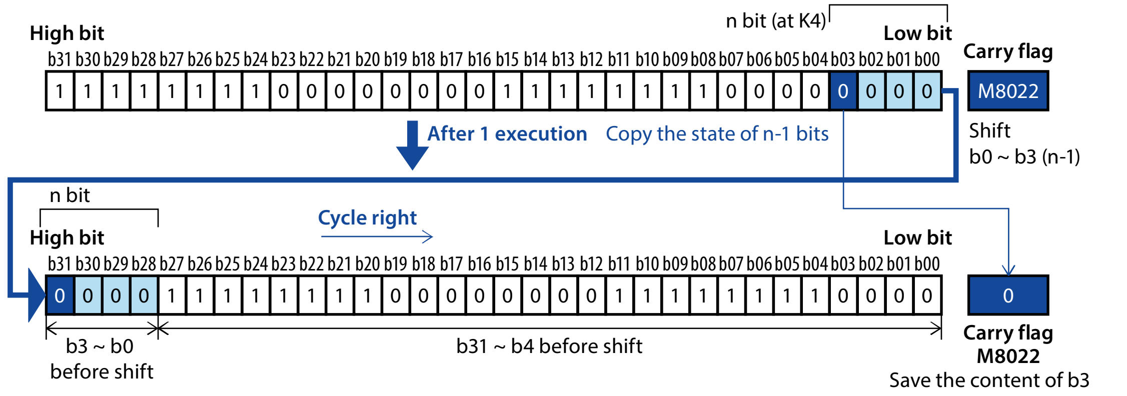
- The last bit is stored in the carry flag (M8022).
- When the number of bits is specified, K4 (16 bit command) is valid.



32-bit Operation (DROR, DRORP)

The n bits of the 32 bits of [D+1,D] are rotated to the right.

- The last bit is stored in the carry flag (M8022).
- In the case of a bit-specified soft component, K8 (32-bit instruction) is valid.



Related Soft Component

Soft Component	Name	Content
M8022	Carry	Finally, the bit from the lowest displacement is 1 when it is ON.

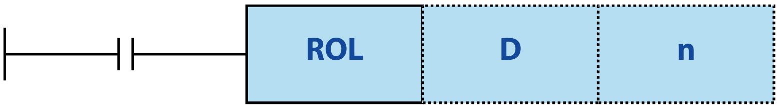
Note

Note		Description
1	Continuous execution type (ROR, DROR) instruction	Note that cyclic shifts are performed for each scan cycle (operation cycle).
2	When specifying the number of bits in D to specify the soft component	Only K4 (16 bit instruction) or K8 (32 bit instruction) is valid (eg: K4Y010, K8M0).

4.4.2 FN 31 - ROL/Loop Left Shift

Outline

An instruction that cyclically shifts the specified number of bits of information that does not include the carry flag.



Loop Left Shift FN31 - ROL	Instruction Mark	Execution Condition	Instruction Type	Instruction Step
	ROL	Continuous type	16 bit	5
	ROLP	Pulse type	16 bit	5
	DROL	Continuous type	32 bit	9
	DROLP	Pulse type	32 bit	9

Operand	Setting Data	Data Type
	D: Save the word soft component number of the left shift data	16/32 bit
	n: Number of bits of rotational movement [n ≤ 16 (16 bit command), n ≤ 32 (32 bit command)]	16/32 bit

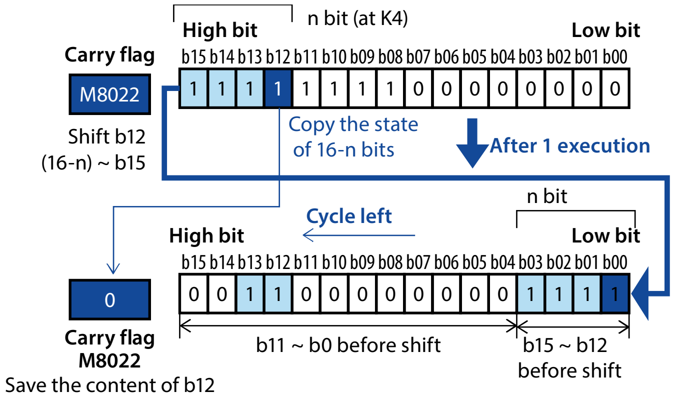
Operand Object Soft Component	Bit Soft Component							Word Soft Component								Others			
	X	Y	M	T	C	S	D.b	KnX	KnY	KnM	KnS	T	C	D	V, Z	H	K	E	P
D									●	●	●	●	●	●	●				
n														●		●	●		

Function and Action Description

16-bit Operation (ROL, ROLP)

The n bit of the 16 bit of D is rotated to the left.

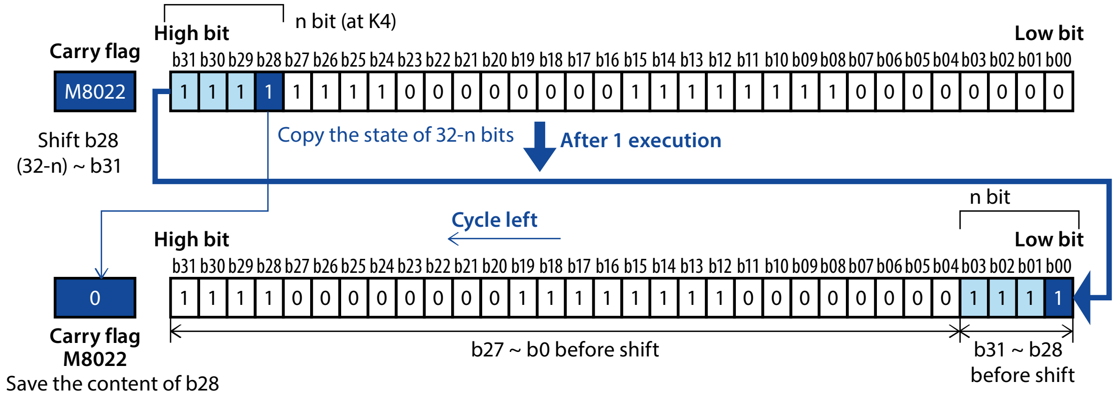
- The last bit is stored in the carry flag (M8022).
- When the number of bits is specified, K4 (16 bit command) is valid.



32-bit Operation (DROL, DROLP)

The n bits of the 32 bits of [D+1,D] are shifted left.

- The last bit is stored in the carry flag (M8022).
- In the case of a bit-specified soft component, K8 (32-bit instruction) is valid.



Related Soft Component

Soft Component	Name	Content
M8022	Carry	Finally, when the bit from the highest displacement is 1, it is ON.

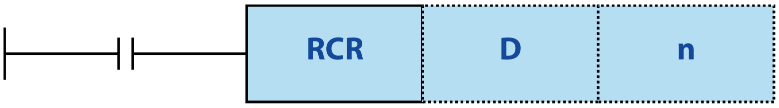
Note

Note		Description
1	Continuous execution type (ROL, DROL) instruction	Note that cyclic shifts are performed for each scan cycle (operation cycle).
2	When specifying the number of bits in D to specify the soft component	Only K4 (16 bit instruction) or K8 (32 bit instruction) is valid (eg: K4Y010, K8M0).

4.4.3 FN 32 - RCR/Right Shift of Carry-in Cycle

Outline

An instruction that rotates the specified number of bits including the carry flag to the right.



<table>
<tr><th></th><th>Instruction Mark</th><th>Execution Condition</th><th>Instruction Type</th><th>Instruction Steps</th></tr>
<tr><td>Right Shift of
Carry-in Cycle
FN32 - RCR</td><td>RCR</td><td>Continuous type</td><td>16 bit</td><td>5</td></tr>
<tr><td></td><td>RCRP</td><td>Pulse type</td><td>16 bit</td><td>5</td></tr>
<tr><td></td><td>DRCR</td><td>Continuous type</td><td>32 bit</td><td>9</td></tr>
<tr><td></td><td>DRCR P</td><td>Pulse type</td><td>32 bit</td><td>9</td></tr>
</table>

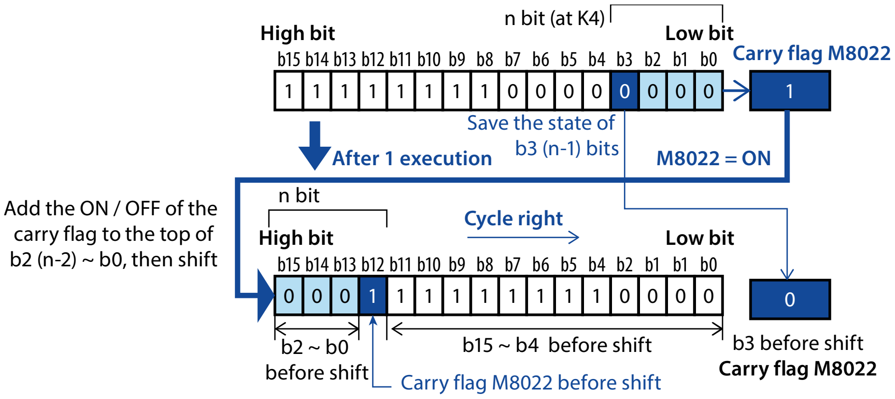
<table>
<tr><td rowspan="6">Operand</td><td colspan="14">Setting Data</td><td colspan="6">Data Type</td></tr>
<tr><td colspan="14">D: Save the word soft component number of the right shift data</td><td colspan="6">16/32 bit</td></tr>
<tr><td colspan="14">n: Number of bits of rotational movement [n ≤ 16 (16 bit command), n ≤ 32 (32 bit command)]</td><td colspan="6">16/32 bit</td></tr>
<tr><td colspan="20">Operand Object Soft Component</td></tr>
<tr><td colspan="7">Bit Soft Component</td><td colspan="8">Word Soft Component</td><td colspan="5">Others</td></tr>
<tr><td>X</td><td>Y</td><td>M</td><td>T</td><td>C</td><td>S</td><td>D.b</td><td>KnX</td><td>KnY</td><td>KnM</td><td>KnS</td><td>T</td><td>C</td><td>D</td><td>V, Z</td><td>H</td><td>K</td><td>E</td><td>P</td></tr>
<tr><td>D</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td>●</td><td>●</td><td>●</td><td>●</td><td>●</td><td>●</td><td>●</td><td></td><td></td><td></td><td></td></tr>
<tr><td>n</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td>●</td><td></td><td>●</td><td>●</td><td></td><td></td></tr>
</table>

Function and Action Description

16-bit Operation (RCR, RCRP)

The 16 bit +1 bit of D (carry flag M8022) shifts n bits to the right.

- Because there is a carry flag in the loop, if the M8022 is turned ON or OFF before the cyclic shift instruction is executed, it will be sent to the destination operand.

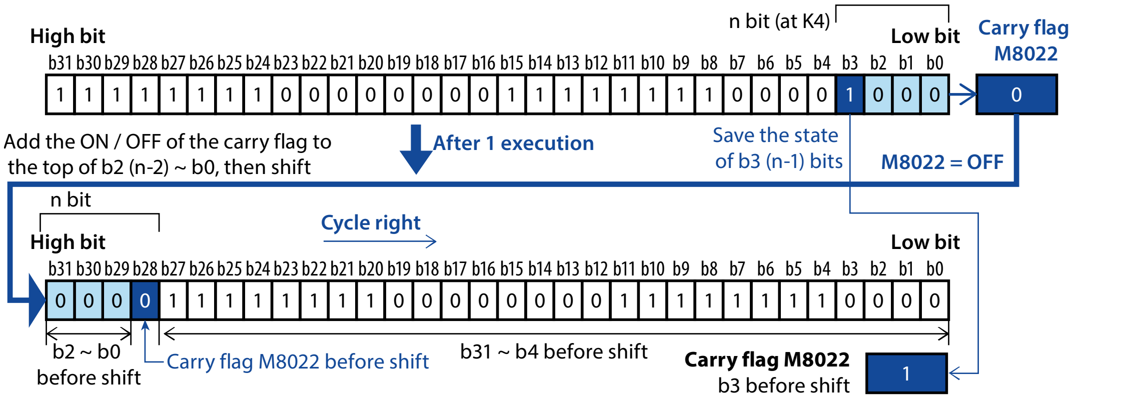


b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0	Carry flag M8022
1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	1

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0	Carry flag M8022
0	0	0	1	1	1	1	1	1	1	1	1	0	0	0	0	0

32-bit Operation (DRCR, DRCRP)

32 bits +1 bit of [D+1,D] (carry flag M8022) moves n bits to the right.



b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0	Carry flag M8022
1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	0	0	0	0	1	0	0	0	0

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0	Carry flag M8022
0	0	0	0	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	0	0	0	0	1

Related Soft Component

Soft Component	Name	Content
M8022	Carry	Finally, the bit from the lowest displacement is 1 when it is ON.

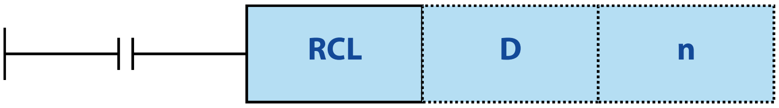
Note

Note		Description
1	Continuous execution type (RCR, DRCR) instruction	Note that cyclic shifts are performed for each scan cycle (operation cycle).
2	When specifying the number of bits in D to specify the soft component	Only K4 (16 bit instruction) or K8 (32 bit instruction) is valid (eg: K4Y010, K8M0).

4.4.4 FN 33 - RCL/Left Shift of Carry-in Cycle

Outline

An instruction that cyclically shifts the specified number of bits including the carry flag.



	Instruction Mark	Execution Condition	Instruction Type	Instruction Steps
Left Shift of Carry-in Cycle FN33 - RCL	RCL	Continuous type	16 bit	5
	RCLP	Pulse type	16 bit	5
	DRCL	Continuous type	32 bit	9
	DRCLP	Pulse type	32 bit	9

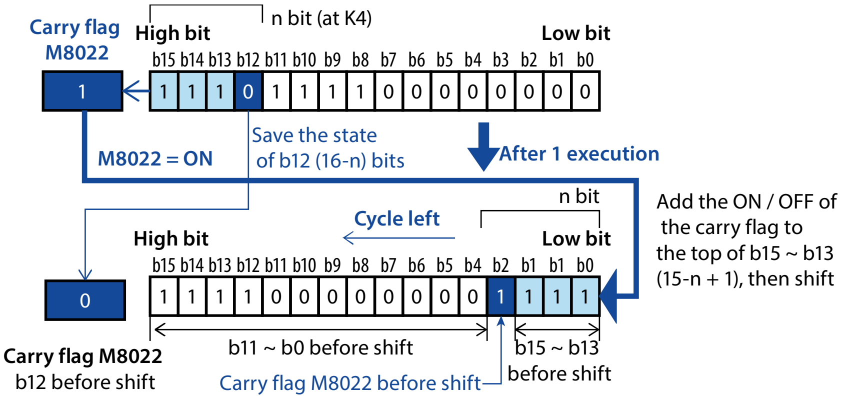
Operand	Setting Data															Data Type			
	D: Save the word soft component number of the left shift data															16/32 bit			
	n: Number of bits of rotational movement [n ≤ 16 (16 bit command), n ≤ 32 (32 bit command)]															16/32 bit			
	Operand Object Soft Component																		
	Bit Soft Component							Word Soft Component								Others			
	X	Y	M	T	C	S	D.b	KnX	KnY	KnM	KnS	T	C	D	V, Z	H	K	E	P
D									●	●	●	●	●	●	●				
n														●		●	●		

Function and Action Description

16-bit Operation (RCL, RCLP)

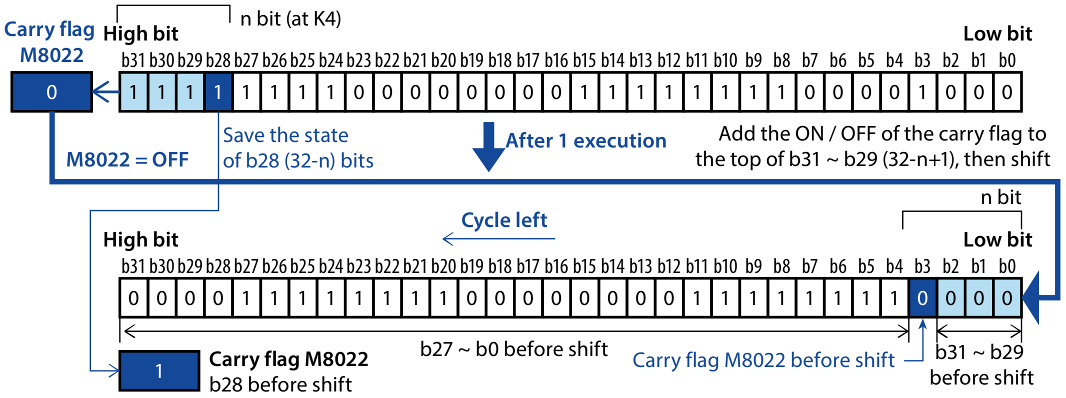
The 16 bit +1 bit of D (carry flag M8022) is shifted to the left by n bits.

- Because there is a carry flag in the loop, if the M8022 is turned ON or OFF before the cyclic shift instruction is executed, it will be sent to the destination operand.


Carry flag M8022
High bit
n bit (at K4)
Low bit
b15 b14 b13 b12 b11 b10 b9 b8 b7 b6 b5 b4 b3 b2 b1 b0
1
1 1 1 0 1 1 1 1 0 0 0 0 0 0 0 0
M8022 = ON
Save the state of b12 (16-n) bits
After 1 execution
n bit
Cycle left
High bit
Low bit
b15 b14 b13 b12 b11 b10 b9 b8 b7 b6 b5 b4 b2 b1 b1 b0
0
1 1 1 1 0 0 0 0 0 0 0 0 1 1 1 1
Add the ON / OFF of the carry flag to the top of b15 ~ b13 (15-n + 1), then shift
Carry flag M8022
b12 before shift
b11 ~ b0 before shift
b15 ~ b13 before shift
Carry flag M8022 before shift

32-bit Operation (DRCL, DRCLP)

32 bits +1 bit of [D+1,D] (carry flag M8022) moves n bits to the left.


Carry flag M8022
High bit
n bit (at K4)
Low bit
b31 b30 b29 b28 b27 b26 b25 b24 b23 b22 b21 b20 b19 b18 b17 b16 b15 b14 b13 b12 b11 b10 b9 b8 b7 b6 b5 b4 b3 b2 b1 b0
0
1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 0 0 0 0 1 0 0 0
M8022 = OFF
Save the state of b28 (32-n) bits
After 1 execution
Add the ON / OFF of the carry flag to the top of b31 ~ b29 (32-n+1), then shift
n bit
Cycle left
High bit
Low bit
b31 b30 b29 b28 b27 b26 b25 b24 b23 b22 b21 b20 b19 b18 b17 b16 b15 b14 b13 b12 b11 b10 b9 b8 b7 b6 b5 b4 b3 b2 b1 b0
0 0 0 0 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 0 0 0 0
1
Carry flag M8022
b28 before shift
b27 ~ b0 before shift
Carry flag M8022 before shift
b31 ~ b29 before shift

Related Soft Component

Soft Component	Name	Content
M8022	Carry	Finally, the bit from the lowest displacement is 1 when it is ON.

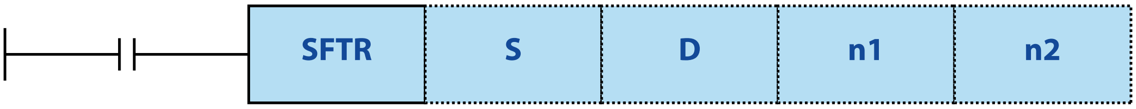
Note

Note		Description
1	Continuous execution type (RCL, DRCL) instructions	Note that cyclic shifts are performed for each scan cycle (operation cycle).
2	When specifying the number of bits in D to specify the soft component	Only K4 (16 bit instruction) or K8 (32 bit instruction) is valid (eg: K4Y010, K8M0).

4.4.5 FN 34 - SFTR/Bit Right Shift

Outline

An instruction merges the soft component to the right.



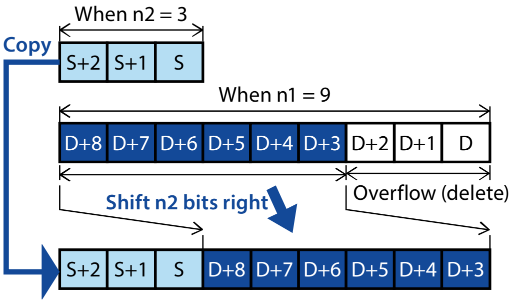
	Instruction Mark	Execution Condition	Instruction Type	Instruction Step
Bit Right Shift FN34 - SFTR	SFTR	Continuous type	16 bit	9
	SFTRP	Pulse type	16 bit	9

Operand	Setting Data	Data Type
	S: Start bit soft component number saved in the shift data after shifting right	16 bit
	D: Start bit soft component number shifted right	16 bit
	N1: Bit data length of shift data n2 ≤ n1 ≤ 1024	16 bit
	n: Number of sites shifted to the right n2 ≤ n1 ≤ 1024	16 bit

Operand	Operand Object Soft Component																		
	Bit Soft Component							Word Soft Component								Others			
	X	Y	M	T	C	S	D.b	KnX	KnY	KnM	KnS	T	C	D	V, Z	H	K	E	P
S	●	●	●			●	●								●				
D		●	●			●									●				
n1																●	●		
n2														●		●	●		

Function and Action Description

16-bit Operation (SFTR, SFTRP)
For the n1 bit (shift register length) data starting with D, shift n2 bits to the right; After shifting, transfer S start n2 bit data to n2 bits starting from D+n1-n2.



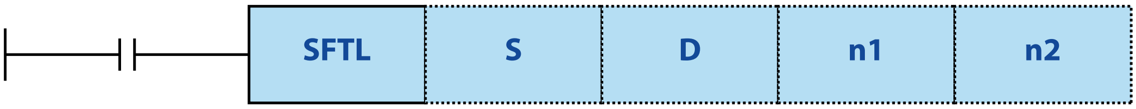
Note

Note	
1	In the SFTRP instruction, the n2 shift bit is executed each time the command input changes from OFF to ON. However, please note that in the SFTR instruction, shift is performed every scan cycle (operation cycle).

4.4.6 FN 35 - SFTL/Bit Left Shift

Outline

An instruction shifts a bit soft component of a specified length to the left by a specified bit length each time. After moving, the S bit soft component of length n2 points is transmitted from the lowest bit.

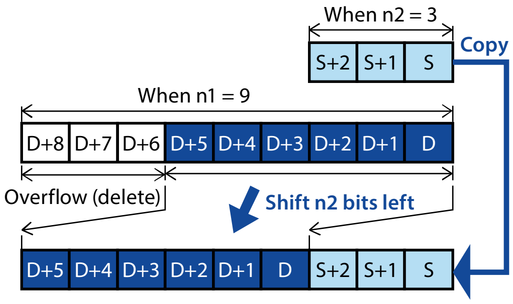


Bit Left Shift FN35 - SFTL	Instruction Mark	Execution Condition	Instruction Type	Instruction Step
	SFTL	Continuous type	16 bit	9
	SFTLP	Pulse type	16 bit	9

Operand	Setting Data	Data Type
	S: Start bit soft component number saved in shift data after shifting left	16 bit
	D: Start bit soft component number shifted to the left	16 bit
	n1: Bit data length of shift data n2 ≤ n1 ≤ 1024	16 bit
	n2: Number of left shifts n2 ≤ n1 ≤ 1024	16 bit

Operand	Operand Object Soft Component																		
	Bit Soft Component							Word Soft Component								Others			
	X	Y	M	T	C	S	D.b	KnX	KnY	KnM	KnS	T	C	D	V, Z	H	K	E	P
S	●	●	●			●	●								●				
D		●	●			●									●				
n1																●	●		
n2														●		●	●		

Function and Action Description

16-bit Operation (SFTL, SFTLP)
For the n1 bit (the length of the shift register) starting with D, shift n2 bits to the left; After shifting, transfer S to start n2 bits of data to the n2 bit starting from D. 

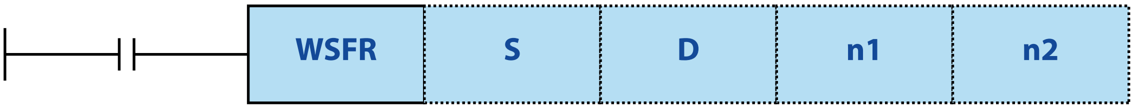
Note

Note	
1	In the SFTLP instruction, the n2 shift bit is executed each time the command input changes from OFF to ON. However, please note that in the SFTL instruction, the shift is performed every scan cycle (operation cycle).

4.4.7 FN 36 - WSFR/Word Right Shift

Outline

Move n1 word-length word soft components to the right by n2 words.



	Instruction Mark	Execution Condition	Instruction Type	Instruction Step
Word Right Shift FN36 - WSFR	WSFR	Continuous type	16 bit	9
	WSFRP	Pulse type	16 bit	9

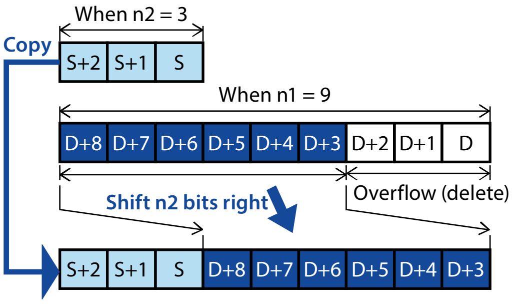
	Setting Data	Data Type
Operand	S: Start bit soft component number saved in shift data after shifting right	16 bit
	D: Save the start word soft component number of the right shift data	16 bit
	n1: The length of the word data of the shifted data is n2 ≤ n1 ≤ 512	16 bit
	n2: Number of words shifted to the right n2 ≤ n1 ≤ 512	16 bit

	Operand Object Soft Component																		
	Bit Soft Component							Word Soft Component								Others			
	X	Y	M	T	C	S	D.b	KnX	KnY	KnM	KnS	T	C	D	V, Z	H	K	E	P
S								●	●	●	●	●	●	●	●				
D									●	●	●	●	●	●	●				
n1																●	●		
n2														●		●	●		

Function and Action Description

16-bit Operation (WSFR, WSFRP)

For n1 word soft components starting with D, shift n2 words to the right; After shifting, the S start n2 bit data is transferred to the n2 point starting from [D+n1-n2].



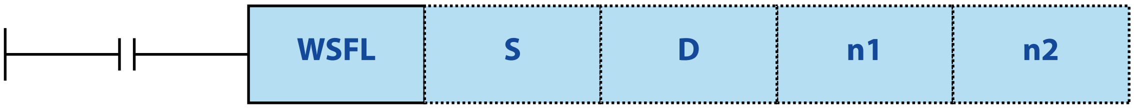
Note

Note	
1	After the drive input is ON in the WSFRP instruction, move n2 words. However, the movement is performed every scan cycle in the WSFR instruction, so be sure to pay attention.

4.4.8 FN 37 - WSFL/Word Left Shift

Outline

Move the n1 word-length word soft component to the left by n2 words.

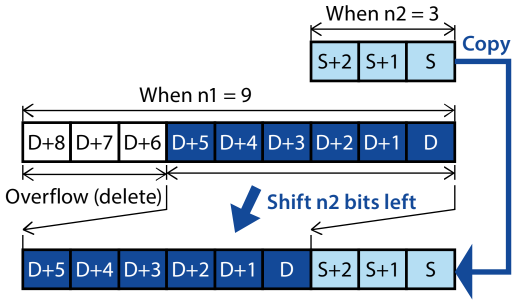


Word Left Shift FN37 - WSFL	Instruction Mark	Execution Condition	Instruction Type	Instruction Steps
	WSFL	Continuous type	16 bit	9
	WSFLP	Pulse type	16 bit	9

Operand	Setting Data	Data Type
	S: Start bit soft component number saved in shift data after shifting left	16 bit
	D: Save the start word soft component number of the left shift data	16 bit
	n1: The length of the word data of the shifted data is n2 ≤ n1 ≤ 512	16 bit
	n2: Number of words shifted to the left n2 ≤ n1 ≤ 512	16 bit

Operand Object Soft Component	Bit Soft Component							Word Soft Component								Others			
	X	Y	M	T	C	S	D.b	KnX	KnY	KnM	KnS	T	C	D	V, Z	H	K	E	P
S								●	●	●	●	●	●	●	●				
D									●	●	●	●	●	●	●				
n1																●	●		
n2														●		●	●		

Function and Action Description

16-bit Operation (WSFL, WSFLP)	
For the n1 word soft components starting with D, shift n2 words to the left; After shifting, transfer S to start n2 bits of data to the n2 point starting from D.	

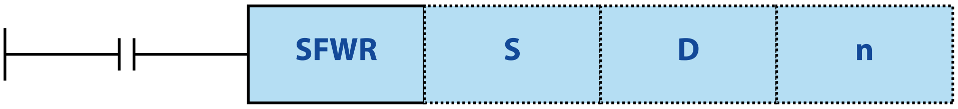
Note

Note	
1	In the WSFLP instruction, each time the instruction input changes from OFF to ON, the shift of n2 words is moved. However, in the WSFL instruction, the movement is performed every calculation cycle, so be sure to pay attention.

4.4.9 FN 38 - SFWR/Shift Writing

Outline

Data shift write instructions.



Shift Writing FN38 - SFWR	Instruction Mark	Execution Condition	Instruction Type	Instruction Steps
	SFWR	Continuous type	16 bit	7
	SFWRP	Pulse type	16 bit	7

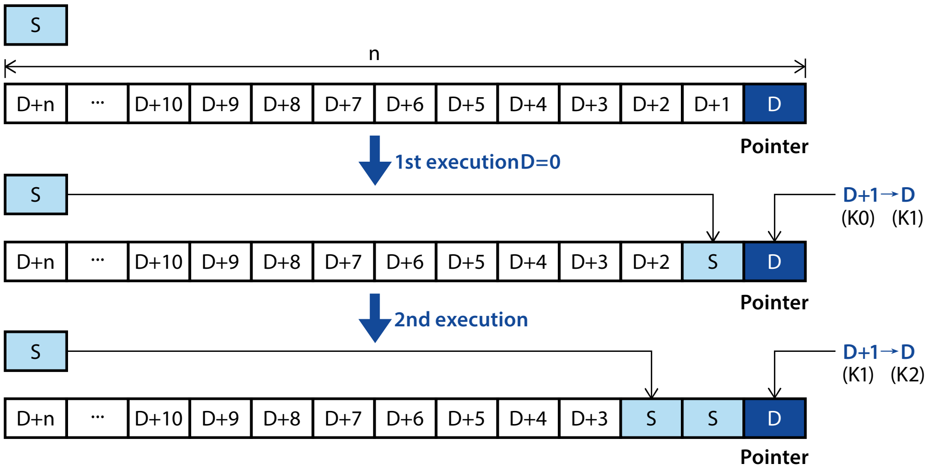
Operand	Setting Data	Data Type
	S: Save the word soft component number of the data you want to advance	16 bit
	D: The start word soft component number of the saved data (the front end is the pointer, and the data starts from D+1)	16 bit
	n: Specify the number of points of the saved data +1 (+1 is the part of the pointer) 2 ≤ n ≤ 512	16 bit

Operand	Operand Object Soft Component																		
	Bit Soft Component							Word Soft Component								Others			
	X	Y	M	T	C	S	D.b	KnX	KnY	KnM	KnS	T	C	D	V, Z	H	K	E	P
S								●	●	●	●	●	●	●	●	●	●		
D									●	●	●	●	●	●	●				
n																●	●		

Function and Action Description

16-bit Operation (SFWR, SFWRP)

- When the condition changes from OFF to ON, the content of S is saved to D+1, and the content of D+1 becomes the value of S.
- After the content of S changes and the input is executed again from OFF to ON, the content of S is saved to D+2, and the content of D+2 is changed to S (because of the continuous execution type instruction SFWR, each operation cycle It is saved in turn, so please use the pulse execution type command SFWRP to program).
- The following execution process is the same, executed from the right end, indicating the number of data save points in the contents of pointer D.



Related Soft Component

Soft Component	Name	Content
M8022	Carry	When the content of the pointer D exceeds n-1, it becomes no processing (no writing), and the carry flag M8022 turns ON.

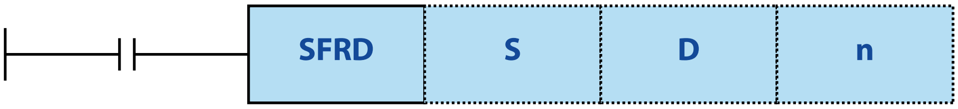
Note

Note	
1	In the case of continuous execution type (SFWR) instructions, please note that each scan cycle (operation cycle) is saved (overwritten) at a time.

4.4.10 FN 39 - SFRD/Shift Readout

Outline

Data shift readout instructions.



Shift Readout FN39 - SFRD	Instruction Mark	Execution Condition	Instruction Type	Instruction Steps
	SFRD	Continuous type	16 bit	7
	SFRDP	Pulse type	16 bit	7

Operand	Setting Data	Data Type
	S: The start word soft component number of the saved data (the front end is the pointer, and the data starts from S+1)	16 bit
	D: Word soft component number for saving first-out data	16 bit
	n: Specify the value of the number of points of the saved data +1 (+1 is the part of the pointer) (2 ≤ n ≤ 512)	16 bit

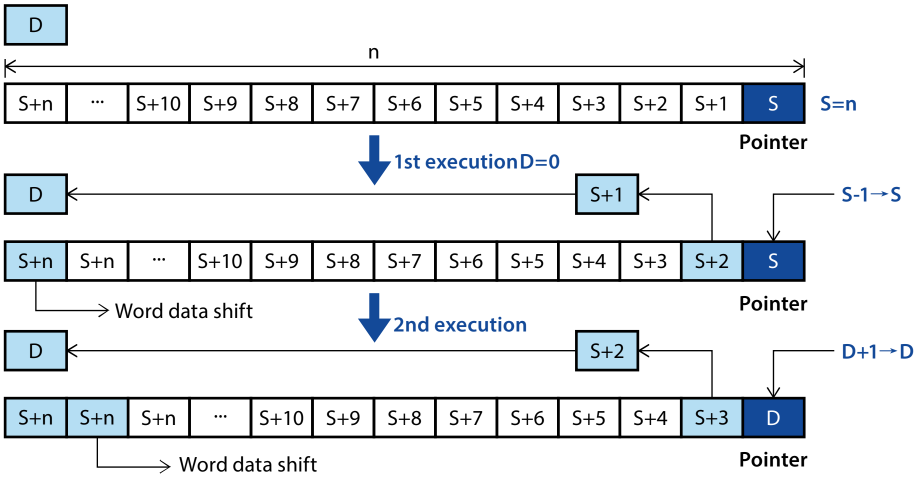
Operand	Operand Object Soft Component																	
	Bit Soft Component							Word Soft Component								Others			
	X	Y	M	T	C	S	D.b	KnX	KnY	KnM	KnS	T	C	D	V, Z	H	K	E	P
S									●	●	●	●	●	●	●				
D									●	●	●	●	●	●	●				
n																●	●		

Function and Action Description

16-bit Operation (SFRD, SFRDP)

The data starting from S+1 is sequentially transferred (read) to D, and the n-1 point data starting from S+1 is shifted word by word to the right, and the data saved in S-1.

- When the command contact bit is ON, the contents of S+1 are transferred (read) to D.
- At the same time, the content of the pointer S is reduced, and the data on the left side is shifted to the right of the word (because the SFRD is executed with the continuity execution type instruction, each operation cycle is shifted, so use the pulse execution type instruction SFRDP to program).



Related Soft Component

Soft Component	Name	Content
M8020	Zero	The data is read out, usually starting from S+1, but when the content of the pointer S is 0, the zero flag M8020 is activated.

Note

Note		Description
1	Execute the readed data	The content of S+n will not change due to reading.
2	Continuous execution type (SFRD) instruction	Please note that each scan cycle (operation cycle) will be in order, but the contents of S+n will not change.

4.5 Data Processing - FN 40 ~ FN 49

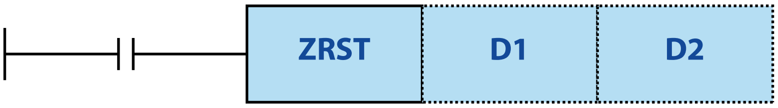
Compared to the basic application instructions of FN10 ~ FN39, the FN40 ~ FN49 instructions can be used for more complicated processing or as instructions for special purposes.

FN No.	Instruction Mark	Instruction Format	Function	Chapter	Page
40	ZRST	ZRST (D1) (D2) ZRSTP (D1) (D2)	Batch reset	4.5.1	88
41	DECO	DECO (S) (D) (n) DECOP (S) (D) (n)	Decoding	4.5.2	89
42	ENCO	ENCO (S) (D) (n) ENCOP (S) (D) (n)	Encoding	4.5.3	91
43	SUM	SUM (S) (D) SUMP (S) (D) DSUM (S) (D) DSUMP (S) (D)	Number of ON bit	4.5.4	92
44	BON	BON (S) (D) (n) BONP (S) (D) (n) DBON (S) (D) (n) DBONP (S) (D) (n)	Judgment of the ON bit	4.5.5	93
45	MEAN	MEAN (S) (D) (n) MEANP (S) (D) (n) DMEAN (S) (D) (n) DMEANP (S) (D) (n)	Average value	4.5.6	94
46	ANS	ANS (S) (m) (D)	Signal alarm set	4.5.7	95
47	ANR	ANR (-) ANRP (-)	Signal alarm reset	4.5.8	96
48	SQR	SQR (S) (D) SQRP (S) (D) DSQR (S) (D) DSQRP (S) (D)	BIN square	4.5.9	97
49	FLT	FLT (S) (D) FLTP (S) (D) DFLT (S) (D) DFLTP (S) (D)	BIN integer→binary floating point number conversion	4.5.10	98

4.5.1 FN 40 - ZRST/Batch Reset

Outline

A batch reset instruction is executed between two specified soft components.



Batch Reset FN40 - ZRST	Instruction Mark	Execution Condition	Instruction Type	Instruction Steps
	ZRST	Continuous type	16 bit	5
	ZRSTP	Pulse type	16 bit	5

Operand	Setting Data	Data Type
	D1: The leading digit/word soft component number of the batch reset	16/32 bit
	D2: Bit/word soft component number at the end of the batch reset	16/32 bit

Operand Object Soft Component	Bit Soft Component							Word Soft Component								Others			
	X	Y	M	T	C	S	D.b	KnX	KnY	KnM	KnS	T	C	D	V, Z	H	K	E	P
D1		●	●			●						●	●	●	●				
D2		●	●			●						●	●	●	●				

Function and Action Description

16-bit Operation (ZRST, ZRSTP)
Reset all the same types of D1 ~ D2. • When D1 to D2 are bit soft components, the soft component ranges of D1 to D2 are all written OFF (reset). • When D1 ~ D2 are word soft components, the soft component ranges of D1 ~ D2 are all written to K0.

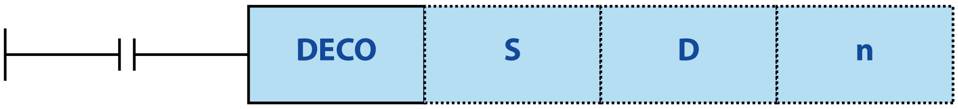
Note

Note		Description
1	Batch reset soft component	D1 and D2 must be specified as the same type of soft component, and D1 number ≤ D2 number. When the D1 number > D2 number, the instruction skips execution and reports an error (6705).
2	About the designation of counters (C0 ~ C255)	The ZRST instruction is a 16-bit processing instruction. It is also possible to specify a 32-bit counter in D1 and D2 (C200 ~ C255). However, the 16-bit counter specified in D1 and the 32-bit counter in D2 are not allowed to be specified.

4.5.2 FN 41 - DECO/Decoding

Outline

An instruction to convert any one of the digital data into an ON bit of 1 point. The bit number can be read as a value according to the position of the ON bit.



Decoding FN41 - DECO	Instruction Mark	Execution Condition	Instruction Type	Instruction Steps
	DECO	Continuous type	16 bit	7
	DECOP	Pulse type	16 bit	7

Operand	Setting Data	Data Type
	S: Save the data to be decoded, or the word soft component number of the data	16 bit
	D: Bit/word soft component number for saving the decoded result	16 bit
	n: Number of bits of the soft component that stores the decoded result (n = 1 ~ 8, when n = 0, it is not processed)	16 bit

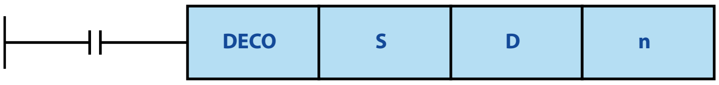
Operand Object Soft Component	Bit Soft Component							Word Soft Component								Others			
	X	Y	M	T	C	S	D.b	KnX	KnY	KnM	KnS	T	C	D	V, Z	H	K	E	P
S	●	●	●			●						●	●	●	●	●	●		
D		●	●			●						●	●	●	●				
n																●	●		

Function and Action Description

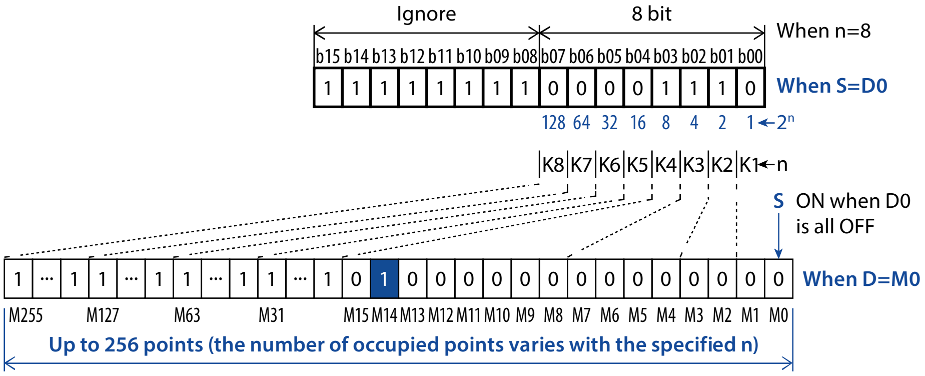
16-bit Operation (DECO, DECOP)

One of D ~ D+2^n-1 corresponding to the value of S is turned ON.

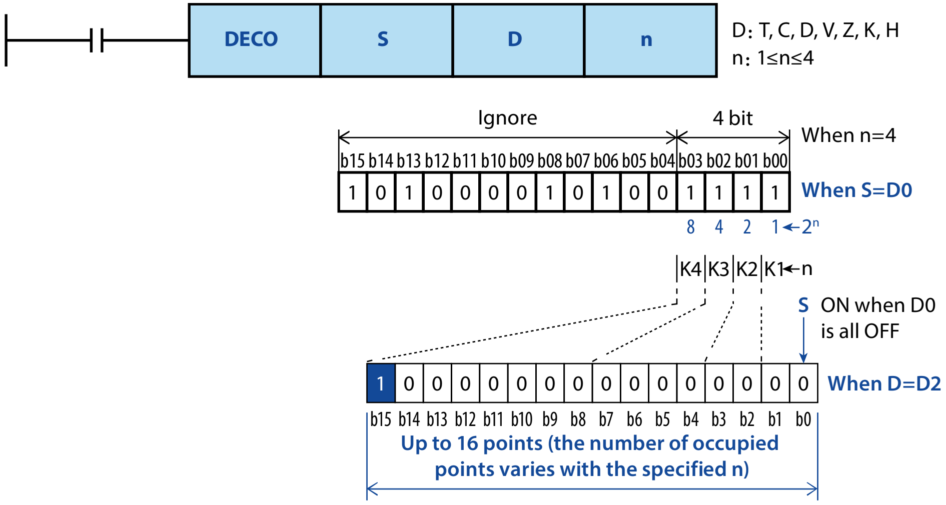
- When D is a bit soft component (1 ≤ n ≤ 8), the n-bit number of the soft component specified in S (1 ≤ n ≤ 8) is decoded in D.
- When S is 0, the soft component D is ON.
- When n = 8, the Max. is 2^8 = 256 points.



D: Y, M, S
n: 1≤n≤8


Ignore
8 bit
When n=8
b15 b14 b13 b12 b11 b10 b09 b08 b07 b06 b05 b04 b03 b02 b01 b00
1 1 1 1 1 1 1 1 0 0 0 0 1 1 1 0
When S=D0
128 64 32 16 8 4 2 1←2ⁿ
K8 K7 K6 K5 K4 K3 K2 K1←n
S ON when D0 is all OFF
1 … 1 1 … 1 1 … 1 1 … 1 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0
When D=M0
M255 M127 M63 M31 M15 M14 M13 M12 M11 M10 M9 M8 M7 M6 M5 M4 M3 M2 M1 M0
Up to 256 points (the number of occupied points varies with the specified n)

16-bit Operation (DECO, DECOP)
• When D is a word soft component (1 ≤ n ≤ 4), the lower n bits of S are decoded in D. • When S is 0, b0 of word soft component D is ON. • When n ≤ 3, the high position of D is 0 (OFF).


DECO
S
D
n
D: T, C, D, V, Z, K, H
n: 1≤n≤4
Ignore
4 bit
When n=4
b15 b14 b13 b12 b11 b10 b09 b08 b07 b06 b05 b04 b03 b02 b01 b00
1 0 1 0 0 0 0 1 0 1 0 0 1 1 1 1
When S=D0
8 4 2 1←2ⁿ
K4 K3 K2 K1←n
S ON when D0 is all OFF
1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
When D=D2
b15 b14 b13 b12 b11 b10 b9 b8 b7 b6 b5 b4 b3 b2 b1 b0
Up to 16 points (the number of occupied points varies with the specified n)

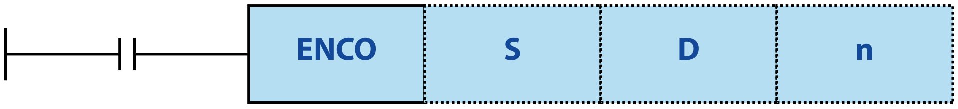
Note

Note	
1	When the command input is OFF, the command is not executed, but the decoded output that is already running will remain in the previous ON/OFF state.
2	The instruction when n = 0 is not processed.

4.5.3 FN 42 - ENCO/Encoding

Outline

Find the instruction of the position of the ON bit in the data.



Encoding FN42 - ENCO	Instruction Mark	Execution Condition	Instruction Type	Instruction Steps
	ENCO	Continuous type	16 bit	7
	ENCOP	Pulse type	16 bit	7

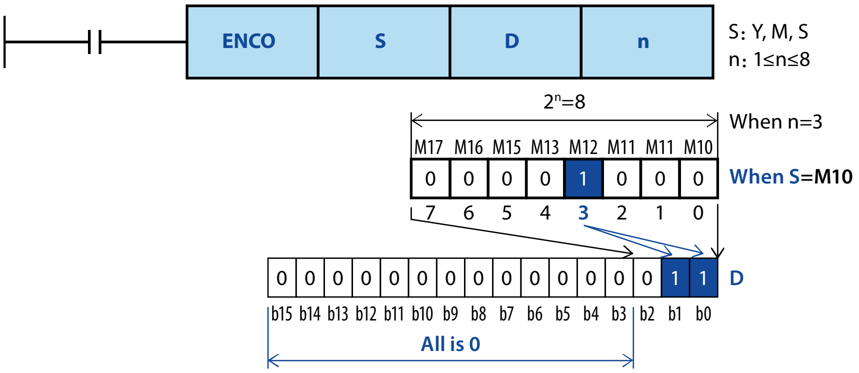
Operand	Setting Data																		
	S: Save the data to be decoded, or the word soft component number of the data														16 bit				
	D: Save the word soft component number of the encoded result														16 bit				
	n: Number of bits of the soft component that stores the decoded result (n = 1 ~ 8, when n = 0, it is not processed)														16 bit				
	Operand Object Soft Component																		
	Bit Soft Component							Word Soft Component								Others			
	X	Y	M	T	C	S	D.b	KnX	KnY	KnM	KnS	T	C	D	V, Z	H	K	E	P
S	●	●	●			●						●	●	●	●				
D												●	●	●	●				
n																●	●		

Function and Action Description

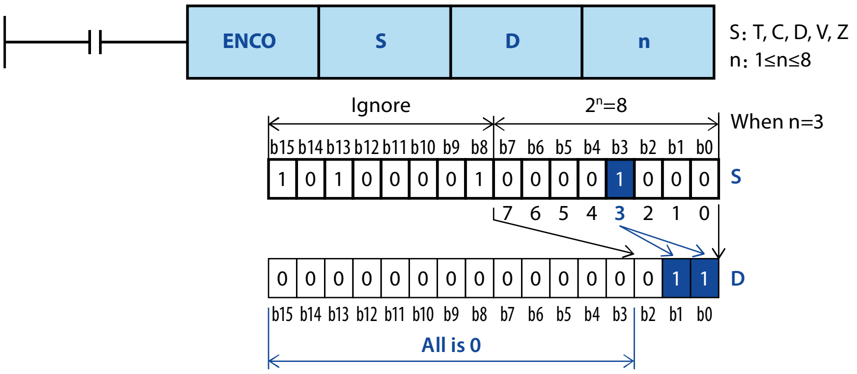
16-bit Operation (ENCO, ENCOP)

Save the 2n-bit encoded value of S in D. The encoding is to convert the position of the ON bit into BIN data.

- When S is a bit soft component (1 ≤ n ≤ 8), find the bit that is ON in the 2n (1 ≤ n ≤ 8) bits at the beginning of S, and encode the ON bit position and write it to D.
    - Note: 2^n is the nth power of 2.
- S starts the 2^n (1 ≤ n ≤ 8) ON bit position and encodes in D.
    - When n = 8, S is the Max. 2^8 = 256 points.
    - The encoding result of D is from high to low n bits, all of which are 0 (OFF).



- When S is a word soft component (1 ≤ n ≤ 4), find the number of bits in the lower 2n bits of S that is ON, and encode the highest bit number into D.
    - The remaining other bits of the D code are all 0 (OFF).



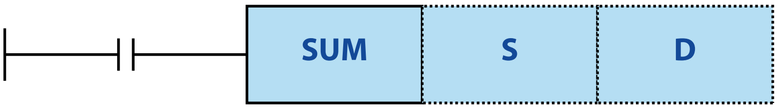
Note

Note	
1	When multiple bits of the data in S are ON, the low side is ignored, and the ON position of the high side is encoded.
2	When the command input is OFF, the command is not executed, but the coded output that is already running will remain in the previous ON/OFF state.

4.5.4 FN 43 - SUM/Number of ON Bit

Outline

Calculates how many 1 (ON) instructions are in the data of the specified soft component.



Number of ON Bit FN43 - SUM	Instruction Mark	Execution Condition	Instruction Type	Instruction Steps
	SUM	Continuous type	16 bit	5
	SUMP	Pulse type	16 bit	5
	DSUM	Continuous type	32 bit	9
	DSUMP	Pulse type	32 bit	9

<table>
<tr><td rowspan="6">Operand</td><td colspan="15">Setting Data</td><td colspan="4">Data Type</td></tr>
<tr><td colspan="15">S: Save the word soft component number of the source data</td><td colspan="4">16/32 bit</td></tr>
<tr><td colspan="15">D: Word soft component number in which the result data is saved</td><td colspan="4">16/32 bit</td></tr>
<tr><td colspan="19">Operand Object Soft Component</td></tr>
<tr><td colspan="7">Bit Soft Component</td><td colspan="8">Word Soft Component</td><td colspan="4">Others</td></tr>
<tr><td>X</td><td>Y</td><td>M</td><td>T</td><td>C</td><td>S</td><td>D.b</td><td>KnX</td><td>KnY</td><td>KnM</td><td>KnS</td><td>T</td><td>C</td><td>D</td><td>V, Z</td><td>H</td><td>K</td><td>E</td><td>P</td></tr>
<tr><td>S</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td>●</td><td>●</td><td>●</td><td>●</td><td>●</td><td>●</td><td>●</td><td>●</td><td>●</td><td>●</td><td></td><td></td></tr>
<tr><td>D</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td>●</td><td>●</td><td>●</td><td>●</td><td>●</td><td>●</td><td>●</td><td></td><td></td><td></td><td></td></tr>
</table>

Function and Action Description

16-bit Operation (SUM, SUMP)	32-bit Operation (DSUM, DSUMP)
The bits that are ON in S are counted and saved to D. • When S is 0 (OFF), the zero mark M8020 is ON.	The number of bits that are ON in [S+1,S] is counted and saved to D. • The number of points in the D that hold the ON bit, and the value of K in the D+1. • When [S+1,S] is 0 (OFF), the zero mark M8020 is ON.

According to the value of S, the operation result of D is shown in the following table (In the case of 16-bit operation).

S																		D	M8020 Zero Mark
Bit Soft Component																Word Soft Component			
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0	Decimal Number	Hexadecimal Number		
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0000	0	ON
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0001	1	OFF
0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	2	0002	1	OFF
0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	3	0003	2	OFF
…																…	…	…	OFF
0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	8	0008	1	OFF
0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	9	0009	2	OFF
0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	10	000A	2	OFF
0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	1	11	000B	3	OFF
…																…	…	…	OFF
1	1	1	1	1	1	1	1	1	1	1	1	1	0	1	1	-5	FFFB	15	OFF
1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	-4	FFFC	14	OFF
1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	1	-3	FFFD	15	OFF
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	-2	FFFE	15	OFF
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	-1	FFFF	16	OFF

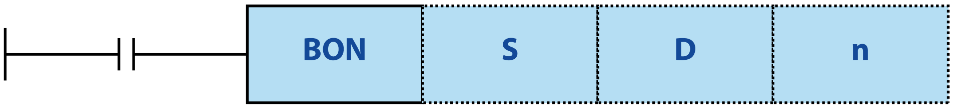
Note

Note	
1	When the command input is OFF, the command is not executed, but the output of the ON bit of the action will remain in the previous ON/OFF state.

4.5.5 FN 44 - BON/ON Bit Judgment

Outline

Check if the position of the specified bit in the soft component is ON or OFF.



	Instruction Mark	Execution Condition	Instruction Type	Instruction Steps
Judgment of the ON Bit FN44 - BON	BON	Continuous type	16 bit	7
	BONP	Pulse type	16 bit	7
	DBON	Continuous type	32 bit	13
	DBONP	Pulse type	32 bit	13

Operand	Setting Data	Data Type
	S: Word soft component number to save data	16/32 bit
	D: Drive bit soft component number	16/32 bit
	n: Bit position to be judged [n: 0 ~ 15 (16 bit command), n: 0 ~ 31 (32-bit command)]	16/32 bit

Operand	Operand Object Soft Component																			
	Bit Soft Component							Word Soft Component								Others				
	X	Y	M	T	C	S	D.b	KnX	KnY	KnM	KnS	T	C	D	V, Z	H	K	E	P
S								●	●	●	●	●	●	●	●	●	●		
D		●	●			●	●								●				
n														●		●	●		

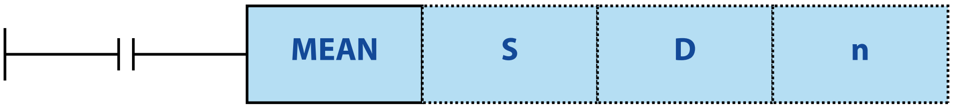
Function and Action Description

16-bit Operation (BON, BONP)	32-bit Operation (DBON, DBONP)
The status of the n-bit of S (ON or OFF) is output to D [ON→D = ON, OFF→D = OFF]. When a constant (K) is specified in the transfer source S, the BIN conversion is automatically performed.	Output the status of n bits (ON or OFF) in [S+1,S] to D [ON→D = ON, OFF→D = OFF]. When a constant (K) is specified in the transfer source [S+1,S], the BIN conversion is automatically performed.

4.5.6 FN 45 - MEAN/Average Value

Outline

An instruction finds the average of the data.



Judgment of the ON Bit FN45 - MEAN	Instruction Mark	Execution Condition	Instruction Type	Instruction Steps
	MEAN	Continuous type	16 bit	7
	MEANP	Pulse type	16 bit	7
	DMEAN	Continuous type	32 bit	13
	DMEANP	Pulse type	32 bit	13

Operand	Setting Data	Data Type
	S: Save the start word soft component number of the desired average data	16/32 bit
	D: Word soft component number for saving the obtained average data	16/32 bit
	n: Average number of data (n = 1 ~ 64)	16/32 bit

Operand	Operand Object Soft Component																		
	Bit Soft Component							Word Soft Component								Others			
	X	Y	M	T	C	S	D.b	KnX	KnY	KnM	KnS	T	C	D	V, Z	H	K	E	P
S								●	●	●	●	●	●	●	●				
D									●	●	●	●	●	●	●				
n														●		●	●		

Function and Action Description

16-bit Operation (MEAN, MEANP)	32-bit Operation (DMENA, DMEANP)
Save the average of the n 16-bit data starting from S to D. The remainder is rounded off.	Save the average of the n 32-bit data starting from [S+1,S] to [D+1,D]. The remainder is rounded off.

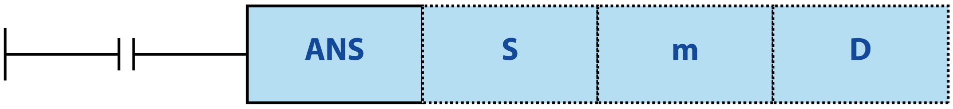
Error

Error	
1	When n is other than 1 to 64, an operation error (M8067) will occur.

4.5.7 FN 46 - ANS/Signal Alarm Set

Outline

Command for setting the signal alarm soft component (S900 ~ S999).



Signal Alarm Set FN46 - ANS	Instruction Mark	Execution Condition	Instruction Type	Instruction Steps
	ANS	Continuous type	16 bit	7

Operand	Setting Data	Data Type
	S: Timing timer number for judging time	16 bit
	m: Data for judging time [m = 1 ~ 32,767 (100ms units)]	16 bit
	D: Set signal alarm soft component	16 bit

Operand	Operand Object Soft Component																		
	Bit Soft Component							Word Soft Component								Others			
	X	Y	M	T	C	S	D.b	KnX	KnY	KnM	KnS	T	C	D	V, Z	H	K	E	P
S												●			●				
m														●		●	●		
D						●									●				

Function and Action Description

16-bit Operation (ANS)
The command input continues to be ON for longer than the judgment time [m × 100ms], set D to 1. When the command input is OFF after the dissatisfaction condition [m × 100ms], the current value of the timer S is reset, and D is not set. In addition, after the instruction input is turned OFF, the reset timer S is reset.

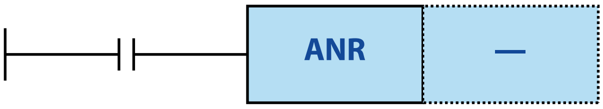
Related Soft Component

Soft Component	Name	Content
M8049	Signal alarm is valid	After M8049 is turned ON, the following M8048 and D8049 work.
M8048	Signal alarm action	M8049 is ON, and when any of the states S900 ~ S999 is activated, M8048 turns ON.
D8049	ON state Min. number	Save the Min. number of actions in S900 ~ S999.

4.5.8 FN 47 - ANR/Signal Alarm Reset

Outline

Reset the soft component with the lowest number that has been turned ON in the signal alarm (S900 ~ S999).



Signal Alarm Reset FN47 - ANR	Instruction Mark	Execution Condition	Instruction Type	Instruction Steps
	ANR	Continuous type	16 bit	1
	ANRP	Pulse type	16 bit	1

<table>
<tr><td rowspan="5">Operand</td><td colspan="14">Setting Data</td><td colspan="5">Data Type</td></tr>
<tr><td colspan="14">No setting data</td><td colspan="5">Independent instruction</td></tr>
<tr><td colspan="19">Operand Object Soft Component</td></tr>
<tr><td colspan="7">Bit Soft Component</td><td colspan="8">Word Soft Component</td><td colspan="4">Others</td></tr>
<tr><td>X</td><td>Y</td><td>M</td><td>T</td><td>C</td><td>S</td><td>D.b</td><td>KnX</td><td>KnY</td><td>KnM</td><td>KnS</td><td>T</td><td>C</td><td>D</td><td>V, Z</td><td>H</td><td>K</td><td>E</td><td>P</td></tr>
<tr><td>—</td><td colspan="19">No object soft component</td></tr>
</table>

Function and Action Description

16-bit Operation (ANR, ANRP)
After the command input is ON, the status of the signal alarm S900 ~ S999 is reset. If there are multiple state actions, reset the state with the lowest number.

Related Soft Component

Devive	Name	Content
M8049	Signal alarm is valid	After M8049 is turned ON, the following M8048 and D8049 work.
M8048	Signal alarm action	M8049 is ON, and when any of the states S900 ~ S999 is activated, M8048 turns ON.
D8049	ON state Min. number	Save the Min. number of actions in S900 ~ S999.

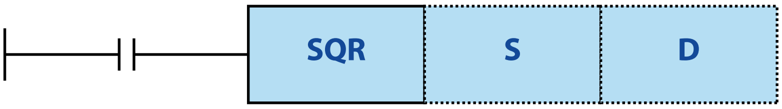
Note

Note		Description
1	Execution of each scan cycle	When using the ANR instruction, each scan cycle is reset in turn. When using the ANRP instruction, only one scan cycle (1 time) is executed.

4.5.9 FN 48 - SQR/BIN Square

Outline

Find the square root (open square root) instruction.



BIN Square FN48 - SQR	Instruction Mark	Execution Condition	Instruction Type	Instruction Steps
	SQR	Continuous type	16 bit	5
	SQRP	Pulse type	16 bit	5
	DSQR	Continuous type	32 bit	9
	DSQRP	Pulse type	32 bit	9

Operand	Setting Data														Data Type				
	S: Save the word soft component number to be square rooted data														16/32 bit				
	D: Save the data register number of the square root operation result														16/32 bit				
	Operand Object Soft Component																		
	Bit Soft Component							Word Soft Component								Others			
	X	Y	M	T	C	S	D.b	KnX	KnY	KnM	KnS	T	C	D	V, Z	H	K	E	P
S														●	●	●	●		
D														●	●				

Function and Action Description

16-bit Operation (SQR, SQRP)	16-bit Operation (DSQR, DSQRP)
After calculating the square root of the data of S, save it to D.	After calculating the square root of the data of [S+1,S], save it to [D+1,D].

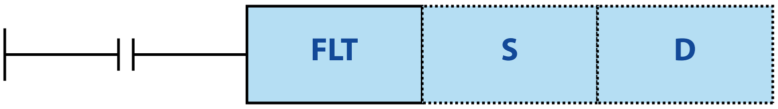
Note

Note		Description
1	About the result of the operation	Round off the decimal point to take an integer. When there is a non-zero fraction, the M8021 (borrow flag) is turned ON. When the operation result has no decimal, M8020 (zero mark) turns ON.

4.5.10 FN 49 - FLT/BIN Integer→Binary Floating Point Number Conversion Outline

Outline

An instruction that converts a BIN integer value into a binary floating point number (real number).



BIN Square FN49 - FLT	Instruction Mark	Execution Condition	Instruction Type	Instruction Steps
	FLT	Continuous type	16 bit	5
	FLTP	Pulse type	16 bit	5
	DFLT	Continuous type	32 bit	9
	DFLTP	Pulse type	32 bit	9

Operand	Setting Data														Data Type					
	S: Data register number holding the BIN integer value														16/32 bit					
	D: Save the data register number of the binary floating point number (real number)														16/32 bit					
	Operand Object Soft Component																			
	Bit Soft Component							Word Soft Component									Others			
	X	Y	M	T	C	S	D.b	KnX	KnY	KnM	KnS	T	C	D	V, Z	H	K	E	P
S														●	●				
D														●	●				

Function and Action Description

16-bit Operation (FLT, FLTP)	32-bit Operation (DFLT, DFLTP)
Converts the BIN integer value data of S into a binary floating point (real number) value and stores it in [D+1,D].	Convert the BIN integer value data of [S+1,S] to a binary floating point (real number) value and store it in [D+1,D].

Note

Note		Description
1	No need for constant (K, H) floating point conversion	Since the value of K and H specified in the binary floating point (real) operation instruction is automatically converted to binary floating point number (real number), there is no need to use the FLT instruction for conversion.

4.6 High Speed Processing - FN 50 ~ FN 59

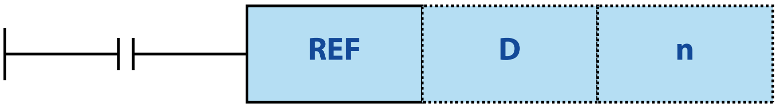
In FN50 ~ FN59, instructions for sequence control with the latest input and output information and high-speed processing instructions for the intelligent controller are provided.

FN No.	Instruction Mark	Instruction Format	Function	Chapter	Page
50	REF	REF (D) (n) REFP (D) (n)	Input and output refresh	4.6.1	100
52	MTR	MTR (S) (D1) (D2) (n)	Matrix input	4.6.2	101
53	HSCS	DHSCS (S1) (S2) (D)	Compare set (for high speed counter)	4.6.3	102
54	HSCR	DHSCR (S1) (S2) (D)	Compare reset (for high speed counter)	4.6.4	103
55	HSZ	DHSZ (S1) (S2) (S) (D)	Interval comparison (for high-speed counters)	4.6.5	104
56	SPD	SPD (S1) (S2) (D) DSPD (S1) (S2) (D)	Pulse density (for high-speed counters)	4.6.6	105

4.6.1 FN 50 - REF/Input and Output Refresh

Outline

In the sequence program scanning process, when you want to get the latest input (X) information, and output the (Y) scan result immediately.



Input and Output Refresh FN50 - REF	Instruction Mark	Execution Condition	Instruction Type	Instruction Step
	REF	Continuous type	16 bit	5
	REFP	Pulse type	16 bit	5

<table>
<tr><td rowspan="6">Operand</td><td colspan="14">Setting Data</td><td colspan="5">Data Type</td></tr>
<tr><td colspan="14">D: Refreshed bit soft component (X, Y) number</td><td colspan="5">Bit</td></tr>
<tr><td colspan="14">n: Refreshed bit soft component points</td><td colspan="5">16 bit</td></tr>
<tr><td colspan="19">Operand Object Soft Component</td></tr>
<tr><td colspan="7">Bit Soft Component</td><td colspan="8">Word Soft Component</td><td colspan="4">Others</td></tr>
<tr><td>X</td><td>Y</td><td>M</td><td>T</td><td>C</td><td>S</td><td>D.b</td><td>KnX</td><td>KnY</td><td>KnM</td><td>KnS</td><td>T</td><td>C</td><td>D</td><td>V, Z</td><td>H</td><td>K</td><td>E</td><td>P</td></tr>
<tr><td>D</td><td>●</td><td>●</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>n</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td>●</td><td>●</td><td></td><td></td></tr>
</table>

Function and Action Description

16-bit Operation (REF, REFP)
When the output (Y) is refreshed, the n point at which the output (D) starts is refreshed. When this instruction is executed, the output status in the specified range is refreshed to the output latch memory area. When the input (X) is refreshed, the n point at which the input (D) starts is refreshed. When the instruction is executed, the filtered input state in the specified range is refreshed to the output latch memory area. This instruction does not change the filter time.

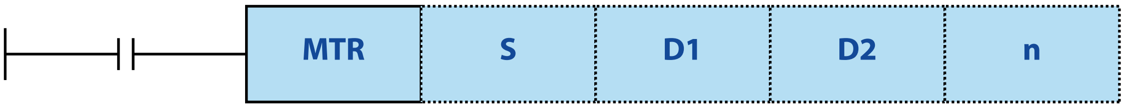
Note

Note	
1	Only the input of X0 ~ X7 and output of Y0 ~ Y7 of the main module can be refreshed, when using other addresses, it will report an overrun error and not execute.

4.6.2 FN 52 - MTR/Matrix Input

Outline

The 8-point input and the n-point output (transistor) are time-divided to read the 8-point n-column input signal (switch) command.



Matrix Input	Instruction Mark	Execution Condition	Instruction Type	Instruction Step
FN52 - MTR	MTR	Continuous type	16 bit	9

Operand	Setting Data	Data Type
	S: Start soft component (X) number of the row signal input of the matrix	Bit
	D1: Start soft component of the column signal output of the matrix (Y) No	Bit
	D2: Start soft component of the ON output destination address (Y, M, S)	Bit
	n: Set the number of columns input by the matrix (K2 ~ K8/H2 ~ H8)	16 bit

Operand	Operand Object Soft Component																		
	Bit Soft Component							Word Soft Component								Others			
	X	Y	M	T	C	S	D.b	KnX	KnY	KnM	KnS	T	C	D	V, Z	H	K	E	P
S	●																		
D1		●																	
D2		●	●			●													
n																●	●		

Function and Action Description

16-bit Operation (MTR)
Time division control is performed on the 8-point S input and the n-point D-transistor output, so that the 8-point n-column input signals are sequentially read and then output to D.

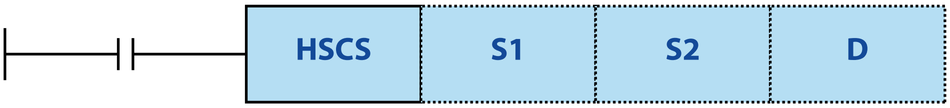
Note

Note		Description
1	Number of occupied soft components	Start with the input specified in S, occupying 8 points of input. Start with the output specified in D1, occupying the output of n points. When specifying the output in D2, be careful not to repeat the output number (occupied with n points) specified in D1.
2	Scan cycle	MTR instructions are updated by switching a set of inputs during the execution of each cycle, so make sure that each execution interval exceeds the set terminal filtering time, otherwise the input state cannot be correctly refreshed. It is recommended to use in constant scanning mode, and ensure that the constant scanning period is longer than the terminal filtering time.

4.6.3 FN 53 - HSCS/Comparing Position (for High-speed Counter)

Outline

An instruction sets the soft component D immediately when the high-speed counter is in accordance with the specified value.



Comparing Position FN53 - HSCS	Instruction Mark	Execution Condition	Instruction Type	Instruction Step
	DHSCS	Continuous type	32 bit	13

Operand	Setting Data	Data Type
	S1: Data compared with the current value of a high-speed counter, or the number of word-soft components that hold the comparative data	32 bit
	S2: Software component number of high speed counter (C235 ~ C255)	32 bit
	D: Bit software component number for on	Bit

Operand	Operand Object Soft Component																		
	Bit Soft Component							Word Soft Component								Others			
	X	Y	M	T	C	S	D.b	KnX	KnY	KnM	KnS	T	C	D	V, Z	H	K	E	I
S1								●	●	●	●	●	●	●	●	●	●		
S2													●		●				
D		●	●			●	●								●				●

Function and Action Description

32-bit Operation (DHSCS)
When the current value of the high-speed counter (C235 ~ C255) specified in S2 is changed to the comparative value [S1+1,S1] (the comparative value K200 is 199→200 or 201→200), the bit soft element D is positioned (ON).

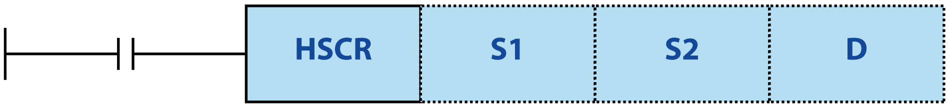
Note

Note		Description
1	Selection of counting and comparing methods	It is not affected by scan time of intelligent controller.
2	Software components that can be specified in S	Only high-speed counters (C235 ~ C255) are valid.
3	When HSCS (FN 53), HSCR (FN 54) responds at the same time, the execution is executed first in the program, and precedes the HSZ (FN 55) instruction.	
4	The high-speed counter interrupt response time interval should be at least 100us, otherwise there may be cases where the interrupt is too late to respond.	
5	Each group of high-speed counters can use up to eight high-speed comparison instructions (HSCS and HSCR) and up to four high-speed interval comparison instructions (HSZ).	

4.6.4 FN 54 - HSCR/Compare Reset (for High Speed Counter)

Outline

When the high-speed counter matches the specified value, the instruction of soft component D is immediately reset.



Compare Reset FN54 - HSCR	Instruction Mark	Execution Condition	Instruction Type	Instruction Step
	DHSCR	Continuous type	32 bit	13

Operand	Setting Data	Data Type
	S1: Data compared with the current value of the high-speed counter, or the word soft component number holding the comparison data	32 bit
	S2: Soft component number of the high-speed counter (C235 ~ C255)	32 bit
	D: Bit soft component number that is reset (OFF) after the match	Bit

Operand	Operand Object Soft Component																	
	Bit Soft Component							Word Soft Component								Others		
	X	Y	M	T	C	S	D.b	KnX	KnY	KnM	KnS	T	C	D	V, Z	H	K	E	P
S1								●	●	●	●	●	●	●	●	●	●		
S2													●		●				
D		●	●			●	●						●		●				

Function and Action Description

32-bit Operation (DHSCR)
When the current value of the high-speed counter (C235 ~ C255) specified in S2 becomes the comparison value [S1+1,S1] (199→200 or 201→200 when the comparison value is K200), the bit soft component D is reset (OFF).

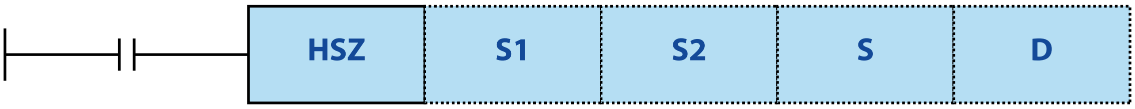
Note

Note	
1	See the HSCS directive note, section 4.6.3.

4.6.5 FN 55 - HSZ/Interval Comparison (for High-speed Counters)

Outline

The current value of the high-speed counter is compared with two values (intervals), and the result is output (refreshed) into the bit soft component (3 points).



Interval Comparison FN55 - HSZ	Instruction Mark	Execution Condition	Instruction Type	Instruction Step
	DHSZ	Continuous type	32 bit	17

Operand	Setting Data	Data Type
	S1: Data compared with the current value of the high-speed counter, or the word soft component number holding the comparison data (comparison value 1)	32 bit
	S2: Data compared with the current value of the high-speed counter, or the word soft component number holding the comparison data (comparison value 2)	32 bit
	S: Soft component number of the high-speed counter (C235 ~ C255)	32 bit
	D: Start bit soft component number of the result of comparison with the comparison upper limit value and the comparison lower limit value	Bit

Operand Object Soft Component	Bit Soft Component							Word Soft Component								Others			
	X	Y	M	T	C	S	D.b	KnX	KnY	KnM	KnS	T	C	D	V, Z	H	K	E	P
S1								●	●	●	●	●	●	●	●	●	●		
S2								●	●	●	●	●	●	●	●	●	●		
S													●		●				
D		●	●			●	●								●				

Function and Action Description

32-bit Operation (DHSZ)

This is an instruction to perform comparison processing after the counting processing of the high-speed counter.

When the current value of the high-speed counter (C235 ~ C255) specified in S2 is compared with two comparison points (comparison value 1, comparison value 2), the results of less than, within, and greater than according to the comparison will be [D,D+1,D+2] turns on the corresponding position.

- Comparison value 1 and comparison 2 must be satisfied: [S1+1,S1] ≤ [S2+1,S2].
- Action: When the current value of the high-speed counter S changes as follows (count), D, D+1, and D+2 output the comparison result.

Comparison Mode	Current Value of S2	Output Contact (D) Change		
		D	D+1	D+2
S1 > S	S1 > S	ON	OFF	OFF
S1 ≤ S ≤ S2	S1 ≤ S ≤ S2	OFF	ON	OFF
S < S2	S > 2000	OFF	OFF	ON

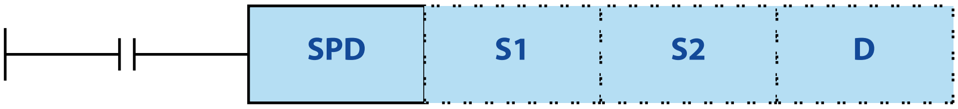
Note

Note	
1	See the HSCS directive note, section 4.6.3.

4.6.6 FN 56 - SPD/Pulse Density (for High-speed Counters)

Outline

It calculates the command of pulse frequency according to the set sampling time and filter coefficient, .



Pulse Density FN56 - SPD	Instruction Mark	Execution Condition	Instruction Type	Instruction Step
	SPD	Continuous type	16 bit	7
	DSPD	Continuous type	32 bit	13

	Setting Data	Data Type
Operand	S1: Input soft component number of (X) pulse	Bit
	S2: Start address of the word device of the collected parameter	16/32 bit
	D: Start word device number for saving pulse frequency data	16/32 bit

Operand	Operand Object Soft Component																		
	Bit Soft Component							Bit Soft Component								Bit Soft Component			
	X	Y	M	T	C	S	D.b	KnX	KnY	KnM	KnS	T	C	D	V, Z	H	K	E	P
S1	●														●				
S2								●	●	●	●	●	●	●	●				
D												●	●	●	●				

Function and Action Description

16-bit Operation (PLSY)	32-bit Operation (DPLSY)
According to the sampling period, calculate the current counting frequency of the S1 terminal corresponding to the high-speed counter, filter according to the filter coefficient, and save it to D after completion. • Sampling period: Set by S2, range: 1 ~ 3000ms, unit: 1ms. • Filter coefficient: Set by [S2+1], range: 1 ~ 100%. • The high-speed counter must be enabled first. Filtering adopts first-order RC filter, the formula is: Y (n) = αX (n) + (1 - α) Y (n-1) • α = filter coefficient; • X (n) = sample value of this time; • Y (n-1) = sample value of that time; • Y (n) = filter output value of this time.	According to the sampling period, calculate the current counting frequency of the S1 terminal corresponding to the high-speed counter, filter according to the filter coefficient, and save it to D after completion. • Sampling period: Set by [S2+1,S2], range: 1 ~ 3000ms, unit: 1ms. • Filter coefficient: Set by [S2+3,S2+2], range: 1 ~ 100%. • The high-speed counter must be enabled first. Filtering adopts first-order RC filter, the formula is: Y (n) = αX (n) + (1 - α) Y (n-1) • α = filter coefficient; • X (n) = sample value of this time; • Y (n-1) = sample value of that time; • Y (n) = filter output value of this time.

Note

Note	
1	Only input terminals that support high-speed counters can use this command, and before use the high-speed counter of the corresponding terminal must be turned on.
2	It can measure single-phase double-counting and double-phase double-counting. When using, first turn on the high-speed counter. S1 of the SPD instruction sets the X terminal with the smaller number in the two-way counter.
3	The same terminal cannot use SPD instructions repeatedly.
4	After SPD runs to modify the sampling parameters, the frequency in D is cleared and the calculation is restarted. • Do not modify frequently during use.
5	Please select appropriate sampling parameters. Too short sampling time will result in inaccurate measurement frequency, and too long response will slow down. The larger the filter coefficient, the faster the response. The smaller the frequency, the smoother the frequency change.
6	HC10-M0808R-C3 does not support this instruction.

4.7 Convenient Instructions - FN 60 ~ FN 69

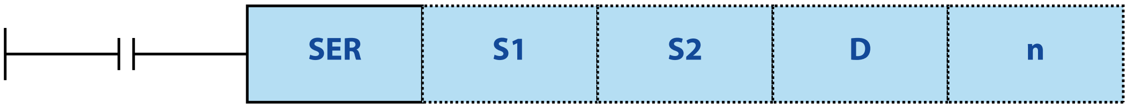
In FN 60 ~ FN 69, a convenient instruction is provided that can implement complex control with a Min. of sequence programs.

FN No.	Instruction Mark	Instruction Format	Function	Chapter	Page
61	SER	SER (S1) (S2) (D) (n) SERP (S1) (S2) (D) (n) DSER (S1) (S2) (D) (n) DSERP (S1) (S2) (D) (n)	Data retrieval	4.7.1	107
62	ABSD	ABSD (S1) (S2) (D) (n) DABSD (S1) (S2) (D) (n)	Cam control absolute mode	4.7.2	109
63	INCD	INCD (S1) (S2) (D) (n)	Cam control relative mode	4.7.3	111
64	TTMR	TIMR (D) (n)	Teaching timer	4.7.4	112
65	STMR	STMR (S) (m) (D)	Special timer	4.7.5	113
66	ALT	ALT (D) ALTP (D)	Alternate output	4.7.6	115
67	RAMP	RAMP (S1) (S2) (D) (n)	Ramp signal	4.7.7	116
69	SORT	SORT (S) (m1) (m2) (D) (n)	Data sorting	4.7.8	117

4.7.1 FN 61 - SER/Data Retrieval

Outline

Retrieve the same data from the data table and the instructions for the Max. and Min. values.



Data Retrieval FN61 - SER	Instruction Mark	Execution Condition	Instruction Type	Instruction Step
	SER	Continuous type	16 bit	9
	SERP	Pulse type	16 bit	9
	DSER	Continuous type	32 bit	17
	DSERP	Pulse type	32 bit	17

Operand	Setting Data	Data Type
	S1: Retrieve the same data and the starting soft component number of the Max. and Min. values	16/32 bit
	S2: Retrieve the same data and the reference value of the Max. value and the Min. value or its save target soft component number	16/32 bit
	D: After retrieving the same data and the Max. and Min. values, save the starting soft component number of these numbers	16/32 bit
	n: Search for the same data and the Max. and Min. values ([1 ~ 256] for 16-bit instructions and [1 ~ 128] for 32-bit instructions)	Bit

Operand Object Soft Component																			
	Bit Soft Component							Word Soft Component								Others			
	X	Y	M	T	C	S	D.b	KnX	KnY	KnM	KnS	T	C	D	V, Z	H	K	E	P
S1								●	●	●	●	●	●	●	●				
S2								●	●	●	●	●	●	●	●	●	●		
D									●	●	●	●	●	●	●				
n														●		●	●		

Function and Action Description

16-bit Operation (SER, SERP)

Search for n data starting with S1, retrieve the same data as S2, and save the result in D ~ D+4.

- The content and results of the retrieved data:
  - When the same data exists: In the five soft components starting with D, the number of the same data, the first/final position, and the positions of the Max. and Min. values are saved.
  - When the same data does not exist: Among the five soft components starting with D, the number of the same data, the initial/final position is set to 0, and the positions of the Max. and Min. values are saved as actual values.
- Action example:
  - Structure and data examples of the search results table.

Retrieved Soft Component S1	Value of the Retrieved Data S1 (Example)	Compare the Value of Data S2 (Example)	Location of the Data	Search Results		
				Max. D+4	Consistent D	Min. D+3
S1	K100	K100	0		○	
S1+1	K111		1			
S1+2	K100		2		○	
S1+3	K98		3			
S1+4	K123		4			
S1+5	K66		5			○
S1+6	K100		6		○ (finally)	
S1+7	K95		7			
S1+8	K210		8	○		
S1+9	K88		9			

16-bit Operation (SER, SERP)

- Search results form, see below.

Soft Component Number	Content	Search Result Item
D	3	The number of identical data
D+1	0	The location of the same data (first time)
D+2	6	The location of the same data (final)
D+3	5	Final position of the Min.
D+4	8	Final position of the Max.

32-bit Operation (DSER, DSERP)

Search for n data starting with [S1+1,S1], retrieve the same data as [S2+1,S2], and save the result in [D+1,D] ~ [D+9, D+8].

- The content and results of the retrieved data.
  - When the same data exists: In the five 32-bit data starting with [D+1,D], the number of the same data, the initial/final position, and the positions of the Max. and Min. values are saved.
  - When the same data does not exist: In the five 32-bit data starting with [D+1,D], the number of the same data, the initial/final position, and the positions of the Max. and Min. values are saved. In the three 32-bit data starting from [D+1,D] (the number of the same data, the initial/final position), 0 is saved.
- Action example:
  - Structure and data examples of the search results table.

Retrieved Soft Component S1	Value of the Retrieved Data S1 (Example)	Compared Value of Data S2 (Example)	Location of the Data	Search Results		
				Max. D+4	Consistent D	Min. D+3
[S1+1,S1]	K100000	K100000	0		○ (initially)	
[S1+3, S1+2]	K110100		1			
[S1+5, S1+4]	K100000		2		○	
[S1+7, S1+6]	K98000		3			
[S1+9, S1+8]	K123000		4			
[S1+11, S1+10]	K66000		5			○
[S1+13, S1+12]	K100000		6		○ (finally)	
[S1+15, S1+14]	K95000		7			
[S1+17, S1+16]	K910000		8	○		
[S1+19, S1+18]	K910000		9	○		

- Search results form, see below.

Soft Component Number	Content	Search Result Item
[D+1,D]	3	The number of identical data
[D+3,D+2]	0	The location of the same data (first time)
[D+5, D+4]	6	The location of the same data (final)
[D+7, D+6]	5	Final position of the Min.
[D+9, D+8]	9	Final position of the Max.

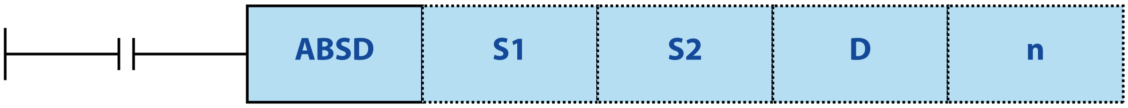
Note

Note		Description
1	Size comparison	Executed algebraically (-10 < 2).
2	When there are multiple Min. and Max. values	When there are multiple Min. and Max. values in the data, the last position is saved.
3	Number of occupied soft components	After driving this instruction, the search result D will occupy the following soft component points. • For 16-bit operation: Occupy [D,D+1,D+2,D+3,D+4] 5 points. • For 32-bit operation: Occupy [[D+1,D], [D+3,D+2], [D+5,D+4], [D+7,D+6], [D+9,D+8]] 10 points.

4.7.2 FN 62 - ABSD/Cam Control Absolute Mode

Outline

An instruction generates multiple output mode corresponding to the current value of the counter.



Cam Control	Instruction Mark	Execution Condition	Instruction Type	Instruction Step
Absolute Mode	ABSD	Continuous type	16 bit	9
FN62 - ABSD	DABSD	Continuous type	32 bit	17

Operand	Setting Data	Data Type
	S1: Start soft component number for saving table data (rising edge, falling edge)	16/32 bit
	S2: Counter value of current value monitoring compared with tabular data	16/32 bit
	D: Output start bit soft component number	Bit
	n: The number of rows in the table and the number of bits of the output bit soft component [1 ≤ n ≤ 64]	16 bit

Operand	Operand Object Soft Component																		
	Bit Soft Component							Word Soft Component								Others			
	X	Y	M	T	C	S	D.b	KnX	KnY	KnM	KnS	T	C	D	V, Z	H	K	E	P
S1								●	●	●	●	●	●	●	●				
S2													●		●				
D		●	●			●	●								●				
n																●	●		

Function and Action Description

16-bit Operation (ABSD)

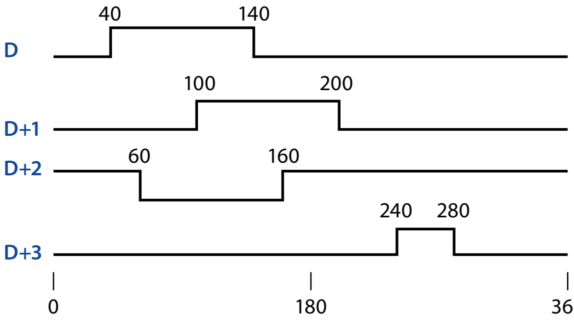
During the rotation of the platform once (0 ~ 360 degrees), the output is turned ON/OFF. This is used as an example for description (1 degree 1 pulse rotation angle signal).

The n-line table data starting from S1 (occupying n rows × 2 points) is compared with the current value S2 of the counter, and during the one rotation, ON/OFF control is performed on the continuous n-point output from D.

- First use the transfer command to write the data shown below in S1 ~ S1+2n+1.

Rising Point		Fall Point		Object Output
	Data Value (Example)		Data Value (Example)	
S1	40	S1	140	D
S1+2	100	S1+3	200	D+1
S1+4	160	S1+5	60	D+2
S1+6	240	S1+7	280	D+3
…	-	…	-	…
S1+2n		S1+2n+1		D+n-1

- Output mode:
  - When the command input is ON, the n point starting with D also changes as follows.
  - Each rising point and falling point can be individually changed by rewriting the data of S1 ~ S1+2n.



32-bit Operation (DABSD)

When the platform is rotated once (0 ~ 360 degrees), the output is turned ON/OFF. This is an example (1 degree 1 pulse rotation angle signal).

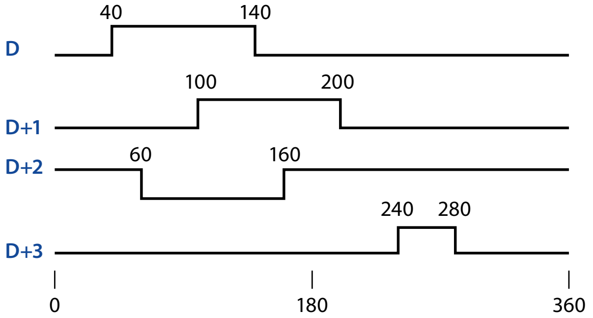
The n-line table data starting from [S1+1,S1] (occupying n rows × 4 points) is compared with the current value S2 of the counter, and during the one rotation, the continuous n-point output from D is turned ON/ OFF control.

- First use the transfer command to write the data shown below in [S1,S1+1] ~ [S1,S1+1] + 4n+3.

Rising Point		Fall Point		Object Output
	Data Value (Example)		Data Value (Example)	
[S1+1,S1]	40	[S1+3, S1+2]	140	D
[S1+5, S1+4]	100	[S1+7, S1+6]	200	D+1
[S1+9, S1+8]	160	[S1+11, S1+10]	60	D+2
[S1+13, S1+12]	240	[S1+15, S1+14]	280	D+3
…	-	…	-	…
[S1+4n+1, S1+4n]		[S1+4n+3, S1+4n+2]		D+n-1

For example, the rising point data is in the even-numbered soft component, and the falling point data is stored in the odd-numbered soft component as 32-bit data.

- Output mode
  - When the command input is ON, the n point starting with D also changes as follows.
  - Each rising point and falling point can be individually changed by rewriting the data of [S1+1,S1] ~ [S1+(n × 2)+3, S1+(n × 2)+2].



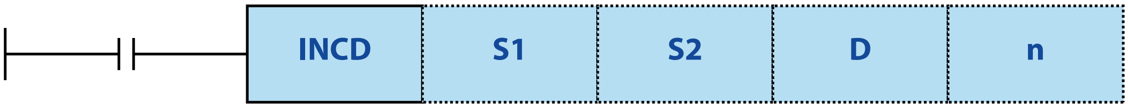
Note

Related Soft Component		Description
1	Designation of high-speed counters (C235 ~ C255)	High-speed counters can also be specified in DABSD instructions. However, at this time, for the current value of the counter, there will be a corresponding delay in the output mode due to the scan period. When responsiveness is required, use the HSZ instruction for high-speed comparison of the table or use the HSCT instruction.
2	When specifying the number of bits of a bit soft component in S1	Soft component number. • Please specify a multiple of 16 (0, 16, 32, 64). Number of digits. • AB4 (16-bit operation) is only K4. • DABSD (for 32-bit operation) is only K8.
3	Other considerations	The value of n determines the number of output points of the object (1 ≤ n ≤ 64). Even if the command input is OFF, the output does not change.

4.7.3 FN 63 - INCD/Cam Control Relative Mode

Outline

Use a pair of counters to generate multiple output mode instructions.



Cam Control Relative Mode FN63 - INCD	Instruction Mark	Execution Condition	Instruction Type	Instruction Step
	INCD	Continuous type	16 bit	9

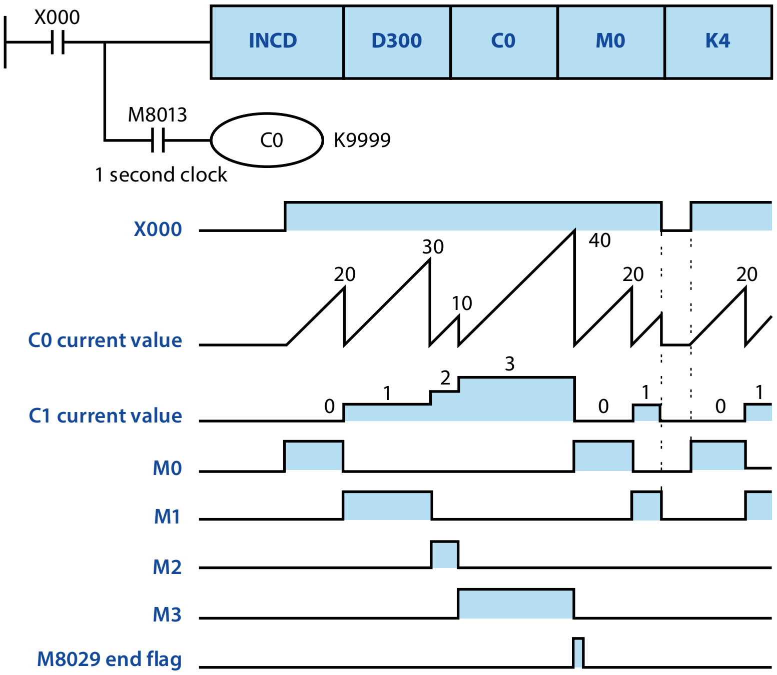
Operand	Setting Data	Data Type
	S1: Save the start word soft component number of the set value	16 bit
	S2: Start number of the counter used to monitor the current value	16 bit
	D: The starting bit soft component number of the output	Bit
	n: Number of points of the output bit soft component [1 ≤ n ≤ 64]	16 bit

Operand Object Soft Component	Bit Soft Component							Word Soft Component								Others			
	X	Y	M	T	C	S	D.b	KnX	KnY	KnM	KnS	T	C	D	V, Z	H	K	E	P
S1								●	●	●	●	●	●	●	●				
S2													●		●				
D		●	●			●	●								●				
n																●	●		

Function and Action Description

16-bit Operation (INCD)

Compare the n-line table data starting from S1 (occupying n lines and 1 point) with the current value S2 of the counter, reset if they match, and turn ON/OFF the output in turn. See the right timing diagram, using the transfer instructions, write the data in the table below.



Save Soft Component		Output	
	Data Value (Example)		Exemple
S1	D300 = 20	D	M0
S1+1	D301 = 30	D+1	M1
S1+2	D302 = 10	D+2	M2
S1+3	D303 = 40	D+3	M3
…	…	…	…
S1+n-1	-	D+n-1	-

- When the command contact is ON, the M0 output is also ON.
- When the current value of C0 reaches the comparison value D300, the output M0 is reset, the count value of the process counter C1 is +1, and the current value of the counter C0 is reset.
- Output M1 is ON.
- The current value of C0 is compared with D301. If the comparison value is reached, input M1 is reset, the count value of process counter C1 is +1, and the current value of counter C0 is reset.
- The same way until you compare n (K4) to the specified number of points (1 ≤ n ≤ 64).
- After the last step specified by n is completed, the execution end flag M8029 is kept ON for one calculation cycle. Since the M8029 instruction for multiple instructions uses the flag for execution completion, it is used directly as a contact after the command. As the end mark dedicated to this instruction.
- Go back to the original, repeat the output.

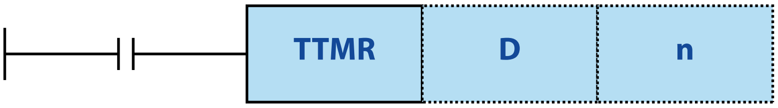
Note

Note	
1	When specifying the number of bit soft components in S1, specify a multiple of 16 (0, 16, 32, 64...) in the soft component number.

4.7.4 FN 64 - TTMR/Teaching Timer

Outline

An instruction for measuring the time when TTMR instruction is ON.



It can be used when buttons are used to adjust the setting time of the timer.

Teaching Timer	Instruction Mark	Execution Condition	Instruction Type	Instruction Step
FN64 - TTMR	TTMR	Continuous type	16 bit	5

Operand	Setting Data	Data Type
	D: Soft component number for saving teaching data	16 bit
	n: The number of times the teaching data is multiplied [K0 ~ K2/H0 ~ H2]	16 bit

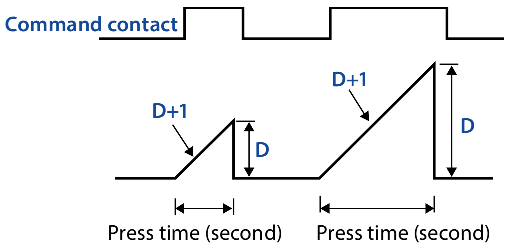
Operand Object Soft Component	Bit Soft Component							Word Soft Component								Others			
	X	Y	M	T	C	S	D.b	KnX	KnY	KnM	KnS	T	C	D	V, Z	H	K	E	P
D														●	●				
n														●		●	●		

Function and Action Description

16-bit Operation (TTMR)

In seconds, measure the press time of the command input (key), then multiply it by the magnification (10n) and transfer it to D.
The time passed to D is when the pressing time is τ0 (1 second unit), the actual value is obtained according to the magnification of n, as shown below.

n	Magnification	D
K0	τ0	D × 1
K1	10τ0	D × 10
K2	100τ0	D × 100



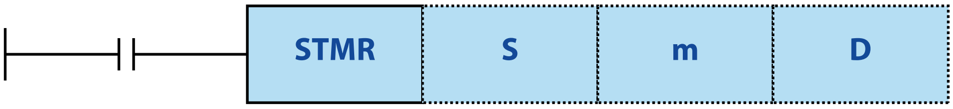
Note

Note		Description
1	When the command contact is OFF	The current value [D+1] of the pressed time is reset, and the teaching time D does not change.
2	Number of occupied soft components	Starting with the soft component specified in the teaching time D, it takes 2 points. Please be careful not to repeat the soft component used in the mechanical control. • D: Teaching time. • D+1: Press the current value of the time.

4.7.5 FN 65 - STMR/Special Timer

Outline

It is used to easily make the instruction of the off-delay timer, single-pulse timer, and flashing timer.



Special Timer FN65 - STMR	Instruction Mark	Execution Condition	Instruction Type	Instruction Step
	STMR	Continuous type	16 bit	7

Operand	Setting Data	Data Type
	S: Timer number used [T0 ~ T199 (100ms timer)]	16 bit
	m: Timer setting value [1 ~ 32,767]	16 bit
	D: Start bit number to be output (occupied 4 points)	Bit

Operand Object Soft Component	Bit Soft Component							Word Soft Component								Others			
	X	Y	M	T	C	S	D.b	KnX	KnY	KnM	KnS	T	C	D	V, Z	H	K	E	P
S												●			●				
m														●		●	●		
D		●	●			●	●								●				

Function and Action Description

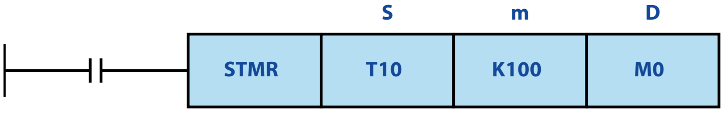
16-bit Operation (STMR)

The value specified by m is set as the timer specified in S, and 4 points are output from D.

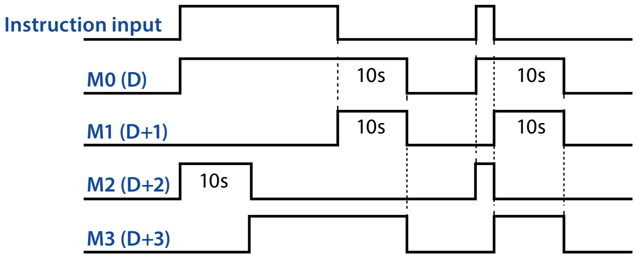
Please refer to the following to change the program according to the purpose of use.

Disconnect delay timer • single pulse timer

- Assign T10 in S, K100 in m, and M0 in D.

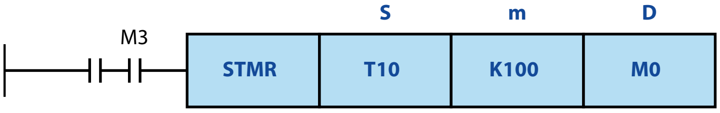


- M0 [D]: After the command contact is turned OFF, it is turned off after the set time of the timer.
- M1 [D+1]: Turns ON when the command contact turns from ON to OFF, and turns OFF after the set time of the timer.
- M2 [D+2]: Occupied, used for flashing.
- M3 [D+3]: Occupied.

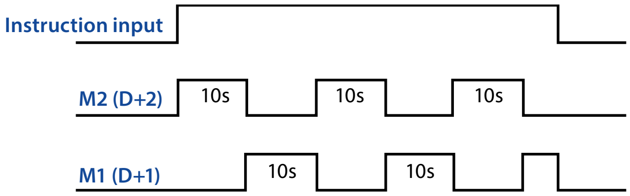


Flashing

- Use the b contact at D+3 to turn off the command. By writing such a program as shown below, the output flashes in D+1, D+2.
- Occupies D, D+3.



- M0 [D]: Occupied (for the off delay timer).
- M1 [D+1]: Repeats the ON/OFF blink (a contact) at the timer interval.
- M2 [D+2]: Repeats the ON/OFF flash (b contact) at the timer interval.
- M3 [D+3]: Occupied.



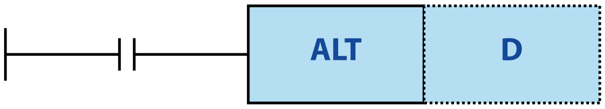
Note

<table>
<tr><th colspan="2">Note</th><th>Description</th></tr>
<tr><td>1</td><td>Specify the use of the timer</td><td>The timer number specified in this instruction cannot be reused in other normal loops (OUT instructions, etc.).
If it is used repeatedly, the timer will not work properly.</td></tr>
<tr><td>2</td><td>Number of occupied soft components</td><td>Takes 4 points starting from the soft component specified in D.
Please be careful not to duplicate the soft components used in other controls of the machine.
<table>
<tr><th rowspan="2">Soft Component</th><th colspan="2">Function</th></tr>
<tr><th>Off Delay Timer/Single Pulse Timer</th><th>Flashing</th></tr>
<tr><td>D</td><td>Disconnect delay timer</td><td>Occupied</td></tr>
<tr><td>D+1</td><td>Single pulse timer</td><td>Flashing (a contact)</td></tr>
<tr><td>D+2</td><td>Occupied</td><td>Flashing (a contact)</td></tr>
<tr><td>D+3</td><td>Occupied</td><td>Flashing (a contact)</td></tr>
</table></td></tr>
<tr><td>3</td><td>When the command contact is OFF</td><td>D, D+1, D+3 turn OFF after the set time has elapsed.
D+2 and timer are reset instantly.</td></tr>
</table>

4.7.6 FN 66 - ALT/Alternate Output

Outline

When the input is ON, the bit soft component is inverted (ON↔OFF).



Alternate Output FN66 - ALT	Instruction Mark	Execution Condition	Instruction Type	Instruction Step
	ALT	Continuous type	16 bit	3
	ALTP	Pulse type	16 bit	3

Operand	Setting Data														Data Type				
	D: Alternate output bit soft component number														Bit				
	Operand Object Soft Component																		
	Bit Soft Component							Word Soft Component								Others			
	X	Y	M	T	C	S	D.b	KnX	KnY	KnM	KnS	T	C	D	V, Z	H	K	E	P
D		●	●			●	●								●				

Function and Action Description

16-bit Operation (ALT, ALTP)
Each time the command input changes from OFF to ON, the bit soft component specified in D performs ON/OFF inversion.

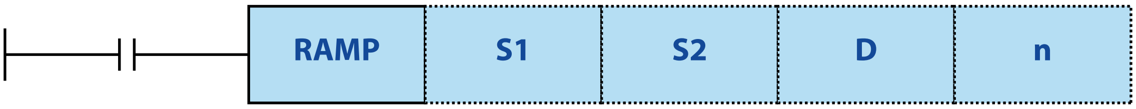
Note

Note	
1	When programming with the ALT instruction, the inversion operation is performed every calculation cycle. When you want to invert the operation by ON/OFF of the instruction, use the ALTP instruction (pulse execution type) or the LDP command contact.

4.7.7 FN 67 - RAMP/Ramp Signal

Outline

Between the two values of the start (initial value) and the end (target value), the instruction to change the data according to the fixed slope (the slope is determined by the scan period n).



Ramp Signal	Instruction Mark	Execution Condition	Instruction Type	Instruction Step
FN67 -RAMP	RAMP	Continuous type	16 bit	9

Operand	Setting Data	Data Type
	S1: Save the soft component number of the set ramp initial value	16 bit
	S2: Save the soft component number of the set ramp target value	16 bit
	D: Soft component number of the current value data of the save ramp	16 bit
	n: Ramp transition time (scan period) [1 ~ 32,767]	16 bit

	Operand Object Soft Component																		
	Bit Soft Component							Word Soft Component								Others			
	X	Y	M	T	C	S	D.b	KnX	KnY	KnM	KnS	T	C	D	V, Z	H	K	E	P
S1														●	●				
S2														●	●				
D															●				
n														●		●	●		

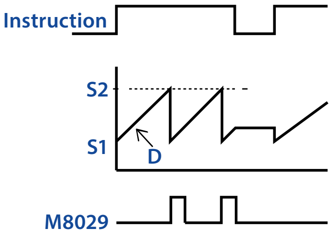
Function and Action Description

16-bit Operation (RAMP)

First, the start value S1 and the end value S2. When the command input is ON, it is equally divided in each operation cycle. S1 accumulates the equal value every operation cycle, and the accumulated result is saved in D.

Combine this instruction with the analog output to output a buffer start/stop command.

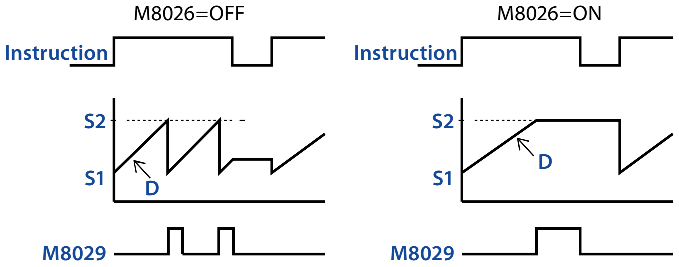
- Save the number of scans in D+1 (0⟶n times).
- The time from start to finish, for the operation cycle × n scans.
  - If the command input is interrupted during the action, the status of the interrupt is executed (D current value data, hold; D+1 scan times are cleared), and after turning ON again, D is cleared and restarts from S1.
  - After the target value is reached, the instruction execution end flag M8029 is activated, and the value of D returns to the value of S1.
- When calculating the operation result at a fixed time interval (constant scan mode):
  - Write the set scan time (a slightly longer value than the actual scan time) to D8039. When M8039 is ON, the intelligent controller is in constant scan mode.
  - For example, if this value specifies 20ms, n = 100 times, it means that the value of D changes from S2 to S2 within 20s.



Mode Flag Bit (M8026) Action

According to the ON/OFF status of the mode flag M8026, the contents of D+1 are also changed as follows.

The intelligent controller is independent of the ON/OFF of the M8026, and is the same as the [M8026 = ON] action shown on the right.



Note

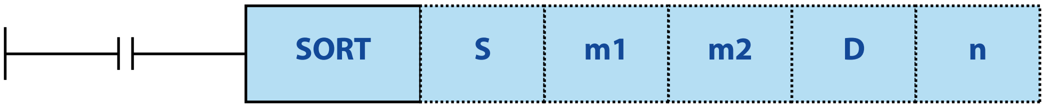
Note	
1	When the power failure holding soft component (holding area) is specified in D, the command input turns ON as it is, and when the intelligent controller is set to RUN, clear D.

4.7.8 FN 69 - SORT/Data Sorting

Outline

This instruction is a data table for data (row) and group data (column). The data table is re-arranged in ascending order according to the specified group data (column). In this instruction, the group data (column) is stored in consecutive soft components.

In addition, the data (row direction) is stored in consecutive soft components. It is also convenient to add data (rows) and support SORT2 (FN 149) instructions in ascending/descending order.



Data Sorting FN69 - SORT	Instruction Mark	Execution Condition	Instruction Type	Instruction Step
	SORT	Pulse type	16 bit	11

Operand	Setting Data	Data Type
	S: Soft component start number of the save data table [occupied m1 × m2 point]	16 bit
	M1: Number of data (rows) [1 ~ 32]	16 bit
	M2: Group data (column) number [1 ~ 6]	16 bit
	D: Soft component start number for saving the operation result [occupied m1 × m2 point]	16 bit
	n: Column number of the group data (column) as the sorting criterion [1 ~ m2]	16 bit

	Operand Object Soft Component																		
	Bit Soft Component							Word Soft Component								Others			
	X	Y	M	T	C	S	D.b	KnX	KnY	KnM	KnS	T	C	D	V, Z	H	K	E	P
S														●					
m1/m2																●	●		
D														●					
n														●		●	●		

Function and Action Description

16-bit Operation (SORT)

In the data table (before sorting) at the (m1 × m2) point at the beginning of S, the data rows in the n-column are used as the standard, the data rows are rearranged in ascending order, and then the data at the (m1 × m2) point starting from D is stored form (after sorting).

- The following example shows the example data before sorting m1 = K3, m2 = K4.
- The structure of the table. If it is a sorted data table, please change S to D.
- Data is arranged when the command input is ON, and sorting is completed in one scan cycle.

Column Number Line Number		The Number of Groups is m2 (m2 = K4)			
		1/Management Number	2/Height	3/Weight	4/Age
The number of data m1 = 3	1	S	S+3	S+6	S+9
	2	S+1	S+4	S+7	S+10
	3	S+2	S+5	S+8	S+11

Action Example

After performing the following pre-sorting data in "n = K2 (column 2)" and "n = K3 (column 3)", it will act as shown on the right.

In addition, if you enter a consecutive number such as a management number in the first column, you can judge the original line number based on its contents, so it is very convenient.

- Pre-sort data, see table below.

Column Number Line Number		The Number of Groups is m2 (m2 = K4)			
		1/Management Number	2/Height	3/Weight	4/Age
The number of data m1 = 5	1	S	S+5	S+10	S+15
		1	150	45	20
	2	S+1	S+6	S+11	S+16
		2	180	50	40
	3	S+2	S+7	S+12	S+17
		3	160	70	30
	4	S+3	S+8	S+13	S+18
		4	100	20	8
	5	S+4	S+9	S+14	S+19
		5	150	50	45

- Sort results when executing instructions with n = K2 (column 2), see the table below.

Column Number Line Number	1/Management Number	2/Height	3/Weight	4/Age
1	D	D+5	D+10	D+15
	4	100	20	8
2	D+1	D+6	D+11	D+16
	1	150	45	20
3	D+2	D+7	D+12	D+17
	5	150	50	45
4	D+3	D+8	D+13	D+18
	3	160	70	30
5	D+4	D+9	D+14	D+19
	2	180	50	40

- Sort results when executing instructions with n = K3 (column 3), see the table below.

Column Number Line Number	1/Management Number	2/Height	3/Weight	4/Age
1	D	D+5	D+10	D+15
	4	100	20	8
2	D+1	D+6	D+11	D+16
	1	150	45	20
3	D+2	D+7	D+12	D+17
	2	180	50	40
4	D+3	D+8	D+13	D+18
	5	150	50	45
5	D+4	D+9	D+14	D+19
	3	160	70	30

Note	
1	SORT is a pulse type instruction. It is only executed once. When it is executed again, please turn the instruction input OFF once.

4.8 External Equipment I/O - FN 70 ~ FN 79

In FN 70 ~ FN 79, the command to exchange data between the input and output of the intelligent controller and the external soft component is mainly prepared.

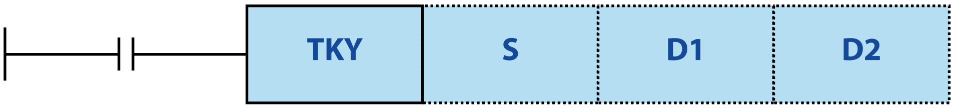
Thanks to these instructions, complex control can be easily implemented with minimal sequence program and external wiring, and therefore has similar features to the convenient instructions described above.

FN No.	Instruction Mark	Instruction Format	Function	Chaper	Page
70	TKY	TKY (S) (D1) (D2) DTKY (S) (D1) (D2)	Number key input	4.8.1	120
71	HKY	HKY (S) (D1) (D2) (D3) DHKY (S) (D1) (D2) (D3)	Hexadecimal numeric key input	4.8.2	122
73	SEGD	SEGD (S) (D) SEGDP (S) (D)	7-segment decoder	4.8.3	124
78	FROM	FROM (m1) (m2) (D) (n)	Module buffer data read	4.8.4	125
79	TO	TO (m1) (m2) (S) (n)	Module buffer data entry	4.8.5	127
176	RD3A	RD3A (m1) (m2) (D)	Analog module readout	4.8.6	129

4.8.1 FN 70 - TKY/Number Key Input

Outline

An instruction sets data such as timers and counters by inputing from 0 to 9 keyboards (number keys).



Data Arrangement FN70 -TKY	Instruction Mark	Execution Condition	Instruction Type	Instruction Step
	TKY	Continuous type	16 bit	7
	DTKY	Continuous type	32 bit	13

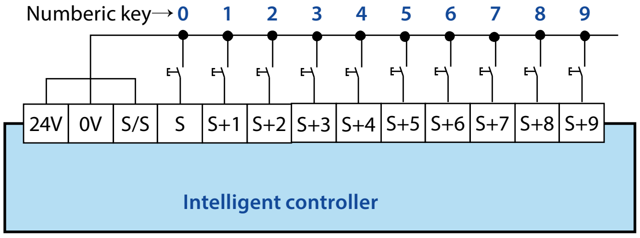
Operand	Setting Data														Data Type				
	S: Enter the start bit soft component of the numeric key [occupies 10 points]														Bit				
	D1: Word soft component number for saving data														16/32 bit				
	D2: Start bit soft component number whose button information is ON [occupies 11 points]														Bit				
	Operand Object Soft Component																		
	Bit Soft Component							Word Soft Component								Others			
	X	Y	M	T	C	S	D.b	KnX	KnY	KnM	KnS	T	C	D	V, Z	H	K	E	P
S	●	●	●			●	●								●				
D1									●	●	●	●	●	●	●				
D2		●	●			●	●								●				

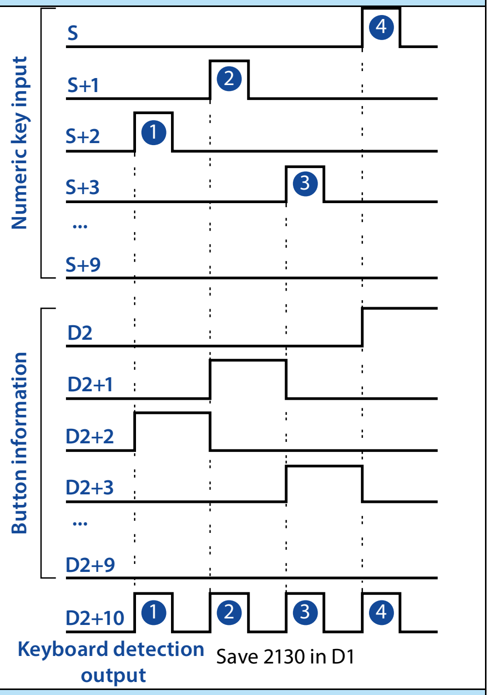
Function and Action Description

16-bit Operation (TKY)

Press [S ~ S+9] on the connected numeric key to press the keyboard, save the input value in D1, and output the keyboard input information and the detected keyboard output in D2 ~ D2+10.

- The value D1 for the input.
  - If it is 9,999 or more, it overflows from a high number of digits.
  - The entered value is saved in BIN (2-digit).
  - In the figure below, press the number keys in the order of 1, 2, 3, and 4, and save as 2130 in D1.
- [D2 ~ D2+10] for button information.
  - Button information of D2 ~ D2+9, according to the pressed button ON/OFF.
  - When any of 0 to 9 is pressed, the keyboard detection output of D2+10 is ON.





32-bit Operation (DTKY)

Press [S ~ S+9] on the connected numeric key to press the keyboard, save the entered value in [D1+1,D1], and output the button information and the detected keyboard output in [D2 ~ D2+10].

- The value D1 for the input.
  - If it is 999,999,999 or more, it overflows from the high digit.
  - The entered value is saved in BIN (2-digit).
- [D2 ~ D2+10] for button information.
  - Button information of D2 ~ D2+9, according to the pressed button ON/OFF.
  - The keyboard detection output of D2+10 is ON, when any one of 0 ~ 9 is pressed.

Note

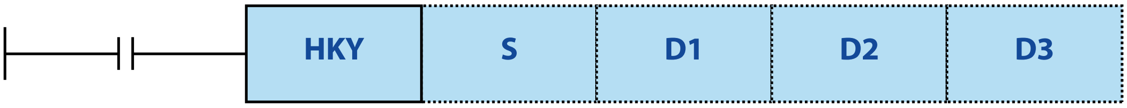
Note		Description
1	When pressing the keyboard at the same time	When multiple keys are pressed at the same time, only the first key pressed is valid.
2	When the command contact is OFF	Even if it is OFF, the content of D1 does not change, but D2 ~ D2+10 turns OFF.
3	Number of occupied soft components	• The input of the number key is connected, occupying 10 points from S. Even if the number key is not connected (not used), it cannot be used for other purposes because it is already occupied. • Occupies 11 points from the start soft component D2 for button information output. Be careful not to repeat the soft components used in other control of the machine. • D2 ~ D2+9: Turn ON according to the input of the number keys 0 ~ 9. • D2+10: Turns ON when any button between 0 and 9 is pressed (keyboard detection output).

4.8.2 FN 71 - HKY/Hexadecimal Numeric Key Input

Outline

Input from 0 to F keyboard (16-key), set the input data for values (0 ~ 9) and operating conditions (A ~ F function keys).

When the extended function is ON, the keyboard can be input using the hexadecimal number from 0 ~ F.



Hexadecimal Data Arrangement FN71 - HKY	Instruction Mark	Execution Condition	Instruction Type	Instruction Step
	HKY	Continuous type	16 bit	9
	DHKY	Continuous type	32 bit	17

Operand	Setting Data	Data Type
	S: Enter the start bit soft component of the 16 key (X) No. (occupies 4 points)	Bit
	D1: Output starting soft component (Y) No. (occupies 4 points)	Bit
	D2: Save the soft component number of the value entered from the 16 key	16/32 bit
	D3: Start bit soft component number whose button information is ON (occupies 8 points)	Bit

Operand	Operand Object Soft Component																		
	Bit Soft Component							Word Soft Component								Others			
	X	Y	M	T	C	S	D.b	KnX	KnY	KnM	KnS	T	C	D	V, Z	H	K	E	P
S	●														●				
D1		●													●				
D2												●	●	●	●				
D3		●	●			●	●								●				

Function and Action Description

16-bit Operation (HKY)

Scan the input of the 16-key (0 ~ F) input [S ~ S+3] and the column output [D1 ~ D1+3], press the 0 ~ 9 button, the value is stored in D2, the keyboard detection output to D3+7 in.

In addition, after pressing the A ~ F keys, the button information corresponding to the keyboard [D3 ~ D3+5] is ON, and the keyboard detection output is to D3+6.

- The value D1 for the input.
  - If it is 9,999 or more, it overflows from a high number of digits.
  - The entered value is stored in D2 as a BIN (binary) value.
  - When any of the keys 0 ~ 9 is pressed, the keyboard detection output D3+7 is ON.

- About the A ~ F key button information D3 ~ D3+6.
  - The 6th point of D3 corresponding to the A ~ F key is ON.
  - When any of the keys A to F is pressed, the keyboard detection output D3+6 is ON.

Keyboard	Button Information	Keyboard	Button Information
A	D3	D	D3+3
B	D3+1	E	D3+4
C	D3+2	F	D3+5

32-bit Operation (DHKY)

Scan the signal connecting the 16-key (0 ~ F) input [S ~ S+3] and the column output [D1,D ~ D1+3], press the 0 ~ 9 button, the value is stored in [D2+1,D2], the keyboard detection output is to D3+7.

In addition, after pressing the A ~ F keys, the button information corresponding to the keyboard [D3 ~ D3+5] is ON, and the keyboard detection output is to D3+6.

- Use the keys from 0 ~ 9 to enter the values [D2+1,D2], D3+7.
  - If it is 999,999,999 or more, it overflows from the high digit.
  - The entered value is stored in [D2+1,D2] in BIN (2-digit) value.
  - When any of the keys 0 to 9 is pressed, the keyboard detection output D3+7 is ON.

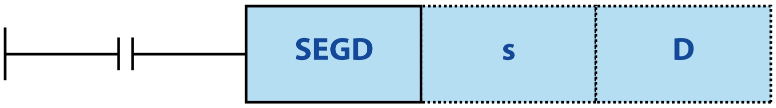
Note

<table>
<tr><th colspan="2">Note</th><th>Description</th></tr>
<tr><td>1</td><td>When pressing the keyboard at the same time</td><td>When multiple keys are pressed at the same time, the first key pressed is valid.</td></tr>
<tr><td>2</td><td>When the command contact is OFF</td><td>Even if it is OFF, the content of D2 does not change, but D3 ~ D3+7 are turned OFF.</td></tr>
<tr><td>3</td><td>Number of occupied soft components</td><td>1) When the 16 button is connected, it takes 4 points from the start soft component S1 of the input (X).
2) When the 16 button is connected, it takes 4 points from the start soft component D1 of the output (Y).
3) It occupies 8 points from the start soft component D3 for button information output.
• Be careful not to repeat the soft components used in other controls in the machine.<table><tr><td>D3 ~ D3+5</td><td>A ~ F button information</td></tr><tr><td>D3+6</td><td>A ~ F key keyboard detection output</td></tr><tr><td>D3+7</td><td>0 ~ 9 key keyboard detection output</td></tr></table></td></tr>
<tr><td>4</td><td>About the read timing of the keyboard input</td><td>Operation cycle of the intelligent controller.
After completing a series of keyboard scans, it takes 8 computation cycles.
In order to prevent read omission due to filter delay of keyboard input, please use the [Constant Scan Mode] and [Timer Interrupt] functions flexibly.</td></tr>
<tr><td>5</td><td>Output form</td><td>Please choose to use the transistor output.</td></tr>
</table>

4.8.3 FN 73 - SEGD/7-segment Decoder

Outline

After digital decoding, light up the 7-segment digital tube (1 digit) instruction.



7-segment Decoder FN73 - SEGD	Instruction Mark	Execution Condition	Instruction Type	Instruction Step
	SEGD	Continuous type	16 bit	5
	SEGDP	Pulse type	16 bit	5

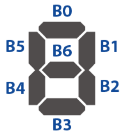
Operand	Setting Data															Data Type		
	S: Decoded start word soft component															16 bit		
	D: Word soft component number for saving data for 7-segment display															16 bit		
	Operand Object Soft Component																	
	Bit Soft Component							Word Soft Component								Others		
	X	Y	M	T	C	S	D.b	KnX	KnY	KnM	KnS	T	C	D	V, Z	H	K	E	P
S								●	●	●	●	●	●	●	●	●	●		
D									●	●	●	●	●	●	●				

Function and Action Description

16-bit Operation (SEGD, SEGDP)

The lower 4 bits (1 digit) of 0 ~ F (16-bit hexadecimal) are decoded into 7-segment code display data and saved to the lower 8 bits of D.

The 7-stage decoding is shown in the table below.

S					7-segment Code Composition	D											Display Data
Hexadecimal Number	b3	b2	b1	b0		B15	…	B8	B7	B6	B5	B4	B3	B2	B1	B0	
0	0	0	0	0		—		—	0	0	1	1	1	1	1	1	0
1	0	0	0	1		—		—	0	0	0	0	0	1	1	0	1
2	0	0	1	0		—		—	0	1	0	1	1	0	1	1	2
3	0	0	1	1		—		—	0	1	0	0	1	1	1	1	3
4	0	1	0	0		—		—	0	1	1	0	0	1	1	0	4
5	0	1	0	1		—		—	0	1	1	0	1	1	0	1	5
6	0	1	1	0		—		—	0	1	1	1	1	1	0	1	6
7	0	1	1	1		—		—	0	0	1	0	0	1	1	1	7
8	1	0	0	0		—		—	0	1	1	1	1	1	1	1	8
9	1	0	0	1		—		—	0	1	1	0	1	1	1	1	9
A	1	0	1	0		—		—	0	1	1	1	0	1	1	1	A
B	1	0	1	1		—		—	0	1	1	1	1	1	0	0	b
C	1	1	0	0		—		—	0	0	1	1	1	0	0	1	C
D	1	1	0	1		—		—	0	1	0	1	1	1	1	0	d
E	1	1	1	0		—		—	0	1	1	1	1	0	0	1	E
F	1	1	1	1		—		—	0	1	1	1	0	0	0	1	F

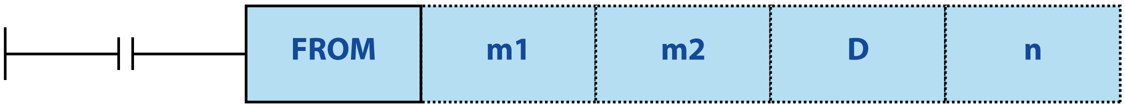
Note

Note		Description
1	Number of occupied soft components	The lower 8 bits of the output of soft component D are occupied, and the upper 8 bits do not change.

4.8.4 FN 78 – FROM/Module Buffer Data Read

Outline

Make the contents of the buffer storage area of the expansion module into the instructions of the programmable controller.



Module Buffer Data Read FN78 - FROM	Instruction Mark	Execution Condition	Instruction Type	Instruction Step
	FROM	Continuous type	16 bit	9

Operand	Setting Data	Data Type
	m1: Unit number (from the right side of the basic unit :K0 ~ K7) [0 ~ 7]	16 bit
	m2: Transmission source (expansion module buffer storage area) [0 ~ 32,765]	16 bit
	D: Soft component number of the transfer destination	16 bit
	n: Number of transfer points (Max. 24 points) [1 ~ 16,383]	16 bit

Operand	Operand Object Soft Component																		
	Bit Soft Component							Bit Soft Component								Bit Soft Component			
	X	Y	M	T	C	S	D.b	KnX	KnY	KnM	KnS	T	C	D	V, Z	H	K	E	P
m1														●		●	●		
m2														●		●	●		
D									●	●	●	●	●	●	●				
n														●		●	●		

Function and Action Description

16-bit Operation (FROM)
Transfer (read out) the n-point 16-bit data, starting from m2 in the buffer memory area of the unit number m1, to the starting n-point of programmable controller D.

Related Soft Component

Soft Component	Name	Content		
M8029	Instruction end flag	Turn ON after completing the current communication, until the next instruction using this flag. It can be placed after this instruction to read the communication status or perform communication control.		
D8262	Expansion module command communication status	0x01: Communication succeeded	0x11: Module does not exist	0x12: Address (channel) overrun
		0x13: Non-analog input module	0x21: Return frame error	0x22: Receive timeout
		0x23: Read data loss	0x25: Lost write data	0x26: Address is not writable

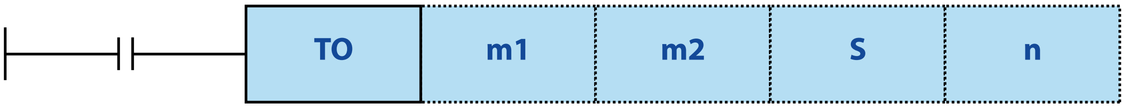
Note

<table>
<tr><th colspan="2">Note</th></tr>
<tr><td>1</td><td>Communication instructions (EXTR/ADPRW/FROM/TO), continuously polling from top to bottom in the order of the program step number, the user only needs to turn on the conditions before the communication instruction, without having to write their own logic for polling control.</td></tr>
<tr><td>2</td><td>Communication instruction (EXTR/ADPRW/FROM/TO), all communicate in a non-blocking way, polling in the background. Each communication instruction may occupy several scan cycles. Do not use pulse signals to control communication instructions (EXTR/ADPRW/FROM/TO), and ensure that the conduction time is long enough, otherwise the communication instructionmay not be triggered.</td></tr>
<tr><td>3</td><td>If need to send a single communication command (EXTR/ADPRW/FROM/TO), or judge whether the current communication command is sent successfully, it can be controlled with M8029.</td></tr>
<tr><td>4</td><td>Communication instruction (EXTR/ADPRW/FROM/TO) is only allowed to be used in the main program. It cannot be used in the following procedures, otherwise it may cause abnormal communication polling.
<table>
<tr><th>Unusable Program Flow</th><th>Note</th></tr>
<tr><td>CJ-P instruction</td><td>Conditional jump</td></tr>
<tr><td>FOR-NEXT instruction</td><td>Cycle</td></tr>
<tr><td>P-SRET instruction</td><td>Subprogram</td></tr>
<tr><td>I-IRET instruction</td><td>Interrupt subprogram</td></tr>
</table>
</td></tr>
</table>

4.8.5 FN 79 – TO/Module Buffer Data Write-in

Outline

An instruction to write data from the programmable controller to the buffer storage area of the expansion module.



Module Buffer Data Write-in FN79 - TO	Instruction Mark	Execution Condition	Instruction Type	Instruction Step
	TO	Continuous type	16 bit	9

Operand	Setting Data	Data Type
	m1: Unit number (from the right side of the basic unit: K0 ~ K7) [0 ~ 7]	16 bit
	m2: Transfer object (expansion module buffer storage area) [0 ~ 32,766]	16 bit
	S: Soft component number of the transfer source data	16 bit
	n: Number of transfer points (Max. 24 points) [1 ~ 32,767]	16 bit

Operand Object Soft Component	Bit Soft Component							Bit Soft Component								Bit Soft Component			
	X	Y	M	T	C	S	D.b	KnX	KnY	KnM	KnS	T	C	D	V, Z	H	K	E	P
m1														●		●	●		
m2														●		●	●		
S								●	●	●	●	●	●	●	●	●	●		
n														●		●	●		

Function and Action Description

16-bit Operation (TO)
Transfer (write in) the first n points of 16-bit data in the programmable controller to the n points starting from m2 in the buffer memory area of the expansion module with unit number m1.

Related Soft Component

Soft Component	Name	Content		
M8029	Instruction end flag	Turn ON after completing the current communication, until the next instruction using this flag. It can be placed after this instruction to read the communication status or perform communication control.		
D8262	Expansion module command communication status	0x01: Communication succeeded	0x11: Module does not exis	0x12: Address (channel) overrun
		0x13: Non-analog input module	0x21: Return frame error	0x22: Receive timeout
		0x23: Read data loss	0x25: Lost write data	0x26: Address is not writable

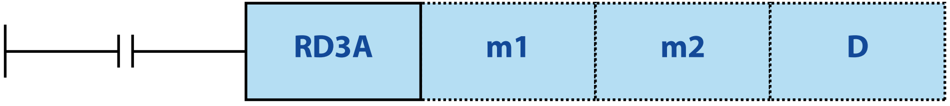
Note

<table>
<tr><th colspan="2">Note</th></tr>
<tr><td>1</td><td>The communication instructions (EXTR/ADPRW/FROM/TO) are continuously polled from top to bottom according to the sequence of the program step number. The user only needs to turn on the conditions before the communication instruction, without having to write their own logic for polling control.</td></tr>
<tr><td>2</td><td>The communication commands (EXTR/ADPRW/FROM/TO) all communicate in a non-blocking manner and poll in the background. Each communication command may occupy several scan cycles. Do not use pulse signals to control the communication commands (EXTR/ADPRW /FROM/TO) and ensure that the conduction time is long enough, otherwise the communication command may not be triggered.</td></tr>
<tr><td>3</td><td>If need to send a single communication command (EXTR/ADPRW/FROM/TO), or judge whether the current communication command is sent successfully, it can be controlled with M8029.</td></tr>
<tr><td>4</td><td>Communication instruction (EXTR/ADPRW/FROM/TO) is only allowed to be used in the main program. It cannot be used in the following procedures, otherwise it may cause abnormal communication polling.
<table>
<tr><th>Unusable Program Flow</th><th>Note</th></tr>
<tr><td>CJ-P instruction</td><td>Conditional jump</td></tr>
<tr><td>FOR-NEXT instruction</td><td>Cycle</td></tr>
<tr><td>P-SRET instruction</td><td>Subprogram</td></tr>
<tr><td>I-IRET instruction</td><td>Interrupt subprogram</td></tr>
</table>
</td></tr>
</table>

4.8.6 FN 176 – RD3A/Analog Module Readout

Outline

The instruction to read the analog input value of the analog module.



Analog Module Readout FN176 - RD3A	Instruction Mark	Execution Condition	Instruction Type	Instruction Step
	RD3A	Continuous type	16 bit	7
	RD3AP	Pulse type	16 bit	7

Operand	Setting Data	Data Type
	m1: Unit number (from the right side of the basic unit: K0 ~ K7)	16 bit
	m2: Analog input channel number	16 bit
	D: Word device that stores the read data	16 bit

Operand Object Soft Component	Bit Soft Component							Bit Soft Component								Bit Soft Component			
	X	Y	M	T	C	S	D.b	KnX	KnY	KnM	KnS	T	C	D	V, Z	H	K	E	P
m1								●	●	●	●	●	●	●	●	●	●		
m2								●	●	●	●	●	●	●	●	●	●		
D									●	●	●	●	●	●	●				

Function and Action Description

16-bit Operation (FROM)
The instruction to read the analog input value of the analog module. The main module of PLC will regularly update the analog input value of the analog module to the buffer, and the analog input value stored in the buffer can be directly read through the RD3A, which is faster than the FROM/TO instruction, and the timeliness of the analog input value has been guaranteed. This instruction can be completed immediately and will not involve multiple cycles.

Related Soft Component

Soft Component	Name	Content		
D8262	Expansion module command communication status	0x01: Communication succeeded	0x11: Module does not exis	0x12: Address (channel) overrun
		0x13: Non-analog input module	0x21: Return frame error	0x22: Receive timeout
		0x23: Read data loss	0x25: Lost write data	0x26: Address is not writable

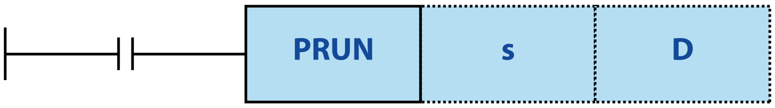
4.9 External Soft Component SER (Option Soft Component) - FN 80 ~ FN 89

FN No.	Instruction Mark	Instruction Format	Function	Chapter	Page
81	PRUN	PRUN (S) (D1) PRUNP (S) (D1) DPRUN (S) (D1) DPRUNP (S) (D1)	Octet bit transfer	4.9.1	131
84	CCD	CCD (S) (D) (n) CCDP (S) (D)	Check code	4.9.2	133
85	PID	PID (S1) (S2) (S3) (D)	PID operation	4.9.3	135

4.9.1 FN 81 - PRUN/Octet Bit Transfer

Outline

The soft component number of the S and D that have been specified by the number of bits is treated as an octal number, and the data is transmitted.



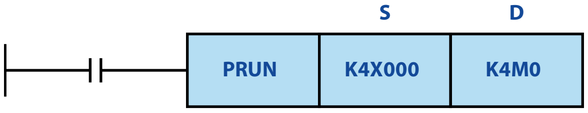
	Instruction Mark	Execution Condition	Instruction Type	Instruction Step
Octet Bit Transfer FN81 - PRUN	PRUN	Continuous type	16 bit	5
	PRUNP	Pulse type	16 bit	5
	DPRUN	Continuous type	32 bit	9
	DPRUNP	Pulse type	32 bit	9

Operand	Setting Data															Data Type				
	S: Source soft component number															16/32 bit				
	D: Target soft component number															16/32 bit				
	Operand Object Soft Component																			
	Bit Soft Component							Word Soft Component									Others			
	X	Y	M	T	C	S	D.b	KnX	KnY	KnM	KnS	T	C	D	V, Z	H	K	E	P
S								●		●					●				
D									●	●					●				

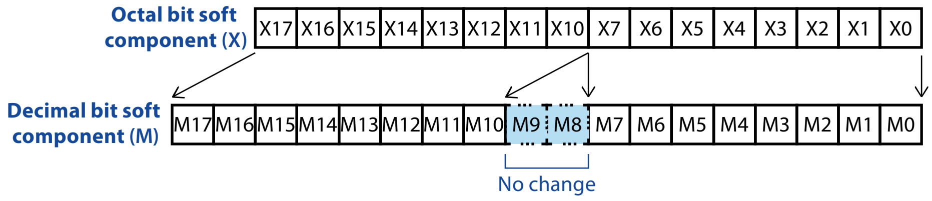
Function and Action Description

16-bit Operation (PRUN, PRUNP)

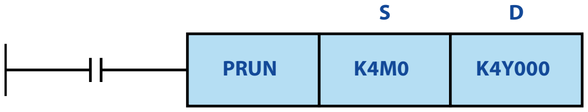
Octal bit soft component→decimal bit soft component:



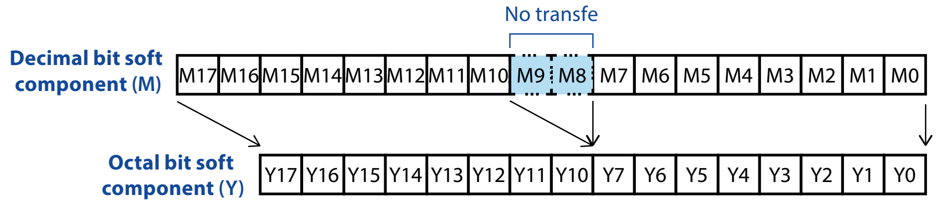
X000~X017→M0~M7, M10~M17


Octal bit soft component (X)
X17 X16 X15 X14 X13 X12 X11 X10 X7 X6 X5 X4 X3 X2 X1 X0
Decimal bit soft component (M)
M17 M16 M15 M14 M13 M12 M11 M10 M9 M8 M7 M6 M5 M4 M3 M2 M1 M0
No change

Decimal bit soft component→octal bit soft component:

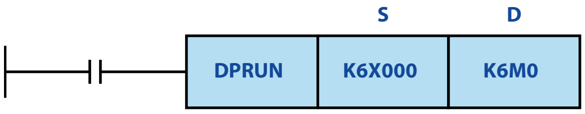


M0~M7, M10~M17→Y0~Y17

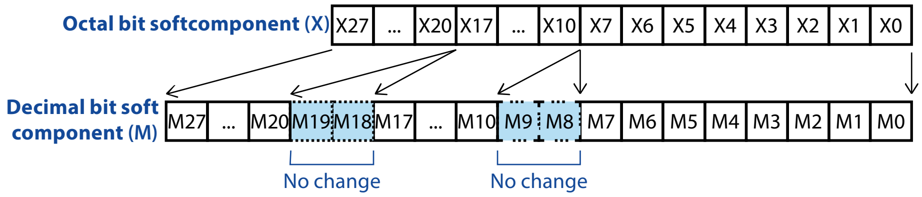

No transfe
Decimal bit soft component (M)
M17 M16 M15 M14 M13 M12 M11 M10 M9 M8 M7 M6 M5 M4 M3 M2 M1 M0
Octal bit soft component (Y)
Y17 Y16 Y15 Y14 Y13 Y12 Y11 Y10 Y7 Y6 Y5 Y4 Y3 Y2 Y1 Y0

32-bit Operation (DPRUN, DPRUNP)

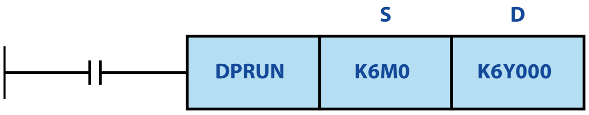
Octal bit soft component→decimal bit soft component:



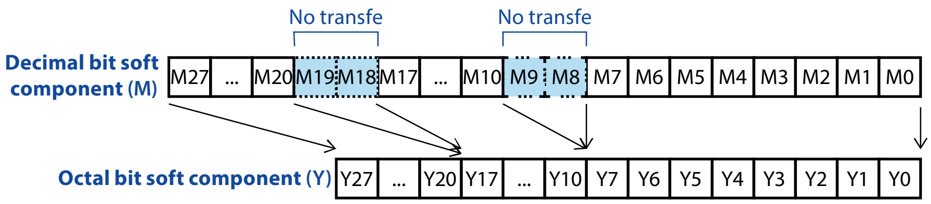
X000~X027→
M0~M7, M10~M17, M20~M27


Octal bit softcomponent (X)
X27 ... X20 X17 ... X10 X7 X6 X5 X4 X3 X2 X1 X0
Decimal bit soft component (M)
M27 ... M20 M19 M18 M17 ... M10 M9 M8 M7 M6 M5 M4 M3 M2 M1 M0
No change
No change

Decimal bit soft component→octal bit soft component:



M0~M7, M10~M17, M20~M27 □
→Y000~Y027


No transfe
No transfe
Decimal bit soft component (M)
M27 ... M20 M19 M18 M17 ... M10 M9 M8 M7 M6 M5 M4 M3 M2 M1 M0
Octal bit soft component (Y)
Y27 ... Y20 Y17 ... Y10 Y7 Y6 Y5 Y4 Y3 Y2 Y1 Y0

Note

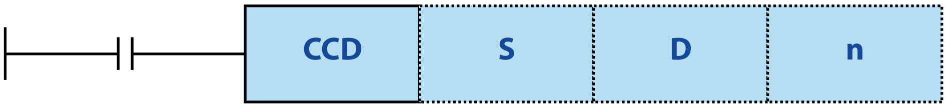
Note	
1	The intelligent controller's own Modbus communication (ADPRW) and CAN communication (EXTR) have their own data verification, no need to add verification by the user.

4.9.2 FN 84 - CCD/Check Code

Outline

The error check method used in communication, etc., has a horizontal check and a checksum, which is used to calculate the check value. In the error check method, in addition to these, there is a CRC (Cyclic Redundancy Check).

When using the CRC value, please use the CRC instruction.



Check Code FN84 -CCD	Instruction Mark	Execution Condition	Instruction Type	Instruction Step
	CCD	Continuous type	16 bit	7
	CCDP	Pulse type	16 bit	7

Operand	Setting Data	Data Type
	S: Starting number of the object soft component	16 bit/string
	D: The starting number of the soft component that saves the calculated data	16 bit/string
	n: Number of data [setting range: 1 ~ 256]	16 bit/string

Operand Object Soft Component	Bit Soft Component							Word Soft Component								Others			
	X	Y	M	T	C	S	D.b	KnX	KnY	KnM	KnS	T	C	D	V, Z	H	K	E	P
S								●	●	●	●	●	●	●	●				
D									●	●	●	●	●	●	●				
n														●		●	●		

Function and Action Description

16-bit Operation (CCD/CCDP)

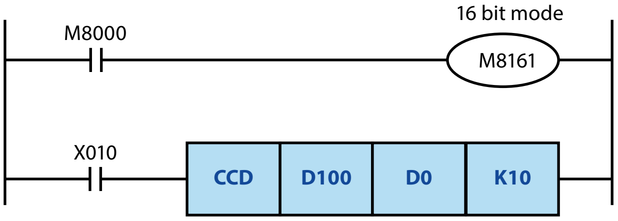
Calculate the sum and level check of the data saved in S ~ S+n-1, save the sum data in D, and save the horizontal check in D+1.

In this command, the modes used for calculation are 16 bit mode and 8-bit mode. For their respective actions, please refer to the following page.

"16 bit conversion mode" when M8161 = OFF

- For the n-point data starting with S, save the sum of the 8-bit data and the horizontal checksum to the D and D+1 soft components.
- When using the 16 bit conversion mode, set the M8161 to OFF all the time.
- M8161 is cleared when RUN→STOP.

Example: When the following program is used, the conversion is performed as shown below.



S	Example of Data Content	
D100 low	K100	= 01100100
D100 high	K111	= 0110111 ①
D101 low	K100	= 01100100
D101 high	K98	= 01100010
D102 low	K123	= 0111101 ①
D102 high	K66	= 01000010
D103 low	K100	= 01100100
D103 high	K95	= 0101111 ①
D104 low	K210	= 11010010
D104 high	K88	= 01011000
Total	K1091	
Level check	1000010 ① • When the number of 1 is odd, the level is 1. • When the number of 1 is even, the horizontal check is 0.	

D0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	1	1	Use BCD to represent 1091

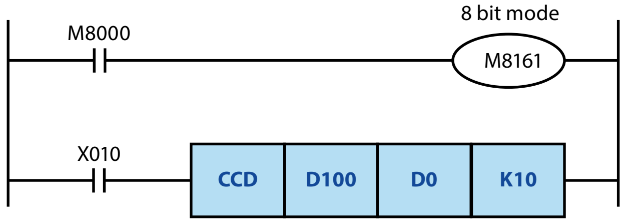
D0	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0	1	Horizontal check

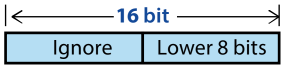
16-bit Operation (CCD/CCDP)

"8-bit conversion mode" when M8161 = ON

- For n-point data starting with S (lower bits are only 8 bits), save the sum and level check to the D and D+1 soft components respectively.
- When using the 8-bit conversion mode, always turn the M8161 ON.
- M8161 is cleared when RUN→STOP.

Example: When the following program is used, the conversion is performed as shown below.





Source data

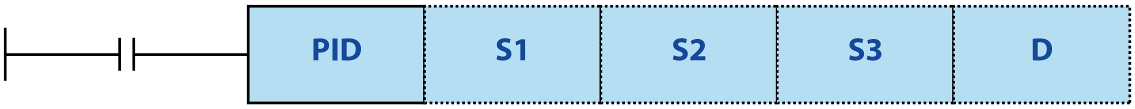
S	Example of Data Content	
D100	K100	= 01100100
D101	K111	= 0110111①
D102	K100	= 01100100
D103	K98	= 01100010
D104	K123	= 0111101①
D105	K66	= 01000010
D106	K100	= 01100100
D107	K95	= 0101111①
D108	K210	= 11010010
D109	K88	= 01011000
Total	K1091	
Level check	1000010① • When the number of 1 is odd, the level is 1. • When the number of 1 is even, the horizontal check is 0.	

D0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	1	1	Use BCD to represent 1091
D0	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0	1	Horizontal check

4.9.3 FN 88 - PID/PID Operation

Outline

This instruction is used to perform PID control that changes the output value according to the amount of change in the input.



PID Operation FN88 -PID	Instruction Mark	Execution Condition	Instruction Type	Instruction Step
	PID	Continuous type	16 bit instruction	9

	Setting Data	Data Type
Operand	S1: Data register number of the save target value (SV)	16 bit
	S2: Save the data register number of the measured value (PV)	16 bit
	S3: Data register number of the saved parameter	16 bit
	D: Save the data register number of the output value (MV)	16 bit

Operand	Operand Object Soft Component																		
	Bit Soft Component							Word Soft Component								Others			
	X	Y	M	T	C	S	D.b	KnX	KnY	KnM	KnS	T	C	D	V, Z	H	K	E	P
S1														●					
S2														●					
S3														●					
D														●					

Function and Action Description

16-bit Operation (PID)

After executing the program for setting the target value S1, the measured value S2, and the parameters S3 to S3+6, the operation result (MV) is saved to the output value D every sampling time S3.

The setting items are shown in the table below.

Setting Item		Content	Occupied Points
S1	Target value (SV)	Set the target value (SV). The PID instruction does not change the setting contents.	1 point
S2	Measured value (PV)	Input value of the PID operation.	1 point
S3	Parameter	Self-tuning: In the case of the limit cycle method, it occupies the 29-point soft component starting from the starting soft component specified in S3.	29 point
D	Output value (MV)	PID control (when normal processing): The initial output value is set on the user side before the command is driven. The result of the operation will be saved. Self-tuning: In the case of the limit cycle method, the ULV value or LLV value is automatically output during the auto-tuning process. When the auto-tuning is finished, the established MV value is set.	1 point

16-bit Operation (PID)

The parameters S3 ~ S3+28 are shown in the table below.

Setting Item			Setting Content	Remarks
S3	Sampling time Ts		1 ~ 32,767ms	A value shorter than the calculation period cannot be executed
S3+1	Action setting ACT	Bit0	0: Positive action 1: Reverse action	Direction of action
		Bit1	0: No input change alarm 1: Input change alarm is valid	
		Bit2	0: No output change alarm 1: Output change alarm is valid	Do not turn ON both Bit2 and Bit5 at the same time
		Bit3	Reserved	
		Bit4	0: Self-tuning does not work 1: Perform auto-tuning	
		Bit5	0: No output value upper and lower limit setting 1: Output value upper and lower limit settings are valid	Do not turn ON both Bit2 and Bit5 at the same time
		Bit6	0: Reserved 1: Limit cycle method	Select the mode of auto-tuning
		Bit7	0: PID auto-tuning 1: PI auto-tuning	
		Bit8 ~ Bit5	Not avaliable	
S3+2	Input filter constant α		0 ~ 99 (%)	0: No input filtering
S3+3	Proportional gain Kp		1 ~ 32,767 (%)	
S3+4	Integration time TI		0 ~ 32,767 (× 100ms)	0: Treated as ∞ (no points)
S3+5	Differential gain TD		0 ~ 100 (%)	0: No differential gain
S3+6	Differential time TD		0 ~ 32,767 (× 100ms)	0: No differentiation
S3+7 … S3+19	It is occupied by the internal processing of the PID operation. Please do not change the data.			
S3+20	Input change amount (increase side) alarm set value		0 ~ 32,767	Action binding ACT (S3+1) Bit1 = 1 is valid
S3+21	Input change amount (reduction side) alarm set value		0 ~ 32,767	Action binding ACT (S3 1) Bit1 = 1 is valid
S3+22	Output change amount (increase side) alarm set value		0 ~ 32,767	Action binding ACT (S3+1) Bit2 = 1, Bit5 = 0 is valid
	Output upper limit setting		-32,768 ~ +32,767	Action binding ACT (S3+1) Bit2 = 0, Bit5 = 1 is valid
S3+23	Output change amount (reduction side) alarm set value		0 ~ 32,767	Action binding ACT (S3+1) Bit2 = 1, Bit5 = 0 is valid
	Set value of output lower limit		-32,768 ~ +32,767	Action binding ACT (S3+1) Bit2 = 0, Bit5 = 1 is valid
S3+24	Alarm output	Bit0	0: Input change amount (increase side) is overflow	Bit0, Bit1: Action setting ACT (S3+1) Bit1 = 1 is valid
		Bit1	0: Input change amount (reduction side) is overflow	
		Bit2	0: Output change amount (increase side) is overflow	Bit2, Bit3: Action setting ACT (S3+1) Bit2 = 1 is valid
		Bit3	0: Output change amount (reduction side) is overflow	
S3+25	PV value threshold (hysteresis) width SHPV		Set according to fluctuations in measured value (PV)	Action setting (ACT) Bit6 = 1 occupied when the limit cycle method (ON) is selected
S3+26	Output value upper limit ULV		Output value (MV) Max. output value ULV setting	
S3+27	Output value lower limit LLV		Output value (MV) Min. output value LLV setting	
S3+28	PID auto-tuning Max. time		1 ~ 32,767 (unit 100ms)	

Note

Note		Description
1	When using multiple instructions	Can be executed multiple times at the same time (the number of loops is not limited). However, please note that the soft component numbers of S3 and D used in the calculation cannot be repeated.
2	Number of occupied points of parameter S3	The case of the limit cycle method. • Occupy 29-point soft component starting from the starting soft component specified in S3.
3	When specifying the soft component of the power failure holding area	For the output value (MV) of the PID instruction, specify the data register D except the power-down holding area. When specifying the data register of the power failure holding area, please clear the contents of the backup.

Error

Error	
1	After an operation error occurs, the special auxiliary relay M8067 is turned ON, and the error code is stored in the special data register D8067.

4.10 Data Transfer 2 - FN 100 ~ FN 109

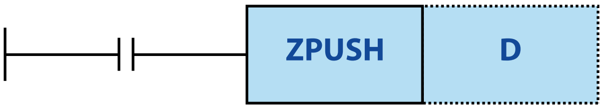
In FN 100 ~ FN 109, instructions for performing special processing are more complex than basic application instruction processing.

FN No.	Instruction Mark	Instruction Format	Function	Chapter	Page
102	ZPUSH	ZPUSH (D) ZPUSHP (D)	Bulk storage of index register	4.10.1	139
103	ZPOP	ZPOP (D) ZPOPP (D)	Restoration of index register	4.10.2	141

4.10.1 FN 102 - ZPUSH/Bulk Storage of Index Register

Outline

Instruction to temporarily save the current values of the index registers V0 ~ V7, Z0 ~ Z7.



To return the temporarily saved current value, use the ZPOP (FN 103) instruction.

Bulk Storage of Index Register FN102 - ZPUSH	Instruction Mark	Execution Condition	Instruction Type	Instruction Step
	ZPUSH	Continuous type	16 bit	3
	ZPUSHP	Pulse type	16 bit	3

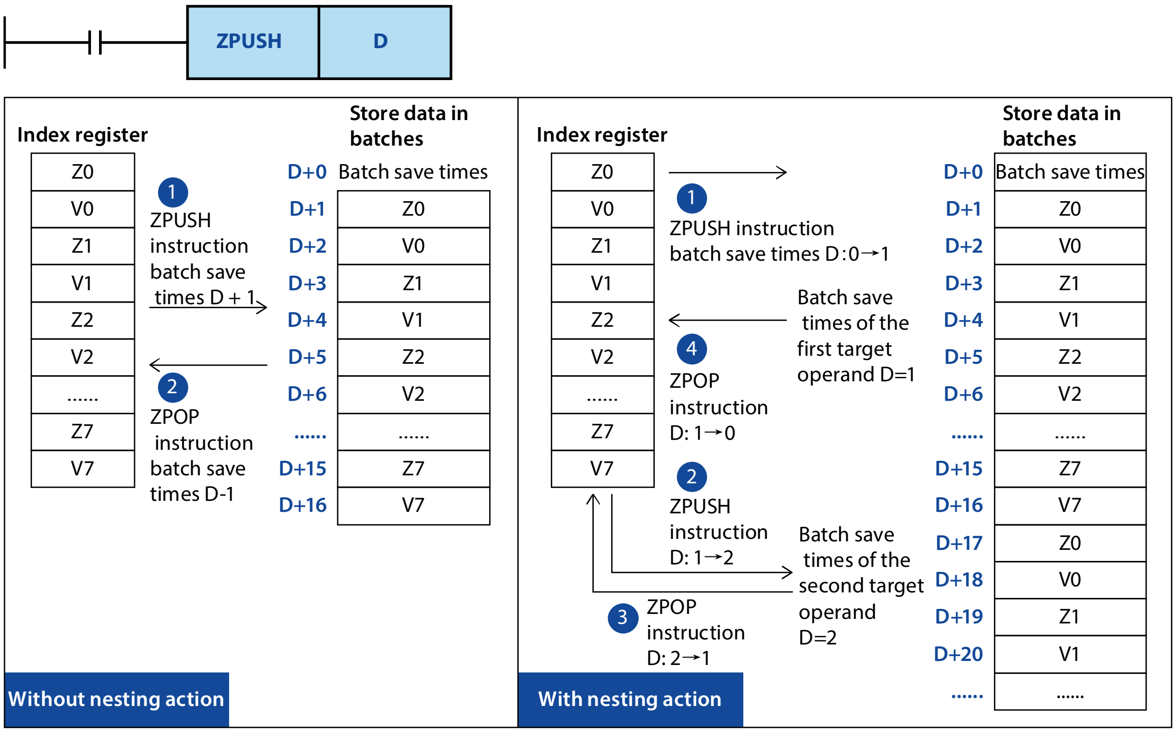
Operand	Setting Data	Data Type
	Soft component start number for temporarily saving the current values of the index registers V0 to V7 and Z0 to Z7 D: Batch save times D+1 ~ D+16 × batch save times: The location where the saved data is saved in batches	16 bit

Operand	Operand Object Soft Component																		
	Bit Soft Component							Word Soft Component								Others			
	X	Y	M	T	C	S	D.b	KnX	KnY	KnM	KnS	T	C	D	V, Z	H	K	E	P
D														●					

Function and Action Description

16-bit Operation (ZPUSH, ZPUSHP)

- The contents of the index registers Z0 ~ Z7, V0 ~ V7 are stored in batches in the soft component starting with D. After the contents of the index register are saved in batches, the batch save count D is +1.
- Use the ZPOP (FN 103) instruction to return data.
  Use the ZPUSH (FN 102), ZPOP (FN 103) instructions in pairs.
- By specifying the same soft component for D, you can nest the ZPUSH (FN 102) ~ ZPOP (FN 103) instructions. At this time, each time the ZPUSH (FN 102) instruction is executed, the area that D starts to use increases by 16 points each time. Therefore, please ensure the area of the number of times used in nesting in advance.
- The structure of the data after D is saved in batches is as follows.



Related Instruction

Instruction	Content
ZPOP (FN 103)	Restoration of index registers V0 ~ V7, Z0 ~ Z7 temporarily saved in batches by the ZPUSH (FN 102) instruction.

Note

Note	
1	When there is no nesting action, please clear the batch save times before executing the ZPUSH (FN 102) instruction.
2	When there is nesting action, please clear the batch save times before the first execution.

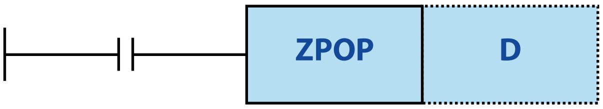
Error

Error	
1	In some cases, an operation error will occur. The error flag M8067 turns ON and the error code is stored in D8067. • In the ZPUSH (FN 102) command, when the range of points at which D starts to use exceeds the range of the corresponding soft component (error code: K6706). • When the ZPUSH (FN 102) instruction is executed, D (the number of batch saves) is negative (error code: K6707).

4.10.2 FN 103 - ZPOP/Restoration of Index Register

Outline

The instruction to restore the index register saved in batch by ZPUSH.



Restoration of the Index Register FN103 - ZPOP	Instruction Mark	Execution Condition	Instruction Type	Instruction Step
	ZPOP	Continuous type	16 bit	3
	ZPOPP	Pulse type	16 bit	3

Operand	Setting Data														Data Type				
	The starting number of the soft component that temporarily stores the contents of the index registers V0 ~ V7, Z0 ~ Z7 in batches D: Batch save times D+1 ~ D +16 × batch save times: Data save location saved in batches														16 bit				
	Operand Object Soft Component																		
	Bit Soft Component							Word Soft Component								Others			
	X	Y	M	T	C	S	D.b	KnX	KnY	KnM	KnS	T	C	D	V, Z	H	K	E	P
D														●					

Function and Action Description

16-bit Operation (ZPOP, ZPOPP)
• The contents of the index registers V0 to V7 and Z0 to Z7 that have been temporarily saved in batches to the soft component starting with D using the ZPUSH (FN 102) instruction are restored to the original index register. The contents of the index register are restored, the number of batch saves D is -1. • Use the ZPUSH (FN 102) command to temporarily save data in batches. ZPUSH (FN 102) and ZPOP (FN 103) instructions are used in pairs.

Related Instruction

Instruction	Content
ZPUSH (FN 102)	The instruction to temporarily store the current values of the index registers V0 ~ V7, Z0 ~ Z7 in batches.

Error

Error	
1	When the ZPOP (FN 103) instruction is executed, when the content of the batch save D is 0 or a negative number, an operation error occurs. The error flag M8067 turns ON, and the error code (K6706) is stored in D8067.

4.11 Floating Point Arithmetic - FN 110 ~ FN 139

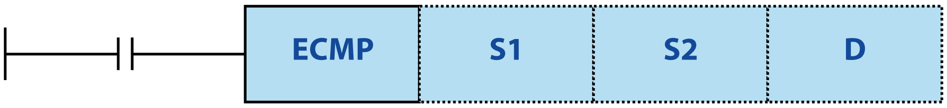
FN 110 ~ FN 119, FN 120 ~ FN 129, FN 130 ~ FN 139 provide instructions for conversion, comparison, four operations, square root operations, trigonometric functions, etc. for floating point numbers.

FN No.	Instruction Mark	Instruction Format	Function	Chapter	Page
110	ECMP	ECMP (S1) (S2) (D) ECMPP (S1) (S2) (D)	Binary floating point ratio	4.11.1	143
111	EZCP	EZCP (S1) (S2) (D) EZCPP (S1) (S2) (D)	Binary floating point interval ratio	4.11.2	144
112	EMOV	DEMOV (S) (D) DEMOVP (S) (D)	Binary floating point data communication	4.11.3	145
118	EBCD	DEBCD (S) (D) DEBCDP (S) (D)	Conversion from binary floating point number to decimal floating point number	4.11.4	146
119	EBIN	DBIN (S) (D) DBINP (S) (D)	Conversion from binary to decimal floating point numbers	4.11.5	147
120	EADD	DEADD (S1) (S2) (D) DEADDP (S1) (S2) (D)	Binary floating point addition	4.11.6	148
121	ESUB	DESUB (S1) (S2) (D) DESUBP (S1) (S2) (D)	Binary floating point subtraction	4.11.7	149
122	EMUL	DEMUL (S1) (S2) (D) DEMULP (S1) (S2) (D)	Binary floating point multiplication	4.11.8	150
123	EDIV	DEDIV (S1) (S2) (D) DEDIVP (S1) (S2) (D)	Binary floating point division division	4.11.9	151
124	EXP	DEXP (S) (D) DEXPP (S) (D)	Binary floating point index operation	4.11.10	152
125	LOGE	LOGE (S) (D) DLOGEP (S) (D)	Binary floating point natural logarithm operation	4.11.11	153
126	LOG10	LOG10 (S) (D) DLOG10P (S) (D)	Binary floating point number common logarithm operation	4.11.12	154
127	ESQR	DESQP (S) (D) DESQPP (S) (D)	Binary floating point number square operation	4.11.13	155
128	ENEG	DENEG (D) DENEGP (D)	Binary floating point number flip	4.11.14	156
129	INT	INT (S) (D) INTP (S) (D) DINT (S) (D) DINTP (S) (D)	Conversion from binary floating point number to BIN integer	4.11.15	157
130	SIN	DSIN (S) (D) DSINP (S) (D)	Binary floating point number SIN operation	4.11.16	158
131	COS	DCOS (S) (D) DCOSP (S) (D)	Binary floating point number COS operation	4.11.16	158
132	TAN	DTAN (S) (D) DTANP (S) (D)	Binary floating point TAN operation	4.11.18	159
133	ASIN	DASIN (S) (D) DASINP (S) (D)	Binary floating point number SIN^{-1} operation	4.11.19	160
134	ACOS	DACOS (S) (D) DACOSP (S) (D)	Binary floating point number COS^{-1} operation	4.11.20	161
135	ATAN	DATAN (S) (D) DATANP (S) (D)	Binary floating point number TAN^{-1} operation	4.11.21	162
136	RAD	DRAD (S) (D) DRADP (S) (D)	Conversion of binary floating point radians→angle	4.11.22	163
137	DEG	DDEG (S) (D) DDEGP (S) (D)	Conversion of binary floating point radians→angle	4.11.22	163

4.11.1 FN 110 - ECMP/Binary Floating Point Ratio

Outline

Compare 2 data (binary floating point numbers) and output the result (greater than, equal to or less than) to the instruction in the bit soft component (3 points).



Binary Floating Point Ratio FN110 - ECMP	Instruction Mark	Execution Condition	Instruction Type	Instruction Step
	DECMP	Continuous type	32 bit	13
	DECMPP	Pulse type	32 bit	13

Operand	Setting Data	Data Type
	S1: Save the soft component number of the binary floating point data to be compared	Real number (binary)
	S2: Save the soft component number of the binary floating point data to be compared	Real number (binary)
	D: Start bit soft component number of the output result (occupies 3 points)	Bit

Operand	Operand Object Soft Component																		
	Bit Soft Component							Word Soft Component								Others			
	X	Y	M	T	C	S	D.b	KnX	KnY	KnM	KnS	T	C	D	V, Z	H	K	E	P
S1														●	●	●	●	●	
S2														●	●	●	●	●	
D		●	●			●	●								●				

Function and Action Description

32-bit Operation (DECMP, DECMPP)
Compare the comparison value [S1+1,S1] and the comparison source [S2+1,S2] as floating point data, and then [D,D+1,D+2] according to the result of the comparison (less than, equal to, greater than) any one of the positions is ON. • When a constant (K, H) is specified in [S1+1,S1], [S2+1,S2], the value is automatically converted from BIN to binary floating point number and then processed. • [D]: [S1+1,S1] > [S2+1,S2] turns ON. • [D+1]: [S1+1,S1] = [S2+1,S2] turns ON. • [D+2]: [S1+1,S1] < [S2+1,S2] turns ON.

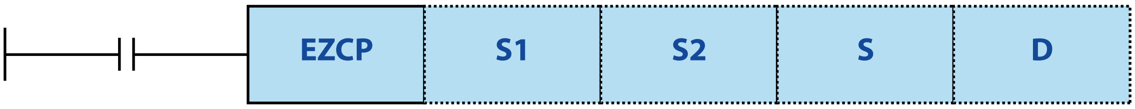
Note

Note		Description
1	Number of occupied soft components	D takes up 3 points. Please be careful not to repeat with other soft components for other purposes.

4.11.2 FN 111 - EZCP/Binary Floating Point Interval Ratio

Outline

The comparison range of the upper and lower points is compared with the data (binary floating point number), and the result is output to the bit soft component (3 points) according to the result.



Binary Floating Point Interval Ratio FN111 - EZCP	Instruction Mark	Execution Condition	Instruction Type	Instruction Step
	DEZCP	Continuous type	32 bit	17
	DEZCPP	Pulse type	32 bit	17

Operand	Setting Data	Data Type
	S1: Save the soft component number of the binary floating point data to be compared	Real number (binary)
	S2: Save the soft component number of the binary floating point data to be compared	Real number (binary)
	S: Save the soft component number of the binary floating point data to be compared	Real number (binary)
	D: Start bit soft component number of the output result (occupies 3 points)	Bit

	Operand Object Soft Component																	
	Bit Soft Component							Word Soft Component							Others			
	X	Y	M	T	C	S	D.b	KnX	KnY	KnM	KnS	T	C	D	V, Z	H	K	E	P
S1														●	●	●	●	●	
S2														●	●	●	●	●	
S														●	●	●	●	●	
D		●	●			●	●								●				

Function and Action Description

32-bit Operation (DEZCP, DEZCPP)
Compare the comparison values [S1+1,S1], [S2+1,S2] and the comparison source [S+1,S] as floating point data, and then [D,D+1,D+2] according to the result (less than, equal to or greater than) any one of the positions is is ON. • When a constant (K, H) is specified in [S1+1,S1], [S2+1,S2], [S+1,S], the value is automatically converted to a binary floating point number and then processed. • [D]: [S1+1,S1] > [S+1,S] turns ON. • [D+1]: When [S1+1,S1] ≤ [S+1,S] ≤ [S2+1,S2] turns ON. • [D+2]: [S+1,S] > [S2+1,S2] turns ON. Even if the command input is OFF and the DEZCP command is not executed, the bits of D ~ D+2 can maintain the state before the command input is turned OFF.

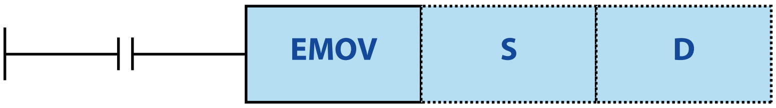
Note

Note		Description
1	Number of occupied soft components	D takes up 3 points. Please be careful not to repeat with other soft components for other purposes.
2	Comparison data about S1 and S2	For the size relationship of the comparison data, set it to [S1+1,S1] ≤ [S2+1,S2]. In the case of [S1+1,S1] > [S2+1,S2], the value of [S2+1,S2] is regarded as the same as [S1+1,S1], and thus is compared.

4.11.3 FN 112 - EMOV/Binary Floating Point Data Communication

Outline

An instruction to transfer binary floating point data.



Binary Floating Point Data Communication FN112 - EMOV	Instruction Mark	Execution Condition	Instruction Type	Instruction Step
	DEMOV	Continuous type	32 bit	9
	DEMOVP	Pulse type	32 bit	9

Operand	Setting Data													Data Type				
	S: Binary floating point data of the transfer source, or the soft component number of the saved data													Real number (binary)				
	D: Soft component number for saving binary floating point data													Real number (binary)				
	Operand Object Soft Component																	
	Bit Soft Component							Word Soft Component								Others			
	X	Y	M	T	C	S	D.b	KnX	KnY	KnM	KnS	T	C	D	V, Z	H	K	E	P
S														●	●			●	
D														●	●				

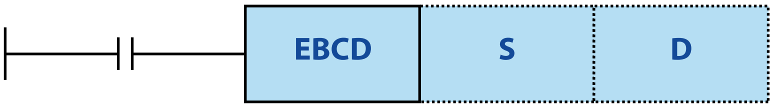
Function and Action Description

32-bit Operation (DEMOV, DEMOVP)
Transfer the contents of the transfer source [S+1,S] (binary floating point data) to [D+1,D]. In addition, you can also specify the real number (E) directly in S.

4.11.4 FN 118 - EBCD/Conversion from Binary Floating Point Number to Decimal Floating Point Number

Outline

An instruction to convert a binary floating point number in a soft component to a→10 floating point number.



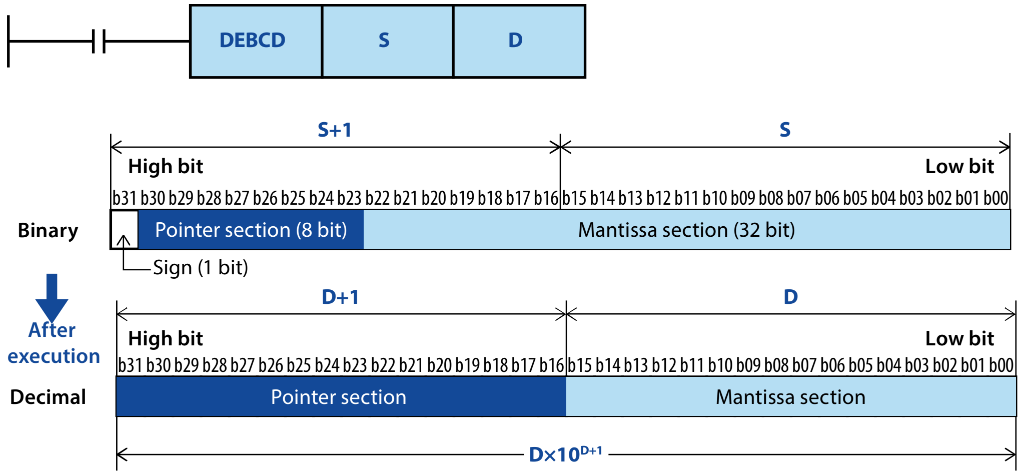
Conversion from Binary Folating Point Number to Decimal Floating Point Number FN118 - EBCD	Instruction Mark	Execution Condition	Instruction Type	Instruction Step
	DEBCD	Continuous type	32 bit	9
	DEBCDP	Pulse type	32 bit	9

Operand	Setting Data														Data Type				
	S: Data register number for saving binary floating point data														Real number (binary)				
	D: Save the data register number of the converted decimal floating point data														Real number (decimal)				
	Operand Object Soft Component																		
	Bit Soft Component							Word Soft Component								Others			
	X	Y	M	T	C	S	D.b	KnX	KnY	KnM	KnS	T	C	D	V, Z	H	K	E	P
S														●	●				
D														●	●				

Function and Action Description

32-bit Operation (DEBCD, DEBCDP)

Convert the binary floating point number of [S+1,S] to a decimal floating point number and transfer it to [D+1,D].


DEBCD
S
D
S+1
S
High bit
Low bit
b31 b30 b29 b28 b27 b26 b25 b24 b23 b22 b21 b20 b19 b18 b17 b16 b15 b14 b13 b12 b11 b10 b09 b08 b07 b06 b05 b04 b03 b02 b01 b00
Binary
Pointer section (8 bit)
Mantissa section (32 bit)
Sign (1 bit)
After execution
D+1
D
High bit
Low bit
b31 b30 b29 b28 b27 b26 b25 b24 b23 b22 b21 b20 b19 b18 b17 b16 b15 b14 b13 b12 b11 b10 b09 b08 b07 b06 b05 b04 b03 b02 b01 b00
Decimal
Pointer section
Mantissa section
D×10^D+1

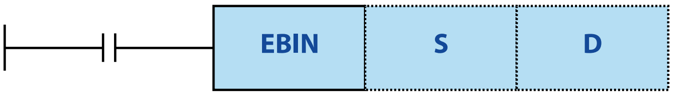
Note

Note		Description
1	Processing of floating point arithmetic	In floating point arithmetic, they are all executed in binary floating point numbers. However, since the binary floating point number itself is an incomprehensible value, it can be easily monitored on a peripheral soft component after being converted into a decimal floating point number operation.

4.11.5 FN 119 - EBIN/Conversion from Binary to Decimal Floating Point Numbers

Outline

An instruction to convert a decimal floating point number in a soft component to a binary floating point number.



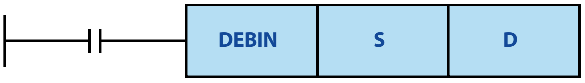
Conversion from Binary to Decimal Floating Point Numbers FN119 - EBCD	Instruction Mark	Execution Condition	Instruction Type	Instruction Step
	DEBIN	Continuous type	32 bit	9
	DEBINP	Pulse type	32 bit	9

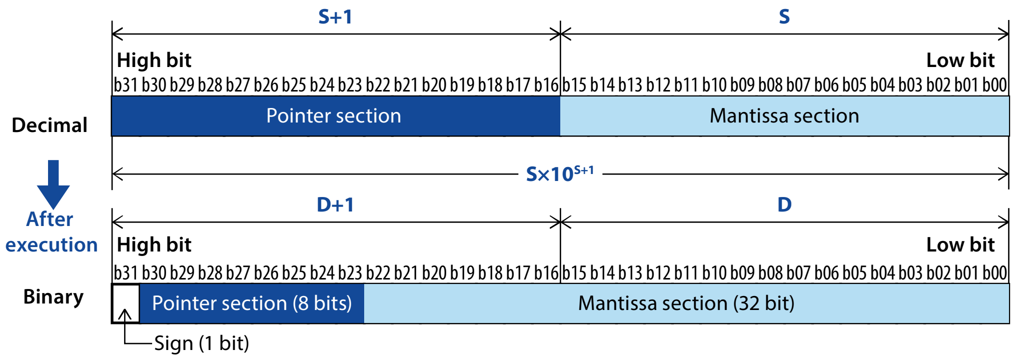
<table>
<tr><td rowspan="5">Operand</td><td colspan="14">Setting Data</td><td colspan="5">Data Type</td></tr>
<tr><td colspan="14">S: Data register number for saving decimal floating point data</td><td colspan="5">Real number (decimal)</td></tr>
<tr><td colspan="14">D: Save the data register number of the converted binary floating point data</td><td colspan="5">Real number (binary)</td></tr>
<tr><td colspan="19">Operand Object Soft Component</td></tr>
<tr><td colspan="7">Bit Soft Component</td><td colspan="8">Word Soft Component</td><td colspan="4">Others</td></tr>
<tr><td></td><td>X</td><td>Y</td><td>M</td><td>T</td><td>C</td><td>S</td><td>D.b</td><td>KnX</td><td>KnY</td><td>KnM</td><td>KnS</td><td>T</td><td>C</td><td>D</td><td>V, Z</td><td>H</td><td>K</td><td>E</td><td>P</td></tr>
<tr><td>S</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td>●</td><td>●</td><td></td><td></td><td></td><td></td></tr>
<tr><td>D</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td>●</td><td>●</td><td></td><td></td><td></td><td></td></tr>
</table>

Function and Action Description

32-bit Operation (DEBIN, DEBINP)

Convert the decimal floating point number of [S+1,S] to a binary floating point number and transfer it to [D+1,D].

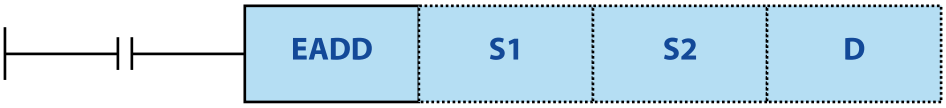



S+1
S
High bit
Low bit
b31 b30 b29 b28 b27 b26 b25 b24 b23 b22 b21 b20 b19 b18 b17 b16 b15 b14 b13 b12 b11 b10 b09 b08 b07 b06 b05 b04 b03 b02 b01 b00
Decimal
Pointer section
Mantissa section
S×10^S+1
After execution
D+1
D
High bit
Low bit
b31 b30 b29 b28 b27 b26 b25 b24 b23 b22 b21 b20 b19 b18 b17 b16 b15 b14 b13 b12 b11 b10 b09 b08 b07 b06 b05 b04 b03 b02 b01 b00
Binary
Pointer section (8 bits)
Mantissa section (32 bit)
Sign (1 bit)

4.11.6 FN 120 - EADD/Binary Floating Point Addition

Outline

Two binary floating point addition instructions.



Binary Floating Point Addition FN120 - EBCD	Instruction Mark	Execution Condition	Instruction Type	Number of Instruction Steps
	DEADD	Continuous type	32 bit	13
	DEADDP	Pulse type	32 bit	13

Operand	Setting Data	Data Type
	S1: Saving the word soft component number of binary floating point data that performs addition operation	Real number (binary)
	S2: Saving the word soft component number of binary floating point data that performs addition operation	Real number (binary)
	D: Saving the data register number of binary floating point data after the addition operation completed	Real number (binary)

Operand	Operand Object Soft Component																	
	Bit Soft Component							Word Soft Component								Others			
	X	Y	M	T	C	S	D.b	KnX	KnY	KnM	KnS	T	C	D	V, Z	H	K	E	P
S1														●	●	●	●	●	
S2														●	●	●	●	●	
D														●	●				

Function and Action Description

32-bit Operation (DEADD, DEADDP)
Add the binary floating point data of [S1+1,S1] and [S2+1,S2], and transfer the result of the operation to [D+1,D] in the form of binary floating point. When a constant (K, H) is specified in [S1+1,S1] and [S2+1,S2], the value is automatically converted to a binary floating point.

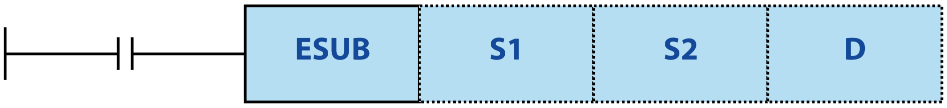
Note

Note		Description
1	When specifying the same soft component	The same soft component number can also be specified in [S1+1,S1] and [S2+1,S2] and [D+1,D]. At this time, if a continuous execution type instruction (DEADD) is used, the result of the addition operation will change every operation cycle, so please note.

4.11.7 FN 121 - ESUB/Binary Floating Point Subtraction

Outline

Two binary floating point subtraction instructions.



Binary Floating Point Subtraction FN121 - ESUB	Instruction Mark	Execution Condition	Instruction Type	Number of Instruction Steps
	DESUB	Continuous type	32 bit	13
	DESUBP	Pulse type	32 bit	13

Operand	Setting Data	Data Type
	S1: Saving the word soft component number of binary floating point data that performs subtraction operation	Real number (binary)
	S2: Saving the word soft component number of binary floating point data that performs subtraction operation	Real number (binary)
	D: Saving the binary floating point data after the subtraction operation completed	Real number (binary)

Operand	Operand Object Soft Component																		
	Bit Soft Component							Word Soft Component								Others			
	X	Y	M	T	C	S	D.b	KnX	KnY	KnM	KnS	T	C	D	V, Z	H	K	E	P
S1														●	●	●	●	●	
S2														●	●	●	●	●	
D														●	●				

Function and Action Description

32-bit Operation (DESUB, DESUBP)
Subtract the [S2+1,S2] binary floating point data from [S1+1,S1], and transfer the result of the operation to [D+1,D] in the form of binary floating point. When a constant (K, H) is specified in [S1+1,S1] and [S2+1,S2], the value is automatically converted to a binary floating point.

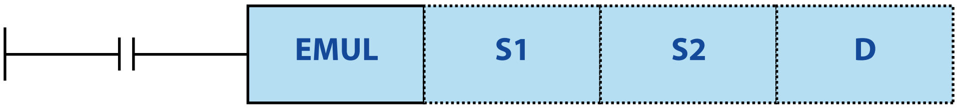
Note

Note		Description
1	When specifying the same soft component	The same soft component number can also be specified in [S1+1,S1] and [S2+1,S2] and [D+1,D]. At this time, if a continuous execution type instruction (DESUB) is used, the result of the subtraction operation will change every operation cycle, so please note.

4.11.8 FN 122 - EMUL/Binary Floating Point Multiplication

Outline

Two binary floating point multiplication instructions.



Binary Floating Point Multiplication FN122 - EMUL	Instruction Mark	Execution Condition	Instruction Type	Number of Instruction Steps
	DEMUL	Continuous type	32 bit	13
	DEMULP	Pulse type	32 bit	13

Operand	Setting Data	Data Type
	S1: Saving the word soft component number of binary floating point data that performs multiplication operation	Real number (binary)
	S2: Saving the word soft component number of binary floating point data that performs multiplication operation	Real number (binary)
	D: Saving the data register number of binary floating point data after the multiplication operation completed	Real number (binary)

Operand	Operand Object Soft Component																		
	Bit Soft Component							Word Soft Component								Others			
	X	Y	M	T	C	S	D.b	KnX	KnY	KnM	KnS	T	C	D	V, Z	H	K	E	P
S1														●	●	●	●	●	
S2														●	●	●	●	●	
D														●	●				

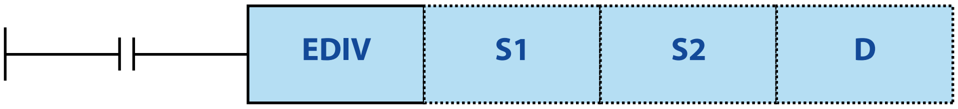
Function and Action Description

32-bit Operation (DEMUL, DEMULP)
Multiply the binary floating point data of [S1+1,S1] and [S2+1,S2], and transfer the result of the operation to [D+1,D] in form of binary floating point. When a constant (K, H) is specified in [S1+1,S1] and [S2+1,S2], the value is automatically converted to a binary floating point.

4.11.9 FN 123 - EDIV/Binary Floating Point Division

Outline

Two binary floating point division instructions.



Binary Floating Point Division FN123 - EDIV	Instruction Mark	Execution Condition	Instruction Type	Number of Instruction Steps
	DEDIV	Continuous type	32 bit	13
	DEDIVP	Pulse type	32 bit	13

Operand	Setting Data	Data Type
	S1: Saving the word soft component number of binary floating point data that performs division operation	Real number (binary)
	S2: Saving the word soft component number of binary floating point data that performs multiplication operation	Real number (binary)
	D: Saving the data register number of binary floating point data after the division operation completed	Real number (binary)

Operand	Operand Object Soft Component																		
	Bit Soft Component							Word Soft Component								Others			
	X	Y	M	T	C	S	D.b	KnX	KnY	KnM	KnS	T	C	D	V, Z	H	K	E	P
S1														●	●	●	●	●	
S2														●	●	●	●	●	
D														●	●				

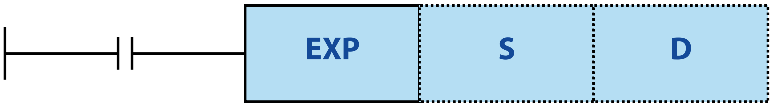
Function and Action Description

32-bit Operation (DEDIV, DEDIVP)
Divide the binary floating point data of [S1+1,S1] and [S2+1,S2], and transfer the result of the operation to [D+1,D] in the form of binary floating point. When a constant (K, H) is specified in [S1+1,S1] and [S2+1,S2], the value is automatically converted to a binary floating point.

4.11.10 FN 124 - EXP/Binary Floating Point Exponential Operation

Outline

This instruction is an exponential operation instruction based on e (2.71828).



Binary Floating Point Exponential Operation FN124 - EXP	Instruction Mark	Execution Condition	Instruction Type	Number of Instruction Steps
	DEXP	Continuous type	32 bit	9
	DEXPP	Pulse type	32 bit	9

<table>
<tr><td rowspan="6">Operand</td><td colspan="14">Setting Data</td><td colspan="5">Data Type</td></tr>
<tr><td colspan="14">S: Saving the soft component start number of the binary floating point data that performs exponential operation</td><td colspan="5">Real number (binary)</td></tr>
<tr><td colspan="14">D: Saving the soft component start number of the operation result</td><td colspan="5">Real number (binary)</td></tr>
<tr><td colspan="19">Operand Object Soft Component</td></tr>
<tr><td colspan="7">Bit Soft Component</td><td colspan="8">Word Soft Component</td><td colspan="4">Others</td></tr>
<tr><td>X</td><td>Y</td><td>M</td><td>T</td><td>C</td><td>S</td><td>D.b</td><td>KnX</td><td>KnY</td><td>KnM</td><td>KnS</td><td>T</td><td>C</td><td>D</td><td>V, Z</td><td>H</td><td>K</td><td>E</td><td>P</td></tr>
<tr><td>S</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td>●</td><td>●</td><td></td><td></td><td>●</td><td></td></tr>
<tr><td>D</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td>●</td><td>●</td><td></td><td></td><td></td><td></td></tr>
</table>

Function and Action Description

32-bit Operation (DEXP, DEXPP)
The operation is performed with [S+1,S] as the exponent, and the operation result is saved to [D+1,D]. In addition, can specify the real number directly in S.

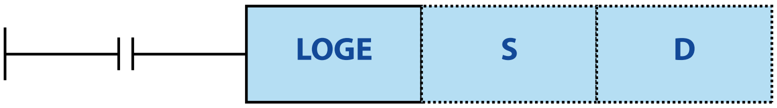
Error

Error
If the operation result is not in the range of 2^{-126} ≤ |operation result| < 2^{128}, an operation error will occur, the error flag bit M8067 is ON, and the error code (K6706) is stored in D8067.

4.11.11 FN 125 - LOGE/Binary Floating Point Natural Logarithm Operation

Outline

This instruction performs binary floating point natural logarithm operation.



Binary Floating Point Natural Logarithm Operation FN125 - LOGE	Instruction Mark	Execution Condition	Instruction Type	Number of Instruction Steps
	DLOGE	Continuous type	32 bit	9
	DLOGEP	Pulse type	32 bit	9

Operand	Setting Data														Data Type				
	S: Saving the soft component start number of the binary floating point data that performs natural logarithm operation														Real number (binary)				
	D: Saving the soft component start number of the operation result														Real number (binary)				
	Operand Object Soft Component																		
	Bit Soft Component							Word Soft Component								Others			
	X	Y	M	T	C	S	D.b	KnX	KnY	KnM	KnS	T	C	D	V, Z	H	K	E	P
S														●	●			●	
D														●	●				

Function and Action Description

32-bit Operation (DLOGE, DLOGEP)
The logarithm operation is performed with the natural logarithm of [S+1,S] as the base, and the operation result is saved to [D+1,D]. In addition, can specify the real number directly in S. • The value specified in [S+1,S] can only be set to a positive number (negative numbers cannot be calculated).

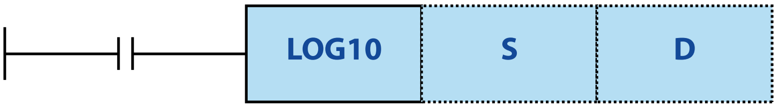
Error

Error
An operation error occurs when the value specified in S is negative or "0", the error flag bit M8067 is ON, and the error code (K6706) is stored in D8067.

4.11.12 FN 126 - LOG10/Binary Floating Point Common Logarithm Operation

Outline

This instruction performs common logarithm operation.



Binary Floating Point Common Logarithm Operation FN126 - LOG10	Instruction Mark	Execution Condition	Instruction Type	Number of Instruction Steps
	DLOG10	Continuous type	32 bit	16
	DLOG10P	Pulse type	32 bit	16

Operand	Setting Data														Data Type				
	S: Saving the soft component start number of the binary floating point data that performs common logarithm operation														Real number(binary)				
	D: Saving the soft component start number of the operation result														Real number (binary)				
	Operand Object Soft Component																		
	Bit Soft Component							Word Soft Component								Others			
	X	Y	M	T	C	S	D.b	KnX	KnY	KnM	KnS	T	C	D	V, Z	H	K	E	P
S														●	●			●	
D														●	●				

Function and Action Description

32-bit Operation (DLOG10, DLOG10P)
The common logarithm (10 is the base) operation is performed with [S+1,S], and the operation result is saved to [D+1,D]. In addition, can specify the real number directly in S.

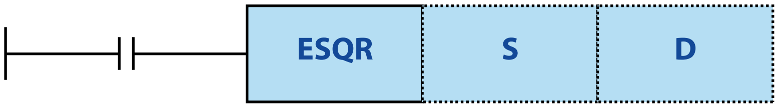
Error

Error
An operation error occurs when the value specified in S is negative or "0", the error flag bit M8067 is ON, and the error code (K6706) is stored in D8067.

4.11.13 FN 127 - ESQR/Binary Floating Point Square Root Operation

Outline

Binary floating point square root operation instructions.



Binary Floating Point Square Root Operation FN127 - ESQR	Instruction Mark	Execution Condition	Instruction Type	Number of Instruction Steps
	DESQR	Continuous type	32 bit	9
	DESQRP	Pulse type	32 bit	9

Operand	Setting Data	Data Type
	S: Saving the soft component start number of the binary floating point data that performs square root operation	Real number (binary)
	D: Saving the data register number of binary floating point data after the square root operation completed	Real number (binary)

Operand	Operand Object Soft Component																				
	Bit Soft Component							Word Soft Component										Others			
	X	Y	M	T	C	S	D.b	KnX	KnY	KnM	KnS	T	C	D	V, Z	H	K	E	P
S														●	●	●	●	●	
D														●	●				

Function and Action Description

32-bit Operation (DESQR, DESQRP)
After binary floating point square root operation is performed with [S+1,S], transfer the result to [D+1,D].

Related Soft Components

Soft Component	Name	Content
M8020	Zero	When the operation result is really 0, it is ON.

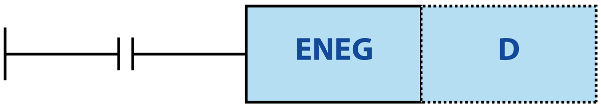
Error

Error
The content of [S1+1,S1] is valid only for positive numbers. If it is negative, the operation error (M8067) is activated and the instruction is not executed.

4.11.14 FN 128 - ENEG/Binary Floating Point Sign Flip

Outline

An instruction to flip the sign of binary floating point (real number) data.



Binary Floating Point Sign Flip FN128 - ENEG	Instruction Mark	Execution Condition	Instruction Type	Number of Instruction Steps
	DENEG	Continuous type	32 bit	5
	DENEGP	Pulse type	32 bit	5

Operand	Setting Data														Data Type				
	D: Saving the soft component start number of the binary floating point data that performs sign flip														Real number (binary)				
	Operand Object Soft Component																		
	Bit Soft Component							Word Soft Component								Others			
	X	Y	M	T	C	S	D.b	KnX	KnY	KnM	KnS	T	C	D	V, Z	H	K	E	P
D														●	●				

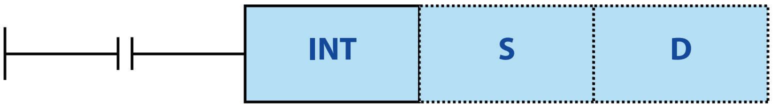
Function and Action Description

32-bit Operation (DENEG, DENEGP)
The sign flip of binary floating point data of [D+1,D] is stored in [D+1,D].

4.11.15 FN 129 - INT/Binary Floating Point→BIN Integer Conversion

Outline

An instruction to convert a binary floating point to a BIN integer.



Binary Floating Point→BIN Integer Conversion FN129 - INT	Instruction Mark	Execution Condition	Instruction Type	Number of Instruction Steps
	INT	Continuous type	16 bit	5
	INTP	Pulse type	16 bit	5
	DINT	Continuous type	32 bit	9
	DINTP	Pulse type	32 bit	9

Operand	Setting Data														Data Type				
	S: Saving the data register number of binary floating point data that will be converted to BIN integer														Real number (binary)				
	D: Saving the data register number of the converted BIN integer														16/32 bit				
	Operand Object Soft Component																		
	Bit Soft Component							Word Soft Component								Others			
	X	Y	M	T	C	S	D.b	KnX	KnY	KnM	KnS	T	C	D	V, Z	H	K	E	P
S														●	●				
D														●	●				

Function and Action Description

16-bit Operation (INT, INTP)	32-bit Operation (DESQR, DESQRP)
The binary floating point of [S+1,S] is converted to BIN integer and then transferred to D. • The inverse conversion action of the INT instruction is the instruction FLT (FN 49).	The binary floating point number of [S+1,S] is converted to BIN integer and then transferred to [D+1,D]. • The inverse conversion action of the DINT instruction is the instruction DFLT (FN 49).

Related Soft Components

Soft Component	Name	Content
M8020	Zero	When the operation result is really 0, it turns ON.
M8021	Borrow	When the borrowing conversion occurs, if it is discarded due to less than 1, it turns ON.
M8022	Carry	When the result of operation exceeds -32,768 ~ +32,767 (16 bit operation), or -2,147,483,648 ~ +2,147,483,647 (32 bit operation) and overflow occurs, it is ON (the operation result is not reflected).

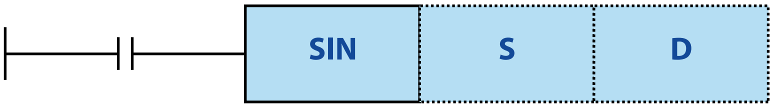
Note

Note		Description
1	Note when calculating	The value after the decimal point is discarded.

4.11.16 FN 130 - SIN/Binary Floating Point SIN Operation

Outline

An instruction to find the SIN value of an angle (RAD).



Binary Floating Point SIN Operation FN130 - SIN	Instruction Mark	Execution Condition	Instruction Type	Number of Instruction Steps
	DSIN	Continuous type	32 bit	9
	DSINP	Pulse type	32 bit	9

Operand	Setting Data														Data Type				
	S: Saving the soft component number of RAD (angle) of binary floating point														Real number (binary)				
	D: Saving the soft component number of SIN value of binary floating point														Real number (binary)				
	Operand Object Soft Component																		
	Bit Soft Component							Word Soft Component								Others			
	X	Y	M	T	C	S	D.b	KnX	KnY	KnM	KnS	T	C	D	V, Z	H	K	E	P
S														●	●			●	
D														●	●				

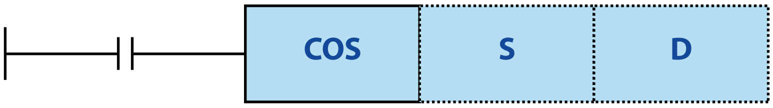
Function and Action Description

32-bit Operation (DSIN, DSINP)
Convert the angle value (binary floating point, radian) specified in [S+1,S] to the SIN value and transfer it to [D+1,D].

4.11.17 FN 131 - COS/Binary Floating Point COS Operation

Outline

An instruction to find the COS value of an angle (RAD).



Binary Floating Point COS Operation FN131 - COS	Instruction Mark	Execution Condition	Instruction Type	Number of Instruction Steps
	DCOS	Continuous type	32 bit	9
	DCOSP	Pulse type	32 bit	9

Operand	Setting Data														Data Type				
	S: Saving the soft component number of RAD (angle) of binary floating point														Real number (binary)				
	D: Saving the soft component number of COS value of binary floating point														Real number (binary)				
	Operand Object Soft Component																		
	Bit Soft Component							Word Soft Component								Others			
	X	Y	M	T	C	S	D.b	KnX	KnY	KnM	KnS	T	C	D	V, Z	H	K	E	P
S														●	●			●	
D														●	●				

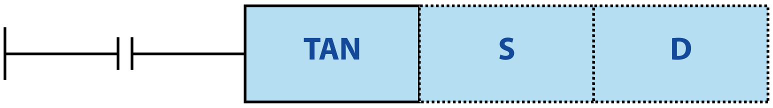
Function and Action Description

32-bit Operation (DCOS, DCOSP)
Convert the angle value (binary floating point, radian) specified in [S+1,S] to the COS value and transfer it to [D+1,D].

4.11.18 FN 132 - TAN/Binary Floating Point TAN Operation

Outline

An instruction to find the TAN value of an angle (RAD).



Binary Floating Point TAN Operation FN132 - TAN	Instruction Mark	Execution Condition	Instruction Type	Number of Instruction Steps
	DTAN	Continuous type	32 bit	9
	DTANP	Pulse type	32 bit	9

Operand	Setting Data															Data Type			
	S: Saving the soft component number of RAD (angle) of binary floating point															Real number (binary)			
	D: Saving the soft component number of TAN value of binary floating point															Real number (binary)			
	Operand Object Soft Component																		
	Bit Soft Component							Word Soft Component								Others			
	X	Y	M	T	C	S	D.b	KnX	KnY	KnM	KnS	T	C	D	V, Z	H	K	E	P
S														●	●			●	
D														●	●				

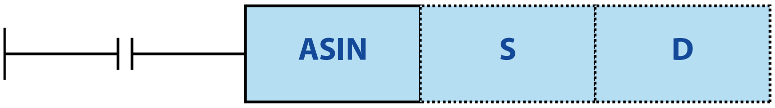
Function and Action Description

32-bit Operation (DTAN, DTANP)
Convert the angle value (binary floating point, radian) specified in [S+1,S] to the TAN value and transfer it to [D+1,D].

4.11.19 FN 133 - ASIN/Binary Floating Point SIN^{-1} Operation

Outline

This instruction performs SIN^{-1} operation.



Binary Floating Point SIN^{-1} Operation FN133 - DASIN	Instruction Mark	Execution Condition	Instruction Type	Number of Instruction Steps
	DASIN	Continuous type	32 bit	9
	DASINP	Pulse type	32 bit	9

Operand	Setting Data															Data Type			
	S: Saving the soft component start number of SIN value that performs SIN^{-1}(inverse SIN) operation															Real number (binary)			
	D: Saving the soft component start number of operation result															Real number (binary)			
	Operand Object Soft Component																		
	Bit Soft Component							Word Soft Component								Others			
	X	Y	M	T	C	S	D.b	KnX	KnY	KnM	KnS	T	C	D	V, Z	H	K	E	P
S														●	●			●	
D														●	●				

Function and Action Description

32-bit Operation (DASIN, DASINP)
The SIN value of [S+1,S] is used to find the angle, and the operation result is saved in [D+1,D]. In addition, you can specify the real number directly in S. • The SIN value of [S+1,S] can be set from -1.0 ~ +1.0. • The angle (operation result) saved in [D+1,D] is the value of the saved radians (-π/2) ~ (+π/2). For the conversion between radians and angles, please refer to the RAD (FN 136) command, DEG (FN 137) instruction, section 4.11.22 and 4.11.23 .

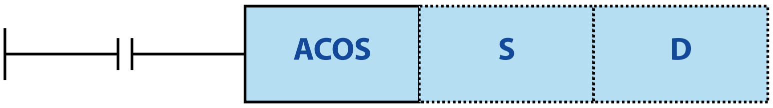
Error

Error
When the value specified in S is not in the range of -1.0 ~ +1.0, an operation error occurs, the error flag bit M8067 is ON, and the error code (K6706) is stored in D8067.

4.11.20 FN 134 - ACOS/Binary Floating Point COS^{-1} Operation

Outline

This instruction performs COS^{-1} operation.



Binary Floating Point COS^{-1} Operation FN134 - ACOS	Instruction Mark	Execution Condition	Instruction Type	Number of Instruction Steps
	DACOS	Continuous type	32 bit	9
	DACOSP	Pulse type	32 bit	9

<table>
<tr><td rowspan="6">Operand</td><td colspan="14">Setting Data</td><td colspan="6">Data Type</td></tr>
<tr><td colspan="14">S: Saving the soft component start number of COS value that performs COS^{-1} (inverse COS) operation</td><td colspan="6">Real number (binary)</td></tr>
<tr><td colspan="14">D: Saving the soft component start number of operation result</td><td colspan="6">Real number (binary)</td></tr>
<tr><td colspan="20">Operand Object Soft Component</td></tr>
<tr><td colspan="7">Bit Soft Component</td><td colspan="9">Word Soft Component</td><td colspan="4">Others</td></tr>
<tr><td>X</td><td>Y</td><td>M</td><td>T</td><td>C</td><td>S</td><td>D.b</td><td>KnX</td><td>KnY</td><td>KnM</td><td>KnS</td><td>T</td><td>C</td><td>D</td><td>V, Z</td><td>H</td><td>K</td><td>E</td><td>P</td><td></td></tr>
<tr><td>S</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td>●</td><td>●</td><td></td><td></td><td>●</td><td></td><td></td></tr>
<tr><td>D</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td>●</td><td>●</td><td></td><td></td><td></td><td></td><td></td></tr>
</table>

Function and Action Description

32-bit Operation (DACOS, DACOSP)
The COS value of [S+1,S] is used to find the angle, and the operation result is saved in [D+1,D]. In addition, you can specify the real number directly in S. • The COS value of [S+1,S] can be set from -1.0 ~ +1.0. • The angle (operation result) saved in [D+1,D] is the value of the saved radians (0 ~ π). For the conversion between radians and angles, please refer to the RAD (FN 136) command, DEG (FN 137) instruction, section 4.11.22 and 4.11.23.

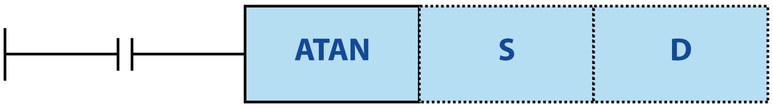
Error

Error
When the value specified in S is not in the range of -1.0 ~ +1.0, an operation error occurs, the error flag bit M8067 is ON, and the error code (K6706) is stored in D8067.

4.11.21 FN 135 - ATAN/Binary Floating Point TAN^{-1} Operation

Outline

This instruction performs TAN^{-1} operation.



Binary Floating Point TAN^{-1} Operation FN135 - ATAN	Instruction Mark	Execution Condition	Instruction Type	Number of Instruction Steps
	DATAN	Continuous type	32 bit	9
	DATANP	Pulse type	32 bit	9

Operand	Setting Data														Data Type				
	S: Saving the soft component start number of TAN value that performs TAN^{-1} (inverse TAN) operation														Real number (binary)				
	D: Saving the soft component start number of operation result														Real number (binary)				
	Operand Object Soft Component																		
	Bit Soft Component							Word Soft Component								Others			
	X	Y	M	T	C	S	D.b	KnX	KnY	KnM	KnS	T	C	D	V, Z	H	K	E	P
S														●	●			●	
D														●	●				

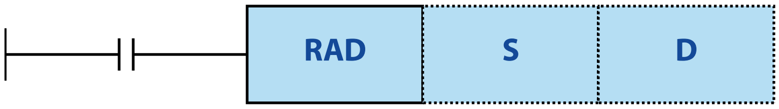
Function and Action Description

32-bit Operation (DATAN, DATANP)
The TAN value of [S+1,S] is used to find the angle, and the operation result is saved in [D+1,D]. In addition, you can specify the real number directly in S. • The angle (operation result) saved in [D+1,D] is the value of the saved radians (-π/2) ~ (+π/2). For the conversion between radians and angles, please refer to the RAD (FN 136) command, DEG (FN 137) instruction, section 4.11.22 and 4.11.23.

4.11.22 FN 136 - RAD/Binary Floating Point Angle→Radian Conversion

Outline

This is an instruction that converts the value of an angle unit into a radian unit.



Binary Floating Point Angle→ Radian Conversion FN136 - RAD	Instruction Mark	Execution Condition	Instruction Type	Number of Instruction Steps
	DRAD	Continuous type	32 bit	9
	DRADP	Pulse type	32 bit	9

Operand	Setting Data	Data Type
	S: Saving the soft component start number of angle that will be converted to radian	Real number (binary)
	D: Saving the soft component start number of operation result	Real number (binary)

Operand	Operand Object Soft Component																		
	Bit Soft Component							Word Soft Component								Others			
	X	Y	M	T	C	S	D.b	KnX	KnY	KnM	KnS	T	C	D	V, Z	H	K	E	P
S														●	●			●	
D														●	●				

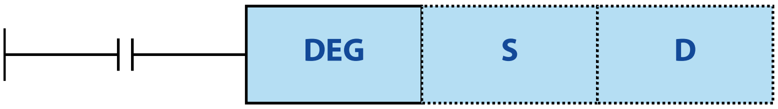
Function and Action Description

32-bit Operation (DRAD, DRADP)
The unit of [S+1,S] is converted from angle to radian and will be saved to [D+1,D]. In addition, the real number can be directly specified in S. • The conversion of the angle unit→radian unit is performed as follows: $\text{Radian unit} = \text{angle unit} \times \frac{\pi}{180}$

4.11.23 FN 137 - DEG/Binary Floating Point Radian→Angle Conversion

Outline

This is an instruction that converts the value of a radian unit into an angle unit.



Binary Floating Point Radian→ Angle Conversion FN137 - DEG	Instruction Mark	Execution Condition	Instruction Type	Number of Instruction Steps
	DDEG	Continuous type	32 bit	9
	DDEGP	Pulse type	32 bit	9

<table>
<tr><td rowspan="6">Operand</td><td colspan="14">Setting Data</td><td colspan="5">Data Type</td></tr>
<tr><td colspan="14">S: Saving the soft component start number of radian that will be converted to angle</td><td colspan="5">Real number (binary)</td></tr>
<tr><td colspan="14">D: Saving the soft component start number of the value that have converted to angle</td><td colspan="5">Real number (binary)</td></tr>
<tr><td colspan="19">Operand Object Soft Component</td></tr>
<tr><td colspan="7">Bit Soft Component</td><td colspan="8">Word Soft Component</td><td colspan="4">Others</td></tr>
<tr><td>X</td><td>Y</td><td>M</td><td>T</td><td>C</td><td>S</td><td>D.b</td><td>KnX</td><td>KnY</td><td>KnM</td><td>KnS</td><td>T</td><td>C</td><td>D</td><td>V, Z</td><td>H</td><td>K</td><td>E</td><td>P</td></tr>
<tr><td>S</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td>●</td><td>●</td><td></td><td></td><td>●</td><td></td></tr>
<tr><td>D</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td>●</td><td>●</td><td></td><td></td><td></td><td></td></tr>
</table>

Function and Action Description

32-bit Operation (DDEG, DDEGP)

The unit of [S+1,S] is converted from radian to angle and will be saved to [D+1,D].

- The conversion of the angle unit→radian unit is performed as follows:

$$\text{Angle unit} = \text{radian unit} \times \frac{180}{\pi}$$

4.12 Data Processing 2 - FN 140 ~ FN 149

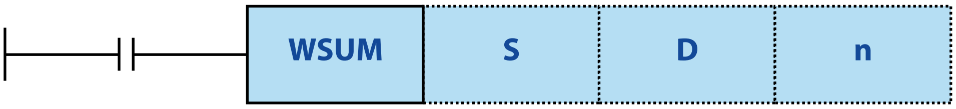
FN No.	Instruction Mark	Instruction Format	Function	Section	Page
140	WSUM	WSUM (S) (D) (n) WSUMP (S) (D) (n) DWSUM (S) (D) (n) DWSUMP (S) (D) (n)	Calculate the total value of the data	4.12.1	166
141	WTOB	WTOB (S) (D) (n) WTOBP (S) (D) (n)	Byte unit data separation	4.12.2	167
142	BTOW	BTOW (S) (D) (n) BTOWP (S) (D) (n)	Byte unit data combination	4.12.3	169
143	UNI	UNI (S) (D) (n) UNIP (S) (D) (n)	4-bit combination of 16-bit data	4.12.4	171
144	DIS	DIS (S) (D) (n) DISP (S) (D) (n)	4-bit separation of 16-bit data	4.12.4	171
147	SWAP	SWAP (S) SWAPP (S) DSWAP (S) DSWAPP (S)	High and low byte swap	4.12.6	173
149	SORT2	SORT2 (S) (m1) (m2) (D) (n) DSORT2 (S) (m1) (m2) (D) (n)	Data sorting 2	4.12.7	174

4.12.1 FN 140 - WSUM/Calculate the Total Value of Data

Outline

This instruction can calculate the total value of consecutive 16-bit or 32-bit data.

When calculating the addition data (total value) in bytes (8 bits), please use the CCD (FN 84) instruction.



	Instruction Mark	Execution Condition	Instruction Type	Number of Instruction Steps
Calculate the Total Value of Data FN140 - WSUM	WSUM	Continuous type	16 bit	7
	WSUMP	Pulse type	16 bit	7
	DWSUM	Continuous type	32 bit	13
	DWSUMP	Pulse type	32 bit	13

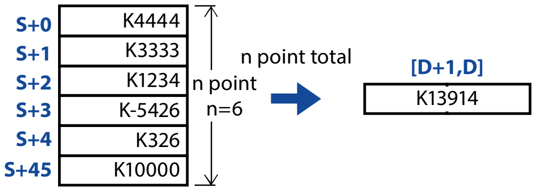
Operand	Setting Data	Data Type
	S: Saving the soft component start number of the data for which the total value is to be calculated	16/32 bit
	D: Saving the soft component start number of the total value	32/64 bit
	n: Number of data (0 < n)	16/32 bit

Operand Object Soft Component	Bit Soft Component							Word Soft Component									Others			
	X	Y	M	T	C	S	D.b	KnX	KnY	KnM	KnS	T	C	D	V, Z	H	K	E	P
S												●	●	●	●				
D												●	●	●	●				
n														●		●	●		

Function and Action Description

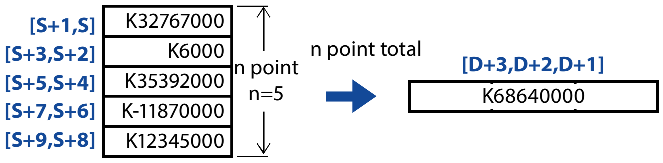
16-bit Operation (WSUM, WSUMP)

The total value of the n-point 16-bit data starting from S is stored in [D+1,D] as 32-bit data.



32-bit Operation (DWSUM, DWSUMP)

The total value of the n-point 32-bit data starting from [S+1,S] is stored in [D+3,D+2,D+1,D] as 64-bit data.



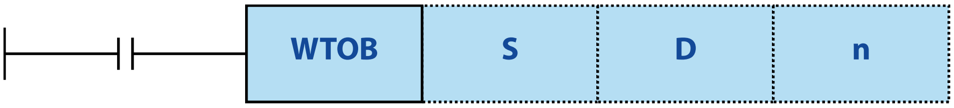
Error

Error	
1	Operation errors may occur in the following cases, the error flag bit M8067 is ON, and the error code (K6706) is saved in D8067. • the n-point soft component starting with S is beyond the range of the specified soft component. • n ≤ 0. • D is beyond the range of soft components.

4.12.2 FN 141 - WTOB/Byte Unit Data Separation

Outline

This instruction can separate consecutive 16-bit data in byte (8-bit) units.



Byte Unit Data Separation FN141 - WTOB	Instruction Mark	Execution Condition	Instruction Type	Number of Instruction Steps
	WTOB	Continuous type	16 bit	7
	WTOBP	Pulse type	16 bit	7

Operand	Setting Data	Data Type
	S: Saving the soft component start number of the data that is to be separated in byte units	16 bit
	D: Saving the soft component start number of the result that has been separated in byte units	16 bit
	n: The number of byte data that is to be separated (0 ≤ n)	16 bit

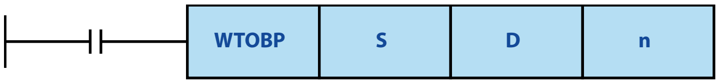
Operand	Operand Object Soft Component																		
	Bit Soft Component							Word Soft Component								Others			
	X	Y	M	T	C	S	D.b	KnX	KnY	KnM	KnS	T	C	D	V, Z	H	K	E	P
S												●	●	●	●				
D												●	●	●	●				
n														●		●	●		

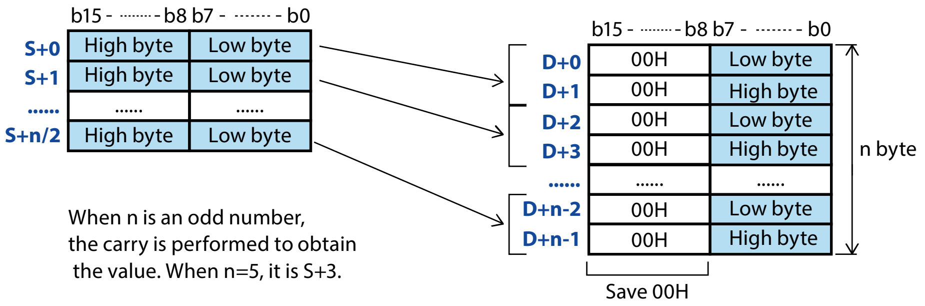
Function and Action Description

16-bit Operation (WTOB, WTOBP)

The 16-bit data stored in n/2 soft components starting from S is separated into n bytes, and stored in the n soft components starting with D as follows.

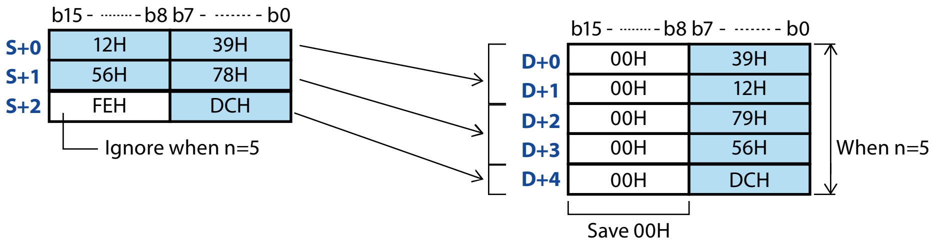
After storing the high byte (8 bits) of the soft component (after D) of the separated byte data, 00H is saved.





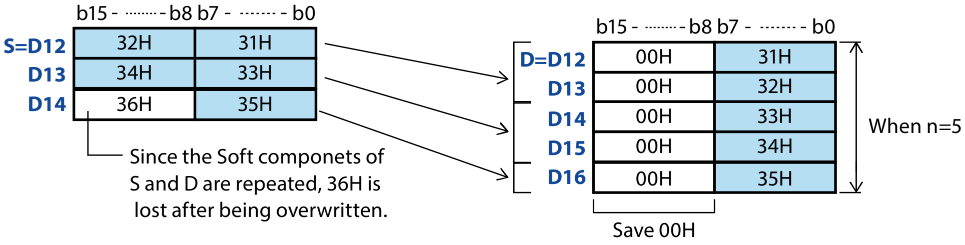
When n is an odd number, as shown in the figure below, in the final data of the separated source, only the low byte (8 bits) is the object data.

For example, when n = 5, the data of the low byte (8 bits) of S ~ (S+2) is stored in D ~ (D+4).



- When n = 0, the instruction is not executed.

Note

Note	
1	Soft componets that store separate source data and separated data can be reused. However, please note that when n is an odd number, as shown in the following example, the high byte (8 bits) data of final data before separation may be lost after being overwritten.  b15 - ········ - b8 b7 - ······· - b0 S=D12 D13 D14 32H 31H 34H 33H 36H 35H Since the Soft componets of S and D are repeated, 36H is lost after being overwritten. b15 - ········ - b8 b7 - ······· - b0 D=D12 D13 D14 D15 D16 00H 31H 00H 32H 00H 33H 00H 34H 00H 35H When n=5 Save 00H

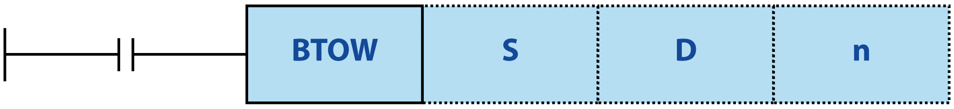
Error

Error	
1	Operation errors may occur in the following cases, the error flag bit M8067 is ON, and the error code (K6706) is saved in D8067. • When S ~ (S+n/2) of the separate source soft component is beyond the range of the specified soft component. When n is an odd number, it is necessary to occupy the soft component of the single digit of the value after the carry. • When the saved soft component D ~ (D+n-1) of the separated data is beyond the range of the specified soft component.

4.12.3 FN 142 - BTOW/Byte Unit Data Combination

Outline

This instruction can combine the low 8 bits (lower byte) of consecutive 16-bit data.



Byte Unit Data Combination FN142 - BTOW	Instruction Mark	Execution Condition	Instruction Type	Number of Instruction Steps
	BTOW	Continuous type	16 bit	7
	BTOWP	Pulse type	16 bit	7

Operand	Setting Data	Data Type
	S: Saving the soft component start number of the data that is to be combined in byte units	16 bit
	D: Saving the soft component start number of the result that has been combined in byte units	16 bit
	n: The number of byte data that is to be combined (0 ≤ n)	16 bit

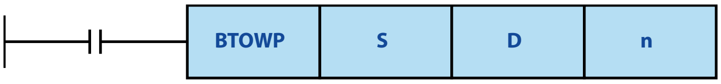
Operand Object Soft Component	Bit Soft Component							Word Soft Component								Others			
	X	Y	M	T	C	S	D.b	KnX	KnY	KnM	KnS	T	C	D	V, Z	H	K	E	P
S												●	●	●	●				
D												●	●	●	●				
n														●		●	●		

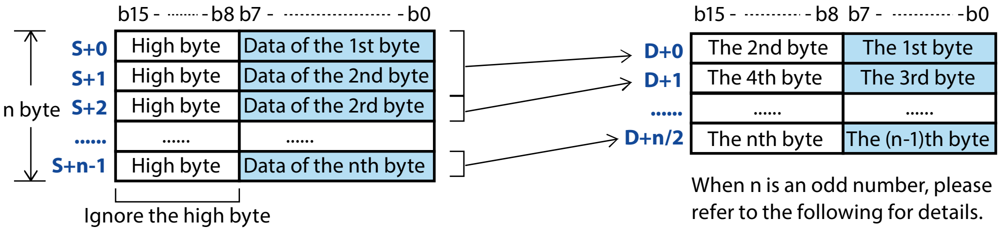
Function and Action Description

16-bit Operation (BTOW, BTOWP)

The 16-bit data after combining the low byte (8 bits) of the n-point 16-bit data starting from S is stored in the n/2 soft components starting with D as shown below.

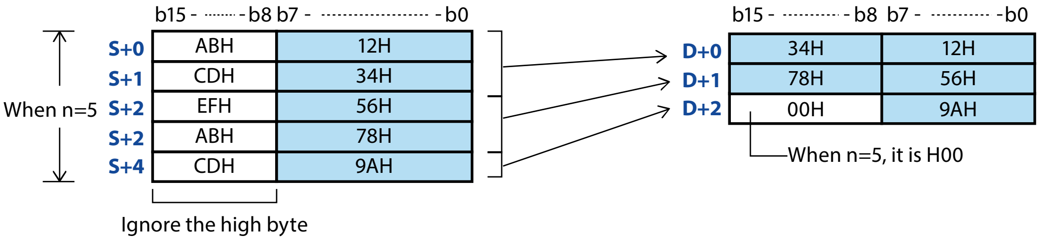
The high byte (8 bits) of the 16-bit data (after S) of combining source is ignored.





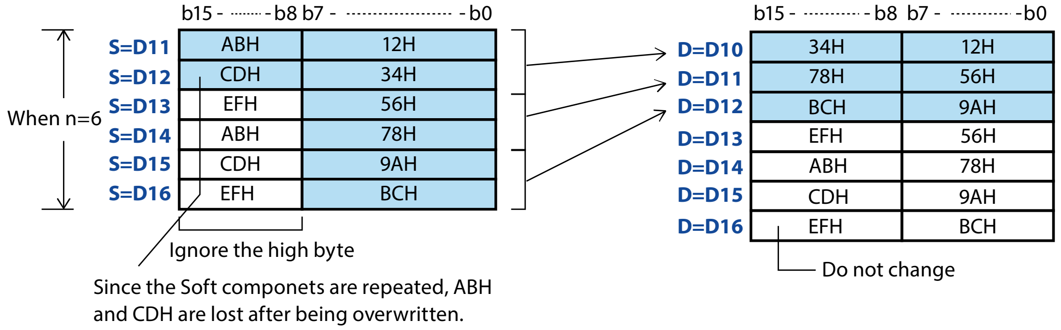
When n is an odd number, as shown in the figure below, the high byte (8 bits) of the finally combined data is 00H.

For example, when n = 5, the data of the low byte (8 bits) of S ~ (S+4) is stored in D ~ (D+2). The high byte (8 bits) of D+2 is 00H.



- When n = 0, the instruction is not executed.

Note

<table>
<tr><th colspan="2">Note</th></tr>
<tr><td>1</td><td>
Soft componets that store combining source data and combined data can be reused.
However, please note that the high byte (8 bits) of the combined source data stored in the reusable soft components, as shown in the following example, the data of the high byte (8 bits) will be lost after being overwritten by the combined data.


</td></tr>
</table>

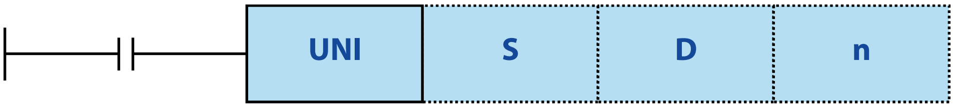
Error

Error	
1	Operation errors may occur in the following cases, the error flag bit M8067 is ON, and the error code (K6706) is saved in D8067. • When the soft component specified in S ~ (S+n-1) of the combining source is beyond the range of this soft component . • When the saved soft component D ~ (D+n/2) of the combined data is beyond the range of the specified soft component. When n is an odd number, it is necessary to occupy the soft component of the single digit of the value after the carry.

4.12.4 FN 143 - UNI/4-bit Combination of 16-bit Data

Outline

This instruction can combine the low 4 bits of consecutive 16-bit data.



4-bit Combination of 16-bit Data FN143 - UNI	Instruction Mark	Execution Condition	Instruction Type	Number of Instruction Steps
	UNI	Continuous type	16 bit	7
	UNIP	Pulse type	16 bit	7

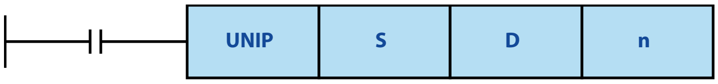
Operand	Setting Data	Data Type
	S: Saving the soft component start number of the data that is to be combined	16 bit
	D: Saving the soft component number of the data that has been combined	16 bit
	n: Combining number (0 ~ 4, do not process when n = 0)	16 bit

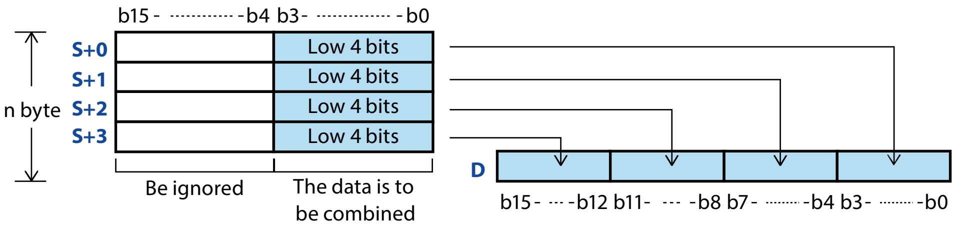
Operand	Operand Object Soft Component																		
	Bit Soft Component							Word Soft Component								Others			
	X	Y	M	T	C	S	D.b	KnX	KnY	KnM	KnS	T	C	D	V, Z	H	K	E	P
S												●	●	●	●				
D												●	●	●	●				
n														●		●	●		

Function and Action Description

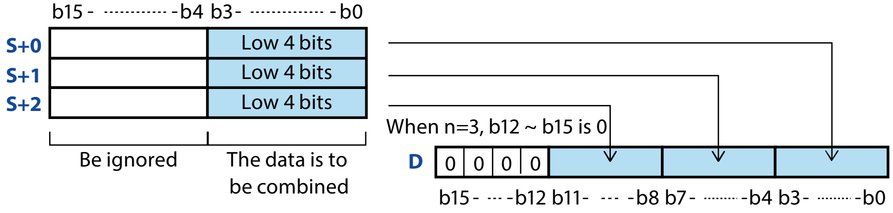
16-bit Operation (UNI, UNIP)

The 16-bit data after combining the low 4 bits of the n-point 16-bit data starting from S is stored in D as shown below.




b15 - ············· -b4 b3- ················ - b0
S+0
S+1
S+2
S+3
n byte
Low 4 bits
Low 4 bits
Low 4 bits
Low 4 bits
Be ignored
The data is to be combined
D
b15 - ···· -b12 b11- ··· -b8 b7 - ········· -b4 b3 - ········· -b0

- Specify 1 ~ 4 in n, when n = 0, the instrution is not be executed.
- When 1 ≤ n ≤ 3, the single digit of high bit {4 × (4 - n)} of D is 0.
  For example, when n = 3, the low 4 bits of S ~ (S+2) are saved to b0 ~ b11 of D, and the high 4 bits of D become 0.


b15 - ············· -b4 b3- ················ - b0
S+0
S+1
S+2
Low 4 bits
Low 4 bits
Low 4 bits
Be ignored
The data is to be combined
When n=3, b12 ~ b15 is 0
D
0
0
0
0
b15 - ···· -b12 b11- ··· -b8 b7 - ········· -b4 b3 - ········· -b0

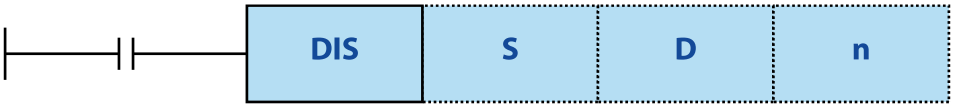
Error

Error	
1	Operation errors may occur in the following cases, the error flag bit M8067 is ON, and the error code (K6706) is saved in D8067. • The soft component specified in S ~ (S+n) is beyond the range of this soft component. • N specifies numbers other than 0 ~ 4.

4.12.5 FN 144 - DIS/4-bit Seperation of 16-bit Data

Outline

An instruction that separates 16-bit data in units of 4 bits.



4-bit Seperation of 16-bit Data FN144 - DIS	Instruction Mark	Execution Condition	Instruction Type	Number of Instruction Steps
	DIS	Continuous type	16 bit	7
	DISP	Pulse type	16 bit	7

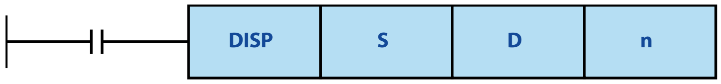
Operand	Setting Data	Data Type
	S: Saving the soft component start number of the data that is to be seperated	16 bit
	D: Saving the soft component number of the data that has been seperated	16 bit
	n: Seperating number (0 ~ 4, do not process when n = 0)	16 bit

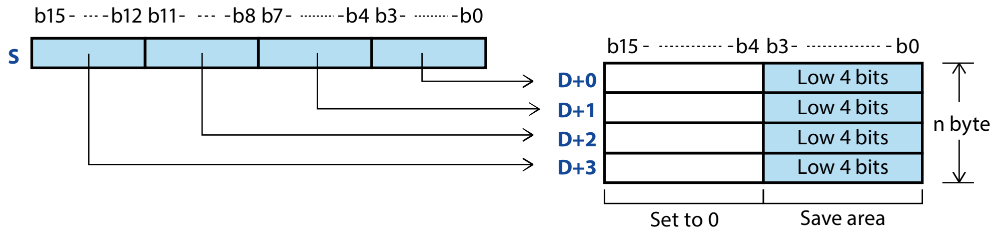
Operand	Operand Object Soft Component																		
	Bit Soft Component							Word Soft Component								Others			
	X	Y	M	T	C	S	D.b	KnX	KnY	KnM	KnS	T	C	D	V, Z	H	K	E	P
S												●	●	●	●				
D												●	●	●	●				
n														●		●	●		

Function and Action Description

16-bit Operation (DIS, DISP)

After separating the 16-bit data of S in units of 4 bits, it is stored in D as shown below.





- Specify 1 ~ 4 in n, the instruction is not executed when n = 0.
- The high 12 bits of the n-point soft component starting form D are set to 0.

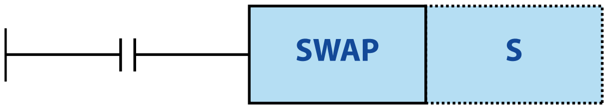
Error

Error	
1	Operation errors may occur in the following cases, the error flag bit M8067 is ON, and the error code (K6706) is saved in D8067. • The n-point soft component starting form D is beyond the range of the specified soft component. • N specifies numbers other than 0 ~ 4.

4.12.6 FN 147 - SWAP/High and Low Byte Swap

Outline

An instruction that swaps the high 8 bits and low 8 bits of the word data.



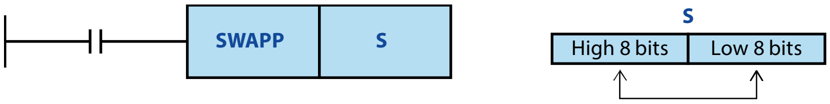
High and Low Byte Swap FN147 - SWAP	Instruction Mark	Execution Condition	Instruction Type	Number of Instruction Steps
	SWAP	Continuous type	16 bit	3
	SWAPP	Pulse type	16 bit	3
	DSWAP	Continuous type	32 bit	5
	DSWAPP	Pulse type	32 bit	5

Operand	Setting Data														Data Type				
	S: Soft component of high and low byte swap														16/32 bit				
	Operand Object Soft Component																		
	Bit Soft Component							Word Soft Component								Others			
	X	Y	M	T	C	S	D.b	KnX	KnY	KnM	KnS	T	C	D	V, Z	H	K	E	P
S									●	●	●	●	●	●	●				

Function and Action Description

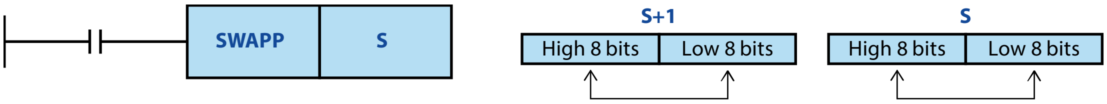
16-bit Operation (SWAP, SWAPP)

Perform the low 8 bits and high 8 bits swap.



32-bit Operation (DSWAP, DSWAPP)

Perform the low 8 bits and high 8 bits swap respectively.



Note

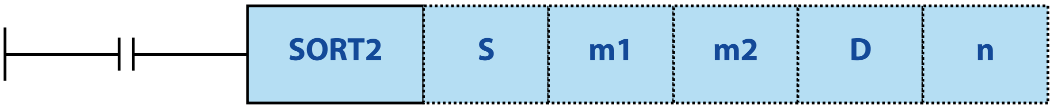
Note	
1	When using continuous type instructions, please note that the swap will be performed in each operation cycle. • Same as the extended function of the XCH (FN 17) instruction.

4.12.7 FN 149 - SORT2/Data Sorting 2

Outline

An instruction for ascending/descending reordering of data tables consisting of data (row) and group data (column) based on the specified group data (column) and in unit of row. In this instruction, data (row) are easily added because it (row direction) is stored in continuous soft components.

In addition, there are SORT (FN 69) instructions that support only ascending order and different data structures (data is composed of continuous soft components in column direction).



Data Sorting 2 FN149 - SORT2	Instruction Mark	Execution Condition	Instruction Type	Number of Instruction Steps
	SORT2	Pulse type	16 bit	11
	DSORT2	Pulse type	32 bit	21

Operand	Setting Data	Data Type
	S: Saving the soft component start number of the data table [occupied m1 × m2 point]	16/32 bit
	m1: Data number (rows) [1 ~ 32]	16/32 bit
	m2: Group data number (column) [1 ~ 6]	16/32 bit
	D: Saving the soft component start number of the operation result [occupied m1 × m2 point]	16/32 bit
	n: Column number of the group data (column) as the sorting criterion [1 ~ m2]	16/32 bit

Operand Object Soft Component	Bit Soft Component							Word Soft Component								Others			
	X	Y	M	T	C	S	D.b	KnX	KnY	KnM	KnS	T	C	D	V, Z	H	K	E	P
S														●					
m1														●		●	●		
m2																●	●		
D														●					
n														●		●	●		

Function and Action Description

16-bit Operation (SORT2)

For the data table of the (m1 × m2) point starting from S (before sorting), the data rows are sorted in ascending or descending order based on the group data of n columns, and then saved to the data table (after sorting) of the (m1 × m2) point starting from D.

The example "m1 = K3, m2 = K4" before sorting inthe following table shows the structure of the data table. In the sorted data table, please rewrite S to D.

		m2 Group Data (when m2 = K4) [Column Number]			
		1: Management Number	2: Height	3: Weight	4: Age
Data Number (when m1 = K3) [Row Number]	1	S	S+1	S+2	S+3
	2	S+4	S+5	S+6	S+7
	3	S+8	S+9	S+10	S+11

- Set the sort by the ON/OFF status of the M8165.

	Set the Order of Sorting
M8165 = ON	Descending
M8165 = OFF	Ascending

- SORT2 is a pulse type instruction, the first cycle of the instruction is turned on to sort the data, and then no longer executed until the next time it is disconnected and then turned on.

32-bit Operation (DSORT2)

For the data table of the (m1 × m2) point starting from [S+1,S] (before sorting), the data rows are sorted in ascending or descending order based on the group data of n columns, and then saved to the data table (after sorting) of the (m1 × m2) point starting from [D+1,D].

The example ″m1 = K3, m2 = K4″ before sorting inthe following table shows the structure of the data table. In the sorted data table, please rewrite S to D.

		m2 Group Data (when m2 = K4) [Column Number]			
		1: Management Number	2: Height	3: Weight	4: Age
Data Number (when m1 = K3) [Row Number]	1	[S+1,S]	[S+3, S+2]	[S+5, S+4]	[S+7, S+6]
	2	[S+9, S+8]	[S+11, S+10]	[S+13, S+12]	[S+15, S+14]
	3	[S+17, S+16]	[S+19, S+18]	[S+21, S+20]	[S+23, S+22]

- Set the sort by the ON/OFF status of the M8165.

	Set the Order of Sorting
M8165 = ON	Descending
M8165 = OFF	Ascending

- When using data register D or extension register R in m1, it is 32-bit data.
  For example, when m1 is specified in D0, m1 is 32-bit data of [D1, D0].
- SORT2 is a pulse type instruction, the first cycle of the instruction is turned on to sort the data, and then no longer executed until the next time it is disconnected and then turned on.

Related Soft Components

Soft Component	Name	Content
M8165	Descending order	When M8165 = ON, sort in descending order. When M8165 = OFF, sort in ascending order.

Note

Note	
1	SORT is a pulse type instruction. It is only executed once after turned on. When it is executed again, please enter ″OFF″ once in the instruction.

4.13 Positioning Control - FN 150 ~ FN 159

In FN 150 ~ FN 159, instructions for positioning control using the pulse output function built into the intelligent controller are provided.

FN No.	Instruction Mark	Instruction Format	Function	Section	Page
57	PLSY	PLSY (S1) (S2) (D) DPLSY (S1) (S2) (D)	Pulse output	4.13.2	179
157	PLSV	PLSV (S1) (D2) (D2) DPLSV (S1) (D2) (D2)	Variable speed pulse output	4.13.3	180
150	DSZR	DSZR (S1) (S2) (D1) (D2)	Return to origin with DOG search	4.13.4	182
156	ZRN	ZRN (S1) (S2) (S3) (D) DZRN (S1) (S2) (S3) (D)	Return to origin	4.13.5	187
151	DVIT	DVIT (S1) (S2) (D1) (D2) DDVIT (S1) (S2) (D1) (D2)	Interrupt positioning	4.13.6	190
158	DRVI	DRVI (S1) (S2) (D1) (D2) DDRVI (S1) (S2) (D1) (D2)	Relative positioning	4.13.7	193
159	DRVA	DRVA (S1) (S2) (D1) (D2) DDRVA (S1) (S2) (D1) (D2)	Absolute positioning	4.13.8	193

4.13.1 Related Soft Component

Special Auxiliary Relay

Y001, Y002, Y003, Y004 are pulse output soft components.

No.	Soft Component Number				Name	Attribute	Instruction
	Y000	Y001	Y002	Y003			
(1)	M8029				Instruction execution end flag bit	Read only	PLSY, PLSV, DSZR, ZRN, DVIT, DRVI, DRVA
(2)	M8329				Instruction execution abnormal end flag	Read only	PLSY, PLSV, DSZR, ZRN, DVIT, DRVI, DRVA
(3)	M8338				Acc. and Dec. action*	Readable and writable	PLSV, DSZR, ZRN, DVIT, DRVI, DRVA
(4)	M8336				The interrupt input specified function is valid*	Readable and writable	DVIT
(5)	M8340	M8350	M8360	M8370	Pulse output monitoring (BUSY/READY)	Read only	PLSY, PLSV, DSZR, ZRN, DVIT, DRVI, DRVA
(6)	M8341	M8351	M8361	M8371	Clear signal output function is valid*	Readable and writable	DSZR, ZRN
(7)	M8342	M8352	M8362	M8372	Origin return direction designation*	Readable and writable	DSZR
(8)	M8343	M8353	M8363	M8373	Forward limit	Readable and writable	PLSV, DSZR, DVIT, DRVI, DRVA
(9)	M8344	M8354	M8364	M8374	Reverse limit	Readable and writable	PLSV, DSZR, DVIT, DRVI, DRVA
(10)	M8345	M8355	M8365	M8375	Near-point signal logic inversion*	Readable and writable	DSZR
(11)	M8346	M8356	M8366	M8376	Origin signal logic inversion*	Readable and writable	DSZR
(12)	M8347	M8357	M8367	M8377	Interrupt signal logic inversion*	Readable and writable	DVIT
(13)	M8348	M8358	M8368	M8378	Positioning instruction driving	Read only	PLSY, DVIT, DRVI, DRVA
(14)	M8349	M8359	M8369	M8379	Pulse stop instruction*	Readable and writable	PLSY, PLSV, DSZR, ZRN, DVIT, DRVI, DRVA
(15)	M8460	M8461	M8462	M8463	User interrupt input instruction	Readable and writable	DVIT
(16)	M8464	M8465	M8466	M8467	The clear signal device designation function is valid	Readable and writable	DSZR, ZRN
*: Clear when RUN→STOP.							

Special Data Relay

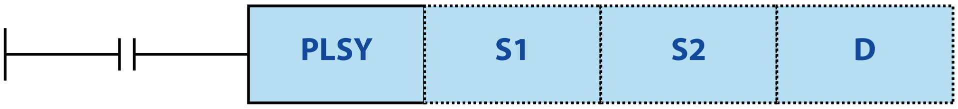
Y001, Y002, Y003, Y004 are pulse output soft components.

No.	Soft Component Number								Name	Data Length	Initial Value	Instruction
	Y000		Y001		Y002		Y003					
(1)	D8336								Interrupt input designation	16 bit	-	DVIT
(2)	D8340	Low	D8350	Low	D8360	Low	D8370	Low	Current value register [PLS]	32 bit	0	PLSY, PLSV, DSZR, ZRN, DVIT, DRVI, DRVA
	D8341	High	D8351	High	D8361	High	D8371	High				
(3)	D8342		D8352		D8362		D8372		Base speed [Hz]	16 bit	0	PLSV, DSZR, ZRN, DVIT, DRVI, DRVA
(4)	D8343	Low	D8353	Low	D8363	Low	D8373	Low	Max. speed [Hz]	32 bit	100,000	PLSV, DSZR, ZRN, DVIT, DRVI, DRVA
	D8344	High	D8354	High	D8364	High	D8374	High				
(5)	D8345		D8355		D8365		D8375		Crawling speed [Hz]	16 bit	1000	DSZR
(6)	D8346	Low	D8356	Low	D8366	Low	D8376	Low	Origin return speed [Hz]	32 bit	50,000	DSZR
	D8347	High	D8357	High	D8367	High	D8377	High				
(7)	D8348		D8358		D8368		D8378		Acc. time [ms]	16 bit	200	PLSV, DSZR, ZRN, DVIT, DRVI, DRVA
(8)	D8349		D8359		D8369		D8379		Dec. time [ms]	16 bit	200	PLSV, DSZR, ZRN, DVIT, DRVI, DRVA
(9)	D8464		D8465		D8466		D8467		Clear signal device designation	16 bit	-	DSZR, ZRN

4.13.2 FN 57 - PLSY/Pulse Output

Outline

An instruction sends out a pulse signal.



Pulse Output FN57 - PLSY	Instruction Mark	Execution Condition	Instruction Type	Instruction Step
	PLSY	Continuous type	16 bit	7
	DPLSY	Continuous type	32 bit	13

Operand	Setting Data	Data Type
	S1: Frequency data (Hz) or word soft component number for saving data	16/32 bit
	S2: Pulse amount data or word soft component number for saving data	16/32 bit
	D: Bit soft component for output pulse (Y) No.	Bit

Operand	Operand Object Soft Component																		
	Bit Soft Component							Word Soft Component								Others			
	X	Y	M	T	C	S	D.b	KnX	KnY	KnM	KnS	T	C	D	V, Z	H	K	E	P
S1								●	●	●	●	●	●	●	●	●	●		
S2								●	●	●	●	●	●	●	●	●	●		
D		●													●				

Function and Action Description

16-bit Operation (PLSY)	32-bit Operation (DPLSY)
Specify the frequency in S1. Setting range: 0 ~ 32,767Hz. Specify the amount of pulses to be sent in S2. Setting range: 0 ~ 32,767 (PLS). • 0 means that the number of transmitted pulses is not limited, and the pulse is sent until the condition is disconnected. Specify the Y number of the high-speed pulse output in D.	Specify the frequency in [S1+1,S1]. Setting range: 0 ~ 100,000Hz. Specify the amount of pulses to be sent in [S2+1,S2]. Setting range: 0 ~ 2,147,483,647 (PLS). • 0 means that the number of transmitted pulses is not limited, and the pulse is sent until the condition is disconnected. Specify the Y number of the high-speed pulse output in D.

Related Soft Component

Please refer to 4.13.1.

Type	Related Soft Component
Special auxiliary relay	(1), (2), (5), (13), (14)
Special data relay	(2)

Note

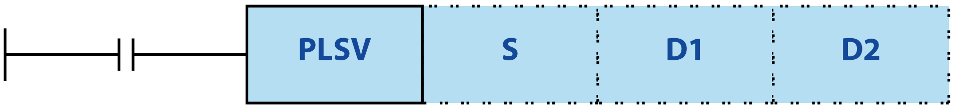
Note	
1	The same high-speed pulse output terminal, can not perform multiple pulse output functions at the same time.
2	During the execution of the instruction, directly modify the value of the operand, the result is different: • Modify the value of operand [S], the modified content will take effect immediately. • Modify the value of operand [S2], the modified content will be effective when the next drive instruction.
3	PLSY is a non-acceleration/deceleration pulse output command, which does not involve the following special data registers: • Acc. time. • Dec. time. • Base speed. • Max. speed.

4.13.3 FN 157 - PLSV/Variable Speed Pulse Output

Outline

This instruction is a variable speed pulse output instruction with a rotary direction output.

There are Acc./Dec. action and no Acc./Dec. action.



Variable Speed Pulse Output FN157 - PLSV	Instruction Mark	Execution Condition	Instruction Type	Number of Instruction Steps
	PLSV	Continuous type	16 bit	9
	DPLSV	Continuous type	32 bit	17

<table>
<tr><td rowspan="7">Operand</td><td colspan="14">Setting Data</td><td colspan="4">Data Type</td></tr>
<tr><td colspan="14">S1: Specify the soft component number of the output pulse frequency
The setting range is:
• 16-bit operation: -32,768 ~ +32,767 (Hz)
• 32-bit operation: -100,000 ~ +100,000 (Hz)</td><td colspan="4">16/32 bit</td></tr>
<tr><td colspan="14">D1: Specify the output number of the output pulse</td><td colspan="4">Bit</td></tr>
<tr><td colspan="14">D2: Specify the output number of the rotary direction signal</td><td colspan="4">Bit</td></tr>
<tr><td colspan="18">Operand Object Soft Component</td></tr>
<tr><td colspan="7">Bit Soft Component</td><td colspan="8">Word Soft Component</td><td colspan="4">Others</td></tr>
<tr><td>X</td><td>Y</td><td>M</td><td>T</td><td>C</td><td>S</td><td>D.b</td><td>KnX</td><td>KnY</td><td>KnM</td><td>KnS</td><td>T</td><td>C</td><td>D</td><td>V, Z</td><td>H</td><td>K</td><td>E</td><td>P</td></tr>
<tr><td>S1</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td>●</td><td>●</td><td>●</td><td>●</td><td>●</td><td>●</td><td>●</td><td>●</td><td>●</td><td>●</td><td></td><td></td></tr>
<tr><td>D1</td><td></td><td>▲</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td>●</td><td></td><td></td><td></td><td></td></tr>
<tr><td>D2</td><td></td><td></td><td>●</td><td></td><td></td><td>●</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td>●</td><td></td><td></td><td></td><td></td></tr>
<tr><td></td><td colspan="19">▲1: Please specify the transistor output Y000 ~ Y003 that supports the high speed output function.
▲2: When using Y000 ~ Y003 as the high-speed pulse output terminal, should use Y004 ~ Y007 for the rotation direction signal.</td></tr>
</table>

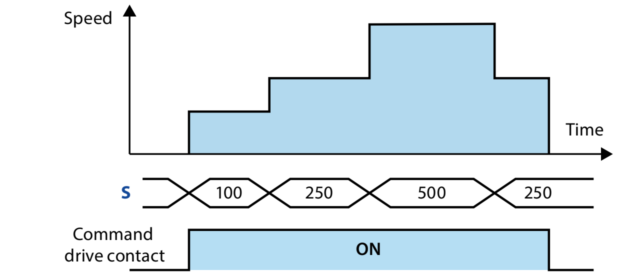
Function and Action Description

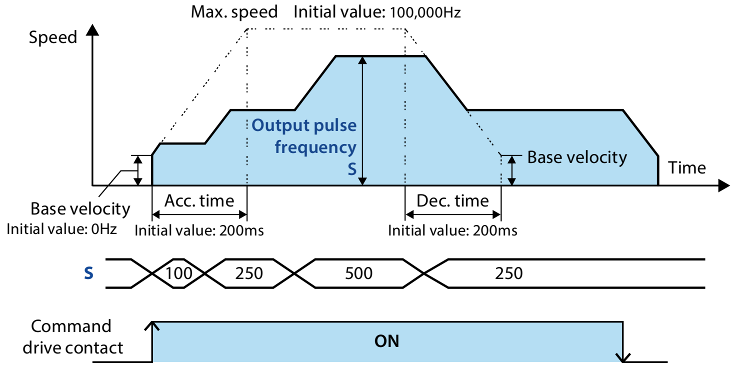
<table>
<tr><th>16-bit Operation (PLSV)</th><th>32-bit Operation (DPLSV)</th></tr>
<tr><td>

- S: S can be changed arbitrarily during pulse output.
  - When there is no Acc. /Dec. action (M8338 = OFF), if changes S, no Acc. or Dec. change in output frequency.
  - When there is Acc./Dec. action (M8338 = OFF), if changes S, the output frequency has Acc. or Dec. change.
- D1: Output number of output pulse.
- D2: The output terminal number of the rotation direction signal, the direction of rotation is shown in the table below.
  - Use Y000 ~ Y003 as high-speed pulse, at the output end, for the rotation direction signal, use Y004 ~ Y007.
  - During the execution of the instruction, please do not control the output specified by D2.

<table>
<tr><th>D2 Specified Device</th><th>Rotary Direction (Increase or Decrease of Current Value)</th></tr>
<tr><td>ON</td><td>Forward
S The value of the number of output pulses is positive.
The current value of D1 output pulse increases</td></tr>
<tr><td>OFF</td><td>Reverse
S The value of the output pulse number is negative, the current value of D1 output pulse decreases</td></tr>
</table>

</td><td>





</td></tr>
</table>

Related Soft Component

Please refer to 4.13.1.

Type	Related Soft Component
Special auxiliary relay	(1), (2), (3), (5), (8), (9), (14)
Special data relay	(2), (3), (4), (7), (8)

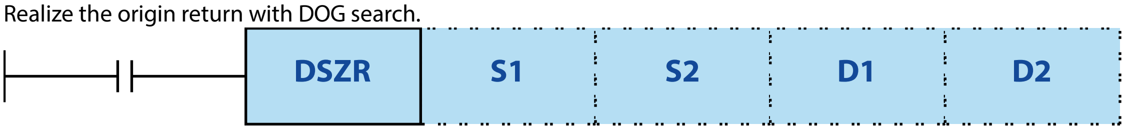
Note

Note	
1	During pulse output, if the command drive contact is OFF, it will decelerate and stop when there is Acc. and Dec., and stop immediately when there is no acceleration and deceleration. At this time, the instruction execution end flag [M8029] does not work.
2	When the limit flag bit of the operating direction (forward or reverse) is in action, it will decelerate and stop when there is Acc. or Dec., and stop immediately when there is no Acc. or Dec. At this time, the instruction execution abnormal end flag bit [M8329] turns ON.
3	The same high-speed pulse output terminal cannot execute multiple pulse output functions at the same time.

4.13.4 FN 150 - DSZR/ Return to Origin with DOG Search

Outline

Realize the origin return with DOG search.

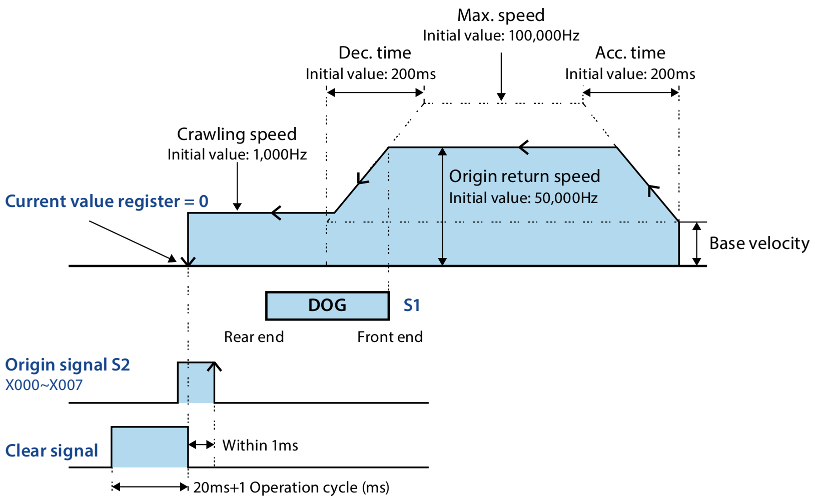


Return to Origin with DOG Search FN150 - DSZR	Instruction Mark	Execution Condition	Instruction Type	Number of Instruction Steps
	DSZR	Continuous type	16 bit	9

Operand	Setting Data															Data Type			
	S1: Specify the device number of the input near-point signal (DOG)															Bit			
	S2: Specify the input number of the input origin signal															Bit			
	D1: Specify the output number of the output pulse															Bit			
	D2: Specify the output number of the rotary direction signal															Bit			
	Operand Object Soft Component																		
	Bit Soft Component							Bit Soft Component								Bit Soft Component			
	X	Y	M	T	C	S	D.b	KnX	KnY	KnM	KnS	T	C	D	V, Z	H	K	E	P
S1	●	●	●			●													
S2	▲1																		
D1		▲2																	
D2		▲3	●			●													
	▲1: Please specify X000 ~ Y007. ▲2: Please specify the transistor output Y000 ~ Y003 that supports the high speed output function. ▲3: When using Y000 ~ Y003 as the high-speed pulse output terminal, Y004 ~ Y007 are recommended for the rotation direction signal.																		

Function and Action Description

16-bit Operation (DSZR)



It is forbidden to specify that the near-point signal S1 and the origin signal S2 are the same input.

16-bit Operation (DSZR)

- S1: Input the device number of the near-point signal (DOG), the logic is specified by the inversion flag.
- S2: The input number of the input origin signal, the logic is specified by the reverse flag bit.

Pulse Output Terminal Soft Component	Near-point Signal Logic Inversion Flag	Origin Signal Logic Inversion Flag	Content
D1 = Y000	M8345	M8346	OFF: Positive logic • When the input is ON, the signal is ON ON: Negative logic • When the input is OFF, the signal is ON
D1 = Y001	M8355	M8356	
D1 = Y002	M8365	M8366	
D1 = Y003	M8375	M8376	

- D1: Output number of output pulse.
- D2: The output terminal number of the rotation direction signal, and the specific rotation direction is shown in the table below.
  - When using Y000 ~ Y003 as the high-speed pulse output terminal, the rotation direction signal is recommended to be Y004 ~ Y007.
  - During the execution of the instruction, please do not control the output specified by D2.

D2 Specified Device	Rotation Direction (Increase or Decrease of Current Value)
ON	Forward rotation: The current value of D1 output pulse increases
OFF	Reverse: The current value of D1 output pulse decreases

- Origin return direction: Specified by the direction flag.

Pulse Output Terminal Soft Element	Origin Signal Logic Inversion Flag	Content
D1 = Y000	M8342	Return to origin in the forward direction: ON Return to origin in the reverse direction: OFF
D1 = Y001	M8352	
D1 = Y002	M8362	
D1 = Y003	M8372	

- Output clear signal: When the valid flag bit of the output function needs to be ON, it will be output after stopping at the origin position for a duration of [20 + 1 operation cycle].
  - Do not use the clear signal device designation function.

Pulse Output Terminal Soft Component	Clear Signal Output Valid Flag	Clear Signal Device Designation Function Valid Flag	Clear Signal Device Number
D1 = Y000	M8341 = ON	M8464 = OFF	Y004
D1 = Y001	M8351 = ON	M8465 = OFF	Y005
D1 = Y002	M8361 = ON	M8466 = OFF	Y006
D1 = Y003	M8371 = ON	M8467 = OFF	Y007

  - Use the clear signal device designation function.

Pulse Output Terminal Soft Component	Clear Signal Output Valid Flag	Clear Signal Device Designation Function Valid Flag	Clear Signal Soft Component
D1 = Y000	M8341 = ON	M8464 = ON	D8464
D1 = Y001	M8351 = ON	M8465 = ON	D8465
D1 = Y002	M8361 = ON	M8466 = ON	D8466
D1 = Y003	M8371 = ON	M8467 = ON	D8467

- Origin return speed: Follow base speed ≤ origin return speed ≤ Max. speed.
  - When the home position return speed > the Max. speed, the operation will be performed at the Max. speed.

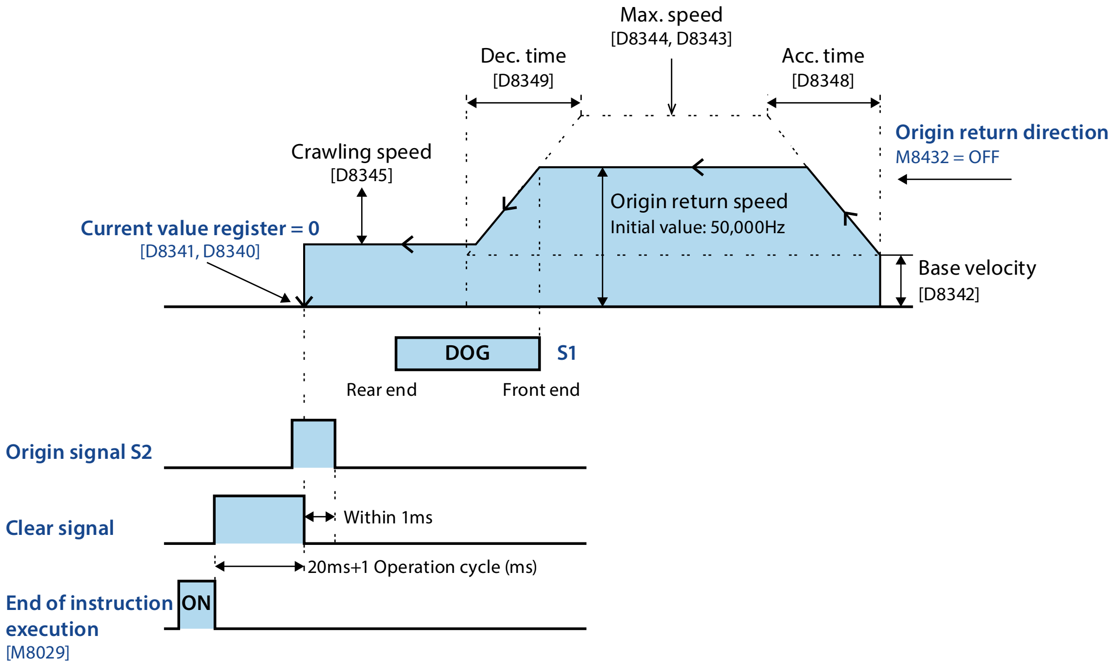
Pulse Output Terminal Soft Component	Base Velocity	Origin Return Speed	Max. Speed	Initial Value
D1 = Y000	D8342	D8347, D8346	D8344, D8343	50,000(Hz)
D1 = Y001	D8352	D8357, D8356	D8354, D8353	
D1 = Y002	D8362	D8367, D8366	D8364, D8363	
D1 = Y003	D8372	D8377, D8376	D8374, D8373	

16-bit Operation (DSZR)

- Crawling speed: Follow the base speed ≤ crawling speed ≤ origin return speed.

Pulse Output Terminal Soft Component	Base Velocity	Crawling Speed	Origin Return Speed	Initial Value
D1 = Y000	D8342	D8345	D8347, D8346	1,000 (Hz)
D1 = Y001	D8352	D8355	D8357, D8356	
D1 = Y002	D8362	D8365	D8367, D8366	
D1 = Y003	D8372	D8375	D8377, D8376	

Operation Description of Origin Return (D1 = Y000 as an Example)



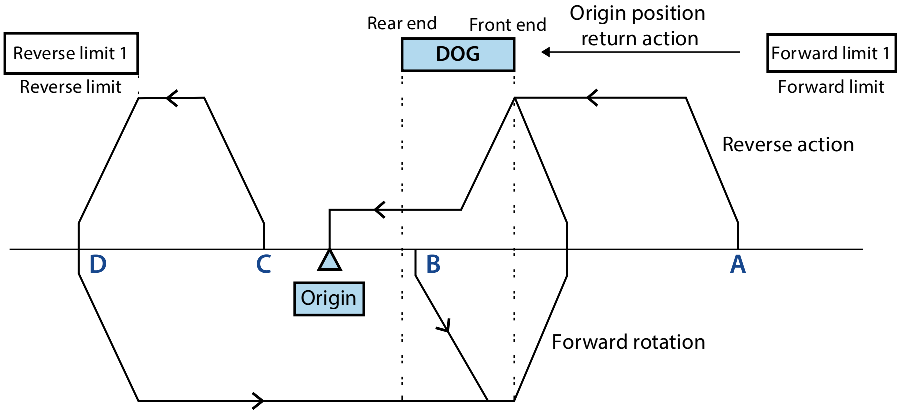
Action illustration:

1. Specify the direction of home return. Specify the direction flag [M8342].
2. Execute the home position return command [DSZR].
3. Origin return start action. Direction designated flag [M8342] designates the direction, origin return speed [D8347, D8346] designates the speed.
4. When the near-point signal (DOG) designated by S1 is ON[1], it starts to decelerate until the crawl speed is [D8345].
5. When it is detected that the designated origin signal of S2 is ON[2], the output pulse will be stopped immediately.
6. The current value register [D8341, D8340] becomes 0 (cleared).
7. If the clear signal output function is valid (D8341 = ON), the clear signal remains ON within 1ms of stopping the output pulse for [20ms + 1 operation cycle].
8. The instruction execution end flag [M8029] is ON, and the home return operation is ended.

1): When the near-point signal logic inversion flag [M8345] is OFF, OFF is valid.

2): When the origin signal logic reversal flag [M8346] is OFF, OFF is valid.

Operation Description of Origin Return



There are forward rotation limit, reverse rotation limit, DOG search origin return action description:

A.	Start Position before Passing DOG
1.	Execute the origin return command [DSZR] to start the origin return action.
2.	At the homing speed, the movement starts in the homing direction.
3.	The front end of the DOG is detected, and it starts to decelerate to the crawling speed.
4.	Stop when the origin is detected.

B.	The Starting Position is within the Passing DOG
1.	Execute the origin return command [DSZR] to start the origin return action.
2.	At the homing speed, the movement starts in the opposite direction of the homing direction.
3.	After detecting the front end of the DOG, it decelerates to a stop (leaves the DOG).
4.	At the homing speed, the movement starts in the homing direction (enter DOG again).
5.	The front end of the DOG is detected, and it starts to decelerate to the crawling speed.
6.	Stop when the origin is detected.
	Note: Actions 4 ~ 6 are the same as A.

C.	The Start Position is at the Near Point Signal OFF (after DOG is Passed)
1.	Execute the origin return command [DSZR] to start the origin return action.
2.	At the homing speed, the movement starts in the homing direction.
3.	Detect reverse limit 1 (reverse limit), decelerate to stop.
4.	At the homing speed, the movement starts in the opposite direction of the homing direction.
5.	After detecting the front end of the DOG, it decelerates to a stop (leaves the DOG).
6.	At the homing speed, the movement starts in the homing direction (enter DOG again).
7.	The front end of the DOG is detected, and it starts to decelerate to the crawling speed.
8.	Stop when the origin is detected.
	Note: Actions 4 ~ 8 are the same as B.

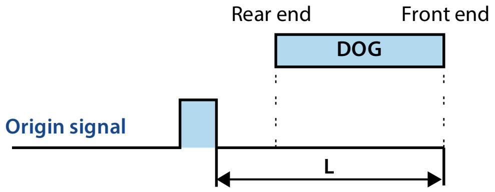
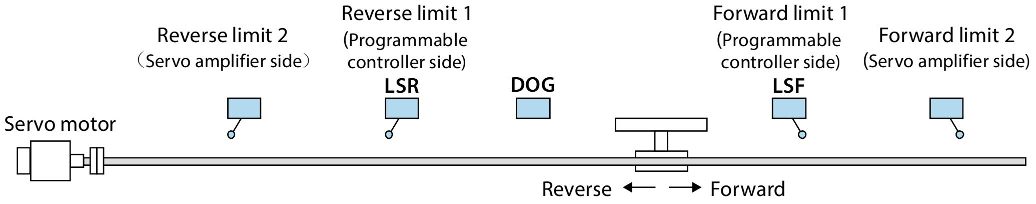
D.	The Direction Limit Switch for Home Return (Forward Rotation Limit 1 or Reverse Rotation Limit 1) is ON
1.	Execute the origin return command [DSZR] to start the origin return action.
2.	At the homing speed, the movement starts in the opposite direction of the homing direction.
3.	After detecting the front end of the DOG, it decelerates to a stop (leaves the DOG).
4.	At the homing speed, the movement starts in the homing direction (enter DOG again).
5.	The front end of the DOG is detected, and it starts to decelerate to the crawling speed.
6.	Stop when the origin is detected.
	Note: The action is the same as B.

Related Soft Component

Please refer to 4.13.1.

Type	Related Soft Component
Special auxiliary relay	(1), (2), (3), (5), (6), (7), (8), (9), (10), (11), (14), (16)
Special data relay	(2), (3), (4), (5), (6), (7), (8), (9)

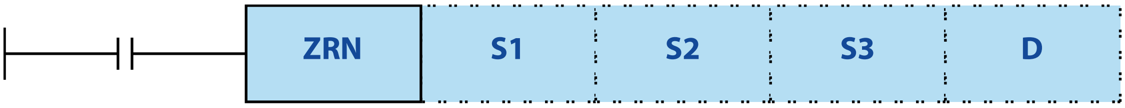
Note

Note	
1	The designated near-point signal (DOG) in S1 is X000 ~ X007, and the near-point signal (DOG) is monitored with a 1ms cycle (interrupt). If it is specified as another device, the signal detection is affected by the following conditions: • The refresh time that is entered. • The scan cycle of the program.
2	The distance (L) between the near-point signal (DOG) and the origin signal (L) must be long enough to ensure that it can decelerate to the crawling speed. Otherwise it will cause position shift. 
3	The near-point signal (DOG) should be set between the forward rotation limit 1 (LSR) and the reverse rotation limit 1 (LSR), as shown in the figure below. Otherwise, you may not be able to perform the action.  Reverse limit 2 （Servo amplifier side） Reverse limit 1 (Programmable controller side) LSR DOG Forward limit 1 (Programmable controller side) LSF Forward limit 2 (Servo amplifier side) Servo motor Reverse Forward
4	The devices designated by the near-point signal S1 and the origin signal S2 can no longer be designated as the following functions: • High-speed counter • Input interrupt • Pulse capture • DVIT • ZRN
5	The crawling speed must be slow enough. The return-to-origin command stops without deceleration. If the speed is too fast, the stop position will shift due to inertia.
6	When the instruction is executed, the value of the operand is directly modified, and the modification content is invalid. It is necessary to disconnect the command drive contact first, and then turn it ON, to modify the content to be effective.
7	During the origin return, when the command drive contact turns off, it decelerates to a stop. The instruction execution end flag [M8029] is not turned ON.
8	The same high-speed pulse output terminal cannot perform multiple pulse output functions at the same time.
9	When the near-point signal (DOG) cannot be detected, it will decelerate to a stop. The instruction execution abnormal end flag bit [M8329] turns ON to end the execution of the instruction.

4.13.5 FN 156 - ZRN/Return to the Origin

Outline

Return to origin.



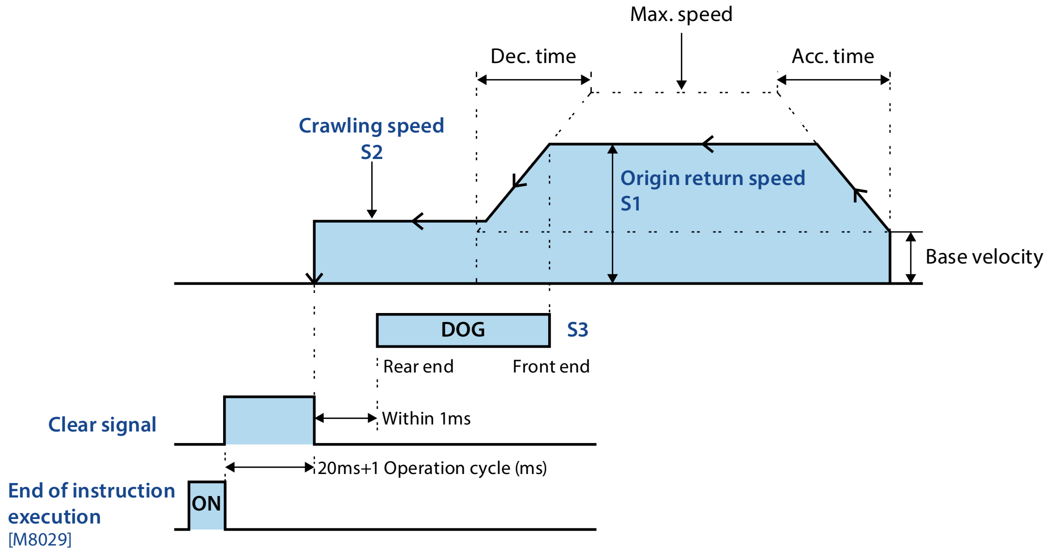
Return to Origin FN156 - ZRN	Instruction Mark	Execution Condition	Instruction Type	Number of Instruction Steps
	ZRN	Continuous type	16 bit	9
	DZRN	Continuous type	32 bit	17

Operand	Setting Data	Data Type
	S1: Specify the speed at the beginning of home return • 16-bit operation, 1 ~ 32,767 (Hz) • 32-bit operation, 1 ~ 100,000 (Hz)	16/32 bit
	S2: Specify crawl speed, 1 ~ 32,767 (Hz)	16/32 bit
	S3: Specify the input number of the input near-point signal (DOG)	Bit
	D: Specify the output number of the output pulse	Bit

Operand Object Soft Component	Bit Soft Component							Bit Soft Component								Bit Soft Component			
	X	Y	M	T	C	S	D.b	KnX	KnY	KnM	KnS	T	C	D	V, Z	H	K	E	P
S1								●	●	●	●	●	●	●	●	●	●		
S2								●	●	●	●	●	●	●	●	●	●		
S3	●	●	●			●													
D		▲																	
	▲: Please specify the transistor output Y000 ~ Y003 that supports the high speed output function.																		

Function and Action Description

16-bit Operation (ZRN)	32-bit Operation (DZRN)



16-bit Operation (ZRN)	32-bit Operation (DZRN)

- S1: Pecify the origin return speed. When the home position return speed ≥ the Max. speed, it will act at the highest speed.
  - The return-to-origin speed specified in the special data register is invalid.

Pulse Output Terminal Soft Element	Origin Return Speed
D = Y000	D8347, D8346
D = Y001	D8357, D8356
D = Y002	D8367, D8366
D = Y003	D8377, D8376

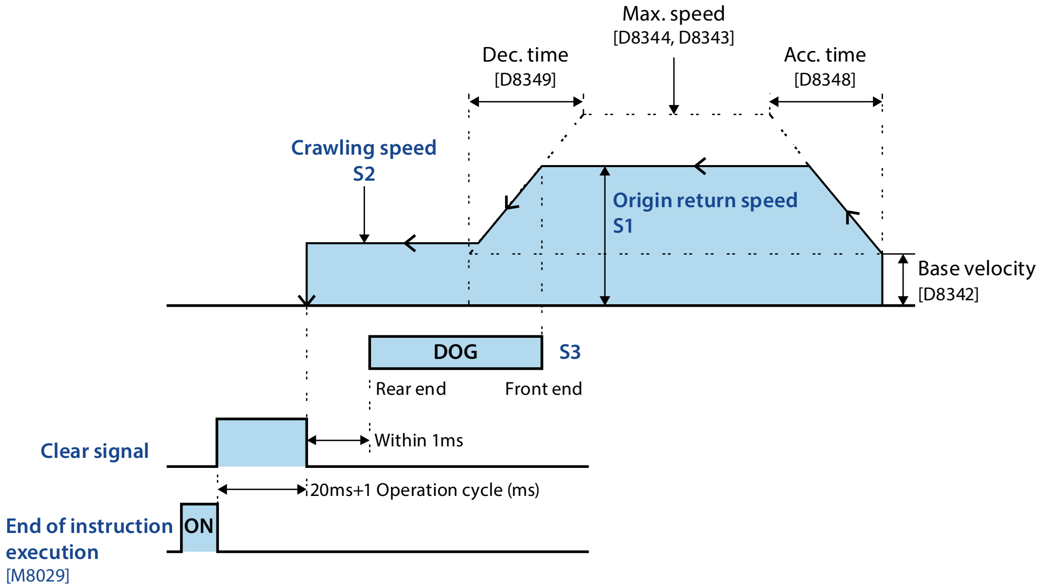
- S2: Specify crawl speed.
- S3: Input the device number of the near-point signal (DOG).
  - When the near-point signal (DOG) is ON, it starts to decelerate to the crawl speed until the DOG is OFF, and the home return is finished.
- D: Output number of output pulse.
- Origin return direction: specified by the direction flag.
  - During home return, the value of the current value register [PLS] decreases.
- Output clear signal: When the valid flag bit of the output function needs to be ON, it will be output after stopping at the origin position, duration [20 + 1 operation cycle] (same as DSZR).
  - Do not use the clear signal device designation function.

Pulse Output Terminal Soft Component	Clear Signal Output Valid Flag	Clear Signal Device Designation Function Valid Flag	Clear Signal Device Number
D = Y000	M8341 = ON	M8464 = OFF	Y004
D = Y001	M8351 = ON	M8465 = OFF	Y005
D = Y002	M8361 = ON	M8466 = OFF	Y006
D = Y003	M8371 = ON	M8467 = OFF	Y007

  - Use the clear signal device designation function.

Pulse Output Terminal Soft Component	Clear Signal Output Valid Flag	Clear Signal Device Designation Function Valid Flag	Clear Signal Device Number
D = Y000	M8341 = ON	M8464 = ON	D8464
D = Y001	M8351 = ON	M8465 = ON	D8465
D = Y002	M8361 = ON	M8466 = ON	D8466
D = Y003	M8371 = ON	M8467 = ON	D8467

Operation Description of Origin Return (D = Y000 as an Example)



Action illustration:

1. Execute the home position return command [ZRN].
2. The homing start action. S1 specifies the homing speed.
3. When the near-point signal (DOG) designated by S3 is ON, it starts to decelerate until the crawl speed (designated by S2).
4. After the near-point signal (DOG) designated by S3 is OFF, the pulse output will stop immediately.
5. The current value register [D8341, D8340] becomes 0 (cleared).
6. If the clear signal output function is valid (D8341 = ON), the clear signal remains ON within 1ms of stopping the output pulse for [20ms + 1 operation cycle].
7. The instruction execution flag [M8029] is ON, and the home return operation is ended.

Related Soft Component

Please refer to 4.13.1.

Type	Related Soft Component
Special auxiliary relay	(1), (2), (3), (5), (6), (14), (16)
Special data relay	(2), (3), (4), (7), (8), (9)

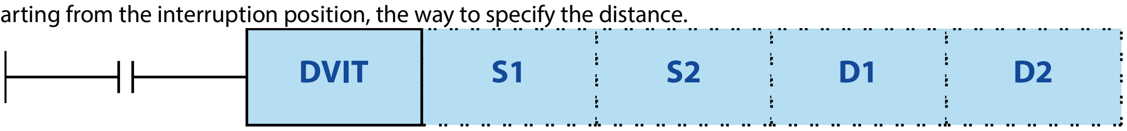
Note

Note	
1	The designated near-point signal (DOG) in S1 is X000 ~ X007, and the near-point signal (DOG) is monitored with a 1ms cycle (interrupt). If it is specified as another device, the signal detection is affected by the following conditions: • Enter the refresh time. • The scan cycle of the program.
2	The time for the near-point signal (DOG) to be ON must be long enough to ensure that it can decelerate to the crawl speed. Otherwise it will cause position shift.
3	The device designated by the near-point signal S1 can no longer be designated as the following functions: • High-speed counter • Input interrupt • Pulse capture • DVIT • ZRN
4	The crawling speed must be slow enough.
5	The return-to-origin command stops without deceleration. If the speed is too fast, the stop position will shift due to inertia.
6	Please start from the front part of the near-point signal (DOG), DOG search is not supported.
7	When you need to fine-tune the position of the origin, adjust the position of the near point (DOG).
8	During home return, when the command drive contact turns off, it decelerates to a stop.
9	The instruction execution end flag [M8029] is not turned ON.

4.13.6 FN 151 - DVIT/Interrupt Positioning

Outline

Starting from the interruption position, the way to specify the distance.



Interrupt Positioning FN151 - DVIT	Instruction Mark	Execution Condition	Instruction Type	Number of Instruction Steps
	DVIT	Continuous type	16 bit	9
	DDVIT	Continuous type	32 bit	17

Operand	Setting Data	Data Type
	S1: Specify the number of output pulses after interruption (absolute address) • 16-bit operation, -32,767 ~ +32,767 (except 0) • 32-bit operation, -2,147,483,648 ~ +2,147,483,648 (except 0)	16/32 bit
	S2: Specify the output pulse frequency • 16-bit operation, 1 ~ 32,767 (Hz) • 32-bit operation, 1 ~ 100,000 (Hz)	16/32 bit
	D1: Specify the output number of the output pulse	Bit
	D2: Specify the output number of the rotary direction signal	Bit

	Operand Object Soft Component																		
	Bit Soft Component							Bit Soft Component								Bit Soft Component			
	X	Y	M	T	C	S	D.b	KnX	KnY	KnM	KnS	T	C	D	V, Z	H	K	E	P
S1								●	●	●	●	●	●	●		●	●		
S2								●	●	●	●	●	●	●		●	●		
D1		▲1																	
D2		▲2	●			●													

▲1: Please specify the transistor output Y000 ~ Y003 that supports the high speed output function.

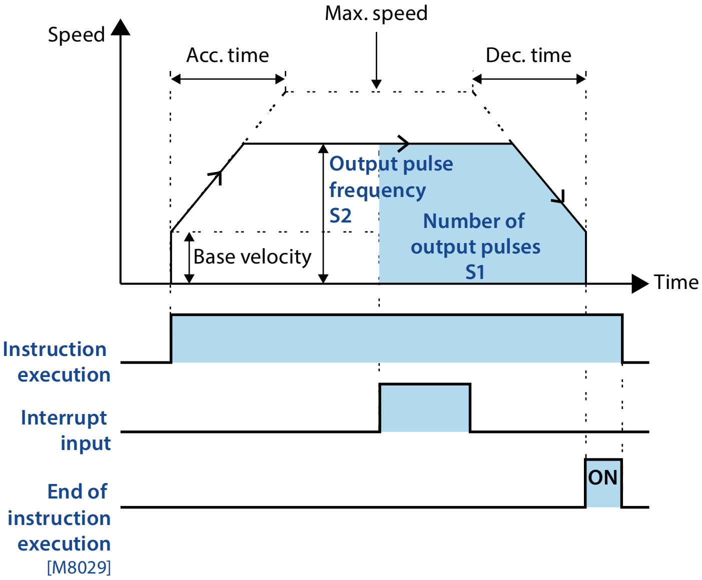
▲2: When using Y000 ~ Y003 as the high-speed pulse output terminal, Y004 ~ Y007 are recommended for the rotation direction signal.

Function and Action Description

16-bit Operation (DVIT)	32-bit Operation (DDVIT)

- S1: Specify the number of output pulses after interruption (relative address value).
- S2: Specify the output pulse frequency.
- D1: The output number of the output pulse.
- D2: The output terminal number of the rotation direction signal, and the specific rotation direction is shown in the table below.
  - When using Y000 ~ Y003 as the high-speed pulse output terminal, the rotation direction signal is recommended to be Y004 ~ Y007.
  - During the execution of the instruction, please do not control the output specified by D2.

D2 Specic the Soft Component ON/OFF	Rotation Direction (Increase or Decrease of Current Value)
ON	Forward The value of S1 output pulse number is positive, and the current value of D1 output pulse increases
OFF	Reverse The value of S1 output pulse number is negative, and the current value of D1 output pulse decreases



16-bit Operation (DVIT)	32-bit Operation (DDVIT)

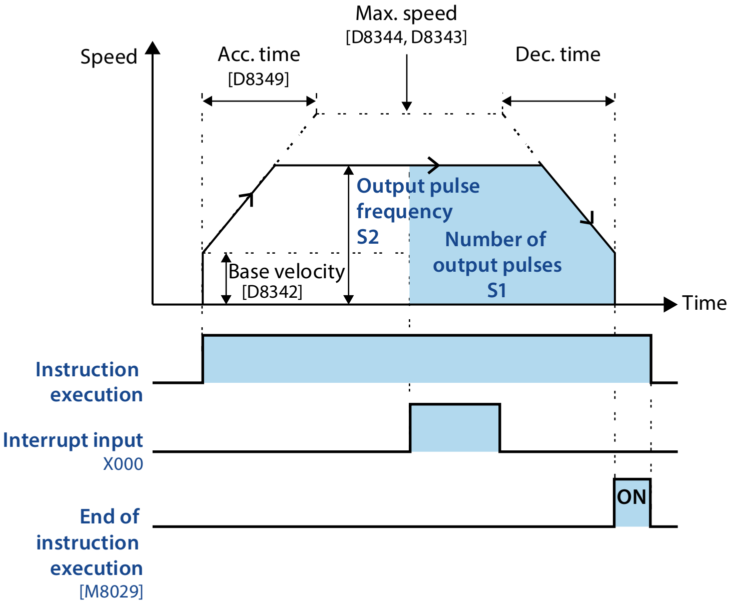
- Interrupt input signal, the specification method is shown in the table below.

Pulse Output Soft Element	Interrupt Input Signal		
	User Interrupt Input Instruction Soft Element	Interrupt Input Designated Function M8336 = ON	
		D8336 Set Interrupt Input Designated Function	Interrupt Signal Logic Inversion Flag*
D1 = Y000	M8460	D8336 = H□□□□ Interrupt input used by Y000 Interrupt input used by Y001 Interrupt input used by Y002 Interrupt input used by Y003 D8336 set: • 0 ~ 7: X000 ~ X007 are designated as interrupt input • 8 ~ F: X010 ~ X017 are designated as interrupt input	M8347
D1 = Y001	M8461		M8357
D1 = Y002	M8462		M8367
D1 = Y003	M8463		M8377
*The logic inversion flag specifies the logic: • OFF: Positive logic (when the input is ON, the interrupt signal is ON) • ON: Negative logic (when the input is OFF, the interrupt signal is ON)			

Description of Interrupt Positioning Action (D1 = Y000 as an Example)

Action illustration:

1. Execute interrupt positioning command [DVIT].
2. Start action. S1 specifies the number of output pulses, and S2 specifies the output pulse frequency.
3. The interrupt input X000 has input, and it stops after outputting the number of pulses specified by S1.
4. The instruction execution flag [M8029] is ON, and the home return operation is ended.

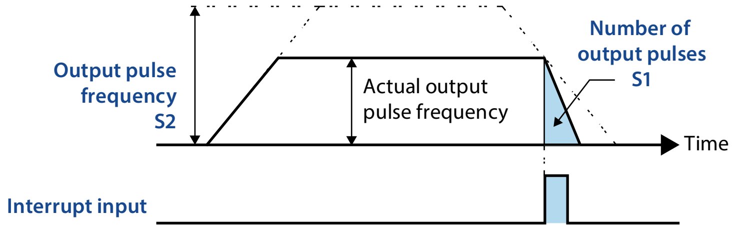
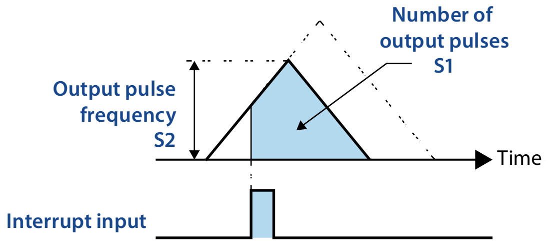


Related Soft Component

Please refer to 4.13.1.

Type	Related Soft Component
Special auxiliary relay	(1), (2), (3), (4), (5), (8), (9), (12), (13), (14), (15)
Special data relay	(1), (2), (3), (4), (7), (8)

Note

Note	
1	When the number of pulses specified in S1 < the number of pulses required for deceleration, the action will be performed according to the specified number of pulses, as shown in the figure below.  Number of output pulses S1 Output pulse frequency S2 Actual output pulse frequency Time Interrupt input
2	During acceleration, the interrupt input may be ON, and the number of pulses specified in S1 follows: Output pulse number ≥ pulse number required for acceleration + pulse number required for deceleration. Otherwise, follow the action as shown in the figure below.  Number of output pulses S1 Output pulse frequency S2 Time Interrupt input
3	The designated interrupt input device can no longer be designated as the following functions: • High-speed counter • Input interrupt • Pulse capture • DVIT • ZRN
4	When the instruction is executed, the value of the operand is directly modified, and the modification content is invalid. It is necessary to disconnect the command drive contact first, and then turn it ON, to modify the content to be effective.
5	During interrupt positioning, when the command drive contact turns off, it decelerates to a stop. The instruction execution end flag [M8029] is not turned ON.
6	The same high-speed pulse output terminal cannot perform multiple pulse output functions at the same time.
7	When the limit flag of the action direction (forward or reverse) is in action, it will decelerate to a stop. The instruction execution abnormal end flag bit [M8329] turns ON to end the execution of the instruction.

4.13.7 FN 158 - DRVI/Relative Positioning

4.13.8 FN 159 - DRVA/Absolute Positioning

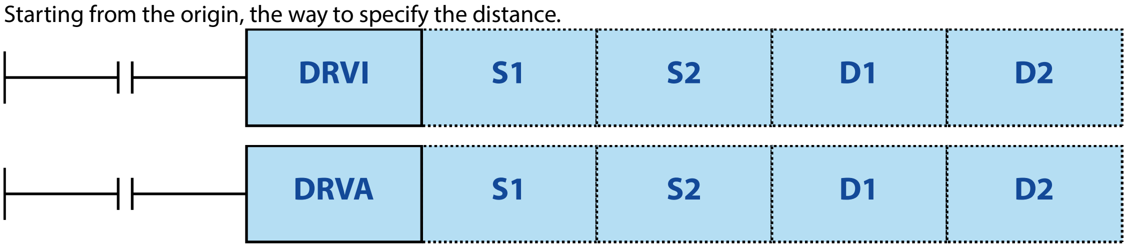
Outline

DRVI/Relative Positioning

Starting from the current position, the way to specify the distance.

DRVA/Absolute Positioning

Starting from the origin, the way to specify the distance.



	Instruction Mark	Execution Condition	Instruction Type	Number of Instruction Steps
Relative Positioning FN158 - DRVI	DRVI	Continuous type	16 bit	9
	DDRVI	Continuous type	32 bit	17
Absolute Positioning FN159 - DRVA	**Instruction Mark**	**Execution Condition**	**Instruction Type**	**Number of Instruction Steps**
	DRVA	Continuous type	16 bit	9
	DDRVA	Continuous type	32 bit	17

	Setting Data	Data Type
Operand	S1: Specify the number of output pulse. Setting range: • For 16 bit operation: -32,768 ~ +32,767 • For 32 bit operation: -2,147,483,648 ~ +2,147,483,647	16/32 bit
	S2: Specify the output pulse frequency. Setting range: • For 16 bit operation: 10 ~ 32,767 (Hz) • For 32 bit operation: 10 ~ 100,000 (Hz)	16/32 bit
	D1: Specify the output number of the output pulse	Bit
	D2: Specify the output number of the rotary direction signal	Bit

	Operand Object Soft Component																		
	Bit Soft Component							Word Soft Component								Others			
	X	Y	M	T	C	S	D.b	KnX	KnY	KnM	KnS	T	C	D	V, Z	H	K	E	P
S1								●	●	●	●	●	●	●	●	●	●		
S2								●	●	●	●	●	●	●	●	●	●		
D1		●													●				
D2		●	●			●									●				

▲1: Please specify the transistor output Y000 ~ Y003 that supports the high speed output function.

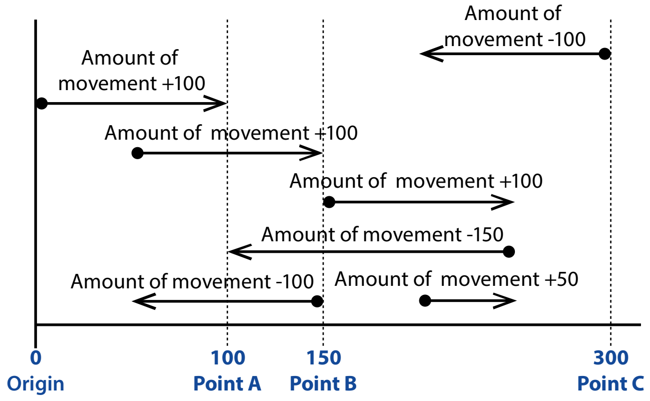
▲2: When using Y000 ~ Y003 as the high-speed pulse output terminal, Y004 ~ Y007 are recommended for the rotation direction signal.

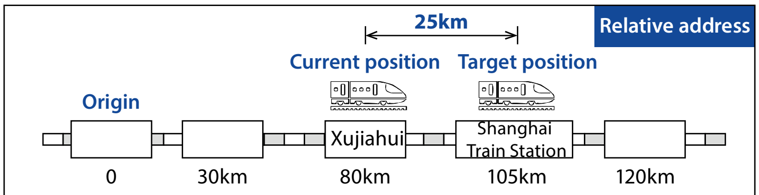
Function and Action Description

Relative Positioning (DRVI)/Absolute Positioning (DRVA)

Relative Positioning (DRVI):
Refers to positioning in incremental mode (relative address).

Use the current stop position as the starting point, specify the direction of movement and the amount of movement (relative address) for positioning.

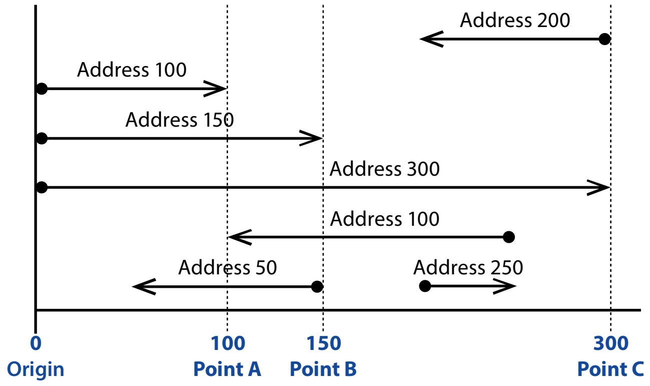


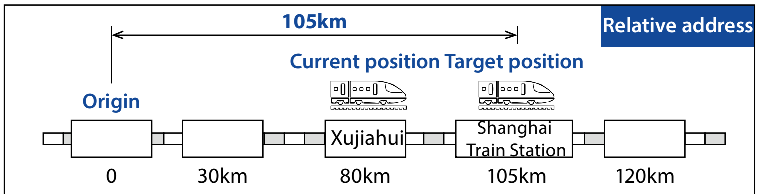


When moving from Xujiahui (current) to Shanghai Train Station (target), set the distance from Xujiahui to +25km.

Absolute Positioning (DRVA):
Refers to positioning in absolute mode (absolute address).

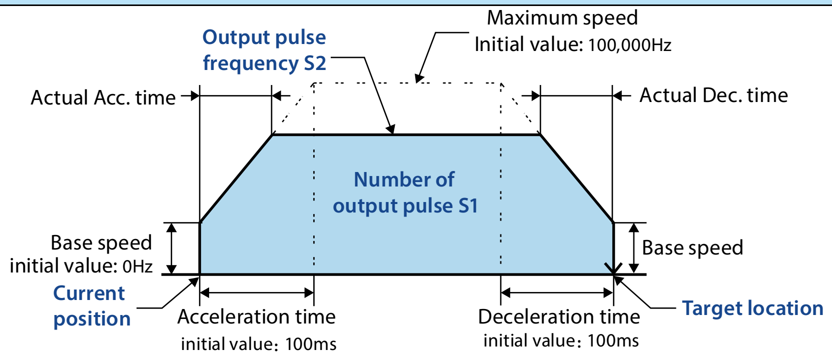
Specify the position (absolute address) based on the origin to locate. It doesn't matter where the starting point is.





When moving from Xujiahui (current) to Shanghai Train Station (target), set the distance from origin to 105km.

Relative Positioning (DRVI)/Absolute Positioning (DRVA)


Maximum speed
Initial value: 100,000Hz
Output pulse frequency S2
Actual Acc. time
Actual Dec. time
Number of output pulse S1
Base speed initial value: 0Hz
Base speed
Current position
Acceleration time initial value: 100ms
Deceleration time initial value: 100ms
Target location

- S1: Specify the number of output pulses.
- S2: Specify the output pulse frequency.
- D1: The output number of the output pulse.
- D2: The output terminal number of the rotation direction signal, and the specific rotation direction is shown in the table below.
  - When using Y000 ~ Y003 as the high-speed pulse output terminal, the rotation direction signal is recommended to be Y004 ~ Y007.
  - During the execution of the instruction, please do not control the output specified by D2.

D2 Specified Device	Rotation Direction (Increase or Decrease of Current Value)		
	Relative Positioning (DRVI)	Absolute Positioning (DRVA)	
ON	Forward The value of S1 output pulse number is positive The current value of D1 output pulse increases	Forward The current value of D1 output pulse increases	Forward or reverse rotation is determined by the following relationship: • The number of output pulses specified by S1 • The value of the current value register (PLS)
OFF	Reverse The value of S1 output pulse number is negative The current value of D1 output pulse decreases	Reverse The current value of D1 output pulse decreases	

Related Soft Component

Please refer to 4.13.1.

Type	Related Soft Component
Special auxiliary relay	(1), (2), (3), (5), (8), (9), (13), (14)
Special data relay	(2), (3), (4), (7), (8)

Note

Note	
1	During the execution of the instruction, directly modify the value of the operand, and the action does not change. It will be effective the next time the command is driven.
2	During relative positioning or absolute positioning, when the command drive contact turns off, it will decelerate to a stop. The instruction execution end flag [M8029] is not turned ON.
3	The same high-speed pulse output terminal cannot perform multiple pulse output functions at the same time.
4	When the limit flag of the action direction (forward or reverse) is in action, it will decelerate to a stop. At this time, the instruction execution abnormal end flag bit [M8329] turns ON to end the execution of the instruction.

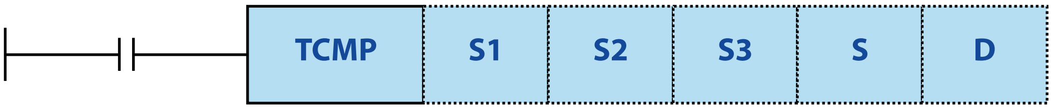
4.14 Clock Operation - FN 160 ~ FN 169

FN No.	Instruction Mark	Instruction Format	Function	Section	Page
160	TCMP	TCMP (S1) (S2) (S3) (S) (D) TCMPP (S1) (S2) (S3) (S) (D)	Clock data comparison	4.14.1	197
161	TZCP	TZCP (S1) (S2) (S) (D) TZCPP (S1) (S2) (S) (D)	Clock data interval comparison	4.14.1	197
162	TADD	TADD (S1) (S2) (D) TADDP (S1) (S2) (D)	Clock data addition	4.14.3	199
163	TSUB	TSUB (S1) (S2) (D) TSUBP (S1) (S2) (D)	Clock data subtraction	4.14.4	200
164	HTOS	HTOS (S) (D) HTOSP (S) (D) DHTOS (S) (D) DHTOSP (S) (D)	Second conversion of hour, minute, and second data	4.14.5	200
165	STOH	STOH (S) (D) STOHP (S) (D) DSTOH (S) (D) DSTOHP (S) (D)	[hour, minute, second] conversion of second data	4.14.6	201
166	TRD	TRD (D) TRDP (D)	Clock data reading	4.14.7	203
167	TWR	TWR (S) TWRP (S)	Clock data writing	4.14.8	204
169	HOUR	HOUR (S) (D1) (D2) DHOUR (S) (D1) (D2)	Timer	4.14.9	205

4.14.1 FN 160 - TCMP/Clock Data Comparison

Outline

The comparison base time and time data are compared in size, and the bit soft component ON/OFF is controlled according to the result of the comparison.



Clock Data Comparison FN160 - TCMP	Instruction Mark	Execution Condition	Instruction Type	Number of Instruction Steps
	TCMP	Continuous type	16 bit	11
	TCMPP	Pulse type	16 bit	11

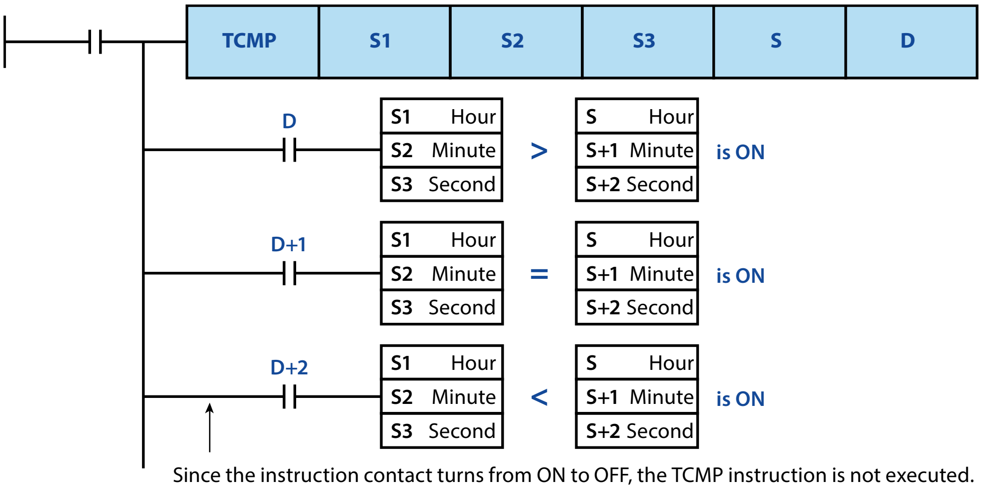
Operand	Setting Data	Data Type
	S1: Specify the "hour" of the comparison base time [setting range: 0 ~ 23]	16 bit
	S2: Specify the "minute" of the comparison base time [setting range: 0 ~ 59]	16 bit
	S3: Specify the "second" of the comparison base time [setting range: 0 ~ 59]	16 bit
	S: Specify the "hour" of the time data (time, minute, second) (occupied 3 points)	16 bit
	D: ON/OFF bit soft component according to the comparison result (occupyed 3 points)	Bit

Operand	Operand Object Soft Component																	
	Bit Soft Component							Word Soft Component								Others		
	X	Y	M	T	C	S	D.b	KnX	KnY	KnM	KnS	T	C	D	V, Z	H	K	E	P
S1								●	●	●	●	●	●	●	●	●	●		
S2								●	●	●	●	●	●	●	●	●	●		
S3								●	●	●	●	●	●	●	●	●	●		
S												●	●	●	●				
D		●	●			●	●								●				

Function and Action Description

16-bit Operation (TCMP, TCMPP)

Compare the time of the comparison base time (hour, minute, second) [S1,S2,S3] with the time data (hour, minute, second) [S,S+1,S+2]. And turn ON/OFF the three points starting from D according to the result of the comparison.



Since the instruction contact turns from ON to OFF, the TCMP instruction is not executed. Even so, D, D+1, D+2 will maintain the status before the instruction contact is OFF.

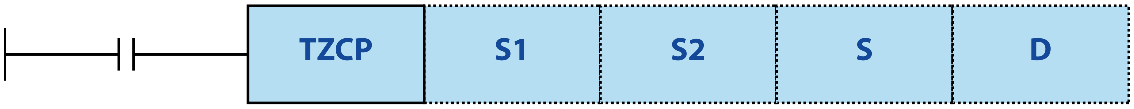
Note

Note		Description
1	Number of occupied points of soft component	S and D respectively occupies 3 points of soft component. Please be careful not to duplicate the soft component used in other control of the machine.
2	When using the time (hour, minute, second) of the clock data of the intelligent controller built-in real-time clock	Please use the TRD (FN 166) instruction to read the value of the special data register and specify its word soft component in each operand.

4.14.2 FN 161 - TZCP/Clock Data Interval Comparison

Outline

The comparison base time and time data of the upper and lower 2 points are compared in size, and the bit soft component ON/OFF is controlled according to the result of the comparison.



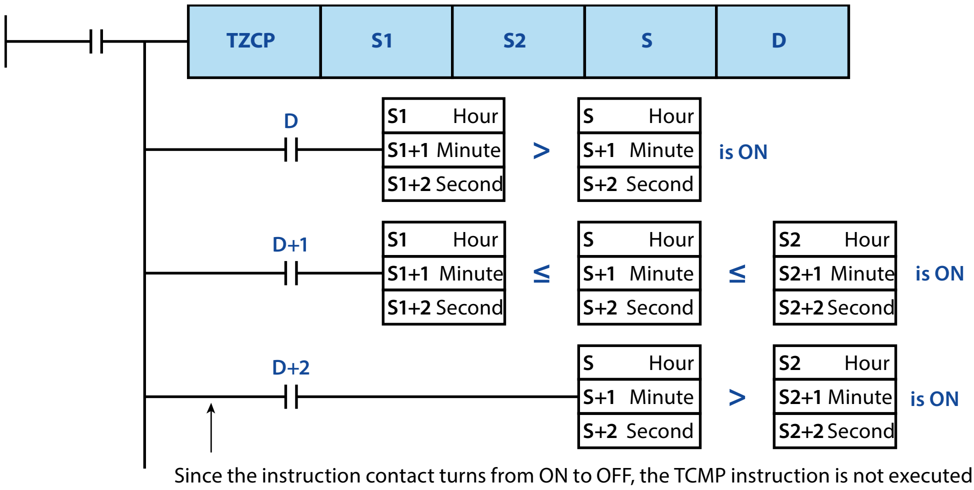
Clock Data Interval Comparison FN161 - TZCP	Instruction Mark	Execution Condition	Instruction Type	Number of Instruction Steps
	TZCP	Continuous type	16 bit	9
	TZCPP	Pulse type	16 bit	9

<table>
<tr><td rowspan="8">Operand</td><td colspan="11">Setting Data</td><td colspan="8">Data Type</td></tr>
<tr><td colspan="11">S1: Specify the "hour" of the comparison lower limit time (hour, minute, second) (occupied 3 points)</td><td colspan="8">16 bit</td></tr>
<tr><td colspan="11">S2: Specify the "hour" of the comparison upper limit time (hour, minute, second) (occupied 3 points)</td><td colspan="8">16 bit</td></tr>
<tr><td colspan="11">S: Specify the "time" of the time data (hour, minute, second) (occupied 3 points)</td><td colspan="8">16 bit</td></tr>
<tr><td colspan="11">D: ON/OFF bit soft component according to the comparison result (occupied 3 points)</td><td colspan="8">Bit</td></tr>
<tr><td colspan="19">Operand Object Soft Component</td></tr>
<tr><td colspan="7">Bit Soft Component</td><td colspan="8">Word Soft Component</td><td colspan="4">Others</td></tr>
<tr><td>X</td><td>Y</td><td>M</td><td>T</td><td>C</td><td>S</td><td>D.b</td><td>KnX</td><td>KnY</td><td>KnM</td><td>KnS</td><td>T</td><td>C</td><td>D</td><td>V, Z</td><td>H</td><td>K</td><td>E</td><td>P</td></tr>
<tr><td>S1</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td>●</td><td>●</td><td>●</td><td>●</td><td></td><td></td><td></td><td></td></tr>
<tr><td>S2</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td>●</td><td>●</td><td>●</td><td>●</td><td></td><td></td><td></td><td></td></tr>
<tr><td>S</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td>●</td><td>●</td><td>●</td><td>●</td><td></td><td></td><td></td><td></td></tr>
<tr><td>D</td><td></td><td>●</td><td>●</td><td></td><td></td><td>●</td><td>●</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td>●</td><td></td><td></td><td></td><td></td></tr>
</table>

Function and Action Description

16-bit Operation (TZCP, TZCPP)

Compare the upper and lower 2-point comparison base time (hour, minute, and second) with the 3-point time data (hour, minute, and second) starting from S, and turn ON/OFF the 3-point soft components starting from D according to the result of the comparison.



Since the instruction contact turns from ON to OFF, the TCMP instruction is not executed. Even so, D, D+1, D+2 will maintain the status before the instruction contact is OFF.

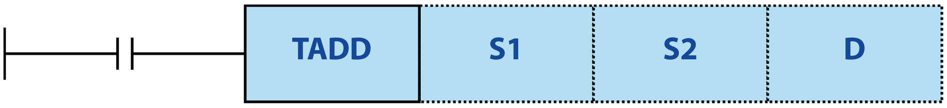
Note

Note		Description
1	Number of occupied points of soft component	S1, S2, S, D respectively occupies 3 points of soft component. Please be careful not to duplicate the soft component used in other control of the machine.
2	When using the time (hour, minute, second) of the clock data of the intelligent controller built-in real-time clock	Please use the TRD (FN 166) instruction to read the value of the special data register and specify its word soft component in each operand.

4.14.3 FN 162 - TADD/Clock Data Addition

Outline

Perform addition operation in 2 times data, the result is saved in the word soft component.



Clock Data Addition FN162 - TADD	Instruction Mark	Execution Condition	Instruction Type	Number of Instruction Steps
	TADD	Continuous type	16 bit	7
	TADDP	Pulse type	16 bit	7

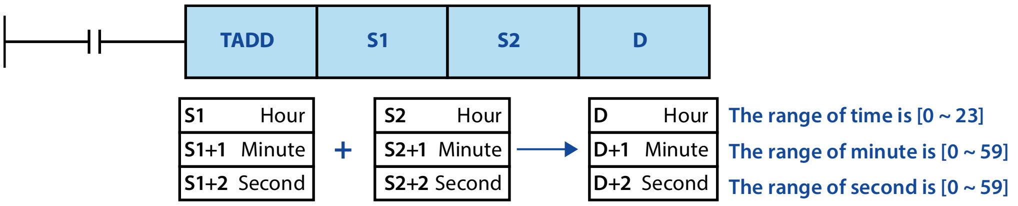
Operand	Setting Data	Data Type
	S1: Specify the "hour" of the time data (hour, minute, second) that performs addition operation (occupied 3 points)	16 bit
	S2: Specify the "hour" of the time data (hour, minute, second) that performs addition operation (occupied 3 points)	16 bit
	D: Save the result of 2 time data (hour, minute, second) addition operation (occupied 3 points)	16 bit

Operand	Operand Object Soft Component																		
	Bit Soft Component							Word Soft Component								Others			
	X	Y	M	T	C	S	D.b	KnX	KnY	KnM	KnS	T	C	D	V, Z	H	K	E	P
S1												●	●	●	●				
S2												●	●	●	●				
D												●	●	●	●				

Function and Action Description

16-bit Operation (TADD, TADDP)

Add the time data (hour, minute, and second) of [S1,S1+1,S1+2] and the time data (hour, minute, and second) of [S2,S2+1,S2+2], and the operation result is saved in [D,D+1,D+2] (hour, minute, second).



- When the operation result exceeds 24 hours, the carry flag bit turns ON, and the time is subtracted from the simple addition value for 24 hours and then is saved as the operation result.
- When the operation result is 0 (0:0:0), the zero flag bit turns ON.

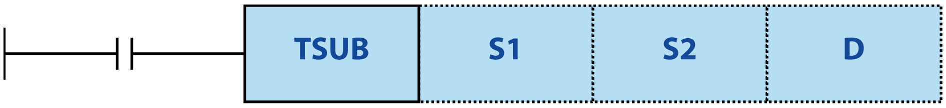
Note

Note		Description
1	Number of occupied points of soft component	S1, S2, D respectively occupies 3 points of soft component. Please be careful not to duplicate the soft component used in other control of the machine.
2	When using the time (hour, minute, second) of the clock data of the intelligent controller built-in real-time clock	Please use the TRD (FN 166) instruction to read the value of the special data register and specify its word soft component in each operand.

4.14.4 FN 163 - TSUB/Clock Data Subtraction

Outline

Perform subtraction operation in 2 time data, the result is saved in the word soft component.



Clock Data Subtraction FN163 - TSUB	Instruction Mark	Execution Condition	Instruction Type	Number of Instruction Steps
	TSUB	Continuous type	16	7
	TSUBP	Pulse type	16	7

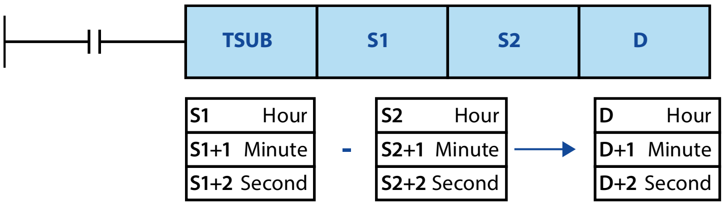
Operand	Setting Data	Data Type
	S1: Specify the "hour" of the time data (hour, minute, second) that performs subtraction operation (occupied 3 points)	16 bit
	S2: Specify the "hour" of the time data (hour, minute, second) that performs subtraction operation (occupied 3 points)	16 bit
	D: Save the result of 2 time data (hour, minute, second) subtraction operation (occupied 3 points)	16 bit

Operand	Operand Object Soft Component																		
	Bit Soft Component							Word Soft Component								Others			
	X	Y	M	T	C	S	D.b	KnX	KnY	KnM	KnS	T	C	D	V, Z	H	K	E	P
S1												●	●	●	●				
S2												●	●	●	●				
D												●	●	●	●				

Function and Action Description

16-bit Operation (TSUB, TSUBP)

Subtract the time data (hour, minute, and second) of [S2,S2+1,S2+2] from the time data (hour, minute, and second) of [S1,S1+1,S1+2], and the operation result is saved in [D,D+1,D+2] (hour, minute, second).



The range of time is [0 ~ 23]

The range of minute is [0 ~ 59]

The range of second is [0 ~ 59]

- When the operation result is less than 0, the borrow flag bit turns ON, and the time is added from the simple subtraction value for 24 hours and then is saved as the operation result.
- When the operation result is 0 (0:0:0), the zero flag bit turns ON.

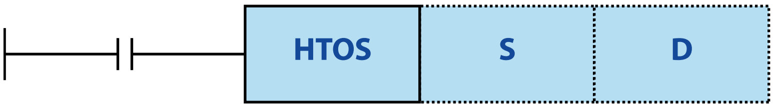
Note

Note		Description
1	Number of occupied points of soft component	S1, S2, D respectively occupies 3 points of soft component. Please be careful not to duplicate the soft component used in other control of the machine.
2	When using the time (hour, minute, second) of the clock data of the intelligent controller built-in real-time clock	Please use the TRD (FN 166) instruction to read the value of the special data register and specify its word soft component in each operand.

4.14.5 FN 164 - HTOS/Second Conversion of Hour, Minute, and Second Data

Outline

An instruction to convert time/moment data in [hour, minute, second] unit into data in second unit.



Second Conversion of Hour, Minute, and Second Data FN164 - HTOS	Instruction Mark	Execution Condition	Instruction Type	Number of Instruction Steps
	HTOS	Continuous type	16	5
	HTOSP	Pulse type	16	5
	DHTOS	Continuous type	32	9
	DHTOSP	Pulse type	32	9

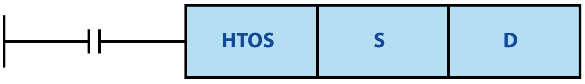
Operand	Setting Data	Data Type
	S: An instruction to convert time/moment data in [hour, minute, second] unit into data in second unit	16 bit
	D: Saving the soft component number of the time/moment data (second) after conversion	16/32 bit

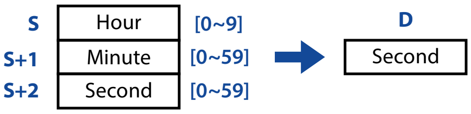
Operand	Operand Object Soft Component																			
	Bit Soft Component							Word Soft Component									Others			
	X	Y	M	T	C	S	D.b	KnX	KnY	KnM	KnS	T	C	D	V, Z	H	K	E	P
S								●	●	●	●	●	●	●	●				
D									●	●	●	●	●	●	●				

Function and Action Description

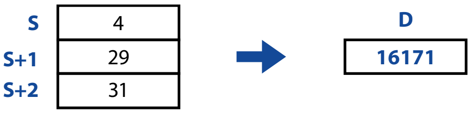
16-bit Operation (HTOS, HTOSP)

After converting the time/moment data (hour, minute, second) of [S,S+1,S+2] into seconds, save the result in D.





For example, specify 4 hours and 29 minutes and 31 seconds, as shown below.



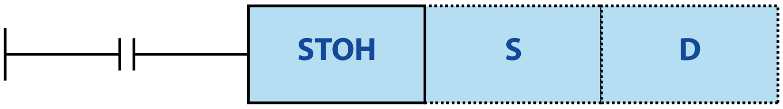
Error

Error	
1	When the data of [S,S+1,S+2] is out of range, an operation error occurs. The error flag bit M8067 is ON, and the error code (K6706) is stored in D8067.

4.14.6 FN 165 - STOH/ [Hour, Minute, Second] Conversion of Second Data

Outline

An instruction to convert time/moment data in second unit into data in [hour, minute, second] unit.



[Hour, Minute, Second] Conversion of Second Data FN165 - STOH	Instruction Mark	Execution Condition	Instruction Type	Number of Instruction Steps
	STOH	Continuous type	16	5
	STOHP	Pulse type	16	5
	DSTOH	Continuous type	32	9
	DSTOHP	Pulse type	32	9

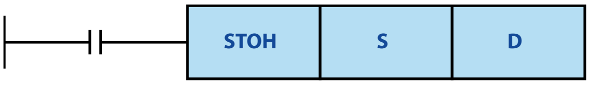
Operand	Setting Data	Data Type
	S: Saving the soft component number BIN 16/32 bit of the time/moment data (second) before conversion	16/32 bit
	D: Saving the soft component start number of the time/moment data (hour, minute, second) after conversion	16 bit

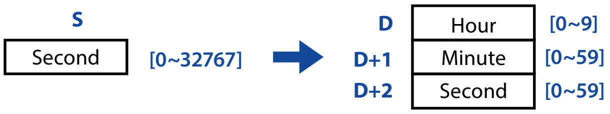
Operand	Operand Object Soft Component																		
	Bit Soft Component							Word Soft Component								Others			
	X	Y	M	T	C	S	D.b	KnX	KnY	KnM	KnS	T	C	D	V, Z	H	K	E	P
S								●	●	●	●	●	●	●	●				
D									●	●	●	●	●	●	●				

Function and Action Description

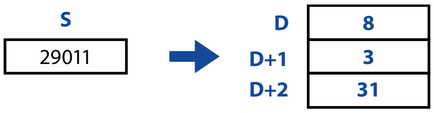
16-bit Operation (STOH, STOHP)

Convert the second data of S into hours, minutes, and seconds, and save the result in [D,D+1,D+2] (hour, minute, second).



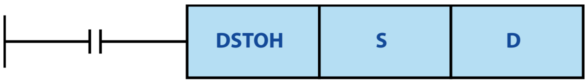


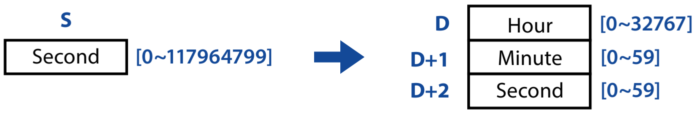
For example, specify 29011 seconds, as shown below.



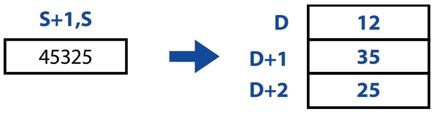
32-bit Operation (DSTOH, DSTOHP)

Convert the second data of [S+1,S] into hours, minutes, and seconds, and the result is saved in 3 points (hour, minute, second) starting from [D,D+1,D+2].





For example, specify 45325s, as shown below.



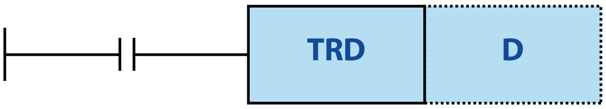
Error

Error	
1	When the data of S is out of range, an operation error occurs. The error flag bit M8067 is ON, and the error code (K6706) is stored in D8067.

4.14.7 FN 166 - TRD/Clock Data Reading

Outline

An instruction to read out the clock data of the built-in real-time clock of the intelligent controller.



Clock Data Reading FN166 - TRD	Instruction Mark	Execution Condition	Instruction Type	Number of Instruction Steps
	TRD	Continuous type	16	3
	TRDP	Pulse type	16	3

Operand	Setting Data															Data Type			
	D: Specify saving the starting soft component number of the readed time data (occupied 7 points)															16 bit			
	Operand Object Soft Component																		
	Bit Soft Component							Word Soft Component								Others			
	X	Y	M	T	C	S	D.b	KnX	KnY	KnM	KnS	T	C	D	V, Z	H	K	E	P
D												●	●	●	●				

Function and Action Description

16-bit Operation (TRD, TRDP)

The clock data (D8013 ~ D8019) of the built-in real-time clock of the intelligent controller is read out to D ~ D+6 according to the following format.

	Soft Component	Item	Clock Data		Soft Component	Item
Special Data Register	D8018	Year (gregorian calendar)	0 ~ 99 (last two digits of the gregorian calendar)	→	D0	Year (gregorian calendar)
	D8017	Month	1 ~ 12	→	D1	Month
	D8016	Day	1 ~ 31	→	D2	Day
	D8015	Hour	0 ~ 23	→	D3	Hour
	D8014	Minute	0 ~ 59	→	D4	Minute
	D8013	Second	0 ~ 59	→	D5	Second
	D8019	Week	0 (Sunday) ~ 6 (Saturday)	→	D6	Week

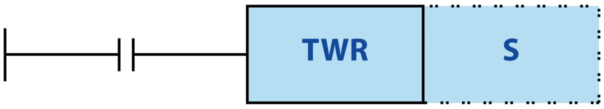
Note

Note		Description
1	Number of occupied points of soft component	D occupies 7 points of soft component. Please be careful not to duplicate the soft component used in other control of the machine.

4.14.8 FN 167 - TWR/Clock Data Writing

Outline

An instruction to write the clock data to the built-in real-time clock of the intelligent controller.



Clock Data Writing FN167 - TWR	Instruction Mark	Execution Condition	Instruction Type	Number of Instruction Steps
	TWR	Continuous type	16 bit	3
	TWRP	Pulse type	16 bit	3

Operand	Setting Data	Data Type
	S: Specify the starting soft component number of the source address of the writed time data (occupied 7 points)	16 bit

Operand	Operand Object Soft Component																			
	Bit Soft Component							Word Soft Component								Others				
	X	Y	M	T	C	S	D.b	KnX	KnY	KnM	KnS	T	C	D	V, Z	H	K	E	P
S												●	●	●	●				

Function and Action Description

16-bit Operation (TWR, TWRP)

Write the setted clock data S ~ S+6 to the clock data (D8013 ~ D8019) of the built-in real-time clock of the intelligent controller.

	Soft Component	Item	Clock Data		Soft Component	Item	
Data Used for Time Setting	D10	Year (gregorian calendar)	0 ~ 99 (last two digits of the gregorian calendar)	→	D8018	Year (gregorian calendar)	Special Data Register
	D11	Month	1 ~ 12	→	D8017	Month	
	D12	Day	1 ~ 31	→	D8016	Day	
	D13	Hour	0 ~ 23	→	D8015	Hour	
	D14	Minute	0 ~ 59	→	D8014	Minute	
	D15	Second	0 ~ 59	→	D8013	Second	
	D16	Week	0 (Sunday) ~ 6 (Saturday)	→	D8019	Week	

- The clock data of the real-time clock is immediately changed after the TWR (FN 167) instruction is executed.
- When using this instruction to set the clock data (time calibration), it is not necessary to control the special auxiliary relay M8015 (time stop and time calibration).
- When the date and time value that cannot be displayed is set, the clock data is not changed.
  In this case, please set the correct clock data and write again.

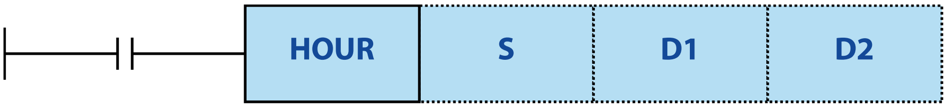
Note

Note		Description
1	Number of occupied points of soft component	Occupy continuous 7 points of soft component starting from S. Please be careful not to duplicate the soft component used in other control of the machine.

4.14.9 FN 169 - HOUR/Timer

Outline

An instruction to accumulate the time when the input contact is continuously ON in 1 hour.



Timer FN169 - HOUR	Instruction Mark	Execution Condition	Instruction Type	Number of Instruction Steps
	HOUR	Continuous type	16 bit	7
	DHOUR	Continuous type	32 bit	13

Operand	Setting Data	Data Type
	S: Time to make D2 ON (set in units of 1 hour)	16/32 bit
	D1: Current value in units of 1 hour (specify the data register for power failure maintenance)	16/32 bit
	D2: Start number of the alarm output	16/32 bit

Operand Object Soft Component	Bit Soft Component							Word Soft Component									Others			
	X	Y	M	T	C	S	D.b	KnX	KnY	KnM	KnS	T	C	D	V, Z	H	K	E	P
S								●	●	●	●	●	●	●	●	●	●		
D1														●	●				
D2		●	●			●	●								●				

Function and Action Description

16-bit Operation (HOUR)

When the cumulative ON time of the instruction input exceeds S, D2 turns ON.

The current value that is less than 1 hour in D1+1 is saved in units of 1 second.

Operand	Description
S	Time until D2 turns ON
D1	Current value in units of 1 hour
D1+1	Current value that is less than 1 hour (in units of 1 second)
D2	Alarm output destination address number When the current value D1 exceeds the specified time of S, it turns ON

- The current value data can be used even after the power supply of the intelligent controller is turned off, so please specify the data register for power failure maintenance in D1. When using a general data register, the current value will be cleared when the power supply of the intelligent controller is turned OFF or STOP→RUN.
- After the alarm output D2 is ON, the measurement can continue.
- Stop measurement when the current value D1 reaches the Max. value of 16 bits.
  To continue measuring, please clear the current value of D1 ~ D1+1.

32-bit Operation (DHOUR)

Operand	Description
S	The time until D2 turns ON, specified by S1+1 (high) and S1 (low)
D1	Current value in units of 1 hour is saved in D1+1 (high bit) and D1 (low bit)
D1+1	Current value that is less than 1 hour (in units of 1 second)
D2	Alarm output destination address number When the current value [D1,D1+1] exceeds the specified time of S, it turns ON

- The current value data can be used even after the power supply of the intelligent controller is turned off, so please specify the data register for power failure maintenance in D1. When using a general data register, the current value will be cleared when the power supply of the intelligent controller is turned OFF or STOP→RUN.
- After the alarm output D2 is ON, the measurement can continue.
- Stop measurement when the current value [D1+1,D1] reaches the Max. value of 32 bits.
  To continue measuring, please clear the current value of D1 ~ D1+2.

Note

Note		Description
1	Number of occupied points of soft component	D1 occupies 2 (16 bit operation) or 3 (32 bit operation) soft components. Please be careful not to duplicate the soft component used in other control of the machine.

4.15 External Device - FN 170 ~ FN 179

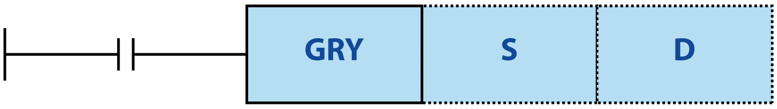
In FN 170 ~ FN 179, the instructions for gray code conversion are provided.

FN No.	Instruction Mark	Instruction Format	Function	Section	Page
170	GRY	GRY (S) (D) GRYP (S) (D) DGRY (S) (D) DGRYP (S) (D)	Gray code conversion	4.15.1	207
171	GBIN	GBIN (S) (D) GBINP (S) (D) DGBIN (S) (D) DGBINP (S) (D)	Gray code inverse conversion	4.15.2	208

4.15.1 FN 170 - GRY/Gray Code Conversion

Outline

An instruction to transfer after converting the BIN value to the Gray code.



	Instruction Mark	Execution Condition	Instruction Type	Number of Instruction Steps
Gray Code Conversion FN170 - GRY	GRY	Continuous type	16 bit	5
	GRYP	Pulse type	16 bit	5
	DGRY	Continuous type	32 bit	9
	DGRYP	Pulse type	32 bit	9

Operand	Setting Data															Data Type			
	S: Converting source data, or saving the word soft components that convert source data															16/32 bit			
	D: Saving the word soft components of the converted data															16/32 bit			
	Operand Object Soft Component																		
	Bit Soft Component							Word Soft Component								Others			
	X	Y	M	T	C	S	D.b	KnX	KnY	KnM	KnS	T	C	D	V, Z	H	K	E	P
S								●	●	●	●	●	●	●	●	●	●		
D									●	●	●	●	●	●	●				

Function and Action Description

16-bit Operation (GRY, GRYP)

When S is K1234, D is K3Y10.

- For S, the following range of value is valid: 0 ~ 32,767.

	b15															b00
BIN 1234	0	0	0	0	0	1	0	0	1	1	0	1	0	0	1	0
					Y23			Y20	Y17							Y10
GRY 1234					0	1	1	0	1	0	1	1	1	0	1	1

32-bit Operation (DGRY, DGRYP)

Up to 32 bit gray code conversion can be performed.

- S is valid from 0 to 2,147,483,647.

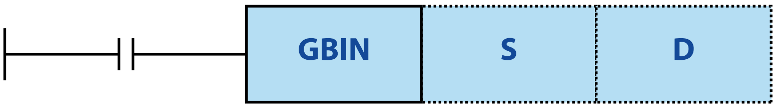
Note

Note	
1	The conversion speed of the data depends on the scan time of the intelligent controller.

4.15.2 FN 171 - GBIN/Gray Code Inverse Conversion

Outline

An instruction to transfer after converting the Gray code to the BIN value.



Gray Code Inverse Conversion FN171 - GBIN	Instruction Mark	Execution Condition	Instruction Type	Number of Instruction Steps
	GBIN	Continuous type	16 bit	5
	GBINP	Pulse type	16 bit	5
	DGBIN	Continuous type	32 bit	9
	DGBINP	Pulse type	32 bit	9

Operand	Setting Data	Data Type
	S: Saving the word soft components that convert source data	16/32 bit
	D: Saving the word soft components of the converted data	16/32 bit

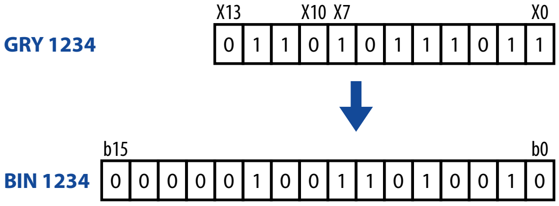
Operand Object Soft Component	Bit Soft Component							Word Soft Component								Others			
	X	Y	M	T	C	S	D.b	KnX	KnY	KnM	KnS	T	C	D	V, Z	H	K	E	P
S								●	●	●	●	●	●	●	●	●	●		
D									●	●	●	●	●	●	●				

Function and Action Description

16-bit Operation (GBIN, GBINP)

When S is K3X000, D is 10.

- S = 0 ~ 32,767 is valid.



GRY 1234					0	1	1	0	1	0	1	1	1	0	1	1
BIN 1234	0	0	0	0	0	1	0	0	1	1	0	1	0	0	1	0

32-bit Operation (DGBIN, DGBINP)

Up to 32 bit BIN conversion can be performed.

- S = 0 ~ 2,147,483,647 is valid.

Note

Note	
1	When the input relay (X) is specified in S, the response delay is [intelligent controller scan time + input filter constant]. By executing the REFF (FN 51) instruction or D8020 (filter adjustment), the input filter value of normal input terminal can be converted to remove the delay of the filter constant part.

4.16 Other Instructions - FN184 ~ FN 189

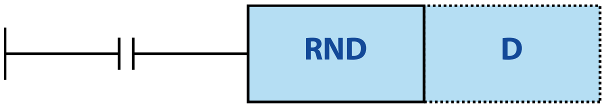
In FN 184 ~ FN 189, data processing instructions for generation of random numbers, CRC data operations, and high-speed counter operations are provided.

FN No.	Instruction Mark	Instruction Format	Function	Section	Page
184	RND	RND (D) RNDP (D)	Generation of random numbers	4.16.1	210
186	DUTY	DUTY (n1) (n2) (D)	Generation of timing pulse	4.16.2	211
188	CRC	CRC (S) (D) (n) CRCP (S) (D) (n)	CRC operation	4.16.3	213

4.16.1 FN 184 - RND/Generation of Random Numbers

Outline

An instruction to generate random numbers.



Generation of Random Numbers FN184 - RND	Instruction Mark	Execution Condition	Instruction Type	Number of Instruction Steps
	RND	Continuous type	16 bit	3
	RNDP	Pulse type	16 bit	3

Operand	Setting Data														Data Type				
	D: Saving the soft component number that generates the random number														16 bit				
	Operand Object Soft Component																		
	Bit Soft Component							Word Soft Component								Others			
	X	Y	M	T	C	S	D.b	KnX	KnY	KnM	KnS	T	C	D	V, Z	H	K	E	P
D									●	●	●	●	●	●	●				

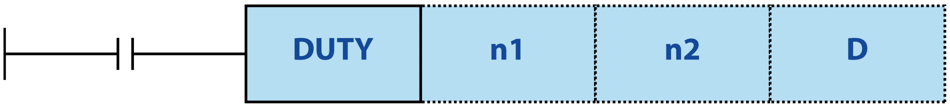
Function and Action Description

16-bit Operation (RND, RNDP)
This instruction generates a 16-bit pseudo-random number instruction through a pseudo-random number seed (D8311, D8310). • When use it, only need to turn on the condition, and each cycle will generate a 16-bit random number. • This instruction generates a pseudo-random number from 0 ~ 32,767, and stores its value as a random number in D. The random number seed is also updated to ensure that different random numbers are produced during the next run. • (D8311, D8310) as the initial value is 1, it is recommended to write a non-negative value (0 ~ 2,147,483,647) to this address when STOP→RUN. The time data can be written to ensure that the random number generated by each power-on is different.

4.16.2 FN 186 - DUTY/Generation of Timing Pulse

Outline

An instruction to generate a timing signal by taking the operation cycle of the specified number of times as one cycle.



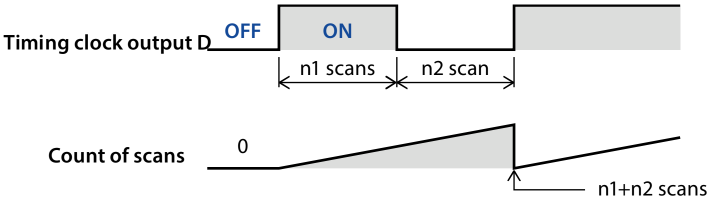
Generation of Timing Clock FN186 - DUTY	Instruction Mark	Execution Condition	Instruction Type	Number of Instruction Steps
	DUTY	Continuous type	16 bit	7

<table>
<tr><td rowspan="7">Operand</td><td colspan="15">Setting Data</td><td colspan="4">Data Type</td></tr>
<tr><td colspan="15">n1: ON scan count (operation cycle) [n1 > 0]</td><td colspan="4">16 bit</td></tr>
<tr><td colspan="15">n2: OFF scan count (operation cycle) [n2 > 0]</td><td colspan="4">16 bit</td></tr>
<tr><td colspan="15">D: Destination address of the timing clock output</td><td colspan="4">Bit</td></tr>
<tr><td colspan="19">Operand Object Soft Component</td></tr>
<tr><td colspan="7">Bit Soft Component</td><td colspan="8">Word Soft Component</td><td colspan="4">Others</td></tr>
<tr><td>X</td><td>Y</td><td>M</td><td>T</td><td>C</td><td>S</td><td>D.b</td><td>KnX</td><td>KnY</td><td>KnM</td><td>KnS</td><td>T</td><td>C</td><td>D</td><td>V, Z</td><td>H</td><td>K</td><td>E</td><td>P</td></tr>
<tr><td>n1</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td>●</td><td>●</td><td>●</td><td></td><td>●</td><td>●</td><td></td><td></td></tr>
<tr><td>n2</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td>●</td><td>●</td><td>●</td><td></td><td>●</td><td>●</td><td></td><td></td></tr>
<tr><td>D</td><td></td><td></td><td>▲</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td>●</td><td></td><td></td><td></td><td></td></tr>
<tr><td></td><td colspan="19">▲: Please specify M8330 ~ M8334</td></tr>
</table>

Function and Action Description

16-bit Operation (DUTY)

The timing clock output D is ON/OFF in such a manner that n1 scans are ON and n2 scans are OFF.



- Specify M8330 ~ M8334 in the destination address D of the timing clock output.
- The count value of the number of scans corresponding to the destination address D of the timing clock output is saved in D8330 ~ D8334.
  The count value of the number of scans D8330 ~ D8334 is reset when the count value becomes n1 n2, or when the instruction input (instruciton) turns ON.

Destination Address D of Timing Clock Output	Soft Component for Counting the Number of Scans
M8330	D8330
M8331	D8331
M8332	D8332
M8333	D8333
M8334	D8334

- Start at the rising edge of the instruction input. At the END instruction, the D turns ON/OFF the timing clock output. In addition, the instruction input does not stop even if it is cut off. STOP is realized by interruption or power failure.
- When n1 and n2 are set to 0, as shown in the table below.

Status of n1, n2	ON/OFF Status of D
n1 = 0, n2 ≥ 0	D is fixed to OFF
n1 > 0, n2 = 0	D is fixed to ON

Related Soft Components

Soft Component	Name	Content
M8330	Timing clock output 1	Timing clock output of the instruction DUTY (FN 186)
M8331	Timing clock output 2	
M8332	Timing clock output 3	
M8333	Timing clock output 4	
M8334	Timing clock output 5	
D8330	Scan count of timing clock output 1	Count value of the number of scans used by the timing clock output 1 of the instruction DUTY (FN 186)
D8331	Scan count of timing clock output 2	Count value of the number of scans used by the timing clock output 2 of the instruction DUTY (FN 186)
D8332	Scan count of timing clock output 3	Count value of the number of scans used by the timing clock output 3 of the instruction DUTY (FN 186)
D8333	Scan count of timing clock output 4	Count value of the number of scans used by the timing clock output 4 of the instruction DUTY (FN 186)
D8334	Scan count of timing clock output 5	Count value of the number of scans used by the timing clock output 5 of the instruction DUTY (FN 186)

Note

Note	
1	This instruction can be used 5 times (point). However, the same timing clock output destination address D cannot be used in multiple DUTY (FN 186) instructions.

Error

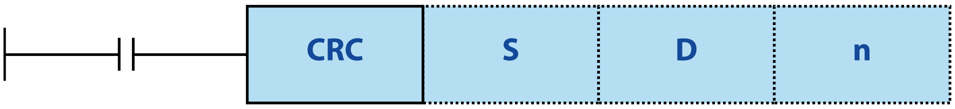
Error	
1	Operation errors may occur in the following cases. The error flag bit M8067 turns ON and the error code is stored in D8067. • When n1 and n2 are not full (error code: K6706). • D is beyond the range of M8330 ~ M8334 (error code: K6705).

4.16.3 FN 188 - CRC/CRC Operation

Outline

This instruction can be used to calculate the CRC value (Cyclic Redundancy Check). In this instruction, CRC-16 ([$X^{16} + X^{15} + X^{2} + 1$] generator polynomial) is used to calculate the CRC.

In addition, besides CRC, there are parity check and sum check (checksum) in error checking methods. CCD instruction (FN 84) can be used when calculating horizontal check value.



CRC Operation FN188 - CRC	Instruction Mark	Execution Condition	Instruction Type	Number of Instruction Steps
	CRC	Continuous type	16	7
	CRCP	Pulse type	16	7

Operand	Setting Data	Data Type
	S: Saving the soft component start number of the data that is the CRC value generation object	16 bit
	D: Saving the soft component number of the generated CRC value	16 bit
	n: Calculating the number of 8-bit data (byte) of the CRC value, or saving the soft component number of the number of data	16 bit

Operand Object Soft Component	Bit Soft Component							Word Soft Component								Others			
	X	Y	M	T	C	S	D.b	KnX	KnY	KnM	KnS	T	C	D	V, Z	H	K	E	P
S								▲	▲	▲	▲	●	●	●	●				
D									▲	▲	▲	●	●	●	●				
n														●		●	●		

▲: When specifying the number of digits of the bit soft component, please be sure to specify 4 digits (K4□○○○)

Function and Action Description

16-bit Operation (CRC, CRCP)

The n-point 8-bit data (byte unit) starting with the soft component specified in S, and generating the CRC value and saving it to D. There are 8-bit and 16-bit conversion modes in this instruction, switch the conversion mode according to M8161 ON/OFF.

16-bit Conversion Mode [M8161 = OFF]

- The high 8 bits (bytes) and low 8 bits (bytes) of the soft component S are operated in 16-bit mode.
- Save the operation result in 16 bits of the 1 soft component specified by D.

			For Example: S = D100, D = D0, n = 6		
			Soft Component	Content of Object Data	
				8 Bit	16 Bit
Save the address of the object data that generated the CRC value	S	Low byte	D100 low byte	01H	0301H
		High byte	D100 high byte	03H	
	S+1	Low byte	D101 low byte	03H	0203H
		High byte	D101 high byte	02H	
	S+2	Low byte	D102 low byte	00H	1400H
		High byte	D102 high byte	14H	
	/		-		
	S + n/2-1	Low byte	-		
Save the address of CRC value	D	Low byte	D100 low byte	E4H	41E4H
		High byte	D100 high byte	41H	

16-bit Operation (CRC, CRCP)

8-bit Conversion Mode [M8161 = ON]

- Only the lower 8 bits (bytes) of the soft component S are operated in 8-bit conversion mode.
- The operation results are saved in 2 soft components specified by D, the low 8 bits (bytes) in D, and the high 8 bits (bytes) in D+1.

			For Example: S = D100, D = D0, n = 6	
			Soft Component	Content of Object Data
Save the address of the object data that generated the CRC value	S	Low byte	D100 low byte	01H
	S+1	Low byte	D101 low byte	03H
	S+2	Low byte	D102 low byte	03H
	S+3	Low byte	D103 low byte	02H
	S+4	Low byte	D104 low byte	00H
	S+5	Low byte	D105 low byte	14H
	/		-	
	S+n-1	Low byte	-	
Save the address of CRC value	D	Low byte	D0 low byte	E4H
	D+1	Low byte	D1 low byte	41H

Related Soft Components

Related Soft Component	Content
M8161	ON: CRC instruction operates in 8-bit mode
	OFF: CRC instruction operates in 16-bit mode
	Clear when RUN→STOP

Note

Note	
1	In this instruction, the generator polynomial [$X^{16}+X^{15}+X^{2}+1$] of the CRC value (CRC-16) is used. In addition, there are various standardized generator polynomials for the CRC values. Note that if different generator polynomials are used, a completely different CRC value will result.
2	The intelligent controller's own Modbus communication (ADPRW) and CAN communication (EXTR) have their own data check, no need to add check by the user.

Error

<table>
<tr><th colspan="2">Error</th></tr>
<tr><td>1</td><td>Operation errors may occur in the following cases. The error flag bit M8067 turns ON and the error code (K6706) is stored in D8067.
• The number of bits of the bit soft component used in S and D specifies a value other than 4 digits.
• n is beyond the specified range (1 ~ 256).
• S+n-1 and D+1 are beyond the range of soft component.
<table>
<tr><th>Name</th><th>Generator Polynomial</th></tr>
<tr><td>CRC-12</td><td>$X^{12}+X^{11}+X^{3}+X^{2}+X+1$</td></tr>
<tr><td>CRC-16</td><td>$X^{16}+X^{15}+X^{2}+1$</td></tr>
<tr><td>CRC-32</td><td>$X^{32}+X^{26}+X^{23}+X^{22}+X^{16}+X^{12}+X^{11}+X^{10}+X^{8}+X^{7}+X^{5}+X^{4}+X^{2}+X+1$</td></tr>
<tr><td>CRC-CCITT</td><td>$X^{16}+X^{12}+X^{5}+1$</td></tr>
</table>
</td></tr>
</table>

4.17 Data Block Processing - FN 190 ~ FN 199

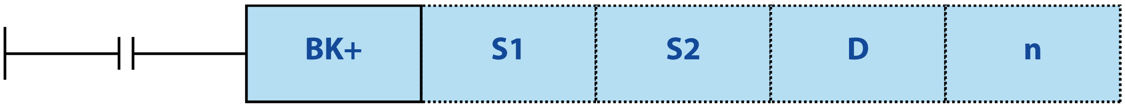
In FN 190 ~ FN 199, instructions for performing addition, subtraction, and comparison of data blocks are provided.

FN No.	Instruction Mark	Instruction Format	Function	Section	Page
192	BK+	BK+ (S1) (S2) (D) (n) BK+P (S1) (S2) (D) (n) DBK+ (S1) (S2) (D) (n) DBK+P (S1) (S2) (D) (n)	Data block addition	4.17.1	216
193	BK-	BK- (S1) (S2) (D) (n) BK-P (S1) (S2) (D) (n) DBK- (S1) (S2) (D) (n) DBK-P (S1) (S2) (D) (n)	Data block subtraction	4.17.2	218
194	BKCMP=	BKCMP= (S1) (S2) (D) (n) BKCMP= P (S1) (S2) (D) (n) DBKCMP= (S1) (S2) (D) (n) DBKCMP= P (S1) (S2) (D) (n)	Data block comparison S1 = S2	4.17.3	220
195	BKCMP>	BKCMP> (S1) (S2) (D) (n) BKCMP> P (S1) (S2) (D) (n) DBKCMP> (S1) (S2) (D) (n) DBKCMP> P (S1) (S2) (D) (n)	Data block comparison S1 > S2	4.17.3	220
196	BKCMP<	BKCMP< (S1) (S2) (D) (n) BKCMP< P (S1) (S2) (D) (n) DBKCMP< (S1) (S2) (D) (n) DBKCMP< P (S1) (S2) (D) (n)	Data block comparison S1 < S2	4.17.3	220
197	BKCMP<>	BKCMP<> (S1) (S2) (D) (n) BKCMP<> P (S1) (S2) (D) (n) DBKCMP<> (S1) (S2) (D) (n) DBKCMP<>P (S1) (S2) (D) (n)	Data block comparison S1 ≠ S2	4.17.3	220
198	BKCMP<=	BKCMP<= (S1) (S2) (D) (n) BKCMP<= P (S1) (S2) (D) (n) DBKCMP<= (S1) (S2) (D) (n) DBKCMP<= P (S1) (S2) (D) (n)	Data block comparison S1 ≤ S2	4.17.3	220
199	BKCMP>=	BKCMP>= (S1) (S2) (D) (n) BKCMP>= P (S1) (S2) (D) (n) DBKCMP>= (S1) (S2) (D) (n) DBKCMP>= P (S1) (S2) (D) (n)	Data block comparison S1 ≥ S2	4.17.3	220

4.17.1 FN 192 - BK+/Data Block Addition

Outline

An instruction to perform data block BIN addition operation.



Data Block Addition FN192 - BK+	Instruction Mark	Execution Condition	Instruction Type	Number of Instruction Steps
	BK+	Continuous type	16 bit	9
	BK+P	Continuous type	16 bit	9
	DBK+	Pulse type	32 bit	17
	DBK+P	Pulse type	32 bit	17

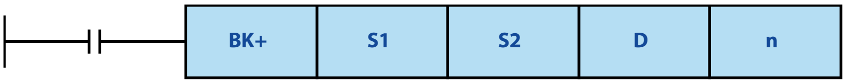
Operand	Setting Data	Data Type
	S1: Saving the soft component start number of the data that performs the addition operation	16/32 bit
	S2: A constant for performing the addition operation, or saving the soft component start number of the data that performs the addition operation	16/32 bit
	D: Saving the soft component start number of the operation result	16/32 bit
	n: The number of data	16/32 bit

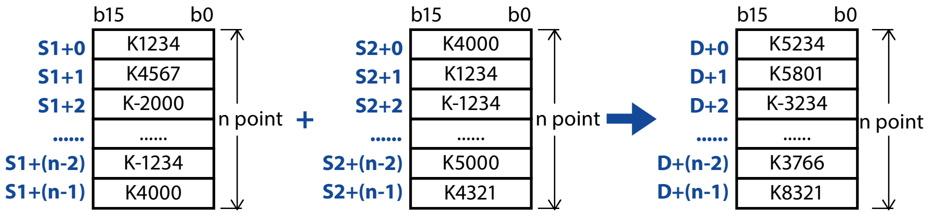
Operand Object Soft Component	Bit Soft Component							Word Soft Component								Others			
	X	Y	M	T	C	S	D.b	KnX	KnY	KnM	KnS	T	C	D	V, Z	H	K	E	P
S1												●	●	●	●				
S2												●	●	●	●	●	●		
D												●	●	●	●				
n														●		●	●		

Function and Action Description

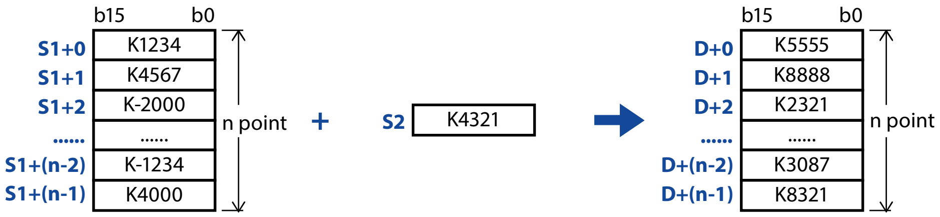
16-bit Operation (BK+, BK+P)

After adding the n-point 16-bit data starting from S1 and the n-point 16-bit data (BIN) starting from S2, the operation result is saved in the n point starting from D.



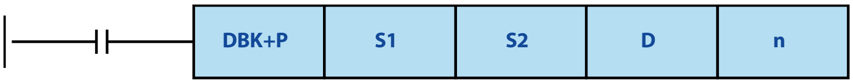


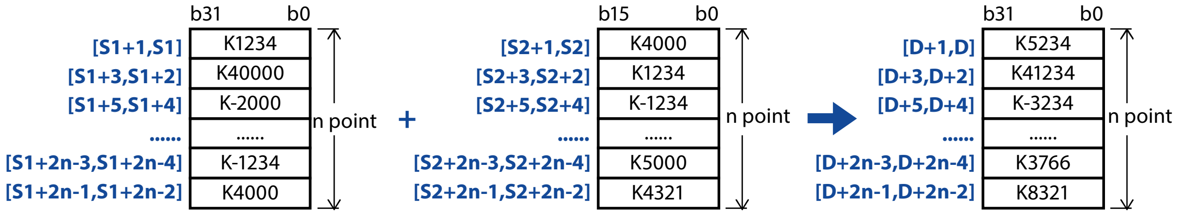
You can directly specify a constant of -32768 ~ +32767 (16 bit) in S2.



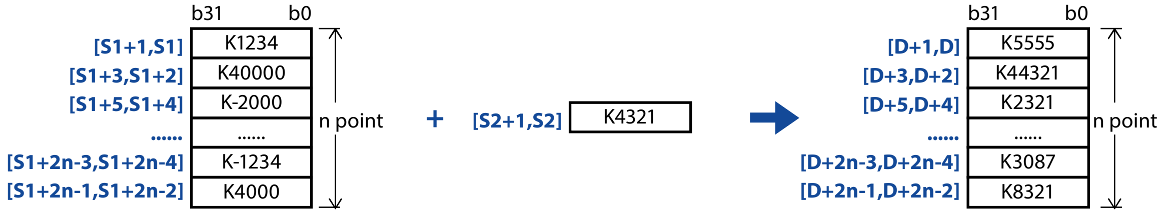
32-bit Operation (DBK+, DBK+P)

After adding the 2n point 32-bit data starting from [S1+1,S1] and 2n point 32-bit data (BIN) starting from [S2+1,S2], the operation result is saved in the 2n point starting from [D+1,D].





You can directly specify a constant of -2,147,483,648 ~ +2,147,483,647 (32 bit) in [S2+1,S2].



Note

Note	
1	When an underflow or overflow occurs in the operation result, as shown below. At this time, the carry flag bit is not turned ON. • 16-bit operation: • K32767 (H7FFF) + K2 (H0002)→K-32767 (H8001) • K-32768 (H8000) + K-2 (HFFFE)→K32766 (H7FFE) • 32-bit operation: • K2,147,483,647 (H7FFFFFFF) + K2 (H00000002)→K-2,147,483,647 (H80000001) • K-2,147,483,648 (H80000000) + K-2 (HFFFFFFFE)→K2,147,483,646 (H7FFFFFFE) • When D and R are specified as n for a 32-bit instruction, please note that the 32-bit value of [n+1,n] will take effect. When DBK + D0 D100 D200 R0, n = [R,R0].

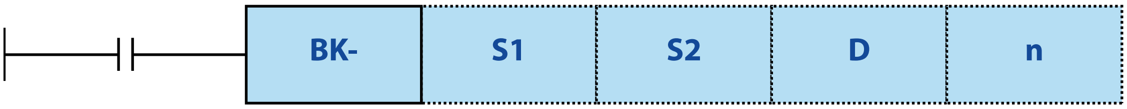
Error

Error	
1	Operation errors may occur in the following cases. The error flag bit M8067 turns ON and the error code (K6706) is stored in D8067. • The n-point (2n point for 32-bit operation) soft component starting from S1, S2, and D is beyond the range of corresponding soft component. • The n-point (2n point for 32-bit operation) soft component starting from S1 and the n-point soft component starting from D are repeated. • The n-point (2n point for 32-bit operation) soft component starting from S2 and the n-point soft component starting from D are repeated.

4.17.2 FN 193 - BK-/Data Block Subtraction

Outline

An instruction to perform data block BIN subtraction operation.



Data Block Subtraction FN193 - BK-	Instruction Mark	Execution Condition	Instruction Type	Number of Instruction Steps
	BK-	Continuous type	16 bit	9
	BK-P	Pulse type	16 bit	9
	DBK-	Continuous type	32 bit	17
	DBK-P	Pulse type	32 bit	17

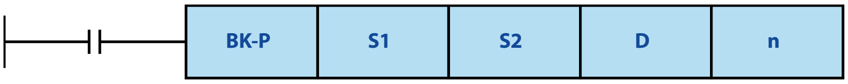
Operand	Setting Data	Data Type
	S1: Saving the soft component start number of the data that performs the subtraction operation	16/32 bit
	S2: A constant for performing the subtraction operation, or saving the soft component start number of the data that performs the subtraction operation	16/32 bit
	D: Saving the soft component start number of the operation result	16/32 bit
	n: The number of data	16/32 bit

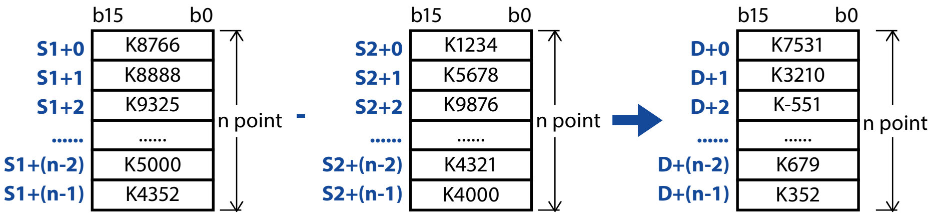
Operand Object Soft Component	Bit Soft Component							Word Soft Component								Others			
	X	Y	M	T	C	S	D.b	KnX	KnY	KnM	KnS	T	C	D	V, Z	H	K	E	P
S1												●	●	●	●				
S2												●	●	●	●	●	●		
D												●	●	●	●				
n														●		●	●		

Function and Action Description

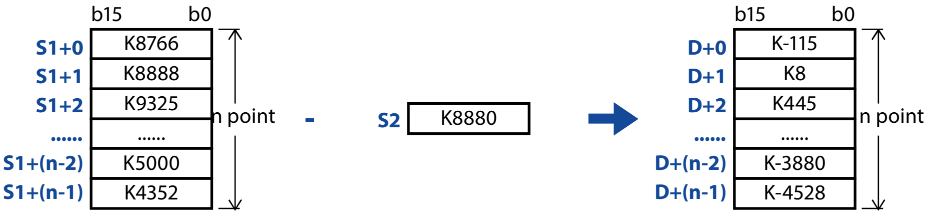
16-bit Operation (BK-, BK-P)

After subtracting the n-point 16-bit data starting from S1 and the n-point 16-bit data (BIN) starting from S2, the operation result is saved in the n point starting from D.

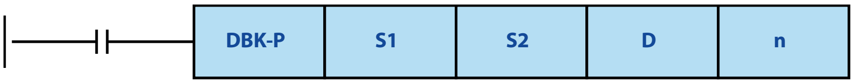


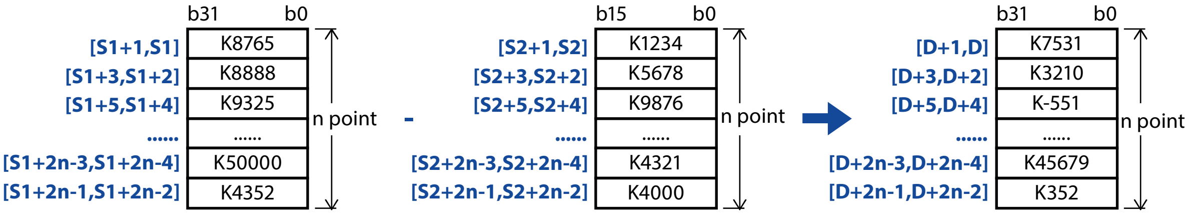


You can directly specify a constant of -32768 ~ +32767 (16 bit) in S2.

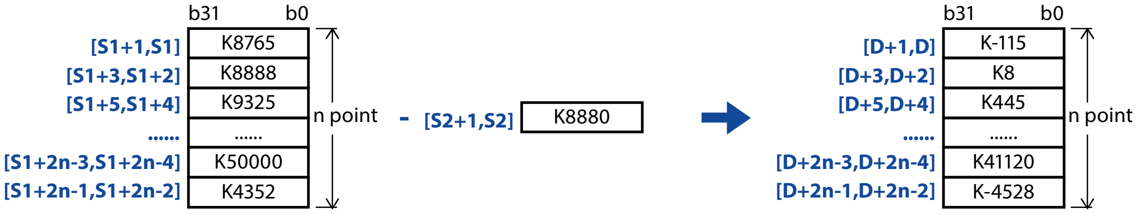


32-bit Operation (DBK-, DBK-P)
After subtracting the 2n point 32-bit data starting from [S1+1,S1] and 2n point 32-bit data (BIN) starting from [S2+1,S2], the operation result is saved in the 2n point starting from [D+1,D].





You can directly specify a constant of -2,147,483,648 ~ +2,147,483,647 (32 bit) in [S2+1,S2].



Note

Note	
1	When an underflow or overflow occurs in the operation result, as shown below. At this time, the carry flag bit is not turned ON. • 16-bit operation: • K-32768 (H8000) - K2 (H0002)→K32766 (H7FFE) • K32,767 (H7FFF) - K-2 (HFFFE)→K-32,767 (H8001) • 32-bit operation: • K-2,147,483,648 (H80000000) - K2 (H00000002)→K2,147,483,646 (H7FFFFFFE) • K2,147,483,647 (H7FFFFFFF) - K-2 (HFFFFFFFE)→K-2,147,483,647 (H80000001) • When D and R are specified as n for a 32-bit instruction, please note that the 32-bit value of [n+1,n] will take effect. When DBK-D0 D100 D200 R0, n = [R,R0].

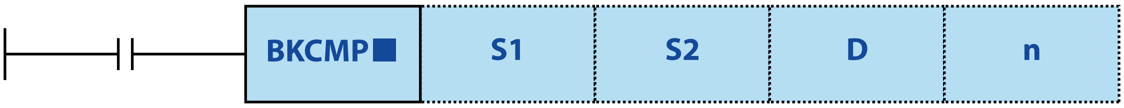
Error

Error	
1	Operation errors may occur in the following cases. The error flag bit M8067 turns ON and the error code (K6706) is stored in D8067. • The n-point (2n point for 32-bit operation) soft component starting from S1, S2, and D is beyond the range of corresponding soft component. • The n-point (2n point for 32-bit operation) soft component starting from S1 and the n-point soft component starting from D are repeated. • The n-point (2n point for 32-bit operation) soft component starting from S2 and the n-point soft component starting from D are repeated.

4.17.3 FN 194 ~ 199-BKCMP =, >, <, <>, <=, >=/Data Block Comparison

Outline

An instruction to compare the data block according to the comparison conditions of each instruction.



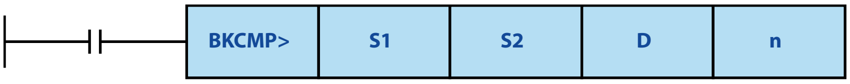
Data Block Comparison FN 194 - BKCMP= 195 - BKCMP> 196 - BKCMP< 197 - BKCMP<> 198 - BKCMP<= 199 - BKCMP>=	Instruction Mark	Execution Condition	Instruction Type	Number of Instruction Steps
	BKCMP■	Continuous type	16 bit	9
	BKCMP■P	Pulse type	16 bit	9
	DBKCMP■	Continuous type	32 bit	17
	DBKCMP■P	Pulse type	32 bit	17
	■: Comparison conditions = , >, <, <>, ≤, ≥			

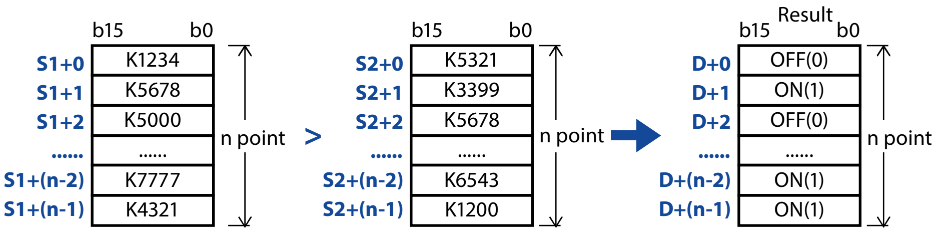
Operand	Setting Data																		Data Type
	S1: Comparison value or saving the soft component number of the comparison value																		16/32 bit
	S2: Saving the soft component start number of the comparison source data																		16/32 bit
	D: Saving the soft component start number of the comparison result																		Bit
	n: Number of data to compare																		16/32 bit
	Operand Object Soft Component																		
	Bit Soft Component							Word Soft Component									Others		
	X	Y	M	T	C	S	D.b	KnX	KnY	KnM	KnS	T	C	D	V, Z	H	K	E	P
S1												●	●	●	●	●	●		
S2												●	●	●	●				
D		●	●			●	●								●				
n														●		●	●		

Function and Action Description

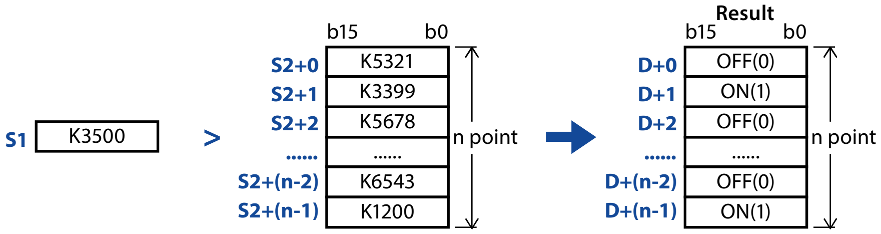
16-bit Operation (BKCMP■, BKCMP■P)

After comparing the n-point 16-bit data (BIN) starting from S1 with the n-point 16-bit data (BIN) starting from S2, the comparison result is saved in the n point start from D.





The constant can be specified directly in S1.



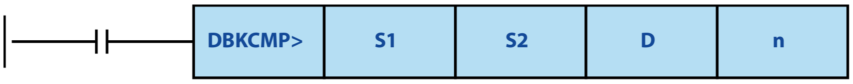
16-bit Operation (BKCMP■, BKCMP■P)

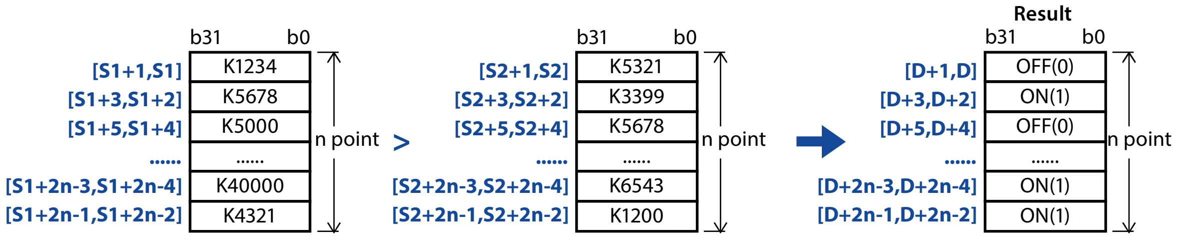
The comparison results of each instruction are as follows.

Instruction	Condition of Comparison Result ON (1)	Condition of Comparison Result OFF (0)
BKCMP = (FN 194)	S1 = S2	S1 ≠ S2
BKCMP> (FN 195)	S1 > S2	S1 ≤ S2
BKCMP< (FN 196)	S1 <S 2	S1 ≥ S2
BKCMP<> (FN 197)	S1 ≠ S2	S1 = S2
BKCMP<= (FN 198)	S1 ≤ S2	S1 > S2
BKCMP>= (FN 199)	S1 ≥ S2	S1 < S2

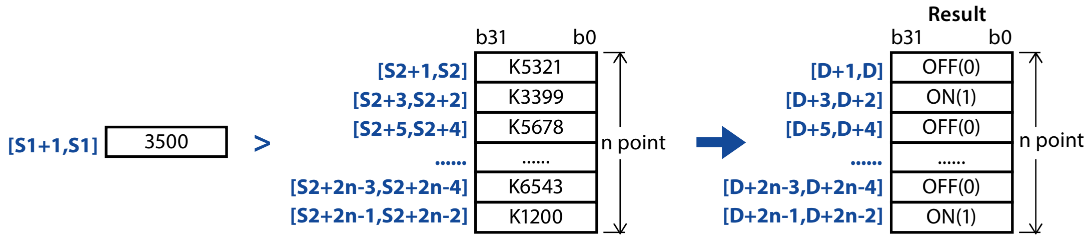
32-bit Operation (DKCMP■, DKCMP■P)

After comparing the n-point 32-bit data (BIN) starting from [S1+1,S1] with the n-point 32-bit data (BIN) starting from [S2+1,S2], the comparison result is saved in the n point starting from [D+1,D].





You can specify a constant directly in [S1+1,S1].



The comparison results of each instruction are as follows.

Instruction	Condition of Comparison Result ON (1)	Condition of Comparison Result OFF (0)
DKCMP= (FN 194)	[S1+1,S1] = [S2+1,S2]	[S1+1,S1] ≠ [S2+1,S2]
DKCMP> (FN 195)	[S1+1,S1] > [S2+1,S2]	[S1+1,S1] ≤ [S2+1,S2]
DKCMP< (FN 196)	[S1+1,S1] < [S2+1,S2]	[S1+1,S1] ≥ [S2+1,S2]
DKCMP<> (FN 197)	[S1+1,S1] ≠ [S2+1,S2]	[S1+1,S1] = [S2+1,S2]
DKCMP<= (FN 198)	[S1+1,S1] ≤ [S2+1,S2]	[S1+1,S1] > [S2+1,S2]
DKCMP>= (FN 199)	[S1+1,S1] ≥ [S2+1,S2]	[S1+1,S1] < [S2+1,S2]

Note

Note		Description
1	When using a 32-bit counter (including a high-speed counter)	Comparison of 32-bit counters (C200 ~ C255) must be compared under 32-bit operation (DBKCMP=, DBKCMP>, DBKCMP<, etc.). If specified under 16-bit operation (BKCMP=, BKCMP>, BKCMP<, etc.), an operation error occurs (error code: K6705)
2	Specify D as the n of the 32-bit instruction	Please note that the 32-bit value of [n+1,n] will take effect.

Error

Error	
1	Operation errors may occur in the following cases. The error flag bit M8067 turns ON and the error code is stored in D8067. • The n-point (2n point for 32-bit operation) soft component starting from S1, S2 is beyond the range of corresponding soft component (error code: K6706). • The n-point soft component starting from D is beyond the range of corresponding soft component (error code: K6706). • When "D" is specified, the data register of D and the n-point soft component starting from S1 (2n point for 32-bit operation) are repeated (error code: K6706). • When "D" is specified, the data register of D and the n-point soft component starting from S2 (2n point for 32-bit operation) are repeated (error code: K6706). • In 16-bit operation, when 32-bit counter (C200 ~ C255) is specified in S1 and S2 (error code: K6705). Use the 32-bit operation instructions (DBKCMP=, DBKCMP>, DBKCMP<, etc.) to compare the 32-bit counters.

4.18 Data Processing 3 - FN 210 ~ FN 219

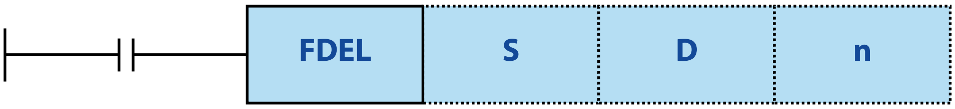
Instructions for reading the last-in data and controling the left and right shift with carry are provided.

FN No.	Instruction Mark	Instruction Format	Function	Section	Page
210	FDEL	FDEL (S) (D) (n) FDELP (S) (D) (n)	Data deletion of data table	4.18.1	224
211	FINS	FINS (S) (D) (n) FINSP (S) (D) (n)	Data insertion of data table	4.18.2	225
212	POP	POP (S) (D) (n) POPP (S) (D) (n)	Read the last-in data [for first-in, last-out control]	4.18.3	226
213	SFR	SFR (D) (n) SFRP (D) (n)	N bit right shift (with carry) of 16-bit data	4.18.4	228
214	SFL	SFL (D) (n) SFLP (D) (n)	N bit left shift (with carry) of 16-bit data	4.18.5	229

4.18.1 FN 210 - FDEL/Data Deletion of Data Table

Outline

An instruction to delete any data in a data table.



Data Deletion of Data Table FN210 - FDEL	Instruction Mark	Execution Condition	Instruction Type	Number of Instruction Steps
	FDEL	Continuous type	16 bit	7
	FDELP	Pulse type	16 bit	7

Operand	Setting Data	Data Type
	S: Saving the soft component number of the deleted data	16 bit
	D: Starting soft component number of the data table	16 bit
	n: The table position of the data to be deleted	16 bit

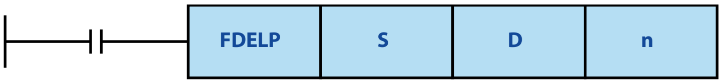
Operand Object Soft Component	Bit Soft Component							Word Soft Component									Others			
	X	Y	M	T	C	S	D.b	KnX	KnY	KnM	KnS	T	C	D	V, Z	H	K	E	P
S												●	●	●	●				
D												●	●	●	●				
n														●		●	●		

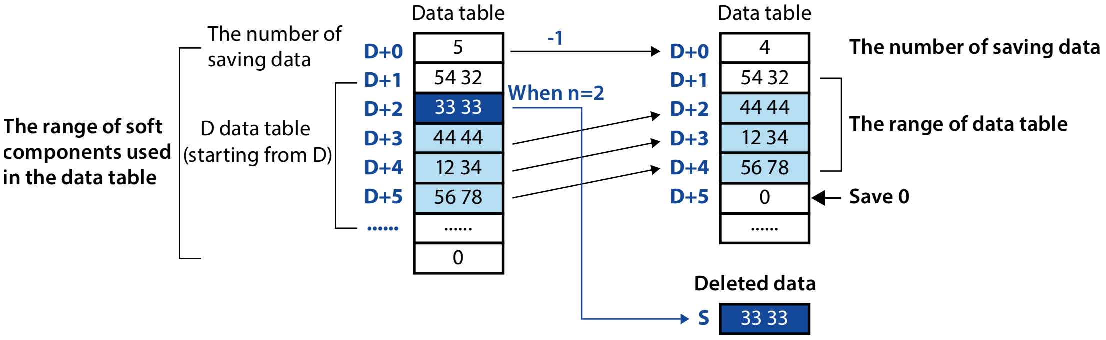
Function and Action Description

16-bit Operation (FDEL, FDELP)

Delete the nth data of the data table (start) and save the deleted data to S.

The data starting from the n+1th of the data table is moved forward one by one, and the number of savingdata is reduced by 1.





Note

Note	
1	The users need to manage the range of soft components used in the data table themselves. The range of the data table is D starting from the next soft component (D+1) of saving data D.

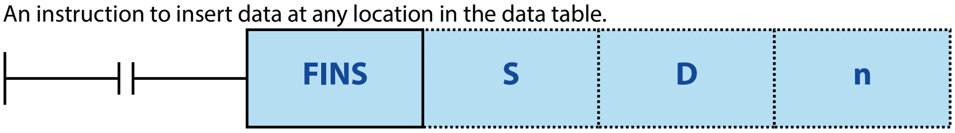
Error

Error	
1	Operation errors may occur in the following cases. The error flag bit M8067 turns ON, and the error code (K6706) is stored in D8067. • The table position n of the data to be deleted is larger than the number of saving data. • The value of n is beyond the soft component range of the data table. • The instruction is executed in the case of n ≤ 0. • The value of the number of the savingdata is 0. • The range of the data table is beyond the range of the corresponding soft component.

4.18.2 FN 211 - FINS/Data Insertion of Data Table

Outline

An instruction to insert data at any location in the data table.



Data Insertion of Data Table FN211 - FINS	Instruction Mark	Execution Condition	Instruction Type	Number of Instruction Steps
	FINS	Continuous type	16 bit	7
	FINSP	Pulse type	16 bit	7

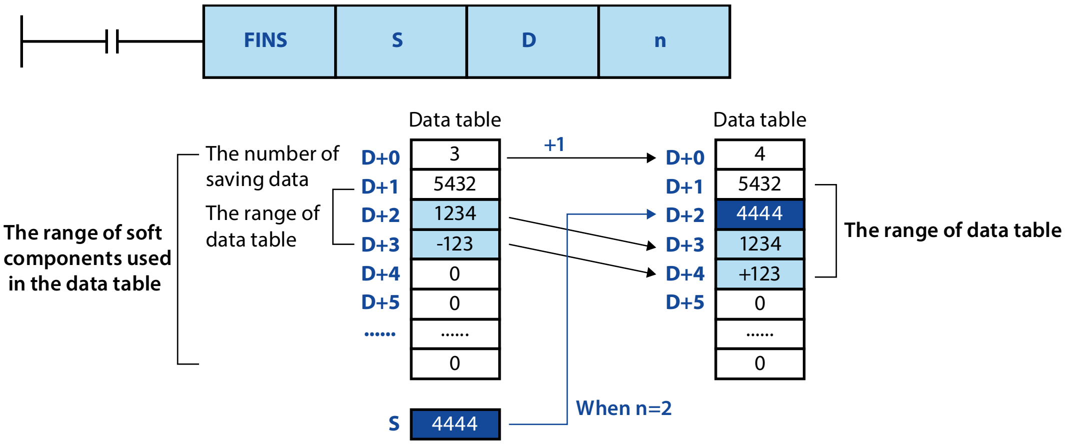
Operand	Setting Data	Data Type
	S: Saving the soft component number of the inserted data	16 bit
	D: Starting soft component number of the data table	16 bit
	n: The table position of the data to be inserted	16 bit

Operand Object Soft Component	Bit Soft Component							Word Soft Component								Others			
	X	Y	M	T	C	S	D.b	KnX	KnY	KnM	KnS	T	C	D	V, Z	H	K	E	P
S												●	●	●	●	●	●		
D												●	●	●	●				
n														●		●	●		

Function and Action Description

16-bit Operation (FINS, FINSP)

Insert the 16-bit data S into the nth number of the data table (after D). The data after the nth of the data table is moved back one by one, and the number of savingdata is increased by 1.



Note

Note	
1	The users need to manage the range of soft components used in the data table themselves. The range of the data table is D starting from the next soft component (D+1) of saving data D.

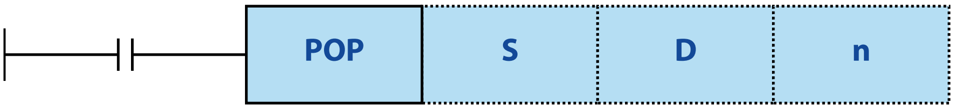
Error

Error	
1	Operation errors may occur in the following cases. The error flag bit M8067 turns ON, and the error code (K6706) is stored in D8067. • The table position n of the data to be inserted is larger than the number of saving data after increased by 1. • The value of n is beyond the soft component range of the data table. • The instruction is executed in the case of n ≤ 0. • The range of the data table is beyond the range of the corresponding soft component.

4.18.3 FN 212 - POP/Read the Last-in Data

Outline

An instruction to read the last data stored by the SFWR instruction.



Read the Last-in Data FN212 - POP	Instruction Mark	Execution Condition	Instruction Type	Number of Instruction Steps
	POP	Continuous type	16 bit	7
	POPP	Pulse type	16 bit	7

Operand	Setting Data	Data Type
	S: Saving the starting soft component number of the first-in data (including the pointer data) (saving the starting word soft component number of the data)	16 bit
	D: Saving the soft component number of the last-out data	16 bit
	n: The number of points of the saved data, because the pointer data is included, please set the value after +1. (2 ≤ n ≤ 512)	16 bit

Operand	Operand Object Soft Component																		
	Bit Soft Component							Word Soft Component								Others			
	X	Y	M	T	C	S	D.b	KnX	KnY	KnM	KnS	T	C	D	V, Z	H	K	E	P
S									●	●	●	●	●	●	●				
D									●	●	●	●	●	●	●				
n																●	●		

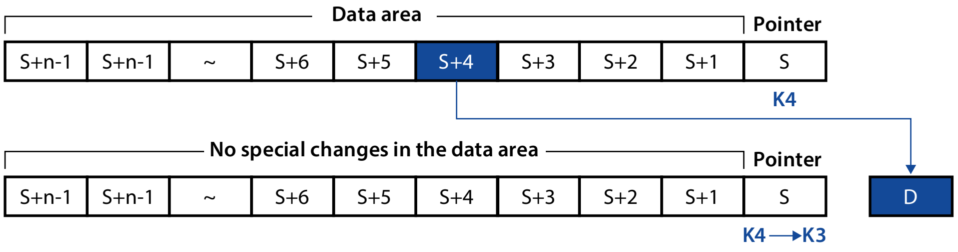
Function and Action Description

16-bit Operation (POP, POPP)

Insert the 16-bit data S into the nth number of the data table (after D). The data after the nth of the data table is moved back one by one, and the number of savingdata is increased by 1.

First-in, Last-out Control Data	Content
S	Pointer data (number of saving data)
S+1	Data area (the first-in data using the shift write instruction (SFWR))
S+2	
S+3	
~	
S+n-3	
S+n-2	
S+n-1	

- For the word soft component of [S ~ S+n-1], the soft component that reads [S + pointer data S] is saved in D each time the instruction is executed. n can be specified as 2 ~ 512.
- The value of the pointer data is reduced by 1.



Related Soft Components

Soft Component	Name	Content
M8020	Zero	When the pointer S = 0, it turns ON after executing the instruction.

Note

Note	
1	When this instruction is programmed in continuous type, the instruction is processed every operation cycle, so please note that unexpected actions may occur sometimes. Generally, it is programmed using [Pulse Type] or by [Pulsed Instruction Contact].
2	When the current value of pointer S is 0, the zero flag bit M8020 is ON, and the instruction is not processed. In this case, first use the comparison instruction to confirm whether the current value of S satisfies 1 ≤ S ≤ (n-1), and then execute this instruction.
3	When the current value of the pointer S is 1, the S is written with 0, and the zero flag bit M8020 is ON.

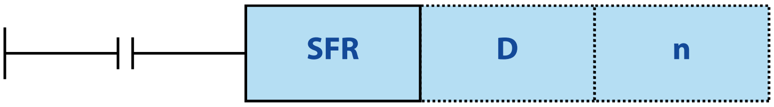
Error

Error	
1	Operation errors may occur in the following cases. The error flag bit M8067 turns ON, and the error code (K6706) is stored in D8067. • When S > n-1. • When S < 0.

4.18.4 FN 213 - SFR/n Bit Right Shift (with Carry) of 16-bit Data

Outline

An instruction that shifts the16-bit data of word soft component to the right by n bits.



n Bit Right Shift (with Carry) of 16-bit Data FN213 - SFR	Instruction Mark	Execution Condition	Instruction Type	Number of Instruction Steps
	SFR	Continuous type	16 bit	5
	SFRP	Pulse type	16 bit	5

Operand	Setting Data														Data Type				
	D: Saving the soft component number of the data to be moved														16 bit				
	n: The number of moves (0 ≤ n ≤ 15)														16 bit				
	Operand Object Soft Component																		
	Bit Soft Component							Word Soft Component								Others			
	X	Y	M	T	C	S	D.b	KnX	KnY	KnM	KnS	T	C	D	V, Z	H	K	E	P
D									●	●	●	●	●	●	●				
n								●	●	●	●	●	●	●		●	●		

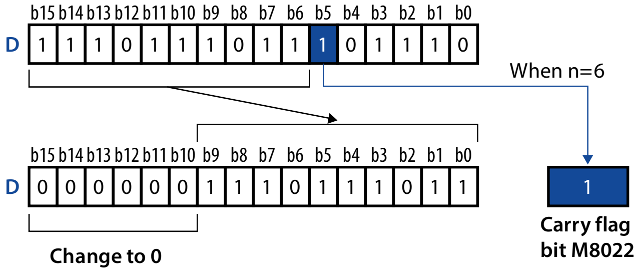
Function and Action Description

16-bit Operation (SFR, SFRP)

The 16 bits of word soft component D is shifted right by n bits.

- n is specified as a number from 0 to 15.
- When a value of 16 or more is specified in n, it moves according to the remainder of n/16. If n = 18, 18/16 = 1 and the remainder is 2, so shift 2 bits to the right.

The ON (1)/OFF (0) status of the nth bit (n-1 bit) in word soft component D is transferred to the carry flag bit M8022, and the n bits starting from the highest bit change to 0, as the figure shown below.


b15 b14 b13 b12 b11 b10 b9 b8 b7 b6 b5 b4 b3 b2 b1 b0
D 1 1 1 0 1 1 1 0 1 1 1 0 1 1 1 0
When n=6
b15 b14 b13 b12 b11 b10 b9 b8 b7 b6 b5 b4 b3 b2 b1 b0
D 0 0 0 0 0 0 1 1 1 0 1 1 1 0 1 1
1
Carry flag
bit M8022
Change to 0

Related Soft Components

Soft Component	Name	Content
M8022	Carry	The status of moving (n-1) bits (ON/OFF)

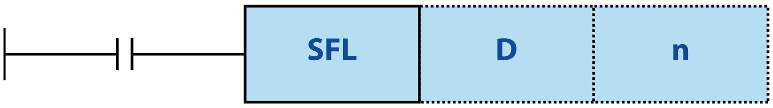
Error

Error	
1	An operation error occurs when n is specified as a negative value. The error flag bit M8067 turns ON, and the error code (6706) is stored in D8067.

4.18.5 FN 214 - SFL/n Bit Left Shift (with Carry) of 16-bit Data

Outline

An instruction that shifts the16-bit data of word soft component to the left by n bits.



n Bit Left Shift (with Carry) of 16-bit Data FN214 - SFL	Instruction Mark	Execution Condition	Instruction Type	Number of Instruction Steps
	SFL	Continuous type	16 bit	5
	SFLP	Pulse type	32 bit	5

Operand	Setting Data														Data Type				
	D: Saving the soft component number of the data to be moved														16 bit				
	n: The number of moves (0 ≤ n ≤ 15)														16 bit				
	Operand Object Soft Component																		
	Bit Soft Component							Word Soft Component								Others			
	X	Y	M	T	C	S	D.b	KnX	KnY	KnM	KnS	T	C	D	V, Z	H	K	E	P
D									●	●	●	●	●	●	●				
n								●	●	●	●	●	●	●		●	●		

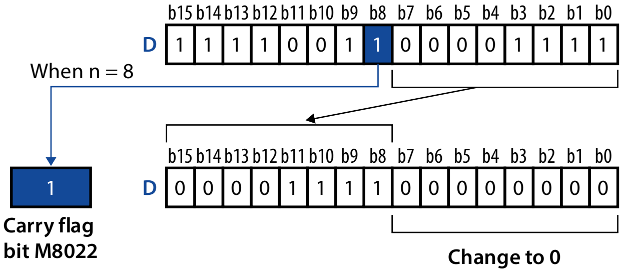
Function and Action Description

16-bit Operation (SFL, SFLP)

The 16 bits of word soft component D is shifted left by n bits.

- n is specified as a number from 0 to 15.
- When a value of 16 or more is specified in n, it moves according to the remainder of n/16. If n = 18, 18/16 = 1 and the remainder is 2, so shift 2 bits to the left.

The ON (1)/OFF (0) status of the (n+1)th bit (n bit) in word soft component D is transferred to the carry flag bit M8022, and the n bits starting from the lowest bit change to 0, as the figure shown below.


b15 b14 b13 b12 b11 b10 b9 b8 b7 b6 b5 b4 b3 b2 b1 b0
D 1 1 1 1 0 0 1 1 0 0 0 0 1 1 1 1
When n = 8
b15 b14 b13 b12 b11 b10 b9 b8 b7 b6 b5 b4 b3 b2 b1 b0
1
D 0 0 0 0 1 1 1 1 0 0 0 0 0 0 0 0
Carry flag bit M8022
Change to 0

Related Soft Components

Soft Component	Name	Content
M8022	Carry	The status of moving n bits (ON/OFF)

Error

Error	
1	An operation error occurs when n is specified as a negative value. The error flag bit M8067 turns ON, and the error code (6706) is stored in D8067.

4.19 Contact Comparison Instructions - FN 220 ~ FN 249

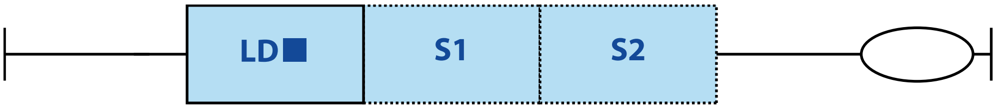
Instructions for data comparison using LD, AND, and OR contact symbols are provided in FN 220 ~ FN 249.

FN No.	Instruction Mark	Instruction Format	Function	Section	Page
224	LD=	LD= (S1) (S2) LDD= (S1) (S2)	Contact comparison LDS1= S2	4.19.1	231
225	LD>	LD> (S1) (S2) LDD> (S1) (S2)	Contact comparison LDS1> S2	4.19.1	231
226	LD<	LD< (S1) (S2) LDD< (S1) (S2)	Contact comparison LDS1< S2	4.19.1	231
228	LD<>	LD<> (S1) (S2) LDD<> (S1) (S2)	Contact comparison LDS1<> S2	4.19.1	231
229	LD<=	LD<= (S1) (S2) LDD<= (S1) (S2)	Contact comparison LDS1<= S2	4.19.1	231
230	LD>=	LD>= (S1) (S2) LDD>= (S1) (S2)	Contact comparison LDS1>= S2	4.19.1	231
232	AND=	AND= (S1) (S2) ANDD= (S1) (S2)	Contact comparison ANDS1= S2	4.19.2	232
233	AND>	AND> (S1) (S2) ANDD> (S1) (S2)	Contact comparison ANDS1> S2	4.19.2	232
234	AND<	AND< (S1) (S2) ANDD< (S1) (S2)	Contact comparison ANDS1< S2	4.19.2	232
236	AND<>	AND<> (S1) (S2) ANDD<> (S1) (S2)	Contact comparison ANDS1<> S2	4.19.2	232
237	AND<=	AND<= (S1) (S2) ANDD<= (S1) (S2)	Contact comparison ANDS1<= S2	4.19.2	232
238	AND>=	AND>= (S1) (S2) ANDD>= (S1) (S2)	Contact comparison ANDS1>= S2	4.19.2	232
240	OR=	OR= (S1) (S2) ORD= (S1) (S2)	Contact comparison ORS1= S2	4.19.3	233
241	OR>	OR> (S1) (S2) ORD> (S1) (S2)	Contact comparison ORS1> S2	4.19.3	233
242	OR<	OR< (S1) (S2) ORD< (S1) (S2)	Contact comparison ORS1< S2	4.19.3	233
244	OR<>	OR<> (S1) (S2) ORD<> (S1) (S2)	Contact comparison ORS1<> S2	4.19.3	233
245	OR<=	OR<= (S1) (S2) ORD<= (S1) (S2)	Contact comparison ORS1<= S2	4.19.3	233
246	OR>=	OR>= (S1) (S2) ORD>= (S1) (S2)	Contact comparison ORS1>= S2	4.19.3	233

4.19.1 FN 224 ~ 230 - LD =, >, <, <>, <=, >=/Contact Comparison

Outline

A contact comparison operation instruction to compare the execution values, and when the condition is satisfied, the contact turns ON.



Contact Comparison FN224 - LD= FN225 - LD> FN226 - LD< FN228 - LD<> FN229 - LD<= FN230 - LD>=	Instruction Mark	Execution Condition	Instruction Type	Number of Instruction Steps
	LD■	Continuous type	16 bit	5
	LDD■	Continuous type	32 bit	9
	■: Comparison conditions =, >, <, <>, <=, >=			

Operand	Setting Data														Data Type				
	S1: Saving the soft component number of the comparison data														16/32 bit				
	S2: Saving the soft component number of the comparison data														16/32 bit				
	Operand Object Soft Component																		
	Bit Soft Component							Word Soft Component								Others			
	X	Y	M	T	C	S	D.b	KnX	KnY	KnM	KnS	T	C	D	V, Z	H	K	E	P
S1								●	●	●	●	●	●	●	●	●	●		
S2								●	●	●	●	●	●	●	●	●	●		

Function and Action Description

16-bit Operation (LD■), 32-bit Operation (LDD■)

Contact comparison instructions connected to the bus.

The BIN comparison is performed on the contents of S1 and S2, and the conduction or non-conduction of the contacts is controlled according to the result.

FNNo	16 Bit Instruction	32 Bit Instruction	Conduction Condition	Non-conduction Condition
224	LD=	LDD=	S1 = S2	S1 ≠ S2
225	LD>	LDD>	S1 > S2	S1 <= S2
226	LD<	LDD<	S1 < S2	S1 >= S2
228	LD<>	LDD<>	S1 ≠ S2	S1 = S2
229	LD<=	LDD<=	S1 <= S2	S1 > S2
230	LD>=	LDD>=	S1 >= S2	S1 < S2

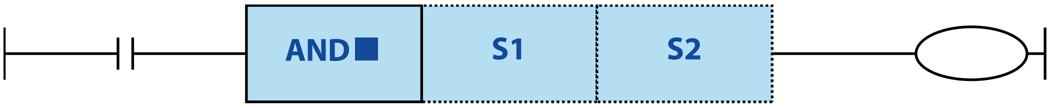
Note

Note		Description
1	About negative numbers	When the highest bit of S1 and S2 is 1, its value is compared as a negative number. • B15 or b31 is the highest bit.
2	When using a 32-bit counter (including a high-speed counter)	The comparison of 32-bit counters (C200 ~ C255) must be performed with 32 bit (LDD=, LDD>, LDD<, etc.). If 16 bit operation (LD=, LD>, LD<, etc.) is specified, a program error or an operation error will occur.

4.19.2 FN 232 ~ 238 - AND=, >, <, <>, <=, >=/Contact Comparison

Outline

A contact comparison operation instruction to compare the execution values, and when the condition is satisfied, the contact turns ON.



Contact Comparison FN232 - AND= FN233 - AND> FN234 - AND< FN236 - AND<> FN237 - AND<= FN238 - AND>=	Instruction Mark	Execution Condition	Instruction Type	Number of Instruction Steps
	AND■	Continuous type	16 bit	5
	ANDD■	Continuous type	32 bit	9
	■: Comparison conditions =, >, <, <>, <=, >=			

<table>
<tr><td rowspan="6">Operand</td><td colspan="14">Setting Data</td><td colspan="5">Data Type</td></tr>
<tr><td colspan="14">S1: Saving the soft component number of the comparison data</td><td colspan="5">16/32 bit</td></tr>
<tr><td colspan="14">S2: Saving the soft component number of the comparison data</td><td colspan="5">16/32 bit</td></tr>
<tr><td colspan="19">Operand Object Soft Component</td></tr>
<tr><td colspan="7">Bit Soft Component</td><td colspan="8">Word Soft Component</td><td colspan="4">Others</td></tr>
<tr><td>X</td><td>Y</td><td>M</td><td>T</td><td>C</td><td>S</td><td>D.b</td><td>KnX</td><td>KnY</td><td>KnM</td><td>KnS</td><td>T</td><td>C</td><td>D</td><td>V, Z</td><td>H</td><td>K</td><td>E</td><td>P</td></tr>
<tr><td>S1</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td>●</td><td>●</td><td>●</td><td>●</td><td>●</td><td>●</td><td>●</td><td>●</td><td>●</td><td>●</td><td></td><td></td></tr>
<tr><td>S2</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td>●</td><td>●</td><td>●</td><td>●</td><td>●</td><td>●</td><td>●</td><td>●</td><td>●</td><td>●</td><td></td><td></td></tr>
</table>

Function and Action Description

16-bit Operation (AND■), 32-bit Operation (ANDD■)

Contact comparison instructions in series with other contacts.
The BIN comparison is performed on the contents of S1 and S2, and the conduction or non-conduction of the contacts is controlled according to the result.

FNNo	16 Bit Instruction	32 Bit Instruction	Conduction Condition	Non-conduction Condition
232	AND=	ANDD=	S1 = S2	S1 ≠ S2
233	AND>	ANDD>	S1 > S2	S1 <= S2
234	AND<	ANDD<	S1 < S2	S1 >= S2
236	AND<>	ANDD<>	S1 ≠ S2	S1 = S2
237	AND<=	ANDD<=	S1 <= S2	S1 > S2
238	AND>=	ANDD>=	S1 >= S2	S1 < S2

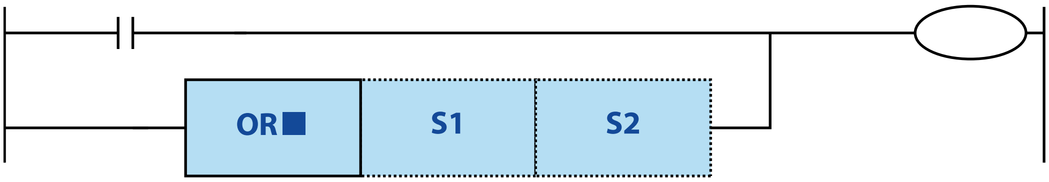
Note

Note		Description
1	About negative numbers	When the highest bit of S1 and S2 is 1, its value is compared as a negative number. • B15 or b31 is the highest bit.
2	When using a 32-bit counter (including a high-speed counter)	The comparison of 32-bit counters (C200 ~ C255) must be performed with 32 bit (ANDD=, ANDD>, ANDD<, etc.). If 16 bit operation (AND=, AND>, AND<, etc.) is specified, a program error or an operation error will occur.

4.19.3 FN 240 ~ 246 - OR=, >, <, <>, <=, >=/Contact Comparison

Outline

A contact comparison operation instruction to compare the execution values, and when the condition is satisfied, the contact turns ON.



Contact Comparison FN240 - OR= FN241 - OR> FN242 - OR< FN244 - OR<> FN245 - OR<= FN246 - OR>=	Instruction Mark	Execution Condition	Instruction Type	Number of Instruction Steps
	OR■	Continuous type	16 bit	5
	ORD■	Continuous type	32 bit	9
	■: Comparison conditions =, >, <, <>, <=, >=			

Operand	Setting Data														Data Type				
	S1: Saving the soft component number of the comparison data														16/32 bit				
	S2: Saving the soft component number of the comparison data														16/32 bit				
	Operand Object Soft Component																		
	Bit Soft Component							Word Soft Component								Others			
	X	Y	M	T	C	S	D.b	KnX	KnY	KnM	KnS	T	C	D	V, Z	H	K	E	P
S1								●	●	●	●	●	●	●	●	●	●		
S2								●	●	●	●	●	●	●	●	●	●		

Function and Action Description

16-bit Operation (OR■), 32-bit Operation (ORD■)

Contact comparison instructions in parallel with other contacts.

The BIN comparison is performed on the contents of S1 and S2, and the conduction or non-conduction of the contacts is controlled according to the result.

FNNo	16 Bit Instruction	32 Bit Instruction	Conduction Condition	Non-conduction Condition
240	OR=	ORD=	S1 = S2	S1 ≠ S2
241	OR>	ORD>	S1 > S2	S1 <= S2
242	OR<	ORD<	S1 < S2	S1 >= S2
244	OR<>	ORD<>	S1 ≠ S2	S1 = S2
245	OR<=	ORD<=	S1 <= S2	S1 > S2
246	OR>=	ORD>=	S1 >= S2	S1 < S2

Note

Note		Description
1	About negative numbers	When the highest bit of S1 and S2 is 1, its value is compared as a negative number. • B15 or b31 is the highest bit.
2	When using a 32-bit counter (including a high-speed counter)	The comparison of 32-bit counters (C200 ~ C255) must be performed with 32 bit (ORD=, ORD>, ORD<, etc.). If 16 bit operation (OR=, OR>, OR<, etc.) is specified, a program error or an operation error will occur.

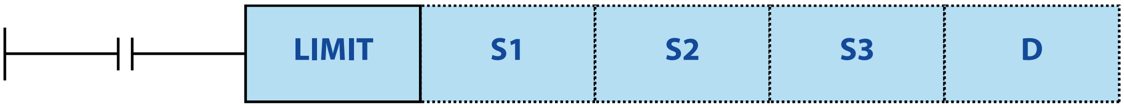
4.20 Data Table Processing - FN 250 ~ FN 269

FN No.	Instruction Mark	Instruction Format	Function	Section	Page
256	LIMIT	LIMIT (S1) (S2) (S3) (D) LIMITP (S1) (S2) (S3) (D) DLIMIT (S1) (S2) (S3) (D) DLIMITP (S1) (S2) (S3) (D)	Upper and lower limit control	4.20.1	235
257	BAND	BAND (S1) (S2) (S3) (D) BANDP (S1) (S2) (S3) (D) DBAND (S1) (S2) (S3) (D) DBANDP (S1) (S2) (S3) (D)	Dead band control	4.20.2	237
258	ZONE	ZONE (S1) (S2) (S3) (D) ZONEP (S1) (S2) (S3) (D) DZONE (S1) (S2) (S3) (D) DZONEP (S1) (S2) (S3) (D)	Zone control	4.20.3	239
259	SCL	SCL (S1) (S2) (D) SCLP (S1) (S2) (D) DSCL (S1) (S2) (D) DSCLP (S1) (S2) (D)	Fixed coordinates (coordinate data of different point)	4.20.4	241
269	SCL2	SCL2 (S1) (S2) (D) SCL2P (S1) (S2) (D) DSCL2 (S1) (S2) (D) DSCL2P (S1) (S2) (D)	Fixed coordinates 2 (X/Y coordinate data)	4.20.5	244

4.20.1 FN 256 - LIMIT/Upper and Lower Limit Control

Outline

An instruction to set the upper/lower limit value of the input value and then output.



Upper and Lower Limit Control FN256 - LIMIT	Instruction Mark	Execution Condition	Instruction Type	Number of Instruction Steps
	LIMIT	Continuous type	16 bit	9
	LIMITP	Pulse type	16 bit	9
	DLIMIT	Continuous type	32 bit	17
	DLIMITP	Pulse type	32 bit	17

Operand	Setting Data	Data Type
	S1: Lower limit value (Min. output limit value)	16/32 bit
	S2: Upper limit limit value (Max. output limit value)	16/32 bit
	S3: Input value required the upper and lower limit control	16/32 bit
	D: Saving the soft component start number of the output value that has passed the upper and lower limit control	16/32 bit

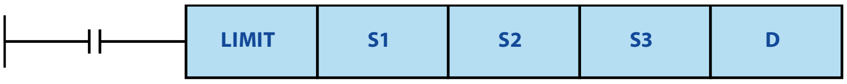
Operand Object Soft Component	Bit Soft Component							Word Soft Component								Others			
	X	Y	M	T	C	S	D.b	KnX	KnY	KnM	KnS	T	C	D	V, Z	H	K	E	P
S1								●	●	●	●	●	●	●	●	●	●		
S2								●	●	●	●	●	●	●	●	●	●		
S3								●	●	●	●	●	●	●	●				
D									●	●	●	●	●	●	●				

Function and Action Description

16-bit Operation (LIMIT, LIMITP)

By judging whether the input value (BIN 16-bit value) specified in S3 is within the range of the upper and lower limit specified by S1 and S2, the output value stored in the soft component specified by D is controlled.

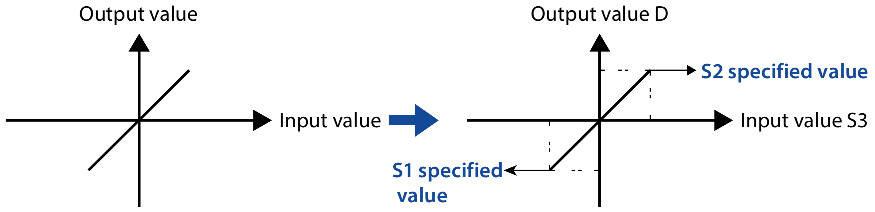
The output values are controlled as shown below.



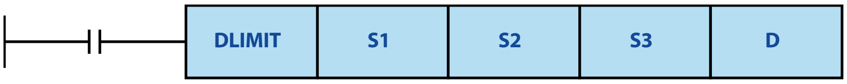
S1 lower limit value > S3 input value → S1 lower limit value → D output value

S2 upper limit value < S3 input value → S2 upper limit value → D output value

S1 lower limit value <= S3 input value <= S2 upper limit value → S3 input value → D output value



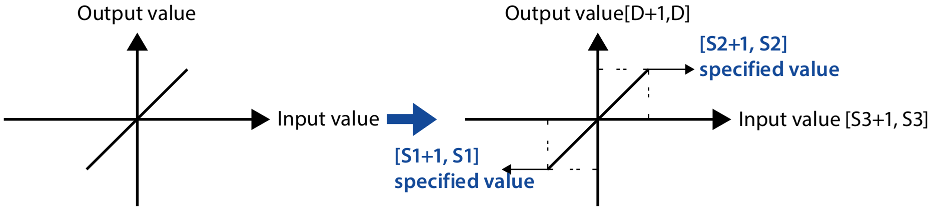
32-bit Operation (DLIMIT, DLIMITP)
By judging whether the input value (BIN 32-bit value) specified in [S3+1,S3] is within the range of the upper and lower limit specified by [S1+1,S1] and [S2+1,S2], the output value stored in the soft component specified by [D+1,D] is controlled. The output values are controlled as shown below.



[S1+1, S1] lower limit value > [S3+1, S3] input value → [S1+1, S1] lower limit value → [D+1, D] output value

[S2+1, S2] upper limit value < [S3+1, S3] input value → [S2+1, S2] upper limit value → [D+1, D] output value

[S1+1, S1] lower limit value <= S3 input value <= [S2+1, S2] upper limit value → [S3+1, S3] input value → [D+1, D] output value



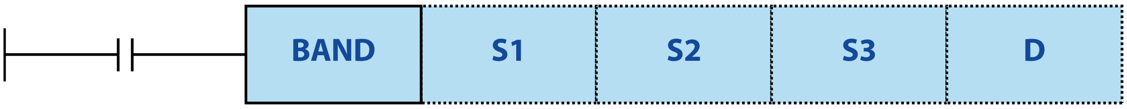
Error

Error		
1	An operation error occurs after executing the instruction in the following setting status, the error flag bit M8067 turns ON, and the error code (K6706) is stored in D8067.	
		Size Relationship
	16 bit operation	S1 > S2
	32 bit operation	[S1+1,S1] > [S2+1,S2]

4.20.2 FN 257 - BAND/Dead Band Control

Outline

An instruction to control the output value by judging whether the input value is within the range of upper and lower limit of the specified dead band.



Dead Band Control FN257 - BAND	Instruction Mark	Execution Condition	Instruction Type	Number of Instruction Steps
	BAND	Continuous type	16 bit	9
	BANDP	Pulse type	16 bit	9
	DBAND	Continuous type	32 bit	17
	DBANDP	Pulse type	32 bit	17

Operand	Setting Data	Data Type
	S1: Lower limit value of dead band (non-output area)	16/32 bit
	S2: Upper limit value of dead band (non-output area)	16/32 bit
	S3: Input value required the dead band control	16/32 bit
	D: Saving the soft component number of the output value that has passed the dead band control	16/32 bit

Operand Object Soft Component	Bit Soft Component							Word Soft Component								Others			
	X	Y	M	T	C	S	D.b	KnX	KnY	KnM	KnS	T	C	D	V, Z	H	K	E	P
S1								●	●	●	●	●	●	●	●	●	●		
S2								●	●	●	●	●	●	●	●	●	●		
S3								●	●	●	●	●	●	●	●				
D									●	●	●	●	●	●	●				

Function and Action Description

16-bit Operation (BAND, BANDP)

By judging whether the input value (BIN 16-bit value) specified in S3 is within the range of the dead band specified by S1 and S2, the output value stored in the soft component specified by D is controlled.

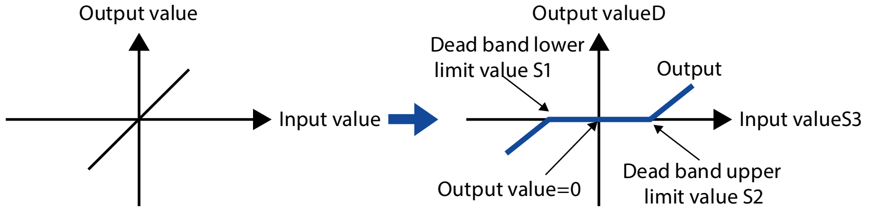
The output values are controlled as shown below.

BAND
S1
S2
S3
D

S3 input value < S1 lower limit value → S3 input value - S1 lower limit value → D output value

S3 input value > S2 upper limit value → S3 input value - S2 upper limit value → D output value

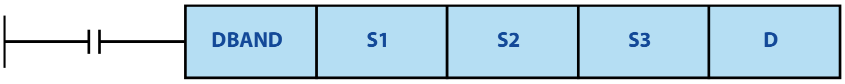
S1 lower limit value < = S3 input value → 0 → D output value
< = S2 upper limit valu



32-bit Operation (DBAND, DBANDP)

By judging whether the input value (BIN 32-bit value) specified in [S3+1,S3] is within the range of the dead band specified by [S1+1,S1] and [S2+1,S2], the output value stored in the soft component specified by [D+1,D] is controlled.

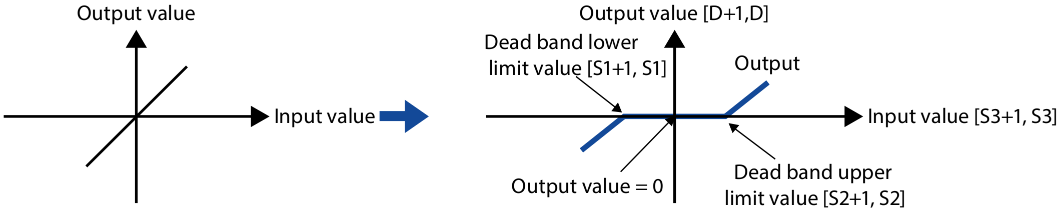
The output values are controlled as shown below.



[S3+1, S3] input value < [S1+1, S1] lower limit value → [S3+1,S3] input value - [S1+1,S1] lower limit value → [D+1, D] output value

[S3+1, S3] input value > [S2+1, S2] upper limit value → [S3+1,S3] input value - S2 upper limit value → [D+1, D] output value

[S1+1, S1] lower limit value <= [S3+1, S3] input value <= [S2+1, S2] upper limit value → 0 → [D+1, D] output value



Note

Note

1 When the output value overflows, as shown below.

- For 16 bit operation: The output value is a 16-bit BIN value with a sign. Therefore, when the operation result is not within the range -32,768 ~ +32,767, as shown below.

  Dead band lower limit valueS1=10
  Input value S3=-32768
  → Output value =-32768-10
  =8000H-AH
  =7FF6H
  =32758

- For 32 bit operatio: The output value is a 32-bit BIN value with a sign. Therefore, when the operation result is not within the range -2,147,483,648 ~ +2,147,483,647, as shown below.

  Dead band lower limit value[S1+1, S1]=1000
  Input value [S3+1, S3]=-2,147,483,648
  → Output value =-2,147,483,648-1000
  =80000000H-000003E8H
  =7FFFFC18H
  =2,147,482,648

Error

Error

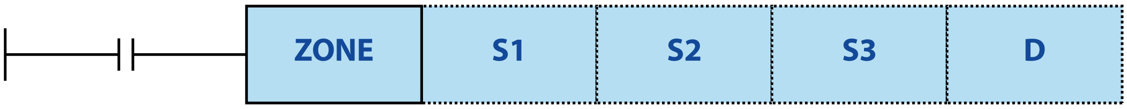
1 An operation error occurs after executing the instruction in the following setting status, the error flag bit M8067 turns ON, and the error code (K6706) is stored in D8067.

	Size Relationship
16 bit operation	S1 > S2
32 bit operation	[S1+1,S1] > [S2+1,S2]

4.20.3 FN 258 - ZONE/Zone Control

Outline

An instruction to control the output value by judging whether the input value is within the range of upper and lower limit of the specified zone.



	Instruction Mark	Execution Condition	Instruction Type	Number of Instruction Steps
Zone Control FN258 - ZONE	ZONE	Continuous type	16 bit	9
	ZONEP	Pulse type	16 bit	9
	DZONE	Continuous type	32 bit	17
	DZONEP	Pulse type	32 bit	17

Operand	Setting Data	Data Type
	S1: Negative deviation value added to the input value	16/32 bit
	S2: Positive deviation value added to the input value	16/32 bit
	S3: Input value required the zone control	16/32 bit
	D: Saving the soft component start number of the output value that has passed the zone control	16/32 bit

Operand Object Soft Component	Bit Soft Component							Word Soft Component								Others			
	X	Y	M	T	C	S	D.b	KnX	KnY	KnM	KnS	T	C	D	V, Z	H	K	E	P
S1								●	●	●	●	●	●	●	●	●	●		
S2								●	●	●	●	●	●	●	●	●	●		
S3								●	●	●	●	●	●	●	●				
D									●	●	●	●	●	●	●				

Function and Action Description

16-bit Operation (ZONE, ZONEP)

Add the deviation value specified by S1 or S2 to the input value specified by S3, and then save it to the soft component number specified by D.

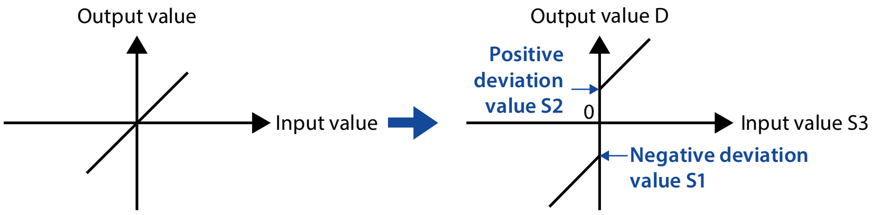
The attachment of the deviation value is executed as shown below.

ZONE
S1
S2
S3
D

S3 input value < 0 → S3 input value + S1 negative deviation value → D output value

S3 input value = 0 → 0 → D output value

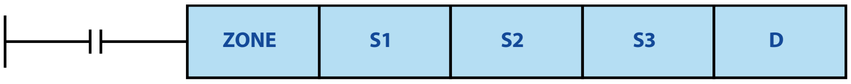
S3 input value > 0 → S3 input value + S2 positive deviation value → D output value



32-bit Operation (DZONE, DZONEP)

Add the deviation value specified by [S1+1,S1] or [S2+1,S2] to the input value specified by [S3+1,S3], and then save it to the soft component number specified by [D,D+1].

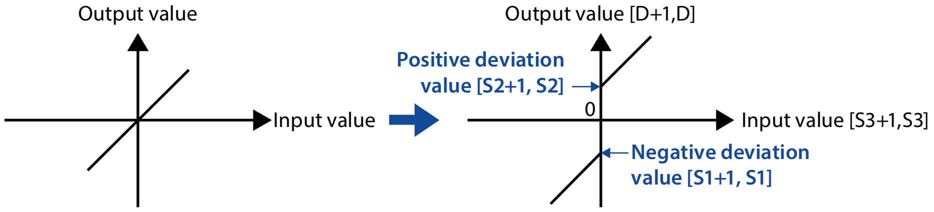
The attachment of the deviation value is executed as shown below.



[S3+1, S3] input value < 0- ·· ·· ·· → [S3+1, S3] input value + [S1+1, S1] negative deviation value → [D+1, D] output value

[S3+1, S3] input value = 0- ·· ·· ·· → 0 → [D+1, D] output value

[S3+1, S3] input value > 0- ·· ·· ·· → [S3+1, S3] input value + [S2+1, S2] positive deviation value → [D+1, D] output value



Note

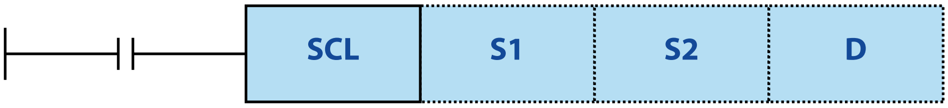
Note	
1	When the output value overflows, as shown below. • For 16 bit operation: The output value is a 16-bit BIN value with a sign. Therefore, when the operation result is not within the range -32768 ~ +32767, as shown below. Negative eviation value S1 = -100 Input value S3 = -32,768 → Output value = -32,768 + (-100) = 8000H + FF9CH = 7F9CH = 32,668 • For 32 bit operation: The output value is a 32-bit BIN value with a sign. Therefore, when the operation result is not within the range -2,147,483,648 ~ +2,147,483,647, as shown below. Negative eviation value [S1+1,S1] = -1000 Input value [S3+1,S3] = -2,147,483,648 → Output value = -2,147,483,648 + (-1000) = 80000000H + FFFFFC18H = 7FFFFC18 = 2,147,482,648

4.20.4 FN 259 - SCL/Fixed Coordinates

Outline

An instruction to execute fix coordinates on the input value and then output according to the specified data table.

In addition, there are SCL2 (FN 269) instructions with different data table structures.



Fixed Coordinates (Coordinate Data of Different Point) FN259 - SCL	Instruction Mark	Execution Condition	Instruction Type	Number of Instruction Steps
	SCL	Continuous type	16 bit	7
	SCLP	Pulse type	16 bit	7
	DSCL	Continuous type	32 bit	13
	DSCLP	Pulse type	32 bit	13

Operand	Setting Data	Data Type
	S1: The input value executed the fixed coordinates or saving the soft component number of the input value	16/32 bit
	S2: Start number of the conversion table soft component for fixed coordinates	16/32 bit
	D: Saving the soft component number of the output value controlled by the fixed coordinates	16/32 bit

Operand	Operand Object Soft Component																		
	Bit Soft Component							Word Soft Component								Others			
	X	Y	M	T	C	S	D.b	KnX	KnY	KnM	KnS	T	C	D	V, Z	H	K	E	P
S1								●	●	●	●	●	●	●	●	●	●		
S2														●	●				
D									●	●	●	●	●	●	●				

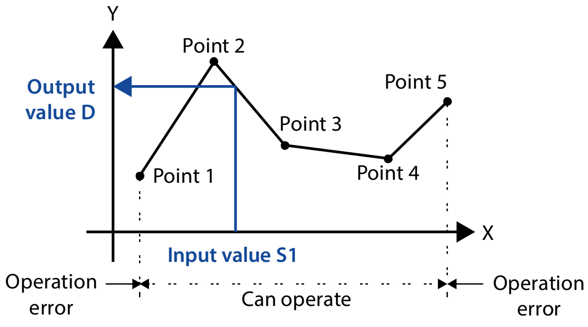
Function and Action Description

16-bit Operation (SCL, SCLP)

According to the specified conversion characteristics, fix coordinates are executed on the input value specified by S1 , and then saved to the soft component number specified by D.

The conversion for fixed coordinates is executed based on the data table stored at the beginning of the soft component specified in S2.

However, when the output data is not an integer value, the first digit of the decimal is rounded off and output.



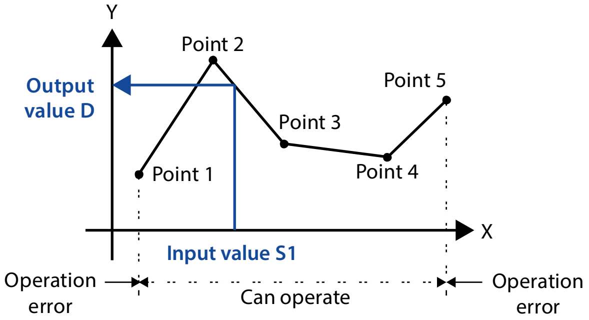
Setting Item		Soft Component Allocation of the Setting Data Table
Number of coordinate points (changes to "5" when it is the left picture)		S2
Point 1	x coordinate	S2+1
	y coordinate	S2+2
Point 2	x coordinate	S2+3
	y coordinate	S2+4
Point 3	x coordinate	S2+5
	y coordinate	S2+6
Point 4	x coordinate	S2+7
	y coordinate	S2+8
Point 5	x coordinate	S2+9
	y coordinate	S2+10

32-bit Operation (DSCL, DSCLP)

According to the specified conversion characteristics, fix coordinates are executed on the input value specified by [S1+1,S1], and then saved to the soft component number specified by [D+1,D].

The conversion for fixed coordinates is executed based on the data table stored at the beginning of the soft component specified in [S2+1,S2].

However, when the output data is not an integer value, the first digit of the decimal is rounded off and output.



Setting Item		Soft Component Allocation of the Setting Data Table
Number of coordinate points (changes to "5" when it is the left picture)		[S2+1,S2]
Point 1	x coordinate	[S2+3, S2+2]
	y coordinate	[S2+5, S2+4]
Point 2	x coordinate	[S2+7, S2+6]
	y coordinate	[S2+9, S2+8]
Point 3	x coordinate	[S2+11, S2+10]
	y coordinate	[S2+13, S2+12]
Point 4	x coordinate	[S2+15, S2+14]
	y coordinate	[S2+17, S2+16]
Point 5	x coordinate	[S2+19, S2+18]
	y coordinate	[S2+21, S2+20]

Fixed Coordinate Conversion Table Setting

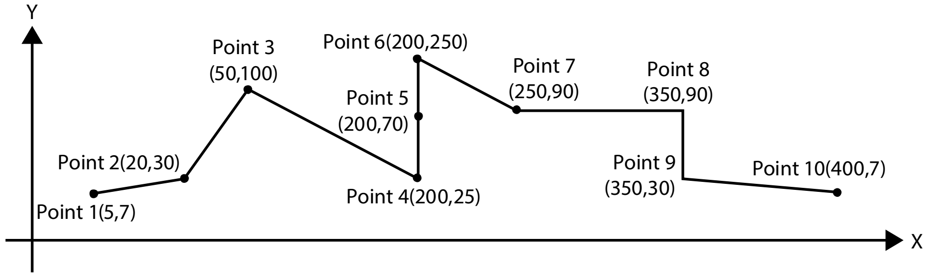
The conversion table for the fixed coordinate is is executed based on the data table stored at the beginning of the soft component specified in [S2+1,S2].

The structure of the data table is shown on the right.

Setting Item		Soft Component Allocation of the Setting Data Table	
		16 Bit Operation	32 Bit Operation
Number of coordinate points		S2	[S2+1,S2]
Point 1	x coordinate	S2+1	[S2+3,S2+2]
	y coordinate	S2+2	[S2+5,S2+4]
Point 2	x coordinate	S2+3	[S2+7,S2+6]
	y coordinate	S2+4	[S2+9,S2+8]
…	…	…	…
Point n (last)	x coordinate	S2+2n-1	[S2+4n-1,S2+4n-2]
	y coordinate	S2+2n	[S2+4n+1,S2+4n]

The setting example of the fixed coordinate conversion table is as follows. Take 16 bit operation as an example.

- When executing 32 bit operation, set the data in each setting item with BIN 32 bit data.
- When using the conversion characteristics for the fixed coordinates shown in the figure below, set the data table as shown in the table below.



Setting Item		Setting Soft Component and Setting Content			Remark
		When D0 is Specified in S2		Setting Content	
Number of coordinate points		S2	D0	K10	
Point 1	x coordinate	S2+1	D1	K5	
	y coordinate	S2+2	D2	K7	
Point 2	x coordinate	S2+3	D3	K20	
	y coordinate	S2+4	D4	K30	
Point 3	x coordinate	S2+5	D5	K50	
	y coordinate	S2+6	D6	K100	

Fixed Coordinate Conversion Table Setting

Setting Item		Setting Soft Component and Setting Content			Remark
		When D0 is Specified in S2		Setting Content	
Point 4	x coordinate	S2+7	D7	K200	If the coordinates of 3 points are specified like this, the output value is the intermediate value. In this example, the y coordinate of point 5 is specified as the output value (intermediate value). In addition, when the x coordinates of 3 or more points are the same, also output the value of the 2nd point.
	y coordinate	S2+8	D8	K25	
Point 5	x coordinate	S2+9	D9	K200	
	y coordinate	S2+10	D10	K70	
Point 6	x coordinate	S2+11	D11	K200	
	y coordinate	S2+12	D12	K250	
Point 7	x coordinate	S2+13	D13	K250	
	y coordinate	S2+14	D14	K90	
Point 8	x coordinate	S2+15	D15	K350	If the coordinates of 2 points are specified like this, the output value takes the y coordinate value of the next point. In this example, the y coordinate of point 9 is specified as the output value.
	y coordinate	S2+16	D16	K90	
Point 9	x coordinate	S2+17	D17	K350	
	y coordinate	S2+18	D18	K30	
Point 10	x coordinate	S2+19	D19	K400	
	y coordinate	S2+20	D20	K7	

Error

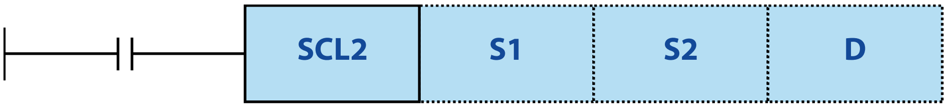
Error	
1	Operation errors may occur in the following cases. The error flag bit M8067 turns ON and the error code (K6706) is stored in D8067. • The Xn data of the data table is not in ascending order. • However, since the operation is searched from the lower bit side of the soft component number of the data table, even if a part of the data table is not arranged in ascending order, the operation up to this part does not cause an operation error, and the instruction is executed. • When S1 is beyond the range set by the data table. • When the value in the operation exceeds the range of 32 bit data, please confirm that the distance between each point does not exceed 65535. • If the distance exceeds 65535, please shorten the distance between each point.

4.20.5 FN 269 - SCL2/Fixed Coordinates 2

Outline

An instruction to execute fix coordinates on the input value and then output according to the specified data table.

In addition, there are SCL2 (FN 259) instructions with different data table structures.



Fixed Coordinates 2 (Coordinate Data of Different Point) FN269 - SCL2	Instruction Mark	Execution Condition	Instruction Type	Number of Instruction Steps
	SCL2	Continuous type	16 bit	7
	SCL2P	Pulse type	16 bit	7
	DSCL2	Continuous type	32 bit	13
	DSCL2P	Pulse type	32 bit	13

Operand	Setting Data	Data Type
	S1: The input value executed the fixed coordinates or saving the soft component number of the input value	16/32 bit
	S2: Start number of the conversion table soft component for fixed coordinates	16/32 bit
	D: Saving the soft component number of the output value controlled by the fixed coordinates	16/32 bit

Operand	Operand Object Soft Component																		
	Bit Soft Component							Word Soft Component								Others			
	X	Y	M	T	C	S	D.b	KnX	KnY	KnM	KnS	T	C	D	V, Z	H	K	E	P
S1								●	●	●	●	●	●	●	●	●	●		
S2														●	●				
D									●	●	●	●	●	●	●				

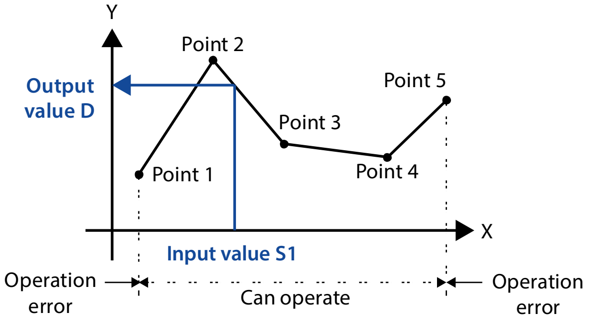
Function and Action Description

16-bit Operation (SCL2, SCL2P)

According to the specified conversion characteristics, fix coordinates are executed on the input value specified by S1, and then saved to the soft component number specified by D.

The conversion for fixed coordinates is executed based on the data table stored at the beginning of the soft component specified in S2.

However, when the output data is not an integer value, the first digit of the decimal is rounded off and output.



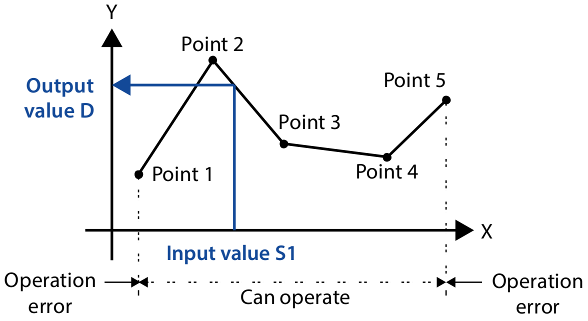
Setting Item		Soft Component Allocation of the Setting Data Table
Number of coordinate points (changes to "5" when it is the left picture)		S2
x coordinate	Point 1	S2+1
	Point 2	S2+2
	Point 3	S2+3
	Point 4	S2+4
	Point 5	S2+5
y coordinate	Point 1	S2+6
	Point 2	S2+7
	Point 3	S2+8
	Point 4	S2+9
	Point 5	S2+10

32 Bit Operation (DSCL2, DSCL2P)

According to the specified conversion characteristics, fix coordinates are executed on the input value specified by [S1+1,S1] , and then saved to the soft component number specified by [D+1,D].

The conversion for fixed coordinates is executed based on the data table stored at the beginning of the soft component specified in [S2+1,S2].

However, when the output data is not an integer value, the first digit of the decimal is rounded off and output.



Setting Item		Soft Component Allocation of the Setting Data Table
Number of coordinate points (changes to "5" when it is the left picture)		[S2+1,S2]
x coordinate	Point 1	[S2+3, S2+2]
	Point 2	[S2+5, S2+4]
	Point 3	[S2+7, S2+6]
	Point 4	[S2+9, S2+8]
	Point 5	[S2+11, S2+10]
y coordinate	Point 1	[S2+13, S2+12]
	Point 2	[S2+15, S2+14]
	Point 3	[S2+17, S2+16]
	Point 4	[S2+19, S2+18]
	Point 5	[S2+21, S2+20]

Fixed Coordinate Conversion Table Setting

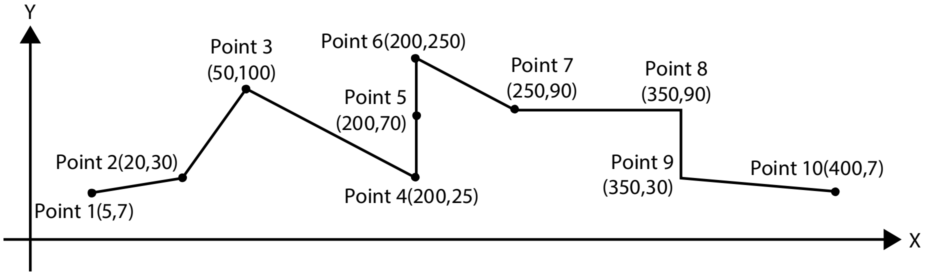
The conversion table for the fixed coordinate is is executed based on the data table stored at the beginning of the soft component specified in [S2+1,S2].

The structure of the data table is shown on the right.

Setting Item		Soft Component Allocation of the Setting Data Table	
		16 Bit Operation	32 Bit Operation
Number of coordinate points		S2	[S2+1,S2]
x coordinate	Point 1	S2+1	[S2+3,S2+2]
	Point 2	S2+2	[S2+5,S2+4]
	…	…	…
	Point n (last)	S2+n	[S2+2n+1,S2+2n]
y coordinate	Point 1	S2+n+1	[S2+2n+3,S2+2n+2]
	Point 2	S2+2n+2	[S2+2n+5,S2+2n+4]
	…	…	…
	Point n (last)	S2+2n	[S2+4n+1,S2+4n]

The setting example of the fixed coordinate conversion table is as follows. Take 16 bit operation as an example.

- When executing 32 bit operation, set the data in each setting item with BIN 32 bit data.
- When using the conversion characteristics for the fixed coordinates shown in the figure below, set the data table as shown in the table below.



Fixed Coordinate Conversion Table Setting

Setting Item		Setting Soft Component and Setting Content			Remark
		When D0 is Specified in S2		Setting Content	
Number of coordinate		S2	D0	K10	
x coordinate	Point 1	S2+1	D1	K5	
	Point 2	S2+2	D2	K20	
	Point 3	S2+3	D3	K50	
	Point 4	S2+4	D4	K200	When 4, 5, and 6 specify the coordinates of 3 points, the output value is the intermediate value. In this example, the y coordinate of point 5 is specified as the output value (intermediate value). In addition, even if the x coordinates of 3 or more points are the same, also output the value of the 2nd point.
	Point 5	S2+5	D5	K200	
	Point 6	S2+6	D6	K200	
	Point 7	S2+7	D7	K250	
	Point 8	S2+8	D8	K350	8, 9 specifies the coordinates of 2 points, then the output value takes the value of the y coordinate of the next point. In this example, the y coordinate of point 9 is specified as the output value.
	Point 9	S2+9	D9	K350	
	Point 10	S2+10	D10	K400	
y coordinate	Point 1	S2+11	D11	K7	
	Point 2	S2+12	D12	K30	
	Point 3	S2+13	D13	K100	
	Point 4	S2+14	D14	K25	When 4, 5, and 6 specify the coordinates of 3 points, the output value is the intermediate value. In this example, the y coordinate of point 5 is specified as the output value (intermediate value). In addition, even if the x coordinates of 3 or more points are the same, also output the value of the 2nd point.
	Point 5	S2+15	D15	K70	
	Point 6	S2+16	D16	K250	
	Point 7	S2+17	D17	K90	
	Point 8	S2+18	D18	K90	8, 9 specifies the coordinates of 2 points, then the output value takes the value of the y coordinate of the next point. In this example, the y coordinate of point 9 is specified as the output value.
	Point 9	S2+19	D19	K30	
	Point 10	S2+20	D20	K7	

Error

Error	
1	Operation errors may occur in the following cases. The error flag bit M8067 turns ON and the error code (K6706) is stored in D8067. • The Xn data of the data table is not in ascending order. • However, since the operation is searched from the lower bit side of the soft component number of the data table, even if a part of the data table is not arranged in ascending order, the operation up to this part does not cause an operation error, and the instruction is executed. • When S1 is beyond the range set by the data table. • When the value in the operation exceeds the range of 32 bit data, please confirm that the distance between each point does not exceed 65535. • If the distance exceeds 65535, please shorten the distance between each point.

4.21 Communication - FN 180/FN 276

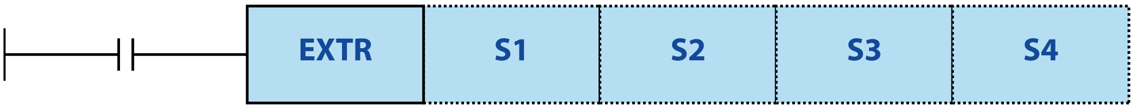
FN No.	Instruction Mark	Instruction Format	Function	Section	Page
180	EXTR	EXTR (S1) (S2) (S3) (S4)	CAN communication	4.21.1	248
276	ADPRW	ADPRW (S) (S1) (S2) (S3) (S4/D)	Modbus read and write	4.21.2	250

4.21.1 FN 180 - EXTR/CAN Communication

Outline

Instruction for communication with the slave station corresponding to the CAN master station (data reading/writing).

Please see section 5.2 for detailed usage of CAN communication.



CAN Communication	Instruction Mark	Execution Condition	Instruction Type	Number of Instruction Steps
FN180 - EXTR	EXTR	Continuous type	16 bit	9

Operand	Setting Data	Data Type
	S1: The high byte indicates the command code, and the low byte indicates the slave station address (0x00 ~ 0xFF)	16 bit
	S2: Slave data address	16 bit
	S3: Access points (word data: 1 ~ 2, bit data 1 ~ 32)	16 bit
	S4: Data storage soft component start	16 bit

Operand	Operand Object Soft Component																		
	Bit Soft Component							Word Soft Component								Others			
	X	Y	M	T	C	S	D.b	KnX	KnY	KnM	KnS	T	C	D	V, Z	H	K	E	P
S1																●	●		
S2								●	●	●	●	●	●	●	●	●	●		
S3								●	●	●	●	●	●	●	●	●	●		
S4								●	●	●	●	●	●	●	●				

Function and Action Description

16-bit Operation (EXTR)

The function parameters required for each function code are shown in the table below.

S1: High Byte is the Command Code	S2: Slave Data Address	S3: Access Points	S4: Data Storage Soft Component Start
0x03 (register readout)	0000H ~ FFFFH	1 ~ 2	Read out the object soft component (starting address) Occupied word count: S3
0x10 (register write)	0000H ~ FFFFH	1 ~ 2	Write to the object soft component (starting address) Occupied word count: S3
0x01 (bit readout)	0000H ~ FFFFH	1 ~ 32	Read out the object soft component (starting address) Occupied word count: (S3 + 15) ÷ 16
0x0F (bt write)	0000H ~ FFFFH	1 ~ 32	Write to the object soft component (starting address) Occupied word count: (S3 + 15) ÷ 16

Related Soft Components

Soft Component	Name	Content
M8029	Instruction end flag	Turn ON after completing the current communication, until the next instruction using this flag. It can be placed after this instruction to read the communication status or perform communication control.

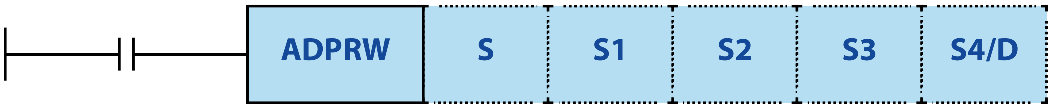
Note

<table>
<tr><th colspan="2">Note</th></tr>
<tr><td>1</td><td>This command can only be used when the machine is set as the master station. The communication parameters can be configured through a special address, see CAN communication function for details.</td></tr>
<tr><td>2</td><td>The communication instructions (EXTR/ADPRW/FROM/TO) are continuously polled from top to bottom in the order of the program step number. The user only needs to turn on the conditions before the communication instruction, without writing logic for polling control.</td></tr>
<tr><td>3</td><td>Communication commands (EXTR/ADPRW/FROM/TO), all communicate in a non-blocking way, polling in the background. Each communication command may occupy several scan cycles. Do not use pulse signals to control communication commands (EXTR/ADPRW /FROM/TO) and ensure that the conduction time is long enough, otherwise the communication command may not be triggered.</td></tr>
<tr><td>4</td><td>If need to send a single communication command (EXTR/ADPRW/FROM/TO), or judge whether the current communication command is sent successfully, it can be controlled with M8029.</td></tr>
<tr><td>5</td><td>Communication commands (EXTR/ADPRW/FROM/TO) are only allowed to be used in the main program. They cannot be used in the following procedures, otherwise it may cause abnormal communication polling.
<table>
<tr><th>Unusable Program Flow</th><th>Note</th></tr>
<tr><td>CJ-P instructions</td><td>Conditional jump</td></tr>
<tr><td>FOR-NEXT instructions</td><td>Cycle</td></tr>
<tr><td>P-SRET instructions</td><td>Subroutine</td></tr>
<tr><td>I-IRET instructions</td><td>Interrupt subroutine</td></tr>
</table>
</td></tr>
</table>

4.21.2 FN 276 - ADPRW/Modbus Read and Write

Outline

As a host, the instructions for Modbus communication are performed.



Modbus Read and Write	Instruction Mark	Execution Condition	Instruction Type	Number of Instruction Steps
FN276 - ADPRW	ADPRW	Continuous type	16 bit	11

Operand	Setting Data	Data Type
	S: High byte: Local MOD port number; Low byte: Slave station address	16 bit
	S1: Command code	16 bit
	S2: Slave data address	16 bit
	S3: Access points	16 bit
	S4/D: Data storage soft component start address	16 bit

Operand	Operand Object Soft Component																			
	Bit Soft Component							Word Soft Component								Others				
	X	Y	M	T	C	S	D.b	KnX	KnY	KnM	KnS	T	C	D	V, Z	H	K	E	P
S														●	●	●	●		
S1														●	●	●	●		
S2														●	●	●	●		
S3														●	●	●	●		
S4/D	●	●	●			●								●	●	●	●		

Function and Action Description

16-bit Operation (ADPRW)

S1: Command Code	S2: Modbus Slave Data Address	S3: Access Points	S4/D: Data Storage Soft Component Start Address
01H, 02H Bit data readout	0000H ~ FFFFH	1 ~ 2000	Read out the object soft component (start address) Object soft component: D•M•Y•S (for index modification) Occupied word count: (S3 + 15) ÷ 16
03H, 04H Register readout	0000H ~ FFFFH	1 ~ 125	Read out the object soft component (start address) Occupied word count: S3
05H 1 coil write	0000H ~ FFFFH	Reserved	Write the object soft component Object soft component: D·K·H·X·Y·M·S (for index modification) Zero = bit OFF, non-zero = bit ON Occupied word count: 1 point
06H, 41H 1 register write	0000H ~ FFFFH	Reserved	Write the object soft component Object soft component: D·K·H (for index modification) Occupied word count: 1 point
0FH Bulk coil write	0000H ~ FFFFH	1 ~ 1968	Write the object soft component (start address) Object soft component: D·M·X·Y·S (for index modification) Occupied word count: (S3 + 15) ÷ 16
10H, 43H Bulk register write	0000H ~ FFFFH	1 ~ 123	Write the object soft component (start address) Object soft component: D·K·H (for index modification) Occupied word count: S3

Related Soft Components

Soft Component	Name	Content
M8029	Instruction end flag	Turn ON after completing the current communication, until the next instruction using this flag. It can be placed after this instruction to read the communication status or perform communication control.

Note

<table>
<tr><th colspan="2">Note</th></tr>
<tr><td>1</td><td>This command can only be used when the machine is set as the master station. Communication parameters can be configured through special addresses, see Modbus communication function for details.</td></tr>
<tr><td>2</td><td>The communication instructions (EXTR/ADPRW/FROM/TO) are continuously polled from top to bottom in the order of the program step number. The user only needs to turn on the conditions before the communication instruction, without writing logic for polling control.</td></tr>
<tr><td>3</td><td>Communication commands (EXTR/ADPRW/FROM/TO), all communicate in a non-blocking way, polling in the background. Each communication command may occupy several scan cycles. Do not use pulse signals to control communication commands (EXTR/ADPRW /FROM/TO) and ensure that the conduction time is long enough, otherwise the communication command may not be triggered.</td></tr>
<tr><td>4</td><td>If you need to send a single communication command (EXTR/ADPRW/FROM/TO), or judge whether the current communication command is sent successfully, it can be controlled with M8029.</td></tr>
<tr><td>5</td><td>Communication commands (EXTR/ADPRW/FROM/TO) are only allowed to be used in the main program. They cannot be used in the following procedures, otherwise it may cause abnormal communication polling.
<table>
<tr><th>Unusable program flow</th><th>Note</th></tr>
<tr><td>CJ-P instructions</td><td>Conditional jump</td></tr>
<tr><td>FOR-NEXT instructions</td><td>Cycle</td></tr>
<tr><td>P-SRET instructions</td><td>Subroutine</td></tr>
<tr><td>I-IRET instructions</td><td>Interrupt subroutine</td></tr>
</table>
</td></tr>
</table>

Chapter 5 Communication

5.1.1 Function Outline

Provide 2 RS485 communication interfaces MOD1 and MOD2, which can support Modbus master station protocol, Modbus slave station protocol and internal communication protocol.

5.1.2 Special Soft Components

Special Soft Components Supported by MOD1 Port

<table>
<tr><th>Address</th><th>Description</th><th>Default</th></tr>
<tr><td>D8120</td><td>Define MOD1 communication parameters, the default is 0x8089, the specific meaning is shown in the table below.
<table>
<tr><th rowspan="2">Bit Number</th><th rowspan="2">Name</th><th colspan="2">Content</th></tr>
<tr><th>0 (Bit OFF)</th><th>1 (Bit ON)</th></tr>
<tr><td>b0</td><td>Data length</td><td>7 bit</td><td>8 bit</td></tr>
<tr><td>b2&b1</td><td>Parity</td><td colspan="2">00: No parity 01: Odd parity 11: Even parity</td></tr>
<tr><td>b3</td><td>Stop bit</td><td>1 bit</td><td>2 bit</td></tr>
<tr><td>B7&b6&
b5&b4</td><td>Communication rate (bps)</td><td colspan="2">0111: 4800bps 1010: 38400bps 1100: 115200bps
1000: 9600bps 1011: 57600bps Others: 9600bps
1001: 19200bps</td></tr>
<tr><td>b8, b10 - b14</td><td>Reserved</td><td>/</td><td>/</td></tr>
<tr><td>b9</td><td>Protocol</td><td>Modbus</td><td>Internal protocol (slave only)</td></tr>
<tr><td>b15</td><td>Host and slave selection</td><td>Slave</td><td>Host</td></tr>
</table>
The communication parameter setting is recommended to be set in the first execution cycle of the first part of the user program. The default is 0x8089, that is, the data format is 1-8-2, no parity, the baud rate is 9600bps, as the host.
<i>Note: Data length in RTU mode fixed to 8 bits, bit0 is set to 1.</i></td><td>0x8089</td></tr>
<tr><td>D8122</td><td>MOD1 port number (valid when MOD1 port is slave, 0 ~ 255).</td><td>2</td></tr>
<tr><td>D8126</td><td>MOD1 port communication interval time (0 ~ 1000ms): When as the host, the waiting interval from the current communication ends to the next frame communication sends.</td><td>4ms</td></tr>
<tr><td>D8127</td><td>MOD1 port response delay (0 ~ 1000ms): Slave response waiting time (valid when MOD1 port is slave).</td><td>4ms</td></tr>
<tr><td>D8129</td><td>MOD1 port timeout judgment time (ms): When the host is running, it starts timing when sending data. If there is no data reception within D8129, the communication timeout.</td><td>200ms</td></tr>
<tr><td>D8063</td><td>MOD1 port communication error number, as shown below:
<table>
<tr><td>0:</td><td>No meaning, initial value</td><td></td></tr>
<tr><td>1:</td><td>Normal communication</td><td></td></tr>
<tr><td>2:</td><td>Communication timeout</td><td></td></tr>
<tr><td>10:</td><td>Send failed</td><td></td></tr>
<tr><td>11:</td><td>Send data error code</td><td>Illegal function code</td></tr>
<tr><td>12:</td><td>Send data error code</td><td>Illegal data address or data address cross category</td></tr>
<tr><td>13:</td><td>Send data error code</td><td>Illegal data length</td></tr>
<tr><td>101:</td><td>Receive data error code</td><td>Illegal function code</td></tr>
<tr><td>102:</td><td>Receive data error code</td><td>Illegal address</td></tr>
<tr><td>103:</td><td>Receive data error code</td><td>Illegal data</td></tr>
<tr><td>104:</td><td>Receive data error code</td><td>Slave operation failed</td></tr>
<tr><td>122:</td><td>Receive data error code</td><td>Illegal operation</td></tr>
<tr><td>123:</td><td>Receive data error code</td><td>Number of registers incorrect</td></tr>
<tr><td>124:</td><td>Receive data error code</td><td>Information frame error, including length error and check error</td></tr>
<tr><td>132</td><td>Receive data error code</td><td>Parameter read only cannot be modified</td></tr>
<tr><td>133:</td><td>Receive data error code</td><td>Parameter cannot be modified while running</td></tr>
<tr><td>134:</td><td>Receive data error code</td><td>Parameter encryption cannot be modified</td></tr>
<tr><td>140:</td><td>Receive data error code</td><td>Sending and receiving station addresses are inconsistent (host)</td></tr>
<tr><td>141:</td><td>Receive data error code</td><td>Sending and receiving command codes are inconsistent (host)</td></tr>
<tr><td>142:</td><td>Receive data error code</td><td>Sending and receiving start address are inconsistent (host)</td></tr>
</table></td><td></td></tr>
</table>

Address	Description	Default
	2xx: When the host communication receives the slave return error code, command frame will display 200+ exception code (if received 0x01 0x86 0x03 0x02 0x61, it will display 203)	
M8063	MOD1 port communication error flag: The communication flag is set after the communication is completed or an error occurs, and continues until the next communication starts.	
M8123	MOD1 port communication completion flag: The communication flag is set after the communication is completed or an error occurs, and continues until the next communication starts. Note: Do not use the M8123 communication completion flag to start the next communication, and timing errors may occur.	

Special Soft Components Supported by MOD2 Port

<table>
<tr><th>Address</th><th>Description</th><th>Default</th></tr>
<tr><td>D8400</td><td>Define MOD2 communication parameters, the default is 0x8089, the specific meaning is shown in the table below.
<table>
<tr><th rowspan="2">Bit Number</th><th rowspan="2">Name</th><th colspan="2">Content</th></tr>
<tr><th>0 (Bit OFF)</th><th>1 (Bit ON)</th></tr>
<tr><td>b0</td><td>Data length</td><td>7 bit</td><td>8 bit</td></tr>
<tr><td>b2&b1</td><td>Parity</td><td colspan="2">00: No parity 01: Odd parity 00: Even parity</td></tr>
<tr><td>b3</td><td>Stop bit</td><td>1 bit</td><td>2 bit</td></tr>
<tr><td>B7&b6&
b5&b4</td><td>Communication rate (bps)</td><td colspan="2">0111: 4800bps 1010: 38400bps 1100: 115200bps
1000: 9600bps 1011: 57600bps Others: 9600bps
1001: 19200bps</td></tr>
<tr><td>b8, b10 - b14</td><td>Reserved</td><td>/</td><td>/</td></tr>
<tr><td>b9</td><td>Protocol</td><td>Modbus</td><td>Internal protocol (slave only)</td></tr>
<tr><td>b15</td><td>Master and slave selection</td><td>Slave</td><td>Master</td></tr>
</table>
The communication parameter setting is recommended to be set in the first execution cycle of the first part of the user program. The default is 0x8089, that is, the data format is 1-8-2, no parity, the baud rate is 9600bps, as the host.
<i>Note: Data length in RTU mode fixed to 8 bits, bit0 is set to 1.</i></td><td>0x8089</td></tr>
<tr><td>D8402</td><td>MOD2 port number (valid when MOD2 port is slave, 0 ~ 255).</td><td>2</td></tr>
<tr><td>D8406</td><td>Communication interval time (0 ~ 1000ms): When as the host, the waiting interval from the current communication ends to the next frame communication sends.</td><td>4ms</td></tr>
<tr><td>D8407</td><td>MOD2 port response delay (0 ~ 1000ms): Slave response waiting time (valid when MOD2 port is slave).</td><td>4ms</td></tr>
<tr><td>D8409</td><td>MOD2 port timeout judgment time (ms): When the host is running, it starts timing when sending data. If there is no data reception within D8129, the communication timeout.</td><td>200ms</td></tr>
<tr><td>D8438</td><td>MOD1 port communication error number, as shown below:
<table>
<tr><td>0:</td><td>No meaning, initial value</td><td></td></tr>
<tr><td>1:</td><td>Normal communication</td><td></td></tr>
<tr><td>2:</td><td>Communication timeout</td><td></td></tr>
<tr><td>10:</td><td>Send failed</td><td></td></tr>
<tr><td>11:</td><td>Send data error code</td><td>Send data error code</td></tr>
<tr><td>12:</td><td>Send data error code</td><td>Send data error code</td></tr>
<tr><td>13:</td><td>Send data error code</td><td>Illegal data length</td></tr>
<tr><td>101:</td><td>Receive data error code</td><td>Illegal function code</td></tr>
<tr><td>102:</td><td>Receive data error code</td><td>Illegal address</td></tr>
<tr><td>103:</td><td>Receive data error code</td><td>Illegal data</td></tr>
<tr><td>104:</td><td>Receive data error code</td><td>Slave operation failed</td></tr>
<tr><td>122:</td><td>Receive data error code</td><td>Illegal operation</td></tr>
<tr><td>123:</td><td>Receive data error code</td><td>Number of registers incorrect</td></tr>
<tr><td>124:</td><td>Receive data error code</td><td>Information frame error, including length error and check error</td></tr>
<tr><td>132:</td><td>Receive data error code</td><td>Parameter read only cannot be modified</td></tr>
<tr><td>133:</td><td>Receive data error code</td><td>Parameter cannot be modified while running</td></tr>
<tr><td>134:</td><td>Receive data error code</td><td>Parameter encryption cannot be modified</td></tr>
<tr><td>140:</td><td>Receive data error code</td><td>Sending and receiving station addresses are inconsistent (host)</td></tr>
<tr><td>141:</td><td>Receive data error code</td><td>Sending and receiving command codes are inconsistent (host)</td></tr>
<tr><td>142:</td><td>Receive data error code</td><td>Sending and receiving start address are inconsistent (host)</td></tr>
</table></td><td></td></tr>
</table>

Address	Description	Default
	2xx: When the host communication receives the slave return error code, command frame will display 200+ exception code (if received 0x01 0x86 0x03 0x02 0x61, it will display 203)	
M8438	MOD2 port communication error flag: The communication flag is set after the communication is completed or an error occurs, and continues until the next communication starts.	
M8403	MOD2 port communication completion flag: The communication flag is set after the communication is completed or an error occurs, and continues until the next communication starts. Note: Do not use the M8403 communication completion flag to start the next communication, and timing errors may occur.	

5.1.3 Modbus Function

Bit 9 of D8120 (MOD1) or D8400 (MOD2) takes 0 to enable Modbus communication.

Modbus Function Code

Command Code	Meaning
0x01, 0x02	Read one or more bits, range 1 ~ 2000
0x03, 0x04	Read one or more registers, range 1 ~ 125
0x05	Write a bit, range 1
0x06, 0x41	Write a register, range 1
0x0F	Write multiple bits, range 1 ~ 1968
0x10, 0x43	Write multiple registers, range 1 ~ 123

Modbus Soft Component Address

Modbus Communication Bit Component Address Number		Modbus Communication Word Component Address Number	
Bit Component	Address Number (16 bit)	Register	Address Number
M0 ~ M7679	0x0000 ~ 0x1DFF	D0 ~ D7999	0x0000 ~ 0x1F3F
M8000 ~ M8511	0x1E00 ~ 0x1FFF	D8000 ~ D8511	0x1F40 ~ 0x213F
S0 ~ S4095	0x2000 ~ 0x2FFF	TN0 ~ TN511	0xA140 ~ 0xA33F
TS0 ~ TS511	0x3000 ~ 0x31FF	CN0 ~ CN199	0xA340 ~ 0xA407
CS0 ~ CS255	0x3200 ~ 0x32FF	CN200 ~ CN255 (32bit occupies 2 addresses)	0xA408 ~ 0xA477
Y0 ~ Y377	0x3300 ~ 0x33FF	M0 ~ M7679	0xA478 ~ 0xA657
X0 ~ X377	0x3400 ~ 0x34FF	M8000 ~ M8511	0xA658 ~ 0xA677
		S0 ~ S4095	0xA678 ~ 0xA777
		TS0 ~ TS511	0xA778 ~ 0xA797
		CS0 ~ CS255	0xA798 ~ 0xA7A7
		Y0 ~ Y377	0xA7A8 ~ 0xA7B7
		X0 ~ X377	0xA7B8 ~ 0xA7C7

Host

When HC10 is used as the host, please configure special soft comoonent first, and then communicate through the Modbus read and write instruction ADPRW (see the instruction "Description" for more details).

HC10 will automatically poll the ADPRW instruction which is conditionally connected according to the program execution order to communicate.

Slave

When the slave communicates, you only need to configure special soft comoonent (communication format, station number, etc.) to communicate.

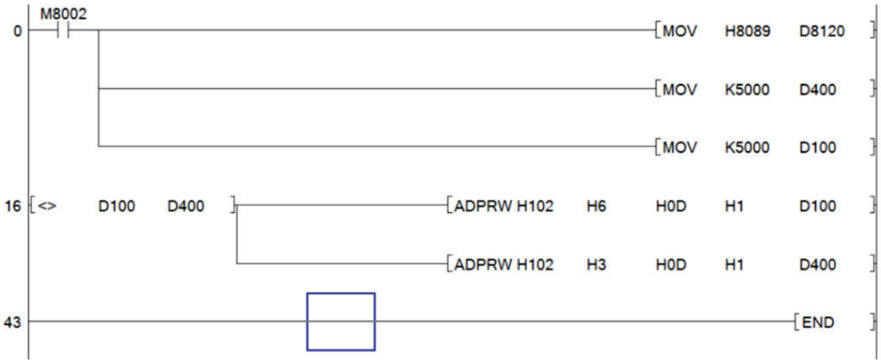
For supported command words and soft comoonents address mapping, please see the Modbus soft comoonent address table. Continuous read and write operations are not allowed across address types.

Program Example

Case 1: Communication between HC10 as a Host and an HD30 Inverter

The MOD1 port of HC10 is used as the host to set the frequency of an inverter, and the frequency of the inverter is set by D100. Only when the set frequency of D100 changes, the communication is written.

After the writing is completed, the data is read and judged. If the writing is successful, the writing is stopped. If the writing fails, the writing will continue.



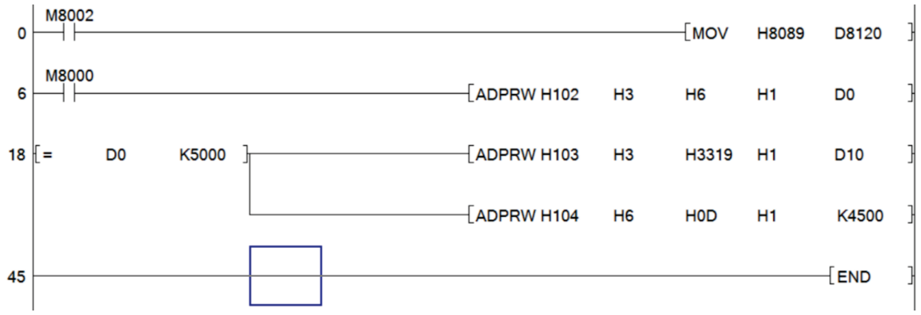
```
   M8002
0 ─┤├─┬──────────────────────────[MOV    H8089  D8120 ]
      ├──────────────────────────[MOV    K5000  D400  ]
      └──────────────────────────[MOV    K5000  D100  ]
16 [<>  D100  D400 ]─┬───────[ADPRW H102  H6  H0D  H1  D100 ]
                     └───────[ADPRW H102  H3  H0D  H1  D400 ]
43 ──────────────────────────────────────────────[END ]
```

Execution Steps:	
1	Set the MOD1 port communication parameter D8120 through the initial pulse M8002. The 0x8089 set in the figure above is the default value (that is, the data format is 1-8-2, no parity, and the baud rate is 9600bps, as the host). • If you need to use other communication parameters for communication, please refer to the MOD1 special soft component table to set the corresponding settings for D8120. • The host can also set the communication interval D8126 and D8129. If there is no special requirement, it can generally be set to the default value, here is the default.
2	D400 is used to save the actual frequency of the inverter, and D100 is the frequency set by the user. D400 and D100 are given an initial value of 5000 by powering on M8002, which makes it consistent with the factory default value of HD30 inverter 50.00Hz.
3	When the value of D100 is changed, the communication conditions are connected to change the frequency. For example: To set the frequency to 45.00Hz, you need to set D100 to 4500.
4	The set frequency D100 of the inverter is inconsistent with the current frequency D400. Control the ADPRW instruction to write (command word H6, write register) the set frequency (corresponding address 0x000D) of the inverter from MOD1 slave 2 (H102) to 4500. Communication data frame: • HC10→HD30, HC10 transmission: 02 06 000d119415c5 • HD30→HC10, HC10 receiving: 02 06 000d119415c5 The writing is successful, and the running frequency of the inverter is changed to 45.00Hz.
5	The set frequency D100 of the inverter is inconsistent with the current frequency D400. Read (command word H3, read the register) the frequency of the inverter (corresponding to address 0x000D) to D400 through the ADPRW instruction from MOD1 slave 2 (H102). Communication data frame: • HC10→HD30, HC10 transmission: 02 03 00 0d 00 01 15 fa • HD30→HC10, HC10 receiving: 02 03 02 1194f1 bb Read the inverter running frequency successfully, D400 was changed to 5000.
6	After rewriting the frequency of the inverter, the conditions will be disconnected again. Wait for the next D100 value to change, and then perform the communication to change the frequency.

Case 2: Communication between HC10 as a Host and Three HD30 Inverters

HC10 as the host, reads the Max. output frequency of the first HD30 inverter (slave 2) to D0 through the MOD1 port, and determines whether the Max. output frequency of the first HD30 inverter is equal to 50.00Hz. If it is equal to 50.00Hz, read the DC bus voltage of the second HD30 inverter (slave 3) to D10, and set the frequency of the third HD30 inverter (slave 4) to 45.00Hz.

HD30 inverter is set according to the default communication parameters, that is, the communication format is 9600bps, 1-8-2 format, no verification, RTU mode. Station number is 2, 3, 4. The ladder diagram programming of HC10 host is as follows:



Execution Steps:	
1	Set the communication parameter D8120 through the initial pulse M8002. The 0x8089 set here is the default value (that is, data format 1-8-2, no parity, baud rate 9600bps, as the host). • If you need to use other communication parameters for communication, please refer to the MOD1 special soft component table to set the D8120 accordingly. • The host can also set the communication interval D8126 and D8129. If there is no special requirement, it can generally be set to the default value, here is the default.
2	The ADPRW instruction is controlled by the RUN monitor M8000 to read (command word H3, read the register) the Max. output frequency of the first inverter (address 0x0006) from the MOD1 slave 2 (H102) to D0, and the length is H1. Communication data frame: • HC10→HD30, HC10 transmission: 02 03 00 06 00 01 64 38 • HD30→HC10, HC10 receiving: 02 03 02 13 88 f1 12 The reading is successful. The Max. output frequency of the first inverter is 5000, which is 50.00Hz.
3	By judging that the Max. output frequency of the first inverter is 50.00Hz, control the ADPRW instruction to read (command word H3, read the register) the DC bus voltage (corresponding address 0x3319) of the second inverter from the MOD1 port slave station 3 (H103) to D10, and the length is H1. Communication data frame: • HC10→HD30, HC10 transmission: 03 03 33 19 00 01 5b 6b • HD30→HC10, HC10 receiving: 03 03 02 02 19 1 2e The reading is successful, and the DC bus voltage of the second inverter is 537V.
4	By judging that the Max. output frequency of the first inverter is 50.00Hz, control the ADPRW instruction to write (command word H6, write register) the set frequency (corresponding address 0x000D) of the third inverter from MOD1 slave 4 (H104) to 45.00Hz and the length to H1. Communication data frame: • HC10→HD30, HC10 transmission: 04 06 00 0d 11 94 15 a3 • HD30→HC10, HC10 receiving: 04 06 00 0d 11 94 15 a3 The writing is successful, and the set frequency of the third inverter is 4500, which is 45.00Hz.

5.2 CAN Communication Function

5.2.1 Fuction Outline

Provide 1 CAN communication interface:

- Support CAN protocol of CAN2.0A and CAN2.0B versions. Provide Hpmont connection protocol (for details) and free port protocol (only communicate with one of the protocol at the same time).
- A 120Ω matching resistor has been connected to the CAN interface. When wiring, you only need to connect CAN+ and CAN- to each other to complete the CAN communication wiring.

5.2.2 Connection Protocol

The connection protocol consists of two types of data frames, including access data frames (ADF for short) and quick data frames (QDF for short). Users can use it alone or at the same time for CAN communication.

Connection Protocol Special Soft Component

<table>
<tr><th>Address</th><th>Description</th></tr>
<tr><td>D8470</td><td>Define CAN communication parameters, the default value is 0xA005. The specific meaning is shown in the table below.
<table>
<tr><th rowspan="2">Bit Number</th><th rowspan="2">Name</th><th colspan="3">Content</th></tr>
<tr><th colspan="2">0 (Bit OFF)</th><th>1 (Bit ON)</th></tr>
<tr><td>b3&b2&
b1&b0</td><td>Baud rate</td><td>0000: 5kbps
0001: 10kbps
0010: 20kbps</td><td>0011: 50kbps
0100: 100kbps
0101: 125kbps</td><td>0110: 250kbps
0111: 500kbps
1000: 1Mbps</td></tr>
<tr><td>b4 ~ b10</td><td>Slave node number</td><td colspan="3">Slave node number (1 ~ 127, 0 is broadcast frame)</td></tr>
<tr><td>b11, b14</td><td>Reserved</td><td colspan="2">/</td><td>/</td></tr>
<tr><td>b12</td><td>Format</td><td colspan="2">CAN2.0A (11-bit identifier)</td><td>CAN2.0B (29-bit identifier)</td></tr>
<tr><td>b13</td><td>Host-slave selection</td><td colspan="2">Slave</td><td>Host</td></tr>
<tr><td>b15</td><td>Protocol</td><td colspan="2">Freeport protocol</td><td>Connection protocol</td></tr>
</table>
The communication parameter defaults 0xA005. It can be configured through the host computer, and the parameter will be saved after power-off.
<i>Note:</i>
<i>1. Host -slave selection is only valid under the connection protocol.</i>
<i>2. The slave node number is valid only when it is selected as a slave under the connection protocol.</i>
<i>3. The connection protocol is fixed using CAN2.0A, and the format setting is only valid in the free port protocol.</i></td></tr>
<tr><td>D8471</td><td>CAN timeout time (only valid when there is host under connection protocol, default 20ms)</td></tr>
<tr><td>D8473</td><td>ADF sending interval time (0 ~ 1000ms, default 10ms)</td></tr>
<tr><td>D8474</td><td>QDF sending interval time (0 ~ 1000m, default 2ms)</td></tr>
<tr><td>D8475</td><td>CAN communication error number, see below:
<table>
<tr><td>0:</td><td>No meaning, initial value</td><td></td></tr>
<tr><td>1:</td><td>Normal communication</td><td></td></tr>
<tr><td>2:</td><td>Communication timeout</td><td></td></tr>
<tr><td>10:</td><td>Send failed</td><td></td></tr>
<tr><td>11:</td><td>Send data error code</td><td>Illegal function code</td></tr>
<tr><td>12:</td><td>Send data error code</td><td>Illegal data address or data address cross category</td></tr>
<tr><td>13:</td><td>Send data error code</td><td>Illegal data length</td></tr>
<tr><td>101:</td><td>Received data error code</td><td>Illegal function code</td></tr>
<tr><td>102:</td><td>Received data error code</td><td>Illegal address</td></tr>
<tr><td>103:</td><td>Received data error code</td><td>Illegal data</td></tr>
<tr><td>104:</td><td>Received data error code</td><td>Slave operation failed</td></tr>
<tr><td>122:</td><td>Received data error code</td><td>Illegal operation</td></tr>
<tr><td>123:</td><td>Received data error code</td><td>Register number error</td></tr>
<tr><td>124:</td><td>Received data error code</td><td>Information frame error, including length error and check error</td></tr>
<tr><td>132:</td><td>Received data error code</td><td>The parameter is read only and cannot be modified</td></tr>
<tr><td>133:</td><td>Received data error code</td><td>Parameter cannot be modified during operation</td></tr>
<tr><td>134:</td><td>Received data error code</td><td>Parameter encryption cannot be modified</td></tr>
</table></td></tr>
</table>

Address	Description
	140: Received data error code Receive and send station addresses are inconsistent (host) 141: Received data error code Receive and send command codes are inconsistent (host) 142: Received data error code Receive and send start addresses are inconsistent (host) 2xx: When the host communication receives the error code from the slave, the command frame will display 200+ exception code. The exception code content is returned by the slave
D8476	QDF error station number (host)
D8481	QDF1 sending data storage address (slave)
D8482	QDF1 receiving data storage address (slave)
D8484	QDF2 sending data storage address (slave)
D8485	QDF2 receiving data storage address (slave)
D8487	QDF3 sending data storage address (slave)
D8488	QDF3 receiving data storage address (slave)
M8471	CAN communication completion flag
M8475	CAN communication error flag
M8476	QDF host communication error flag
M8480	QDF1 enable flag
M8481	QDF1 communication success flag
M8483	QDF2 enable flag
M8484	QDF2 communication success flag
M8486	QDF3 enable flag
M8487	QDF3 communication success flag

5.2.3 ADF Connection Protocol

ADF Communication Function

ADF uses 1 host multi-slave mode for communication. The host sends data to the slave, and the slave returns after receiving the data.

The ADF data frame includes an 11-bit identifier and an 8-bit data field. The data field contains the command code, the number of registers, the high bit of register start address, the low bit of register start address, and the data content.

ADF Data Frame Format

Data Frame Format

11-bit Identifier		Data Field (8 Byte is fixed)				
bit10 ~ 7	bit6 ~ 0	byte0	byte1	byte2	byte3	byte4 ~ 7
Frame ID	Node address	Command code	The high bit of register start address	The low bit of register start address	The number of registers	—

Frame ID	Distinguish between different communication objects 1100b host accesses the node's register, 1011b node responds to the host
Node Address	The slave node number of this communication 1 ~ 127, 0 is broadcast frame
Command Code	0x03 (register read) 0x10 (register write) 0x01 (bit read) 0x0F (bit write)
The High Bit of Register Start Address	The high bit of register start address
The Low Bit of Register Start Address	The low bit of register start address
The Number of Registers	Number of data requested to be read or written, register type is 1 to 2, bit type is 1 to 32
—	Data content

CAN Soft Component Address

CAN Communication Bit Component Address Number		CAN Communication Word Component Address Number	
Bit Component	**Address Number (16 Bit)**	**Register**	**Address Number**
M0 ~ M7679	0x0000 ~ 0x1DFF	D0 ~ D7999	0x0000 ~ 0x1F3F
M8000 ~ M8511	0x1E00 ~ 0x1FFF	D8000 ~ D8511	0x1F40 ~ 0x213F
S0 ~ S4095	0x2000 ~ 0x2FFF	TN0 ~ TN511	0xA140 ~ 0xA33F
TS0 ~ TS511	0x3000 ~ 0x31FF	CN0 ~ CN199	0xA340 ~ 0xA407
CS0 ~ CS255	0x3200 ~ 0x32FF	CN200 ~ CN255 (32bit occupies 2 addresses)	0xA408 ~ 0xA477
Y0 ~ Y377	0x3300 ~ 0x33FF	M0 ~ M7679	0xA478 ~ 0xA657
X0 ~ X377	0x3400 ~ 0x34FF	M8000 ~ M8511	0xA658 ~ 0xA677
		S0 ~ S4095	0xA678 ~ 0xA777
		TS0 ~ TS511	0xA778 ~ 0xA797
		CS0 ~ CS255	0xA798 ~ 0xA7A7
		Y0 ~ Y377	0xA7A8 ~ 0xA7B7
		X0 ~ X377	0xA7B8 ~ 0xA7C7

ADF Communication Usage

ADF communication needs to assign a separate node number to each slave device, which is a master-slave mode.

- When the host, you need to configure the communication parameters D8470 (protocol type, slave node number and baud rate), and then use the EXTR (only continuous type single word form) instruction for communication. For EXTR usage, please refer to the corresponding instruction description. The EXTR instruction that needs to be sent should be always on, and the multiple-on EXTR will be automatically polled from front to back according to the scanning order.
- When acting as a slave, you only need to configure the communication parameters D8470 (protocol type, local node number and baud rate) to communicate.
- D8471 is the CAN timeout time. If the host communication does not receive a return frame after this time, it will report the communication timeout and switch to the next frame.
- D8473 is the ADF sending interval. When using ADF and QDF at the same time, please do not change the value to 0, otherwise it will affect the communication speed of QDF.
- D8475 is the CAN communication error number. When a communication error occurs, the value can be read to determine the type of error.
- As a broadcast frame, the host only sends data and does not receive data; The slave only receives data and does not send data.
- The EXTR instruction supports the M8029 end flag, which can be used to judge the completion status of each communication.

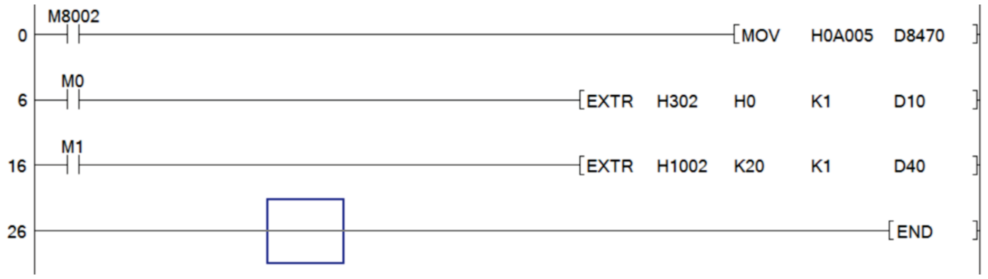
Program Example:

There is a host-slave communication between two HC10.

The host turns on through M0, reads the data from the D0 register of the slave to D10. The value written into the D40 register of the master through M1 is turned on to the D20 register of the slave.

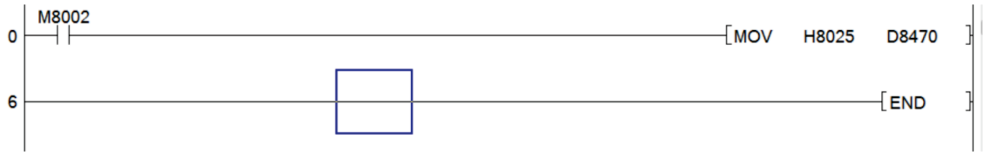
Connect the cable before use: CAN+ of HC10 host must be connected to CAN+ of the slave, CAN- of HC10 host must be connected to CAN- of the slave.

Host programming:



```
0   M8002  [MOV   H0A005  D8470     ]
6   M0     [EXTR  H302    H0    K1  D10 ]
16  M1     [EXTR  H1002   K20   K1  D40 ]
26         [END                     ]
```

Slave programming:



```
0   M8002  [MOV   H8025  D8470 ]
6          [END                ]
```

Execution Steps:	
1	The host sets the communication parameter D8470 to HA005 through M8002, that is, the connection protocol is used as the host, and the baud rate is 125k.
2	The slave sets the communication parameter D8470 to H8025 through M8002, that is, the connection protocol is used as the slave, the slave node number is 2, and the baud rate is 125k.
3	The HC10 host controls the EXTR instruction by connecting M0, and reads the data from the D0 register of the HC10 slave to the D10 register of the host. H302 represents the register read, and the read slave address is 0x02; H0 means the slave data address is 0x0000 (map slave register D0); K1 indicates that the access is a word; D10 indicates that the object soft component is written as D10.
4	The first communication command reads the data from the slave D0 and saves it to the host D10. Set the value of the HC10 slave D0 register to 100, M0 is connected, and the EXTR instruction is executed. The HC10 host reads the value of the HC10 slave D0 register 100 and places it in the HC10 host D10 register with a length of one word. Communication data frame: • Host→slave, data frame: 60203 00 00 01 00 00 00 00 • Slave→host, data frame: 58203 00 00 02 00 64 00 00
5	The second communication command is for the HC10 host to write D40 data to the slave D20 register. M1 is connected, EXTR instruction is executed, H1002 means register write, the slave address written is 0x02; K20 means slave data address is 0x0020 (map slave register D20); K1 means access is a word; D40 write The object soft component is D40, and the D40 register of the HC10 host is set to 500. That is, the value written by the HC10 host to the HC10 slave D20 register is 500, and the length is one word. Communication data frame: • Host→slave, data frame: 60210 00 14 01 01 F4 00 00 • Slave→host, data frame: 58210 00 14 0101 F4 00 00

5.2.4 QDF Connection Protocol

QDF Communication Function

QDF also uses a host-slave mode for communication, but unlike ADF, the data content transmitted by QDF is data, does not contain control command words, and is used for agreed paired data exchange.

When the HC10 is used as the host, the QDF communication data table can be configured through the HCStudio host computer, and it will automatically poll the communication in the background when it is running, regardless of the scan cycle.

When HC10 is a slave, it cannot actively send data, but can only respond to data reception. By enabling the corresponding receiving mailbox, the slave receives the data sent by the host, and then sends the set data to the host.

QDF Data Frame Format

Data Frame Format

11-bit Identifier		Data Field (8 Byte is Fixed)
Bit10 ~ 7	Bit6 ~ 0	Byte0 ~ 7
Frame ID	Node address	—

Frame ID	Host sends QDRF to modify slave data, slave sends QDAF to upload data	
	QDRF1: 0011b	QDAF1: 0100b
	QDRF2: 0101b	QDAF2: 0110b
	QDRF3: 0111b	QDAF3: 1000b
Node Address	The slave node number of this communication • 1 ~ 127, 0 is broadcast frame	
—	Data content	

QDF Communication Usage

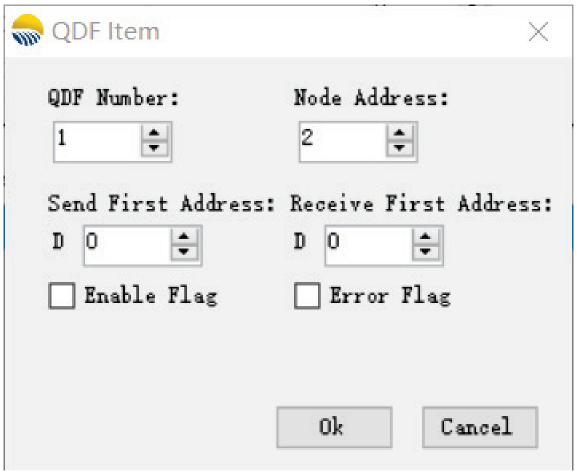
QDF can use up to three groups of mailboxes (each group contains one sending mailbox QDRF and one receiving mailbox QDAF). The host and slave mailboxes correspond one-to-one. For example, the host's QDF1 mailbox can only correspond to the slave's QDF1 mailbox.

- QDF adopts 1 host multi-slave mode for communication. The host initiates communication and the slave responds to communication.
- The QDF host automatically polls the communication table configured by the host computer in the background, and supports up to 50 entries.
- The QDF slave cannot actively initiate communication. When receiving the QDRF request frame sent by the host, it will reply with the corresponding QDAF return frame. When HC10 is used as a slave, you first need to configure the baud rate, protocol and node number, and then set it up to send (D8481, D8484, D8487) and receive (D8482, D8485, D8488) data mapping address, and enable the corresponding mailbox (M8480, M8483, M8486). The data sent and received will occupy the 4 consecutive starting D address corresponding to the set address single word. If D8481 is set to 10, it means D10 ~ D13 are used to store the sending data of QDAF1.
- D8471 is the CAN timeout time. If the host has not completed the communication after this time has passed since the start of communication, it will be considered that the communication has failed and the next communication will be started.
- D8474 is the QDF sending interval.
- When the QDF master communication error occurs, M8476 will be set, D8476 will store the slave station number of the communication failure, and if there are multiple node errors, the node number with the smallest number will be stored. 0 means transmission failure. D8475 does not display QDF master errors.

- As a broadcast frame, the master only sends data and does not receive data; All slaves will receive data and do not return data.

QDF Communication Configuration

The content of each communication of the QDF host is set as follows:



QDF Item

QDF Number: 1

Node Address: 2

Send First Address: D 0

Receive First Address: D 0

□ Enable Flag

□ Error Flag

- QDF Number: Setting range 1 ~ 3, corresponding to QDF1 ~ QDF3.
- Node address: The setting range is 0 ~ 127, 1 ~ 127 corresponds to the QDF target slave node address, and 0 is a broadcast frame.
- First address of sending data: Setting range D0 ~ D7996, 4 consecutive D registers starting from the first address are mapped as sending mailboxes.
- First address of receiving data: Setting range D0 ~ D7996, 4 consecutive D registers starting from the first address are mapped as receiving mailboxes.
- Disabled flag bit: Enable after ticking, the setting range is M0 ~ M7679, when the set M bit is ON, it is disabled, and when it is OFF, it is enabled. This communication frame is always enabled when it is not checked.
- Communication error flag: Enable after ticking. The setting range is M0 ~ M7679. When an error occurs in this communication frame, the set M position is ON, and it is turned OFF when the communication is normal.

5.2.5 Free Port Protocol

The free port protocol allocates two receiving mailboxes which can set filters and one sending mailbox. And the user can program CAN for sending and receiving.

Support CAN2.0A (11-bit identifier) and CAN2.0B (29-bit identifier).

Free Port Protocol Special Soft Component

<table>
<tr><th>Address</th><th>Description</th></tr>
<tr><td>D8470</td><td>Define CAN communication parameters and the default value is 0xA005. The specific meaning is shown in the table below.
<table>
<tr><th rowspan="2">Bit Number</th><th rowspan="2">Name</th><th colspan="2">Content</th></tr>
<tr><th>0 (Bit OFF)</th><th>1 (Bit ON)</th></tr>
<tr><td>b3&b2&
b1&b0</td><td>Baud rate</td><td colspan="2">0010: 20kbps 0101: 125kbps 0111: 500kbps
0011: 50kbps 0110: 250kbps 1000: 1Mbps
0100: 100kbps</td></tr>
<tr><td>b4 ~ b10</td><td>Slave node number</td><td colspan="2">Slave node number (1 ~ 127, 0 is broadcast frame)</td></tr>
<tr><td>b11, b14</td><td>Reserved</td><td>/</td><td>/</td></tr>
<tr><td>b12</td><td>Format</td><td>CAN2.0A (11-bit identifier)</td><td>CAN2.0A (11-bit identifier)</td></tr>
<tr><td>b13</td><td>Host-slave selection</td><td>Slave</td><td>Slave</td></tr>
<tr><td>b15</td><td>Protocol</td><td>Free port protocol</td><td>Free port protocol</td></tr>
</table>
The communication parameter setting is recommended to be set during the first execution cycle of the first part of the user program. The default value is 0xA005 (that is, the connection protocol is used as the host, and the baud rate is 125k).
The free port protocol is not running by default. If you want to use the free port protocol, the communication parameters need to be reconfigured.
<i>Note:</i>
<i>1. Host-slave selection is only valid under the connection protocol.</i>
<i>2. The slave node number is valid only when it is selected as a slave under the connection protocol.</i>
<i>3. The connection protocol is fixed using CAN2.0A, and the format setting is only valid in the free port protocol.</i></td></tr>
<tr><td>D8471</td><td>CAN timeout time (only valid when it's host under connection protocol, default 20ms)</td></tr>
<tr><td>D8475</td><td>CAN communication error number, see below:
<table>
<tr><td>0:</td><td>Meaningless, initial value</td><td></td></tr>
<tr><td>1:</td><td>Normal communication</td><td></td></tr>
<tr><td>2:</td><td>Communication timeout</td><td></td></tr>
<tr><td>10:</td><td>Send data error code</td><td>Illegal function code</td></tr>
<tr><td>11:</td><td>Send data error code</td><td>Send data length error</td></tr>
<tr><td>12:</td><td>Send data error code</td><td>Illegal data address</td></tr>
<tr><td>13:</td><td>Send data error code</td><td>Illegal data length</td></tr>
<tr><td>101:</td><td>Receive error code</td><td>Illegal command code</td></tr>
<tr><td>102:</td><td>Receive error code</td><td>Illegal register address</td></tr>
<tr><td>103:</td><td>Receive error code</td><td>Data error</td></tr>
<tr><td>122:</td><td>Receive error code</td><td>Unsupported operation (attribute, factory value, upper and lower limits are not supported)</td></tr>
<tr><td>123:</td><td>Receive error code</td><td>Register in request frame</td></tr>
<tr><td>124:</td><td>Receive error code</td><td>Message frame error, including message length error and check error</td></tr>
<tr><td>132</td><td>Receive error code</td><td>Parameter cannot be modified</td></tr>
<tr><td>133:</td><td>Receive error code</td><td>Parameter cannot be modified while running</td></tr>
<tr><td>134:</td><td>Receive error code</td><td>Parameter is password protected</td></tr>
<tr><td>140:</td><td>Receive error code</td><td>The address of the receiving data station and sending data station are inconsistent (host communication)</td></tr>
<tr><td>141:</td><td>Receive error code</td><td>The receive data command code and send data command code are inconsistent (host communication)</td></tr>
<tr><td>2xx:</td><td colspan="2">When the host communication receives the error code returned from the slave, the command frame will display 200+ exception codes, and the contents of the exception code are returned by the slave</td></tr>
</table></td></tr>
<tr><td>D8479</td><td>Send data start address</td></tr>
</table>

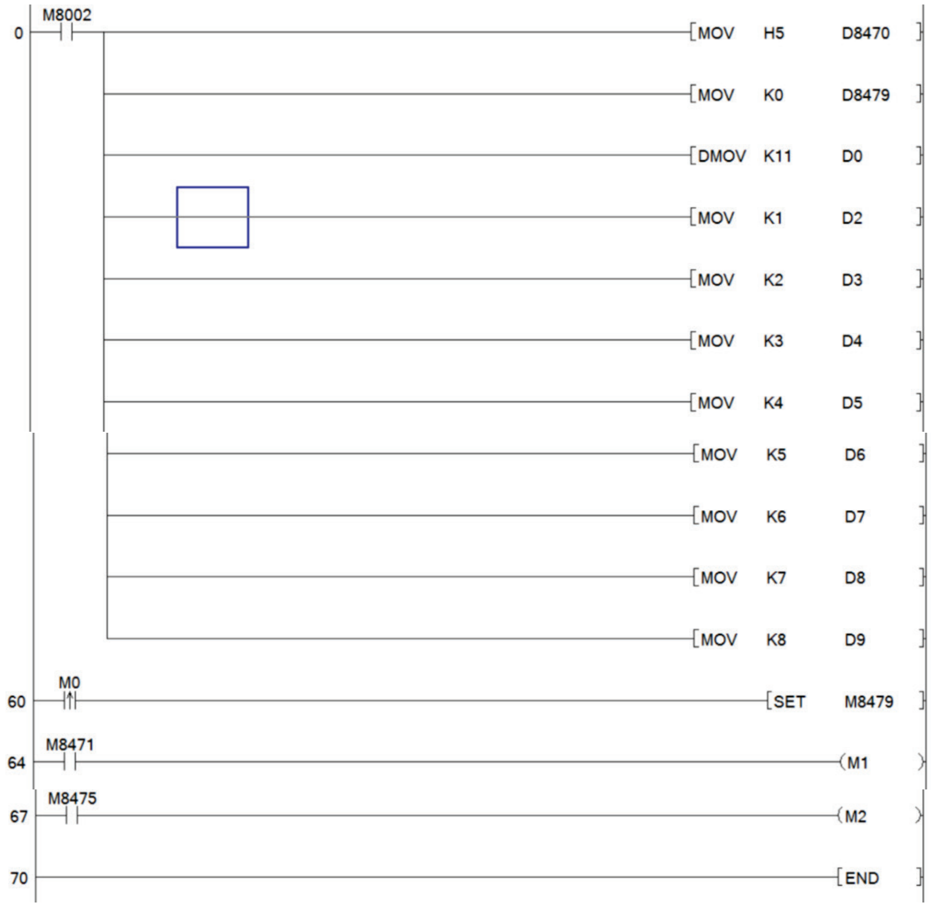
Address	Description
D8480	Receiving mailbox 0 identifier 1/L
D8481	Receiving mailbox 0 identifier 2/H
D8482	Receiving mailbox 0 mask code 1/L
D8483	Receiving mailbox 0 mask code 2/H
D8484	Receive mailbox 0 data start address
D8485	Receiving mailbox 1 identifier 1/L
D8486	Receiving mailbox 1 identifier 2/H
D8487	Receiving mailbox 1 mask code 1/L
D8488	Receiving mailbox 1 mask code 2/H
D8489	Receive mailbox 1 data start address
M8471	CAN communication completion flag
M8475	CAN communication error flag
M8479	Send data command
M8484	Mailbox 0 received data flag
M8489	Mailbox 1 received data flag

Data Transmission

D8479 is used to specify the starting address for sending data (only D variables can be specified), and the length is 10 consecutive data. If D8479 is set to 200, then D200 ~ D209 are used to store CAN sending data.

- 32 bits composed of D200 and D201 are used to store identifiers (CAN2.0A takes the lower 11 bits, CAN2.0B takes the lower 29 bits).
- The lower eight bits of D202 ~ D209 are used to store the 8-byte data of CAN. The upper eight bits are invalid.
- Start sending by setting M8479. If the sending mailbox is idle, put the CAN communication message into the mailbox to wait for sending and M8479 will be turned off. If the mailbox is occupied, wait for the mailbox to be idle, and M8479 status will not change.
- M8471 is set for successful data transmission, M8475 is set for failed data transmission and the error type is set to D8475.
- When preparing the data identifier, please note that the upper 7 bits of the CAN identifier are forbidden according to the CAN protocol (that is, the bit is 1).

Program Example: Send a CAN2.0A Data Frame Using the Free Port Protocol.



```
    M8002
0 ──┤ ├──┬──[MOV   H5    D8470]
         ├──[MOV   K0    D8479]
         ├──[DMOV  K11   D0   ]
         ├──[MOV   K1    D2   ]
         ├──[MOV   K2    D3   ]
         ├──[MOV   K3    D4   ]
         ├──[MOV   K4    D5   ]
         ├──[MOV   K5    D6   ]
         ├──[MOV   K6    D7   ]
         ├──[MOV   K7    D8   ]
         └──[MOV   K8    D9   ]
    M0
60──┤↑├─────[SET   M8479]
    M8471
64──┤ ├─────(M1)
    M8475
67──┤ ├─────(M2)
70──────────[END]
```

Execution Steps:	
1	Set the communication parameter D8470 to 0x0005 through M8002 (that is, the free port protocol is used, the baud rate is 125k, and CAN2.0A is used).
2	Set send data address mapping 10, that is, fill in the send input in D10 ~ D19.
3	Set the sending data message, the 11-bit identifier is 11, and the 8-byte data message is 1, 2, 3, 4, 5, 6, 7, 8 in turn.
4	M8479 is set by the rising edge pulse of M0 to start a transmission.

Data Reception

The CAN free port protocol is assigned two receiving mailboxes, each mailbox has a 32-bit identifier and mask code.

- CAN2.0A (11-bit standard identifier) can be configured with 2 pairs of 16-bit filters. CAN2.0B (29-bit extended identifier) can be configured with 1-pair 32-bit filters.

When receiving a message, the receiver node will determine whether the software needs the message according to the value of the identifier. If the filter passes, it will be stored in the corresponding mailbox.

- When the mailbox receives data and the corresponding flag (M8484, M8489) is OFF, the received data is stored in the 10 consecutive addresses pointed to by the starting address of the receiving mailbox data (the first two addresses store the identifier, the last eight are the address stores data, which is similar to the data transmission structure) and set the corresponding flag bit.
- The receiving mailbox has a three-level cache structure. When the mailbox receives the data flag bit is ON, and then receives the data, it is stored in the cache mailbox one by one. Take it out when the received data flag bit turns OFF. When the L3 cache mailbox is full, it will no longer receive new data.
  Therefore, after receiving the data, please clear the corresponding flag bit in time to enable the next reception in time. The mailbox filter consists of a mask code and an identifier.

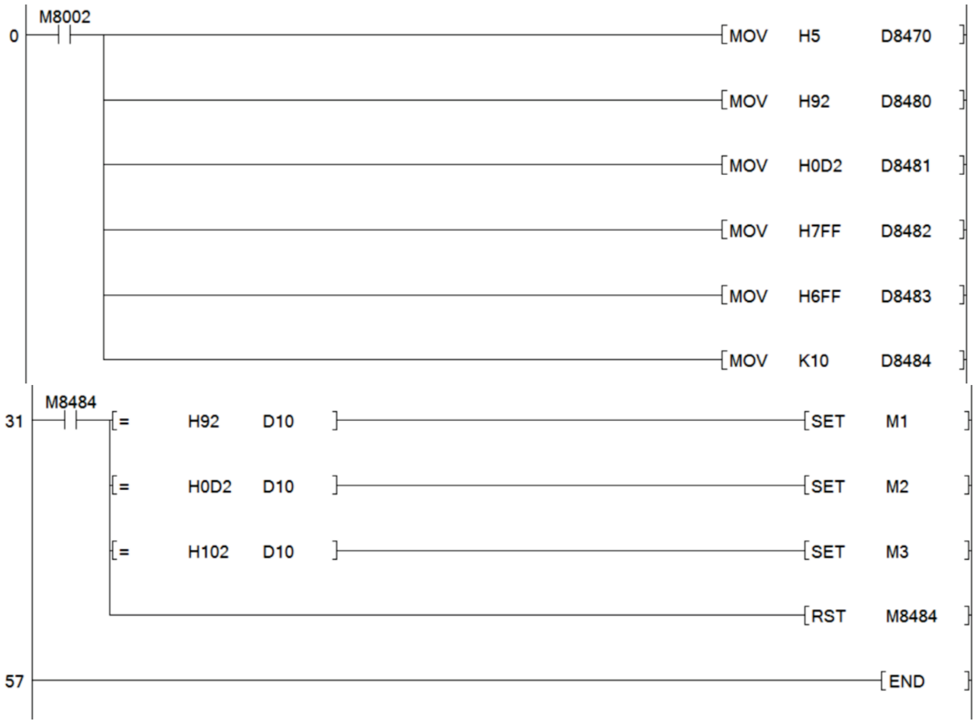
The identifier set in the bit with the mask code 1 and the received data identifier must match, but the bit with 0 is ignored. That is:

- Received data identifier & mask code = set the data identifier & mask code.
- When the identifier of a CAN data frame can pass through two receiving mailboxes at the same time, it will be stored in the receiving mailbox 0.

Each mailbox of CAN2.0A has two sets of mask/identifier code filters. When only one set of filters is used, please note whether the other set will filter unwanted data. Can match the two sets with the same filter or a non-existent identifier filter.

- If there are too many types of data identifiers to be received, the data cannot be completely filtered out one by one. Can narrow the scope through the filter and then programmatically filter out the required data.

Program Example: To Receive CAN2.0A Data Frames Using the Free Port Protocol, Need to Receive Data with Identifiers 0x92, 0xD2, 0x1D2 to Mailbox 0.



```
 0  M8002  [MOV  H5    D8470]
           [MOV  H92   D8480]
           [MOV  H0D2  D8481]
           [MOV  H7FF  D8482]
           [MOV  H6FF  D8483]
           [MOV  K10   D8484]
31  M8484  [=  H92   D10]  [SET  M1]
           [=  H0D2  D10]  [SET  M2]
           [=  H102  D10]  [SET  M3]
                           [RST  M8484]
57                         [END]
```

Execution Steps:	
1	Set the communication parameter D8470 to 0x0005 through M8002 (that is, the free port protocol is used, the baud rate is 125k, and CAN2.0A is used).
2	Set the mask and identifier code. The frame with the mask code 1 of H7FF, that is, exactly matching the identifier 1 (H92), will be received. The mask code 2 with H6FF will need to match the identifier 2 (HD2) except the 9th bit. Frames will be received, namely HD2 and H1D2.
3	When the mailbox 0 receives the data, M8484 is ON. By comparing D10, the received data identifier number is set and the corresponding flag is set. The user can take out the frame data or participate in the calculation according to the actual needs.
4	After the receiving data processing is completed, reset M8484 to run the next reception.

Chapter 6 SFC Program/Step Ladder Diagram

6.1 SFC Program

6.1.1 Outline

In HC10 intelligent controller, can use SFC (Sequential Function Chart) to achieve sequence control.

The use of SFC programs can facilitate the understanding of the role of each process based on mechanical action and the entire control flow.

In addition, the SFC program and step ladder instructions are programmed according to established rules, so they can be converted to each other. Therefore, the substance is completely the same and can also be used as a familiar relay ladder diagram.

6.1.2 Function and Action Description

In the SFC program, the state S is regarded as a control process in which the order of input conditions and output control is programmed.

As the previous process becomes inactive when the process advances, the machine can be controlled in a simple sequence of each process.

In the SFC program, the state is used to indicate each program of the mechanical operation.

- When the status is ON, the corresponding ladder diagram connect to SFC operates.
- When the status is OFF, the corresponding ladder diagram connect to SFC does not operate.

  After one operation cycle, the instruction OFF is not executed (jump state).

When the conditions (transition conditions) set between each state are satisfied, the next state turns ON and the state that was previously ON turns OFF (transition action).

During the state transition, only one moment (1 operation cycle), the two states will be turned on at the same time.

- You cannot reuse the same state number.
- Please use SET S or OUT S (same as SET S in STL instruction) to switch between SFC and STL states.

6.1.3 Use and Effect of Initial State

Use of Initial State

The state occupying the starting position of the SFC program is called the initial state, and the state numbers of S0 ~ S9 can be used.

The initial state is also driven by other states, but it needs to be driven by other means before the start of the operation. For example, it is driven by using the special auxiliary relay M8002 (the first operation cycle of the intelligent controller). General states other than the initial state must be driven by the “Others“ state.

Effect of Initial State

1. As a recognition soft component required for reverse conversion

- When inverting from the instruction list to the SFC program, it is necessary to identify the starting position of the flow. Therefore, use S0 ~ S9 as the initial state. Inverse conversion cannot be performed when using other numbers.

2. Prevent double start

Power Failure Hold State

The state for power failure retention is to use off-chip flash to back up its operating state.

You can use these states when want to continue the operation from the previous state when the power is turned on again during a mechanical operation.

6.1.4 Effect of RET Instruction

In the SFC program, the RET instruction is used at the end of the SFC program. However, when the SFC program is input, the RET instruction does not need to be input (it is automatically written).

In the intelligent controller, multiple SFC blocks can be made from step 0 to the END instruction. When the ladder block and the SFC block are mixed together, write the RET instruction at the end of each SFC program.

Special Auxiliary Relay

In order to be able to make SFC programs more effectively, several special auxiliary relays are needed. The main contents are shown in the table below.

Soft Component Number	Name	Function and Use
M8000	RUN monitoring	Relay that is always ON during the operation of the intelligent controller. Can be used as input conditions for programs that need to be driven all the time, and can also be used to display intelligent controllers.
M8002	Initial pulse	This relay is ON only when the intelligent controller switches from STOP to RUN (1 operation cycle). Used for initial setting and initial state setting of the program.
M8040	No transfer	After this relay is driven, transitions between all states are prohibited. In addition, in the state where the transition is prohibited, the program in the state is still operating, so the output coils, etc. will not be automatically disconnected.
M8046[1)]	STL action	As long as one of the states S0 ~ S899, S1000 ~ S4095 is ON, M8046 will automatically turn ON. It is used to avoid starting with other processes at the same time, or it can be used as an action flag for a process, or to avoid multiple processes in STL start at the same time.
M8047[1)]	STL monitoring is effective	After this relay is driven, the latest number of the status relays that are operating (ON) among status relays S0 to S899, S100 to S4095 are stored in D8040, and the status number of the next operation (ON) is saved to D8041. And so on, save until D8047 (Max. 8 points).
1): Processed when the END instruction is executed.		

6.2 Step Ladder Diagram

6.2.1 Outline

A program using step ladder diagram instructions, based on the operation of the machine, assigns state S to each process as a circuit connected to the state contact (STL contact), and programs the order of input conditions and output control.

- The thinking methods, types of states, and actions of writing a program are the same as those of an SFC program. Since it can be represented by a ladder diagram, its substance is completely the same as an SFC program, and it can be used as a familiar relay ladder diagram.
- In addition, step ladder diagrams can also be programmed in the form of instruction lists.

This chapter describes the writing and precautions of step ladder diagram, and the input sequence in the form of instruction list.

6.2.2 Fuction Description

In the step ladder diagram, treat the state S as a control process, and write a sequence program for input conditions and output control.

As the previous process becomes inactive when the process advances, the machine can be controlled in a simple sequence of each process.

Operations of Step Ladder Diagram Instructions

In the step ladder diagram, states are used to represent the various steps of the mechanical operation.

This way of thinking can be adopted: Thinking that the state likes relay, which is composed of a drive coil and a contact (STL contact).

Use SET and OUT instructions in the drive coil and STL instructions in the contacts.

- After the status is ON, the ladder diagram (internal ladder diagram) connected to it will be operated by STL electric shock.

  When the status is OFF, the internal ladder diagram connected to it is not operated by the STL contact.

  After one operation cycle, the instruction OFF is not executed (jump state).
- When the conditions (transition conditions) set in the transition of each state are satisfied, the next state is turned on, and the state that was previously ON is turned off (transition operation).
  During the state transition, only one moment (1 operation cycle), the two states will be turned on at the same time.

  The state before the transition is turned OFF (reset) in the next operation cycle after the transition.
  However, when using the pre-transition state S by the contact instruction, the contact image is turned off after the transition condition is satisfied.
- You cannot reuse the same state number.

Sequence Instruction List That Can be Used between STL Instruction and RET Instruction

		Instructions		
State		LD/LDI/LDP/LD, AND/ANI/ANDP/ANDF, OR/ORI/ORP/ORF, INV, MEP/ MEF, OUT, SET/RST, PLS/PLF	ANB/ORB/MPS/ MRD/MPP	MC/MCR
Initial state/general state		Can be used	Can be used	Can be used
Branch and confluence state	Can be used	Can be used	Can be used	Can not be used
	Can be used	Can be used	Can be used	Can not be used

It is not forbidden to use the jump instruction in the state, but it is recommended to avoid using it because it will cause complicated actions. Even if it drives to process the ladder diagram, the MPS instruction cannot be used directly after the STL instruction.

For a series of step ladder diagrams, program from the initial state in the order of the states to be transitioned. In addition, be sure to program the RET instruction at the end of the step ladder diagram.

When multiple relay ladder diagrams and step ladder diagrams are mixed together, enter the RET instruction at the end of the step ladder diagram.

The intelligent controller starts the processing of the step ladder diagram according to the STL instruction, and returns from the step ladder diagram to the relay ladder diagram according to the RET instruction. However, when programming immediately after the step ladder diagram of different processes (there is no relay ladder diagram between multiple processes of step ladder diagrams), it is allowed to omit the RET instruction between the processes, and only write the RET instruction at the end of the last process.

Special Auxiliary Relay

In order to be able to write step ladder diagrams more effectively, several special auxiliary relays are needed. The main contents are shown in the table below.

Soft Component Number	Name	Function and Use
M8000	RUN monitoring	Relay that is always ON during the operation of the intelligent controller. It can be used as an input condition for a program that needs to be constantly driven and as a display of the operating state of the intelligent controller.
M8002	Initial pulse	Relay that is ON only when the intelligent controller switches from STOP to RUN (1 operation cycle). Used for initial setting and initial state setting of the program.
M8046	STL action	Even if only one state of S0 ~ S899, S1000 ~ S4095 is ON, M8046 will automatically turn ON. It is used to avoid starting at the same time as other processes, or as an action flag for a process.
M8047	STL monitoring is effective	After this relay is driven, the latest number of the active (ON) state in states S0 to S899, S1000 to S4095 is saved to D8040, and the state number of the next action (ON) is saved to D8041. The operation state (up to 8 points) is sequentially saved until D8047.

Chapter 7 Interrupt Function and Pulse Capture Function

In this chapter, it mainly describes the built-in interrupt function and pulse capture function in the HC10 intelligent controller.

7.1 Outline

It mainly describes the function of executing the interrupt program (interrupt subroutine) immediately without being affected by the operation cycle of the sequence program (main program), using the following interrupt functions as trigger signals.

In general sequence program processing, the delay caused by the operation cycle and the time deviation have an impact on the mechanical action, and this situation can be improved.

Input Interrupt Function (Interruption of External Signal Input (X))

Use the input signals X000 ~ X005 to interrupt the general sequence program, and execute the interrupt subroutine first.

In addition, the execution timing of the input interrupt can be specified by either the pointer number or the rising or falling edge of the signal.

Timer Interrupt Function (Timer Interrupt that Operates at a Fixed Period)

Interrupt the general sequence program at a fixed cycle interval of 10 ~ 99ms, and execute the interrupt subroutine first.

High-speed Counter Interrupt Function (Interrupt Function during Up Counting)

When the current value of the high-speed counter reaches the specified value, the general sequence program is interrupted and the interrupt subroutine is given priority.

Pulse Capture Function

By changing the input signals X000 ~ X005 from OFF to ON, the special auxiliary relays M8170 ~ M8175 are set to interrupt processing. By using this M8170 ~ M8175 in a general sequence program, it can be easily obtained in general input processing unable to get the ON width signal.

However, if processing such as ON/OFF is performed several times in one operation cycle, use the input interrupt function.

7.2 General Matters

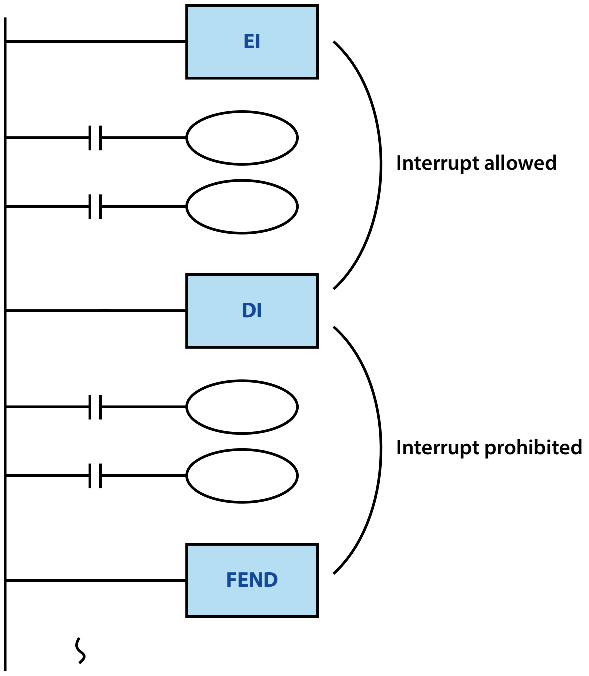
Describe how to disable the interrupt function and pulse capture function.

1. Limitation of the Interrupt Range of the Program [Interrupt Function, Pulse Capture Function]

By programming the FN 05 (DI) instruction, the area where interrupts are disabled can be set.

Interrupt events that occur between DI ~ EI instructions (interrupt prohibited area) will wait until interrupt prohibition ends (EI instruction) to respond.

The program example is shown below.



Note	
a	Special auxiliary relays (M8050 ~ M8059) for disabling interrupts do not include interrupt inputs that are already turned ON. This special auxiliary relay has no effect on the pulse capture function.
b	Interrupt 100us refresh once. Loss of the same interrupt occurs multiple times within 100us, please be careful not to generate interrupts too densely. Interrupts that occur within 100us will be executed according to priority (the lower the number, the higher the priority), and they will not respond in time.
c	Interrupts will not be nested, but interrupts generated during the execution of the interrupt will be recorded and respond at the end. However, the number of interrupt records is limited (5 interrupt refresh status). If the interrupt is too dense, the interrupt may be lost.
d	The watchdog still keeps counting when interrupts are executed, so be careful to avoid watchdog failures caused by long interruptions.
e	Use a timer in the interrupt. Using ordinary timers in interrupts may get unexpected results. For timers in interrupt subroutines, please use timers T192 ~ T199 for subroutines.
f	The X terminal can only perform one special function at the same time. The terminal interrupt function, high-speed counting function, and positioning function cannot be used simultaneously.
g	Interrupt execution is equivalent to only executing one cycle. Pay attention to the difference between the terminal and instruction status and the main program continuous execution.

2. Disable the Interrupt of the Interrupt Pointer (Each Interrupt Subroutine) [Interrupt Function]

Interrupts that are disabled when the interrupt disable flag (M8050 to M8059) are ON. After that, even if the interrupt prohibition flag is turned off, the interrupt signal generated during the interrupt prohibition period will not be executed again.

Input Interrupt	The input interrupts of X000 ~ X005 correspond to M8050 ~ M8055, which are disabled when ON.
Timer Interrupt	I6□□ ~ I8□□ timer interrupts correspond to M8056 ~ M8058, and are disabled when ON.
High-speed Counter Interrupt	All counter interrupts from I010 to I060 are disabled when M8059 is ON.

7.3 Input Interrupt

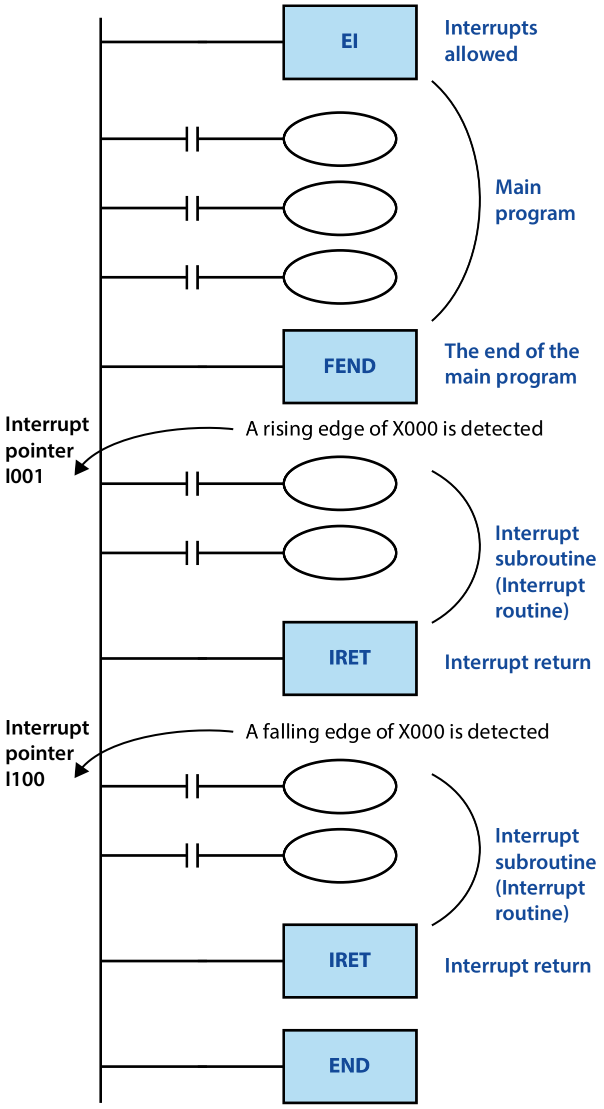
Outline

Use the input signals X000 ~ X005 to execute the interrupt subroutine.

Usage

Since external input signals can be processed without being affected by the operation cycle of the intelligent controller, it is suitable for performing high-speed control and obtaining short-time pulses.

Basic Procedures (Programming Tips)


EI
Interrupts allowed
Main program
FEND
The end of the main program
Interrupt pointer I001
A rising edge of X000 is detected
Interrupt subroutine (Interrupt routine)
IRET
Interrupt return
Interrupt pointer I100
A falling edge of X000 is detected
Interrupt subroutine (Interrupt routine)
IRET
Interrupt return
END

The main program receives the interrupt input reception after the EI instruction is valid. In addition, it is not necessary to write a DI (disable interrupt) instruction when there is no need to disable the input interrupt area.

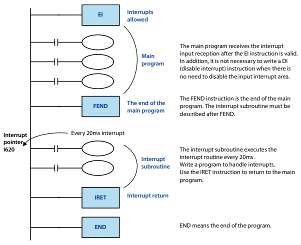
The FEND instruction is the end of the main program. The interrupt subroutine must be described after FEND.

When X000 is turned on, its rising edge is detected, and the interrupt subroutine returns to the main routine through the IRET instrution.

When X000 is disconnected, its falling edge is detected, and the interrupt subroutine returns to the main routine through the IRET instrution.

END means the end of the program.

Number and Operation of Interrupt Pointer (6 o'clock)

Interrupt Pointer	Input Number	Pointer Number		Disable Interrupt Instructions
		Rising Edge Interrupt	Falling Edge Interrupt	
 EI Interrupts allowed Main program FEND The end of the main program Interrupt pointer I620 Every 20ms interrupt Interrupt subroutine IRET Interrupt return END The main program receives the interrupt input reception after the EI instruction is valid. In addition, it is not necessary to write a DI (disable interrupt) instruction when there is no need to disable the input interrupt area. The FEND instruction is the end of the main program. The interrupt subroutine must be described after FEND. The interrupt subroutine executes the interrupt routine every 20ms. Write a program to handle interrupts. Use the IRET instruction to return to the main program. END means the end of the program.	X000	I001	I000	M8050 [1)]
	X001	I101	I100	M8051 [1)]
	X002	I201	I200	M8052 [1)]
	X003	I301	I300	M8053 [1)]
	X004	I401	I400	M8054 [1)]
	X005	I501	I500	M8055 [1)]
	1): Cleared from RUN to STOP.			

Individual Disable Method of Interrupt Input

When M8050 ~ M8055 are turned on in the program, their corresponding interrupts are disabled. Refer to the table above for the corresponding content.

Note

Note		Description
1	Function multiplexing of input relays	The number of the input relay used as the interrupt pointer should not be repeated with application instructions such as "high-speed counter" and "pulse capture function" that use the same input range.
2	Automatic adjustment of the input filter	When the input interrupt pointer I□0□ is specified, the input filter of the input relay is automatically changed to high-speed reading. Therefore, there is no need to use the special data register D8020 (adjustment of input filter) to change the adjustment of the filter.
3	Pulse width of input interrupt	In order to be able to perform input interruption through external signals, the input width needs to be above the hardware filtering time, please refer to the hardware planning description section for the corresponding time.
4	Reuse of pointer numbers	Rising and falling interrupts on the same input cannot be programmed simultaneously.
5	Rising edge falling edge	The edge here is a logic level. For example, the rising edge refers to the state when the X terminal is turned on, and the falling edge refers to the state when the X terminal is turned off.

7.4 Timer Interrupt

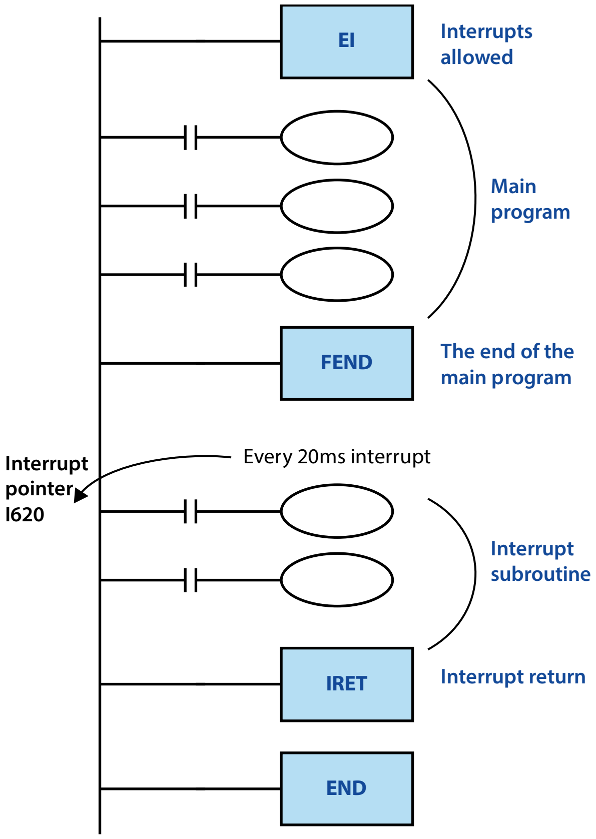
Outline

Not affected by the operation cycle of the intelligent controller, the interrupt program is executed every 1 ~ 99ms.

Usage

It is suitable for the case where the main program has a long operation cycle, high-speed processing for a specific program, or a specific program that needs to be executed at a certain interval.

Basic Procedures (Programming Tips)



The main program receives the interrupt input reception after the EI instruction is valid. In addition, it is not necessary to write a DI (disable interrupt) instruction when there is no need to disable the input interrupt area.

The FEND instruction is the end of the main program. The interrupt subroutine must be described after FEND.

The interrupt subroutine executes the interrupt routine every 20ms. Write a program to handle interrupts. Use the IRET instruction to return to the main program.

END means the end of the program.

Number and Operation of Timer Interrupt Pointer (3 o'clock)

The interrupt subroutine is executed every specified interrupt cycle time (1 ~ 99ms).

It is used for control that requires cyclic interrupt processing outside the operation cycle of the intelligent controller.

Input Number	Interrupt Cycle	Terminal Disable Flag
I6□□	In the pointer name, enter an integer from 1 to 99. Example: I610 = timer interrupt every 10ms	M8056[1)]
I7□□		M8057[1)]
I8□□		M8058[1)]
1): Cleared from RUN to STOP.		

Note

Note	
1	The pointer numbers (I6, I7, I8) cannot be reused. When M8056 ~ M8058 are turned on in the program, their corresponding timer interrupts are disabled.

7.5 Counter Interrupt

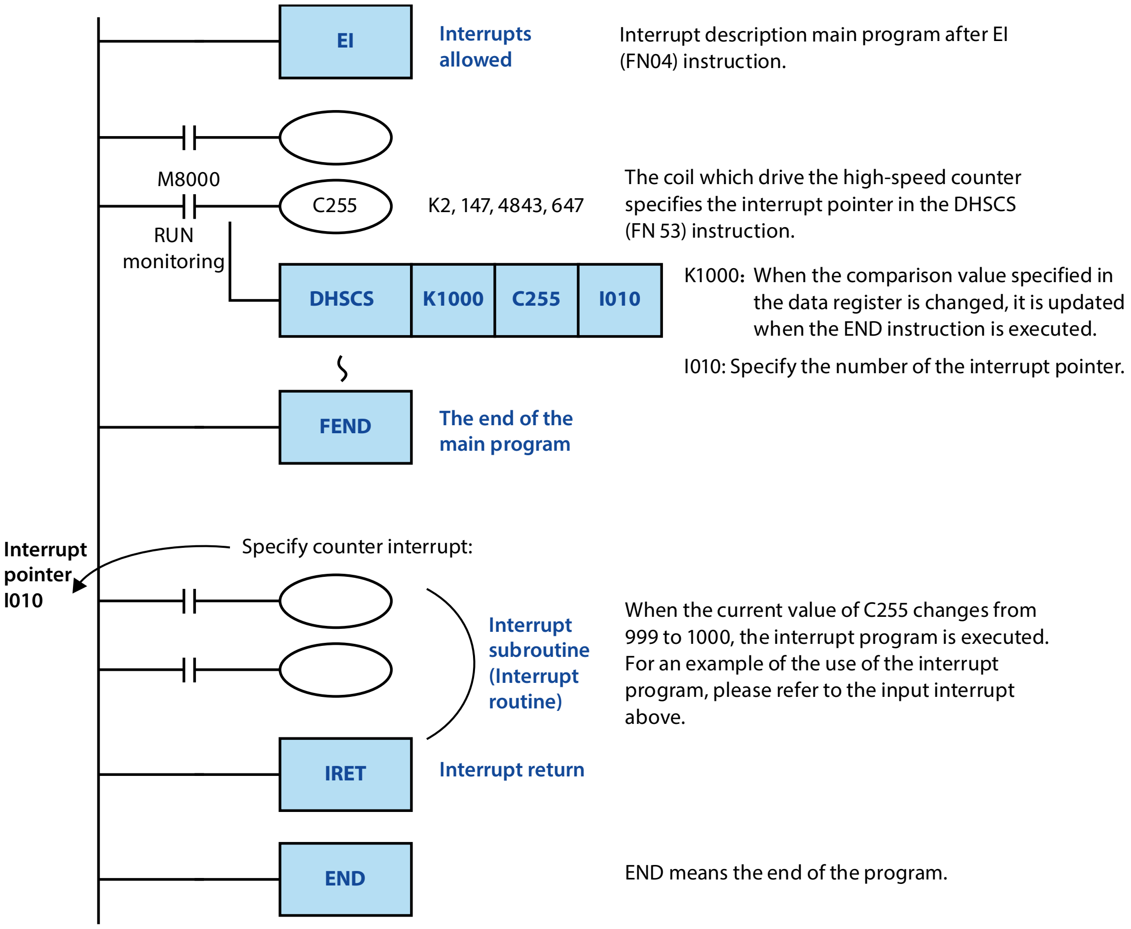
Outline

Execute high-speed count interrupt.

Usage

Used with the compare set instruction of DHSCS (FN 53) to execute the interrupt routine when the current value of the high-speed counter reaches the specified value.

Basic Procedures (Programming Tips)


EI
Interrupts allowed
Interrupt description main program after EI (FN04) instruction.
M8000
RUN monitoring
C255
K2, 147, 4843, 647
The coil which drive the high-speed counter specifies the interrupt pointer in the DHSCS (FN 53) instruction.
DHSCS
K1000
C255
I010
K1000: When the comparison value specified in the data register is changed, it is updated when the END instruction is executed.
I010: Specify the number of the interrupt pointer.
FEND
The end of the main program
Interrupt pointer I010
Specify counter interrupt:
Interrupt subroutine (Interrupt routine)
When the current value of C255 changes from 999 to 1000, the interrupt program is executed. For an example of the use of the interrupt program, please refer to the input interrupt above.
IRET
Interrupt return
END
END means the end of the program.

Number and Operation of Timer Interrupt Pointer (6 o'clock)

Pointer Number (6 o'clock)	Interrupt Disable Flag
I010, I020, I030, I040, I050, I060	M8059 [1]
1): Cleared from RUN to STOP.	

ON/OFF of Interrupt Output (Y, M) Using High-speed Counter

When only the ON/OFF output relay (Y) and auxiliary relay (M) are controlled based on the current value of the high-speed counter, the DHSCS (FN 53), DHSCR (FN 54), DHSZ (FN 55) instructions can be easily programmed.

Note

Note		Description
1	Duplicated pointer numbers	You cannot reuse pointer numbers.
2	Prohibition of interruption	After the special auxiliary relay M8059 is turned on in the program, all counter interrupts are disabled.

7.6 Pulse Capture Function [M8170 ~ M8175]

After executing the FN 04 (EI) instruction, when the X000 ~ X007 of input relays change from OFF to ON, the special auxiliary relays M8170 ~ M8177 are set by interrupt processing.

Input Number and Assignment of Special Auxiliary Relays

Pulse Capture Input	Pulse Capture Relay
X000	M8170[1)]
X001	M8171[1)]
X002	M8172[1)]
X003	M81731)
X004	M8174[1)]
X005	M8175[1)]
1): Cleared from RUN to STOP.	

Note

Note	
1	To read the input again, the set soft component needs to be reset by the program. Therefore, the set soft component cannot read the new input until it is reset.
2	To read continuous short-time pulses (input signals), use the external input interrupt function or the high-speed counter function.
3	No need to adjust the filter.
4	It has nothing to do with the operation of auxiliary relays M8050 ~ M8055.

Chapter 8 Analog Usage Introduction

HC10 comes with 4 analog inputs, including 2 analog inputs and 2 analog outputs.

Scan every 1ms for analog (input sampling or output refresh). It is independent of program execution. Through special address input and output, STOP state analog output is 0V.

The analog range and offset can be set, the analog range can be flexibly adjusted according to the application, and the deviation of each channel can be calibrated by fine adjustment.

The special addresses are as follows:

Special Address for Analog Input and Output Related Software					
Category	**Terminal**	**Address**	**Voltage and Current Selection (ON/OFF)**	**Range**	**Offset**
Analog input	AI1	D8256	M8256	D8220	D8221
	AI2	D8257	M8257	D8222	D8223
Analog output	AO1	D8258	M8258	D8224	D8225
	AO2	D8259	M8259	D8226	D8227

Note:

1 The range of the analog output can be modified by the special address (1 ~ 32767) and offset (-32768 ~ +32767) special address (D8256 ~ D8257). For example, when AI1 is a voltage type, set the range D8220 to 1000 and D8221 to 2000, then after the change, AI1 input 0 ~ 10V corresponds to D8256 output 2000 ~ 3000.

2 The default range is 32000, and the offset is 0, that is, the default range of analog input value and analog output value is 0 ~ 32000.

3 When the set value of the analog output is lower than the lower limit, press the limit output; When it is higher than the upper limit, press the upper limit output.

Chapter 9 Expansion Module Usage Introduction

HC10 can expand up to 8 expansion modules, which are connected by a cable. HC10 automatically scans the type and number of expansion modules after power-on. It cannot be changed after power-on. If need to add or change the module type, need to power on again.

When using expansion module, make sure that the cable is connected before powering on.

There are two ways to update the data of the expansion module:

1. Auto Update

X, Y terminal status and analog input are automatically updated
The X and Y terminals of the module are arranged in sequence after the main module, and the XY terminals of the module can be controlled by reading and writing the corresponding XY buffer address.

The analog input will also be cached in the PLC through automatic update, and the cached analog input value can be directly read through the RD3A to ensure the real-time refresh of the analog input of the module.

2. Active Access

Except for the above automatically updated data, the rest of the data is accessed through the FROM/TO instruction to read and write the module's buffer area. For specific usage, please refer to the FROM/TO instruction. For the module's buffer area definition, please refer to the module manual.

D8260 ~ D8279, M8260 ~ M8279 indicate special addresses for module communication status, definition:

D Address	Definition	M Address	Definition
D8260	Number of expansion modules		
D8262	Expansion module command communication status (for FROM/TO/RD3A):		
	0x01: Communication succeeded	0x11: Module does not exist	0x12: Address (channel) overrun
	0x13: Non-analog input module	0x21: Return frame error	0x22: Receive timeout
	0x23: Read data loss	0x25: Read data loss	0x26: Address is not writable
D8265	Module 1 model	M8265	Module 1 communication flag, 0: Disconnection 1: Communication in progress
D8267	Module 2 model	M8267	Module 2 communication flag, 0: Disconnection 1: Communication in progress
D8269	Module 3 model	M8269	Module 3 communication flag, 0: Disconnection 1: Communication in progress
D8271	Module 4 model	M8271	Module 4 communication flag, 0: Disconnection 1: Communication in progress
D8273	Module 5 model	M8273	Module 5 communication flag, 0: Disconnection 1: Communication in progress
D8275	Module 6 model	M8275	Module 6 communication flag, 0: Disconnection 1: Communication in progress
D8277	Module 7 model	M8277	Module 7 communication flag, 0: Disconnection 1: Communication in progress
D8279	Module 8 model	M8279	Module 8 communication flag, 0: Disconnection 1: Communication in progress

Chapter 10 Special Soft Components (M8000 ~, D8000 ~)

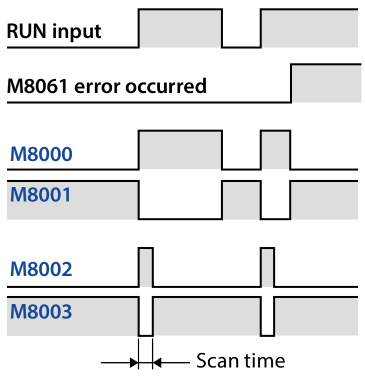
10.1 Special Soft Components (M8000 ~, D8000 ~)

The types and functions of special auxiliary relays (referred to as special M in the table) and special data registers (referred to as special D in the table) are shown below.

In addition, depending on the series of the intelligent controller, even if the same soft component number is used, the function content may be different, so please note.

Undefined and undocumented special auxiliary relays and special data registers are areas occupied by the CPU. Therefore, do not use them in sequence programs.

10.1.1 Special Auxiliary Relays (M8000 ~ M8511)

Number and Soft Components	Action and Function	Corresponding Special Soft Components
Intelligent Controller State		
M8000 RUN monitoring (a contact)	 RUN input M8061 error occurred M8000 M8001 M8002 M8003 Scan time	-
M8001 RUN monitoring (b contact)		-
M8002 Initial pulse (a contact)		-
M8003 Initial pulse (b contact)		-
M8004 Error occurred	Connected when any of M8060, M8061, M8064, M8065, M8066, M8067 is ON	D8004
M8005 Low battery voltage (power-on detection only)	Connected when the battery voltage is abnormally low (only power-on detection, turn on when the voltage is detected below 2.8V, and the corresponding LED is on)	D8005
Clock		
M8011 10ms clock	ON/OFF in 10ms per cycle (ON: 5ms, OFF: 5ms)	–
M8012 100ms clock	ON/OFF in 100ms per cycle (ON: 50ms, OFF: 50ms)	–
M8013 1s clock	ON/OFF in 1s per cycle (ON: 500ms, OFF: 500ms)	–
M8014 1min clock	ON/OFF in 1min per cycle (ON: 30s, OFF: 30s)	–
M8015*1	Calibration time For real-time clock	D8013 ~ D8019
M8016	Show time stop For real-time clock	D8013 ~ D8019
M8018*1	Installation detected (always ON) For real-time clock	D8013 ~ D8019
M8019	Real-time clock (RTC) errors For real-time clock	–
**1. Only some models and versions are supported.*		

Number and Soft Components	Action and Function	Corresponding Special Soft Components
Flag		
M8020 Zero	Turn on when the result of addition and subtraction is 0	–
M8021 Borrow	Turns on when the subtraction result exceeds the Max. negative value	–
M8022 Carry	Turns on when the carry result of the addition operation occurs, or when the shift result overflows	–
M8024	Specify BMOV direction (FN 15)	–
M8026	RAMP mode (FN 67)	–
M8029 Instruction execution completed	Connected when the operation of PLSY etc. is completed	–
Intelligent Controller Mode		
M8031 Clear all non-retentive memory	After driving this special M, the ON/OFF image area of Y/M/S/T/C and the current value of T/C/D are cleared	–
M8032 Keep all memory cleared		
M8033 Memory keeps stopping	From RUN to STOP, the contents of the image storage area and data storage area are maintained as they are	–
M8034 Suppress all output	All external output contacts of the intelligent controller are open	–
M8035 Forced RUN mode		–
M8036 Forced RUN instruction		–
M8037 Forced STOP instruction		–
M8039 Constant scan mode	After M8039 is turned on, wait until the scan time specified in D8039 until the intelligent controller executes such a loop operation	D8039
Step Ladder Diagram・Signal Alarm		
M8046 STL state action	When M8047 is on, any of S0 ~ S899, S1000 ~ S4095 is ON	M8047
M8047 STL monitoring is effective	After driving this special M, D8040 ~ D8047 are effective	D8040 ~ D8047
M8048 Signal alarm action	When M8049 is on, any of S900 ~ S999 is ON	–
M8049 Signal alarm is effective	When this special M is driven, the action of D8049 is effective	D8049 M8048

Number and Soft Components	Action and Function	Corresponding Special Soft Components
Disable Interrupt		
M8050 (input interrupt) I00□ disabled *1	When the speacial M of disable input interrupt or timer interrupt is connected: • Even if an input interrupt or a timer interrupt occurs, the interrupt routine is not processed because the reception of the corresponding interrupt is disabled. • For example, when M8050 is turned on, the reception of interrupt I00□ is disabled, so the interrupt program will not be processed even if it is within the range of the interrupt-enabled program. When the speacial M of disable input interrupt or timer interrupt is disconnected: • Receive interrupt when input interrupt or timer interrupt occurs. • If interrupts are enabled using the EI (FN 04) instruction, the interrupt routine will be executed immediately. However, if the interrupt is disabled by the DI (FN 05) instruction, the interrupt will not be responded to.	
M8051 (input interrupt) I10□ disabled *1		
M8052 (input interrupt) I20□ disabled *1		
M8053 (input interrupt) I30□ disabled *1		
M8054 (input interrupt) I40□ disabled *1		
M8055 (input interrupt) I50□ disabled *1		
M8056 (timer interrupt) I6□□ disabled *1		
M8057 (timer interrupt) I7□□ disabled *1		
M8058 (timer interrupt) I8□□ disabled *1		
M8059 counter interrupt disabled *1	Using I010 ~ I060 disable interrupt.	
**1. Cleared from RUN to STOP.*		
Error Detection		
M8061	Intelligent controller hardware error	D8061
M8063	MOD1 communication error 1	D8063
M8064	Parameter error	D8064
M8065	Grammatical errors	D8065, D8069, D8314, D8315
M8066	Loop error	D8066, D8069, D8314, D8315
M8067	Arithmetic error	D8067, D8069, D8314, D8315
M8068	Operation error latch	D8068, D8312, D8313
M8069	I/O bus detection	-
High-speed Ring Counter		
M8099	High-speed ring counter (0.1ms unit, 16 bit) operation	D8099
Memory Information		
M8101 ~ M8108	Can not be used	-
MOD1 Communication Flag		
M8123	MOD1 communication completion flag	-
Expansion Fuction		
M8161	8-bit processing mode	-
M8165	SORT2 (FN 149) instruction in descending order	-
M8167	HKY (FN 71) function for processing HEX data	-
M8168	SMOV (FN 13) instruction function for processing HEX data	-
Pulse Capture Function		
M8170 *1	Input X000 pulse capture	-
M8171 *1	Input X001 pulse capture	-
M8172 *1	Input X002 pulse capture	-
M8173 *1	Input X003 pulse capture	-
M8174 *1	Input X004 pulse capture	-
M8175 *1	Input X005 pulse capture	-
**1. Cleared from STOP to RUN, EI is required.*		

Number and Soft Components	Action and Function		Corresponding Special Soft Components
Counting Direction of the Counter Up or Down			
M8196*1	C251	1 times/4 times switch of C251	–
M8197*1	C252	1 times/4 times switch of C252	–
M8198	C251*2	C251*2 1 times/4 times switch	–
M8199	C253*3	C253*3 1 times/4 times switch	–
M8200	C200	The counting mode of C□□□ is set by the corresponding M8□□□. • When M8□□□ is ON, C□□□ counts down • When M8□□□ is OFF, C□□□ counts up	–
M8201	C201		–
M8202	C202		–
M8203	C203		–
M8204	C204		–
M8205	C205		–
M8206	C206		–
M8207	C207		–
M8208	C208		–
M8209	C209		–
M8210	C210		–
M8211	C211		–
M8212	C212		–
M8213	C213		–
M8214	C214		–
M8215	C215		–
M8216	C216		–
M8217	C217		–
M8218	C218		–
M8219	C219		–
M8220	C220		–
M8221	C221		–
M8222	C222		–
M8223	C223		–
M8224	C224		–
M8225	C225		–
M8226	C226		–
M8227	C227		–
M8228	C228		–
M8229	C229		–
M8230	C230		–
M8231	C231		–
M8232	C232		–
M8233	C233		–
M8234	C234		–

**1. Only supported by HC10-M0808R-C3-AB model.*

**2. For HC10-M0808R-C3-AB model, M8198 corresponds to C253.*

**3. For HC10-M0808R-C3-AB model, M8199 corresponds to C254.*

Number and Soft Components	Action and Function		Corresponding Special Soft Components
Counting Direction of the High-speed Counter Up or Down			
M8235	C235	C235 ~ C238 is a single-phase single-input counter; • When M8□□□ is ON, C□□□ counts down • When M8□□□ is OFF, C□□□ counts up	–
M8236	C236		–
M8237	C237		–
M8238	C238		–

M8246	C246	C246 ~ C248 is a single-phase dual-input counter;
		C251 ~ C254 is a dual-phase dual-input counter;
		• When C□□□ counts down, M8□□□ is ON
		• When C□□□ counts up, M8□□□ is OFF
			–

M8248	C248		–
M8251	C251		–
M8252	C252		–
M8253	C253		–
M8254	C254		–
Analog Voltage and Current Selection			
M8256	AI1 voltage and current selection (power-off save)		D8256
M8257	AI2 voltage and current selection (power-off save)		D8257
M8258	AO1 voltage and current selection (power-off save)		D8258
M8259	AO2 voltage and current selection (power-off save)		D8259
Expansion Module			
M8265	Module 1 communication flag 0: Disconnection 1: Communication in progress		
M8267	Module 2 communication flag 0: Disconnection 1: Communication in progress		
M8269	Module 3 communication flag 0: Disconnection 1: Communication in progress		
M8271	Module 4 communication flag 0: Disconnection 1: Communication in progress		
M8273	Module 5 communication flag 0: Disconnection 1: Communication in progress		
M8275	Module 6 communication flag, 0: Disconnection 1: Communication in progress		
M8277	Module 7 communication flag 0: Disconnection 1: Communication in progress		
M8279	Module 8 communication flag 0: Disconnection 1: Communication in progress		
Flag			
M8304 zero	ON when the result of the multiplication and division operation is 0		–
M8306 carry	ON when division result overflows		–
M8329	Instruction execution abnormal end flag (for high-speed pulse output commands)		–
Timing Clock・Positioning			
M8330	DUTY (FN 186) instruction timing clock output 1		D8330
M8331	DUTY (FN 186) instruction timing clock output 2		D8331
M8332	DUTY (FN 186) instruction timing clock output 3		D8332
M8333	DUTY (FN 186) instruction timing clock output 4		D8333
M8334	DUTY (FN 186) instruction timing clock output 5		D8334

Number and Soft Components	Action and Function	Corresponding Special Soft Components
Pulse Output Positioning		
M8338	PLSV (FN 157) command acceleration and deceleration action	–
M8340	[Y000] monitoring during pulse output (ON: BUSY/OFF: READY)	–
M8341	[Y000] clear signal output function is valid	–
M8342	[Y000] origin return direction designation	–
M8343	[Y000] forward limit	–
M8344	[Y000] reverse limit	–
M8345	[Y000] near-point signal logic inversion	–
M8346	[Y000] origin signal logic inversion	–
M8347	[Y000] inte positioning command drivingrrupt signal logic inversion	–
M8348	[Y000] positioning command driving	–
M8349	[Y000] command to stop pulse output	–
M8350	[Y001] monitoring during pulse output (ON: BUSY/OFF: READY)	–
M8351	[Y001] clear signal output function is valid	–
M8352	[Y001] origin return direction designation	–
M8353	[Y001] forward limit	–
M8354	[Y001] reverse limit	–
M8355	[Y001] near-point signal logic inversion	–
M8356	[Y001] origin signal logic inversion	–
M8357	[Y001] inte positioning command drivingrrupt signal logic inversion	–
M8358	[Y001] positioning command driving	–
M8359	[Y001] command to stop pulse output	–
M8360	[Y002] monitoring during pulse output (ON: BUSY/OFF: READY)	–
M8361	[Y002] clear signal output function is valid	–
M8362	[Y002] origin return direction designation	–
M8363	[Y002] forward limit	–
M8364	[Y002] reverse limit	–
M8365	[Y002] near-point signal logic inversion	–
M8366	[Y002] origin signal logic inversion	–
M8367	[Y002] inte positioning command drivingrrupt signal logic inversion	–
M8368	[Y002] positioning command driving	–
M8369	[Y002] command to stop pulse output	–
M8370	[Y003] monitoring during pulse output (ON: BUSY/OFF: READY)	–
M8371	[Y003] clear signal output function is valid	–
M8372	[Y003] origin return direction designation	–
M8373	[Y003] forward limit	–
M8374	[Y003] reverse limit	–
M8375	[Y003] near-point signal logic inversion	–
M8376	[Y003] origin signal logic inversion	–
M8377	[Y003] inte positioning command drivingrrupt signal logic inversion	–
M8378	[Y003] positioning command driving	–
M8379	[Y003] command to stop pulse output	–
Ring Counter		
M8398	1ms ring count (32 bit) action	D8398, D8399
MOD2 Communication Flag		
M8403	MOD2 communication completion flag	–
M8438	MOD2 communication error flag	D8438
Pulse Output Positioning User Interrupt Input Instruction		
M8460	[Y000] user interrupt input instruction	–
M8461	[Y001] user interrupt input instruction	–
M8462	[Y002] user interrupt input instruction	–

Number and Soft Components	Action and Function	Corresponding Special Soft Components
M8463	[Y003] user interrupt input instruction	–
M8464	[Y000] clear signal soft element designation function is valid	–
M8465	[Y001] clear signal soft element designation function is valid	–
M8466	[Y002] clear signal soft element designation function is valid	–
M8467	[Y003] clear signal soft element designation function is valid	–
CAN Communication		
M8471	CAN communication completion flag	–
M8475	CAN communication error flag	
M8476	QDF host communication error flag	D8476
M8479	CAN communication error flag	
M8480	CAN free port send data command	
M8481	QDF1 enable flag	
M8483	QDF1 communication success flag	
M8484	QDF2 enable flag	
M8486	QDF2 communication success flag or CAN free port mailbox 0	
M8487	QDF3 enable flag	
M8489	QDF3 communication success flag	
Program Protection Function		
M8511	Program disable read enable	–

10.1.2 Special Data Register (D8000 ~ D8511)

Number and Soft Components	Action and Function	Corresponding Special Soft Components
Intelligent Controller Status		
D8000 Watchdog timer	Max. scan time of one cycle of the program, Max.: 3000ms, unit: ms, initial value: 200ms	-
D8001	System parameter, not available	
D8002	System parameter, not available	
D8003	System parameter, not available	
D8004 Error M number	Error M number Min.	M8004
D8005 Battery voltage	Detection only at power-on (unit: 0.1V)	M8005
D8007 Power supply voltage detection	Detection of intelligent controller power supply voltage (unit: V)	M8007
Clock		
D8010 Scan current value	Cumulative execution time of instructions starting at step 0 (0.1ms unit)	-
D8011 MIN scan time	Min. scan time (0.1ms unit)	-
D8012 MAX scan time	Max. scan time (0.1ms unit)	-
D8013 Second	0 ~ 59 seconds (for real-time clock)	-
D8014 Minute	0 ~ 59 minutes (for real-time clock)	-
D8015 Hour	0 ~ 23 hours (for real-time clock)	-
D8016 Day	1 ~ 31 days (for real-time clock)	-
D8017 Month	January to December (for real-time clock)	-
D8018 Year	2-digit western calendar (0 ~ 99) (for real-time clock)	-
D8019 Week	0 (Sun) ~ 6 (Sat) (for real-time clock)	-
Input Filter		
D8020 Input filter adjustment	Normal input terminal input filter value, initial value: 10ms (power-off save)	–
D8021	User program version number	–
D8022	Can not be used	–
D8023		–
D8024		–
D8025		–
D8026		–
D8027		–
Index Register ZO, VO		
D8028	Z0 (Z) register contents (Z1 ~ Z7 contents are stored in D8182 ~ D8195)	–
D8029	V0 (V) register contents (the contents of V1 ~ V7 are stored in D8182 ~ D8195)	–
Constant Scan		
D8039 Constant scan time	Initial value: 0ms, unit: ms	M8039

Number and Soft Components	Action and Function	Corresponding Special Soft Components
Step Ladder Diagram · Signal Alarm		
D8040 *1 ON state number 1	The smallest number of states that are ON in S0 ~ S899 and S1000 ~ S4095 is stored in D8040, and the lowest number is ON in D8041 The following will save the running status (up to 8 points) to D8047	M8047
D8041 *1 ON state number 2		
D8042 *1 ON state numbe r3		
D8043 *1 ON state number 4		
D8044 *1 ON state number 5		
D8045 *1 ON state number 6		
D8046 *1 ON state number 7		
D8047 *1 ON state number 8		
D8048	Can not be used	-
D8049 *1 ON state Min. number	When M8049 is ON, the Min. number of signal alarm relays S900 ~ S999 that are ON is stored	M8049
D8050 ~ D8060	Can not be used	
D8061	Intelligent controller hardware error code number	M8061
D8063	MOD1 communication error code number	M8063
D8064	Parameter error code number	M8064
D8065	Syntax error code number	M8065
D8066	Ladder diagram error code number	M8066
D8067	Operation error code number	M8067
D8068	Latch of step number where operation error occurred	M8068
D8069	M8065 ~ 7 error step number	M8065 ~ M8067
**1: Processed when the END instruction is executed.*		
High-speed Ring Counter		
D8099	Ring counter of incremental action from 0 ~ 32,767 (unit: 0.1ms, 16 bit)	M8099
System Internal Parameters		
D8101	Can not be used	
D8102	Can not be used	
D8103	Can not be used	
D8104	Can not be used	
D8105	Can not be used	
D8106	Can not be used	
D8107	Can not be used	
D8108	Can not be used	
MOD1 Communication Parameters		
D8120	MOD1 communication format, default 0x8089 (power-off save)	
D8122	MOD1 station number, default value 2 (power-off save)	
D8126	MOD1 communication interval, default 4ms (power-off save)	
D8127	MOD1 response delay, default 4ms (power-off save)	
D8129	MOD1 communication timeout judgment time, default value is 200ms (power-off save)	

<table>
<tr><th>Number and Soft Components</th><th colspan="2">Action and Function</th><th>Corresponding Special Soft Components</th></tr>
<tr><td colspan="4"><b>Index Register</b></td></tr>
<tr><td>D8182</td><td colspan="2">Contents of the Z1 register</td><td>-</td></tr>
<tr><td>D8183</td><td colspan="2">Contents of the V1 register</td><td>-</td></tr>
<tr><td>D8184</td><td colspan="2">Contents of the Z2 register</td><td>-</td></tr>
<tr><td>D8185</td><td colspan="2">Contents of the V2 register</td><td>-</td></tr>
<tr><td>D8186</td><td colspan="2">Contents of the Z3 register</td><td>-</td></tr>
<tr><td>D8187</td><td colspan="2">Contents of the V3 register</td><td>-</td></tr>
<tr><td>D8188</td><td colspan="2">Contents of the Z4 register</td><td>-</td></tr>
<tr><td>D8189</td><td colspan="2">Contents of the V4 register</td><td>-</td></tr>
<tr><td>D8190</td><td colspan="2">Contents of the Z5 register</td><td>-</td></tr>
<tr><td>D8191</td><td colspan="2">Contents of the V5 register</td><td>-</td></tr>
<tr><td>D8192</td><td colspan="2">Contents of the Z6 register</td><td>-</td></tr>
<tr><td>D8193</td><td colspan="2">Contents of the V6 register</td><td>-</td></tr>
<tr><td>D8194</td><td colspan="2">Contents of the Z7 register</td><td>-</td></tr>
<tr><td>D8195</td><td colspan="2">Contents of the V7 register</td><td>-</td></tr>
<tr><td colspan="4"><b>Analog</b></td></tr>
<tr><td>D8220</td><td colspan="2">AI1 range, default 32000 (power-off save)</td><td>D8256</td></tr>
<tr><td>D8221</td><td colspan="2">AI1 bias, default 0 (power-off save)</td><td>D8256</td></tr>
<tr><td>D8222</td><td colspan="2">AI2 range, default 32000 (power-off save)</td><td>D8257</td></tr>
<tr><td>D8223</td><td colspan="2">AI2 bias, default 0 (power-off save)</td><td>D8257</td></tr>
<tr><td>D8224</td><td colspan="2">AO1 range, default 32000 (power-off save)</td><td>D8258</td></tr>
<tr><td>D8225</td><td colspan="2">AO1 offset, default 0 (power-off save)</td><td>D8258</td></tr>
<tr><td>D8226</td><td colspan="2">AO2 range, default 32000 (power-off save)</td><td>D8259</td></tr>
<tr><td>D8227</td><td colspan="2">AO2 offset, default 0 (power-off save)</td><td>D8259</td></tr>
<tr><td>D8256</td><td colspan="2">AI1 input value</td><td>D8220, D8221</td></tr>
<tr><td>D8257</td><td colspan="2">AI2 input value</td><td>D8222, D8223</td></tr>
<tr><td>D8258</td><td colspan="2">AO1 output value, default 0</td><td>D8224, D8225</td></tr>
<tr><td>D8259</td><td colspan="2">AO2 output value, default 0</td><td>D8226, D8227</td></tr>
<tr><td colspan="4"><b>High-speed Counter Input</b></td></tr>
<tr><td>D8244</td><td colspan="2">X2*1 high-speed counter input filter value (power-off save)
(the larger the value, the stronger the filtering effect. when a higher frequency input is required, the filtering value can be lowered appropriately)</td><td></td></tr>
<tr><td>D8245</td><td colspan="2">X3*1 high-speed counter input filter value (power-off save)</td><td></td></tr>
<tr><td>D8246</td><td>Low bit</td><td rowspan="2">X2*1 high-speed counter input frequency</td><td></td></tr>
<tr><td>D8247</td><td>High bit</td><td></td></tr>
<tr><td>D8248</td><td>Low bit</td><td rowspan="2">X3*1 high-speed counter input frequency</td><td></td></tr>
<tr><td>D8249</td><td>High bit</td><td></td></tr>
<tr><td>D8250</td><td colspan="2">X0 high-speed counter input filter value (power-off save)</td><td></td></tr>
<tr><td>D8251</td><td colspan="2">X1*1 high-speed counter input filter value (power-off save)</td><td></td></tr>
<tr><td>D8252</td><td>Low bit</td><td rowspan="2">X0 high-speed counter input frequency</td><td rowspan="2"></td></tr>
<tr><td>D8253</td><td>High bit</td></tr>
<tr><td>D8254</td><td>Low bit</td><td rowspan="2">X1*1 high-speed counter input frequency</td><td rowspan="2"></td></tr>
<tr><td>D8255</td><td>High bit</td></tr>
<tr><td colspan="4"><i>*1. HC10-M0808R-C3-AB corresponds to 4 high-speed counter inputs:
The special registers corresponding to X0 are: [D8250], [D8253,D8252];
The special registers corresponding to X2 are: [D8251], [D8255,D8254];
The special registers corresponding to X4 are: [D8244], [D8246,D8247];
The special registers corresponding to X6 are: [D8245], [D8248,D8249].</i></td></tr>
</table>

Number and Soft Components	Action and Function		Corresponding Special Soft Components
Expansion Module			
D8260	Number of expansion modules		
D8262	Expansion module command communication status		
D8265	Module 1 model		
D8267	Module 2 model		
D8269	Module 3 model		
D8271	Module 4 model		
D8273	Module 5 model		
D8275	Module 6 model		
D8277	Module 7 model		
D8279	Module 8 model		
RND (FN 184)			
D8310	Low bit	RND (FN 184) data for generating random numbers, initial value: K1	
D8311	High bit		
Syntax • Circuit • Operation • I/O Incorrect Installation Step Number Specified by the Actual Installation			
D8312	Low bit	Latch of step number where operation error occurred (32bit)	M8068
D8313	High bit		
D8314	Low bit	M8065 ~ M8067 error step number (32bit)	M8065 ~ M8067
D8315	High bit		
Timing Clock • Positioning			
D8330	DUTY (FN 186) counter for the number of scans of instruction timing clock output 1		M8330
D8331	DUTY (FN 186) counter for the number of scans of instruction timing clock output 2		M8331
D8332	DUTY (FN 186) counter for the number of scans of instruction timing clock output 3		M8332
D8333	DUTY (FN 186) counter for the number of scans of instruction timing clock output 4		M8333
D8334	DUTY (FN 186) counter for the number of scans of instruction timing clock output 5		M8334
Pulse Output Positioning			
D8336	Interrupt input designation		–
D8340	Low bit	[Y000] current value register, initial value: 0[PLS] (power-off save)	–
D8341	High bit		
D8342	[Y000] base speed, initial value: 0[Hz] (power-off save)		–
D8343	Low bit	[Y000] Max. speed, initial value: 100,000 (power-off save)	–
D8344	High bit		
D8345	[Y000] crawling speed, initial value: 1,000[Hz] (power-off save)		–
D8346	Low bit	[Y001] origin return speed, initial value: 50,000[Hz] (power-off save)	–
D8347	High bit		
D8348	[Y000] Acc. time, initial value: 200 (power-off save)		–
D8349	[Y000] Dec. time, initial value: 200 (power-off save)		
D8350	Low bit	[Y001] current value register, initial value: 0[PLS] (power-off save)	–
D8351	High bit		
D8352	[Y001] base speed, initial value: 0[Hz] (power-off save)		–
D8353	Low bit	[Y001] Max. speed, initial value: 100000[Hz] (power-off save)	–
D8354	High bit		
D8355	[Y001] crawling speed, initial value: 1,000[Hz] (power-off save)		–
D8356	Low bit	[Y001] origin return speed, initial value: 50,000[Hz] (power-off save)	–
D8357	High bit		
D8358	[Y001] Acc. time, initial value: 200 (power-off save)		–
D8359	[Y001] Dec. time, initial value: 200 (power-off save)		–

Number and Soft Components	Action and Function		Corresponding Special Soft Components
D8360	Low bit	[Y002] current value register, initial value: 0[PLS] (power-off save)	–
D8361	High bit		
D8362	[Y002] base speed, initial value: 0[Hz] (power-off save)		–
D8363	Low bit	[Y002] Max. speed, initial value: 100,000[Hz] (power-off save)	–
D8364	High bit		
D8365	[Y002] crawling speed, initial value: 1,000[Hz] (power-off save)		–
D8366	Low bit	[Y002] origin return speed, initial value: 50,000[Hz] (power-off save)	–
D8367	High bit		
D8368	[Y002] Acc. time, initial value: 200 (power-off save)		M8338
D8369	[Y002] Dec. time, initial value: 200 (power-off save)		M8338
D8370	Low bit	[Y003] current value register, initial value: 0[PLS] (power-off save)	–
D8371	High bit		
D8372	[Y003] base speed, initial value: 0[Hz] (power-off save)		–
D8373	Low bit	[Y003] Max. speed, initial value: 100,000[Hz] (power-off save)	–
D8374	High bit		
D8375	[Y003] crawling speed, initial value: 1,000[Hz] (power-off save)		–
D8376	Low bit	[Y003] origin return speed, initial value: 50,000[Hz] (power-off save)	–
D8377	High bit		
D8378	[Y003] Acc. time, initial value: 200 (power-off save)		M8338
D8379	[Y003] Dec. time, initial value: 200 (power-off save)		M8338
Ring Counter			
D8398	Low bit	-2,147,483,648 ~ +2,147,483,647 (unit: 1ms) circular up count	M8398
D8399	High bit		
MOD2 Communication Parameters			
D8400	MOD2 communication format, default 0x8089 (power-off save)		
D8402	MOD2 station number, default 2 (power-off save)		
D8406	MOD2 communication interval, default 4ms (power-off save)		
D8407	MOD2 response delay, default 4ms (power-off save)		
D8409	MOD2 communication timeout judgment time, default 200ms (power-off save)		
D8438	MOD2 communication error flag		M8438
Origin Return Reset Signal Device Designation			
D8464	[Y000] clear signal device designation		M8341, M8464
D8465	[Y001] clear signal device designation		M8351, M8465
D8466	[Y002] clear signal device designation		M8361, M8466
D8467	[Y003] clear signal device designation		M8371, M8467
CAN Communication Parameters			
D8470	CAN communication format, default value is 0xA005 (power-off save)		
D8471	CAN communication timeout time, default 20ms (power-off save)		M8471
D8473	ADF send interval time (0 ~ 1000ms, default 10ms) (power-off save)		
D8474	QDF send interval time (0 ~ 1000ms, default 2ms) (power-off save)		
D8475	CAN communication error		
D8476	QDF error station number (host)		M8476
D8479	Send data start address (only free port protocol host is valid) (power-off save)		M8479, M8471
D8480	Receive mailbox 0 identifier 1/lower bit (free port protocol host) (power-off save)		
D8481	Receive mailbox 0 identifier 2/high bit (free port protocol host) or QDF1 send data storage address (connection protocol slave) (power-off save)		

Number and Soft Components	Action and Function	Corresponding Special Soft Components
D8482	Receiving mailbox 0 mask code 1/low bit (free port protocol host) or QDF1 receiving data storage address (connection protocol slave) (power-off save)	
D8483	Receiving mailbox 0 mask code 2/high (free port protocol host) (power-off save)	
D8484	Receive mailbox 0 data start address (free port protocol host) or QDF2 send data storage address (connection protocol slave) (power-off save)	M8484
D8485	Receiving mailbox 1 identifier 1/low bit (free port protocol host) or QDF2 receiving data storage address (connection protocol slave) (power-off save)	
D8486	Receiving mailbox 1 identifier 2/high bit (free port protocol host) (power-off save)	
D8487	Receiving mailbox 1 mask code 1/low bit (free port protocol host) or QDF3 sending data storage address (connection protocol slave) (power-off save)	
D8488	Receiving mailbox 1 mask code 2/high (free port protocol host) or QDF3 receiving data storage address (connection protocol slave) (power-off save)	
D8489	Receiving mailbox 1 data start address (free port protocol host) (power-off save)	D8489

10.2 Supplement of Special Soft Components (M8000 ~, D8000 ~)

Special soft components are the soft components with built-in functions that are prepared in advance from the perspective of intelligent controller operation. The following describes their use.

RUN Monitoring, Use of Initial Pulse [M8000 ~ M8003]

RUN Monitoring (M8000, M8001)

The RUN monitor (M8000, M8001) that displays the operating status of the intelligent controller can be used as a driving condition for instructions, or it can be used in an external display that displays "normal operation".

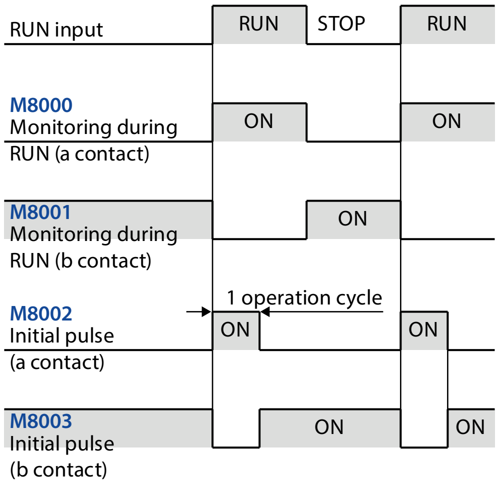
The action timing of the flag bit is shown in the right figure.

Initial Pulse (M8002, M8003)

Initial pulse (M8002, M8003) after the intelligent controller starts running, only momentarily (1 calculation cycle) is ON or OFF.

This pulse can be used as an initial setting signal in a program such as initialization of a program or writing of a predetermined value.

The action timing of the flag bit is shown in the right figure.



Watchdog Timer Time [D8000]

The watchdog timer monitors the calculation (scanning) time of the intelligent controller. When it does not complete within the specified time, the (ERROR (ERR)) LED is turned on, and all outputs are turned OFF.

The initial value of 200ms is transmitted from the system at power-on, but if the executed program exceeds this time, the value of D8000 must be changed in the program.

Watchdog Timer Error Conditions

In the following table, a watchdog timer error may also occur, so please enter the above program near the initial step to extend the watchdog timer time.

Watchdog Timer Error Conditions		
1	Precautions when connecting many special function units/modules	In a system configuration in which a large number of special function units/modules are connected, the initialization time of the buffer memory area executed when the intelligent controller is running becomes longer, the calculation time will be longer, and a watchdog timer error may occur.
2	Precautions when there are many high-speed counters (software counters)	When programming multiple high-speed counters to count high-frequency pulses at the same time, the calculation time will be prolonged, and a watchdog timer error may occur.

Watchdog Timer Reset Method

Unlike the change of the watchdog timer time itself, the WDT (FN 07) instruction can be used to reset the watchdog timer in the sequence program.

It is recommended to use WDT (FN 07) instruction to reset the watchdog timer when the calculation time of a specific sequence program becomes long or when many special function units/modules are connected.

Precautions When Changing the Watchdog Timer Time

The watchdog timer time can be set to a Max. of 32,767ms, so if there is no problem in operation, please set it to the initial value (200ms).

Operation Time (Monitoring) [D8010 ~ D8012]

The current, Min., and Max. values (unit: 0.1ms) of the scan time (computation time) of the intelligent controller are stored in D8010 to D8012.

In addition, when using the constant scan function, these values include the wait time for the constant scan time.

D8010: Current value
D8011: Minimum value
D8012: Maximum value

The values of these soft components can be monitored by peripheral devices.

Internal Clock [M8011 ~ M8014]

With 4 internal time bases of 10ms, 100ms, 1s, and 60s, it starts to work after the intelligent controller is powered on.

Note: The clock keeps running even when the intelligent controller is stopped. Therefore, the rising edge of the RUN monitor (M8000) and the start time of the clock are not synchronized.

Real-time Clock [M8015 ~ M8019, D8013 ~ D8019]

1. Distribution of special auxiliary relays (M8015 ~ M8019) and special data registers (D8013 ~ D8019).

Number	Name	Action • Function
M8015	Calibration time	When ON, the clock stops On the edge of ON→OFF, write the time of D8013 ~ D8019, and act again
M8016	Show time stop	When ON, stop displaying time (timekeeping still works)
M8018	Installation inspection	Always ON
M8019	RTC error	When calibrating the time, when the data of the special data register exceeds the setting range, it is ON

Number	Name	Set Value Range	Action • Function
D8013	Second	0 ~ 59	Write the initial value of the calibration time, or read the initial time • The year corresponds to 1980 ~ 2079 • Leap year correction: Yes
D8014	Minute	0 ~ 59	
D8015	Hour	0 ~ 23	
D8016	Day	1 ~ 31	
D8017	Month	1 ~ 12	
D8018	Year	0 ~ 99 (last two digits of the gregorian calendar)	
D8019	Week	0 (Sunday) ~ 6 (Saturday)	

2. To calibrate the real-time clock, perform any of the following operations:

- Time calibration dedicated instruction TWR.

  For the setting method, please refer to the introduction of TWR instruction.

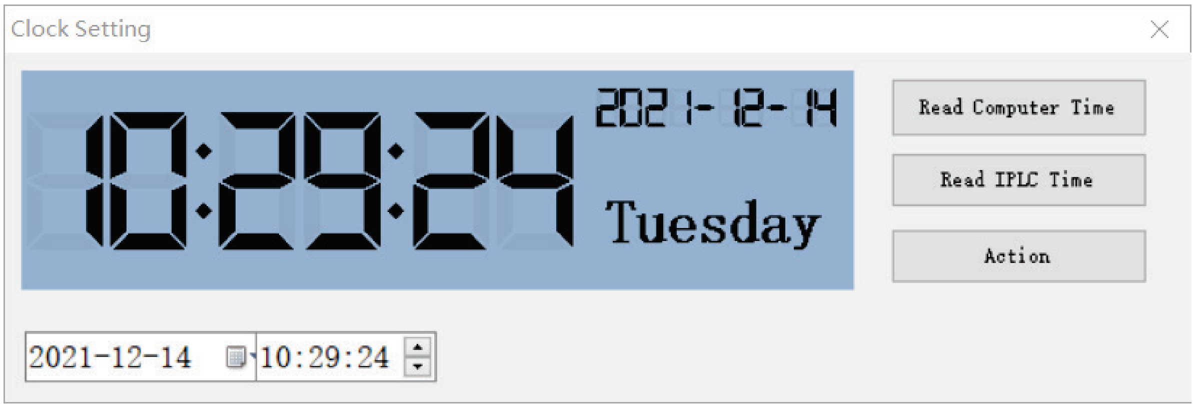
- Programming software settings.

  Use HCStudio programming software to set up.

  Confirm that HCStudio is connected to HC10; Select "Clock Setting" of "PLC(P)" in the menu bar to enter the clock setting interface, as shown in the figure below.

  Click "Read computer time" to get the current computer time (can also set it manually).

  Click "Execute" to write the time into HC10. If the setting is successful, a prompt box of "Completed" will be displayed.

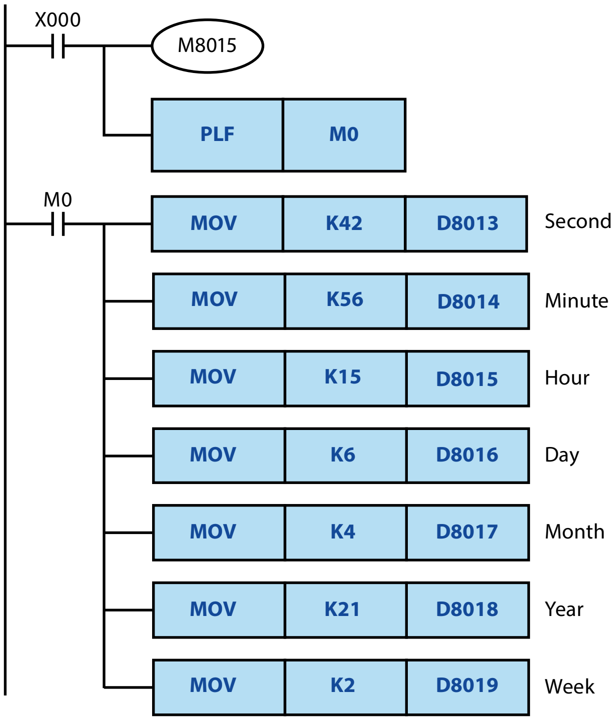


Clock Setting

10:29:24 2021-12-14 Tuesday

2021-12-14 10:29:24

- Special address settings.



For example: **15:56:42 Tuesday, April 6, 2021**

When setting, please set 2~3 minutes earlier than the correct time, write the program on the left into the programmable controller and run it. Turn X000 ON. When the correct time is reached, set the time after turning the input switch X000 from ON to OFF.
Start timing action.

Adjustment of the Input Filter [D8020]

The ordinary input terminals are respectively equipped with a digital filter circuit of 0 ~ 100ms. The content of special data register D8020 0 ~ 100 determines which digital filter constant to use.

After the power is turned off and on, the content of D8020 will automatically change to 10 (10ms).

Note: For the main module with more than 32 points, only X0 ~ X7 of the input terminals on the main module are set by D8020 to set the filter value, and the subsequent X terminal filter value is fixed at 40ms.

Clear Instruction [M8031, M8032].

All devices (image memory area) of the intelligent controller can be cleared without holding or holding area.

M8031 (does not keep clearing all memory areas), M8032 (does not keep clearing all memory areas) all are executed during the program execution cycle, that is, setting this bit during operation will take effect after the END instruction.

Soft Component Nunber	Clear Soft Component
M8031 (no holding area)	• Contact image of output relay (Y), general auxiliary relay (M), general status (S) • Timer (T) contacts, timing coils • Contacts for general counters, counting coils, reset coils • Current value of general-purpose data register (D) • Timer (T) current value register • Current value register for general counter (C) • General extension register
M8032 (holding area)	• Contact image of auxiliary relay (M), holding state (S) • Contacts for holding counters and high-speed counters, counting coils, reset coils • Current value register of holding data register (D) • Current value register for holding counter and high-speed counter

Memory Hold Stop [M8033] (Output Hold during STOP)

If the special auxiliary relay M8033 is driven, after the intelligent controller changes from RUN to STOP, the output state at RUN can be maintained as it is.

Constant Scan Mode [M8039, D8039] (Fixed Operation Processing Time)

Turn on the special auxiliary relay M8039, and after writing the target scan time (unit 1ms) in the special data register D8039, the calculation cycle of the intelligent controller will not be lower than this value. That is, even if the operation ends early, it will consume the remaining time before returning to step 0.

Note		
1	When an instruction that is executed in synchronization with the scan is used	• When using RAMP (FN 67), HKY (FN 71), SEGL (FN 74) and other instructions that are executed synchronously with the scan, it is recommended to use this constant scan mode, or to switch on at regular intervals through a timer interrupt. • When using the HKY (FN 71) instruction, the keyboard input filter may cause a response delay, so a scan time of more than 20ms is required.
2	Display scan time (D8010 ~ D8012)	The time specified in the constant scan mode is included in the display of the scan time of D8010 to D8012.

Program Encryption Function

HC10 supports two encryption methods: Hardware encryption and password encryption. The two encryption methods are mutually exclusive, and the other encryption cannot be turned on in one encryption state.

- Hardware encryption: It uses M8511 for encryption. After encryption, the program is forbiddened to be read, and program downloading and monitoring can still be performed. Downloading the program will not clear the encryption state, and only use the program clear function to clear the encryption state.
- Password encryption: HCStudio is used for password encryption, decryption and clearing. In the encrypted state, the program reads and downloads require a password, and can still be monitored freely. The program clear function can still clear the program and the password together.

Chapter 11 Troubleshooting and Error Code

11.1 Supplementary Description of Soft Components for Error Detection

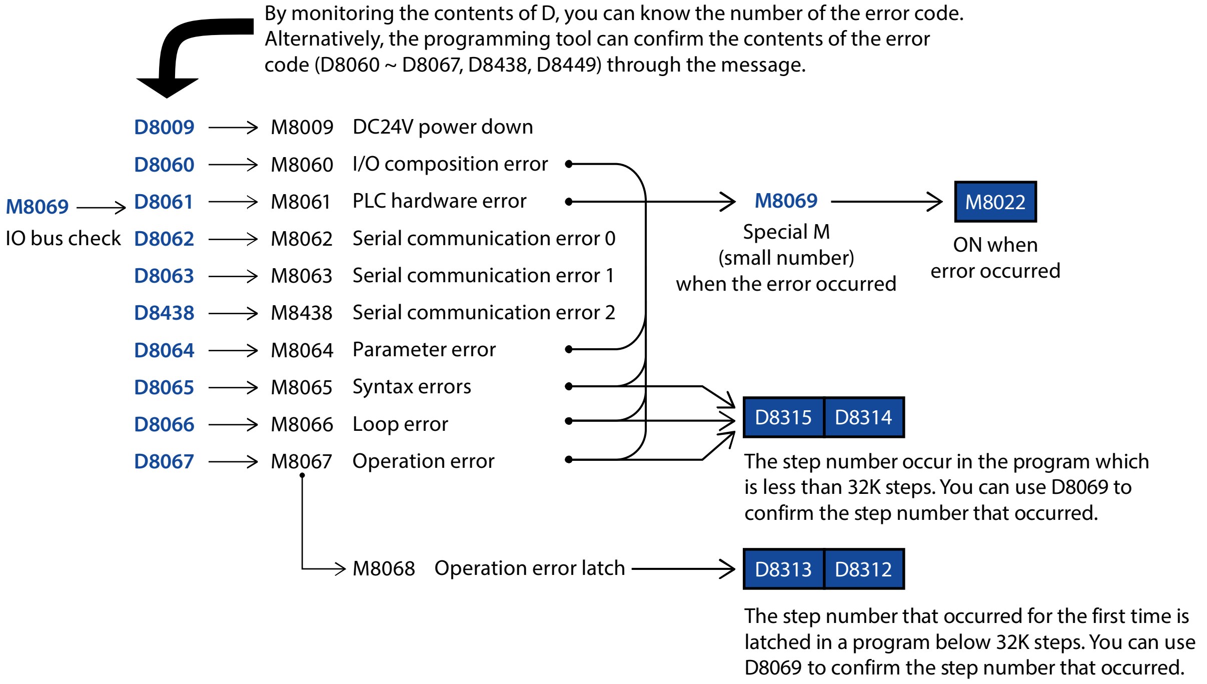
Error Detection (M8060 ~ /D8060 ~)

When any one of M8060, M8061, M8064 ~ M8067 is turned on, the smaller number is stored in D8004, and M8004 operates.

Operation Relationship of Special Soft Components Error Detection

The special auxiliary relays (M8000 ~ M8511) for error detection and the special data registers (D8000 to D8511) operate in the following relationship.

Monitor the contents of the auxiliary relays and data registers from the programming tool and use the intelligent controller diagnostics to see what happened.


D8009 → M8009 DC24V power down
D8060 → M8060 I/O composition error
M8069 → D8061 → M8061 PLC hardware error
IO bus check D8062 → M8062 Serial communication error 0
D8063 → M8063 Serial communication error 1
D8438 → M8438 Serial communication error 2
D8064 → M8064 Parameter error
D8065 → M8065 Syntax errors
D8066 → M8066 Loop error
D8067 → M8067 Operation error
M8069
M8022
D8315
D8314
M8068 Operation error latch
D8313
D8312

By monitoring the contents of D, you can know the number of the error code. Alternatively, the programming tool can confirm the contents of the error code (D8060 ~ D8067, D8438, D8449) through the message.

Special M (small number) when the error occurred

ON when error occurred

The step number occur in the program which is less than 32K steps. You can use D8069 to confirm the step number that occurred.

The step number that occurred for the first time is latched in a program below 32K steps. You can use D8069 to confirm the step number that occurred.

Detection Timing of Error

Error Item	State of ERROR LED	State of Intelligent Controller	Detection Timing of Error		
			Power OFF→ ON	STOP→RUN	Others
M8060 I/O composition error	Light off	RUN	Check	Check	-
M8061 intelligent controller hardware error	Light on	STOP	Check	-	Always
M8062 serial communication error 0 [channel 0]	Light off	RUN	-	-	When receiving a signal from the other station
M8063 serial communication error 1 [channel 1]	Light off	RUN	-	-	When receiving a signal from the other station
M8438 serial communication error 2 [channel 2]	Light off	RUN	-	-	When receiving a signal from the other station
M8064 parameter error	Light flash	STOP	Check	Check	When changing programs (STOP) when transferring programs (STOP)
M8065 syntax error	Light flash	STOP			
M8066 loop error	Light flash	STOP			
M8067 operation error	Light off	RUN	-	-	RUN
M8068 operation error latch	Light off	RUN			
M8109 output refresh error	Light off	RUN	-	-	Always

Error Item	State of ERROR LED	State of Intelligent Controller	Detection Timing of Error		
			Power OFF→ ON	STOP→RUN	Others
M8316 specified error when I/O not installed	Light off	RUN	-	-	RUN
M8318 BFM initialization failed	Light off	RUN	-	Check	-
M8449 special module error	Light off	RUN	-	-	Always

11.2 Error Code List and Solutions

When a program error of the intelligent controller occurs, the error codes stored in the special data registers D8060 ~ D8067, D8438, and D8449 and their solutions are shown below.

Error Code	Action on Error	Error Content	Solutions
Intelligent Controller Hardware Error			
0000	-	Nothing unusual	
6101	Stop running	RAM error	
6102		Operation loop error	
6105		Watchdog timer error	Sampling (computation time) exceeds the value of D8000. Please confirm the procedure.
Parameter Error			
0000	-	Nothing unusual	
6401	Stop running	Procedure and verification are inconsistent	Please stop the intelligent controller and set the parameters correctly.
6402		Incorrect memory capacity setting	
6403		Incorrect holding area setting	
6404		Incorrect comment area setting	
6405		Incorrect file register area setting	
6406		BFM initial value data and verification are inconsistent	
6407		BFM initial value data abnormal	
6409		Other setting errors	
Syntax Error			
0000	-	Nothing unusual	
6501	Stop running	Wrong combination of command-soft component symbol- soft component number	When writing a program, please check that each instruction is used correctly. If an error occurs, please modify the instruction in programming mode.
6502		No OUT T, OUT C before the set value	
6503		No setting value after OUT T, OUT C Insufficient Operand to apply instructions	
6504		Label number duplicate Interrupt input and high-speed counter input duplicate	
6505		Soft component number is out of range	
6506		Undefined directive used	
6507		Label number (P) is incorrectly defined	
6508		Interrupt input (I) is incorrectly defined	
6509		Others	
6510		MC's nested number has wrong size relationship	

Error Code	Action on Error	Error Content	Solutions
Loop Error			
0000	-	Nothing unusual	
6610	Stop running	LD and LDI have been used more than 9 times	Such an error occurs when the instruction combination method as a whole of the circuit block is incorrect or when the relationship of the paired instructions is incorrect. Please modify the interrelationship of the instructions in programming mode.
6611		Too many ANB and ORB instructions compared to LD and LDI instructions	
6612		Too little ANB and ORB instructions compared to LD and LDI instructions	
6613		MPS has been used continuously for more than 12 times	
6614		Missing MPS	
6615		Missing MPP	
6616		Missing coils between MPS-MRD and MPP, or relationship error	
6617		Instructions that should start from the bus are not connected to the bus STL, RET, MCR, P, I, DI, EI, FOR, NEXT, SRET, IRET, FEND, END	
6618		Instructions that can only be used in the main program are outside the main program (interrupts, subroutines, etc.) STL, MC, MCR	
6619		Instruction STL, RET, MC, MCR, I, IRET cannot be used between FOR-NEXT	Such an error occurs when the instruction combination method as a whole of the circuit block is incorrect or when the relationship of the paired instructions is incorrect. Please modify the interrelationship of the instructions in programming mode.
6620		FOR-NEXT nested beyond	
6621		Relationship between FOR-NEXT numbers is incorrect	
6622		No NEXT instruction	
6623		No MC instruction	
6624		No MCR instruction	
6625		STL has been continuously used more than 9 times	
6626		Instruction MC, MCR, I, SRET, IRET cannot be uesed between STL-RET	
6627		No STL instruction	
6628		Instruction I, SRET, IRET in the main program that cannot be used by the main program	
6629		No P, I	
6630		No SRET, IRET instructions STL-RET or MC-MCR instruction in subroutine	
6631		SRET instruction is available in places where SRET instruction cannot be used	
6632		FEND instruction is available in places where FEND instruction cannot be used	
6633		No END instruction	

Error Code	Action on Error	Error Content	Solutions
Operation Error			
0000		Nothing unusual	
6701	Keep running	• Jump destination address without CJ, CALL • Index modification result, label is undefined, and when it is out the range of P0 ~ P4095 • P63 was executed in the CALL instruction. Because P63 is a label that jumps to END, it cannot be used in the CALL instruction	These are errors that occur during the execution of the operation. Please modify the program or check the contents of the operand of the application instructions. Even if no syntax or loop errors occur, operation errors may occur for the following reasons. For example: T500Z itself has no errors, but if the operation result is Z = 100, it will become T600, so the device number will exceed.
6702		CALL nesting exceeds 6	
6703		Broken nesting exceeds 3	
6704		FOR-NEXT nesting exceeds 6	
6705		Operand of application instruction is a soft component other than object soft component	
6706		The soft component number range or data value of applied instruction operand exceeds	
6707		Access to file registers without setting file register parameters	
6709		Others (incorrect branch, etc.)	These are errors that occur during the execution of the operation. Please modify the program or check the contents of the operand of the application instructions. Even if no syntax or loop errors occur, operation errors may occur for the following reasons. For example: T500Z itself has no errors, but if the operation result is Z = 100, it will become T600, so the device number will exceed.
6710		Mismatch between parameters	In a shift instruction or the like, there is a case where the source operand and the target operand overlap.
6730		Sampling time (Ts) is ouside the target range (Ts ≤ 0)	"Stop PID Calculation" A data error occurred in the setting value of the control parameter or in the PID calculation. Please check the contents of the parameters.
6732		Input filter constant (α) is outside the target range (α < 0 or 100 ≤ α)	
6733		Proportional gain (KP) is outside the target range (KP < 0)	
6734		Integration time (TI) is out of range (TI < 0)	
6735		Differential gain (KD) is out of range (KD < 0 or 201 ≤ KD)	
6736		Differential time (TD) is out of the target range (TD < 0)	
6740		Sampling time (TS) ≤ operation period	"Continue Self-tuning" Treated as sampling time (TS) = cycle time (computation period) Calculate and continue execution.
6742		Measured value change exceeds (ΔPV <-32,768 or +32,767 < ΔPV)	"Continue PID Calculation" Each parameter continues to run at the Max. or Min. value.
6743		Deviation exceeds (EV < -32,768 or +32,767 < EV)	
6744		Integral calculated value exceeds (other than -32,768 ~ +32,767)	
6745		Derivative value exceeded due to differential gain (KD) exceeded	
6746		Derivative calculation value exceeded (other than -32,768 ~ +32,767)	
6747		PID operation result exceeds (other than -32,768 ~ +32,767)	

Error Code	Action on Error	Error Content	Solutions
Operation Error			
6748	Keep running	PID output upper limit set value < output lower limit set value	"Replace Output Upper Limit and Output Lower Limit →Continue PID Calculation" Please check if the settings of the target are correct.
6749		PID input change alarm set value and output change alarm set value are abnormal (set value < 0)	"No Alarm Output→Continue PID Calculation" Lease check if the settings of the target are correct.
6753		"The Limit Cycle Act" Output setting value for auto tuning is abnormal [ULV (upper limit) ≤ LLV (lower limit)]	"Auto-tuning Forced End→Do Not Transfer to PID Calculation" Please check if the settings of the target are correct.
6754		"The Limit Cycle Act" Auto-tuning PV threshold (lag) set value abnormal (SHPV <0)	
6755		"The Limit Cycle Act" Self-tuning state transition abnormal (the data of the device that manages the state transition has been rewritten abnormally)	"Auto-tuning Forced End→Do Not Transfer to PID calculation" Please check whether the device occupied by the PID instruction has been rewritten in the program.
6756		"The Limit Cycle Act" The result is abnormal due to the self-tuning measurement time (τon > τ, τon < 0, τ < 0)	"Auto-tuning Forced End→Do Not Transfer to PID Calculation" The time required for auto-tuning is longer than originally required. Please confirm that the difference between the upper and lower limits of the output value for auto-tuning (ULV-LLV) becomes larger, and the values of the input filter constant α and the PV threshold SHPV for auto-tuning become smaller after waiting for the measures, do you see the effect of improvement.
6757		"The Limit Cycle Act" The proportional gain of the self-tuning result exceeds (KP = 0 ~ 32767)	"Auto Tuning Completed (KP = 32767)→Move to PID Calculation" The change in the measured value (PV) is small relative to the output value. Please increase the measured value (PV) by 10 times and input it to amplify the change in PV during auto-tuning.
6758		"The Limit Cycle Act" Integration time of auto-tuning result exceeds (TI = 0 ~ 32767)	"Auto Tuning Completed (KP = 32767)→Move to PID Calculation" The time required for auto-tuning is longer than originally required. Please confirm that the difference between the upper and lower limits of the output value for auto-tuning (ULV-LLV) becomes larger, and the values of the input filter constant α and the PV threshold SHPV for auto-tuning become smaller after waiting for the measures, do you see the effect of improvement.
6759		"The Limit Cycle Act" Differential time of auto-tuning result (TD = 0 ~ 32767)	
6765		Application instruction used incorrectly	Please confirm whether you have exceeded the limit of application instructions that are limited in the program.

Chapter 12 Instruction List

Basic Instruction Summary Table

Instruction Mark	Function	Reference Page
LD	The logical operation of the A contact begins	24
LDI	The logic operation of the B contact begins	24
LDP	Operation begins when the rising edge is detected	29
LDF	Operation begins when the falling edge is detected	29
AND	A contact in serial	27
ANI	B contact in serial	27
ANDP	Serial connection detected at rising edge	29
ANDF	Tandem connection detected at falling edge	29
OR	A contact in parallel	28
ORI	B contact in parallel	28
ORP	Parallel connection detected at rising edge	29
ORF	Parallel connection detected at falling edge	29
ANB	Serial connection of circuit blocks	30
ORB	Parallel connection of circuit blocks	30
MPS	Push into the stack	31
MRD	Read stack	31
MPP	Popup stack	31
INV	Reverse of operation result	33
MEP	Conduction on rising edge	34
MEF	Conduction on falling edge	34
OUT	Coil drive	25
SET	Action retention	35
RST	Release the hold action, clear the current value and register	35
PLS	Rising edge differential output	34
PLF	Falling edge differential output	34
MC	Connect to the public contact	32
MCR	Disconnect to the public contact	32
NOP	No processing	37
END	End of program and input and output processing and return 0 step	37

Step Ladder Diagram Instruction

Instruction Mark	Function	Reference Page
STL	Step ladder diagram (beginning of step ladder diagram)	38
RET	Back (end of step ladder diagram)	38

Summary of Application Instructions

Instruction Mark	FN No.	Function	Reference Page
CJ	00	Conditional jump	40
CALL	01	Subroutine call	42
SRET	02	Subroutine return	43
IRET	03	Interrupt return	43
EI	04	Allow interrupt	44
DI	05	Disable interrupt	44
FEND	06	End of main program	45
WDT	07	Watchdog timer	46
FOR	08	Start of loop range	47
NEXT	09	End of loop range	48
CMP	10	Compare	50
ZCP	11	Interval comparison	51

Instruction Mark	FN No.	Function	Reference Page
MOV	12	Transfer	52
SMOV	13	Bit shift	53
CML	14	Reverse transfer	54
BMOV	15	Bulk transfer	55
FMOV	16	Multicast	56
XCH	17	Exchange	57
BCD	18	BCD conversion	58
BIN	19	BIN conversion	59
ADD	20	BIN addition	61
SUB	21	BIN subtraction	62
MUL	22	BIN multiplication	63
DIV	23	BIN division	64
INC	24	BIN plus one	65
DEC	25	BIN minus one	66
WAND	26	Logical AND	67
WOR	27	Logical OR	68
WXOR	28	Logical XOR	69
NEG	29	Complement	70
ROR	30	Cycle shift right	72
ROL	31	Cycle shift left	74
RCR	32	Shift right with carry	76
RCL	33	Shift left with carry	78
SFTR	34	Bit shift right	80
SFTL	35	Bit shift left	81
WSFR	36	Word shift right	82
WSFL	37	Word shift left	83
SFWR	38	Shift write (for FIFO/FILO control)	84
SFRD	39	Shift readout (for FIFO control)	85
ZRST	40	Batch reset	88
DECO	41	Decode	89
ENCO	42	Coding	91
SUM	43	ON bit	92
BON	44	Judgement of ON bit	93
MEAN	45	Average value	94
ANS	46	Signal alarm set	95
ANR	47	Signal alarm reset	96
SQR	48	BIN square operation	97
FLT	49	BIN integer→binary floating point conversion	98
REF	50	Input and output refresh	100
MTR	52	Matrix input	101
HSCS	53	Comparison set (for high-speed counter)	102
HSCR	54	Comparison reset (for high-speed counter)	103
HSZ	55	Section comparison (for high-speed counter)	104
SPD	56	Pulse density	105
SER	61	Data retrieval	107
ABSD	62	Cam control absolute mode	109
INCD	63	Cam control relative mode	111
TTMR	64	Teach timer	112
STMR	65	Special timer	113
ALT	66	Alternate output	115
RAMP	67	Ramp signal	116
SORT	69	Data sorting	117
TKY	70	Numeric key input	120

Instruction Mark	FN No.	Function	Reference Page
HKY	71	Hex number key input	122
SEGD	73	7-segment decoder	124
FROM	78	Module buffer data read	124
TO	79	Module buffer data write	127
RD3A	176	Analog module readout	129
PRUN	81	Octal transmission	131
CCD	84	Check code	133
PID	88	PID operation	135
ZPUSH	102	Batch saving of index registers	139
ZPOP	103	Index register recovery	141
ECMP	110	Binary floating point comparison	143
EZCP	111	Binary floating point interval comparison	144
EMOV	112	Binary floating point data transfer	145
EBCD	118	Conversion from binary floating point number to decimal floating point number	146
EBIN	119	Conversion from decimal floating point number to binary floating point number	147
EADD	120	Binary floating point addition	148
ESUB	121	Binary floating point subtraction	149
EMUL	122	Binary floating point multiplication	150
EDIV	123	Binary floating point division	151
EXP	124	Binary floating point exponential operation	152
LOGE	125	Binary floating point number natural logarithmic operation	153
LOG10	126	Binary floating point number common logarithmic operation	154
ESQR	127	Binary floating-point number square operation	155
ENEG	128	Binary floating point sign flip	156
INT	129	Conversion from binary floating point number to BIN integer	157
SIN	130	Binary floating point number SIN operation	158
COS	131	Binary floating point number COS operation	158
TAN	132	Binary floating point number TAN operation	159
ASIN	133	Binary floating point number SIN^{-1} operation	160
ACOS	134	Binary floating point number COS^{-1} operation	161
ATAN	135	Binary floating point number TAN^{-1} operation	162
RAD	136	Conversion of binary floating point number angle→ radian	163
DEG	137	Conversion of binary floating point numbers in radian →angle	163
WSUM	140	Data separation in bytes	167
WTOB	141	Data combination in bytes	169
BTOW	142	4-bit combination of 16-bit data	171
UNI	143	4-bit separation of 16-bit data	171
DIS	144	High and low byte swap	173
SWAP	147	Data sorting 2	174
SORT2	149	Data separation in bytes	167
PLSY	57	Pulse output	179
PLSV	157	Variable speed pulse output	179
DSZR	150	Return to origin with DOG search	182
ZRN	156	Return to origin	187
DVIT	151	Interrupt positioning	190
DRVI	158	Relative positioning	193
DRVA	159	Absolute positioning	193
TCMP	160	Clock data comparison	197

Instruction Mark	FN No.	Function	Reference Page
TZCP	161	Clock data interval comparison	197
TADD	162	Clock data addition	199
TSUB	163	Clock data subtraction	200
HTOS	164	Second conversion of hour, minute, and second data	201
STOH	165	[Hour, minute, second] conversion of second data	202
TRD	166	Read clock data	203
TWR	167	Write clock data	204
HOUR	169	Chronograph	205
GRY	170	Gray code conversion	207
GBIN	171	Gray code inverse conversion	208
RND	184	Generate random numbers	210
DUTY	186	Generate timing pulses	211
CRC	188	CRC operation	213
BK+	192	Addition of data blocks	216
BK-	193	Subtraction of data blocks	218
BKCMP=	194	Comparison of data blocks S1 = S2	220
BKCMP>	195	Comparison of data blocks S1 > S2	220
BKCMP<	196	Comparison of data blocks S1 < S2	220
BKCMP<>	197	Comparison of data blocks S1 ≠ S2	220
BKCMP<=	198	Comparison of data blocks S1 <= S2	220
BKCMP>=	199	Comparison of data blocks S1 >= S2	220
FDEL	210	Data deletion of data table	224
FINS	211	Data insertion of data table	225
POP	212	Read last-In data [for FILO control]	226
SFR	213	16-bit data n-bit shift right (with carry)	228
SFL	214	16-bit data n-bit shift left (with carry)	229
LD=	224	Contact comparison LD S1 = S2	231
LD >	225	Contact comparison LD S1 > S2	231
LD <	226	Contact comparison LD S1 < S2	231
LD<>	228	Contact comparison LD S1 ≠ S2	231
LD<=	229	Contact comparison LD S1 <= S2	231
LD>=	230	Contact comparison LD S1 >= S2	231
AND=	232	Contact comparison AND S1 = S2	232
AND>	233	Contact comparison AND S1 > S2	232
AND<	234	Contact comparison AND S1 < S2	232
AND<>	236	Contact comparison AND S1 ≠ S2	232
AND<=	237	Contact comparison AND S1 <= S2	232
AND>=	238	Contact comparison AND S1 >= S2	232
OR=	240	Contact comparison OR S1 = S2	233
OR>	241	Contact comparison OR S1 > S2	233
OR<	242	Contact comparison OR S1 < S2	233
OR<>	244	Contact comparison OR S1 ≠ S2	233
OR<=	245	Contact comparison OR S1 <= S2	233
OR>=	246	Contact comparison OR S1 >= S2	233
LIMIT	256	Upper and lower limit position control	235
BAND	257	Dead band control	237
ZONE	258	Zone control	239
SCL	259	Fixed coordinates (coordinate data of different points)	241
SCL2	269	Fixed coordinate 2 (X/Y coordinate data)	244
EXTR	180	CAN communication	248
ADPRW	276	Modbus read/write	250

Hpmont Group Company

Shenzhen Hpmont Techmology Co., Ltd.
Add: Building 28, Wangjingkeng Industry Park, Xili Dakan, Nanshan District, Shenzhen, China, 518055
Tel: 86 755 2679 1688
Fax: 86 755 2699 4395
Email: marketing@hpmont.com

HPMONT (Hong Kong) Co., Ltd.
Add: Room 709, 7/F, Silvercord Tower 1, 30 Canton Road, Tsim Sha Tsui, -Kowloon. Hong Kong
Tel: +852 6607 2243
Email: info.hk@hpmont.com.hk

Mont Korea Co., Ltd.
Add: Ace pyungchon tower, #811, 361 Slimin-daero, Dongan-gu, Anyang-si, Gyeonggi-Do, 14057
Tel: +82-31-345-8181
Email: info.kr@hpmont.com.hk

Hpmont (Malaysia) Sdn Bhd
Add: VO3-11-20, Lingkaran SV, Sunway Velocity, 55100 Kuala Lumpur
Tel: +603 9202 8812
Email: info.ma@hpmont.com.hk

Hpmont (Taiwan) Co., Ltd.
Add: 17F., No. 368-3, Sec. 2, Gaotie S. Rd., Zhongli Dist., Taoyuan City 320, Taiwan
Tel: +886 905 333 600
Email: info.tw@hpmont.com.hk

Hpmont (Turkey) Teknoloji Ltd. Sti.
Add: Floor 3, Building 20, Fil Yokuşu Street, Cevizli District, Maltepe/Istanbul
Tel: +90 533 261 38 76
Email: info.tr@hpmont.com.hk

www.hpmont.com