X3 Series PLC Software

User's manual

Prior to use, please read this user's manual carefully.

CAUTION:Please keep this user's manual for future reference.

Revision History:

Version Number	Revision Date	Revision Description
V1.00	2019.10.01	First edition
V1.01	2020.04.20	(1) Updated the module address and added a host computer configuration example; (2) Organized the error code table; (3) Correct errors in the document .
V1.02	2020.05.07	(1) Add support for ALT and SORT commands; (2) Add instructions for using basic commands; (3) Added a brief introduction to the use of BD board .
V1.03	2020.06.29	(1) Add module offline configuration clear flag and module; configuration examples for considerations of upper computer configuration module; (2) DSZR supports common terminal input; (3) Added RS2 frame format instructions; (4) Added simple instructions for using step ladder diagram.
V1.04	2020.09.10	(1) Most instructions are combined with 16-bit and 32-bit instructions. Note that for 32-bit registers, they are two adjacent 16-bit word elements; (2) Rearrange instruction description and ladder diagram; (3) BD board ID name changed, available types increased; (4) Added RS2 instruction header, footer, etc. usage precautions and the minimum length of data transmission and reception is 1; (5) Changed module register error description, added offline configuration precautions; (6) The single coil writing function is now supported (version 2129 and above); (7) Added concise instructions for using SFC and step ladder diagrams; (8) Encryption feature changes; (9) Added the configuration table method for MODBUS RTU configuration and adjusted the structure of the communication part of the manual; (10) Added ABSD (62), INCD (63) instructions; (11) Added 4 special data registers for local IP display and a special auxiliary register for MODBUS RTU execution status mode.
V1.05	2020.10.19	(1) Added ZRN2/PLSN/PLSF/ABSLINE/RELLINE instructions; (2) Modify some positioning instructions, DVIT/DRVA/DRVI supports dynamic frequency modification; (3) The default base speed is 0; (4) The encryption function is reclassified into primary encryption and advanced encryption; (5) Added detailed description of MODBUS related register actions and error information.
V1.06	2020.12.7	(1) Add 8 IP timeout flags for abnormal error reporting. The usage is in the Ethernet precautions; (2) Corrected some errors in the document; (3) Add serial port protocol related register descriptions in the register table.
V1.07	2021.12.15	(1) Change the RWER command usage (2138 firmware has changed, but the manual has been changed first); (2) Change SD8506 usage.
V1.08	2021.07.08	Change registers, mainly ring counter, Y5 axis register related.
V1.09	2022.08.22	(1) Added instructions for using Ethernet free port communication commands; (1) Added battery voltage related SM8005, SD8005 registers.
V1.10	2022.10.28	Modify TRECV instruction usage
V1.11	2022.12.29	Add instructions for using the Ethernet free port
V1.12	2022.04.15	New PID function
V1.13		Added X2plc high-speed counter terminal description

Contents

1. Detailed explanation of X series memory

1.1 Raynen X3 Resource Configuration

Name	Content	Component Type	Content
X	256 points	Bit Components	X0 to X377 (octal)
Y	256 points	Bit Components	Y0 to Y377 (octal)
M	7680 points	Bit Components	M0 to M7679
SM	512 points	Bit Components	SM8000~SM8511
S	4096 points	Bit Components	S0~S4095
V	8 o'clock	Word component	V0~V7
Z	8 o'clock	Word component	Z0~Z7
T	512 points	Word component	T0~T511
C	200 points	16-bit word element	C0~C199
C (32-bit)	56 points	32-bit word element	C200~C255
D	8000 points	Word component	D0~D7999
SD	512 points	Word component	SD8000~SD8511
R	32768 points	Word component	R0~R32767
Input interrupt	6 o'clock		I00*~I50*
Timer interrupt	3 o'clock		I6**~I8**
Counter interrupt	6 o'clock		I010~I060
Master control with nesting	8 o'clock		N0~N7
Pointer P	4096 points		P0~P4095
Constant K			16-bit range -32768~+32767 32-bit range -2147483648~+2147483647
Constant H			16-bit range 0~FFFF 32-bit range 0~FFFFFFFF
Real number E			-1.0×2 128 ～ -1.0×2 -126 , 0, 1.0×2 -126 ～ 1.0×2 128 (single-precision floating point range)
String			Half-width characters enclosed by "" Maximum length: 32 characters

Special Notes:

(1) Timer classification

100ms type 0.1~3276.7s	10ms type 0.01~327.67s	1ms cumulative type 0.001~32.767s	100ms cumulative type 0.1~3276.7s	1ms type 0.001~32.767s
T0~T199 200 points T192~T199 Subroutine specific	T200~T245 46 points	T246~T249 4 o'clock Execution interrupt Keep	T250~T255 6 o'clock Keep	T256~T511 256 points

Among them, the usage of subroutine-specific and execution interrupt holding is quite special. The ordinary timer only counts when the subroutine is executed. If the subroutine is not executed, the timing will be interrupted. If the dedicated timer is started in the subroutine, even if the subroutine is not executed, the timer will continue to count, thus ensuring the accuracy of the timing.

(2) When using the counter, pay attention to the registers of the power-off retention part.

After the power-off holding counter is reset, the reset state of the counter will be saved. When you use the counter next time, please add an instruction to reset the counter in the ladder diagram, otherwise the counter will still be in the reset state and will not be able to count normally. The timer used for power-off holding will remain in the set state after the count value reaches the set value. You need to reset it first when you use it next time, otherwise it will not be able to count.

(3) V and Z registers

When the operand of a 32-bit instruction is subjected to an indexing operation, V is the upper 16 bits and Z is the lower 16 bits; when the X and Y soft elements are indexed, the V and Z values must be converted into octal before the addition operation is performed.

(4) Power-off retention range

	symbol	Base	Points	Starting address	End address	Latch start	Latch End	Latch setting range
Auxiliary relay	M	10	7680	0	7679	500	1023	0-1023
state	S	10	4096	0	4095	500	999	0-999
Timer	T	10	512	0	511	T246~T255 power-off retention		
counter (16 bits)	C	10	200	0	199	100	199	0-199
counter (32-bit)	C	10	56	200	255	220	255	200-255
Data Register	D	10	8000	0	7999	200	511	0-511
Extension registers	R	10	32768	0	32767	R0~R32767 power off retention		

Of which:

• When the latch setting range is not set : the default is no retention upon power failure.

• Latch start ~ latch end: You can select, and select to make the range that cannot be retained into the range that can be retained.

• Latch end ~ end address: default power-off retention.

• X and Y are not saved by default, T only keeps T246~T255 by default, and all R registers are kept.

(5) Bit designation Dx.y of word device

The bit of a word soft element can be used as bit data. When specifying the bit of a word soft element, please use the word soft element number and bit number (hexadecimal number) to set it, where x = 0 ~ 7999, y = 0 ~ F. When used as a bit soft element , Dx.y does not support the indexing operation.

(6) Combination bit element

Bit soft elements can also process numerical values by combining them. In this case, they are represented by a combination of the number of the bit Kn and the number of the starting soft element, such as KnX, KnY, KnS, and KnM. K1 to K4 can be represented as 16-bit data, and K1 to K8 can be represented as 32-bit data. Each increase of K by 1 corresponds to 4 bit elements. When the number of bits does not match the number of bits of 16-bit or 32-bit data, the high bit is discarded. Combination bit elements support addressing operations, such as MOV K2M0V0 D0. When V0 = 2, it means that the data of M2 to M9 is combined into a 16-bit number and moved to D0.

1.2 Raynen X3 special memory configuration

1.2.1 Special auxiliary register SM8 (M8)

Register Number	Function	illustrate
SM8000	RUN monitoring a contact: always ON during RUN	
SM8001	RUN monitoring a contact: always OFF during RUN	
SM8002	Initial pulse a contact: ON only in the first scan cycle	
SM8003	Initial pulse b contact: OFF only in the first scan cycle	
SM8004	Error flag	
SM8005	Low battery voltage sign	
SM8011	10ms clock	
SM8012	100ms clock	
SM8013	1s clock	

Register Number	Function	illustrate
SM8014	1min clock	
SM8015	Stop timing and preset	Real-time clock related
SM8016	The display after the time readout is stopped	
SM8017	±30 seconds correction	
SM8018	Installation detected (always ON)	
SM8019	Real Time Clock (RTC) Error	
SM8020	Zero flag	Set when the result of addition or subtraction is 0
SM8021	Borrow flag	Set when the result of a subtraction operation exceeds the maximum negative value
SM8022	Carry flag	Set when a carry occurs in the addition operation or an overflow occurs in the shift operation.
SM8024	Specify BMOV direction	
SM8029	Command action end flag	Generally used for instructions that require multiple cycles to complete the action
SM8030	Battery LED off indication	After driving SM8030, the LED on the PLC panel does not light up even if the battery voltage is low
SM8031	Clear all non-volatile memory	Special SM8, SD8 are not cleared
SM8032	Keep all memory cleared	
SM8033	Memory hold stop	
SM8034	Disable all output	
SM8035	Forced RUN instruction	
SM8036	Forced RUN instruction	
SM8037	Forced STOP instruction	
SM8039	Constant scan mode	
SM8046	STL status action (processed when executing END instruction)	When SM8047 is turned on, any one of S0 ~ S899, S1000 ~ S4095 is turned on.
SM8047	STL monitoring is effective (processed when the END instruction is executed)	After SM8047 is turned on, SD8040~SD8047 are effective
SM8048	Signal alarm action	When SM8049 is on, any one of S900~S999 is on.
SM8049	Signal alarm is effective	After SM8049 is turned on, the action of SD8049 is effective

Register Number	Function	illustrate
SM8050	Disable input interrupt I00□	run→stop clear
SM8051	Disable input interrupt I10□	
SM8052	Disable input interrupt I20□	
SM8053	Disable input interrupt I30□	
SM8054	Disable input interrupt I40□	
SM8055	Disable input interrupt I50□	
SM8056	Disable timer interrupt I6□□	
SM8057	Disable timer interrupt I7□□	
SM8058	Disable timer interrupt I8□□	
SM8059	Counter interrupt disable I010~I060	
SM8061	PLC hardware error	
SM8062	[COM0] Serial communication error latch	
SM8063	[COM1] Serial communication error latch	
SM8064	Parameter error	
SM8065	Syntax Error	
SM8066	Loop Error	
SM8067	Operation error	
SM8068	Operation error latch	
SM8080	MODBUS TCP IP1 communication timeout flag	
SM8081	MODBUS TCP IP2 communication timeout flag	
SM8082	MODBUS TCP IP3 communication timeout flag	
SM8083	MODBUS TCP IP4 communication timeout flag	
SM8084	Communication timeout flag for MODBUS TCP IP5	
SM8085	Communication timeout flag for MODBUS TCP IP6	
SM8086	Communication timeout flag for MODBUS TCP IP7	
SM8087	Communication timeout flag for MODBUS TCP IP8	
SM8090	BKCMP instruction block comparison signal	
SM8091	BINDA instruction outputs character number switching signal	
SM8099	High-speed ring counter (0.1ms unit, 16 bits) operation	
SM8120	[COM0] RS2 send wait flag	

Register Number	Function	illustrate
SM8121	[COM0] MODBUS communicating; [COM0] RS2 sending request	
SM8122	[COM0] MODBUS communication error ; [COM0] RS2 receiving end flag	
SM8123	[COM0] MODBUS communication error latch	
SM8128	[COM0] MODBUS master station retransmission event judgment flag	
SM8129	[COM0] Timeout judgment flag	
SM8130	HSZ (FNC55) instruction table comparison mode	
SM8131	HSZ instruction table comparison execution end flag	
SM8132	HSZ, PLSY instruction frequency control mode	
SM8133	HSZ, PLSY instruction frequency control execution end flag	
SM8160	XCH's SWAP function	The DXCH exchange function does not work. The double word should exchange the high and low bits of each word soft element. However, after the actual execution of the instruction, the high word becomes 0 and only the low word is exchanged.
SM8165	SORT2 (FNC 149) instruction sorts in descending order	run→stop clear
SM8168	SMOV (FNC 13) function to process HEX data	run→stop clear
SM8170	Input X000 pulse capture	run→stop clear
SM8171	Input X001 pulse capture	
SM8172	Input X002 pulse capture	
SM8173	Input X003 pulse capture	
SM8174	Input X004 pulse capture	
SM8175	Input X005 pulse capture	
SM8179	BD board enable configuration	
SM8180	BD board enable command	
SM8182	BD board 2 enable configuration	X3 all-in-one machine can use 2 BD boards
SM8183	BD board 2 enable command	
SM8195~ SM8199	4-fold frequency switching of C251~C255	When set to ON, it switches to 4 times the frequency
SM8200~ SM8234	C200~C234 counting mode control bit ON: Down counting mode OFF: Up counting mode	Run→stop does not clear, power off clears

Register Number	Function	illustrate
SM8235~ SM8245	High-speed C235~C245 counting mode control bit ON: Down counting mode OFF: Up counting mode	Run→stop does not clear, power off clears
SM8246~ SM8255	High-speed counter increase and decrease count status bit ON: Down counting state OFF: Up counting state	Run→stop does not clear, power off clears
SM8259	Module offline configuration clear flag	Set ON to clear, it will reset after clearing
SM8261~ SM8280	Analog special adapter	
SM8304	Zero position, when the result of multiplication and division is 0, it turns ON	
SM8306	Carry, when the result of division operation overflows, it turns ON	
SM8329	The instruction execution ended abnormally	
SM8338	PLSY instruction acceleration and deceleration flag	
SM8336	DVIT instruction interrupt input specified function is valid	
SM8340	[Y000] Pulse output monitoring (ON: BUSY/OFF: READY)	
SM8341	[Y000] Clear signal output function is enabled	
SM8342	[Y000] Origin return start direction specification	1 is forward, 0 is reverse
SM8343	[Y000] Forward limit	
SM8344	[Y000] Reverse limit	
SM8345	[Y000] Proximity DOG signal logic inversion	
SM8346	[Y000] Zero signal logic inversion	
SM8347	[Y000] Interrupt signal logic inversion	
SM8348	[Y000] Positioning command driving	
SM8349	[Y000] Pulse output stop command	
SM8460	DVIT command [Y000] User interrupt input command	
SM8350	[Y001] Pulse output monitoring (ON: BUSY/OFF: READY)	
SM8351	[Y001] Clear signal output function is enabled	
SM8352	[Y001] Origin return start direction specification	1 is forward, 0 is reverse
SM8353	[Y001] Forward limit	
SM8354	[Y001] Reverse limit	

Register Number	Function	illustrate
SM8355	[Y001] Proximity DOG signal logic inversion	
SM8356	[Y001] Zero signal logic inversion	
SM8357	[Y001] Interrupt signal logic inversion	
SM8358	[Y001] Positioning command driving	
SM8359	[Y001] Pulse output stop command	
SM8461	DVIT instruction [Y001] User interrupt input instruction	
SM8360	[Y002] Pulse output monitoring (ON: BUSY/OFF: READY)	
SM8361	[Y002] Clear signal output function is enabled	
SM8362	[Y002] Origin return start direction specification	1 is forward, 0 is reverse
SM8363	[Y002] Forward limit	
SM8364	[Y002] Reverse limit	
SM8365	[Y002] Proximity DOG signal logic inversion	
SM8366	[Y002] Zero signal logic inversion	
SM8367	[Y002] Interrupt signal logic inversion	
SM8368	[Y002] Positioning command driving	
SM8369	[Y002] Pulse output stop command	
SM8462	DVIT instruction [Y002] User interrupt input instruction	
SM8370	[Y003] Pulse output monitoring (ON: BUSY/OFF: READY)	
SM8371	[Y003] Clear signal output function is enabled	
SM8372	[Y003] Origin return start direction specification	1 is forward, 0 is reverse
SM8373	[Y003] Forward limit	
SM8374	[Y003] Reverse limit	
SM8375	[Y003] Proximity DOG signal logic inversion	
SM8376	[Y003] Zero signal logic inversion	
SM8377	[Y003] Interrupt signal logic inversion	
SM8378	[Y003] Positioning command driving	
SM8379	[Y003] Pulse output stop command	
SM8463	DVIT instruction [Y003] User interrupt input instruction	
SM8380 (60 points)	[Y004] Pulse output monitoring (ON: BUSY/OFF: READY)	
SM8381 (60 points)	[Y004] Clear signal output function is enabled	

Register Number	Function	illustrate
SM8382 (60 points)	[Y004] Origin return start direction specification	1 is forward, 0 is reverse
SM8383 (60 points)	[Y004] Forward limit	
SM8384 (60 points)	[Y004] Reverse limit	
SM8385 (60 points)	[Y004] Proximity DOG signal logic inversion	
SM8386 (60 points)	[Y004] Zero signal logic inversion	
SM8387 (60 points)	[Y004] Interrupt signal logic inversion	
SM8388 (60 points)	[Y004] Positioning command is being driven	
SM8389 (60 points)	[Y004] Pulse output stop command	
SM8464 (60 points)	DVIT instruction [Y004] User interrupt input instruction	
SM8390 (60 points)	[Y005] Pulse output monitoring (ON: BUSY/OFF: READY)	
SM8391 (60 points)	[Y005] Clear signal output function is enabled	
SM8392 (60 points)	[Y005] Origin return start direction specification	1 is forward, 0 is reverse
SM8393 (60 points)	[Y005] Forward limit	
SM8394 (60 points)	[Y005] Reverse limit	
SM8395 (60 points)	[Y005] Proximity DOG signal logic inversion	
SM8396 (60 points)	[Y005] Zero signal logic inversion	
SM8397 (60 points)	[Y005] Interrupt signal logic inversion	
SM8398 (60 points)	[Y005] Positioning command is being driven	
SM8399 (60 points)	[Y005] Pulse output stop command	
SM8465 (60 points)	DVIT instruction [Y005] User interrupt input instruction	
SM8401	[COM1] MODBUS communication in progress ; [COM1] RS2 send waiting flag	

Register Number	Function	illustrate
SM8402	[COM1] MODBUS communication error ; [COM1] RS2 sends request	
SM8403	[COM1] MODBUS communication error latch ; [COM1] RS2 receiving end flag	
SM8408	[COM1] MODBUS master station retransmission event judgment flag	
SM8409	[COM1] Timeout judgment flag	
SM8421	[COM2] MODBUS communication in progress ; [COM2] RS2 send waiting flag	
SM8422	[COM2] MODBUS communication error ; [COM2] RS2 sends request	
SM8423	[COM2] MODBUS communication error latch ; [COM2] RS2 receiving end flag	
SM8428	[COM2] MODBUS master station retransmission event judgment flag	
SM8429	[COM2] Timeout judgment flag	
SM8438	[COM2] Serial communication error latch	
SM8466	1ms ring counter (32-bit SD8097, SD8096) action	
SM8480	X0 interrupt terminal redirection enable flag	
SM8481	X1 interrupt terminal redirection enable flag	
SM8482	X2 interrupt terminal redirection enable flag	
SM8483	X3 interrupt terminal redirection enable flag	
SM8484	X4 interrupt terminal redirection enable flag	
SM8485	X5 interrupt terminal redirection enable flag	
SM8486	X6 interrupt terminal redirection enable flag	
SM8487	X7 interrupt terminal redirection enable flag	
SM8499	Contact for setting delay time	Used with SD8499 (delay time setting)
SM8500	The slave is not online during MODBUS TCP communication.	
SM8501	The network cable is not connected or disconnected during communication	
SM8502	Ethernet function control bit	
SM8503	MODBUS TCP function control bits	
SM8511	MODBUS RTU configuration in the host computer form takes effect	

1.2.2 Special Data Register SD8 (D8)

Register Number	Function	illustrate
SD8000	Watchdog timer (default 200), unit ms	Default 200
SD8001	PLC type and system version (compatible with Mitsubishi, not Raynen version)	Default 24320
SD8002	Memory capacity, fixed at 8 (compatible with Mitsubishi)	Default 8
SD8004	Error M code	
SD8005	Battery voltage, unit 0.1v	Default 33
SD8006	Detects the battery voltage low level, unit 0.1v	Default 27
SD8010	Current scan time (0.1ms)	
SD8011	Minimum scan time (0.1ms)	
SD8012	Maximum scan time (0.1ms)	
SD8013	Real time clock (seconds)	
SD8014	Real time clock (minutes)	
SD8015	Real-time clock (time)	
SD8016	Real-time clock (day)	
SD8017	Real time clock (month)	
SD8018	Real time clock (year)	
SD8019	Real time clock (week)	
SD8020	Input filter value	Default 10
SD8028	Contents of the Z0 register	
SD8029	Contents of the V0 register	
SD8030~ SD8038	Module version number usage	
SD8039	Constant scan time setting (ms)	
SD8040	ON state number 1	The smallest number of the active state in states S0~S899, S1000~S4095 is saved to SD8040. The next ON state is saved to the next register, with a maximum of eight points, and the default is -1.
SD8041	ON state number 2	
SD8042	ON state number 3	
SD8043	ON state number 4	
SD8044	ON state number 5	
SD8045	ON state number 6	
SD8046	ON state number 7	
SD8047	ON state number 8	
SD8049	Minimum number in ON state (when SM8049 is valid)	When SM8049 is ON, the minimum number of the signal alarm relays S900~S999 that are ON is saved. The default is -1.

Register Number	Function	illustrate
SD8061	Error code number of PLC hardware error	
SD8062	[COM0] Error code number	
SD8063	[COM1] Error code number	
SD8064	Error code number for parameter errors	
SD8065	The error code number for the syntax error	
SD8066	Error code number of ladder error	
SD8067	Error code number of operation error	
SD8068	Latch of the step number where the operation error occurred	
SD8069	Error step number for syntax, circuit, and operation errors	
SD8093	[COM0] Inter-request delay	
SD8094	[COM0] Set the number of retransmissions	
SD8095	[COM0] slave station number	
SD8 096	0~2147483647 (1ms unit 32 bits) incrementing ring counter	
SD8 097		
SD8099	0~32767 (0.1ms unit 16 bits) incrementing ring counter	
SD8101	PLC type and system version (compatible with Mitsubishi, not Raynen version)	Default 16320
SD8102	Memory capacity 2N is fixed to 16 (compatible with Mitsubishi)	Default 64
SD8110	XPLC software version number (Raynen)	The default value is 2xxx (2 indicates X3 series, xxx indicates the firmware version)
SD8111	XPLC Points	The high eight bits are the X number, and the low eight bits are the Y number.
SD8120	[COM0] Communication format setting	Communication protocol settings for COM port, please refer to chapter 3.4 for details
SD8121	[COM0] Protocol	
SD8122	[COM0] MODBUS communication error register; [COM0] RS2 remaining points of sent data	
SD8123	[COM0] MODBUS communication error latch register; [COM0] RS2 receiving point monitoring	
SD8124	[COM0] RS2 header 1, 2	
SD8125	[COM0] RS2 header 3, 4	
SD8126	[COM0] RS2 tail 1, 2	
SD8127	[COM0] RS2 tail 3, 4	
SD8128	[COM0] displays the current retransmission times of the master station	

Register Number	Function	illustrate
SD8129	[COM0] Set the response timeout	
SD8130	HSZ comparison table counter	
SD8131	HSZ, PLSY frequency table counter	
SD8132	HSZ, PLSY frequency control mode frequency low bit	
SD8133	HSZ, PLSY frequency control mode frequency high	
SD8134	HSZ, PLSY frequency control mode target pulse number low bit	
SD8135	HSZ, PLSY frequency control mode target pulse number high	
SD8140	PLSR, PLSY output to Y0 pulse number low	
SD8141	PLSR, PLSY output to Y0 pulse number high	
SD8142	PLSR, PLSY output to Y1 pulse number low	
SD8143	PLSR, PLSY output to Y1 pulse number high	
SD8144	PLSR, PLSY output to Y2 pulse number low	
SD8145	PLSR, PLSY output to Y2 pulse number high	
SD8146	PLSR, PLSY output to Y3 pulse number low	
SD8147	PLSR, PLSY output to Y3 pulse number high	
SD8148 (60 points)	PLSR, PLSY output to Y4 pulse number low	
SD8149 (60 points)	PLSR, PLSY output to Y4 pulse number high	
SD8150 (60 points)	PLSR, PLSY output to Y5 pulse number low	
SD8151 (60 points)	PLSR, PLSY output to Y5 pulse number high	
SD8182	Contents of the Z1 register	
SD8183	Contents of the V1 register	
SD8184	Contents of the Z2 register	
SD8185	Contents of the V2 register	
SD8186	Contents of the Z3 register	
SD8187	Contents of the V3 register	
SD8188	Contents of the Z4 register	
SD8189	Contents of the V4 register	
SD8190	Contents of the Z5 register	
SD8191	Contents of the V5 register	

Register Number	Function	illustrate
SD8192	Contents of the Z6 register	
SD8193	Contents of register V6	
SD8194	Contents of the Z7 register	
SD8195	Contents of the V7 register	
SD8219~ SD8238	SD8 register used by BD board	
SD8260~ SD8267	SD8 register used by analog modules	
SD8312	Latch of the step number where the operation error occurred (32-bit low order)	
SD8313	Latch of the step number where the operation error occurred (32-bit high order)	
SD8314	Error step number of SM8065~SM8067 (32-bit low order)	
SD8315	Error step number of SM8065~SM8067 (32-bit high)	
SD8336	DVIT interrupt input specification (32bit low)	
SD8337 (60 points)	DVIT interrupt input specification (32bit high)	
SD8340	[Y000] Current register value (32-bit low order)	
SD8341	[Y000] Current register value (32-bit high)	
SD8342	[Y000] Base speed	Default 0 (HZ)
SD8343	[Y000] Maximum speed (32bit low)	Default 200000 (HZ)
SD8344	[Y000] Maximum speed (32-bit high)	
SD8345	[Y000]Crawling speed	Default 1000 (HZ)
SD8346	[Y000] Origin return speed (32bit low)	Default 50000 (HZ)
SD8347	[Y000] Origin return speed (32-bit high)	
SD8348	[Y000] Acceleration time	Default 100 (ms)
SD8349	[Y000] Deceleration time	Default 100 (ms)
SD8350	[Y001] Current register value (32-bit low order)	
SD8351	[Y001] Current register value (32-bit high)	
SD8352	[Y001] Base speed	Default 0 (HZ)
SD8353	[Y001] Maximum speed (32bit low)	Default 200000 (HZ)
SD8354	[Y001] Maximum speed (32-bit high)	
SD8355	[Y001] Crawling speed	Default 1000 (HZ)
SD8356	[Y001] Origin return speed (32bit low)	Default 50000 (HZ)
SD8357	[Y001] Origin return speed (32-bit high)	
SD8358	[Y001] Acceleration time	Default 100 (ms)

Register Number	Function	illustrate
SD8359	[Y001] Deceleration time	Default 100 (ms)
SD8360	[Y002] Current register value (32-bit low bit)	
SD8361	[Y002] Current register value (32-bit high)	
SD8362	[Y002] Base speed	Default 0 (HZ)
SD8363	[Y002] Maximum speed (32bit low)	Default 200000 (HZ)
SD8364	[Y002] Maximum speed (32-bit high)	
SD8365	[Y002] Crawling speed	Default 1000 (HZ)
SD8366	[Y002] Origin return speed (32-bit low)	Default 50000 (HZ)
SD8367	[Y002] Origin return speed (32-bit high)	
SD8368	[Y002] Acceleration time	Default 100 (ms)
SD8369	[Y002] Deceleration time	Default 100 (ms)
SD8370	[Y003] Current register value (32-bit low order)	
SD8371	[Y003] Current register value (32-bit high)	
SD8372	[Y003] Base speed	Default 0 (HZ)
SD8373	[Y003] Maximum speed (32bit low)	Default 200000 (HZ)
SD8374	[Y003] Maximum speed (32bit high)	
SD8375	[Y003] Crawling speed	Default 1000 (HZ)
SD8376	[Y003] Origin return speed (32bit low)	Default 50000 (HZ)
SD8377	[Y003] Origin return speed (32-bit high)	
SD8378	[Y003] Acceleration time	Default 100 (ms)
SD8379	[Y003] Deceleration time	Default 100 (ms)
SD8380 (60 points)	[Y004] Current register value (32-bit low order)	
SD8381 (60 points)	[Y004] Current register value (32-bit high)	
SD8382 (60 points)	[Y004] Base speed	Default 0 (HZ)
SD8383 (60 points)	[Y004] Maximum speed (32bit low)	Default 200000 (HZ)
SD8384 (60 points)	[Y004] Maximum speed (32-bit high)	
SD8385 (60 points)	[Y004] Crawling speed	Default 1000 (HZ)
SD8386 (60 points)	[Y004] Origin return speed (32-bit low)	Default 50000 (HZ)
SD8387 (60 points)	[Y004] Origin return speed (32-bit high)	
SD8388 (60 points)	[Y004] Acceleration time	Default 100 (ms)

Register Number	Function	illustrate
SD8389 (60 points)	[Y004] Deceleration time	Default 100 (ms)
SD8390 (60 points)	[Y005] Current register value (32-bit low order)	
SD8391 (60 points)	[Y005] Current register value (32-bit high)	
SD8392 (60 points)	[Y005] Base speed	Default 0 (HZ)
SD8393 (60 points)	[Y005] Maximum speed (32bit low)	Default 200000 (HZ)
SD8394 (60 points)	[Y005] Maximum speed (32-bit high)	
SD8395 (60 points)	[Y005]Crawling speed	Default 1000 (HZ)
SD8396 (60 points)	[Y005] Origin return speed (32bit low)	Default 50000 (HZ)
SD8397 (60 points)	[Y005] Origin return speed (32-bit high)	
SD8 398 (60 points)	[Y005] Acceleration time	Default 100 (ms)
SD8 399 (60 points)	[Y005] Deceleration time	Default 100 (ms)
SD8464	FNC.150.156 instruction Y000 specifies the clear signal soft element	Default 6 (Y6)
SD8465	FNC.150.156 instruction Y001 specifies the clear signal soft element	Default 7 (Y7)
SD8466	FNC.150.156 instruction Y002 specifies the clear signal soft element	Default 8 (Y10)
SD8467	FNC.150.156 instruction Y003 specifies the clear signal soft element	Default 9 (Y11)
SD8468 (60 points)	FNC.150.156 instruction Y004 specifies the clear signal soft element	Default 10 (Y12)
SD8469 (60 points)	FNC.150.156 instruction Y005 specifies the clear signal soft element	Default 11 (Y13)
SD8400	[COM1] Communication format setting	Communication protocol settings for COM port, please refer to chapter 3.4 for details
SD8401	[COM1] Protocol	
SD8402	[COM1] MODBUS communication error register; [COM1] RS2 remaining points of sent data	
SD8403	[COM1] MODBUS communication error latch register; [COM1] RS2 receiving point monitoring	
SD8405	[COM0] RS2 receive and check (receive data)	
SD8408	[COM1] displays the current retransmission	

Register Number	Function	illustrate
	times of the master station	
SD8409	[COM1] Set the response timeout	
SD8410	[COM1] RS2 header 1, 2	
SD8411	[COM1] Delay between MODBUS requests ; [COM1] RS2 headers 3, 4	
SD8412	[COM1] MODBUS sets the number of retransmissions ; [COM1] RS2 end message 1, 2	
SD8413	[COM1] RS2 tail 3, 4	
SD8414	[COM1] MODBUS slave station number ; [COM1] RS2 receiving and verification (receiving data)	
SD8415	[COM1] RS2 receive and check (calculation result)	
SD8416	[COM1] RS2 send and check	
SD8418	[COM0] RS2 receive and check (calculation result)	
SD8419	[COM0] RS2 send and check	
SD8420	[COM2] Communication format setting	
SD8421	[COM2] Protocol	
SD8422	[COM2] MODBUS communication error register; [COM2] RS2 remaining points of data sent	
SD8423	[COM2] MODBUS communication error latch register; [COM2] RS2 receiving point monitoring	
SD8428	[COM2] Displays the current retransmission times of the master station	
SD8429	[COM2] Set the response timeout	
SD8430	[COM2] RS2 header 1, 2	
SD8431	[COM2] Delay between MODBUS requests ; [COM2] RS2 header 3, 4	
SD8432	[COM2] MODBUS sets the number of retransmissions ; [COM2] RS2 end message 1, 2	
SD8433	[COM2] RS2 tail 3, 4	
SD8434	[COM2] MODBUS slave station number ; [COM2] RS2 receiving and verification (receiving data)	
SD8435	[COM2] RS2 reception and verification (calculation result)	
SD8436	[COM2] RS2 send and check	
SD8438	[COM2] Error code number	

Register Number	Function	illustrate
SD8449	Module-specific error codes	
SD8499	Set the interrupt delay time (in ms)	
SD8500	Displays the number of PLC currently assigned protocol connections	Ethernet dedicated registers, see section 3.6 for specific usage
SD8501	The IP address of the offline slave station (the fourth segment)	
SD8502	Display MODBUS TCP error codes	
SD8503	Set the Ethernet master station receiving timeout (in ms)	
SD8504	Displays the current number of TCP connections	
SD8505	For MAC address burning (not available to users)	
SD8506	Store the slave station IP and error location when receiving timeout	
SD8507	Reconnection interval (in ms)	
SD8508	The first segment of the local IP (for display only)	
SD8509	The second segment of the local IP (for display only)	
SD8510	The third segment of the local IP (for display only)	
SD8511	The fourth segment of the local IP (for display only)	

Special Notes:

(1) Communication dedicated register

Some of the MODBUS and RS2 communication registers are shared . For specific usage, please refer to Chapter 3.4.

(2) The 60-point mark can only be used in a 60-point PLC and is invalid in a PLC with other points.

usage of Mitsubishi special registers is M8 000~M8511, D8 000~D8511, and the usage of special registers on the X3 host computer is SM8 000~SM8511, SD8 000~SD8511. M8 000~M8511, D8 000~D8511 appearing in the document is for compatibility with GX programming usage.

2. Instructions

The instruction sets supported by X3 are listed in the following table.

Basic instructions		
instruction	Function Number	Instruction Description
LD	/	Load
LDI	/	Load Negation
LDP	/	Take the rising edge of the pulse
LDF	/	Take the falling edge of the pulse
AND	/	And
ANI	/	And not
ANDP	/	With pulse rising edge
ANDF	/	With the pulse falling edge
OR	/	Or
ORI	/	Or not
ORP	/	Or pulse rising edge
ORF	/	Or pulse falling edge
MEP	/	Operation result rising edge pulse instruction
MEF	/	Operation result falling edge pulse instruction
ANB	/	Circuit block and
ORB	/	Circuit block or
MPS	/	Store into stack
MRD	/	Storage Read Stack
MPP	/	Storage stack
instruction	Function Number	Instruction Description
INV	/	Negation
OUT	/	Output
SET	/	Set
RST	/	Reset
PLS	/	Rising edge pulse detection
PLF	/	Falling edge pulse detection
MC	/	Master
MCR	/	Master reset
NOP	/	No operation
END	/	End of program
STL	/	Step ladder diagram starts
RET	/	Step ladder diagram ends

Program flow instructions		
instruction	Function Number	illustrate
CJ	00	Conditional Jumps
CALL	01	Subroutine call
SRET	02	Subroutine returns
IRET	03	Interrupt return
EI	04	Enable interrupts
DI	05	Disable interrupts
FEND	06	End of main program
WDT	07	Watchdog Timer
FOR	08	Start of loop range
NEXT	09	End of loop range
Transmit comparison instruction		
instruction	Function Number	illustrate
CMP	10	Compare
ZCP	11	Interval comparison
MOV	12	Teleport
SMOV	13	Shift
CML	14	Negate Teleport
BMOV	15	Batch Transfer
FMOV	16	Multicast
XC	17	Exchange
BCD	18	BCD Conversion
BIN	19	BIN Conversion
Four logical operation instructions		
instruction	Function Number	illustrate
ADD	20	BIN addition
SUB	twenty one	BIN subtraction
MUL	twenty two	BIN multiplication
DIV	twenty three	BIN division
INC	twenty four	BIN plus 1
DEC	25	BIN minus 1
WAND	26	Logical AND
WOR	27	Logical OR
WXOR	28	Logical XOR
NEG	29	Find the two's complement

Rotate Instructions		
instruction	Function Number	illustrate
ROR	30	Circular right shift
ROL	31	Circular left shift
RCR	32	Rotate right with carry
RCL	33	Rotate left with carry
SFTR	34	Bit shift right
SFTL	35	Bit Shift Left
WSFR	36	Word right
WSFL	37	Word left
SFWR	38	Shift Write
SFRD	39	Shift read
Data processing instructions		
instruction	Function Number	illustrate
ZRST	40	Batch reset
DECO	41	Decoding
ENCO	42	coding
SUM	43	Total number of ON bits
BON	44	ON determination
MEAN	45	average value
ANS	46	Signal alarm set
ANR	47	Signal alarm reset
SQR	48	BIN Prescription
FLT	49	BIN integer → binary floating point conversion
High-speed instruction processing		
instruction	Function Number	illustrate
REF	50	Input and output refresh
REFF	51	Filter adjustment
HSCS	53	Compare Set (High-speed Counter)
HSCR	54	Comparison reset (high-speed counter)
HSZ	55	Interval comparison (high-speed counter)
SPD	56	Pulse density
PLSY	57	Pulse output
PWM	58	Pulse Width Modulation
PLSR	59	Pulse output with acceleration and deceleration

Communication instructions		
instruction	Function Number	Instruction Description
RS2	87	Free port communication
ADPRW	276	MODBUS master communication instructions
Floating point instructions		
instruction	Function Number	illustrate
ECMP	110	Binary floating point comparison
EZCP	111	Binary floating point number interval comparison
EMOV	112	Binary floating point data transmission
ESTR	116	Binary floating point number → string conversion
EVAL	117	String → binary floating point number conversion
EBCD	118	Binary floating point → decimal floating point conversion
EBIN	119	Decimal floating point number → binary floating point number conversion
instruction	Function Number	Instruction Description
EADD	120	Binary floating point addition
ESUB	121	Binary floating point subtraction
EMUL	122	Binary floating point multiplication
EDIV	123	Binary floating point division operation
EXP	124	Binary floating point exponentiation
LOGE	125	Binary floating point natural logarithm operation
LOG10	126	Common logarithmic operations for binary floating point numbers
ESQR	127	Binary floating point square root operation
ENEG	128	Binary floating point sign flip
INT	129	Conversion from binary floating point to BIN integer
SIN	130	Binary floating point SIN operation
COS	131	Binary floating point COS operation
TAN	132	Binary floating point TAN operation
ASIN	133	Binary floating point SIN -1 operation
ACOS	134	Binary floating point COS -1 operation
ATAN	135	Binary floating point number TAN -1 operation
RAD	136	Conversion from binary floating point angle to radians
DEG	137	Conversion of binary floating point radians to degrees

Data Processing 2		
instruction	Function Number	Instruction Description
WSUM	140	Calculate the total value of data
WTOB	141	Byte-based data separation
BTOW	142	Byte-based data combination
UNI	143	4-bit combination of 16-bit data
DIS	144	4-bit separation of 16-bit data
SWAP	147	Swap high and low bytes
SORT2	149	Data sorting 2
position		
instruction	Function Number	Instruction Description
DR	150	Origin regression with dog search
DVIT	151	Interrupt positioning
ZRN	156	Return to origin
ZRN2	215	Origin Return 2
PLSV	157	Variable speed pulse output
DRVI	158	Relative position control
DRVA	159	Absolute position control
PLSN	288	Multi-segment positioning
PLSF	289	Follow
Clock Operation		
instruction	Function Number	illustrate
TCMP	160	Clock data comparison
TZP	161	Clock data interval comparison
TADD	162	Clock data addition
TSUB	163	Clock data subtraction operation
HTOS	164	Second conversion of hour, minute and second data
STOH	165	Seconds data [hour, minute, second] conversion
TRD	166	Reading of clock data
TWR	167	Writing clock data
HOUR	169	Long time timing
Gray code		
instruction	Function Number	Instruction Description
GRY	170	Gray code conversion
GBIN	171	Gray code inverse conversion

Data block instructions		
instruction	Function Number	Instruction Description
BK+	192	Addition of data blocks
BK-	193	Subtraction of data blocks
BKCMP=	194	Data Block Comparison
BKCMP>	195	
BKCMP<	196	
BKCMP<>	197	
BKCMP<=	198	
BKCMP>=	199	
String control instructions		
instruction	Function Number	Instruction Description
STR	200	BIN to string conversion
VAL	201	String → BIN conversion
$+	202	String combination
LEN	203	Detect the length of the string
RIGHT	204	Start taking from the right side of the string
LEFT	205	Start from the left side of the string
MIDR	206	Take any part from the string
MIDW	207	Any replacement in a string
INSTR	208	String search
$MOV	209	String transmission
Data processing instructions 3		
instruction	Function Number	Instruction Description
FDEL	210	Deleting data from a table
FINS	211	Inserting data into the data table
POP	212	Read the last entered data [for first-in, last-out control]
SFR	213	16-bit data n bits right shift (with carry)
SFL	214	16-bit data n bits left shift (with carry)
Contact comparison instructions		
instruction	Function Number	illustrate
LD=	224	(S1)=(S2)
LD>	225	(S1)>(S2)
LD<	226	(S1)<(S2)
LD<>	228	(S1)!=(S2)
LD<=	229	(S1)<=(S2)
LD>=	230	(S1)>=(S2)
AND=	232	(S1)=(S2)

instruction	Function Number	Instruction Description
AND>	233	(S1)>(S2)
AND<	234	(S1)<(S2)
AND<>	236	(S1)!=(S2)
AND<=	237	(S1)<=(S2)
AND>=	238	(S1)>=(S2)
OR=	240	(S1)=(S2)
OR>	241	(S1)>(S2)
OR<	242	(S1)<(S2)
OR<>	244	(S1)!=(S2)
OR<=	245	(S1)<=(S2)
OR>=	246	(S1)>=(S2)
Data table processing		
instruction	Function Number	Instruction Description
LIMIT	256	Upper and lower limit control
BAND	257	Dead zone control
ZONE	258	Zone Control
SCL	259	Coordinate setting (coordinate data of different points)
DABIN	260	Conversion from decimal ASCII to BIN
BINDA	261	BIN → Decimal ASCII conversion
SCL2	269	Coordinate setting 2 (X/Y coordinate data)
Interpolation instructions		
instruction	Function Number	Instruction Description
G90G01	281	Linear absolute position interpolation
G91G01	282	Straight line relative position interpolation
G90G02	283	Two-axis circular absolute position interpolation
G91G02	284	Relative position interpolation of two axes along circular arc
G90G03	285	Two-axis reverse arc absolute position interpolation
G91G03	286	Relative position interpolation of two axes in reverse arc
ABSLINE	306	Three-axis linear absolute position interpolation
RELLINE	307	Three-axis linear relative position interpolation
Extended file registers		
instruction	Function Number	Instruction Description
L OADR	290	Read extended file register
S AVER	291	Batch write extended file registers
I NIT	292	Initialization of extended registers
LOGR	293	Login to the extension register
R WE	294	Deleting and writing extended file registers
I NITER	295	Initialization of extended file registers

Convenient instructions		
instruction	Function Number	Instruction Description
ABSD	62	Cam control absolute mode
INCD	63	Cam control relative mode
ALT	66	Alternate output
SORT	69	Data Sorting

Notice:

(1) If programming is done using a Mitsubishi host computer (not allowed in principle, but XPLC is compatible with GX host computers), a shutdown error 6506 will be reported if unimplemented instructions appear in the program.

(2) The +P after the instruction represents edge triggering, and the +D before the instruction represents a 32-bit instruction.

(3) When the instruction operand is 32 bits, the word element (low 16 bits) that it constitutes and an adjacent word element (high 16 bits) are combined to form this 32-bit number. Be sure to distinguish them when using the instructions, and it is best to use an even address as the starting element for 32-bit instructions.

2.1 Basic instructions

When programming in ladder diagram mode, some basic instructions are invisible and can only be viewed by converting them into instruction lists.

Basic instructions are roughly divided into five categories:

1. Contact instruction: It is usually placed next to the bus in the ladder diagram and serves as an action switch for the execution of subsequent instructions according to different triggering methods.

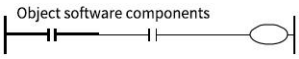
2. Combine instructions: connect ladder diagrams to form multiple execution logics.

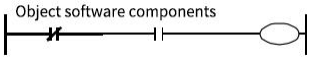
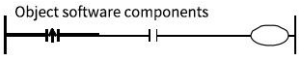
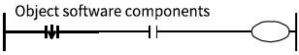
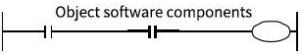
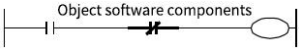
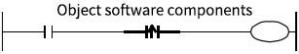
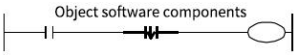
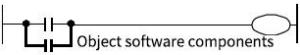
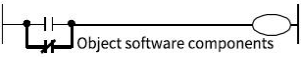
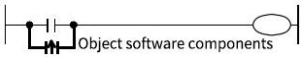
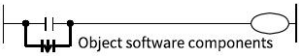
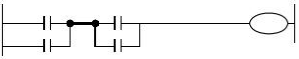
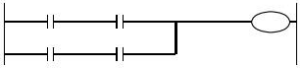
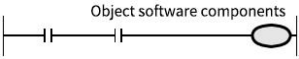
3. Output instruction: Assign initial value to the object soft element according to the output method.

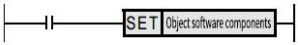
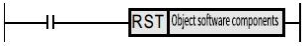
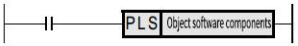
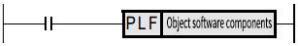
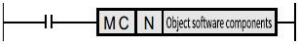
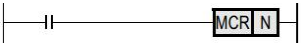
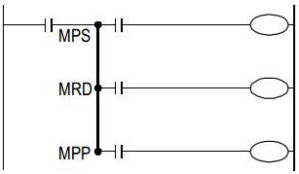
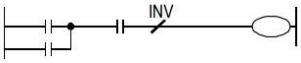
4. Main control instruction: controls the opening and closing of a program block.

5. End instruction: END, placed at the end of the ladder diagram; FEND is used to end the main program. It will perform the same output processing, input processing, and refresh of the watchdog timer as the END instruction, and then return to the program at step 0. The FEND instruction needs to be used when writing subroutines and interrupt programs.

The basic instruction table is as follows (contact a: normally open contact; contact b: normally closed contact):

Mark	Call	Symbol	Function	Target software
LD	Pick		a contact logic operation start	X,Y,M,S,T, C,Dx.y

Mark	Call	Symbol	Function	Target software
LDI	Negation		Logical operation of b contact start	X,Y,M,S,T, C,Dx.y
LDP	Take the rising edge of the pulse		Rising edge detected start	X,Y,M,S,T, C,Dx.y
LDF	Take the falling edge of the pulse		Falling edge detected start	X,Y,M,S,T, C,Dx.y
AND	and		Series a contact	X,Y,M,S,T, C,Dx.y
ANI	With reverse		Series b contact	X,Y,M,S,T, C,Dx.y
ANDP	With pulse rising edge		Rising edge detection series connect	X,Y,M,S,T, C,Dx.y
ANDF	With the pulse falling edge		Falling edge detection series connect	X,Y,M,S,T, C,Dx.y
OR	or		Parallel a contact	X,Y,M,S,T, C,Dx.y
ORI	or reverse		Parallel contact b	X,Y,M,S,T, C,Dx.y
ORP	Or pulse rising edge		Rising edge detection parallel connect	X,Y,M,S,T, C,Dx.y
ORF	Or pulse falling edge		Falling edge detection parallel connect	X,Y,M,S,T, C,Dx.y
ANB	Circuit block and		Series connection of circuit blocks	-
ORB	Circuit block or		Parallel connection of circuit blocks	-
OUT	Output		Coil drive	Y,M,S,T,C, Dx.y

Mark	Call	Symbol	Function	Target software
SET	Set		Action Hold	Y,M,S,Dx.y
RST	Reset		Release the hold action. Clear the current value and register	Y,M,S,T,C, Dx.y D,R,V,Z
PLS	pulse		Rising edge differential output	Y,M
PLF	Falling edge pulse		Falling edge differential output	Y,M
MEP	Operation result rising edge pulse instruction		The operation result is turned on when the rising edge	-
MEF	Operation result falling edge pulse instruction		The falling edge of the remote calculation result turns ON	-
MC	Master		Connect to common contact	Y,M
MCR	Master reset		Disconnect from Public contact	-
MPS	Store into stack		Push onto stack	-
MRD	Storage Read Stack		Reading the stack	-
MPP	Storage stack		Pop the stack	-
INV	Reversal		Inversion of operation results	-
NOP	No operation		No treatment	-
END	Finish		End of program and Input and output processing and return to step 0	-

2.1.1 LD, LDI/take, invert instructions

1. Overview

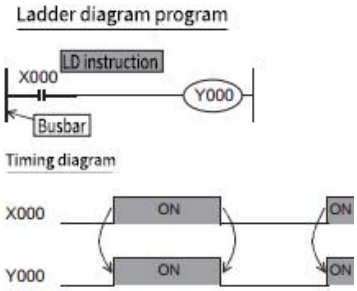
Normally open and normally closed contacts connected to the busbar.

2. Available element types for operands

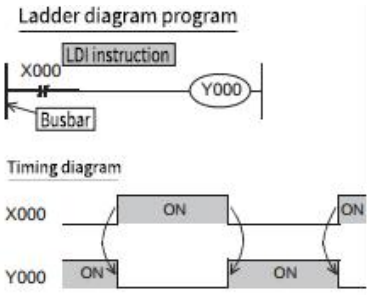
instruction	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
LD	*	*	*	*	*	*	*											*					
LDI	*	*	*	*	*	*	*											*					

3. Function and action description

(1) LD instruction (normally open contact)



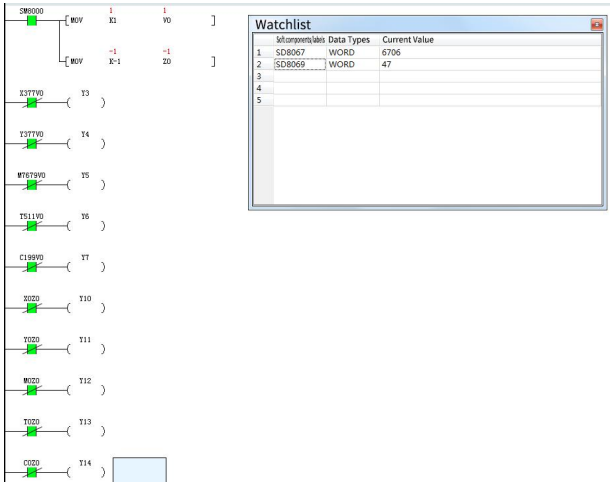
(2) LDI instruction (normally closed contact)



4. Notes

When using contact instructions , pay attention to the situation where the address modification exceeds the boundary. Take LDI as an example. LDI+bit soft element address modification operation. When the bit soft element address modification operation exceeds

the boundary, XPLC monitoring returns 0, but an error is reported when executing LDI, and the inversion operation is not performed. (For errors in instructions, XPLC will not execute the instruction as long as there is an error, that is, output 0)



This ladder diagram shows the execution when the address is out of bounds. Output Y is not lit, while it will be lit under normal circumstances.

2.1.2 OUT/Output instruction

1. Overview

The OUT instruction is an instruction to drive the coil of the output relay (Y), auxiliary relay (M), status (S), timer (T), and counter (C) .

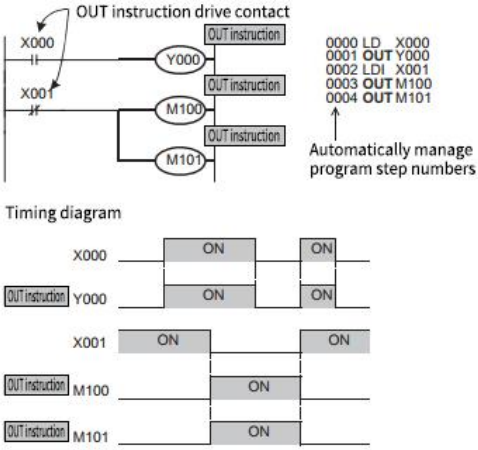
2. Available element types for operands

instruction	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
OUT		*	*	*	*	*	*											*					
Settings														*	*			*	*				

3. Function and action description

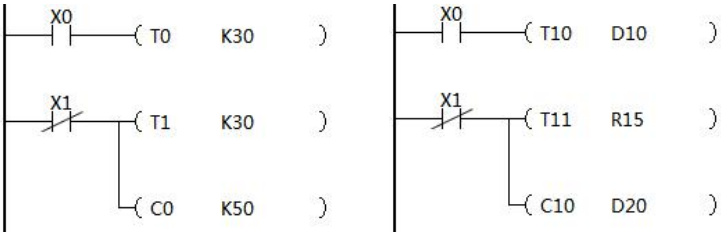
(1) When using bit devices

The soft element programmed with the OUT instruction is turned on/off according to the state of the driving contact.



(2) Using timers and counters

The setting value needs to be added after the OUT instruction for the timer timing coil and the counter counting coil. The setting value can be directly specified using a decimal number (K) or indirectly specified using a data register (D) or an extended register (R).



4. Notes

(1) For the setting range of the timer and counter setting values and the actual timer constants (including setting values), refer to the resource configuration table in Section 1.1.

(2) When the corresponding soft element does not exist after the address change, error 6706 is reported , and the soft element after the address change cannot be the corresponding high-speed counter.

2.1.3 AND, ANI/AND, AND inversion instructions

1. Overview

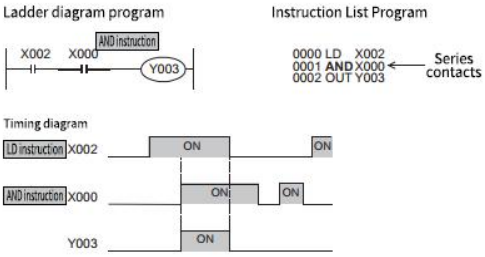
AND and ANI instructions are used to connect contacts in series. There is no limit on the number of contacts in series, and this instruction can be used multiple times in succession.

2. Available element types for operands

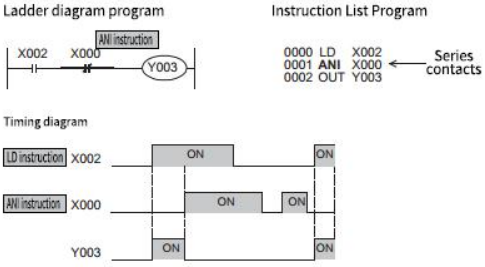
instruction	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
AND	*	*	*	*	*	*	*											*					
ANI	*	*	*	*	*	*	*											*					

3. Function and action description

(1) AND instruction (normally open contacts in series)



(2) ANI instruction (normally closed contacts in series)



4. Notes

Pay attention to the situation where the address modification is out of bounds , refer to the LDI instruction.

2.1.4 OR, ORI/OR, OR inversion instructions

1. Overview

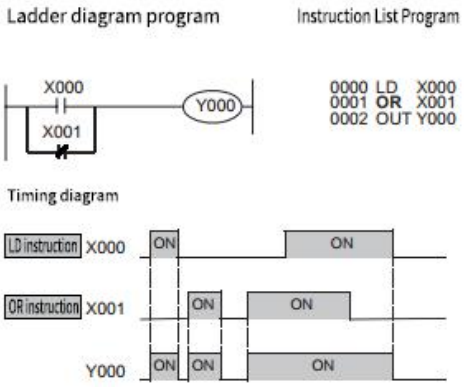
The OR and ORI instructions are used to connect contacts in parallel.

2. Available element types for operands

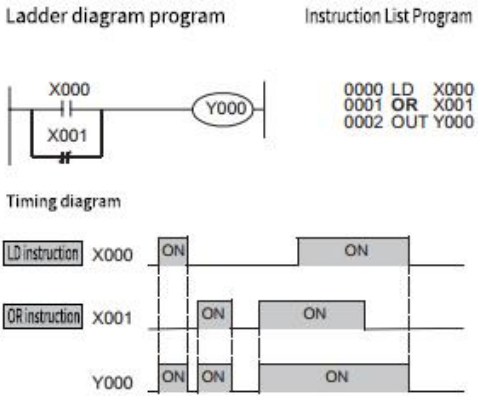
instruction	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
OR	*	*	*	*	*	*	*											*					
ORI	*	*	*	*	*	*	*											*					

3. Function and action description

(1) OR instruction (parallel normally open contacts)



(2) ORI instruction (parallel normally closed contacts)



4. Notes

Pay attention to the situation where the address modification is out of bounds , refer to the LDI instruction.

2.1.5 LDP, LDF, ANDP, ANDF, ORP, ORF /edge-triggered instructions

1. Overview

The LDP , ANDP , and ORP instructions are contact instructions that detect rising edges. They are connected for one operation cycle only when the specified bit soft element has a rising edge (when it changes from OFF to ON).

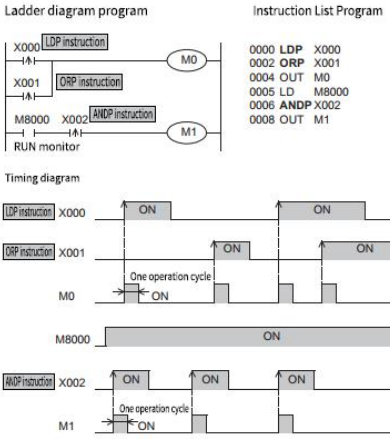
The LDF, ANDF, and ORF instructions are contact instructions that detect falling edges. They are connected for one operation cycle only when the specified bit soft element has a falling edge (when it changes from ON to OFF).

2. Available element types for operands

instruction	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
LDP	*	*	*	*	*	*	*																
LDF	*	*	*	*	*	*	*																
ANDP	*	*	*	*	*	*	*																
ANDF	*	*	*	*	*	*	*																
ORP	*	*	*	*	*	*	*																
ORF	*	*	*	*	*	*	*																

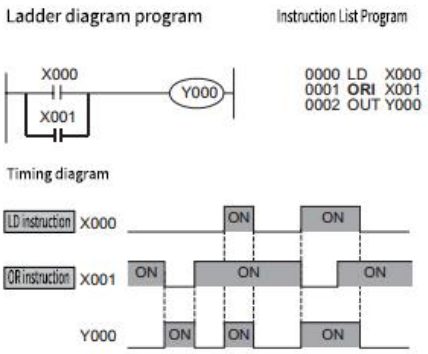
3. Function and action description

(1) LDP, ANDP, ORP instructions (operation starts when a rising edge is detected , series connection, parallel connection)



When X000~X002 changes from OFF to ON, M0 or M1 only remains ON for one operation cycle.

operation starts when a falling edge is detected, series connection, parallel connection)



When X000~X002 changes from ON to OFF, M0 or M1 only remains ON for one operation cycle.

4. Notes

(1) Pay attention to the situation where the address modification exceeds the boundary , refer to the LDI instruction.

(2) Pulse trigger instructions are generally not suitable for interrupt subroutines.

(3) Pulse trigger instructions generally cannot be used continuously in the same line of a ladder diagram .

2.1.6 ORB/Loop Block or Instruction

1. Overview

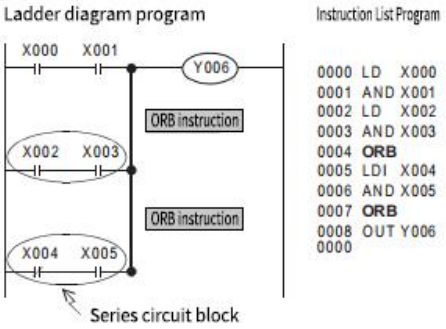
Parallel connection of circuit blocks.

2. Available element types for operands

none.

3. Function and action description

When connecting series circuit blocks in parallel, use LD and LDI instructions at the start of the branch and ORB instructions at the end of the branch. ORB instructions, like ANB instructions described later, are independent instructions without device numbers. When there are multiple parallel circuits, use ORB instructions in each circuit block to connect them.



4. Notes

The ORB instruction can be used indefinitely, and is usually used in conjunction with LD and LDI. However, the LD and LDI instructions can be repeated less than 8 times, so please be careful.

2.1.7 ANB/loop blocks and instructions

1. Overview

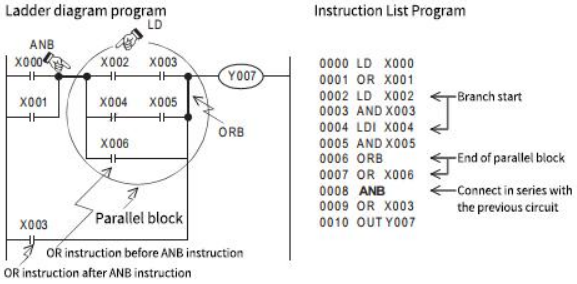
Series connection of circuit blocks.

2. Available element types for operands

None.

3. Function and action description

The starting point of the branch is LD and LDI instructions. After the parallel circuit block is completed, the ANB instruction can be used to connect it in series with the previous circuit. When there are multiple parallel circuits, use the ANB instruction for each circuit block to connect them.



4. Notes

The ANB instruction can be used indefinitely and is usually used in conjunction with LD and LDI. However, the LD and LDI instructions can be repeated less than 8 times, so please be careful.

2.1.8 MPS, MRD, MPP/stack instructions

1. Overview

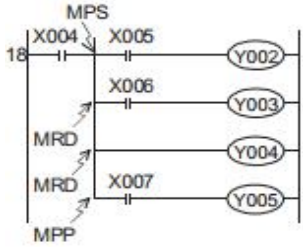
The memory of the stack is used to store the intermediate results of the operation (ON or OFF). The three instructions represent push into the stack, read from the stack, and pop from the stack.

2. Available element types for operands

Slightly.

3. Function and action description

These instructions are convenience instructions for writing multiple branch output loops.



① Use the MPS instruction to store the intermediate result of the operation, and then drive output Y002.

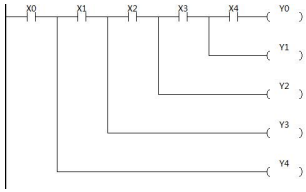
② Use the MRD instruction to read the stored Content, and then drive output Y003. The MRD instruction can be programmed multiple times.

③ Use the MPP instruction to replace the MRD instruction in the final output loop, so that the above stored Content is reset while reading it.

4. Notes

When using stack instructions, pay attention to the execution order of the ladder diagram. Under normal circumstances (no program jump instructions and interrupts occur), the ladder diagram is executed from top to bottom and from left to right.

for example:



When X1 is turned ON, if X0 is set at this time, OUT Y3 should be executed first and OUT Y4 should be executed later.

2.1.9 MC, MCR/master control, master control reset instructions

1. Overview

After executing the MC instruction, the busbar (LD, LDI point) moves to the MC contact point , and the program between MC and MCR begins to execute . Using the MCR instruction, it can be returned to the original busbar position.

2. Available element types for operands

instruction	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
MC		*	*																				

3. Function and action description

After the MC instruction is executed, the bus moves to the back of the MC contact. The drive instructions connected to the bus after the MC contact will only perform various actions when the MC instruction is executed. When the MC instruction is not executed, regardless of the contact state before the drive instruction, it will be executed as OFF (the same action as when the contact is OFF).

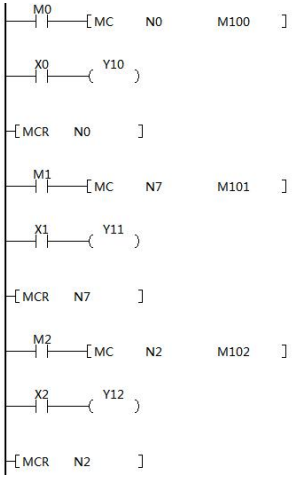
In a circuit using a master control, if there is an instruction that does not require a trigger condition (FOR , NEXT , EI , DI instructions, etc.) in the contact instruction, the instruction is executed regardless of the execution instruction of the MC instruction. In the program example shown below, when the input condition is ON, the instructions from MC to MCR are executed, but when the input condition is OFF, the actions of each drive soft element are as follows.

Soft elements that turn OFF: Timers (except cumulative timers), soft elements driven by OUT instructions

Soft elements that maintain status: Cumulative timers, counters, soft elements driven by SET/RST instructions

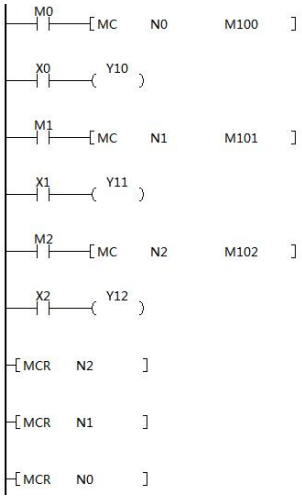
4. Notes

(1) Non-nested use of master control



When there is no nesting, the number of N can be arranged arbitrarily and there is no limit to the number of times it can be used.

(2) Nested use of master control



When nesting is allowed, a maximum of 8 layers of nesting is allowed. When executing the MC instruction, the nesting level N number increases in sequence, N0->N7,

When returning, the MCR instruction is executed to release the nesting level from the highest level, N7->N0.

2.1.10 INV/Invert instruction

1. Overview

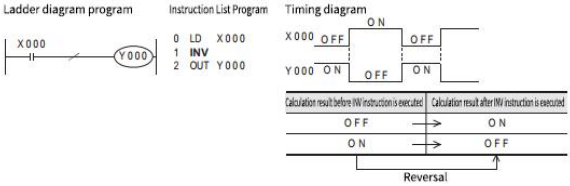
This instruction inverts the calculation result before the INV instruction is executed. It is not necessary to specify the device number.

2. Available element types for operands

None.

3. Function and action description

The result of the operation is inverted.



4. Notes

Note that the scope of the INV instruction is to invert the output result of the previous program block.

2.1.11 MEP, MEF/Operation result pulse instruction

1. Overview

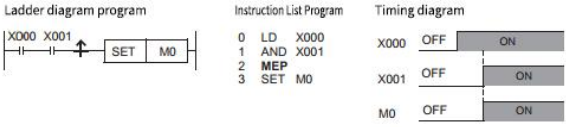
The MEP and MEF instructions are instructions that convert the calculation results into pulses, and do not require the designation of device numbers.

2. Available element types for operands

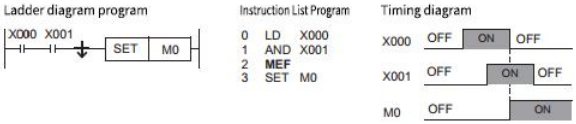
Slightly.

3. Function and action description

(1) MEP instruction (ON at the rising edge of the calculation result)



(2) MEF instruction (ON at the falling edge of the operation result)



4. Notes

Note that the scope of the instruction is based on the calculation results of the previous program block.

2.1.12 PLS, PLF/pulse output instructions

1. Overview

When the PLS instruction is used, the target device operates only in one operation cycle after the drive input is turned on.

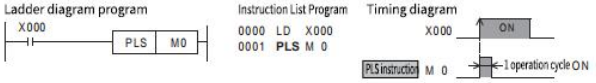
When the PLF instruction is used, the target device operates only in one operation cycle after the drive input is turned off.

2. Available element types for operands

instruction	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
PLS		*	*															*					
PLF		*	*															*					

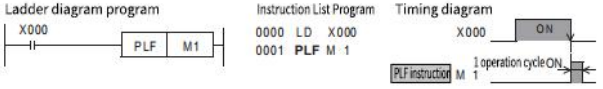
3. Function and action description

(1) PLS instruction (differential output on rising edge)



When X000 changes from OFF to ON, M0 is ON for only one operation cycle.

(2) PLF instruction (differential output at the falling edge)



When X000 changes from ON to OFF, M1 is ON for only one operation cycle.

4.Precautions

The differential output of the same coil can only use one PLS or PLF instruction, otherwise logical confusion may occur.

2.1.13 SET, RST/set, reset instructions

1. Overview

(1) SET bit element

(2) RST reset element

(3) RST clears the current value of the word element

2. Available element types for operands

instruction	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
SET		*	*			*	*											*					
RST		*	*	*	*	*	*					*	*	*	*	*	*	*					

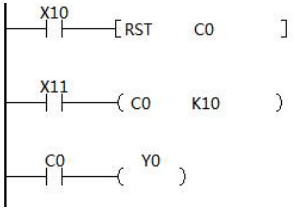
3. Function and action description

(1) SET command

This command drives the coils of the output relay (Y), auxiliary relay (M), status (S), and designated bits of the data register (D).

(2) RST instruction

①Reset ordinary timers and cumulative counters



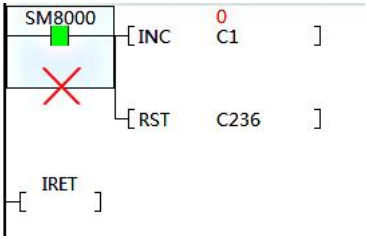
C0 counts the number of times X011 turns OFF→ON. When the count reaches the set value K10, the output contact C0 is activated. After that, even if X011 turns from OFF to ON, the current value of the counter does not change, and the output contact remains in action. In order to clear these and restore the output contact, X010 must be turned ON. After the OUT C 0 instruction, it is necessary to specify the constant K or the data register number for indirect setting. In the case of a power failure holding (holding) counter, the current value and the action status and reset status of the output contact can be maintained even if the power fails.

② Reset common components or word components to 0

4. Notes

When operating special registers, the instructions will be executed, but the results may not be set or reset. For example, RST SM8000, SM8000 is still ON.

When using the RST instruction to operate counter C or high-speed counter, the contact before RST cannot be normally closed, otherwise the counter will always be in the reset state and cannot count, even in the interrupt service function.



2.1.14 NOP/Null Instruction

1. Overview

No-operation instruction .

2. There are no operand component types available

3. Function and action description

When NOP is added between general instructions, the PLC will ignore its existence and continue to run. If NOP is added in the middle of the program, when you need to change or add a program, you only need to change the step number slightly, but the program requires margin. In addition, if you replace the already written instruction with a NOP instruction, the circuit will change, so please be careful.

4. Notes

When programming, NOP will occupy program steps but will not affect program execution.

2.1.15 END/Program end instruction

1. Overview

The instruction indicating the end of the program is at the end of the ladder diagram program and is not displayed.

2. There are no operand component types available

3. Function and action description

(1) Program end processing

The programmable controller repeatedly executes input processing → executes program → output processing. If the END instruction is executed at the end of the program, output processing is performed directly. FEND is the same as END.

(2) Refresh important parameters

When END is executed, the watchdog time and the error register will be refreshed.

4. Notes

When programming in UNISYS, the END instruction is hidden, automatically added to the end of the program and downloaded to the PLC .

2.1.16 Instruction steps and instruction software

The number of instruction steps for basic instructions is as follows .

Software		instruction						
		LD, LDI, AND, ANI, OR, ORI	OUT	SET	RST	PLS, PLF	LDP, LDF, ANDP、ANDF、ORP、ORF	MC
Bit device	X0~X377	1	-	-	-	-	2	-
	Y0~Y377	1	1	1	1	2	2	3
	M0~M1535	1	1	1	1	2	2	3
	M1536~M3583	2	2	2	2	2	2	3
	M3584~M7679	3	3	3	3	3	3	4
	S0~S1023	1	2	2	2	-	2	-
	S1024~S4095	2	2	2	2	-	2	-
	T0~T191, T200~T245	1	3	-	2	-	2	-
	T192~T199, T246~T511	1	3	-	2	-	2	-
	C0~C199	1	3	-	2	-	2	-
	C200~C255	1	5	-	2	-	2	-
	SM8000~SM8255	1	2	2	2	-	2	-
	SM8256~SM8511	2	2	2	2	-	2	-
Bit soft element with indexing	X0~X357	3	-	-	-	-	-	-
	Y0~Y357	3	3	3	3	3	-	-
	M0~M7679	3	3	3	3	3	-	-
	T0~T511	3	4	-	-	-	-	-
	S0~S4095	-	-	-	-	-	-	-
	C0~C199	3	4	-	3	-	-	-
	C200~C255	-	-	-	-	-	-	-
	SM8000~SM8511	3	3	3	3	-	-	-
Word software	D0~D7999, SD8000~SD8511	-	-	-	3	-	-	-
	R0~R32767	-	-	-	3	-	-	-
Word device with indexing	D0~D7999, SD8000~SD8511, SD8000~SD8511	-	-	-	-	-	-	-
Word device bit specification	Dx.y	3	3	3	3	-	3	-

2.2 Program flow instructions

Used to control the logical execution sequence (process) of the ladder diagram.

2.2.1 CJ/Program Jump Instruction

Normal program jump, jump to the corresponding P program location.

1. Instruction format

Instruction number	The number of program steps occupied by the instruction	Instruction Format
FNC0	3	⊣[CJ(P) D]

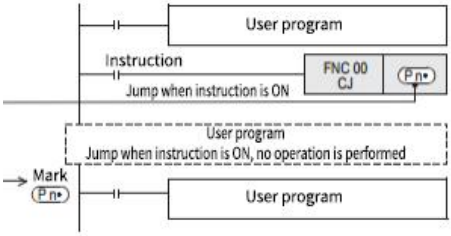
Operand Description

Operands	Content	Type
D	Jump target marker pointer number (P)	Pointer number

The operands are pointer numbers P0~P4095, of which P63 is the step where END is located and does not need to be marked. Pointer numbers can be modified using index registers.

2. Function and action description

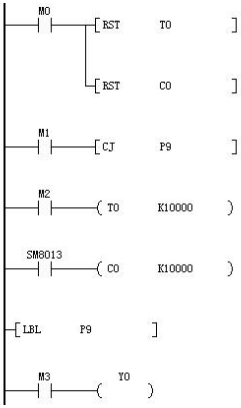
When the command input is ON, the program at the specified label (pointer number) is executed.



3. Program example

Steps of conditional jump instruction CJ:

As shown in the figure, when M1 is connected, the CJ P9 instruction jumps to the instruction labeled P9 and starts execution, skipping part of the program and reducing the scan cycle. If M1 is disconnected, the jump will not be executed, and the program will be executed in the original order.



4. Notes

(1) A label can only appear once in a program, otherwise an error will occur;

(2) During the jump execution, even if the driving conditions of the skipped program change, its coil (or result) still maintains the state before the jump because this program is not executed at all during the jump.

(3) If the timer and counter are already working before the jump starts , they will stop working during the jump execution and continue working after the jump condition is no longer met. However, the working high-speed counters C235~C255 or subroutine dedicated timers will continue to work regardless of whether there is a jump or not.

(4) If the reset (RST) instruction of the cumulative timer and counter is outside the jump area, even if their coils are jumped, their reset is still valid.

(5) Jump instructions can reduce program cycles according to specific needs and skip program segments that do not need to be run.

2.2.2 CALL/Subroutine call instruction

Used for subroutine jump (call).

1. Instruction format

Instruction number	The number of program steps occupied by the instruction	Instruction Format
FNC1	3	CALL(P) D

Operand Description

Operands	Content	Type
D	Jump target marker pointer number (P)	Pointer number

The operand is the pointer number P0~P4095

2. Function and action description

For detailed usage, see Chapter 3.1.

2.2.3 SRET/Subroutine Return Instruction

Use the CALL instruction to call a subroutine, and use SRET to return after the subroutine ends.

1. Instruction format

Instruction number	The number of program steps occupied by the instruction	Instruction Format
FNC2	1	SRET

2. Function and action description

Mainly used for returning after the CALL instruction. For usage, refer to the routine in the CALL instruction.

2.2.4 IRET/Interrupt Return Instruction

After the interrupt program ends, use IRET to return to the next step of the main program call location.

1. Instruction format

Instruction number	The number of program steps occupied by the instruction	Instruction Format
FNC3	1	IRET

=2. Function and action description

If an interrupt (input, timer, counter) occurs during the processing of the main program, it jumps to the interrupt (I) program and then uses the IRET instruction to return to the main program.

For instructions, refer to Chapter 3.2.

2.2.5 EI/Enable Interrupt Instruction

Interrupt enable control, write this instruction where interrupt is needed.

1. Instruction format

Instruction number	The number of program steps occupied by the instruction	Instruction Format
FNC4	1	EI

2. Function and action description

The PLC is usually in the interrupt disabled state. Use the EI (FNC4) instruction to change the PLC to the interrupt enabled state. Please use this instruction when using the input interrupt, timer interrupt, and counter interrupt functions. Refer to Section 3.2 for instructions on the use of the interrupt subroutine.

2.2.6 DI/Disable Interrupt Instruction

For interrupt disable control, write this instruction at a location where an interrupt response is not required so that the interrupt is executed at a location where EI is allowed.

1. Instruction format

Instruction number	The number of program steps occupied by the instruction	Instruction Format
FNC5	1	DI

2. Function and action description

After changing to enable interrupts, you can use the DI instruction to change to disable interrupts again. This instruction can be used to set the area where interrupts are disabled. For instructions, refer to Section 3.2.

2.2.7 FEND/Main program end instruction

This is a dedicated instruction for ending the main program.

1. Instruction format

Instruction number	The number of program steps occupied by the instruction	Instruction Format
FNC6	1	FEND

2. Function and action description

The FEND instruction is the main program end command, and its function is the same as the END instruction, but it is a dedicated end instruction for the main program. The FEND instruction needs to be used when writing subroutines and interrupt programs.

2.2.8 WDT/Watchdog Timer Instructions

Use this command to refresh the watchdog time.

1. Instruction format

Instruction number	The number of program steps occupied by the instruction	Instruction Format
FNC7	1	WDT(P)

2. Function and action description

If the operation cycle of the PLC (0 to END or FEND instruction execution time) exceeds 200ms, the PLC will have a watchdog timer error (operation abnormality detected), and then the ERROR (ERR) LED will light up and then stop. In the case of a long operation cycle like this, inserting a WDT instruction in the middle of the program can avoid such errors.

3. Program example

Set the watchdog timer to 300ms;

Watchdog timer refresh: When the WDT instruction is not programmed, the value of SD8000 becomes valid during END processing .

4. Notes

Ways to avoid watchdog timer timeout:

(1) Control the ladder diagram operation cycle.

(2) Do not use process instructions such as FOR and NEXT too many times in the ladder diagram.

(3) When there is a time-consuming program in the ladder diagram, refresh the watchdog timer in advance or change the watchdog timer time.

2.2.9 FOR/loop range start instruction

Used for repeated execution of programs.

1. Instruction format

Instruction number	The number of program steps occupied by the instruction	Instruction Format
FNC8	3	FOR S

Operand Description

Operands	Content	Type
S	Number of repetitions between FOR ~ NEXT instructions (1 ~ 32767, -32768 ~ 0 is treated as 1)	BIN16 bits

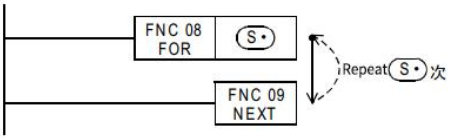
Available operand component types

Operands	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
D								*	*	*	*	*	*	*	*	*	*	*					

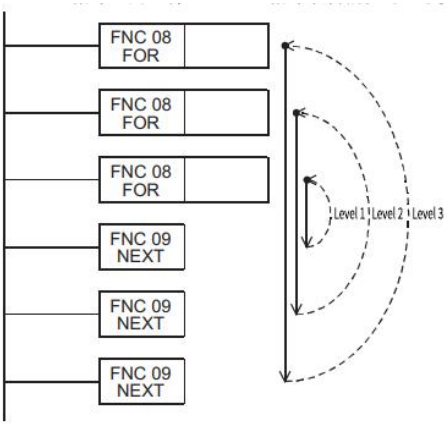
2. Function and action description

The program from the FOR (FNC 08) instruction to the NEXT instruction is repeated the specified number of times.

(1) Single-layer FOR-NEXT



(2) Multi-layer FOR-NEXT nesting



3. Program Example

Refer to the functional action description above.

4. Precautions

(1) 5 levels of FOR-NEXT nesting are allowed. If the number exceeds 5, a shutdown error of 66 20 will be reported .

(2) When using multiple layers of nesting, pay attention to the watchdog timeout, which may cause a timeout. If necessary, refresh the watchdog time.

(3) If the corresponding number of FOR-NEXT does not match, a shutdown error of 66 19 will be reported .

(4) The number after FOR refers to the number of times the loop is executed in one ladder diagram scan cycle.

2.2.10 NEXT/End of loop range instruction

End of the program block for the FOR loop.

1. Instruction format

Instruction number	The number of program steps occupied by the instruction	Instruction Format
FNC9	1	⊣[NEXT]

2. Function and action description

Used in conjunction with the FOR instruction as the end of the loop. For specific usage, refer to the FOR instruction.

2.3 Transmit comparison instruction

Used for basic data transfer, comparison and other operation instructions.

2.3.1 CMP/Compare Instructions

Compares two values and sets an output bit element based on the result.

1. Instruction Format

FNC10	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
16-bit	7	CMP	⊣[(D)CMP(P) S1 S2 D]
		CMPP	
32-bit	13	DCMP	
		DCMPP	

Operand Description

Operands	Content	Type
S1	Comparison value	BIN16/32 bits
S2	Compare Sources	BIN16/32 bits
D	Output comparison result start bit element	Bit

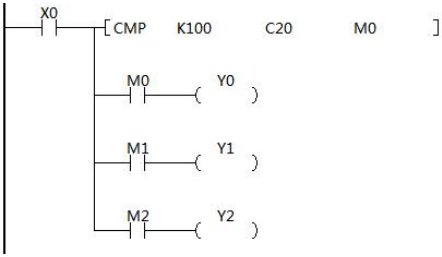
Available operand component types

Operands	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
S1								*	*	*	*	*	*	*	*	*	*	*	*	*			
S2								*	*	*	*	*	*	*	*	*	*	*	*	*			
D		*	*			*	*												*				

2. Function and action description

The comparison value S1 is compared with the Contents of the comparison source S2, and depending on the result (smaller, identical, larger), one of D, D+1, and D+2 is turned on.

3. Program Example



When the current value of C20 is < 100, M0 turns ON and Y0 lights up;

When the current value of C20 = 100, M1 turns ON and Y1 lights up;

When the current value of C20 is > 100, M2 turns ON and Y2 lights up.

4. Precautions

(1) The number of soft elements specified in D is three consecutive. Be careful not to overlap with other soft elements in use.

(2) Even if the command input is OFF and the CMP instruction is not executed, D to D+2 will maintain the state before the command input changes from ON to OFF.

2.3.2 ZCP/Zone Comparison Instructions

A range consisting of two values , the comparison source value is compared with the range, and the output bit device is set according to the result (left side of the range, inside the range, right side of the range) .

1. Instruction format

FNC11	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
16-bit	9	ZCP	[(D)ZCP(P) S1 S2 S D]
		ZCPP	
32-bit	17	ZGar	
		ZGar	

Operand Description

Operands	Content	Type
S1	Comparison value of the left interval	BIN16/32 bits
S2	Comparison value of right interval	BIN16/32 bits
S	Compare Sources	BIN16/32 bits
D	Output comparison result start bit device	Bit

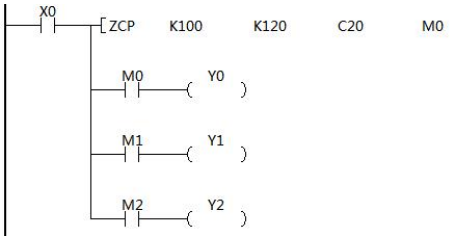
Available operand component types

Operands	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
S1								*	*	*	*	*	*	*	*	*	*	*	*	*			
S2								*	*	*	*	*	*	*	*	*	*	*	*	*			
S								*	*	*	*	*	*	*	*	*	*	*	*	*			
D		*	*			*	*												*				

2. Function and action description

the Contents of the comparison source S with the left interval comparison value S1 and the right interval comparison value S2, and according to the result (left side of the interval, inside the interval, right side of the interval), turn one of D, D+1, or D+2 to ON.

3. Program example



When the current value of C20 is < 100, M0 turns ON and Y0 lights up;

100 ≤ C20 Current ≤ 120, M1 is turned ON and Y1 is lit;

When the current value of C20 is > 120, M2 turns ON and Y2 lights up.

4. Notes

(1) The number of devices specified in D should be three consecutive. Be careful not to overlap with other devices in use.

(2) Even if the command input is OFF and the ZCP command is not executed, D to D+2 will maintain the status before the command input changed from ON to OFF.

(3) When S2 > S1, S2 is treated as S1.

2.3.3 MOV/Move Instructions

To transfer (copy) the Contents of a source device to a destination device.

1. Instruction format

FNC12	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
16-bit	5	MOV	⊣[(D)MOV(P) S D]
		MOVP	
32-bit	9	DMOV	
		DMOVP	

Operand Description

Operands	Content	Type
S	Transmission Source	BIN16/32 bits
D	Teleport Target	BIN16/32 bits

Available operand component types

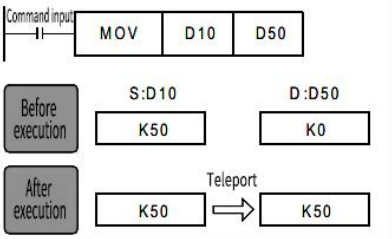
Operands	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
D								*	*	*	*	*	*	*	*	*	*	*	*	*			
S									*	*	*	*	*	*	*	*	*	*					

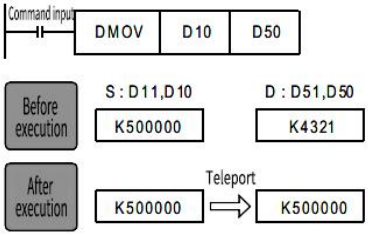
2. Function and action description

Transmit 1 point of data from source S to target D.

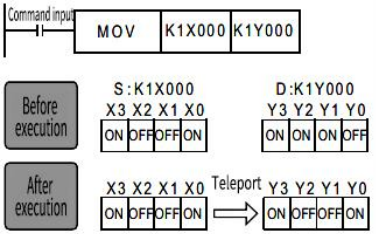
3. Program Example

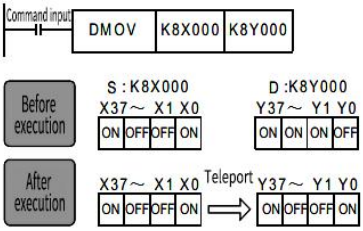
(1) Word transmission software





(2) Transmission components





4. Precautions

If the data to be transmitted is a combined bit element, a 16-bit instruction can transmit at most 16 (a multiple of 4) bit soft elements, and a 32-bit instruction can transmit at most 32 (a multiple of 4) bit soft elements.

2.3.4 SMOV/Bit Move Instructions

This is an instruction to allocate and combine data in bit units (4 bits).

1. Instruction Format

FNC13	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
16-bit	11	SMOV SMOVP	SMOV(P) S m1 m2 D n

Operand Description

Operands	Content	Type
S	Data to be bit-shifted	BIN16 bits
m1	The starting position of the move	BIN16 bits
m2	The number of bits shifted	BIN16 bits
D	Target data after bit shifting	BIN16 bits
n	The starting position of the moving target	BIN16 bits

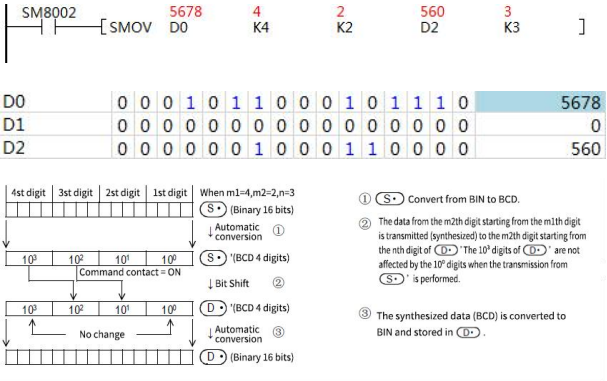
Available operand component types

Operands	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
S								*	*	*	*	*	*	*	*	*	*	*					
m1																			*	*			
M2																			*	*			
D									*	*	*	*	*	*	*	*	*	*					
n																			*	*			

2. Function and action description

The Contents of the transmission source S and the transmission target D are converted (0000~9999) into 4-digit BCD, and the lower m2-digit part starting from the m1-digit is transmitted (synthesized) to the starting point of the n-digit number of the transmission target D, and then converted into BIN and saved in the transmission target D.

3. Program Example



4. Precautions

(1) The original data value range is 0~9999.

(2) The range of m1, m2, and n is 1 to 4.

(3) Turning SM8168 ON, when executing the SMOV instruction, BIN→BCD conversion cannot be performed , and the bit shift is performed in 4-bit units.

2.3.5 CML/Reverse Instruction

This instruction inverts all bits of the source data and then transmits it.

1. Instruction format

FNC14	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
16-bit	5	CML	[(D)CML(P) S D]
		CMLP	
32-bit	9	DCML	
		DCMLP	

Operand Description

Operands	Content	Type
S	The original data to be reversed	BIN16/32 bits
D	Target data after reversal	BIN16/32 bits

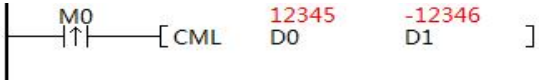
Available operand component types

Operands	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
S								*	*	*	*	*	*	*	*	*	*	*	*	*			
D									*	*	*	*	*	*	*	*	*	*					

2. Function and action description

After reversing each bit of the device specified in S (0→1, 1→0), transfer it to D.

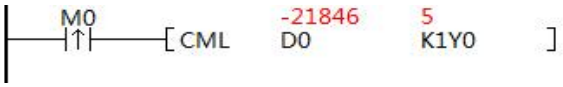
3. Program Example



D0	0	0	1	1	0	0	0	0	0	0	1	1	1	0	0	1	12345
D1	1	1	0	0	1	1	1	1	1	1	0	0	0	1	1	0	-12346

5. Precautions

When a bit combination element is specified, the transmission is reversed according to the actual number of bits, but this number of bits cannot exceed the number of instruction bits and must be a multiple of 4. The following example specifies a number of 4 bits.



soft components	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0	value
D0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	-21846

soft components	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0	value
Y0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	5

2.3.6 BMOV/Batch Move Instruction

Transfer (copy) multiple data of the specified number of points in one batch.

1. Instruction format

FNC15	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
16-bit	7	BMOV BMOVP	BMOV(P) S D n

Operand Description

Operands	Content	Type
S	Transmission Source	BIN16
D	Teleport Target	BIN16
n	Number of teleport points [1≤ n ≤ 512]	BIN16

Available operand component types

Operands	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
S								*	*	*	*	*	*	*	*			*					
D									*	*	*	*	*	*	*			*					
n														*					*	*			

2. Function and action description

The data of n points starting from S are transferred in batches to n points starting from D. If the device number range is exceeded, the data is transferred within the possible range.

3. Program example

4. Notes

(1) Regardless of whether the data of the transmission source is transmitted or not, in order to prevent the data source from being overwritten without being transmitted, the number overlapping method is adopted to automatically transmit in the order of ① to ③ above.

(2) Bidirectional transmission function, when the S M8024 flag is ON, the transmission direction becomes D->S.

(3) For bit devices with bit number specification, S and D should use the same K factor.

2.3.7 FMOV/Multicast Instructions

To transfer the same data to multiple devices.

1. Instruction format

FNC16	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
16-bit	7	FMOV	(D)FMOV(P) S D n
		FMOVP	
32-bit	13	DFMOV	
		DFMOVP	

Operand Description

Operands	Content	Type
S	Transmission Source	BIN16/32 bits
D	The starting word device of the transfer destination	BIN16/32 bits
n	Teleport points [1≦n≦512]	BIN16

Available operand component types

Operands	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
S								*	*	*	*	*	*	*	*	*	*	*	*	*			
D									*	*	*	*	*	*	*			*					
n																			*	*			

2. Function and action description

The Content of S is transferred to the soft elements of n points starting with D. The Content of the soft elements of n points are the same. When the number specified by n exceeds the soft element number range, it is transferred within the possible range.

3. Program example

Before execution

D0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1
D1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	2
D2	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	3
D3	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	4
D4	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	5

After execution

D0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
D1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
D2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
D3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
D4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

4. Notes

When the number specified by n exceeds the device number range, it is transferred within the possible range.

2.3.8 XCH/Exchange Instructions

Exchange data between two devices

1. Instruction format

FNC17	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
16-bit	5	XC	⊣[(D)XCH(P) D1 D2]
		XCHP	
32-bit	9	DXCH	
		DXCHP	

Operand Description

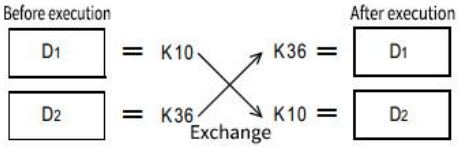
Operands	Content	Type
D1	To exchange data	BIN16/32 bits
D2	To exchange data	BIN16/32 bits

Available operand component types

Operands	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
D1									*	*	*	*	*	*	*	*	*	*					
D2									*	*	*	*	*	*	*	*	*	*					

2. Function and action description

D1 and D2 exchange data with each other.



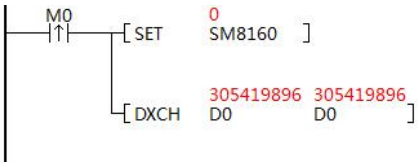
3. Program Example

Slightly.

4. Precautions

(1) When the instruction is executed while SM8160 is ON, the upper 8 bits (bytes) and lower 8 bits (bytes) of the word soft element are exchanged. This action is the same as the SWAP (FNC 147) instruction.

(2) In 32-bit operation, the upper 8 bits (bytes) and lower 8 bits (bytes) of each word soft element are exchanged.



Before execution

soft components	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0	value
D0	0	1	0	1	0	1	1	0	0	1	1	1	1	0	0	0	5678
D1	0	0	0	1	0	0	1	0	0	0	1	1	0	1	0	0	1234

After execution

soft components	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0	value
D0	0	1	1	1	1	0	0	0	0	1	0	1	0	1	1	0	7856
D1	0	0	1	1	0	1	0	0	0	0	0	1	0	0	1	0	3412

(3) S M8160 is ON, if the soft element numbers of D1 and D2 are inconsistent, 6706 operation error will be reported.

2.3.9 BCD/BCD conversion instructions

This command converts BIN (binary number) into BCD (decimal number).

1. Instruction format

FNC18	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
16-bit	5	BCD	⊣[(D)BCD(P) S D]
		BCDP	
32-bit	9	DBCD	
		DBCDP	

Operand Description

Operands	Content	Type
S	Source (binary) data	BIN16/32 bits
D	Target (decimal) data	BIN16/32 bits

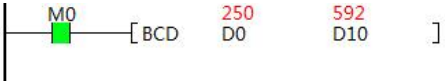
Available operand component types

Operands	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
S								*	*	*	*	*	*	*	*	*	*	*					
D									*	*	*	*	*	*	*	*	*	*					

2. Function and action description

Convert the BIN (binary) data of S into BCD (decimal) data and transfer it to D. For 16-bit instructions, the data of S can be converted into BCD (decimal) of K0 ~ K9999 , and for 32 -bit instructions, the data of S can be converted into BCD (decimal) of K0 ~ K99999999.

3. Program Example



D0	0	0	0	0	0	0	0	0	1	1	1	1	1	0	1	0	250
D10	0	0	0	0	0	0	1	0	0	1	0	1	0	0	0	0	592

The data 250 is written into D0. The BCD code of 250 is 0010 (2) 0101 (5) 0000 (0).

Precautions

4. Pay attention to the value range of the source operand.

2.3.10 BIN/BIN conversion instructions

This command converts a decimal number (BCD) into a binary number (BIN).

Instruction Format

FNC19	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
16-bit	5	BIN	(D)BIN(P) S D
		BIN	
32-bit	9	DBIN	
		DBINP	

Operand Description

Operands	Content	Type
S	Source data (decimal)	BIN16/32 bits
D	Target data (binary number)	BIN16/32 bits

Available operand component types

Operands	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
S								*	*	*	*	*	*	*	*	*	*	*					
D									*	*	*	*	*	*	*	*	*	*					

2. Function and action description

Convert the BCD (decimal) data of S into BIN (binary) data and transfer it to D. For 16-bit instructions, S can be converted within the range of 0 to 9999 (BCD) , and for 32-bit instructions, S can be converted within the range of 0 to 99999999 (BCD).

3. Program example

The data 592 is written into D10. 592 corresponds to the BCD data of 250. 250 converted to binary is 1111 1010.

4. Notes

Pay attention to the value range of the source operand.

2.4 Four logical operation instructions

Used for basic addition, subtraction, multiplication, division and logical operations.

2.4.1 ADD/Addition Instruction

An instruction to add two data .

1. Instruction Format

FNC20	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
16-bit	7	ADD	(D)ADD(P) S1 S2 D
		ADDP	
32-bit	13	DADD	
		DADDP	

Operand Description

Operands	Content	Type
S1	Data to be added 1	BIN16/32 bits
S2	Data to be added 2	BIN16/32 bits
D	Addition result	BIN16/32 bits

Available operand component types

Operands	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
S1								*	*	*	*	*	*	*	*	*	*	*	*	*			
S2								*	*	*	*	*	*	*	*	*	*	*	*	*			
D									*	*	*	*	*	*	*	*	*	*					

2. Function and action description

Perform binary addition on the Contents of S1 and S2 and transfer them to D.

3. Program example

Slightly.

4. Notes

The relationship between the action of the flag bit and the positive and negative value:

Software	Name	Content
SM8020	Zero	ON: When the calculation result is 0 OFF: When the calculation result is other than 0
SM8021	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: When the operation result is not less than -32,768 (16-bit operation) or -2,147,483,648 (32-bit operation)
SM8022	Carry	ON: When the calculation result is greater than 32,767 (16-bit calculation) or 2,147,483,647 (32-bit calculation), the carry flag is activated. OFF: When the calculation result is not greater than 32,767 (16-bit calculation) or 2,147,483,647 (32-bit calculation)

2.4.2 SUB/Subtraction Instructions

To subtract two data .

1. Instruction Format

FNC21	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
16-bit	7	SUB	(D)SUB(P) S1 S2 D
		SUBP	
32-bit	13	DSUB	
		DSUBP	

Operand Description

Operands	Content	Type
S1	Data to be subtracted 1	BIN16/32 bits
S2	Data to be subtracted 2	BIN16/32 bits
D	Subtraction result	BIN16/32 bits

Available operand component types

Operands	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
S1								*	*	*	*	*	*	*	*	*	*	*	*	*			
S2								*	*	*	*	*	*	*	*	*	*	*	*	*			
D									*	*	*	*	*	*	*	*	*	*					

2. Function and action description

Perform binary subtraction on the Contents of S1 and S2 and transfer them to D.

3. Program example

Slightly.

4. Notes

The relationship between the action of the flag bit and the positive and negative value:

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

2.4.3 MUL/Multiplication Instruction

To multiply two data.

1. Instruction Format

FNC22	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
16-bit	7	MUL	(D)MUL(P) S1 S2 D
		MULP	
32-bit	13	DMUL	
		DMULP	

Operand Description

Operands	Content	Type
S1	To be multiplied 1	BIN16/32 bits
S2	To be multiplied 2	BIN16/32 bits
D	Multiplication result	BIN32/64 bit

Available operand component types

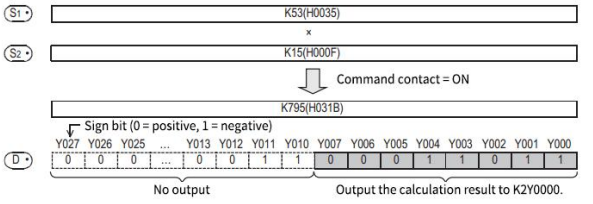
Operands	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
S1								*	*	*	*	*	*	*	*	*	*	*	*	*			
S2								*	*	*	*	*	*	*	*	*	*	*	*	*			
D									*	*	*	*	*	*	*	*	*	*					

2. Function and action description

For 16-bit instructions, the Contents of S1 and S2 (both S1 and S2 are single words) are binary multiplied and transferred to the 32-bit data starting from D.

For 32-bit instructions, the Contents of S1 and S2 (both S1 and S2 are double words) are binary multiplied and transferred to 64 bits (four consecutive word soft elements).

3. Program example



(D+1, D) When the number of bits (K1 to 8) is specified, for example, when K2 is specified, only the lower 8 bits of the product (32 bits) can be obtained.

4. Notes

(1) When using combination bits, the calculation result is the K coefficient multiplied by the data of the combination of 4 bit elements .

(2) When the operation result is 0, SM8304 is turned ON, and other results are turned OFF.

(3) When performing 32-bit instruction operations, the destination operand cannot be the Z register.

(4) The host computer cannot directly monitor the 64-bit data calculated by the 32-bit instructions. It can only combine the results through multiple registers. For a more intuitive display, it is recommended to use floating-point operations.

(5) Note that for 16-bit and 32-bit instructions, the operands represent the actual register range.

2.4.4 DIV/Division Instructions

To divide two data.

1. Instruction Format

FNC23	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
16-bit	7	DIV	┤[(D)DVI(P) S1 S2 D]
		DIVP	
32-bit	13	DDIV	
		DDIVP	

Operand Description

Operands	Content	Type
S1	Dividend	BIN16/32 bits
S2	divisor	BIN16/32 bits
D	Division result	BIN32/64 bit

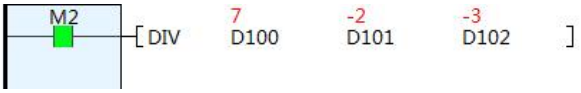
Available operand component types

Operands	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
S1								*	*	*	*	*	*	*	*	*	*	*	*	*			
S2								*	*	*	*	*	*	*	*	*	*	*	*	*			
D									*	*	*	*	*	*	*	*	*	*					

2. Function and action description

After performing binary division on the Contents of S1 and S2, the quotient is transferred to D and the remainder is transferred to D+1 . For 16-bit instructions, D occupies 2 word elements, and for 32-bit instructions, D occupies 4 word elements.

3. Program example



soft components	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0	value
D100	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	7
D101	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	-2
D102	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	1	-3
D103	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1

4. Notes

(1) Regarding the operation results: The highest digit of the quotient and remainder is displayed as a positive (0) or negative (1) sign. When either the dividend or the divisor is negative, the quotient is negative. When the dividend is negative, the remainder is negative.

(2) When specifying a bit device by specifying the number of digits, the remainder cannot be obtained. When using 32-bit operations (DDIV, DDIVP), the Z register cannot be specified.

(3) When the divisor S2 is 0, an operation error occurs. When the operation result exceeds the value corresponding to the number of bits in the instruction, an error is also reported and the carry flag is set to ON.

Software	Name	Content
SM8304	Zero	ON: When the calculation result is 0 OFF: When the calculation result is other than 0
SM8306	Carry	ON: When the calculation result exceeds 32,767 (16-bit calculation) or 2,147,483,647 (32-bit calculation), the carry flag is activated. OFF: When the calculation result is 32,767 (16-bit calculation) or 2,147,483,647 (32-bit calculation) or less

2.4.5 INC/Increment Instruction

Data increment instruction .

1. Instruction format

FNC24	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
16-bit	3	INC	⊣[(D)INC(P) D]
		INCP	
32-bit	5	DINC	
		DINCP	

Operand Description

Operands	Content	Type
D	Data that is incremented by one	BIN16/32 bits

Available operand component types

Operands	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
D									*	*	*	*	*	*	*	*	*	*					

2. Function and action description

Add 1 to the Content of D and transfer it to D.

3. Program example

Slightly.

4. Notes

(1) 16-bit operation: After adding 1 to +32,767, it becomes -32,768, but the flag bits (zero, borrow, carry) are not affected.

(2) 32-bit operation: After adding 1 to +2,147,483,647, it becomes -2,147,483,648, but the flag bits (zero, borrow, carry) are not affected.

2.4.6 DEC/Decrement Instruction

Data decrement instruction .

1. Instruction format

FNC25	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
16-bit	3	DEC	-[(D)DEC(P) D]
		DECP	
32-bit	5	DDEC	
		DDECP	

Operand Description

Operands	Content	Type
D	Data that is decremented by one	BIN16/32 bits

Available operand component types

Operands	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
D									*	*	*	*	*	*	*	*	*	*					

2. Function and action description

The Content of D is decremented by one and then transferred to D.

3. Program example

Slightly.

4. Notes

(1) 16-bit operation: After subtracting 1 from -32,768, it becomes +32,767, but the flag bits (zero, borrow, carry) are not affected.

(2) 32-bit operation **:** After subtracting 1 from -2,147,483,648, it becomes +2,147,483,647, but the flag bits (zero, borrow, carry) are not affected.

2.4.7 WAND/Logical AND Instruction

An instruction that performs a logical AND operation on two data .

1. Instruction format

FNC26	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
16-bit	7	WAND WANDP	WAND(P) S1 S2 D
32-bit	13	DAND DANDP	DAND(P) S1 S2 D

Operand Description

Operands	Content	Type
S1	Data to be logically ANDed 1	BIN16/32 bits
S2	Data for logical AND operation 2	BIN16/32 bits
D	Logic and results	BIN16/32 bits

Available operand component types

Operands	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
S1								*	*	*	*	*	*	*	*	*	*	*	*	*			
S2								*	*	*	*	*	*	*	*	*	*	*	*	*			
D									*	*	*	*	*	*	*	*	*	*					

2. Function and action description

The Contents of S1 and S2 are logically ANDed per bit and then transferred to D.

3. Program example

Slightly .

4. Notes

The logical AND operation is performed bit by bit, as shown in the following table (1∧1=1 0∧1=0 1∧0=0 0∧0=0).

	S1	S2	D
Logical operations on bits	0	0	0
	1	0	0
	0	1	0
	1	1	1

2.4.8 WOR/Logical OR Instruction

An instruction that performs a logical OR operation on two data .

1. Instruction format

FNC27	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
16-bit	7	WOR WOR P	WOR(P) S1 S2 D
32-bit	13	DOR DOR P	DOR(P) S1 S2 D

Operand Description

Operands	Content	Type
S1	Data to be logically ORed 1	BIN16/32 bits
S2	Data to be logically ORed 2	BIN16/32 bits
D	Logical or result	BIN16/32 bits

Available operand component types

Operands	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KX	KjY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
S1								*	*	*	*	*	*	*	*	*	*	*	*	*			
S2								*	*	*	*	*	*	*	*	*	*	*	*	*			
D									*	*	*	*	*	*	*	*	*	*					

2. Function and action description

The Contents of S1 and S2 are logically ORed per bit and then transferred to D.

3. Program example

Slightly.

4. Notes

The logical OR operation is performed in bits, and changes as shown in the following table (1∨1=1 0∨1=1 0∨0=0 1∨0=1).

	S1	S2	D
Logical operations on bits	0	0	0
	1	0	1
	0	1	1
	1	1	1

2.4.9 WXOR/Logical Exclusive OR Instruction

An instruction that performs a logical XOR operation on two data .

1. Instruction format

FNC28	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
16-bit	7	WXOR WXORP	WXOR(P) S1 S2 D
32-bit	13	DXOR DXORP	DXOR(P) S1 S2 D

Operand Description

Operands	Content	Type
S1	Data 1 for logical XOR operation	BIN16/32 bits
S2	Data 2 for logical XOR operation	BIN16/32 bits
D	Logical XOR result	BIN16/32 bits

Available operand component types

Operands	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
S1								*	*	*	*	*	*	*	*	*	*	*	*	*			
S2								*	*	*	*	*	*	*	*	*	*	*	*	*			
D									*	*	*	*	*	*	*	*	*	*					

2. Function and action description

The Contents of S1 and S2 are logically XORed bit by bit and then transferred to D.

3. Program example

Slightly.

4. Notes

The logical XOR operation is in bits, and changes as shown in the following table (1∀1=0 0∀0=0 1∀0=1 0∀1=1).

	S1	S2	D
Logical operations on bits	0	0	0
	1	0	1
	0	1	1
	1	1	1

2.4.10 NEG/Complement Instructions

This command calculates the two's complement of a value (the value after each digit is inverted and added 1). By using this command, the sign of a value can be inverted.

1. Instruction format

FNC29	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
16-bit	3	NEG	-[(D)NEG(P) D]
		NEGP	
32-bit	5	DNEG	
		DNEGP	

Operand Description

Operands	Content	Type
D	or operation result of the complement code	BIN16/32 bits

Available operand component types

Operands	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
D									*	*	*	*	*	*	*	*	*	*					

2. Function and action description

Invert each bit in the D Content (0→1, 1→0) and then add 1 to save the result in the original device.

3. Program example

Slightly.

4. Notes

When using continuous execution type (NEG, DNEG) instructions, they are executed in each scan cycle (each operation cycle), so please be careful.

2.5 Rotate Instructions

Move bit data or word data in the specified direction or by the specified number of bits.

2.5.1 ROR/Rotate Right Instruction

An instruction to right shift and rotate the bit information of the specified number of bits without a carry flag .

1. Instruction format

FNC30	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
16-bit	5	ROR	(D)ROR(P) D n
		RORP	
32-bit	9	DROR	
		DRORP	

Operand Description

Operands	Content	Type
D	The data to be rotated right or the result after shifting	BIN16/32 bits
n	Number of bits to rotate [1≤ n ≤ 16 (16-bit instruction), 1≤ n ≤ 32 (32-bit instruction)]	BIN16/32 bits

Available operand component types

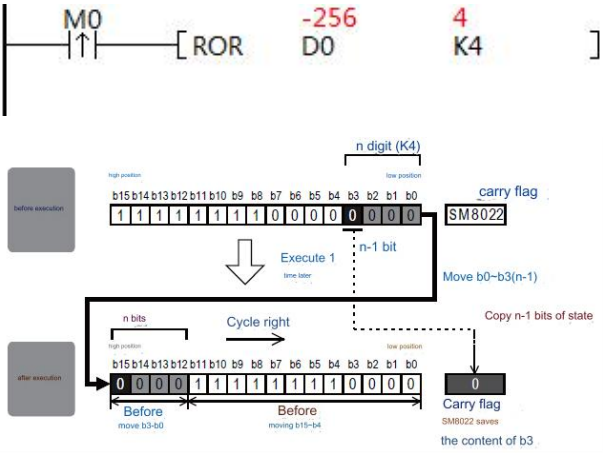
Operands	Bit Components							Word component							Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
D									*	*	*	*	*	*	*	*	*	*					
n														*	*				*	*			

2. Function and action description

All n bits of D are circularly shifted right. The last bit is stored in the carry flag (M8022).

3. Program example

Shift D0 = -256 = 0xFF00 right by 4 bits



4. Notes

(1) Related soft components: M8022, carry flag, turns ON when the last bit shifted out from the lowest bit is 1.

(2) When using continuous execution type (ROR, DROR) instructions, please note that circular shift is executed in each scan cycle (operation cycle).

a combined bit element in D , only K4 (16-bit instruction) or K8 (32-bit instruction) is valid. (For example, K4Y010, K8M0).

2.5.2 ROL/Rotate Left Instruction

An instruction to shift the bit information of the specified number of bits to the left and rotate it , without a carry flag .

1. Instruction format

FNC31	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
16-bit	5	ROL	(D)ROL(P) D n
		ROLP	
32-bit	9	DROL	
		DROLP	

Operand Description

Operands	Content	Type
D	The data to be rotated left or the result after shifting	BIN16/32 bits
n	Number of bits to rotate [1≤ n ≤ 16 (16-bit instruction), 1≤ n ≤ 32 (32-bit instruction)]	BIN16/32 bits

Available operand component types

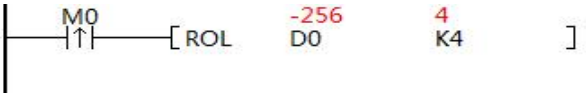
Operands	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
D									*	*	*	*	*	*	*	*	*	*					
n														*	*				*	*			

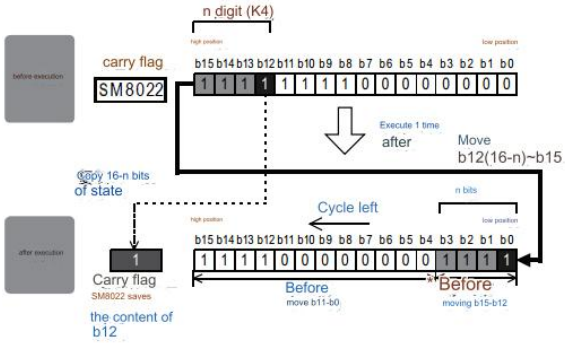
2. Function and action description

All the bits of D are circularly shifted left by n bits . The last bit is stored in the carry flag (SM8022).

3. Program example

Shift D0 = -256 = 0xFF00 4 bits to the left





4. Notes

(1) Related soft components: S M8022, carry flag, turns ON when the last bit shifted out from the highest bit is 1.

(2) When using continuous execution type (ROL, DROL) instructions, please note that circular shift is executed in each scan cycle (operation cycle).

(3) When specifying the number of bits in D to specify a device, only K4 (16-bit instruction) or K8 (32-bit instruction) is valid. (For example, K4Y010, K8M0).

2.5.3 RCR/Rotate Right with Carry Instruction

An instruction to right shift and rotate the bit information of the specified number of bits , with a carry flag.

1. Instruction format

FNC32	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
16-bit	5	RCR	(D)RCR(P) D n
		RCRP	
32-bit	9	DRCR	
		DRCRP	

Operand Description

Operands	Content	Type
D	The data to be rotated right or the result after shifting	BIN16/32 bits
n	Number of bits to rotate [1≤ n ≤ 16 (16-bit instruction), 1≤ n ≤ 32 (32-bit instruction)]	BIN16/32 bits

Available operand component types

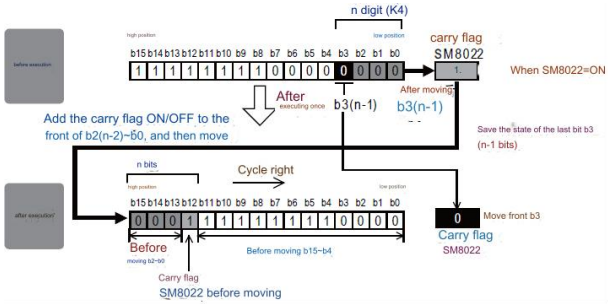
Operands	Bit Components							Word component							Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
D									*	*	*	*	*	*	*	*	*	*					
n														*	*				*	*			

2. Function and action description

All bits of D + 1 bit (carry flag M8022) are shifted right by n bits. Because there is a carry flag in the loop, if M8022 is turned ON or OFF before executing the circular shift instruction, it will be sent to the target operand.

3. Program example

Shift D0 = -256 = 0xFF00 right by 4 bits with carry



4. Notes

(1) Related soft components: M8022, carry flag, turns ON when the last bit shifted out from the lowest bit is 1.

(2) When using continuous execution type (RCR, DRCR) instructions, please note that circular shift is executed in each scan cycle (operation cycle).

(3) When specifying the number of bits in D to specify a device, only K4 (16-bit instruction) or K8 (32-bit instruction) is valid. (For example, K4Y010, K8M0).

2.5.4 RCL/Rotate Left with Carry Instruction

An instruction to shift the bit information of the specified number of bits to the left and rotate it , with a carry flag.

1. Instruction format

FNC33	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
16-bit	5	RCL	(D)RCL(P) D n
		RCLP	
32-bit	9	DRCL	
		DRCLP	

Operand Description

Operands	Content	Type
D	The data to be rotated left or the result after shifting	BIN16/32 bits
n	Number of bits to rotate [1≤ n ≤ 16 (16-bit instruction), 1≤ n ≤ 32 (32-bit instruction)]	BIN16/32 bits

Available operand component types

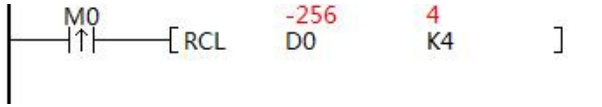
Operands	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
D									*	*	*	*	*	*	*	*	*	*					
n														*	*				*	*			

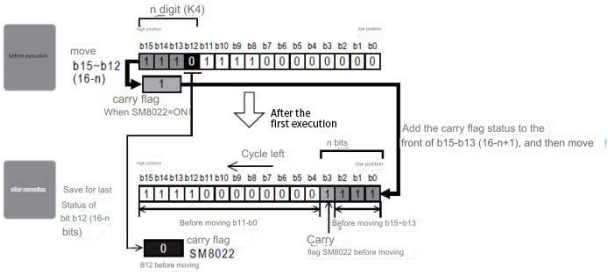
2. Function and action description

All bits of D + 1 bit (carry flag M0822) are shifted left by n bits. Because there is a carry flag in the loop, if M8022 is turned ON or OFF before executing the circular shift instruction, it will be sent to the target operand.

3. Program example

Shift D0 = -256 = 0xFF00 left by 4 bits with carry





4. Notes

(1) Related soft components: M8022, carry flag, turns ON when the last bit shifted out from the highest bit is 1.

(2) When using continuous execution type (RCL, DRCL) instructions, please note that circular shift is executed in each scan cycle (operation cycle).

(3) When specifying the number of bits in D to specify a device, only K4 (16-bit instruction) or K8 (32-bit instruction) is valid. (For example, K4Y010, K8M0).

2.5.5 SFTR/Bit Shift Right Instruction

This command shifts the specified bit length right. After the shift, n2 points of S-bit soft elements are transferred starting from the highest bit.

1. Instruction format

FNC34	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
16-bit	9	SFTR SFTRP	SFTR(P) S D n1 n2

Operand Description

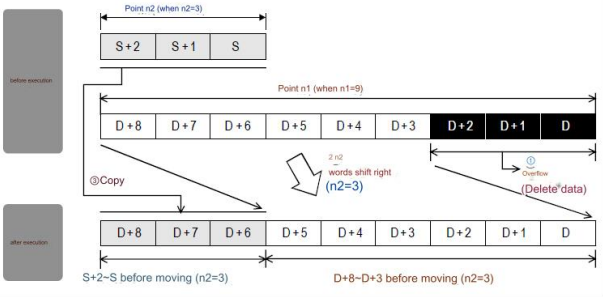
Operands	Content	Type
S	The starting bit element of the transmitted bit data	Bit
D	Right shift data start bit soft element	Bit
n1	of the right shifted data is n2≦n1≦1024	BIN16
n2	The number of bits shifted right is n2≦n1≦1024	BIN16

Available operand component types

Operands	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
S	*	*	*			*	*											*					
D		*	*			*												*					
n1														*	*				*	*			
n2														*	*				*	*			

2. Function and action description

For the n1-bit data starting with D, shift right by n2 bits (①, ② below). After the shift, transfer the n2-bit data starting with S (③ below) to the n2-bit data starting with D + n1-n2.



3. Program example

shifting the 8-bit components Y10~Y17 right by 3 bits, transfer Y0~Y2 to the high bits Y15~Y17.

Before execution

soft components	7	6	5	4	3	2	1	0
Y0	0	0	0	0	0	1	0	1
Y10	0	0	0	1	1	1	1	1

After execution

soft components	7	6	5	4	3	2	1	0
Y0	0	0	0	0	0	1	0	1
Y10	1	0	1	0	0	0	1	1

4. Notes

(1) Each time the instruction input changes from OFF to ON, n2 bit shifts are executed. When the instruction input state remains ON, the shift is executed in each scan cycle (operation cycle).

(2) When the transmission source S and the shift data D are repeated, error 6710 is reported.

2.5.6 SFTL/Bit Shift Left Instruction

This command shifts the specified bit length to the left . After the shift , n2 points of S-bit soft elements are transferred starting from the lowest bit.

1. Instruction format

FNC35	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
16-bit	9	SFTL	SFTL(P) S D n1 n2
		SFTLP	

Operand Description

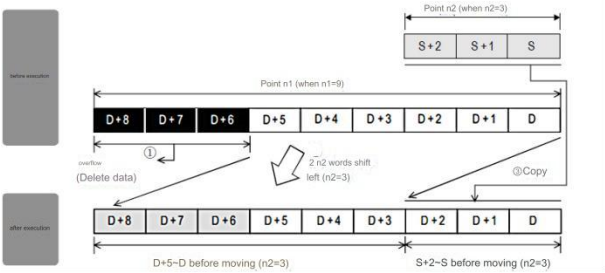
Operands	Content	Type
S	The starting bit element of the transmitted bit data	Bit
D	Left shift data start bit soft element	Bit
n1	of the left shifted data is n2≦n1≦1024	BIN16
n2	The number of bits shifted left is n2≦n1≦1024	BIN16

Available operand component types

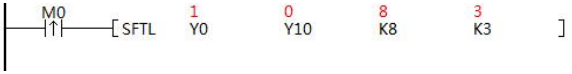
Operands	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
S	*	*	*			*	*											*					
D		*	*			*												*					
n1																			*	*			
n2														*	*				*	*			

2. Function and action description

For the n1-bit data starting with D, shift left by n2 bits (①, ② below). After the shift, transfer the n2-bit data starting with S (③ below) to the n2-bit data starting with D.



3. Program example



Shifting the 8-bit components Y10~Y17 to the left by 3 bits, Y0~Y2 are transferred to the low-bit Y10~Y12.

Before execution

soft components	7	6	5	4	3	2	1	0
Y0	0	0	0	0	0	1	0	1
Y10	1	1	1	1	1	0	0	0

After execution

soft components	7	6	5	4	3	2	1	0
Y0	0	0	0	0	0	1	0	1
Y10	1	1	0	0	0	1	0	1

4. Notes

(1) Each time the instruction input changes from OFF to ON, n2 bit shifts are executed. When the instruction input state remains ON, the shift is executed in each scan cycle (operation cycle).

(2) When the transmission source S and the shift data D are repeated, error 6710 is reported.

2.5.7 WSFR/Word Shift Right Instruction

This is an instruction to shift n1 word devices to the right by n2 words.

1. Instruction format

FNC36	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
16-bit	9	WSFR WSFRP	WSFR(P) S D n1 n2

Operand Description

Operands	Content	Type
S	The starting word element of the word data to be transferred	BIN16
D	The starting word device of the right shift data	BIN16
n1	The word length of the right shifted data is n2≦n1≦512	BIN16
n2	The number of words shifted right is n2≦n1≦512	BIN16

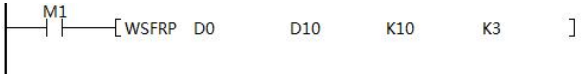
Available operand component types

Operands	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
S								*	*	*	*	*	*	*	*			*					
D									*	*	*	*	*	*	*			*					
n1																			*	*			
n2														*	*				*	*			

2. Function and action description

For n1-word soft elements starting with D, shift right by n2 words (①, ② below). After shifting, transfer n2-point data starting with S (③ below) to n2 points starting with (D+n1-n2).

3. Program example



Shifting the 10-bit components D10~D19 right by 3 bits, D0~D2 are transferred to D17~D19.

Before execution

soft components	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0	value
D0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
D1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1
D2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	2
D3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
D4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
D5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
D6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
D7	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
D8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
D9	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
D10	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	10
D11	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	1	11
D12	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	12
D13	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	1	13
D14	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	0	14
D15	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	15
D16	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	16
D17	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	17
D18	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	18
D19	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	1	19

After execution

soft components	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0	value
D0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
D1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1
D2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	2
D3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
D4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
D5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
D6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
D7	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
D8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
D9	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
D10	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	1	13
D11	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	0	14
D12	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	15
D13	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	16
D14	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	17
D15	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	18
D16	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	1	19
D17	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
D18	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1
D19	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	2

4. Notes

(1) In the WSFRP instruction, a shift of n2 words is executed each time the instruction input changes from OFF to ON. However, please note that in the WSFR instruction, a shift is executed in every operation cycle.

(2) When the transfer source S and the shift device D are duplicated, an operation error occurs. (Error code: K6710)

2.5.8 WSFL/Word Shift Left Instruction

to shift n1 words of data left by n2 words .

1. Instruction format

FNC37	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
16-bit	9	WSFL WSFLP	WSFL(P) S D n1 n2

Operand Description

Operands	Content	Type
S	The starting word element of the word data to be transmitted	BIN16
D	The starting word device of the left shift data	BIN16
n1	The word length of the left shifted data is n2≦n1≦512	BIN16
n2	The number of words shifted left is n2≦n1≦512	BIN16

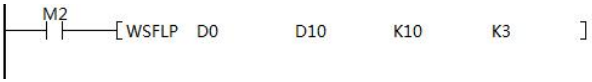
Available operand component types

Operands	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
S								*	*	*	*	*	*	*	*			*					
D									*	*	*	*	*	*	*			*					
n1																			*	*			
n2														*	*				*	*			

2. Function and action description

Shift the n1-word soft element starting with D to the left by n2 words (①, ② below). After the shift, transfer the n2 points starting with S (③ below) to the n2 points starting with D.

3. Program example



Shifting the 10 bit components D10~D19 to the left by 3 bits, D0~D2 are transferred to D10~D12.

4. Notes

(1) In the WSFLP instruction, a shift of n2 words is executed each time the instruction input changes from OFF to ON. However, please note that in the WSFL instruction, a shift is executed in every operation cycle.

(2) When the transfer source S and the shift device D are duplicated, an operation error occurs. (Error code: K6710)

2.5.9 S FWR/Shift Write Instruction

To implement the first-in, first-out or first-in, last-out control function, the prepared data is first written into the instruction , and the data can be read out using the SFRD or POP instruction.

1. Instruction format

FNC38	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
16-bit	7	SFWR	SFWR(P) S D n1 n2
		SFWRP	

Operand Description

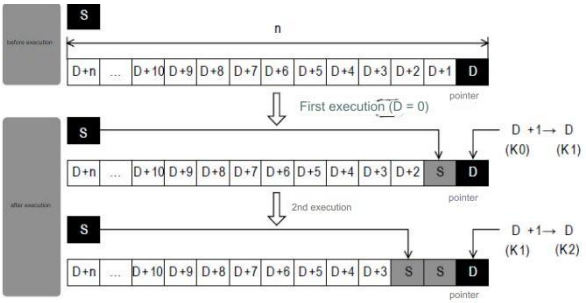
Operands	Content	Type
S	Data written	BIN16
D	The starting word device for storing data and shifting (D stores the number of data points, and D+1 starts to store the written data)	BIN16
n	The value of the number of saved data points + 1 is 2≦n≦512	BIN16

Available operand component types

Operands	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
S								*	*	*	*	*	*	*	*	*	*	*	*	*			
D									*	*	*	*	*	*	*			*					
n																			*	*			

2. Function and action description

Write the Contents of S in sequence at points n-1 starting from D+1, and increase the number of data stored in D by 1.



(1) When X000 changes from OFF to ON, the Content of S is saved in D+1, and the Content of D+1 becomes the value of S.

(2) After the Content of S changes, the input is executed again from OFF to ON. The Content of S is saved in D+2, and the Content of D+2 becomes S. (Since the continuous

execution type instruction SFWR is used, each operation cycle is saved in sequence, so please use the pulse execution type instruction SFWRP for programming.)

(3) Execute from the right end in sequence, and the number of saved data points is stored in D.

3. Program example

Slightly.

4. Notes

(1) The action of the carry flag M8022: When the Content of pointer D exceeds n-1, it becomes no process (not written) and the carry flag M8022 is turned ON.

(2) Please note that each scan cycle (operation cycle) will be saved (overwritten) in sequence.

2.5.10 SF RD/Shift Read Instruction

Data read instruction in first-in-first-out format .

1. Instruction format

FNC39	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
16-bit	7	SFRD SFRDP	SFRD(P) S D n1 n2

Operand Description

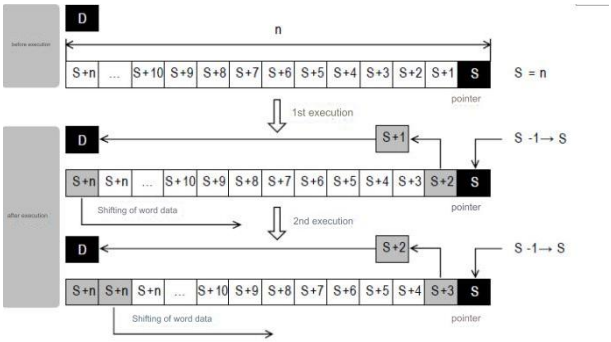
Operands	Content	Type
S	The starting word element of the data to be read (S is the number of data points, and the data starts from S + 1)	BIN16
D	Read data	BIN16
n	The value of the number of points of the saved data + 1. 2≦n≦512	BIN16

Available operand component types

Operands	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
S									*	*	*	*	*	*	*			*					
D									*	*	*	*	*	*	*	*	*	*					
n																			*	*			

2. Function and action description

After the S1+1 written in sequence is transferred (read) to D using the SFRD (FNC 38) instruction, the n-1 points starting from S1+1 are shifted right word by word. The number of data stored in S is -1.



(1) When the command contact is ON, the Content of S+1 is transferred (read) to D.

(2) At the same time, the Content of S decreases and the data on the left is shifted right word by word.

3. Program example

Slightly.

4. Notes

(1) The action of the zero flag S M8020. The data is read out usually starting from S+1. However, when the Content of pointer S is 0, the zero flag S M8020 is activated.

(2) The Content of S+n will not change due to reading.

(3) Each scan cycle (operation cycle) will execute the readout in sequence, but the Content of S+n will not change.

(4) When S is 0, the instruction is not executed.

2.6 Data processing instructions

Process complex data or meet special data purposes.

2.6.1 ZRST/Bulk Reset Instruction

Reset (clear) instruction for batch data.

1. Instruction format

FNC40	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
16-bit	5	ZRST	ZRST(P) D1 D2
		ZRSTP	

Operand Description

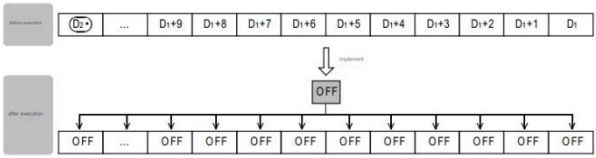
Operands	Content	Type
D1	Start element for batch reset	Bit / BIN16 bit
D2	End element of batch reset	Bit / BIN16 bit

Available operand component types

Operands	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
D1		*	*			*						*	*	*	*			*					
D2		*	*			*						*	*	*	*			*					

2. Function and action description

Reset all D1~D2 of the same type , turn off the bit components, and clear the word components .



3. Program example

Slightly.

4. Notes

(1) D1 and D2 must be specified as the same type of soft elements, and the number D1 ≤ D2. When the number D1 is greater than D2, the soft element specified in D1 is reset by only one point.

(2) ZRST is used as a 16-bit processing instruction. A 32-bit counter can also be specified in D1 and D2, but the number of register bits in D1 and D2 cannot be different.

2.6.2 DECO/Decode Instructions

This command converts any one of the digital data into a 1-point ON bit. The bit number can be read as a numerical value based on the position of the ON bit.

1. Instruction Format

FNC41	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
16-bit	7	DECO DECOP	DECO(P) S D n

Operand Description

Operands	Content	Type
S	Data to be decoded	BIN16 bits
D	Decoding results	BIN16 bits
n	The number of bits of the soft element that stores the decoding result (n=1~8) (no processing when n=0)	BIN16 bits

Available operand component types

Operands	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
S	*	*	*			*						*	*	*	*	*	*	*	*	*			
D		*	*			*						*	*	*	*			*					
n																			*	*			

2. Function and action description

The Sth position in D to D+2 n -1 corresponding to the value of S is turned on.

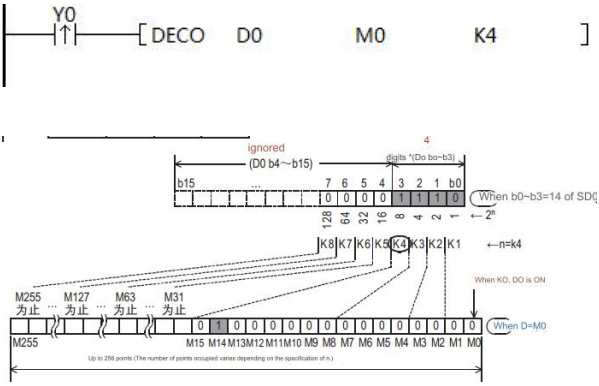
(1) D is a bit soft element: 1≦n≦8, the n-digit number of the soft element specified in S (1≦n≦8) is decoded in D. When S is all 0, the bit soft element D is ON. When n=8, the maximum 2 8 =256 points.

(2) When D is a word device: 1≦n≦4, the lower n bits of S are decoded in D. When all S are 0, b0 of the bit device D is ON.

3. Program example

(1) D is the value of the bit soft element

D0 (the current value is 14) which is decoded in M0~M15 . M14 is set to ON and the other bits are 0 .



• When the value of b0 to b3 of D0 is 14 (0+2+4+8), M14, the 15th number from M0, is 1 (ON).

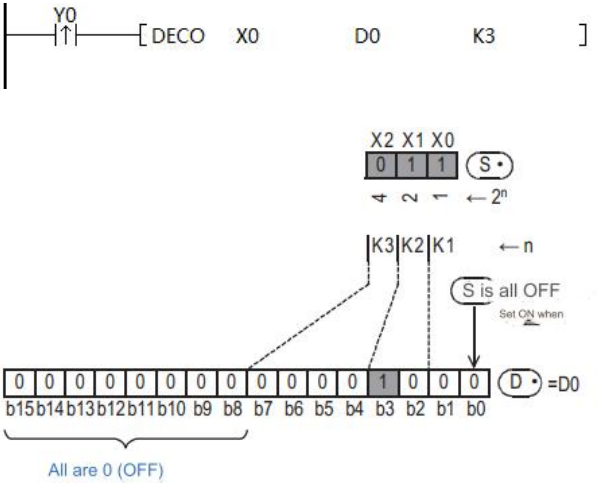
• When D0=0, M0 is 1 (ON).

• When n=K4, any one of M0 to M15 is 1 point (ON) according to the value of D0 (0 to 15).

• If n is changed between K1 to K8, D0 can correspond to the value of 0 to 255.

(2) D is a word soft element

The values of X000 to X002 (X000 and X001 are ON, and X002 is OFF) are decoded in D0.



• When the value of X000 to X002 is 3 (1+2+0), b3 of the fourth bit starting from b0 is 1 (ON).

• When X000 to X002 are all 0 (OFF), b0 is 1 (ON).

4. Notes

(1) When n changes between K1 and K8, please note that the range of the bit soft element of the destination register D cannot overlap with other controls.

(2) When n is 0, the instruction is not processed.

2.6.3 ENCO/Encoding Instructions

This command finds the position of the ON bit in the data.

1. Instruction format

The instruction format is shown in the table below.

FNC42	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
16-bit	7	ENCO ENCOP	ENCO(P) S D n

Operand Description

Operands	Content	Type
S	Data to be encoded	BIN16 bits
D	Coding results	BIN16 bits
n	stores the encoding result (n=1 to 8) (no processing when n=0)	BIN16 bits

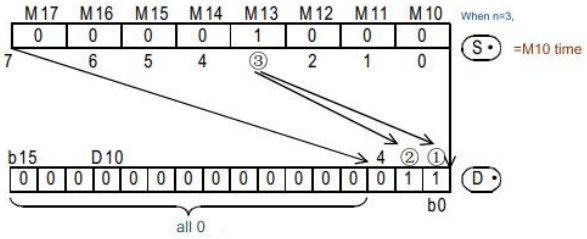
Available operand component types

Operands	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
S	*	*	*			*						*	*	*	*	*	*	*					
D												*	*	*	*	*	*	*					
n																			*	*			

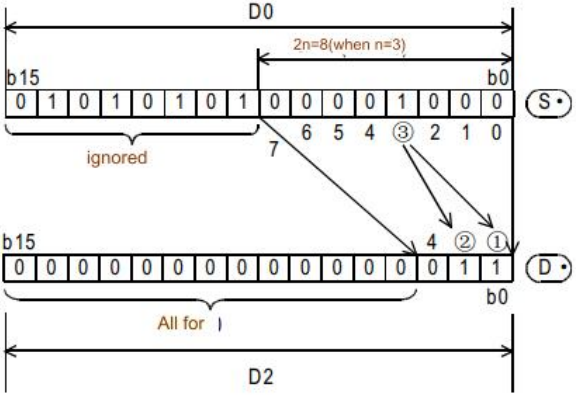
2. Function and action description

$^{2n-}$ bit encoded value of S is stored in D. Encoding is to convert the position value of the ON bit into BIN data.

(1) S is a bit soft element: $1 \leqq n \leqq 8$. The 2n ON bit positions starting from S ^{are} encoded in D. When n=8, S is a maximum of 256 points. The encoding result of D is n bits from the high bit to the low bit, all of which are 0 (OFF).



(2) S is a word device: 1 ≤ n ≤ 4, and the ON bit positions up to bit 2 of the device specified in S [are] encoded in D. The encoding result of D is n bits from the highest bit to the lowest bit, and all are 0 (OFF).



3. Program example

Slightly.

4. Notes

When multiple bits are ON in the data of S, the lower bits are ignored and the ON bits on the higher bits are encoded.

2.6.4 SUM/Count ON digits instruction

bits in the specified data are "1" (ON).

1. Instruction format

FNC43	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
16-bit	5	SUM	┤[(D)SUM(P) S D]
		SUMP	
32-bit	9	DSUM	
		DSUMP	

Operand Description

Operands	Content	Type
S	Source Data	BIN16/32 bits
D	Target data	BIN16/32 bits

Available operand component types

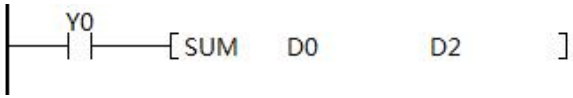
Operands	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
S								*	*	*	*	*	*	*	*	*	*	*	*	*			
D									*	*	*	*	*	*	*	*	*	*					

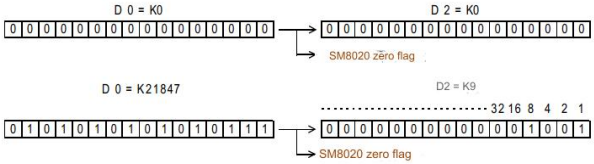
2. Function and action description

The ON bits in S are counted and stored in D. When S is all 0 (OFF), the zero bit flag S M802 0 is ON.

3. Program example

Y0 is ON, the number of ON bits in D0 is counted and stored in D2





4. Notes

The related flag SM8020.

2.6.5 BON/ON bit determination instructions

detects whether the specified bit position is ON or OFF .

1. Instruction format

FNC44	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
16-bit	7	BON	(D)BON(P) S D n
		BONP	
32-bit	13	DBON	
		DBONP	

Operand Description

Operands	Content	Type
S	Source Data	BIN16/32 bits
D	Target data	Bit
n	Bit position to be determined [n: 0 to 15 (16-bit instruction), n: 0 to 31 (32-bit instruction)]	BIN16/32 bits

Available operand component types

Operands	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
S								*	*	*	*	*	*	*	*	*	*	*	*	*			
D		*	*			*	*											*					
n														*	*				*	*			

2. Function and action description

The n-bit state (ON/OFF) of S is output to D.

3. Program example

soft components	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0	value
D0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	1	1	75

When the 6th bit of D0 is 1, Y0=ON

soft components	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0	value
D0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	1	11

When the 6th bit of D0 is 0, Y0=OFF

4. Notes

When D and R are specified as n of a 32-bit instruction, the 32-bit value [n+1,n] becomes effective. Please note that

when DBON D0 M0 R0, n=[R1, R0]

2.6.6 MEAN/Average value instruction

Command to find the average of data.

1. Instruction format

FNC45	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
16-bit	7	MEAN	⊣[(D)MEAN(P) S D n]
		MEANP	
32-bit	13	DMEAN	
		DMEANP	

Operand Description

Operands	Content	Type
S	Source data for averaging	BIN16/32 bits
D	average value	BIN16/32 bits
n	data to be averaged (n=1～64)	BIN16/32 bits

Available operand component types

Operands	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
S								*	*	*	*	*	*	*	*			*					
D									*	*	*	*	*	*	*	*	*	*					
n														*	*				*	*			

2. Function and action description

n 16-bit or 32-bit (depending on the instruction type) data starting from S into D and discard the remainder.

3. Program example

Add the data of D0, D1, and D2, divide by 3, and save the resulting value in D10

D0 = 33, D1 = 44, D2 = 66, after M0 is set, D10 = (D0 + D1 + D2) / 3 = 47.

4. Notes

(1) When the device number overflows, n is processed as a smaller value within the possible range.

(2) Pay attention to the range of n (n=1～64). If it exceeds the range, an error will be reported.

2.6.7 ANS/Signal Alarm Set Command

This is a command for setting the status (S900 to S999) for the signal alarm.

1. Instruction format

FNC46	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
16-bit	7	ANS \	ANS S m D

Operand Description

Operands	Content	Type
S	Timing timer	BIN16
m	Determination time [m = 1 to 32,767 (100ms unit)]	BIN16
D	Signal alarm	Bit

Available operand component types

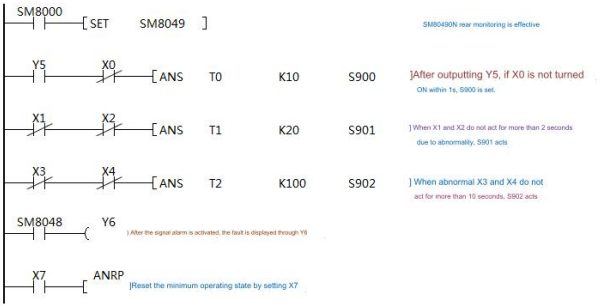
Operands	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
S												*						*					
m														*	*				*	*			
D						*												*					

2. Function and action description

When the command input remains ON for more than the judgment time [m×100ms, timer S], D is set. When the command input is OFF before the judgment time [m×100ms], the current value of the judgment timer S is reset and D is not set. In addition, when the command input is OFF, the judgment timer is reset.

3. Program example

As shown below, a program for diagnosing external faults is written. For example, when monitoring the Content of S D8049 (the smallest number in the ON state), the smallest number in the ON state among S900 to S999 will be displayed. When multiple faults occur at the same time, the next fault number can be found after eliminating the fault with the smallest number.



4. Notes

(1) When S is T soft element, the range is T00 to T199, and the alarm status soft element range is S900 to S999.

(2) Note the actions of special registers related to S900 to S999.

Software	Name	Content
SM8049	Signal alarm is effective	SM8049 is turned ON, the following SM8048 and SD8049 work.
SM8048	Signal alarm action	SM8049 is ON and any of the states S900 to S999 occurs, SM8048 is turned ON.
SD8049	ON state minimum number	Saves the smallest action number among S900 to S999.

2.6.8 ANR/Signal Alarm Reset Command

Reset the small numbers that have been turned on in the signal alarm (S900~S999).

1. Instruction format

FNC47	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
16-bit	1	ANR	ANR(P)
		ANRP	

2. Function and action description

16 -bit operation instruction, when the ANR instruction input is turned on, the operating state of the signal alarm S900~S999 is reset. If there are multiple states in action, the state with the smallest number is reset. When the instruction input is turned on again, the next smallest number is reset in the state of the signal alarm (S900~S999) in action.

Software	Name	Content
SM8049	Signal alarm is effective	SM8049 is turned ON, the following SM8048 and SD8049 work.
SM8048	Signal alarm action	SM8049 is ON and any of the states S900 to S999 occurs, SM8048 is turned ON.
SD8049	ON state minimum number	Saves the smallest action number among S900 to S999.

3. Program example

Please refer to ANS instructions.

4. Notes

(1) Actions of related registers:

Software	Name	Content
SM8049	Signal alarm is effective	SM8049 is turned ON, the following SM8048 and SD8049 work.
SM8048	Signal alarm action	SM8049 is ON and any of the states S900 to S999 occurs, SM8048 is turned ON.
SD8049	ON state minimum number	Saves the smallest action number among S900 to S999.

(2) When using the ANR instruction, the minimum label is reset in each operation cycle. The ANPR instruction only executes one operation cycle. Select the corresponding instruction to reset the corresponding alarm according to actual needs.

2.7 High-speed instruction processing

Program control can be performed according to the latest input and output information, and interrupt processing can be performed by effectively utilizing the high-speed data processing capability.

2.7.1 REF/Input/Output Refresh Instruction

An instruction to obtain the latest input (X) and output the output (Y) scan result immediately.

1. Instruction format

FNC50	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
16-bit	5	REF REFP	⊣[REF(P) D n]

Operand Description

Operands	Content	Type
D	Refreshed bit soft element (X, Y)	Bit
n	Refresh points (multiples of 8 from 8 to 256)	BIN16 bits

Available operand component types

Operands	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
D	*	*																					
n																			*	*			

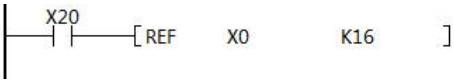
2. Function and action description

Update the status of n components starting from address D immediately. Since the PLC access port is accessed by bytes, the address of S is required to be X0, X10, ... Y0, Y10, ... and other numbered components with the lowest bit being 0. And the value of n must be an integer multiple of 8 (n=8~256).

Under normal circumstances, the status of input port X is read before each program starts to execute the scan. The actual input port status change delay is determined by the filter time of the input element. X0~X7, X10~X17 have digital filtering functions, and the filter time can be set from 0 to 60ms (REFF instruction, the default filter time is 10ms); the status of output port Y is refreshed in batches after each program execution scan is completed (executed to END), so there will be a certain delay in IO processing. If the application requires the latest input signal and hopes to output the operation result immediately, you can use the immediate refresh instruction REF.

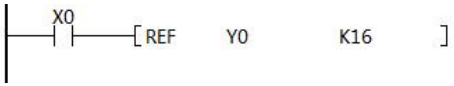
3. Program example

(1) Refresh input signal immediately



When X20 is ON, the input point status of X0~X17 will be read immediately and the input signal will be updated without any input delay.

(2) Refresh output signal immediately



When X0 is ON, the status of Y0~Y17 will be refreshed immediately, and the output signal will change immediately without having to wait until the END instruction to output.

4. Notes

(1) Apply this instruction according to the actual application requirements to refresh the input and output immediately at the current position, rather than refreshing the input and output according to the running process of the ladder diagram.

(2) Immediate refresh is only valid for the main terminals.

2.7.2 REFF/Input/Output Refresh Instruction with Filtering

The input filter of input X0~X17 is a digital filter. The filter time can be changed by using REFF instruction and SD8020 .

1. Instruction format

FNC51	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
16-bit	3	REFF REFFP	-[REFF(P) n]

Operand Description

Operands	Content	Type
n	Filter time [0~60ms]	BIN16 bits

Available operand component types

Operands	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
n														*	*				*	*			

2. Function and action description

The 16-point image storage area of input X000~X017 is refreshed according to the input digital filter time.

The filter time will be different in different usage scenarios. Here we take X3 -32 as an example:

(1) When n is set to 0, the actual corresponding input filter filtering time is as follows

Input number	The filter value when the input filter constant is set to 0
X0~X7 (high-speed terminal)	1.5us
X10 ~ X17 (common terminal)	0.4ms

(2) For input terminals used for high speed or interruption, when the high speed or interruption program is running, the filter setting shall be based on the minimum filter time of 1.5us.

3. Program example

Slightly.

4. Notes

(1) Pay attention to the input used in high speed and the input of the interrupt pointer specified in the input interrupt function. In this scenario, the filter time of the REFF instruction to refresh the SD8020 cannot be processed normally, but the filter setting is performed according to the above instructions.

(2) Immediate refresh is only valid for the main terminals.

2.7.3 HSCS/High Speed Compare Set Instruction

Each time it is executed, the count value of the high-speed counter is compared with the specified value. If the two values are equal, the comparison output instruction is immediately set.

1. Instruction format

FNC53	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
32-bit	13	DHSCS \	⊣[DHSCS S1 S2 D]

Operand Description

Operands	Content	Type
S1	High-speed comparison of set value	BIN32 bits
S2	High-speed counter [C235～C255]	BIN32 bits
D	The object to be positioned	Bit

Available operand component types

Operands	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
S1								*	*	*	*	*	*	*	*		*	*	*	*			
S2													*					*					
D		*	*			*	*											*					*

2. Function and action description

(1) When the current value of the S2 counter is equal to the set value S1, D is immediately set. The S2 variable must be a high-speed counter C235~C255.

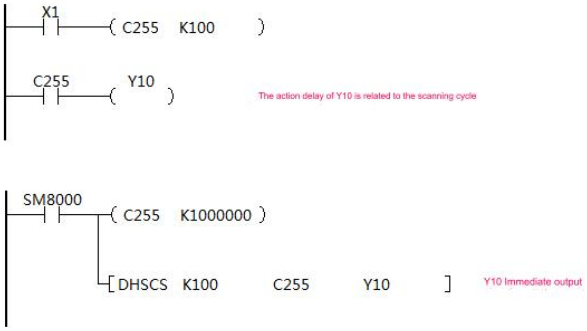
(2) When D is a common bit element: when it is the output range port of the main body, it is output immediately; when it is the port after the main body , it will wait until the current user

program scan is completed before outputting; when it is an M, S, or Dx.y variable, it is refreshed immediately.

(3) When D is the pointer to call the counter interrupt subroutine: When D is I010~I060, it is the subroutine to call the high-speed counter interrupt 0~5. Of course, the corresponding interrupt subroutine must be written, the corresponding interrupt enable flag and the global interrupt enable flag must be turned on, etc., in order to respond to the counter interrupt normally. When SM8059 is turned on, all high-speed counter interrupts (I010~I060) are disabled.

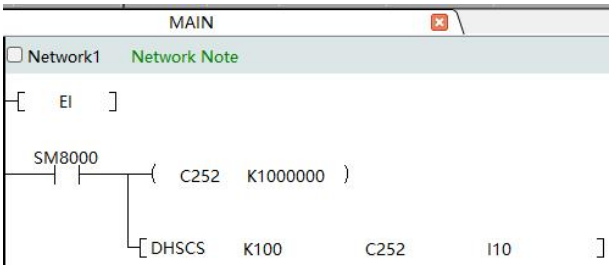
3. Program example

(1) Setting a normal bit element

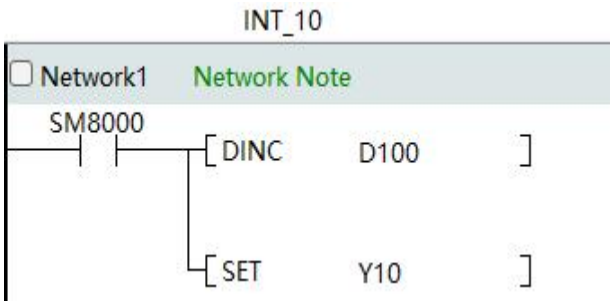


(2) Call the counting interrupt subroutine

Main program:



I010 interrupt subroutine:



4. Notes

(1) When using the HSCS instruction, ensure that the counter used is enabled, otherwise the counter value will not change;

(2) The counter responds to the input signal of the counter in an interrupt mode and compares in time. If the matching relationship is met during this comparison, the comparison output is immediately set. For example, in the above example 1, if the current value of C255 changes from 99->100 or 101->100, Y10 is immediately set (only if it is an output terminal of the main body, it will be immediately set) and remains in this state. After that, even if the comparison result of C255 and K100 becomes unequal, Y10 still remains in the ON state unless there is another reset operation.

(3) The comparison output result of the instruction depends only on the comparison result when the pulse is input. Even if the Contents of C235~C255 are rewritten using instructions such as DMOV and DADD, the comparison output will not change if there is no pulse input.

(4) When the output target of the HSCS instruction is counter interrupts I010 to I060, each interrupt number can only be used once and cannot be repeated.

(5) HSCS, HSCR, and HSZ can be used multiple times like ordinary instructions, but the number of instructions driven simultaneously is limited to 6 instructions per high-speed input, and a maximum of 24 instructions for 4 high-speed inputs. However, the HSZ instruction special mode (high-speed table comparison mode, frequency control mode) can only drive 1 instruction at the same time in the ladder diagram, and does not affect the other 6 instructions .

2.7.4 HSCR/High Speed Comparator Reset Instruction

Each time it is executed, the count value of the high-speed counter is compared with the specified value. If the two values are equal, the comparison output instruction is reset immediately.

1. Instruction format

FNC54	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
32-bit	13	DHSCR \	DHSCR S1 S2 D

Operand Description

Operands	Content	Type
S1	High-speed comparison of set value	BIN32 bits
S2	High-speed counter [C235～C255]	BIN32 bits
D	The bit element to be reset	Bit

Available operand component types

Operands	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
S1								*	*	*	*	*	*	*	*		*	*	*	*			
S2													*										
D		*	*			*	*						*					*					

2. Function and action description

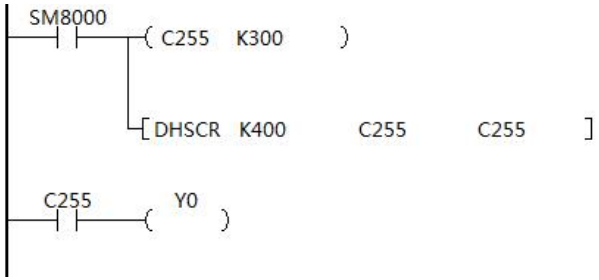
(1) When the current value of S2 counter is equal to the set value S1, D is reset immediately. S2 variable must be a high-speed counter C235~C255.

(2) When D is a normal bit element: when it is the output range port of the main body, it is reset immediately ; when it is a port after the main body , it will wait until the current user program scan is completed before resetting ; when it is an M, S, or Dx.y variable, it is reset immediately .

(3) When D is a counter: any counter can be reset. It is often used to reset the main high-speed counter.

3. Program example

After the current value of C255 becomes 400, C255 will immediately reset itself , the current value will be 0, and the output contact will be OFF.



4. Notes

(1) When using the HSCSR instruction, ensure that the counter used is enabled, otherwise the counter value will not change;

(2) The counter responds to the input signal of the counter in an interrupt manner and compares in time. If the matching relationship is satisfied during this comparison, the comparison output is reset immediately.

(3) The comparison output result of the instruction depends only on the comparison result when the pulse is input. Even if the Contents of C235~C255 are rewritten using instructions such as DMOV and DADD, the comparison output will not change if there is no pulse input.

(4) HSCS, HSCR, and HSZ can be used multiple times like ordinary instructions, but the number of instructions driven simultaneously is limited to 6 instructions per high-speed input, and a maximum of 24 instructions for 4 high-speed inputs. However, the HSZ instruction special mode (high-speed table comparison mode, frequency control mode) can only drive 1 instruction at the same time in the ladder diagram, and does not affect the other 6 instructions .

2.7.5 HSZ/High Speed Zone Comparison Instructions

Compare the current value of the high-speed counter with the two interval values , and set the output according to the comparison result . This instruction can be set to high-speed table comparison mode and frequency control mode.

1. Instruction format

FNC55	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
32-bit	17	DZD \	[DHSZ S1 S2 S D]

Operand Description

Operands	Content	Type
S1	Comparison interval lower limit	BIN32 bits
S2	Comparison interval upper limit	BIN32 bits
S	High-speed counter [C235～C255]	BIN32 bits
D	Output comparison results	Bit

Available operand component types

Operands	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
S1								*	*	*	*	*	*	*	*		*	*	*	*			
S2								*	*	*	*	*	*	*	*		*	*	*	*			
S													*					*					
D		*	*			*	*																

2. Function and action description

(1) Normal mode

• S1 is the lower limit of the interval, S2 is the upper limit of the interval, and S1<=S2;

• S must be C235~C255;

• D occupies three consecutive address units. When it is in the port range of Y0~Y17, it is output immediately. Other Y ports will be output after the current user program scan is completed. When it is M, S, Dx.y variable, it is refreshed immediately.

When S1 > S, D is set;

When S1 ≤ S < S2, D+1 is set;

When S ≥ S2, D+2 is set;

(2) High-speed table comparison mode

When parameter D is the special auxiliary relay M8130, the high-speed table comparison mode is specified. In this mode, pay attention to the meaning of the operands.

• S1 represents the starting address of the comparison table, which can only be data register D;

• S2 means the number of rows in the table (only K and H) is limited to K1~K128/H1~H80;

• S must be C235~C255;

• D is M8130, which specifies the high-speed table comparison mode.

The comparison table looks like this:

Comparison data (32 bits)	Output Y number	SET/RST	Table counter (D8130)
S1+1，S1	S1+2	S1+3	0 ↓
S1+5, S1+4	S1+6	S1+7	1 ↓
S1+9, S1+8	S1+10	S1+11	2 ↓
...	...	...	...
S1+(S2*4-3),S1+(S2*4-4)	S1+(S2*4-2)	S1+(S2*4-1)	S2-1 ↓ Repeat from 0

Among them: each row of the table needs to occupy 4 soft elements; the output Y number is specified with a hexadecimal number, Y0~Y7 corresponds to H0~H7, Y10~Y17 corresponds to H10~H17.....Y370~Y377 corresponds to H370~H377; the output Y in the corresponding table is controlled on/off by the set Content 0/1.

Tables are usually constructed using data transfer instructions, such as

(3) Frequency control mode

When parameter D is the special auxiliary relay S M8132, the frequency control mode is specified. The frequency of the high-speed output of the PLSY instruction is controlled according to the table frequency switching. In this mode, pay attention to the meaning of the operand.

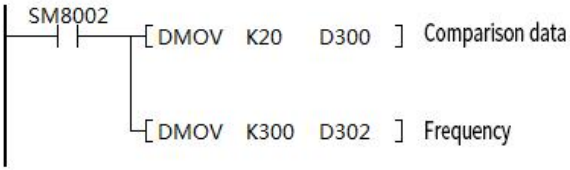
• S1 represents the starting address of the comparison table, which can only be data register D;

• S2 means the number of rows in the table (only K and H) is limited to K1~K128/H1~H80;

• S must be C235~C255;

• D is for S M8132, specifies the frequency control mode.

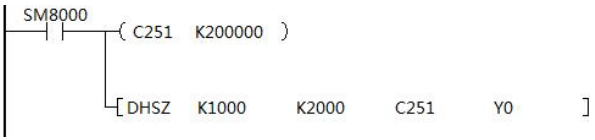
Frequency tables are usually constructed using data transfer instructions, such as



3. Program example

(1) Normal mode

When the current value of high-speed counter C251 changes (counts) as follows, the comparison result is output in any one of Y000, Y001, and Y002.

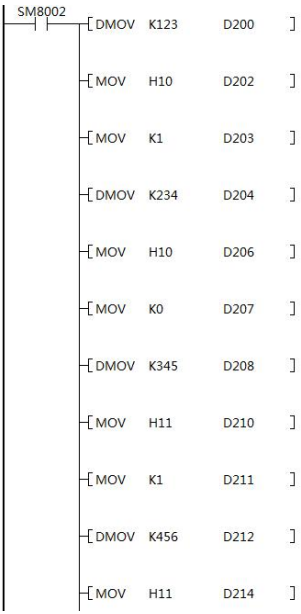


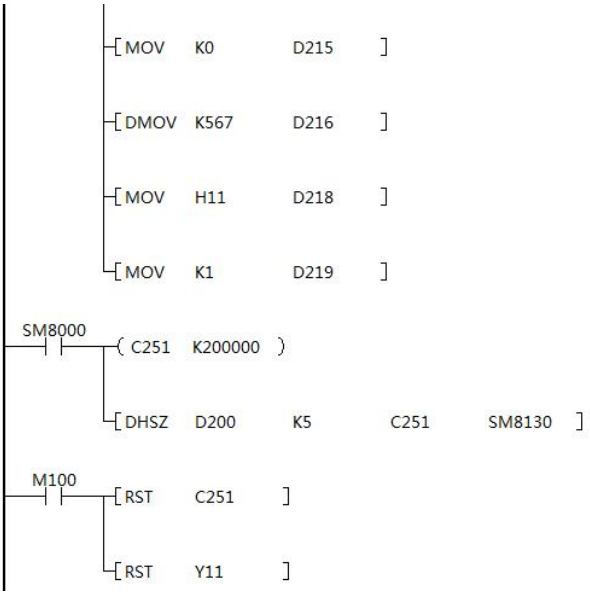
When K1000 > C251, set Y0;

When K1000 ≤ C251 < K2000, set Y1;

When C251 ≥ K2000, set Y2;

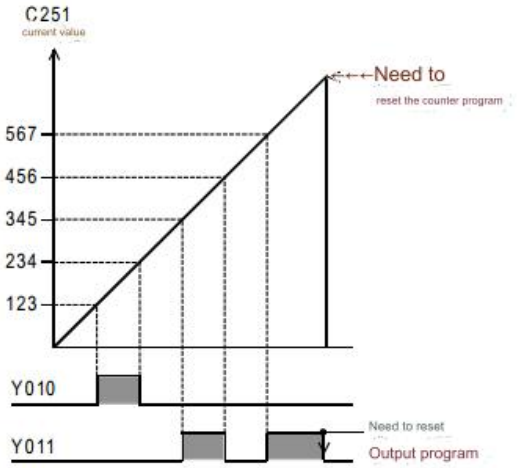
(2) High-speed table comparison mode





The ladder diagram is converted into a comparison table:

Comparison data (32 bits)	Output Y number	SET/RST	Table counter (SD8130)
D201,D200 K123	D202 H10	D203 K1	0 ↓
D205,D204 K234	D206 H1	D207 K0	1 ↓
D209,D208 K345	D210 H11	D211 K1	2 ↓
D213,D212 K456	D214 H11	D215 K0	3 ↓
D217，D216 K567	D218 H11	D219 K1	4 ↓ Repeat from 0



① After executing this instruction, the table of the top row of table data is set as the comparison target data.

② When the current value of high-speed counter C251 matches the table data of the comparison target, the output Y number specified in the table data of the comparison target is SET/RST. And this output Y is directly outputted without waiting for the output refresh of the END instruction.

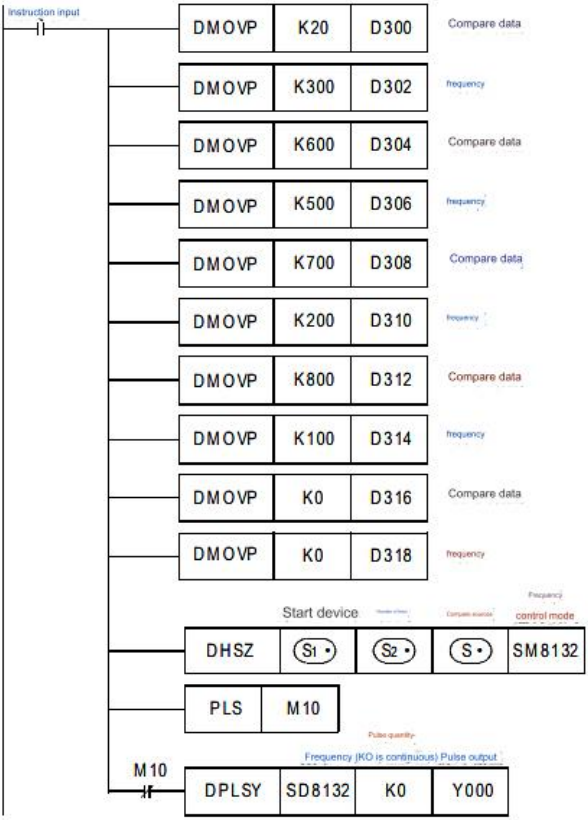
③ The current value of table counter SD8130 is increased by 1.

④ The table data of the comparison target is moved to the table of the next row.

⑤ Repeat the actions of ② to ③ until the current value of table counter SD8130 becomes 4. When the current value becomes 4, it returns to action ① and the table counter is reset (D8130=0). At this time, the end mark flag SM8131 is ON.

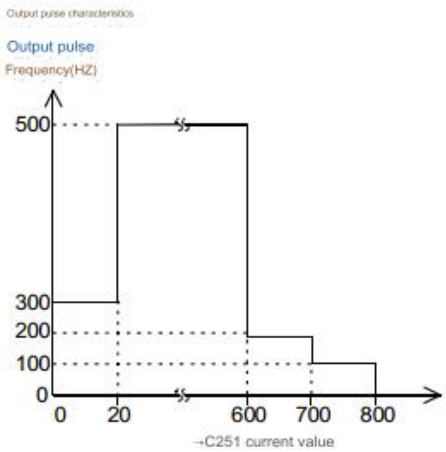
⑥ After the instruction contact is disconnected (instruction execution OFF), the instruction is terminated and the table counter SD8130 is reset (D8130=0).

(3) Frequency control mode



Ladder diagram converted to frequency table:

Comparison data (32 bits)	Output Y number	SET/RST	Table counter (SD8130)
D301,D300 K20	D302, D303 K300	D203 K1	0 ↓
D305,D304 K600	D306, D307 K500	D207 K0	1 ↓
D309,D308 K700	D310, D311 K200	D211 K1	2 ↓
D313,D312 K800	D314, D315 K100	D215 K0	3 ↓
D317, D316 K0	D318, D319 K0	D219 K1	4 ↓ Repeat from 0



① In the data registers that constitute the table, write the specified data in advance, as in the example of this program.

② Until the current value of the high-speed counter specified in S is equal to the value of (D301, D300), the output frequency of the PLSY instruction is the value of (D303, D302). (D302 is the lower 16 bits. D303 is the upper 16 bits, which is generally always 0)

③ After that, enter the action of the second row and execute the actions of each row in sequence.

④ After executing the action of the last row, the end flag M8133 is activated and returns to the first row to repeat the action.

⑤ When you want to stop the action in the last row, set the frequency of the last table to K0.

⑥ After the instruction input is OFF, the pulse output becomes OFF, and the table counter S D8131 is also reset.

⑦ Complete the table creation at the END instruction after the first execution of this instruction, and the instruction will be valid thereafter.

⑧ Therefore, the execution of the PLSY instruction starts from the second scan after the instruction input is ON, so the contact of PLS M10 is used.

4. Precautions

(1) HSCS, HSCR, and HSZ can be used multiple times like ordinary instructions, but the number of instructions driven simultaneously is limited to 6 instructions per high-speed input, and a maximum of 24 instructions for 4 high-speed inputs. However, the HSZ instruction special mode (high-speed table comparison mode, frequency control mode) can only drive 1 instruction at the same time in the ladder diagram, and does not affect the other 6 instructions.

(2) Since the table is not completed until END is executed, the action in special mode starts in the second cycle. In high-speed table mode, the table comparison set or reset output starts in the second scan cycle; in frequency control mode, the second scan cycle PLSY controls the output terminal to output the corresponding frequency according to the frequency table.

(3) In frequency control mode, two pulse outputs cannot be obtained simultaneously by using the PLSY instruction and the PLSR instruction in programming.

(4) In normal mode, even if the high-speed counter is not turned on, the interval comparison will be performed to set the corresponding soft element. In special mode, no comparison will be performed to set the soft element.

(5) Related special registers

SM8130	HSZ high speed table comparison mode
SM8131	High-speed table comparison execution end flag
SM8132	HSZ frequency control mode
SM8133	Frequency table output execution end mark
SD8130	Table counter in HSZ high speed table comparison mode
SD8131	Table counter in HSZ frequency control mode
SD8132	Frequency low bit in HSZ and PLSY frequency control modes
SD8133	Frequency high bit in HSZ and PLSY frequency control modes
SD8134	HSZ, PLSY frequency control mode target pulse number low
SD8135	HSZ, PLSY frequency control mode target pulse number high

2.7.6 SPD/Pulse Density Directive

An instruction that uses interrupt input to count input pulses within a specified time.

1. Instruction format

FNC56	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
16-bit	7	SPD	(D)SPD S1 S2 D
		\	
32-bit	13	DSPD	
		\	

Operand Description

Operands	Content	Type
S1	Input (X) pulse soft element number	Bit
S2	Time (ms) data or word device number for storing data	BIN16/32 bits
D	The starting word device number for storing pulse density data	BIN16/32 bits

Available operand component types

Operands	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
S1	*																						
S2								*	*	*	*	*	*	*	*	*	*	*	*	*			
D												*	*	*	*	*	*	*					

2. Function and action description

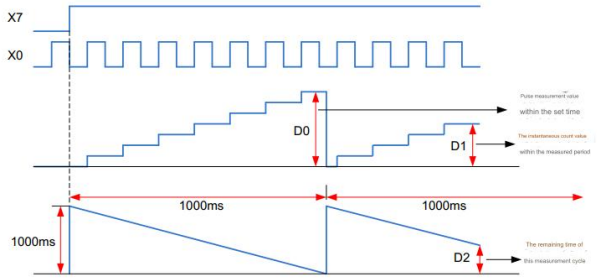
(1) 16-bit instructions

The pulses of input S1 are counted only within the time S2×1ms, and the measured value is saved in D, the current value is saved in D+1, and the remaining time is saved in D+2. Repeating this operation, the pulse density (that is, the value proportional to the rotation speed) can be obtained in the measured value D.

(2) 32-bit instructions

The pulses of input S1 are counted only within (S2+1, S2)×1ms, and the measured value is saved in (D+1, D), the current value is saved in (D+3, D+2), and the remaining time is saved in (D+5, D+4). Repeat this operation to obtain the pulse density (that is, the value proportional to the rotation speed) in the measured value (D+1, D).

3. Program example



When x7 is turned ON, D1 counts the OFF->ON action of x0, and stores the result in D0 after 1000ms. Then D1 is reset and counts the action of X0 again. D2 is used to measure the remaining time. Therefore, the pulse frequency can be obtained according to the setting values of D0 and S2. If the pulse signal is taken from a rotary encoder, the speed can be obtained.

4. Notes

(1) The specified S1 inputs X0 to X3 cannot be used for high-speed counter and input interrupt at the same time.

(2) Only one instruction can be used for each input point.

2.7.7 PLSY/Pulse output instruction

A command for sending a pulse signal.

1. Instruction format

FNC57	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
16-bit	7	PLSY	(D)PLSY S1 S2 D
		\	
32-bit	13	D PLS	
		\	

Operand Description

Operands	Content	Type
S1	Frequency data (Hz) or word device number to store data	BIN16/32 bits
S2	Pulse quantity data or word device number for storing data	BIN16/32 bits
D	Output pulse bit soft element (Y) number	Bit

Available operand component types

Operands	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
S1								*	*	*	*	*	*	*	*	*	*	*	*	*			
S2								*	*	*	*	*	*	*	*	*	*	*	*	*			
D		*																					

2. Function and action description

(1) 16-bit instructions

S2 pulse trains with frequency S1 are output from output Y.

• S1: Frequency, range 1~32767 (HZ).

• S2: Pulse quantity, range 1~32767 (PLS); when S2 is set to 0, pulses can be output infinitely at the frequency of S1.

• D: Y number of pulse output, ranging from Y0 to Y3.

(2) 32-bit instructions

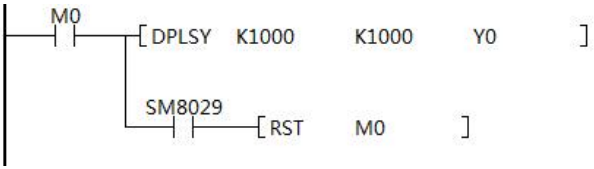
A pulse train (S2+1, S2) with a frequency of (S1+1, S1) is output from output Y.

• (S1+1, S1): frequency, range 1~200000 (HZ).

• (S2+1, S2): Pulse quantity, range 1~2147483647 (PLS); when S2 is set to 0, pulses can be output infinitely at the frequency of S1.

• D: Y number of pulse output, ranging from Y0 to Y3.

3. Program example



Output 1000 pulse trains with a frequency of 1000HZ. After the output is completed, the M8029 instruction ends the action.

4. Notes

(1) Instruction execution end flag and abnormal end flag

If other instructions that change the flag bit and multiple PLSY/PLSR instructions are used, be sure to use them right below the monitoring instruction.

That is, SM8029 or SM8329 must be connected in parallel with the instruction.

(2) Output pulse current value monitoring

Software		Content	Command name
High	Low		
SD8141	SD8140	Y000 output pulse count	PLSY and PLSR instructions count the number of pulses output from Y000
SD8143	SD8142	Y001 output pulse count	PLSY and PLSR instructions count the number of pulses output from Y001
SD8145	SD8144	Y002 output pulse count	PLSY and PLSR instructions count the number of pulses output from Y002
SD8147	SD8146	Y003 output pulse count	PLSY and PLSR instructions count the number of pulses output from Y003

(3) Pulse output status monitoring

SM8340	Y0 pulse output monitoring flag (the flag is set during output)
SM8350	Y1 pulse output monitoring flag (the flag is set during output)
SM8360	Y2 pulse output monitoring flag (the flag is set during output)
SD8370	Y3 pulse output monitoring flag (the flag is set during output)

(4) Stop pulse output

SM8349	Stop Y000 pulse output (immediate stop)
SM8359	Stop Y001 pulse output (immediate stop)
SM8369	Stop Y002 pulse output (immediate stop)
SD8379	Stop Y003 pulse output (immediate stop)

(5) During the execution of the PLSY instruction:

When the data of S1 is changed, the output frequency changes accordingly;

When S2 is changed, the changes will take effect from the next time the command is driven.

(6) The pulse output instruction and positioning instruction that specify the same output port cannot be driven at the same time, otherwise an error 6764 will be reported.

2.7.8 PWM/Pulse Width Modulation Instructions

This is a pulse output instruction that specifies the pulse cycle and ON time.

1. Instruction format

FNC58	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
16-bit	7	PWM \	PWM S1 S2 D

Operand Description

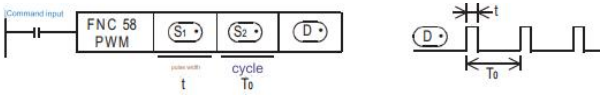
Operands	Content	Type
S1	Pulse width (ms) data or word device number for storing data	BIN16 bits
S2	Cycle (ms) data or word device number for storing data	BIN16 bits
D	Output pulse bit soft element (Y) number	Bit

Available operand component types

Operands	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
S1								*	*	*	*	*	*	*	*	*	*	*	*	*			
S2								*	*	*	*	*	*	*	*	*	*	*	*	*			
D		*																					

2. Function and action description

A pulse with an ON pulse width of S1 (ms) is output in units of a period of S2 (ms).



• S1: Pulse width, range 0~32767ms.

• S2: period, 1~32767ms.

• D: Output pulse number Y0~Y3.

3. Program example

Slightly.

4. Notes

When setting the pulse width and period, S1<=S2.

2.7.9 PLSR/Pulse output instruction with acceleration and deceleration

This is a pulse output instruction that specifies the pulse cycle and ON time.

1. Instruction format

FNC59	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
16-bit	9	PLS R	(D)PLSR S1 S2 S3 D
		\	
32-bit	17	D PLS R	
		\	

Operand Description

Operands	Content	Type
S1	Stores the highest frequency (Hz) data, or the word device number of the data	BIN16/32 bits
S2	Stores the total pulse number (PLS) data or the word device number of the data	BIN16/32 bits
S3	Stores the acceleration/deceleration time (ms) data, or the word device number of the data	BIN16/32 bits
D	Output pulse bit soft element (Y) number	Bit

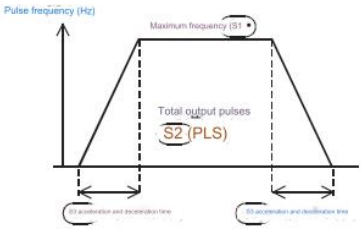
Available operand component types

Operands	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
S1								*	*	*	*	*	*	*	*	*	*	*	*	*			
S2								*	*	*	*	*	*	*	*	*	*	*	*	*			
S3								*	*	*	*	*	*	*	*	*	*	*	*	*			
D		*																					

2. Function and action description

(1) 16-bit instructions

Pulses are output from output Y with a maximum frequency of S1. Acceleration and deceleration are performed in S3 (ms) time and the number of output pulses is only S2.



• S1: Highest frequency, range 10~32767 (HZ).

• S2: Total output pulse number, range 1~32767 (PLS).

• S3: Acceleration and deceleration time, range 50~5000 (ms).

• D: Output pulse number Y0~Y3.

(2) 32-bit instructions

Pulses are output from output Y with a maximum frequency of (S1+1, S1). Acceleration and deceleration are performed in (S3+1, S3) (ms) time, and the number of output pulses is only (S2+1, S2).

• (S1+1, S1): Maximum frequency, ranging from 10 to 200000 (HZ).

• (S2+1, S2): Total output pulse number, range 1~2147483647 (PLS).

• (S3+1, S3): Acceleration and deceleration time, ranging from 50 to 5000 (ms).

• D: Output pulse number Y0~Y3.

3. Program example

The acceleration and deceleration time is 50ms, and 1000 pulse trains with a frequency of 1000HZ are output. After the output is completed, the S M8029 instruction ends the action.

4. Precautions

(1) Instruction execution end flag and abnormal end flag

If other instructions that change the flag bit and multiple PLSY/PLSR instructions are used, be sure to use them right below the monitoring instruction.

That is, SM8029 or SM8329 must be connected in parallel with the instruction.

(2) Output pulse current value monitoring

Software		Content	Command name
High	Low		
SD8141	SD8140	Y000 output pulse count	PLSY and PLSR instructions count the number of pulses output from Y000
SD8143	SD8142	Y001 output pulse count	PLSY and PLSR instructions count the number of pulses output from Y001
SD8145	SD8144	Y002 output pulse count	PLSY and PLSR instructions count the number of pulses output from Y002
SD8147	SD8146	Y003 output pulse count	PLSY and PLSR instructions count the number of pulses output from Y003

(3) Pulse output status monitoring

SM8340	Y0 pulse output monitoring flag (the flag is set during output)
SM8350	Y1 pulse output monitoring flag (the flag is set during output)
SM8360	Y2 pulse output monitoring flag (the flag is set during output)
SD8370	Y3 pulse output monitoring flag (the flag is set during output)

(4) Pulse output stop flag

SM8349	Stop Y000 pulse output (immediate stop)
SM8359	Stop Y001 pulse output (immediate stop)
SM8369	Stop Y002 pulse output (immediate stop)
SD8379	Stop Y003 pulse output (immediate stop)

(5) During the execution of the PLSR instruction: When the values of S1, S2, and S3 are changed during the pulse output, they will only take effect when the next instruction is driven.

(6) The pulse output instruction and positioning instruction that specify the same output port cannot be driven at the same time, otherwise an error 6764 will be reported.

2.6.9 SQR/Square Root Operation Instruction

This is a command for finding the square root (square root sign).

1. Instruction format

FNC48	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
16-bit	5	SQR	[(D)SQR(P) S D]
		SQRP	
32-bit	9	DSQR	
		DSQRP	

Operand Description

Operands	Content	Type
S	Source Data	BIN16/32 bits
D	Target data	BIN16/32 bits

Available operand component types

Operands	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
S														*	*			*	*	*			
D														*	*			*					

2. Function and action description

After calculating the square root of the data in S, save it in D.

3. Program example

Slightly.

4. Notes

(1) When the calculation result is a decimal, the decimal part is discarded and the integer result is filled in D, and S M8021 is turned ON.

(2) When the result is 0, SM8020 turns ON.

2.6.10 FLT/BIN Integer → Binary Floating Point Conversion Instruction

This command converts a BIN integer value into a binary floating point number (real number).

1. Instruction format

FNC49	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
16-bit	5	FTL	[(D)FTL(P) S D]
		FTLP	
32-bit	9	DFTL	
		DFTLP	

Operand Description

Operands	Content	Type
S	BIN integer	BIN16/32 bits
D	Binary floating point number (real number)	Real number (binary)

Available operand component types

Operands	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
S														*	*			*					
D														*	*			*					

2. Function and action description

Convert the BIN integer value data of S into a binary floating point number (real number) and store it in D.

3. Program example

Slightly.

4. Notes

The D in both 16-bit and 32-bit instructions occupies 2 word soft elements.

2.8 Communication instructions

Two communication instructions are used to facilitate the serial port to establish a special communication network, namely RS2 (serial data communication instruction) and ADPRW (MODBUS master communication instruction) ; four instructions are used for the special free communication network of the network port component, namely TCON (free port connection), TSEND (free port send), TRECV (free port receive), and TDCON (free port disconnect).

2.8.1 RS2/Serial Data Transfer 2 Instruction

Refer to section 3.4.2.

2.8.2 ADPRW/MODBUS master communication instructions

Refer to section 3.4.3.

2.8.3 TCON/Ethernet Free Port Connection Instructions

TCON is used to initiate a TCP communication connection from the CPU to a communication partner.

1. Instruction format

	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
16-bit	3	TCON	TCON S

Operand Description

Operands	Content	Type
S	S: Connection status S+1: Connection ID (range 0~7) S+2, S+3: remote IP address S+4: Remote port S+5: Local port	BIN16 bits

Available operand component types

Operands	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
S														*									

2. Function and action description

Once the CPU establishes a connection, it automatically maintains and monitors the connection. To initiate a connection, set the REQ bit in the connection information to TRUE. While the TCON instruction is active, the connection is being initialized, and the Active bit is TRUE, the CPU ignores the REQ bit. After the CPU establishes the connection, the TCON instruction sets the Done bit. If there is a problem with the connection parameters, or if the CPU cannot establish a connection with the remote device, the Error bit is set. If the Error bit is set, the error code indicates why the connection failed.

The TCON instruction is an asynchronous instruction and may take several scans to complete execution. The Active bit is set while a connection operation is pending.

The TCON instruction can create an active (client) connection or a passive (server) connection. An active connection is when the CPU initiates a connection with a remote device. A passive connection is when the CPU waits for the remote device to connect to the CPU.

You can also use the TCON instruction to determine the status of the current connection. If the REQ bit of the TCON instruction is set to FALSE, the CPU reports the connection status when the program calls the instruction:

If the CPU establishes the connection and the connection is available, the instruction sets the "Done" bit (no error).

If the connection is still in progress, the instruction sets the "Active" bit.

If a connection cannot be established, the instruction sets the Done bit and the Error bit. The error code will give the reason why the connection failed.

In the connection information is a level-triggered bit. It is recommended to place a rising edge trigger at the REQ input to initiate the connection, so that the CPU only needs to establish the connection once.

During the connection process (calling the TCON instruction), the program assigns a connection ID to the connection. The connection ID is a 16-bit number selected by the user and passed to the TCON instruction. The connection ID can be any number between 0 and 7. The CPU will not allow the connection ID to be set to any other value. The connection ID value is input to all OUC instructions to identify the connection to be used for a given operation.

You can choose the connection ID value to make it more logical based on the number of OUC connections you actually allocate . Please note that after the connection is closed, there is no automatic attempt to reconnect to the device. After the connection is broken, your program must execute another TCON instruction to reconnect to the device. This is true for both active and passive connections.

Connection status operand S is defined as:

Position 15	Positions 14 to 9	Bit 8	Bit 7	Position 6	Bit 5	Bit 4~0
A/P	reserve	REQ	D	A	E	Error Code

A/P: Active/Passive connection, 1 is active, 0 is passive

Reserved: Temporarily unused

REQ: You can use the REQ bit to initiate a new operation. The REQ bit is a level-triggered value. If necessary, the program code must provide this single-step operation (rising edge contact). If the operation is not busy, a new operation will be initiated when the REQ value is TRUE. For example: If there is no TSEND instruction currently being executed, the REQ bit being TRUE will cause the program to initiate a new TSEND instruction operation

D: Command completed

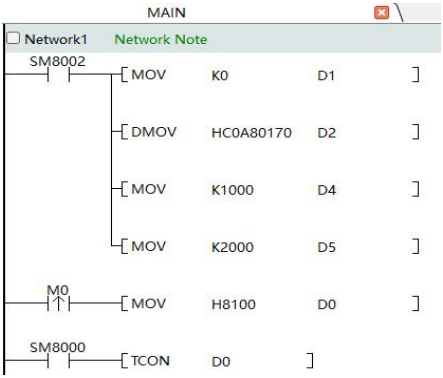
A: Command activation

E: Command error (command completed with errors)

Error Code: If an error occurs, both the Done bit and the Error bit are set

Error code description: See section 2.8.7.

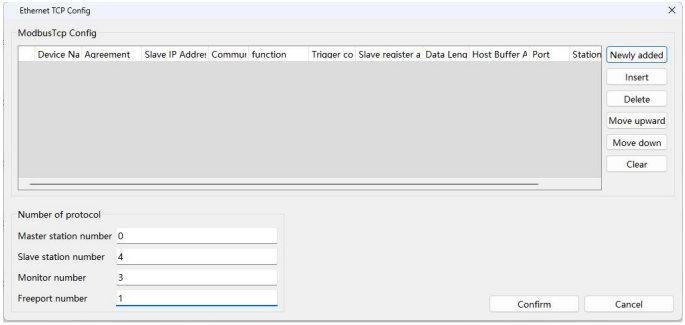
3. Program example



Trigger M0, TCON instruction will initiate an active connection request. If passive connection is used, it can be written as MOV H100 D0

4. Notes

(1) Before using the free port communication command, you must first configure the number of free ports in the Ethernet configuration.



(2) TCON is successful when DONE is set and there are no errors.

2.8.4 T SEND/Ethernet free port send command

The TSEND instruction can be used to send data via an existing communication connection.

1. Instruction format

	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
16-bit	3	TSEND	TSEND S

Operand Description

Operands	Content	Type
S	S: Send status S+1: Connection ID (range 0~7) S+2: length of data sent S+3~S+4: Data storage area address	BIN16 bits

Available operand component types

Operands	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
S														*									

2. Function and action description

If REQ is set and the connection is not currently in use by another operation, the TSEND instruction begins sending the specified number of bytes when your program calls it.

The REQ bit is level-triggered. It is recommended to place a rising edge trigger at the REQ input to initiate the connection so that the CPU does not accidentally initiate a send operation. When TSEND is "Active", the CPU ignores the REQ bit. The status bits and error code show the status of TSEND at each call:

Completed without error means that the TSEND instruction is completed without error.

• Active means that the TSEND instruction is still busy.

Completed with Error means that a problem occurred in TSEND. The error code contains the cause of the failure.

• the completion/activation/error status of each TSEND instruction call is displayed . A maximum of 1024 bytes of data can be sent in one message. Only one TSEND can be active at a time in a given connection. When the TSEND instruction is executed with REQ set, the program copies the data from the send buffer in the user memory to the internal buffer, so that you can modify the send buffer after the TSEND instruction is executed.

Send status operand S definition:

Positions 15 to 9	Bit 8	Bit 7	Position 6	Bit 5	Bit 4~0
reserve	REQ	D	A	E	Error Code

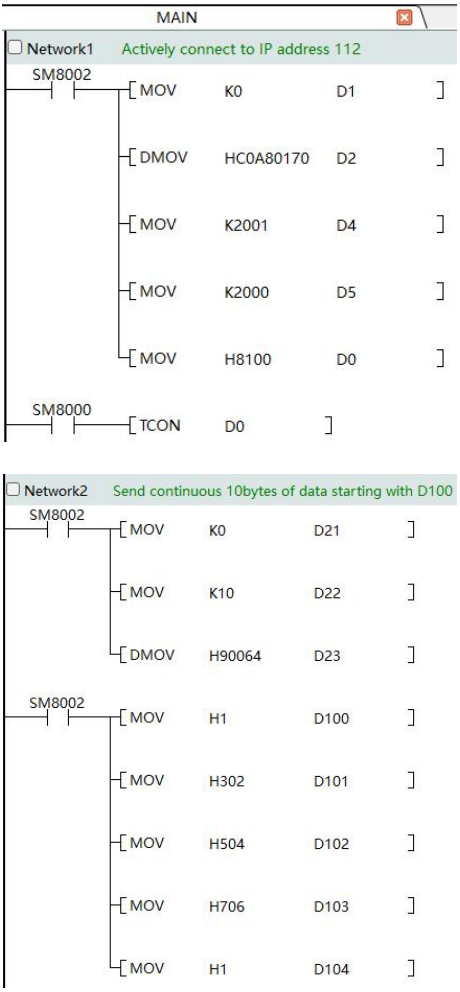
Error code description: See section 2.8.7.

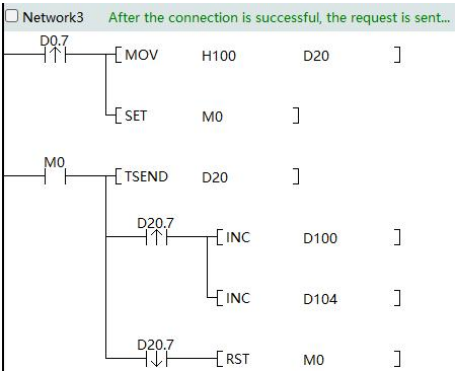
Data storage area address description:

Bits 31 to 16	Bit 15~0
Storage area register type (T/C/D/R), all are 32-bit registers 0x6:T 0x7:C 0x9:D 0x11:R	Storage area location offset

Example: H90064 means the storage area is D100; H117530 means the storage area is R30000

3. Program example





After the TCON instruction establishes the connection, it sets the REQ (request) bit of TSEND, and then uses the Done (completed) bit of TSEND to determine whether the transmission is completed. The next data can be sent only after the transmission is completed.

4. Notes

(1) Before using the free port communication command, you must first configure the number of free ports in the Ethernet configuration.

(2) Note that the length of the data sent cannot exceed the available address length of the available buffer.

(3) After TSEND is executed, it may take multiple scan cycles to send out data. The Done signal is used to determine whether the current TSEND instruction has been sent. This bit will be automatically set when the sending is completed and will be automatically cleared before the next sending. However, it is important to distinguish the situations when errors occur. When an error occurs, both the Error bit and the Done bit will be set.

2.8.5 T RECV/Ethernet Free Port Receive Command

Data received by the CPU over an existing communication connection can be retrieved using the TRECV instruction .

1. Instruction format

	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
16-bit	3	TRECV	TSECV S

Operand Description

Operands	Content	Type
S	S: receiving status S+1: Connection ID (range 0~7) S+2: The length of data allowed to be received S+3: The actual length of the received data S+4~S+5: Data storage area address	BIN16 bits

Available operand component types

Operands	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
S														*									

2. Function and action description

The TRECV instruction has no REQ bit based on the selected connection protocol. After the first execution of the TRECV instruction, the status bits show that the instruction is "Active". If the CPU does not receive data for this connection, all subsequent calls to the TRECV instruction show the "Active" status.

After successfully receiving the data, the instruction sets the "Done" bit of the status byte in the table, and the returned data length value is the number of bytes actually received. Only when the TRECV instruction is executed and the "Done" bit is TRUE, the TRECV instruction will copy the received data from the internal buffer to your receive buffer , and a maximum of 1024 bytes of data can be received. If the number of bytes received by the CPU exceeds the capacity of the user buffer, the TRECV instruction will copy the maximum number of bytes allowed (the data length in the table) and discard the other received bytes. In this case, an error message appears after the TRECV instruction is executed to remind the user that the bytes are discarded.

The receive status operand S is defined as:

Positions 15 to 8	Bit 7	Position 6	Bit 5	Bit 4~0
reserve	D	A	E	Error Code

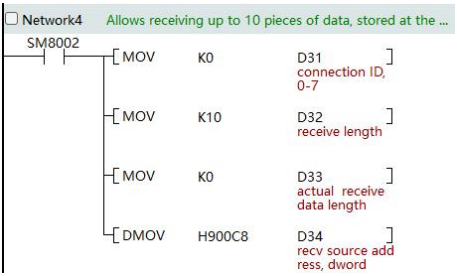
Error code description: See section 2.8.7.

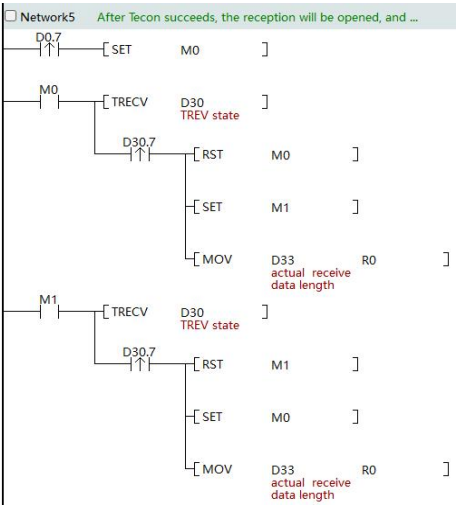
Data storage area address description:

Bits 31 to 16	Bit 15~0
Storage area register type (T/C/D/R), all are 32-bit registers 0x6:T 0x7:C 0x9:D 0x11:R	Storage area location offset

Example: H90064 means the storage area is D100; H117530 means the storage area is R30000

3. Program example





4. Notes

(1) Before using the free port communication command, you must first configure the number of free ports in the Ethernet configuration.

(2) When the received data exceeds the available buffer, the data will only be saved according to the available cache length. In addition, the buffer address should not be occupied by other instructions as much as possible to avoid inaccurate data.

(3) If you want to receive data of different lengths, you can adjust the parameter of the data length allowed to be received according to your needs.

(4) S+2 is the length of data allowed to be received, and S+3 is the actual length of data received. If the actual length is greater than the allowed length, the data will be cached according to the allowed length.

(5) When the instruction is turned on, data is not necessarily received. Therefore, the actual length of the received data can only be read when the reception is completed and DONE is set.

2.8.6 T DCON/Ethernet free port disconnect command

You can use the TDCON instruction to terminate an existing communication connection.

1. Instruction format

	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
16-bit	3	TDCON	TDCON S

Operand Description

Operands	Content	Type
S	S: disconnected state S+1: Connection ID (range 0~7)	BIN16 bits

Available operand component types

Operands	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
S														*									

2. Function and action description

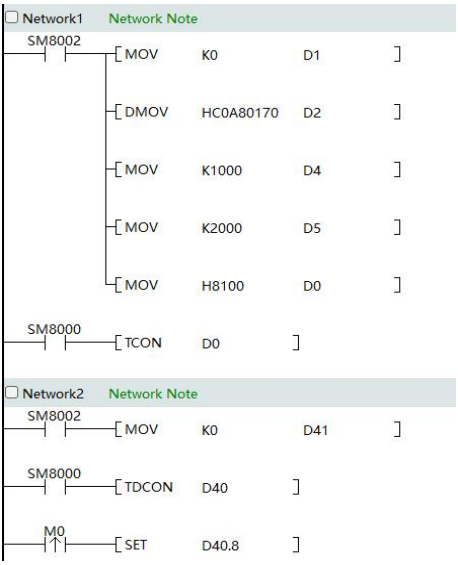
When REQ is set, the instruction terminates the connection. It is recommended to place a rising edge trigger on the REQ input.

Definition of disconnect state operand S:

Positions 15 to 9	Bit 8	Bit 7	Position 6	Bit 5	Bit 4~0
reserve	REQ	D	A	E	Error Code

Error code description: See section 2.8.7.

3. Program example



Setting M0 to enable the REQ bit of TDCON will disconnect the TCON connection.

4. Notes

Before using the free port communication command, you must first configure the number of free ports in the Ethernet configuration.

2.8.7 Ethernet Freeport Communication Error Codes

Error Code	Describe	TCON	TSEND	TRECV	TDCON
0	No Error	X	X	X	X
1	The data length parameter is greater than the maximum allowed length (1024 bytes) or is set to zero.		X	X	
2	The data buffer is not in the valid storage area		X	X	
3	The length of the sent data exceeds the valid storage area		X		
8	The connection ID does not exist because the connection was never created, or the connection was terminated at your request (using the TDCON instruction).	X	X	X	X
9	A TCON operation using this connection ID is in progress		X	X	X

Error Code	Describe	TCON	TSEND	TRECV	TDCON
10	A TDCON operation using this connection ID is in progress	X	X	X	
13	The connection partner refuses or actively disconnects the connection (the partner will disconnect from this CPU)	X	X	X	
15	The connection is suddenly interrupted or the network cable is disconnected				X
17	No connection resources are available. All connections of the requested type (active/passive) are in use	X			
21	Invalid connection ID	X			
25	Receive buffer too small: The number of bytes received by the CPU exceeds the length supported by the buffer. The CPU discards the extra bytes			X	
31	Unknown error	X	X	X	X

2.9 Floating point instructions

Used for floating-point data analysis and calculation with high precision requirements, including basic floating-point arithmetic, floating-point conversion, trigonometric functions, etc.

2.9.1 ECMP/Binary Floating Point Comparison Instructions

compares two floating-point data values and outputs the result (greater than, equal to, or less than) to a bit device (3 points).

1. Instruction format

FNC110	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
32-bit	13	DECMP DECMPP	DECMP(P) S1 S2 D

Operand Description

Operands	Content	Type
S1	Comparison value 1	Real number (binary)
S2	Comparison value 2	Real number (binary)
D	Comparison results	Bit

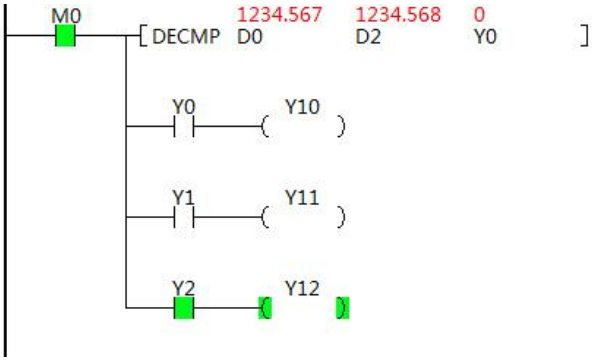
Available operand component types

Operands	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
S1														*	*			*	*	*	*		
S2														*	*			*	*	*	*		
D		*	*			*	*											*					

2. Function and action description

32-bit operation instruction, compares comparison value S1 and comparison source S2 as floating point data , and then turns on any one of D, D+1, and D+2 according to the comparison result. When constants (K, H) are specified in S1 and S2, the value is automatically converted from BIN to binary floating point before processing.

3. Program example



When the instruction is executed, the 32-bit floating point number represented by D0 is less than D2, so Y2 is set and Y12 is output;

If D0 = D2, Y1 is set and output is Y11;

If D0 > D2, Y0 is set and output is Y10.

4. Notes

(1) D occupies 3 points, (D, D+1, D+2). Be careful not to overlap with soft elements for other purposes.

(2) Even if the command input is OFF and the DECMP instruction is not executed, D to D+2 can maintain the status before the command input is OFF.

2.9.2 EZCP/Binary floating point number interval comparison instruction

compares the floating-point numbers in two intervals with the floating-point comparison source and outputs the result to the bit device (3 points).

1. Instruction format

FNC111	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
32-bit	17	DEZCP DEZCPP	DEZCP(P) S1 S2 D

Operand Description

Operands	Content	Type
S1	Left interval floating point number	Real number (binary)
S2	Right interval floating point number	Real number (binary)
S	Floating point comparison source	Real number (binary)
D	The starting soft element number of the output result (occupies 3 points)	Bit

Available operand component types

Operands	Bit Components							Word component								Address Indexing		constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
S1														*	*			*	*	*	*		
S2														*	*			*	*	*	*		
S														*	*			*	*	*	*		
D		*	*			*	*											*					

2. Function and action description

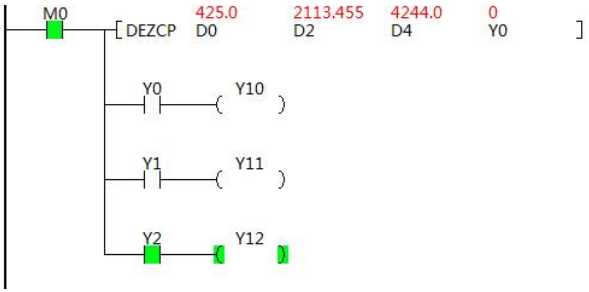
32-bit operation instruction, compares comparison values S1, S2 and comparison source S as floating-point data.

When S1 > S, D is set;

D+1 is set when S1 ≤ S ≤ S2;

D+2 is set when S2 < S.

3. Program example



When the instruction is executed, the comparison source D4 is greater than the right interval floating point number, so Y2 is set and output Y12;

If D0 > D4, Y0 is set and Y10 is output;

If D0 ≤ D4 ≤ D2, Y1 is set and output is Y11.

4. Notes

(1) S1 ＞ When S2 is used, the value of S2 is considered to be the same as S1 for comparison.

(2) ZCP command is not executed , D to D+2 can maintain the status before the command input is OFF.

2.9.3 EMOV/Binary floating point number transfer instruction

This command transmits binary floating point data.

1. Instruction format

FNC112	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
32-bit	9	DEMOV DEMOVP	[DEMOV(P) S D]

Operand Description

Operands	Content	Type
S	Transmission Source	Real number (binary)
D	Target data	Real number (binary)

Available operand component types

Operands	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
S														*	*			*			*		
D														*	*			*					

2. Function and action description

32-bit operation instruction transfers the Contents of the transfer source S (binary floating point data) to D. In addition, a real number (E) can be directly specified in S.

3. Program example

When M0 is ON, the real number - 123.456 is saved in the 32-bit floating point number represented by D0 , (D0, D1) = 1.23456

4. Notes

Note the difference from the normal MOV instruction.

2.9.4 ESTR/Binary floating point number to string conversion instruction

This instruction is used to convert binary floating point data into a character string (ASCII code) of specified digits.

1. Instruction format

FNC116	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
32-bit	13	DESTR DESTRP	DESTR(P) S1 S2 D

Operand Description

Operands	Content	Type
S1	Transfer source (binary floating point number)	Real number (binary)
S2	Specify the starting number of the device to be converted	BIN16 bits
D	Target data (stores the converted string)	String

Available operand component types

Operands	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
S1														*	*			*			*		
S2								*	*	*	*	*	*	*	*			*					
D									*	*	*	*	*	*	*			*					

2. Function and action description

The 32-bit operation instruction converts the Content of (S1+1, S1) (binary floating point data) into a character string according to the Content specified in S2, S2+1, S2+2, and saves it to the soft element starting with D.

S2 indicates the conversion format, and the converted data result is related to the specified display of S2. When S2 is 0, it indicates the decimal point format conversion, and when it is 1, it indicates the exponential format conversion;

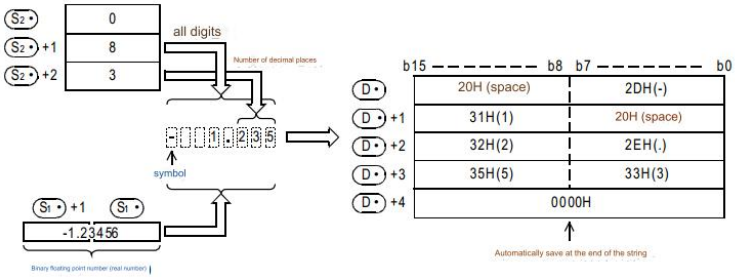
S2+1 represents all digits;

S2+2 represents the number of decimal places.

3. Program example

(1) Decimal point format

For example, when the total number of digits is 8 and the number of digits in the decimal part is 3, and -1.23456 is specified, the following will be stored in the device starting with D.



• In characters, "20H" (space) is stored when the binary floating point data is positive, and "2DH" (-) is stored when it is negative.

• If the decimal part of the binary floating point data cannot fit within the range of the decimal digits, the lower decimal part is rounded off.

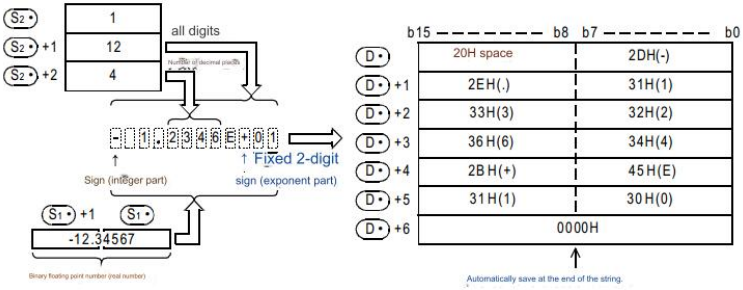
• When the number of decimal places is set to a number other than "0", "2EH" (.) is automatically stored in the place where the number of decimal places is the specified number of places + 1. However, when the number of decimal places is "0", "2EH" (.) is not stored.

• When the number of digits after removing the sign, decimal point, and fractional digits from all digits is greater than the integer part of the binary floating point data, "20H" (space) is stored between the sign and the integer part.

• At the end of the converted character string, "00H" or "0000H" is automatically stored.

(2) Exponential form

For example, if the total number of digits is 12 and the number of digits in the decimal part is 4, and -12.34567 is specified, the following will be stored in the device starting with D.



• In the sign of the integer part, "20H" (space) is stored when the binary floating point data is positive, and "2DH" (-) is stored when it is negative.

• The integer part is fixed to 1 digit. "20H" (space) is stored between the integer part and the sign.

• If the decimal part of the binary floating point data cannot be accommodated within the range of the decimal digits, the lower decimal part is rounded off.

• When the number of decimal places is set to a number other than "0", "2EH" (.) is automatically stored in the place where the number of decimal places is the specified number of places + 1. However, when the number of decimal places is "0", "2EH" (.) is not stored.

• In the sign of the exponent part, "2BH" (+) is stored when the exponent is positive, and "2DH" (-) is stored when the exponent is negative.

• The exponent part is fixed to 2 digits. When the exponent part is 1 digit, "30H" (0) is stored between it and the sign of the exponent part.

• At the end of the converted character string, "00H" or "0000H" is automatically stored.

4. Notes

Operation errors may occur in the following situations:

(1) When S1 is not in the range of 0, ±2 -126 ≤ S1＜±2 128.

(2) When the format specified in S2 is other than 0 or 1.

(3) When all the digits specified in S2+1 do not fit within the following range

Decimal point format:

When the number of decimal places is "0", all digits are ≥ 2;

when the number of decimal places is a number other than "0", all digits are ≥ (the number of decimal places + 3).

Exponential format:

When the number of decimal places is "0", all digits are ≥ 6;

when the number of decimal places is a number other than "0", all digits are ≥ (the number of decimal places + 7).

(4) When the number of decimal places specified in S2+2 does not fit within the following range

In decimal point format: The number of decimal places ≤ (total number of digits - 3).

In exponential format: The number of decimal places ≤ (total number of digits - 7).

(5) When the range of the device storing the specified character string exceeds the range of the corresponding device.

(6) When the conversion result exceeds the number of digits specified.

2.9.5 EVAL/string → binary floating point number conversion instruction

This command is used to convert a character string (ASCII code) into binary floating point data.

1. Instruction format

FNC117	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
32-bit	9	DEVAL DEVALP	DEVAL(P) S D

Operand Description

Operands	Content	Type
S	Transfer source (string data)	String
D	Target data (binary floating point data)	Real number (binary)

Available operand component types

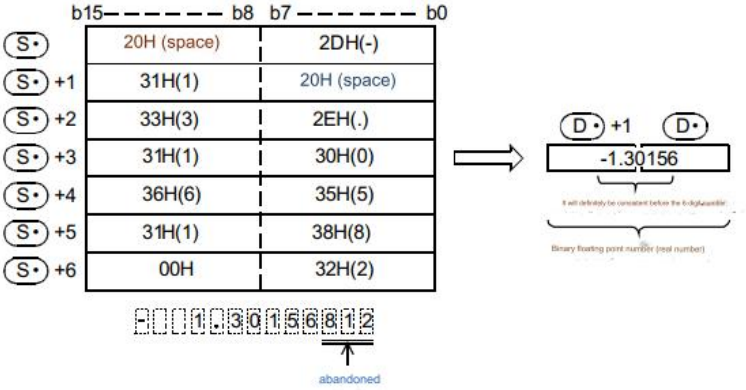
Operands	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
S								*	*	*	*	*	*	*	*			*					
D														*	*			*					

2. Function and action description

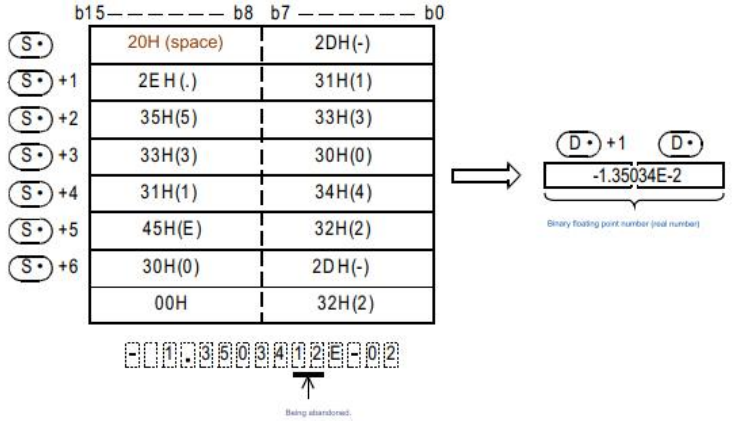
32-bit operation instruction, converts the string stored in the soft element starting with S into binary floating point data, and stores it in (D+1, D). The specified string can be converted into binary floating point data regardless of whether it is in decimal point form or exponential form.

3. Program example

(1) Decimal point format



(2) Exponential form



• If the string specified by S to be converted into a binary floating-point number still has 7 or more digits after removing the sign, decimal point, and exponent, the digits after the 7th digit will be discarded.

• In decimal form, if the sign is specified as "2BH" (+) or the sign is omitted, it is converted as a positive value. Also, if the sign is specified as "2DH" (-), it is converted as a negative value.

• In exponential form, if the sign of the exponent part is specified as "2BH" (+) or the sign is omitted, it is converted as a positive exponent. If the sign of the exponent part is specified as "2DH" (-), it is converted as a negative exponent.

• In the specified character string, if "20H" (space) or "30H" (0) exists between the numerical values other than the initial "0", "20H" or "30H" is ignored and the conversion is performed.

• If "30H" (0) exists between "E" and the value in the exponential form character string, "30H" is ignored and conversion is performed.

• The maximum length of a character string can be 24 characters. "20H" (space) and "30H" (0) in the character string are also counted as one character.

4. Notes

(1) Related software components

Software	Name	Content	
		Condition	Action
M8020	Zero	The conversion result is truly zero (when the mantissa is "0")	The zero position flag (M8020) is ON.
M8021	Borrow	The absolute value of the conversion result is less than 2^-126	The part of D that is less than the minimum value of a 32-bit real number (2^-126) is discarded and the borrow flag (M8021) is ON.
M8022	Carry	The absolute value of the conversion result ≧ 2^128	The value greater than the maximum value (2^128) of a 32-bit real number is discarded and the carry flag (M8022) is ON.

(2) In the following cases, an operation error will occur, and error 6706 will be reported.

• When there are characters other than "30H" (0) to "39H" (9) in the integer part or the decimal part.

• When there are two or more "2EH" (.) in the character string specified by S. • When there are characters other than "45H" (E), "2BH" (+), "2DH" (-) in the exponent part, or when there are multiple exponent parts.

• When there is no "00H" in the corresponding soft element range starting from S.

• When the number of characters after S is 0 or exceeds 24 characters.

2.9.6 EBCD/Binary Floating Point Number → Decimal Floating Point Number Conversion Instructions

This command converts a binary floating point number in a device into a decimal floating point number.

1. Instruction format

FNC118	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
32-bit	9	DEBCD DEBCDP	DEBCD(P) S D

Operand Description

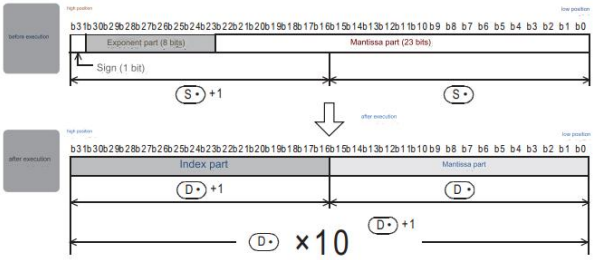
Operands	Content	Type
S	Source data (binary data)	Real number (binary)
D	Target data (decimal floating point data)	Real number (decimal)

Available operand component types

Operands	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
S														*	*			*					
D														*	*			*					

2. Function and action description

32-bit operation instruction, the binary floating point number (S+1, S) is converted into a decimal floating point number and saved in (D+1, D).



3. Program example

Slightly.

4. Notes

Floating point calculations are all performed as binary floating point numbers. However, since binary floating point numbers are difficult to understand (a special monitoring

method), converting them to decimal floating point numbers makes it easier to monitor them on peripheral devices.

2.9.7 EBIN/Decimal floating point number → binary floating point number conversion instruction

This command converts a decimal floating point number in a software component into a binary floating point number.

1. Instruction format

FNC119	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
32-bit	9	DEBIN DEBINP	DEBIN(P) S D

Operand Description

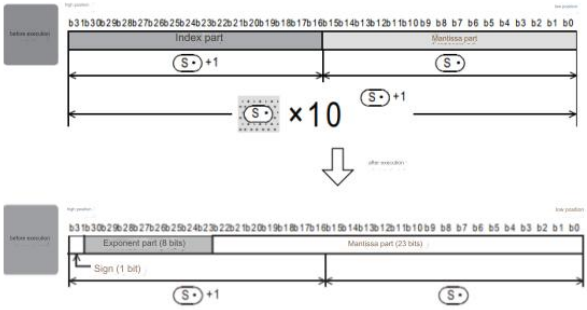
Operands	Content	Type
S	The data register number for storing decimal floating point data	Real number (decimal)
D	The data register number to store the converted binary floating point data	Real number (binary)

Available operand component types

Operands	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
S														*	*			*					
D														*	*			*					

2. Function and action description

32-bit operation instruction, the decimal floating point number (S+1, S) is converted into a binary floating point number and saved in (D+1, D).



3. Program example

Slightly.

4. Notes

Note the difference between the storage method of binary floating point numbers and decimal numbers.

2.9.8 EADD/Binary Floating Point Addition Instruction

Two instructions for adding binary floating-point numbers.

1. Instruction format

FNC120	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
32-bit	\	DEADD DEADDP	DEADD(P) S1 S2 D

Operand Description

Operands	Content	Type
S1	The word device number for storing the binary floating point data to be added	Real number (binary)
S2	The word device number for storing the binary floating point data to be added	Real number (binary)
D	Save the binary floating point data after addition operation	Real number (binary)

Available operand component types

Operands	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
S2														*	*			*	*	*	*		
S2														*	*			*	*	*	*		
D														*	*			*					

2. Function and action description

32-bit operation instruction, adds the binary floating point data of (S1+1, S1) and (S2+1, S2), and transfers the result of the operation to (D+1, D) in the form of binary floating point. When constants (K, H) are specified in (S1+1, S1) and (S2+1, S2), the values are automatically converted into binary floating point numbers.

3. Program example

Slightly.

4. Notes

None.

2.9.9 ESUB/Binary floating point subtraction instruction

Two instructions for subtracting binary floating-point numbers.

1. Instruction format

FNC121	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
32-bit	13	DESUB DESUBP	DESUB(P) S1 S2 D

Operand Description

Operands	Content	Type
S1	The word device number for storing the binary floating point data to be subtracted	Real number (binary)
S2	The word device number for storing the binary floating point data to be subtracted	Real number (binary)
D	Save the binary floating point data after subtraction operation	Real number (binary)

Available operand component types

Operands	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
S1														*	*			*	*	*	*		
S2														*	*			*	*	*	*		
D														*	*			*					

2. Function and action description

32-bit operation instruction, subtracts the binary floating point data of (S2+1, S2) from (S1+1, S1) , and transfers the operation result to (D+1, D) in the form of binary floating point. When constants (K, H) are specified in (S1+1, S1) and (S2+1, S2) , the values are automatically converted into binary floating point numbers.

3. Program example

Slightly.

4. Notes

None.

2.9.10 EMUL/Binary floating point multiplication instruction

Two instructions for binary floating-point multiplication.

1. Instruction format

FNC122	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
32-bit	13	DEMUL	⊣[DEMUL(P) S1 S2 D]
		DEMULP	

Operand Description

Operands	Content	Type
S1	The word device number for storing the binary floating point data for multiplication	Real number (binary)
S2	The word device number to store the binary floating point data to be multiplied	Real number (binary)
D	Save the binary floating point data after multiplication	Real number (binary)

Available operand component types

Operands	Bit Components							Word component								Address Indexing		constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
S1														*	*			*	*	*	*		
S2														*	*			*	*	*	*		
D														*	*			*					

2. Function and action description

32-bit operation instruction, multiplies the binary floating point data of (S1+1, S1) and (S2+1, S2) , and transfers the operation result to (D+1, D) in the form of binary floating point . When constants (K, H) are specified in (S1+1, S1) and (S2+1, S2) , the values are automatically converted to binary floating point numbers.

3. Program example

Slightly.

4. Notes

When the operation result exceeds the single-precision floating-point range, the corresponding flag bit is set

Software	Name	Content	
		Condition	Action
SM8020	Zero	The conversion result is truly zero (when the mantissa is "0")	The zero position flag (SM8020) is ON.
SM8021	Borrow	The absolute value of the conversion result is less than 2^-126	The part of D that is less than the minimum value of a 32-bit real number (2^-126) is discarded and the borrow flag (SM8021) is ON.
SM8022	Carry	The absolute value of the conversion result ≧ 2^128	The value greater than the maximum value (2^128) of a 32-bit real number is discarded and the carry flag (SM8022) is ON.

2.9.11 EDIV/Binary floating point division instruction

2 instructions for binary floating point division operations.

1. Instruction format

FNC123	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
32-bit	9	DEDIV DEDIVP	DEDIV(P) S1 S2 D

Operand Description

Operands	Content	Type
S1	The word device number for storing the binary floating point data for division operation	Real number (binary)
S2	The word device number for storing the binary floating point data to be divided	Real number (binary)
D	Save the binary floating point data after the division operation	Real number (binary)

Available operand component types

Operands	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
S1														*	*			*	*	*	*		
S2														*	*			*	*	*	*		
D														*	*			*					

2. Function and action description

32-bit operation instruction, divides the binary floating point data of (S1+1, S1) and (S2+1, S2) , and transfers the operation result to (D+1, D) in the form of binary floating point . When constants (K, H) are specified in (S1+1, S1) and (S2+1, S2) , the values are automatically converted to binary floating point numbers.

3. Program example

Slightly.

4. Notes

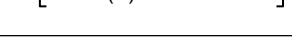
When the calculation result exceeds the single-precision floating-point range, the corresponding flag bits S M8020, S M8021, and S M8022 are activated.

Software	Name	Content	
		Condition	Action
S M8020	Zero	The conversion result is truly zero (when the mantissa is "0")	The zero position flag (SM8020) is ON.
S M8021	Borrow	The absolute value of the conversion result is less than 2^-126	The part of D that is less than the minimum value of a 32-bit real number (2^-126) is discarded and the borrow flag (SM8021) is ON.
S M8022	Carry	The absolute value of the conversion result ≧ 2^128	The value greater than the maximum value (2^128) of a 32-bit real number is discarded and the carry flag (SM8022) is ON.

2.9.12 EXP/Binary floating point exponential operation instruction

This instruction is an exponential operation instruction with e(2.71828) as the base.

1. Instruction format

FNC124	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
32-bit	9	DEXP DEXPP	

Operand Description

Operands	Content	Type
S	The starting number of the device that stores binary floating-point data for exponential calculation	Real number (binary)
D	The starting number of the device that stores the calculation results	Real number (binary)

Available operand component types

Operands	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
S														*	*			*			*		
D														*	*			*					

2. Function and action description

32-bit operation instruction, uses (S+1, S1) as the exponent to perform the operation and saves the operation result to (D+1, D) . In addition, a real number can be directly specified in S.

e (S+1,S) = (D+1,D)

3. Program example

Slightly.

4. Notes

The result is not within the range of 2 -126 ≦ |operation result| < 2 128 , and operation error 6706 is reported.

2.9.13 LOGE/ Binary floating point natural logarithm operation instruction

This instruction performs the natural logarithm operation.

1. Instruction format

FNC125	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
32-bit	9	DLOGE DLOGEP	-[DLOGEP(P) S D]

Operand Description

Operands	Content	Type
S	The starting number of the device that stores binary floating-point data for natural logarithm calculation	Real number (binary)
D	The starting number of the device that stores the calculation results	Real number (binary)

Available operand component types

Operands	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
S														*	*			*			*		
D														*	*			*					

2. Function and action description

32-bit operation instruction, performs the natural logarithm (logarithm with e(2.17828) as the base) operation of (S+1, S) , and saves the operation result to (D+1, D) . In addition, a real number can be directly specified in S.

3. Program example

Slightly.

4. Notes

The value specified in (S+1, S) can only be a positive number. If a negative number or 0 is specified, error 6706 will be reported.

2.9.14 LOG10/ Common logarithmic operation instructions for binary floating point numbers

This instruction performs a common logarithm operation.

1. Instruction format

FNC126	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
32-bit	9	DLOG10 DLOG10P	DLOG10(P) S D

Operand Description

Operands	Content	Type
S	The starting number of the device that stores binary floating-point data for common logarithm calculations	Real number (binary)
D	The starting number of the device that stores the calculation results	Real number (binary)

Available operand component types

Operands	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
S														*	*			*			*		
D														*	*			*					

2. Function and action description

32-bit operation instruction, performs the common logarithm (logarithm with base 10) operation of (S+1, S) , and saves the operation result to (D+1, D) . In addition, real numbers can be directly specified in S.

3. Program example

Slightly.

4. Notes

The value specified in (S+1, S) can only be a positive number. If a negative number or 0 is specified, error 6706 will be reported.

2.9.15 ESQR/Binary floating point square root calculation instruction

This is an instruction for calculating the square root of a binary floating point number.

1. Instruction format see

FNC127	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
32-bit	9	DESQR DESQRP	DESQR(P) S D

Operand Description

Operands	Content	Type
S	The starting number of the device that stores the binary floating-point data for performing square root calculations	Real number (binary)
D	The starting number of the device that stores the calculation results	Real number (binary)

Available operand component types

Operands	Bit Components							Word component								Address Indexing		constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
S														*	*			*			*		
D														*	*			*					

2. Function and action description

32-bit operation instruction, after executing (S+1, S) to perform square root operation (binary floating point operation), the operation result is saved in (D+1, D) .

3. Program example

Slightly.

4. Notes

(1) (S+1, S) is valid if it is a positive number. If it is a negative number, error 6706 will be reported.

(2) When the operation result is 0, SM8020 is set.

Software	Name	Content
SM8020	Zero	ON when the calculation result is 0

2.9.16 ENEG/Binary floating point sign flip instruction

This is a command that inverts the sign of binary floating point (real number) data.

1. Instruction format

FNC128	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
32-bit	5	DENEG DENEGP	DENEG(P) D

Operand Description

Operands	Content	Type
D	The starting number of the device that stores the binary floating point data for which the sign is to be reversed.	Real number (binary)

Available operand component types

Operands	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
D														*	*			*					

2. Function and action description

32-bit operation instruction, flip the sign of the binary floating point data at (D+1, D) and store it in (D+1, D) .

3. Program example

Slightly.

4. Notes

None.

2.9.17 INT/Binary floating point number → BIN integer conversion instruction

This command converts a binary floating point number into a BIN integer, which is the general data format in a programmable controller (binary floating point number → BIN integer).

1. Instruction format

FNC129	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
16-bit	5	INT INTP	(D)INTP(P) S D
32-bit	9	DINT DINTP	

Operand Description

Operands	Content	Type
S	The data register number to store the binary floating point data to be converted into a BIN integer	Real number (binary)
D	The data register number to store the converted BIN integer	BIN16/32 bits

Available operand component types

Operands	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
S														*	*			*					
D														*	*			*					

2. Function and action description

The binary floating point number of S is converted into a BIN integer and then transferred to (D+1, D) .

3. Program example

Slightly.

4. Notes

(1) Actions of related flags

Software	Name	Content
SM8020	Zero	Turns on when the calculation result is 0.
SM8021	Borrow	When a bit is converted and a value is discarded because it is less than 1, it turns ON.
SM8022	Carry	This turns on when the operation result exceeds the range of -32,768 to 32,767 (16-bit operation) or -2,147,483,583 to 2,147,483,583 (32-bit operation) and overflow occurs. (The operation result is not reflected.)

(2) Values after the decimal point are discarded

2.9.18 SIN/Binary Floating Point SIN Operation Instructions

This command calculates the SIN value of an angle (RAD).

1. Instruction format

FNC130	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
32-bit	9	DSIN DSINP	DSIN(P) S D

Operand Description

Operands	Content	Type
S	The device number for storing the binary floating point value RAD (angle)	Real number (binary)
D	The device number that stores the SIN value of a binary floating point number	Real number (binary)

Available operand component types

Operands	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
S														*	*			*			*		
D														*	*			*					

2. Function and action description

32-bit operation instruction, converts the angle value (binary floating point number) specified in (S+1, S) into a SIN value and transmits it to (D+1, D) .

3. Program example

Slightly.

4. Notes

When the operation result is 0, SM8020 is set.

RAD value = angle*π/180.

2.9.19 COS/Binary floating point COS operation instruction

This command calculates the COS value of an angle (RAD).

1. Instruction format

FNC131	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
32-bit	9	DCOS DCOSP	DCOS(P) S D

Operand Description

Operands	Content	Type
S	The device number for storing the binary floating point value RAD (angle)	Real number (binary)
D	The device number that stores the COS value of a binary floating point number	Real number (binary)

Available operand component types

Operands	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
S														*	*			*			*		
D														*	*			*					

2. Function and action description

32-bit operation instruction, converts the angle value (binary floating point number) specified in (S+1, S) into a COS value and transmits it to (D+1, D) .

3. Program example

Slightly.

4. Notes

RAD value = angle*π/180.

2.9.20 TAN/Binary floating point TAN operation instruction

This command calculates the TAN value of an angle (RAD).

1. Instruction format

FNC132	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
32-bit	9	DTAN	DTAN(P) S D
		DTANP	

Operand Description

Operands	Content	Type
S	The device number for storing the binary floating point value RAD (angle)	Real number (binary)
D	The device number that stores the TAN value of a binary floating point number	Real number (binary)

Available operand component types

Operands	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
S														*	*			*			*		
D														*	*			*					

2. Function and action description

32-bit operation instruction, converts the angle value (binary floating point number) specified in (S+1, S) into a TAN value and transmits it to (D+1, D) .

3. Program example

Slightly.

4. Notes

When the operation result is 0, M8020 is set.

RAD value = angle*π/180.

2.9.21 ASIN/Binary Floating Point Number SIN-1 Operation Instructions

Perform SIN -1 operation.

1. Instruction format

FNC133	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
32-bit	9	DASIN DASINP	DASIN(P) S D

Operand Description

Operands	Content	Type
S	The starting number of the device that stores the SIN value for performing the SIN -1 (arc sine) operation	Real number (binary)
D	The starting number of the device that stores the calculation results	Real number (binary)

Available operand component types

Operands	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
S														*	*			*			*		
D														*	*			*					

2. Function and action description

32-bit operation instruction, calculate the angle from the SIN value of (S+1, S) , and save the operation result to (D+1, D) . In addition, a real number can be directly specified in S.

3. Program example

Slightly.

4. Notes

(1) The SIN value of (S+1, S) is set in the range of -1.0 to 1.0. If it is not in this range, error 6706 is reported.

(2) The angle stored in (D+1, D) is in radians (-π/2) to (π/2).

2.9.22 ACOS/Binary floating point COS-1 operation instruction

perform COS $^{-1}$ operation.

1. Instruction format

FNC134	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
32-bit	9	DACOS DACOSP	-[DACOS(P) S D]

Operand Description

Operands	Content	Type
S	The starting number of the device that stores the COS value for executing the COS -1 (inverse cosine) operation	Real number (binary)
D	The starting number of the device that stores the calculation results	Real number (binary)

Available operand component types

Operands	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
S														*	*			*			*		
D														*	*			*					

2. Function and action description

32-bit operation instruction, calculate the angle from the COS value of (S+1, S) , and save the operation result to (D+1, D) . In addition, a real number can be directly specified in S.

3. Program example

Slightly.

4. Notes

(1) The COS value of (S+1, S) is set in the range of -1.0 to 1.0. If it is not within this range, error 6706 is reported.

(2) The angle stored in (D+1, D) is a radian value from 0 to π.

2.9.23 ATAN/Binary floating point number TAN-1 operation instruction

performing TAN -1 operations.

1. Instruction format

FNC135	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
32-bit	9	DATAN DATANP	DATAN(P) S D

Operand Description

Operands	Content	Type
S	The starting number of the device that stores the TAN value for performing TAN -1 (inverse tangent) calculation	Real number (binary)
D	The starting number of the device that stores the calculation results	Real number (binary)

Available operand component types

Operands	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
S														*	*			*			*		
D														*	*			*					

2. Function and action description

32-bit operation instruction, calculate the angle from the TAN value of (S+1, S) , and save the operation result to (D+1, D) . In addition, a real number can be directly specified in S.

3. Program example

Slightly.

4. Notes

The angle stored in (D+1, D) is in radians > (-π/2) or < (π/2)=

2.9.24 RAD/Binary floating point angle to radian conversion command

Command to convert the value in angular units to radian units.

1. Instruction format

FNC136	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
32-bit	9	DRAD DRADP	DRAD(P) S D

Operand Description

Operands	Content	Type
S	The starting number of the device that stores the angle to be converted into radians	Real number (binary)
D	The starting number of the device that stores the calculation results	Real number (binary)

Available operand component types

Operands	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
S														*	*			*			*		
D														*	*			*					

2. Function and action description

32-bit operation instruction, (S+1, S) converts the unit from angle unit to radian unit, and saves the operation result in (D+1, D) . In addition, a real number can be directly specified in S. Radian unit = angle unit*π/180.

3. Program example

Slightly.

4. Notes

None.

2.9.25 DEG/Binary floating point radians to angles conversion command

This command converts the value in radians to degrees (DEG).

1. Instruction format

FNC137	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
32-bit	9	DDEG DDEGP	DDEG(P) S D

Operand Description

Operands	Content	Type
S	The starting number of the device that stores the radians to be converted into angle units	Real number (binary)
D	The starting number of the device that stores the calculation results	Real number (binary)

Available operand component types

Operands	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
S														*	*			*			*		
D														*	*			*					

2. Function and action description

32-bit operation instruction, the unit of (S+1, S) is converted from radian unit to angle unit, and the operation result is saved in (D+1, D) .

3. Program example

Slightly.

4. Notes

None.

2.10 Data processing 2 instructions

2.10.1 WSUM/Calculate total value of data command

This instruction can calculate the total value of continuous 16-bit or 32-bit data.

1. Instruction format

FNC140	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
16-bit	7	WSUM	-[(D)WSUM(P) S D n]
		WSUMP	
32-bit	13	DWSUM	
		DWSUMP	

Operand Description

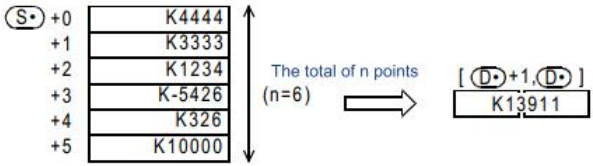
Operands	Content	Type
S	The starting number of the device that stores the data to be totaled	BIN16/32 bits
D	The starting number of the device to store the total value	BIN32/64 bit
n	Number of data (0＜n)	BIN16/32 bits

Available operand component types

Operands	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
S												*	*	*	*			*					
D												*	*	*	*			*					
n														*	*				*	*			

2. Function and action description

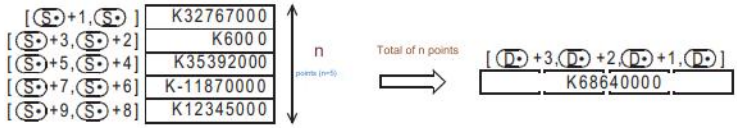
(1) 16-bit instructions



(D+1, D) as 32-bit data .

(2) 32-bit instructions

The total value of the 32-bit data of n points starting from (S+1, S) is stored in (D+3, D+2, D+1, D) as 64-bit data.



3. Program example

slightly.

4. Notes

(1) When the number of data is set to ≤ 0, error 6706 is reported.

(2) Pay attention to the calculation result. For 16-bit instructions, if the number exceeds the 16-bit number, check D+1, where D is the high 16 bits and low 16 bits of the calculation result. If it does not exceed the 16-bit number, just check D. For 32-bit instructions, if the number exceeds the 16-bit number, check (D+3, D+2), where (D+1, D) is the high 32 bits and low 32 bits of the calculation result. If it does not exceed the 16-bit number, just check (D+1, D).

2.10.2 WTOB/Byte-based data separation instruction

This command separates continuous 16-bit data into byte (8-bit) units.

1. Instruction format

FNC141	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
16-bit	7	WTOB	WTOB(P) S D n
		WTOBP	

Operand Description

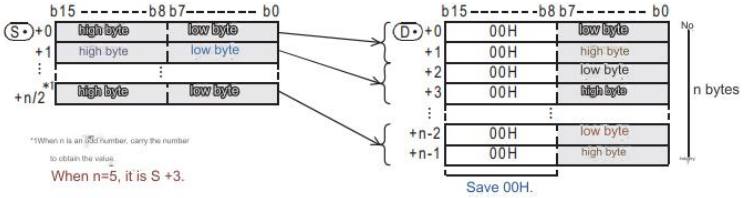
Operands	Content	Type
S	The starting number of the device that stores the data to be separated in byte units	BIN16 bits
D	The starting number of the device that stores the result of separation in byte units	BIN16 bits
n	The number of bytes to be separated (0≦n)	BIN16 bits

Available operand component types

Operands	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
S												*	*	*	*			*					
D												*	*	*	*			*					
n														*	*				*	*			

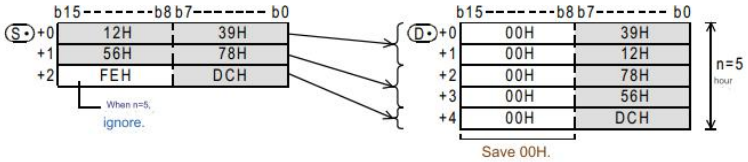
2. Function and action description

(1) 16-bit instruction: Separate the 16-bit data stored in n/2 devices starting with S into n bytes and store them in n-point devices starting with D as shown below.



(2) 00H is stored in the high byte (8 bits) of the device (after D) that stores the separated byte data.

(3) When n is an odd number, as shown in the figure below, in the final data of the separation source, only the low byte (8 bits) is the object data.



3. Program example

Slightly.

4. Notes

(1) When the separation source soft element S to (S+n/2) exceeds the soft element range of the specified soft element, and n is an odd number, the soft element that occupies the number of values after the carry is required, and error 6706 is reported.

(2) When the soft elements D to (D+n-1) storing the separated data exceed the soft element range of the specified soft element, error 6706 is reported.

(3) When n=0, the instruction is not executed.

2.10.3 BTOW/Byte Unit Data Combine Instruction

This instruction combines the lower 8 bits (low byte) of consecutive 16-bit data.

1. Instruction format

FNC142	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
16-bit	7	BTOW	⊣[BTOW(P) S D n]
		BTOWP	

Operand Description

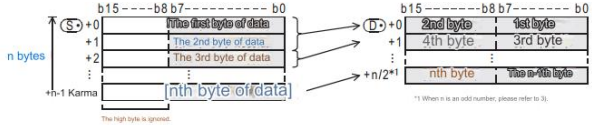
Operands	Content	Type
S	The starting number of the device that stores the data to be combined in byte units	BIN16 bits
D	The starting number of the device that stores the result of combining in byte units	BIN16 bits
n	The number of byte data to be combined (0≦n)	BIN16 bits

Available operand component types

Operands	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
S												*	*	*	*			*					
D												*	*	*	*			*					
n														*	*				*	*			

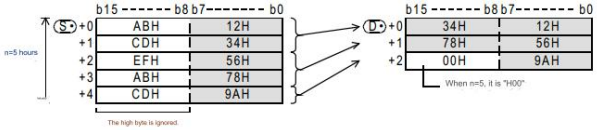
2. Function and action description

(1) 16-bit instruction: The 16-bit data obtained by combining the low bytes (8 bits) of the n-point 16-bit data starting with S is saved in the n/2-point device starting with D as shown below.



(2) The high byte (8 bits) of the 16-bit data (after S) of the combined source is ignored.

(3) When n is an odd number, as shown in the figure below, the high byte (8 bits) of the final combined data is 00H.



3. Program example

Slightly.

4. Notes

(1) If the device specified in S to (S+n-1) of the join source device exceeds the device range, error 6706 is displayed.

(2) When the soft element D to (D +n/2) storing the combined data exceeds the soft element range of the specified soft element. When n is an odd number, the soft element that occupies the number of the value after the carry is required, and error 6706 is reported.

(3) When n=0, the instruction is not executed.

2.10.4 UNI/4-bit combine instruction for 16-bit data

This instruction combines the lower 4 bits of consecutive 16-bit data.

1. Instruction format

FNC143	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
16-bit	7	UNI	[UNI(P) S D n]
		UNIP	

Operand Description

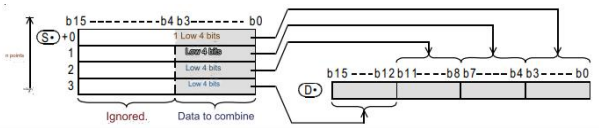
Operands	Content	Type
S	The starting number of the device that stores the data to be combined	BIN16 bits
D	The device number that stores the combined data	BIN16 bits
n	Number of combinations (0 to 4, no processing when n=0)	BIN16 bits

Available operand component types

Operands	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
S												*	*	*	*			*					
D												*	*	*	*			*					
n														*	*				*	*			

2. Function and action description

16-bit instruction, combine the lower 4 bits of the 16-bit data of n points starting from S and save it to D as shown below.



3. Program example

After M0 is turned on, the lower 4 bits of D0~D2 are combined and saved in D3 .

soft components	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0	value
D0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1
D1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	5
D2	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	9
D3	0	0	0	0	1	0	0	1	0	1	0	1	0	0	0	1	2385

4. Notes

Note that the combination will always be performed when the instructions are executed continuously.

2.10.5 DIS/4-bit separation instruction for 16-bit data

This instruction combines the lower 4 bits of consecutive 16-bit data.

1. Instruction format

FNC144	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
16-bit	7	DIS	DIS(P) S D n
		DISP	

Operand Description

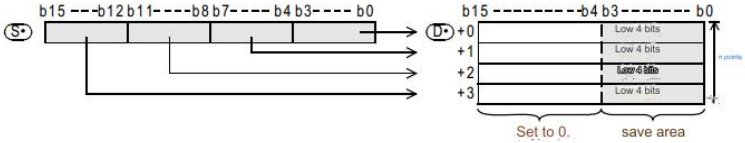
Operands	Content	Type
S	The starting number of the device that stores the data to be separated	BIN16 bits
D	The device number to store the separated data	BIN16 bits
n	Separation number (0 to 4, no processing when n=0)	BIN16 bits

Available operand component types

Operands	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
S												*	*	*	*			*					
D												*	*	*	*			*					
n														*	*				*	*			

2. Function and action description

(1) 16-bit instruction: The 16-bit data starting from S is separated into 4-bit units and stored in D as shown below.



(2) n range (0 to 4, no processing when n=0)

(3) The upper 12 bits of the n-point devices starting from D are set to 0.

3. Program example

Slightly.

4. Notes

(1) When the n-point soft elements starting from D exceed the soft element range, error 6706 is reported.

(2) If n is an unexpected number between 0 and 4, error 6706 is reported.

2.10.6 SWAP/High and Low Byte Swap Instruction

This instruction swaps the upper 8 bits and lower 8 bits of word data.

1. Instruction format

FNC147	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
16-bit	3	SWAP	[(D)SWAP(P) S]
		SWAPP	
32-bit	5	DSWAP	
		DSWAPP	

Operand Description

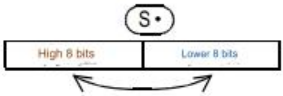
Operands	Content	Type
S	Word soft element with high and low bytes swapped	BIN16 bit/32 bit

Available operand component types

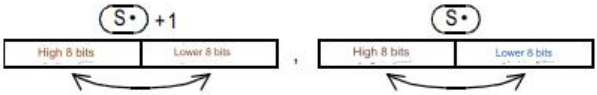
Operands	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
S									*	*	*	*	*	*	*	*	*	*					

2. Function and action description

(1) 16-bit instructions



(2) 32-bit instructions



3. Program example

Slightly.

4. Notes

Even for 32-bit instructions, the lower 8 bits and upper 8 bits are swapped, which is the same as the extended function of the XCH instruction.

2.10.7 SORT2/Data Sort 2 Instruction

This command re-sorts the data table consisting of data (rows) and group data (columns) in ascending/descending order based on the specified group data (column) in units of

rows. In this command, since data is stored in continuous soft elements (row direction), it is easy to add data (rows).

1. Instruction format

FNC149	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
16-bit	11	SORT2	(D)SORT2 S m1 m2 D n
		\	
32-bit	twenty one	DSORT2	
		\	

Operand Description

Operands	Content	Type
S	The starting number of the soft element to save the data table [occupies m1×m2 points]	BIN16 bit/32 bit
m1	Number of data (rows) [1 to 32]	BIN16 bit/32 bit
m2	Number of group data (columns) [1～6]	BIN16 bit/32 bit
D	The starting number of the soft element that stores the calculation results [occupies m1×m2 points]	BIN16 bit/32 bit
n	The column number of the group data (column) used as the sorting standard [1 to m2]	BIN16 bit/32 bit

Available operand component types

Operands	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
S														*	*								
m1														*	*				*	*			
m2																			*	*			
D														*	*								
n														*	*				*	*			

2. Function and action description

(1) 16-bit instructions

For the data table of the starting (m1×m2) point (before sorting), the data rows are arranged in ascending or descending order based on the group data of n columns, and then saved in the data table of the starting (m1×m2) point (after sorting).

(2) 32-bit instructions

For the data table of (m1×m2) points starting from (S+1,S) (before sorting), the data rows are arranged in ascending or descending order based on the group data of n columns, and then saved in the data table of (m1×m2) points starting from (D+1,D) (after sorting).

3. Program example

In the case of "n=K2 (column number 2)" and "n=K3 (column number 3)", the data before sorting shown below is sorted, and the operation is as follows. The following is an example of 16-bit operation. When performing 32-bit operation, use BIN32-bit to form the data table. In addition, if you enter a continuous number such as a management number in the first column first, you can judge the original row number based on its Content, which is very convenient.

Data before sorting:

Column Number Line Number		Number of groups: m2 (when m2=K4)			
		1	2	3	4
		Management Number	height	weight	age
When the number of data m1=K5	1	S	S+1	S+2	S+3
		1	150	45	20
	2	S+4	S+5	S+6	S+7
		2	180	50	40
	3	S+8	S+9	S+10	S+11
		3	160	70	30
	4	S+12	S+13	S+14	S+15
		4	100	20	8
	5	S+16	S+17	S+18	S+19
		5	150	50	45

(1) The sorting result when executing the command based on n=K2 (column number 2) (in ascending order)

Column Number Line Number		Number of groups: m2 (when m2=K4)			
		1	2	3	4
		Management Number	height	weight	age
When the number of data m1=K5	1	D	D+1	D+2	D+3
		4	100	20	8
	2	D+4	D+5	D+6	D+7
		1	150	45	20
	3	D+8	D+9	D+10	D+11
		5	150	50	45
	4	D+12	D+13	D+14	D+15
		3	160	70	30
	5	D+16	D+17	D+18	D+19
		2	180	50	40

(2) The sorting result when executing the command based on n=K3 (column number 3) (in descending order)

Column Number Line Number		Number of groups: m2 (when m2=K4)			
		1	2	3	4
		Management Number	height	weight	age
When the number of data m1=K5	1	D	D+1	D+2	D+3
		3	160	70	30
	2	D+4	D+5	D+6	D+7
		2	180	50	40
	3	D+8	D+9	D+10	D+11
		5	150	50	45
	4	D+12	D+13	D+14	D+15
		1	150	45	20
	5	D+16	D+17	D+18	D+19
		4	100	20	8

4. Notes

(1) Operation of related software components

S M8029: instruction end flag, set when the sequence is completed;

S M8165: Arrange in descending order when ON, and in ascending order when OFF.

(2) The following operations will cause incorrect sorting, please note

• Do not change the operands and data Contents during the operation.

• When executing again, please input OFF the instruction once.

• The number of times the instruction can be used is limited to a maximum of 2 times in the program.

• The original data and the data after sorting should be staggered and not overlapped.

2.11 Positioning instructions

X3 PLC supports positioning instructions such as PLSV, DRVI, DRVA, DVIT, ZRN, DSZR , and ZRN2 . 32-point PLC supports 4-way Y000~Y003, and 60-point PLC supports 6-way Y000~Y005. The pulse and direction output by the positioning instruction are connected to the servo controller to drive the motor to work, control the fixed length and direction of the motor rotation, connect the light sensor on the axis table to the PLC input terminal, and the input signals include forward and reverse limit, interrupt input, near point signal, and zero point signal.

1. I/O point allocation

(1) Input point allocation

Use	Input number	illustrate
Forward limit	All body input points	Please connect to any input terminal If the wired input is ON, the drive forward limit flag is
Reversal Limit	All body input points	Please connect to any input terminal If the wired input is ON, the drive reverse limit flag
Interrupt input	X000~X005	Through the interrupt input designation function, the interrupt input designation register SD8336 can be used to specify the X000~X005 input number corresponding to the pulse output terminal, or to specify the user interrupt instruction flag bit
Near point signal	Support XYMS and other soft components	It is recommended to use the X terminal of the main body
Zero signal	All body input points	It is recommended to use the main body high-speed input point

(2) Output point allocation

Use	Input number	illustrate
Pulse signal	Y000~Y005	Please set the positioning command to the pulse output terminal Y000~Y005 wiring
Direction signal	All main output points	Support any output point of the main body
Clear signal	All main output points	If the clear signal designation function is valid, the clear signal soft element designation register can be used to designate any output corresponding to the pulse output terminal.

2. Positioning instruction related soft components

(1) Special auxiliary relay

Soft component number						Name	Property	Object directives
Y000	Y001	Y002	Y003	Y004 (60 points)	Y005 (60 points)			
SM8029						Instruction execution end flag	Read-only	PLSY/PLSR /ZRN2 /DSZR /DVIT/ZRN /PLSN /DRVI PLSF /DRVA
SM8329						Instruction execution abnormal end flag	Read-only	PLSY/PLSR/DSZR/DVIT /ZRN/ ZRN2/ PLSV/ PLSF/PLSN/ DRVI/DRVA
SM8338						Acceleration and deceleration action	Read/Write	PLSV
SM8340	SM8350	SM8360	SM8370	SM8380	SM8390	Pulse output monitoring	Read-only	PLSY PLSR/ PLSN/ DSZR/DVIT/ZRN/ ZRN2/ PLSV/DRVI/DRVA

Soft component number						Name	Property	Object directives
Y000	Y001	Y002	Y003	Y004 (60 points)	Y005 (60 points)			
SM8341	SM8351	SM8361	SM8371	SM8381	SM8391	Clear signal output function is effective	Read/Write	DSZR/ZRN /ZRN2
SM8342	SM8353	SM8362	SM8372	SM8382	SM8392	Origin return start direction specification	Read/Write	ZRN/ZRN2/ DR
SM8343	SM8353	SM8363	SM8373	SM8383	SM8393	Forward limit	Read/Write	PLSY/PLSR/DSZR/DVIT/ZRN/ ZRN2/ PLSV/ PLSN/ PLSF/ DRVI/ DRVA
SM8344	SM8354	SM8364	SM8374	SM8384	SM8394	Reversal Limit	Read/Write	PLSY/PLSR/DSZR/DVIT/ZRN/ ZRN2/ PLSV/ PLSN/ PLSF/ DRVI/ DRVA
SM8345	SM8355	SM8365	SM8375	SM8385	SM8395	Proximity signal logic inversion	Read/Write	ZRN/ DSZR
SM8346	SM8356	SM8366	SM8376	SM8386	SM8396	Zero signal logic inversion	Read/Write	ZRN2/ DSZR
SM8347	SM8357	SM8367	SM8377	SM8387	SM8397	Interrupt signal logic inversion	Read/Write	DVIT
SM8348	SM8358	SM8368	SM8378	SM8388	SM8466	Positioning command driving	Read-only	PLSY/PWM/ PLSR/ PLSN/PLSF/ DSZR/DVIT/ZRN/ ZRN2/ PLSV/DRVI/DRVA
SM8349	SM8359	SM8369	SM8379	SM8389	SM8399	Pulse stop command (the signal needs to be maintained for 1ms)	Read/Write	PLSY/PLSR/DSZR/DVIT/ZRN/ ZRN2/ PLSV/ PLSN/ PLSF/ DRVI/ DRVA
SM8460	SM8461	SM8462	SM8463	SM8464	SM8465	User interrupt input instruction	Read/Write	DVIT

(2) Special data registers

Soft component number						name	Data length (Bit)	Initial Value	Object directives
Y000	Y001	Y002	Y003	Y004	Y005				
SD8336 (32 points)						Interrupt input assignment	16	—	DVIT
SD8336, SD8337 (60 points)							32		
SD8340 (Low)	SD83 50 (Low)	SD8360 (Low)	SD8370 (Low)	SD8380 (Low)	SD8390 (Low)	Current register value (PLS)	32	0	DSZR/DVI T /ZRN/ ZRN2/ PLSV/ PLSN/PLSF/ DRVI/DRVA
SD8341 (High)	SD8351 (High)	SD8361 (High)	SD8371 (High)	SD8381 (High)	SD8391 (High)				
SD8342	SD8352	SD8362	SD8372	SD8382	SD8392	Base speed (HZ)	16	0	DSZR/DVI T /ZRN/ ZRN2/ PLSV/ PLSN/PLSF/ DRVI/DRVA

Soft component number						name	Data length (Bit)	Initial Value	Object directives
Y000	Y001	Y002	Y003	Y004	Y005				
SD8343 (Low)	SD8353 (Low)	SD8363 (Low)	SD8373 (Low)	SD8383 (Low)	SD8393 (Low)	Top speed (HZ)	32	200000	DSZR/DVI T /ZRN/ ZRN2/ PLSV/ PLSN/PLSF/ DRVI/DRVA
SD8344 (High)	SD8354 (High)	SD8364 (High)	SD8374 (High)	SD8384 (High)	SD8394 (High)				
SD8345	SD8355	SD8365	SD8375	SD8385	SD8395	Crawling speed (HZ)	16	1000	ZRN/ZRN2/ DSZR
SD8346 (Low)	SD8356 (Low)	SD8366 (Low)	SD8376 (Low)	SD8386 (Low)	SD8396 (Low)	Origin return speed (HZ)	32	50000	ZRN/ZRN2/ DSZR
SD8347 (High)	SD8357 (High)	SD8367 (High)	SD8377 (High)	SD8387 (High)	SD8397 (High)				
SD8348	SD8358	SD8368	SD8378	SD8388	SD 8097	Acceleration time (ms)	16	100	DSZR/DVI T /ZRN/ ZRN2/ PLSV/ PLSN/PLSF/ DRVI/DRVA
SD8349	SD8359	SD8369	SD8379	SD8389	SD 8098	Deceleration time (ms)	16	100	DSZR/DVI T /ZRN/ ZRN2/ PLSV/ PLSN/PLSF/ DRVI/DRVA
SD8464	SD8465	SD8466	SD8467	SD8468	SD8469	Clear signal device designation	16	—	ZRN/ZRN2/ DSZR

3. Special instructions:

◆DRVI, DRVA, DVIT are similar. DRVI and DVIT are relative positions, DRVA is absolute position. If an interrupt signal is input before the instruction is closed, DVIT is equivalent to DRVI. However, this situation only supports user input interrupt mode. , does not support X input interrupt mode.

◆It should be noted that the positioning instructions that specify the same output port cannot be driven at the same time, including high-speed instructions PLSY, PLSR, and PWM. If driven at the same time, error 6764 will be reported.

◆If the same input port is specified, input interrupt, high-speed counter, SPD, DVIT, pulse capture, ZRN, DSZR cannot be specified at the same time. Considering that the output pulse driving the motor to rotate may be too small, there is no limit on the number of pulses in all positioning instructions to travel the longest distance possible.

◆The pulse and direction output of all positioning instructions do not support addressing

◆The two axes share the same origin signal or DOG signal, which has no impact and no error.

◆DOG signal and interrupt signal share the same terminal and no error is reported.

◆For safety reasons, except for the return to origin command, all other positioning commands are changed to stop immediately at the left and right extreme positions.

◆The length of the DOG block does not allow to cover both the origin and the DOG switch.

Special attention:

The data length in the above table is 32 bits. For example, SD8340 and SD8341 are combined into a 32-bit register (current value register). When monitoring SD8340 in the monitoring table on the host computer, please select DWORD, otherwise you can only see the value of 16-bit WORD.

The relationship between each instruction and speed:

PLSV: The sign determines the direction. The speed takes an absolute value, which can be 0. If it is greater than the maximum speed, it is the maximum speed; if it is less than the minimum speed, it is the minimum speed .

DRVA/DRVI/DVIT: If it is 0, it will be an error. If it is greater than the maximum speed, it will be the maximum speed. If it is less than the minimum speed, it will be the minimum speed.

PLSN: can be 0 , greater than the maximum speed is the maximum speed, less than the minimum speed is the minimum speed .

ZRN/ZRN2/DSZR: An error is reported if the creeping speed is less than or equal to 0. An error is reported if the return speed is 0, a maximum speed is reported if it is greater than the maximum speed, and a minimum speed is reported if it is less than the minimum speed .

PLSF: Proportional, please change speed within the maximum range of 200K.

2.11.1 DSZR/Home Return Instruction with DOG Search

Execute origin return to make the machine position consistent with the current value register in the programmable controller, origin return with DOG search.

1. Instruction format

FNC150	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
16-bit	9	DR \	DSZR S1 S2 D1 D2

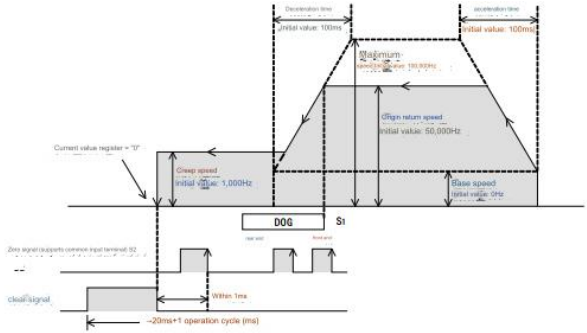
Operand Description

Operands	Content	Type
S1	Input proximity signal device number	Bit
S2	Input number of zero point signal , supports common input terminals	Bit
D1	Output number of the output pulse	Bit
D2	Object number for output pulse direction	Bit

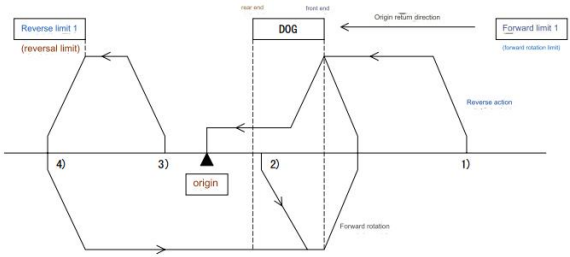
Available operand component types

Operands	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
S1	*	*	*			*																	
S2	*																						
D1		*																					
D2		*																					

2. Function and action description



DOG search function



(1) When the starting position is before the dog

a) Start the home return operation by executing the home return command.

b) Start moving in the home return direction at the home return speed.

c) Once the front end of the dog is detected, start decelerating to the creep speed.

d) After the rear end of the dog is detected, stop when the first zero signal is detected.

(2) When the starting position passes through the DOG

a) Start the homing operation by executing the homing command.

b) Start moving in the direction opposite to the homing direction at the homing speed.

c) After detecting the front end of the DOG, decelerate and stop. (Leave the DOG.)

d) Start moving in the homing direction at the homing speed. (Enter the DOG again.)

e) Once the front end of the DOG is detected, decelerate to the creeping speed.

f) After detecting the rear end of the DOG, stop when the first zero signal is detected.

(3) When the starting position is when the near-point signal is OFF (after passing the DOG),

a) Start the homing operation by executing the homing instruction.

b) Start moving in the homing direction at the homing speed.

c) Decelerate and stop when the reverse limit 1 (reverse limit) is detected.

d) Start moving in the direction opposite to the homing direction at the homing speed.

e) Decelerate and stop after the front end of the DOG is detected. (Detect (leave) the DOG.)

f) Start moving in the homing direction at the homing speed. (Enter the DOG again.)

g) Once the front end of the DOG is detected, decelerate to the creeping speed.

h) After the rear end of the DOG is detected, stop when the first zero signal is detected.

(4) When the limit switch in the origin return direction (forward limit 1 or reverse limit 1) is ON,

a) Start the origin return action by executing the origin return instruction.

b) Start moving in the direction opposite to the origin return direction at the origin return speed.

c) After detecting the front end of the dog, decelerate and stop. (Detect (leave) the dog.)

d) Start moving in the origin return direction at the origin return speed. (Enter the dog again.)

e) Once the front end of the dog is detected, decelerate to the creeping speed.

f) After detecting the rear end of the dog, stop when the first zero point signal is detected.

If the near-point signal and the zero-point signal are specified as the same input, the operation is performed according to the front and rear ends of the near-point signal (DOG) instead of using the zero-point signal, just like the ZRN instruction.

3. Program example

Slightly.

4. Notes

(1) The soft element of the clear signal can only be Y, and it is uniformly placed in SD8XXX. Y000~Y005 are specified by the values of SD8464~SD8469 to specify the clear signal soft element number. The initial value is as shown in the following table. 6/7/8/9 means that the clear signal soft element is Y00 6 /Y00 7 / Y0 10 /Y0 11. If the user wants to specify other numbers , modify the values of SD8464~SD8469 by himself (the decimal numbers starting from 0 are as shown in the following table. Note that the subscript of the Y soft element needs to be converted) .

	SD8464	SD8465	SD8466	SD8467	SD8468	SD8469
16:00	6(Y6)	7(Y7)	/	/	/	/
32 points	6(Y6)	7(Y7)	8(Y10)	9(Y11)	/	/
60 points	6 (Y6)	7 (Y7)	8 (Y10)	9 (Y11)	10 (Y12)	11 (Y13)

(2) If it is not at the near point or limit switch at startup, it will run in the startup direction specified by the SM8342 flag. Otherwise, the background will automatically assign the direction to run according to the position. Here, only the origin startup direction is specified, and the origin return direction cannot be set.

2.11.2 DVIT/Interrupt Positioning Instruction

Execute the command of single-speed interruption fixed-length feed.

1. Instruction format

FNC151	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
16-bit	9	DVIT	-[(D)DVIT S1 S2 D1 D2]
		\	
32-bit	17	D DVIT	
		\	

Operand Description

Operands	Content	Type
S1	Specify the number of output pulses after the interruption (relative address)	BIN16 bit/32 bit
S2	Specify output pulse frequency	BIN16 bit/32 bit
D1	Specify the output number of the output pulse	Bit
D2	Specifies the output object number of the rotation direction signal	Bit

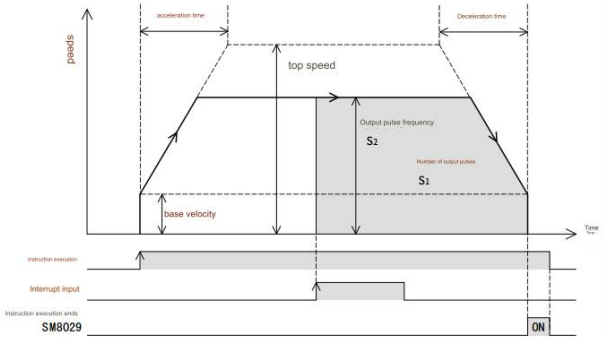
Available operand component types

Operands	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
S1								*	*	*	*	*	*	*	*				*	*			
S2								*	*	*	*	*	*	*	*				*	*			
D1		*																					
D2		*																					

2. Function and action description

S1 setting range: 16-bit -32768~32767 (except 0); 32-bit -2147483648~2147483647 (except 0)

S2 setting range: 16-bit 10~32767 (HZ); 32-bit 10~200000 (HZ).

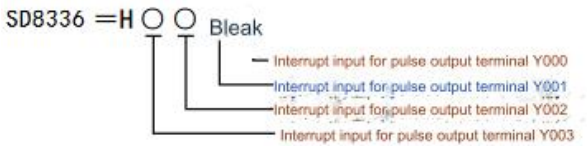


3. Program example

Slightly.

4. Notes

(1) For the output of D1, SD8336 directly specifies the interrupt input as the input soft element number (X000~X007), or specifies the user interrupt instruction soft element. 32-point PLC is specified by SD8336, and 60-point PLC is specified by SD8336 (low bit) and SD8337 (high bit). The two values after HXXXX represented by SD8337 represent Y004 and Y005 from low to high.



<table>
<tr><th>Settings</th><th>Content</th></tr>
<tr><td>0</td><td>Specify X000 as the interrupt input signal</td></tr>
<tr><td>1</td><td>Specify X001 as the interrupt input signal</td></tr>
<tr><td>…</td><td>…</td></tr>
<tr><td>7</td><td>Specify X007 as interrupt input signal</td></tr>
<tr><td>8</td><td>User interrupt instruction device is designated as interrupt input
<table>
<tr><th>Pulse output terminal soft element</th><th>User interrupt instruction device</th></tr>
<tr><td>Y000</td><td>SM8460</td></tr>
<tr><td>Y001</td><td>SM8461</td></tr>
<tr><td>Y002</td><td>SM8462</td></tr>
<tr><td>Y003</td><td>SM8463</td></tr>
<tr><td>Y004 (60 points)</td><td>SM8464</td></tr>
<tr><td>Y005 (60 points)</td><td>SM8465</td></tr>
</table></td></tr>
<tr><td>F</td><td>Specify the pulse output terminal soft element not used in the DVIT instruction as F</td></tr>
</table>

(2) If an interrupt signal is input before the instruction is closed, DVIT is equivalent to pressing DRVI to act, but this situation only supports user input interrupt mode, not external input interrupt mode (external input is edge triggered).

(3) The DVIT instruction can change the output pulse frequency S2 at will even during the pulse output process .

2.11.3 ZRN/Zero Point Return Instruction

This command executes origin return to make the machine position consistent with the current value register in the programmable controller. If the DOG search function is required, use the DSZR (FNC 150) command.

1. Instruction format

<table>
<tr><th>FNC156</th><th>The number of program steps occupied by the instruction</th><th>Instruction Classification</th><th>Instruction Format</th></tr>
<tr><td rowspan="2">16-bit</td><td rowspan="2">7</td><td>ZRN</td><td rowspan="4">[(D)ZRN S1 D1 D2]</td></tr>
<tr><td>\</td></tr>
<tr><td rowspan="2">32-bit</td><td rowspan="2">13</td><td>DZR</td></tr>
<tr><td>\</td></tr>
</table>

Operand Description

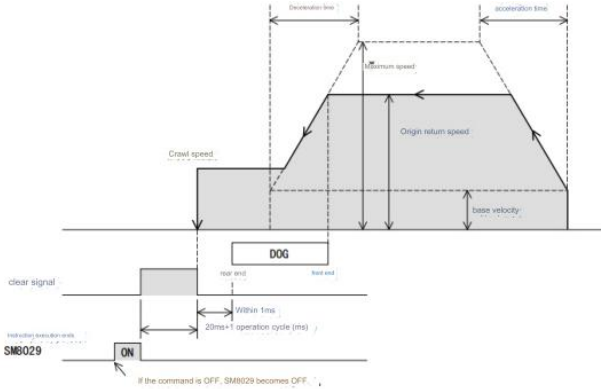
Operands	Content	Type
S1	Specify the device number of the input proximity signal (DOG)	Bit
D1	Specify the terminal to output pulses	Bit
D2	the terminal to export the direction	Bit

Available operand component types

Operands	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
S1	*																						
D1		*																					
D2		*																					

2. Function and action description

Action:



3. Program example

Slightly.

4. Notes

(1) Base speed ≤ origin return speed ≤ maximum speed. If the set origin return speed exceeds the maximum speed, the machine will

run at the maximum speed.

(2) Please set the clear signal terminal Y in the special auxiliary register. For details, see the description of the DSZR instruction .

2.11.4 ZRN 2/Origin return command 2

This command executes origin return to make the machine position consistent with the current value register in the programmable controller. If the DOG search function is required, use the DSZR (FNC 150) command.

1. Instruction format

FNC215	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
16-bit	7	ZRN2	[(D)ZRN2 S1 D1 D2]
		\	
32-bit	13	DZRN2	
		\	

Operand Description

Operands	Content	Type
S 1	Specify the origin signal terminal	Bit
D1	terminal to output pulses	Bit
D 2	Specify the output terminal to output direction	Bit

Available operand component types

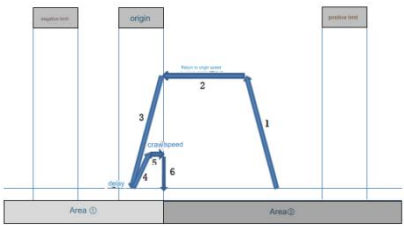
Operands	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
S1	*																						
D1		*																					
D2		*																					

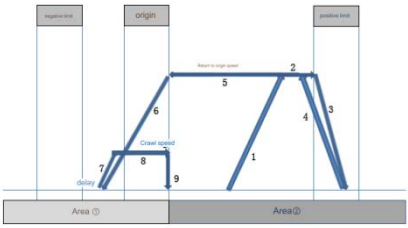
2. Function and action description

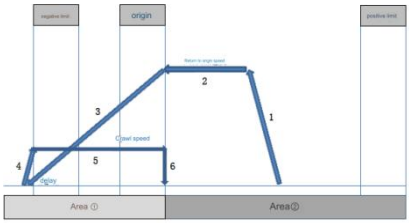
The origin return speed is set in a special register, and the setting range is: 10~32767 (HZ) for 16 bits; 10~200000 (HZ) for 32 bits.

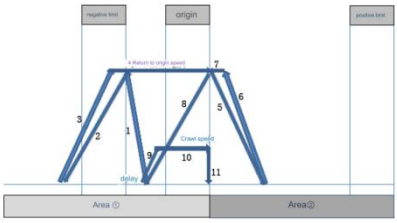
The crawling speed is set in a special register, the same as other origin return instructions. Setting range: 10~32767 (HZ)

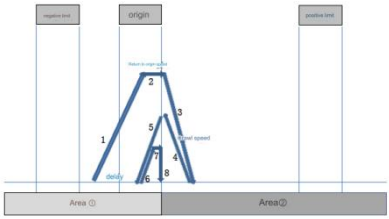
action:

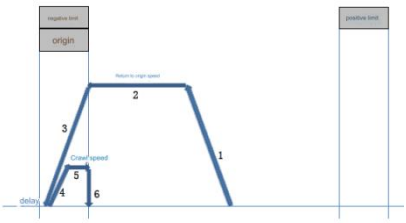












The above represent various situations of ZRN2. Paths with similar paths are not listed. Finally, they all stop on the positive side of the origin.

Some of the functions are the same as DSZR: logic inversion of origin signal, specifying the start direction of origin return, and output of zero reset signal for origin return. The zero reset signal is the same as ZRN and lasts for 20ms.

The user can select the input terminal on the main unit as the origin input signal. When the high-speed input terminal is selected, the origin signal is identified by interruption; when the normal input terminal is selected, the normal input detection is used.

When the workbench is at the negative limit, it will return to the origin, regardless of whether the direction of the return is set to forward or reverse.

When the workbench is at the positive limit, the return to origin is executed, no matter whether the return direction is set to forward or reverse, it can only be executed by default in the reverse return to origin mode.

3. Program example

Slightly.

4. Notes

(1) Base speed ≤ origin return speed ≤ maximum speed. If the set origin return speed exceeds the maximum speed, the machine will

run at the maximum speed.

(2) When the workbench exceeds the limit position, in order to prevent the execution of the return to origin and cause a collision accident, do not execute the return to origin. Please

use the manual jog function to move the workbench back to the negative limit or the positive limit or between them, and then perform the mechanical return to origin.

Point command execution! You can also widen the limit switch width of the negative limit and positive limit to avoid the occurrence of the situation of being out of the positive limit and negative limit when the pulse deceleration stops.

(3) If it is not at the limit switch position at startup, it will run in the startup direction specified by the SM8342 flag. Otherwise, the background will automatically run in the direction assigned by the position.

(4) Please set the clear signal terminal Y in the special auxiliary register. For details, see the description of the DSZR instruction .

2.11.5 PLSV/variable speed pulse output command

Outputs a variable speed pulse command with rotation direction.

1. Instruction format

FNC157	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
16-bit	9	PLSV	-[(D)PLSV S1 D1 D2]
		\	
32-bit	17	D PLSV	
		\	

Operand Description

Operands	Content	Type
S1	Specify the device number of the output pulse frequency	BIN16 bit/32 bit
D1	Specify the output number to output the pulse	Bit
D2	Specifies the output object number of the rotation direction signal	Bit

Available operand component types

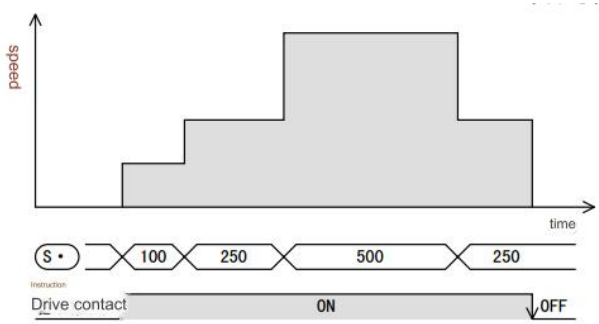
Operands	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
S1								*	*	*	*	*	*	*	*				*	*			
D1		*																					
D2		*																					

2. Function and action description

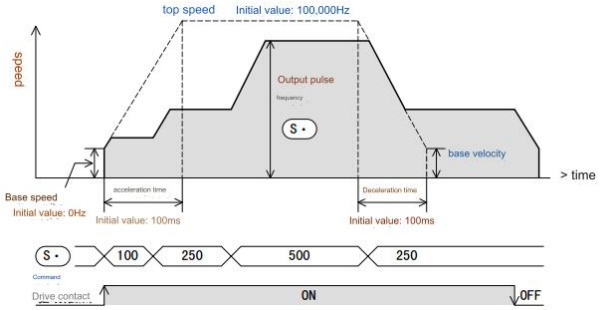
S1 setting range: -32768~32767 (HZ) for 16 bits; -200000~200000 (HZ) for 32 bits.

Action:

(1) When there is no acceleration or deceleration (SM8338=OFF)



(2) When there is acceleration/deceleration (SM8338=ON)



3. Program example

Slightly.

4. Notes

(1) The PLSV instruction can change the output pulse frequency S1 at will even during pulse output.

(2) When S1 is changed to K0, the pulse output will decelerate to stop when there is acceleration or deceleration, and will stop directly when there is no acceleration or deceleration.

2.11.6 DRVI/Relative Positioning Instructions

The command to execute single-speed positioning in relative drive mode. The method of specifying the moving distance from the current position with a positive/negative sign is also called incremental (relative) drive mode.

1. Instruction format

FNC158	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
16-bit	9	DRVI	[(D)DRVI S1 S2 D1 D2]
		\	
32-bit	17	D DRVI	
		\	

Operand Description

Operands	Content	Type
S1	Specify the number of output pulses	BIN16 bit/32 bit
S2	Specify output pulse frequency	BIN16 bit/32 bit
D1	Specify the output number to output the pulse	Bit
D2	Specifies the output object number of the rotation direction signal	Bit

Available operand component types

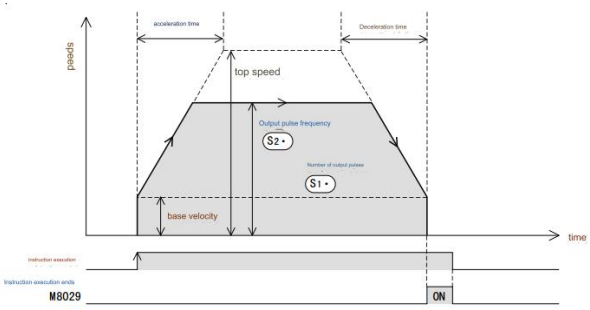
Operands	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
S1								*	*	*	*	*	*	*	*				*	*			
S2								*	*	*	*	*	*	*	*				*	*			
D1		*																					
D2		*																					

2. Function and action description

S1 setting range: 16-bit -32768~32767 (except 0); 32-bit -2147483648~2147483647 (except 0)

S2 setting range: 16-bit 10~32767 (HZ); 32-bit 10~200000 (HZ).

Action:



3. Program example

Slightly.

4. Notes

(1) During the execution of the instruction, only the frequency can be changed . Even if the Contents of the operands other than S2 are changed, it will not be reflected in the current operation. It will be effective only at the next instruction drive.

(2) During the execution of the instruction, when the drive contact is OFF, the deceleration stops. At this time, the instruction execution end flag SM8029 does not act.

(3) When the limit flag of the action direction (forward limit flag or reverse limit flag) acts or the pulse output stop instruction flag acts, it stops immediately. At this time, the instruction execution abnormal end flag SM8329 is set to ON, ending the execution of the instruction.

(4) The DRVI instruction can change the output pulse frequency S2 at will even during the pulse output process .

2.11.7 DRVA/Absolute Positioning Instructions

in absolute drive mode. The method of specifying the moving distance from the current position with a positive/negative sign is also called incremental (absolute) drive mode.

1. Instruction format

FNC159	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
16-bit	9	DRVA	(D)DRVA S1 S2 D1 D2
		\	
32-bit	17	D D V A	
		\	

Operand Description

Operands	Content	Type
S1	Specify the number of output pulses	BIN16 bit/32 bit
S2	Specify output pulse frequency	BIN16 bit/32 bit
D1	Specify the output number to output the pulse	Bit
D2	Specifies the output object number of the rotation direction signal	Bit

Available operand component types

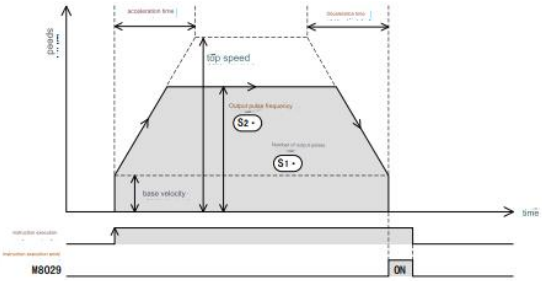
Operands	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
S1								*	*	*	*	*	*	*	*				*	*			
S2								*	*	*	*	*	*	*	*				*	*			
D1		*																					
D2		*																					

2. Function and action description

S1 setting range: 16-bit -32768~32767; 32-bit -2147483648~2147483647

S2 setting range: 16-bit 10~32767 (HZ); 32-bit 10~200000 (HZ).

Action:



3. Program example

Slightly.

4. Notes

(1) During the execution of the instruction, only the frequency can be changed. Even if the Contents of the operands

other than S2 are changed, it will not be reflected in the current operation. It will be effective only at the next instruction drive.

(2) During the execution of the instruction, when the drive contact is OFF, the deceleration stops. At this time, the instruction execution end flag SM8029 does not act.

(3) When the limit flag of the action direction (forward limit flag or reverse limit flag) acts or the pulse output stop instruction flag acts , it stops immediately . At this time, the instruction execution abnormal end flag SM8329 is set to ON, ending the execution of the instruction.

(4) The DRVA instruction can change the output pulse frequency S2 at will even during the pulse output process.

2.11.8 PLSN/multi-segment positioning instructions

It is executed in relative drive mode , with acceleration and deceleration during operation according to the set output pulse frequency and number of pulses in each section.

1. Instruction format

FNC288	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
16-bit	\	\	-[PLSN S1 S2 D1 D2 S3]
		\	
32-bit	twenty one	PLSN	
		\	

Operand Description

Operands	Content	Type
S1	Specify the number of first pulse outputs, occupying 2*n registers	32-bit
S2	Specify the first pulse output frequency, occupying 2*n registers	32-bit
D1	Specify the output number to output the pulse	Bit
D2	Specifies the output object number of the rotation direction signal	Bit
S3	Specify the number of multi-speed stages n (n range 2-16)	32-bit

Available operand component types

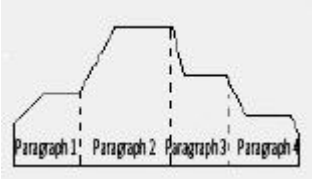
Operands	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
S1														*	*								
S2														*	*								
D1		*																					
D2		*																					
S3																			*	*			

2. Function and action description

S1 setting range: -2147483648 ~2147483647

S2 setting range: 10~200000 (HZ).

Action:



3. Program example

Slightly.

4. Notes

(1) During the execution of the instruction, even if the Contents of the operand are changed, it will not be reflected in the current operation. It will be effective only when the instruction is driven next time.

(2) During the execution of the instruction, when the drive contact is OFF, the deceleration stops. At this time, the instruction execution end flag SM8029 does not act.

(3) When the limit flag of the action direction (forward limit flag or reverse limit flag) acts, the deceleration stops. At this time, the instruction execution abnormal end flag SM8329 is turned ON, and the execution of the instruction ends.

2.11.9 PLSF/Follow instruction

Follow the high-speed counter to output the corresponding frequency and pulse number.

1. Instruction format

FNC289	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
16-bit	\	\	[PLSF S1 S2 S3 D1 D2]
		\	
32-bit	twenty one	PLSF	
		\	

Operand Description

Operands	Content	Type
S1	Molecule N	32-bit
S2	Denominator M	32-bit
S3	High-speed counter C235-C255	32-bit
D1	Specify the output number to output the pulse	Bit
D2	Specifies the output object number of the rotation direction signal	Bit

Available operand component types

Operands	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
S1								*	*	*	*			*	*								
S2								*	*	*	*			*	*								
S3													*										
D1		*																					
D2		*																					

2. Function and action description

S1 setting range: -100000 ~ 100000, cannot be 0

S2 setting range: -100000 ~ 100000, cannot be 0

Action:

Numerator N/denominator M = output frequency/high-speed counter frequency (periodic change value)

3. Program example

Slightly.

4. Notes

(1) During the execution of the instruction, even if the Contents of the operand are changed, it will not be reflected in the current operation. It will be effective only when the instruction is driven next time.

(2) When the limit flag of the action direction (forward limit flag or reverse limit flag) is activated, the motor decelerates and stops. At this time, the instruction execution abnormal end flag S M8329 is set to ON, ending the execution of the instruction.

2.12 Clock operation instructions

Instructions for calculating and comparing clock data can also modify the built-in clock time.

2.12.1 TCMP/Clock Data Compare Instruction

Compare the comparison base time and time data, and control the bit soft element ON/OFF according to the comparison result.

1. Instruction format

FNC160	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
16-bit	11	TCMP TCMPP	-[TCMP(P) S1 S2 S3 S D]

Operand Description

Operands	Content	Type
S1	Specify the comparison base time "hour". [Setting range: 0 to 23]	BIN16 bits
S2	Specify the "minute" of the comparison base time. [Setting range: 0 to 59]	BIN16 bits
S3	Specify the comparison base time in seconds. [Setting range: 0 to 59]	BIN16 bits
S	Specify the "hour" of time data (hour, minute, second). (Occupies 3 points)	BIN16 bits
D	Turn on/off the bit soft element according to the comparison result. (Occupies 3 points)	Bit

Available operand component types

Operands	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
S1								*	*	*	*	*	*	*	*	*	*	*	*	*			
S2								*	*	*	*	*	*	*	*	*	*	*	*	*			
S3								*	*	*	*	*	*	*	*	*	*	*	*	*			
D1												*	*	*	*			*					
D2		*	*			*	*											*					

2. Function and action description

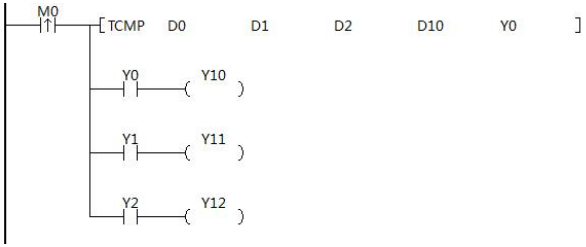
Compare the reference time (hour, minute, second) (S1, S2, S3) with the time data (hour, minute, second) (S, S+1, S+2) , and turn D on or off at the beginning of the three points depending on the match.

When the time of (S1, S2, S3) > the time of (S, S+1, S+2), D is set;

When the time of (S1, S2, S3) = the time of (S, S+1, S+2), D+1 is set;

When the time of (S1, S2, S3) < the time of (S, S+1, S+2), D+2 is set.

3. Program example



When the time corresponding to D0, D1, and D2 is greater than the time composed of D10 to D12, Y0 is set and Y10 is output;

When the time corresponding to D0, D1, and D2 is equal to the time composed of D10 to D12, Y1 is set and output Y11;

When the time corresponding to D0, D1, and D2 is smaller than the time composed of D10 to D12, Y2 is set and Y12 is output.

4. Notes

(1) Even if the command input is OFF and the TCMP command is not executed, D~D+2 will remain as the command input

The state before changing from ON to OFF.

(2) When using the time (hour, minute, second) of the clock data of the real-time clock built into the programmable controller, use the TRD (FNC 166) instruction to read the value of the special data register and then specify its word device in each operand.

2.12.2 TZCP/Clock Data Zone Comparison Instruction

Compare the comparison base time and time data of the upper and lower points, and control the ON/OFF of the specified bit soft element according to the comparison result.

1. Instruction format

FNC161	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
16-bit	9	TZP TZCPP	TZCP(P) S1 S2 S3 D

Operand Description

Operands	Content	Type
S1	Specify the "hour" of the comparison lower limit time (hour, minute, second). (3 points are occupied)	BIN16 bits
S2	Specify the "hour" of the upper limit time (hour, minute, second) for comparison. (3 points are used)	BIN16 bits
S3	Specify the "hour" of time data (hour, minute, second). (Occupies 3 points)	BIN16 bits
D	Turn on/off the bit soft element according to the comparison result. (Occupies 3 points)	Bit

Available operand component types

Operands	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
S1								*	*	*	*	*	*	*	*	*	*	*	*	*			
S2								*	*	*	*	*	*	*	*	*	*	*	*	*			
S3								*	*	*	*	*	*	*	*	*	*	*	*	*			
D2		*	*			*	*											*					

2. Function and action description

Compare the upper and lower 2-point comparison reference time (hour, minute, second) with the 3-point time data (hour, minute, second) starting with S, and turn on/off the 3-point soft elements starting with D according to the comparison result.

When the time of (S1, S1+1, S1+2)>(S, S+1, S+2), D is set;

When the time of (S1, S1+1, S1+2) ≤ the time of (S, S+1, S+2) ≤ the time of (S2, S2+1, S2+2), D+1 is set;

When the time of (S1, S1+1, S1+2)>(S, S+1, S+2), D+2 is set;

3. Program example

Slightly.

4. Notes

When using the time (hour, minute, second) of the clock data of the real-time clock built into the programmable controller, use the TRD (FNC 166) instruction to read the value of the special data register and then specify its word device in each operand.

2.12.3 TADD/Clock Data Addition Instruction

After adding two time data, store them in the word device.

1. Instruction format

FNC162	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
16-bit	7	TADD	[TADD(P) S1 S2 D]
		TADDP	

Operand Description

Operands	Content	Type
S1	Specify the "hour" of the time data (hour, minute, second) to be added. (3 points are occupied)	BIN16 bits
S2	Specify the "hour" of the time data (hour, minute, second) to be added. (3 points are occupied)	BIN16 bits
D	Save the result of the addition of two time data (hour, minute, second). (Occupies 3 points)	BIN16 bits

Available operand component types

Operands	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
S1												*	*	*	*			*					
S2												*	*	*	*			*					
D												*	*	*	*			*					

2. Function and action description

Add the time data (hour, minute, second) of (S1, S1+1, S1+2) to the time data (hour, minute, second) of [S2, S2+1, S2+2) and save the result in (D, D+1, D+2) (hour, minute, second).

"Hour" range: 0~23

"Points" range: 0~59

"Seconds" range: 0~59

3. Program example

Slightly.

4. Notes

The action of the main related flags: When the calculation result exceeds 24 hours, the carry flag turns on, and 24 hours are subtracted from the simple addition calculation value and the time is used as the calculation. When the calculation result is 0 (0 hours, 0 minutes, 0 seconds), the zero flag turns on.

2.12.4 TSUB/Clock Data Subtraction Instruction

After subtracting two time data, store them in word devices.

1. Instruction format

FNC163	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
16-bit	7	TSUB	-[TSUB(P) S1 S2 D]
		TSUBP	

Operand Description

Operands	Content	Type
S1	Specify the "hour" of the time data (hour, minute, second) to be subtracted. (3 points are occupied)	BIN16 bits
S2	Specify the "hour" of the time data (hour, minute, second) to be subtracted. (3 points are occupied)	BIN16 bits
D	Save the result of subtraction of two time data (hour, minute, second). (Occupies 3 points)	BIN16 bits

Available operand component types

Operands	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
S1												*	*	*	*			*					
S2												*	*	*	*			*					
D																		*					

2. Function and action description

Subtract the time data (hour, minute, second) of (S2, S2+1, S2+2) from the time data (hour, minute, second) of (S1, S1 + 1 , S1+2) and save the result in [D, D+1, D+2) (hour, minute, second).

"Hour" range: 0~23

"Points" range: 0~59

"Seconds" range: 0~59

3. Program example

Slightly.

4. Notes

Action of related flags: When the operation result is less than 0, the borrow flag turns on, and 24 hours are added to the simple subtraction value, and the time is saved as the operation result. When the operation result is 0 (0 hours, 0 minutes, 0 seconds), the zero flag turns on.

2.12.5 HTOS/Seconds conversion instructions for hour, minute and second data

[hour, minute, second] units into data in seconds.

1. Instruction format

FNC164	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
16-bit	5	HTOS	-[(D)HTOS(P) S D]
		HTOSP	
32-bit	9	DHTOS	
		DHTOSP	

Operand Description

Operands	Content	Type
S	The starting number of the device that stores the time (time) data (hour, minute, second) before conversion	BIN16 bits
D	The device number that stores the converted time (hour) data (seconds).	BIN16/32 bits

Available operand component types

Operands	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
S								*	*	*	*	*	*	*	*			*					
D									*	*	*	*	*	*	*			*					

2. Function and action description

(1) 16-bit instructions

converting the time (moment) data (hour, minute, second) of (S, S+1, S+2) into seconds, save the result in D.

S range: 0~9

S+1 range: 0~59

S+2 Range: 0~59

(2) 32-bit instructions

converting the time (moment) data (hour, minute, second) of (S, S+1, S+2) into seconds, save the result in (D+1, D).

S range: 0~32767

S+1 range: 0~59

S+2 Range: 0~59

3. Program Example

Slightly.

4. Notes

When the data of S, S+1, and S+2 exceed the settable range corresponding to the instruction, error 6706 is reported.

2.12.6 STOH/second data (hour, minute, second) conversion instruction

converts the time (moment) data in seconds into data in [hour, minute, second] units.

1. Instruction Format

FNC165	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
16-bit	5	STOH	-[(D)STOH(P) S D]
		STOHP	
32-bit	9	DSTOH	
		DSTOHP	

Operand Description

Operands	Content	Type
S	The device number that stores the time (hour) data (seconds) before conversion	BIN16/32 bits
D	The starting number of the device that stores the converted time (moment) data (hour, minute, second)	BIN16 bits

Available operand component types

Operands	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
S								*	*	*	*	*	*	*	*			*					
D									*	*	*	*	*	*	*			*					

2. Function and action description

(1) 16-bit instructions

Convert the second data of S into hours, minutes, and seconds, and save the result in (D, D+1, D+2) (hours, minutes, seconds).

S range: 0~32767

(2) 32-bit instructions

Convert the second data of (S+1, S) into hours, minutes, and seconds, and save the result in (D, D+1, D+2) (hours, minutes, seconds).

(S+1, S) range: 0~117964799

3. Program example

Slightly.

4. Notes

If the data to be converted exceeds the settable range corresponding to the instruction, error 6706 is reported.

2.12.7 TRD/Read Clock Data Instruction

SD8013 ~ SD8019) of the real-time clock built into the programmable controller from D ~ D+6 according to the following format .

1. Instruction format

FNC166	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
16-bit	3	TRD	[TRD(P) D]
		TRDP	

Operand Description

Operands	Content	Type
D	Specify the starting soft element number to store the read time data. (7 points are occupied)	BIN16 bits

Available operand component types

Operands	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
D												*	*	*	*			*					

2. Function and action description

SD8013 ~ SD8019) of the real-time clock built into the programmable controller to D ~ D+6 according to the following format .

	Software	project	Clock data		Software	project
Special Data Registers	SD8018	Year (Gregorian calendar)	0~99 (last 2 digits of the Gregorian calendar)	→	D	Year (Gregorian calendar)
	SD8017	moon	1~12	→	D +1	moon
	SD8016	day	1~31	→	D +2	day
	SD8015	hour	0~23	→	D +3	hour
	SD8014	point	0~59	→	D +4	point
	SD8013	Second	0~59	→	D +5	Second
	SD8019	Week	0 (Sun) ~ 6 (Sat)	→	D +6	Week

3. Program example

Slightly.

4. Notes

D occupies 7 soft elements. Please be careful not to duplicate the soft elements used in other machine controls.

2.12.8 TWR/Write Clock Data Instruction

This command writes clock data to the real-time clock built into the programmable controller.

1. Instruction format

FNC167	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
16-bit	3	TWR TWRP	TWR(P) S

Operand Description

Operands	Content	Type
S	Specify the starting soft element number of the source address where the time data is written. (Occupies 7 points)	BIN16 bits

Available operand component types

Operands	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
S1												*	*	*	*			*					

2. Function and action description

the set clock data S to S+6 into the clock data (SD8013 toSD8019) of the real-time clock built into the programmable controller .

	Software	project	Clock data		Software	project	
Data for setting time	S	Year (Gregorian calendar)	0~99 (last 2 digits of the Gregorian calendar)	→	SD8018	Year (Gregorian calendar)	Special Data Registers
	S+1	moon	1~12	→	SD8017	moon	
	S+2	day	1~31	→	SD8016	day	
	S+3	hour	0~23	→	SD8015	hour	
	S+4	point	0~59	→	SD8014	point	
	S+5	Second	0~59	→	SD8013	Second	
	S+6	Week	0 (Sun) ~ 6 (Sat)	→	SD8019	Week	

3. Program example

Slightly.

4. Notes

(1) After executing the TWR (FNC 167) instruction, the clock data of the real-time clock will be changed immediately. Therefore, please first transfer the clock data that is a few minutes faster to S ~ S+6, and then execute the instruction when it becomes the correct time.

(2) When using this instruction to set the clock data (time calibration), it is not necessary to control the special auxiliary relay S M8015 (time stop and time calibration).

(3) If a date and time value that cannot be displayed is set, the clock data will not be changed. In this case, set the correct clock data and write again.

(4) The day of the week (SD8019) has nothing to do with the written value and is automatically corrected based on the date Content.

(5) When occupying the 7 consecutive soft elements starting with S, be careful not to overlap with the soft elements used in other machine controls.

(6) When setting the time, set it 1 minute faster. When the correct time is reached, set the trigger switch in front of the TWR instruction to update the clock data.

2.12.9 HOUR/Hour meter instructions

This command accumulates and detects the time that the input contact remains ON in units of one hour.

1. Instruction format

FNC169	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
16-bit	7	HOUR	-[(D)HOUR S D1 D2]
		\	
32-bit	13	DHOUR	
		\	

Operand Description

Operands	Content	Type
S	The time to keep D2 ON (set in units of 1 hour)	BIN16/32 bits
D1	Current value in units of one hour (specify data register for power failure retention)	BIN16/32 bits
D2	The starting number of alarm output	Bit

Available operand component types

Operands	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
S								*	*	*	*	*	*	*	*	*	*	*	*	*			
D1														*	*			*					
D2		*	*			*	*											*					

2. Function and action description

(1) 16-bit instructions

When the cumulative ON time of the command input exceeds S, D2 turns ON. The current value of D1+1 that is less than 1 hour is stored in 1-second units.

S: The time until D2 turns ON is specified in units of 1 hour.

D1: Current value in units of 1 hour

D1+1: Current value less than 1 hour (1 second unit)

D2: Alarm output destination address number. When the current value D1 is longer than the time specified by S, it turns ON.

(2) 32-bit instructions

(S+1, S): The time until D2 turns ON is specified by S1+1 (high), S1 (low).

(D1+1, D1): The current value in units of 1 hour is stored in D1+1 (high), D1 (low).

D1+2: The current value less than 1 hour (1 second unit)

D2: Alarm output designation, the current value D1, D1+1 is turned ON when it is longer than the specified time of S.

3. Program example

Slightly.

4. Notes

(1) Since the current value data can be used even after the power of the PLC is turned off, specify the data register for power failure retention in D1. When using a general data register, the current value will be cleared by the PLC power off and STOP→RUN operation.

(2) After the alarm output D2 turns ON, the measurement can be continued. To restart the measurement, clear the current values of D1 to D1+1 (D1 to D1+2 for 32-bit instructions). When the current values are cleared, the alarm output turns OFF.

(3) When the current value D1 reaches the maximum value, the measurement stops. To continue the measurement, clear the current values of D1 to D1+1 (D1 to D1+2 for 32-bit instructions).

(4) D1 occupies 2 (16-bit operation) or 3 (32-bit operation) soft elements. Be careful not to overlap with soft elements used in other machine controls.

2.13 Gray code instructions

Mainly used for conversion operations of Gray code data.

2.13.1 GRY/Gray code conversion instructions

This command converts the BIN value into Gray code and transmits it.

1. Instruction Format

FNC170	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
16-bit	5	GRY	[(D)GRY(P) S D]
		GRYP	
32-bit	9	DGRY	
		DGRYP	

Operand Description

Operands	Content	Type
S	Conversion source data, or word device storing conversion source data	BIN16/32 bits
D	Word device to store converted data	BIN16/32 bits

Available operand component types

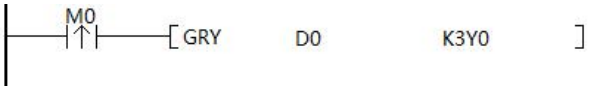
Operands	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
S								*	*	*	*	*	*	*	*	*	*	*	*	*			
D									*	*	*	*	*	*	*	*	*	*					

2. Function and action description

The conversion transmission instruction of source data (BIN) → target data (GRY). The value of Gray code can be obtained by shifting right one bit on the basis of the original binary and then XORing the original binary value.

3. Program Example

S is D0 = 3456 and D is K3Y0



soft components	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0	value
D0	0	0	0	0	1	1	0	1	1	0	0	0	0	0	0	0	3456

soft components	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0	value
Y0	0	0	0	0	1	0	1	1	0	1	0	0	0	0	0	0	2880

4. Precautions

(1) The data conversion speed depends on the scan time of the programmable controller.

(2) For 16-bit instructions, the range of S is 0 to 32767; for 32-bit instructions, the range of S is 0 to 2147483647.

2.13.2 GBIN/Gray Code Inverse Conversion Instructions

This command converts Gray code into BIN value and transmits it.

1. Instruction format

FNC171	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
16-bit	5	GBIN	[(D)GBIN(P) S D]
		GBINP	
32-bit	9	DGBIN	
		DGBINP	

Operand Description

Operands	Content	Type
S	Conversion source data, or word device storing conversion source data	BIN16/32 bits
D	Word device to store converted data	BIN16/32 bits

Available operand component types

Operands	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
S								*	*	*	*	*	*	*	*	*	*	*	*	*			
D									*	*	*	*	*	*	*	*	*	*					

2. Function and action description

Source data (GRY) → target data (BIN) conversion transmission instruction. Binary Gray code is converted into natural binary code. The rule is to retain the highest bit of Gray code as the highest bit of natural binary code, and the second highest bit of natural binary code is the exclusive OR of the high bit of natural binary code and the second highest bit of Gray code, and the rest of the bits of natural binary code are similar to the method of calculating the second highest bit of natural binary code.

3. Program example

When S is K3 Y 0 and D is D0 , after the instruction is executed, the result of D0 is as follows.

soft components	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0
Y0	0	0	0	0	0	1	1	0	1	0	1	1	1	0	1	1

soft components	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0	value
D0	0	0	0	0	0	1	0	0	1	1	0	1	0	0	1	0	1234

4. Notes

For 16-bit instructions, the range of S is 0 to 32767; for 32-bit instructions, the range of S is 0 to 2147483647.

2.14 Data block instructions

An instruction that performs addition, subtraction, and comparison operations on batch data simultaneously.

2.14.1 BK+/Addition instruction of data block

Instruction for BIN addition of data blocks.

1. Instruction format

FNC192	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
16-bit	9	BK+	(D)BK+(P) S1 S2 D n
		BK+P	
32-bit	17	DBK+	
		DBK+P	

Operand Description

Operands	Content	Type
S1	The starting number of the device that stores the data to be added	BIN16/32 bits
S2	The constant to be added, or the starting number of the device that stores the data to be added	BIN16/32 bits
D	The starting number of the device that stores the calculation results	BIN16/32 bits
n	Number of data	BIN16/32 bits

Available operand component types

Operands	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
S1												*	*	*	*			*					
S2												*	*	*	*			*	*	*			
D												*	*	*	*			*					
n														*	*				*	*			

2. Function and action description

(1) 16-bit instructions

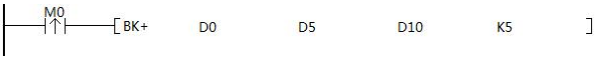
After adding the 16-bit data of n points starting from S1 and the 16-bit data (BIN) of n points starting from S2, the result is stored in the n points starting from D.

(2) 32-bit instructions

After adding the 32-bit data of 2n points starting from (S1+1, S1) and the 32-bit data (BIN) of 2n points starting from (S2+1, S2), the result is stored in the 2n points starting from (D+1, D).

3. Program example

Taking 16-bit instruction operation as an example , the results of D10~D14 after the instruction is executed are as follows.



soft components	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0	value
D0	0	0	0	0	0	1	0	0	1	1	0	1	0	0	1	0	1234
D1	0	0	0	1	0	1	1	0	0	0	1	0	1	1	1	0	5678
D2	1	1	1	1	1	0	0	0	0	0	1	1	0	0	0	0	-2000
D3	1	1	1	1	1	0	1	1	0	0	1	0	1	1	1	0	-1234
D4	0	0	0	0	1	1	1	1	1	0	1	0	0	0	0	0	4000
D5	0	0	0	0	1	1	1	1	1	0	1	0	0	0	0	0	4000
D6	0	0	0	0	0	1	0	0	1	1	0	1	0	0	1	0	1234
D7	1	1	1	1	1	0	1	1	0	0	1	0	1	1	1	0	-1234
D8	0	0	0	1	0	0	1	1	1	0	0	0	1	0	0	0	5000
D9	0	0	0	1	0	0	0	0	1	1	1	0	0	0	0	1	4321
D10	0	0	0	1	0	1	0	0	0	1	1	1	0	0	1	0	5234
D11	0	0	0	1	1	0	1	1	0	0	0	0	0	0	0	0	6912
D12	1	1	1	1	0	0	1	1	0	1	0	1	1	1	1	0	-3234
D13	0	0	0	0	1	1	1	0	1	0	1	1	0	1	1	0	3766
D14	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0	1	8321

4. Notes

(1) When data overflow or underflow occurs in the operation result, the carry flag is not set.

(2) Error 6706 will be reported if the following situations occur.

• When n-point devices starting from S1, S2, and D (2n-point devices for 32-bit operation) exceed the corresponding device range.

• When n-point devices starting from S1 overlap with n-point devices starting from D (2n-point devices for 32-bit operation).

• When n-point devices starting from S2 overlap with n-point devices starting from D (2n-point devices for 32-bit operation).

2.14.2 BK-/Data Block Subtraction Instructions

Instruction for BIN subtraction of data blocks.

1. Instruction format

FNC193	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
16-bit	9	BK-	
		BK-P	
32-bit	17	DBK-	
		DBK-P	

Operand Description

Operands	Content	Type
S1	The starting number of the device that stores the data to be subtracted	BIN16/32 bits
S2	The constant to be subtracted, or the starting number of the device that stores the data to be subtracted	BIN16/32 bits
D	The starting number of the device that stores the calculation results	BIN16/32 bits
n	Number of data	BIN16/32 bits

Available operand component types

Operands	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
S1												*	*	*	*			*					
S2												*	*	*	*			*	*	*			
D												*	*	*	*			*					
n														*	*				*	*			

2. Function and action description

(1) 16-bit instructions

After subtracting the 16-bit data of n points starting from S1 and the 16-bit data (BIN) of n points starting from S2, the result is stored in the n points starting from D.

(2) 32-bit instructions

After subtracting the 2n-point 32-bit data starting from (S1+1, S1) and the 2n-point 32-bit data (BIN) starting from (S2+1, S2), the result is stored in the 2n-point starting from (D+1, D).

3. Program example

Taking 16-bit instruction operation as an example , after the instruction is executed, the results of D10~D14 are as follows.

soft components	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0	value
D0	0	0	1	0	0	0	1	0	0	0	1	1	1	1	0	1	8765
D1	0	0	1	0	0	0	1	0	1	0	1	1	1	0	0	0	8888
D2	0	0	1	0	0	1	0	0	0	1	1	0	1	1	0	1	9325
D3	0	0	0	1	0	0	1	1	1	0	0	0	1	0	0	0	5000
D4	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0	4352
D5	0	0	0	0	0	1	0	0	1	1	0	1	0	0	1	0	1234
D6	0	0	0	1	0	1	1	0	0	0	1	0	1	1	1	0	5678
D7	0	0	1	0	0	1	1	0	1	0	0	1	0	1	0	0	9876
D8	0	0	0	1	0	0	0	0	1	1	1	0	0	0	0	1	4321
D9	0	0	0	0	1	1	1	1	1	0	1	0	0	0	0	0	4000
D10	0	0	0	1	1	1	0	1	0	1	1	0	1	0	1	1	7531
D11	0	0	0	0	1	1	0	0	1	0	0	0	1	0	1	0	3210
D12	1	1	1	1	1	1	0	1	1	1	0	1	1	0	0	1	-551
D13	0	0	0	0	0	0	1	0	1	0	1	0	0	1	1	1	679
D14	0	0	0	0	0	0	0	1	0	1	1	0	0	0	0	0	352

4. Notes

(1) When data overflow or underflow occurs in the operation result, the carry flag is not set.

(2) Error 6706 will be reported if the following situations occur.

• When n-point devices starting from S1, S2, and D (2n-point devices for 32-bit operation) exceed the corresponding device range.

• When n-point devices starting from S1 overlap with n-point devices starting from D (2n-point devices for 32-bit operation).

• When n-point devices starting from S2 overlap with n-point devices starting from D (2n-point devices for 32-bit operation).

2.14.3 BKCMP =, >, <, <>, <=, >=/Data Block Comparison Instructions

These instructions compare blocks of data according to the comparison conditions of each instruction.

1. Instruction format

FNC NO	Number of instruction bits	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
194	16-bit	9	BKCMP=	(D)BKCMP=(P) S1 S2 D n
			BKCMP=P	
	32-bit	17	DBKCMP=	
			DBKCMP=P	
195	16-bit	9	BKCMP>	(D)BKCMP>(P) S1 S2 D n
			BKCMP>P	
	32-bit	17	DBKCMP>	
			DBKCMP>P	
196	16-bit	9	BKCMP<	(D)BKCMP<(P) S1 S2 D n
			BKCMP<P	
	32-bit	17	DBKCMP<	
			DBKCMP<P	
197	16-bit	9	BKCMP<>	(D)BKCMP<>(P) S1 S2 D n
			BKCMP<>P	
	32-bit	17	DBKCMP<>	
			DBKCMP<>P	
198	16-bit	9	BKCMP<=	(D)BKCMP<=(P) S1 S2 D n
			BKCMP<=P	
	32-bit	17	DBKCMP<=	
			DBKCMP<=P	
199	16-bit	9	BKCMP>=	(D)BKCMP>=(P) S1 S2 D n
			BKCMP>=P	
	32-bit	17	DBKCMP>=	
			DBKCMP>=P	

Operand Description

Operands	Content	Type
S1	Comparison value or device number storing comparison value	BIN16/32 bits
S2	The starting number of the device that stores the comparison source data	BIN16/32 bits
D	The starting number of the device that stores the comparison results	Bit
n	The number of data to compare	BIN16/32 bits

Available operand component types

Operands	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
S1												*	*	*	*			*	*	*			
S2												*	*	*	*			*					
D		*	*			*	*																
n														*	*				*	*			

2. Function and action description

(1) 16-bit instructions

After comparing the 16-bit data of n points starting from S1 with the 16-bit data (BIN) of n points starting from S2, the comparison result is saved in the n points starting from D.

(2) 32-bit instructions

After comparing the 32-bit data of 2n points starting from (S1+1, S1) and the 32-bit data (BIN) of 2n points starting from (S2+1, S2), the comparison result is saved in the 2n points starting from (D+1, D).

3. Program example

Taking the 16-bit instruction BKCMP> as an example , after the instruction is executed, the results of Y0~Y4 are as follows.

soft components	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0	value
D0	0	0	0	0	0	1	0	0	1	1	0	1	0	0	1	0	1234
D1	0	0	0	1	0	1	1	0	0	0	1	0	1	1	1	0	5678
D2	0	0	0	1	0	0	1	1	1	0	0	0	1	0	0	0	5000
D3	0	0	0	1	1	1	1	0	0	1	1	0	0	0	0	1	7777
D4	0	0	0	1	0	0	0	0	0	1	1	1	0	0	0	1	4321
D5	0	0	0	1	0	1	0	0	1	1	0	0	1	0	0	1	5321
D6	0	0	0	0	1	1	0	1	0	1	0	0	0	1	1	1	3399
D7	0	0	0	1	0	1	1	0	0	0	1	0	1	1	1	0	5678
D8	0	0	0	1	1	0	0	1	1	0	0	0	1	1	1	1	6543
D9	0	0	0	0	0	1	0	0	1	0	1	1	0	0	0	0	1200

soft components	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0
Y0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	1	0

4. Notes

(1) The instruction comparison results are as follows

FNC No	16-bit instructions	32-bit instructions	Conditions for comparison result to be ON	Conditions for comparison result OFF
194	BKCMP=	DBKCMP=	S1=S2	S1≠S2
195	BKCMP>	DBKCMP>	S1>S2	S1≤S2
196	BKCMP<	DBKCMP<	S1<S2	S1≥S2
197	BKCMP<>	DBKCMP<>	S1≠S2	S1=S2
198	BKCMP<=	DBKCMP<=	S1≤S2	S1>S2
199	BKCMP>=	DBKCMP>=	S1≥S2	S1<S2

(2) Action of the S M8090 block comparison signal: It turns ON when the comparison results of the data block instructions are all ON (1).

2.15 String control instructions

Related operation instructions for controlling strings, including conversion, replacement, retrieval, and retrieval of strings.

2.15.1 STR/BIN → character string conversion instruction

This command converts BIN data into a string (ASCII code).

1. Instruction Format

FNC200	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
16-bit	7	STR	[(D)STR(P) S1 S2 D]
		STRP	
32-bit	13	DSTR	
		DSTRP	

Operand Description

Operands	Content	Type
S1	The starting number of the device that stores the number of digits of the value to be converted	BIN16/32 bits
S2	The device number that stores the BIN data to be converted	BIN16/32 bits
D	The starting number of the device that stores the converted character string	String

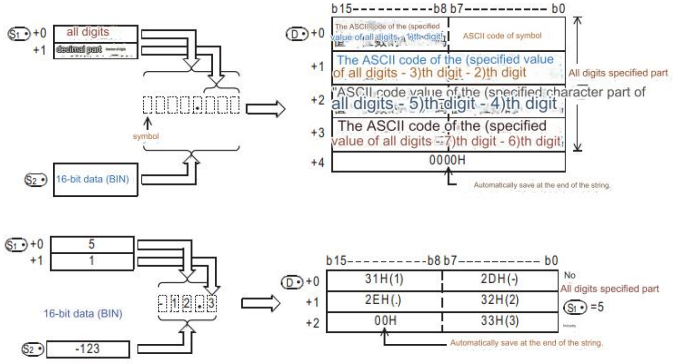
Available operand component types

Operands	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
S1												*	*	*	*			*					
S2								*	*	*	*	*	*	*	*	*	*	*	*	*			
D												*	*	*	*			*					

2. Function and action description

The 16-bit or 32-bit data (BIN) of S2 is converted into a character string after adding a decimal point to the position specified by the total number of digits (S1) and the number of digits for the decimal point (S1 +1), and is stored in the device starting with D.

with 16-bit instruction data:



The converted character string data is stored in the device numbers starting with D as shown below:

• In the sign, "space (20H)" is stored when the data (BIN) S2 is positive, and "-(2DH)" is stored when it is negative.

• When a number other than "0" is set in the decimal place S1+1, a decimal point ". (2EH)" is automatically added at the decimal place + 1 place. When the decimal place + 1 is "0", no decimal point is added.

• When the decimal place of S1+1 has more digits than the 16-bit or 32-bit data (BIN) of S2, it is automatically right-aligned and "0 (30H)" is added to the left before conversion.

• If the total number of digits of S1 excluding the decimal point and sign is greater than the number of digits of the data (BIN) of S2, "space (20H)" is stored between the sign and the value. In addition, if the number of digits of the data (BIN) of S2 is large, an error will occur.

• At the end of the converted character string, "00H" is automatically stored to indicate the end of the character string. When the total number of digits is an even number, "0000H" is stored in the device after the device storing the last character. If the total number of digits is an odd number, "00H" is stored in the high byte (8 bits) of the device storing the last character.

3. Program Example

Slightly.

4. Precautions

Error 6706 will be reported in the following situations:

• When all digits of S1 are outside the range shown below.

16-bit operation: 2 to 8; 32-bit operation: 2 to 13.

• When all the digits S1+1 are outside the range shown below.

16-bit operation: 0 to 5; 32-bit operation: 0 to 10.

• When the relationship between the total number of digits S1 and the number of digits in the decimal part S1+1 is not in the range shown below.

(Total number of digits - 3) ≧Decimal number of digits

• When the total number of digits S1 + sign and decimal point is less than the number of digits in the BIN data of S2.

• When the soft element after D that stores the character string exceeds the corresponding soft element range.

2.15.2 VAL/string → BIN conversion instructions

This command converts a character string (ASCII code) into BIN data.

1. Instruction format

FNC201	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
16-bit	7	VAL	[(D)VAL(P) S D1 D2]
		VALP	
32-bit	13	DVAL	
		DVALP	

Operand Description

Operands	Content	Type
S	The starting number of the device that stores the character string to be converted into BIN data	String
D1	The starting number of the device that stores the converted BIN data digits	BIN16 bits
D2	The starting number of the device that stores the converted BIN data	BIN16/32 bits

Available operand component types

Operands	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
S												*	*	*	*			*					
D1												*	*	*	*			*					
D2									*	*	*	*	*	*	*			*					

2. Function and action description

(1) 16-bit instructions

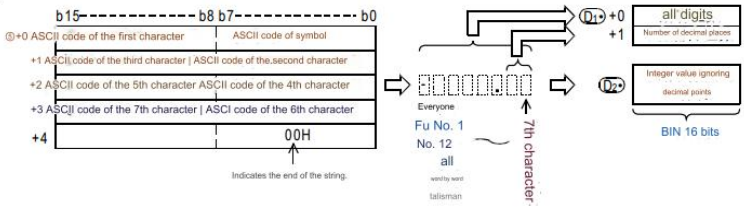
Convert the character string stored in the device starting with S to 16-bit data (BIN), store all digits in D1, store the decimal digits in D1+1, and store the BIN data in D2.

When converting from a character string to BIN, the data from S to the device number storing "00H" is handled as a character string in byte units.

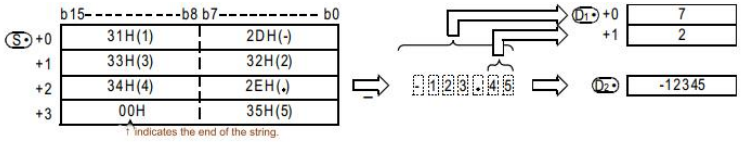
• D1 stores the total number of digits. The total number of digits is the total number of characters (including digits, signs, and decimal points).

• D1 +1 stores the number of digits in the decimal part. The number of digits in the decimal part is the number of characters after the decimal point ". (2EH)".

• D2 ignores the decimal point and converts the character string into 16-bit or 32-bit data (BIN). However, in a character string starting with S, the "space (20H)" or "0 (30H)" between the sign and the first digit other than "0" is ignored .



For example, when a character string of "-123.45" is specified in a device starting with S, the following is stored in D1 and D2.



• The string data range to be converted is limited to :

	Content
All character digits	16-bit instruction: 2 to 8 characters 32-bit instructions: 2 to 13 characters
Number of characters for the decimal part	16-bit instruction: 0 to 5 characters, but the number of digits is less than (total digits - 3) 32-bit instructions: 0 to 10 characters, but the number of digits is less than (total digits - 3)
Numeric range ignoring decimal points	16-bit instruction: -32768~32767 (e.g. "123.45" -> "12345") 32-bit instruction: -2147483648~2147483647 (e.g. "12345.678" -> "12345678")

• Character restrictions used in Characters to Convert :

		Types of characters
symbol	Positive value	"Space (20H)"
	Negative values	"-(2DH)"
Decimal Point		“.(2EH)”
number		"0(30H)"～"9(39H)"

3. Program example

Slightly.

4. Notes

The string to be converted has data range restrictions and character usage restrictions. If it is not within the restrictions, an error 6706 will be reported.

2.15.3 $+/string combination instructions

Instructions for concatenating strings.

1. Instruction format

FNC202	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
16-bit	7	$+	[$+(P) S1 S2 D]
		$+P	

Operand Description

Operands	Content	Type
S1	String 1	String
S2	String 2	String
D	concatenated string	String

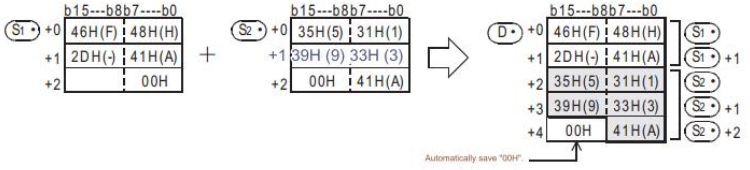
Available operand component types

Operands	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
S1								*	*	*	*	*	*	*	*			*			*		
S2								*	*	*	*	*	*	*	*			*			*		
D									*	*	*	*	*	*	*			*					

2. Function and action description

The string data starting with S2 is connected to the string data starting with S1, and then stored in the soft elements after D.

The string data of S1 and S2 refers to the data from the designated soft element to the position where the first [00H] is detected in byte units.



The string combination means ignoring the "00H" indicating the end of the specified string in S1, and connecting the last character of S1 to the string specified in S2. In addition, after the string combination is executed, "00H" is automatically added to the end.

• When the number of characters after connection is an odd number, "0000H" is stored in the high byte of the soft element that stores the last character.

• When the number of characters after connection is an even number, "0000H" is stored in the soft element next to the soft element that stores the last character.

3. Program example

Slightly.

4. Notes

(1) When a character string is directly specified, the maximum number of characters that can be specified (input) is 32. However, when a word device is specified in S1 and S2, there is no limit to the number of characters.

(2) When the value of either S1 or S2 starts at 00H, "0000H" is stored in D.

(3) Error 6706 is reported in the following situations:

• When the number of devices starting with the device number specified in D is less than the number of devices required to store all the combined character strings. ([00H] cannot be stored after all the character strings and the final character)

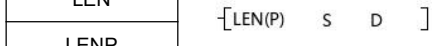
• When the device numbers for storing the character strings specified in S1 and S2 overlap with the device numbers for storing the character strings specified in D.

• When [00H] is not set in the corresponding device range starting with the devices specified in S1 and S2.

2.15.4 LEN/Check the length of a string

This command detects the number of characters (bytes) in a specified string.

1. Instruction format

FNC203	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
16-bit	5	LEN LENP	

Operand Description

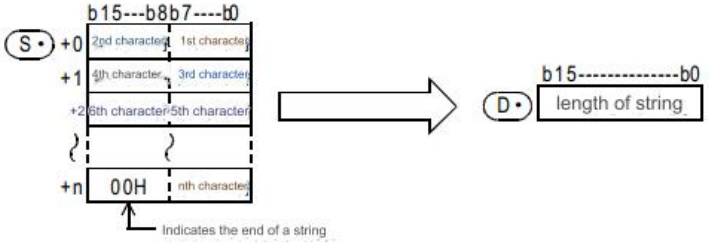
Operands	Content	Type
S	The device start number of the character string whose number of characters is to be detected is stored	String
D	The device number that stores the length (number of bytes) of the detected character string	BIN16 bits

Available operand component types

Operands	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
S								*	*	*	*	*	*	*	*			*					
D									*	*	*	*	*	*	*			*					

2. Function and action description

The length of the string beginning with S is detected and stored in D. S is in bytes, and the data from the beginning to the first device number stored with [00H] is handled as a string.



3. Program example

Slightly.

4. Notes

Error 6706 will be reported in the following situations:

• When [00H] is not set in the corresponding device range starting from the device number specified by S.

• When the number of characters exceeds 32768.

2.15.5 RIGHT/Extract instructions from the right side of a string

This command extracts a specified number of characters from the right side of a specified string.

Instruction Format

FNC204	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
16-bit	7	RIGHT RIGHTP	-[RIGHT(P) S D n]

Operand Description

Operands	Content	Type
S	The starting number of the device that stores the character string	String
D	The starting number of the device that stores the extracted character string	String
n	The number of characters to extract	BIN16 bits

Available operand component types

Operands	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
S								*	*	*	*	*	*	*	*			*					
D									*	*	*	*	*	*	*			*					
n														*	*				*	*			

2. Function and action description

Data of n characters is taken from the right side (end of the string) of the string data stored in the soft element starting with S, and stored in the soft element starting with D. However, when the number of characters specified in n is "0", NULL code (0000H) is stored in D.

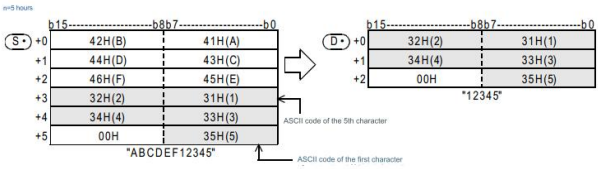
In addition, when the string is taken out, "00H" is automatically appended to the end.

• When the number of characters to be taken out is an odd number, "00H" is stored in the high byte of the soft element that stores the last character.

• When the number of characters to be taken out is an even number, "0000H" is stored in the soft element next to the soft element that stores the last character.

3. Program example

Store the right 5 characters of the string data "ABCDEF12345" starting with S into D:



The character string beginning with S refers to the data starting from the specified soft element and ending at the first [00H] position detected in bytes.

4. Notes

Error 6706 will be reported in the following situations:

• [00H] is not set in the corresponding device range starting with the device specified in S.

• n exceeds the number of characters specified in S.

• The number of devices starting with the device number specified in D is less than the number of devices required to store the extracted character string (n characters). (All character strings and the last character cannot be stored after [00H])

• n is a negative value.

2.15.6 LEFT/Extract instructions from the left side of the string

This command extracts a specified number of characters from the left side of a specified string.

1. Instruction format

FNC205	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
16-bit	7	LEFT LEFTP	-[LEFT(P) S D n]

Operand Description

Operands	Content	Type
S	The starting number of the device that stores the character string	String
D	The starting number of the device that stores the extracted character string	String
n	The number of characters to extract	BIN16 bits

Available operand component types

Operands	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
S								*	*	*	*	*	*	*	*			*					
D									*	*	*	*	*	*	*			*					
n														*	*				*	*			

2. Function and action description

Data of n characters is taken from the left side (the beginning of the string) of the string data stored in the soft element starting with S, and stored in the soft element starting with D. However, when the number of characters specified in n is "0", NULL code (0000H) is stored in D.

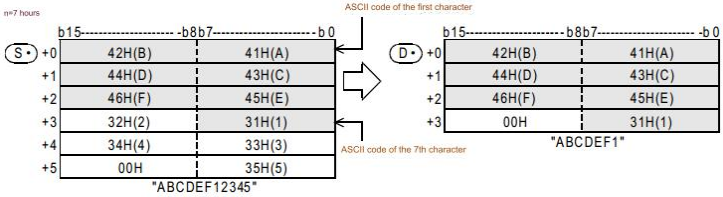
In addition, when the string is taken out, "00H" is automatically appended to the end.

• When the number of characters to be taken out is an odd number, "00H" is stored in the high byte of the soft element that stores the last character.

• When the number of characters to be taken out is an even number, "0000H" is stored in the soft element next to the soft element that stores the last character.

3. Program example

Store the left 7 characters of the string data "ABCDEF12345" starting with S into D:



The character string beginning with S refers to the data starting from the specified soft element and ending at the first [00H] position detected in bytes.

4. Notes

Error 6706 will be reported in the following situations:

• [00H] is not set in the corresponding device range starting with the device specified in S.

• n exceeds the number of characters specified in S.

• The number of devices starting with the device number specified in D is less than the number of devices required to store the extracted character string (n characters). (All character strings and the last character cannot be stored after [00H])

• n is a negative value.

2.15.7 MIDR/Get Anything from a String

This command extracts a character string at any position in the specified character string.

1. Instruction format

FNC206	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
16-bit	7	MIDR MIDRP	-[MIDR(P) S1 D S2]

Operand Description

Operands	Content	Type
S1	The starting number of the device that stores the character string	String
D	The starting number of the device that stores the extracted character string	String
S2	The starting position and number of characters to be retrieved are specified in the soft element starting number S2: starting character position S2+1: number of characters	BIN16 bits

Available operand component types

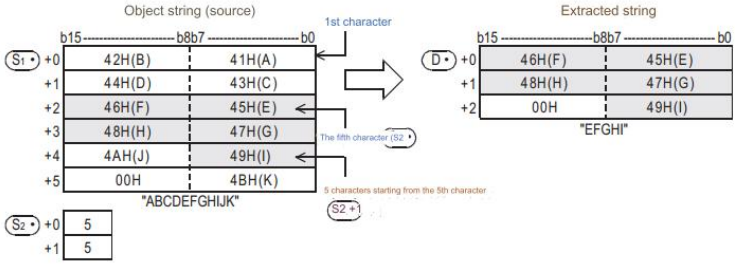
Operands	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
S1								*	*	*	*	*	*	*	*			*					
D									*	*	*	*	*	*	*			*					
S2								*	*	*	*	*	*	*	*			*					

2. Function and action description

Starting from the S2th character on the left side (the beginning of the string) of the string data stored in the soft element starting with S1, data of (S2+1) characters is taken out and stored in the soft element starting with D. In addition, when the string is taken out, "00H" is automatically appended to the end.

• When the number of characters to be taken out (S2+1) is an odd number, "00H" is stored in the high byte of the soft element that stores the last character.

• When the number of characters to be taken out (S2+1) is an even number, "0000H" is stored in the soft element next to the soft element that stores the last character.



• The character string (data) specified in S1 refers to the data from the specified soft element to the detection of the first [00H].

• When the number of characters to be retrieved specified in S2+1 is "0", no processing is performed.

• When the number of characters to be retrieved specified in S2+1 is "-1", the Content up to the last character data of the character string specified by S1 is stored in D.

3. Program example

slightly.

4. Notes

Error 6706 will be reported in the following situations:

• When [00H] is not set in the corresponding device range starting from the device specified in S1.

• When the value of S2 exceeds the number of characters in the string specified in S1.

• When the number of characters starting from D (S2+1) exceeds the device range of D.

• When the number of devices starting from the device number specified in D is less than the number of devices required to store the extracted string (S2+1) characters. (All strings and the last character cannot be stored [00H])

• When S2 is a negative value.

• When S2+1 is a value less than -2.

• When S2+1 exceeds the number of characters in S1.

2.15.8 MIDW/Arbitrary Replacement from String Instruction

An instruction to replace a specified string with a string at any position in the specified string.

1. Instruction format

FNC207	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
16-bit	7	MIDW	[MIDW(P) S1 D S2]
		MIDWP	

Operand Description

Operands	Content	Type
S1	The starting number of the device that stores the character string	String
D	The starting number of the device that stores the replaced character string	String
S2	The starting position of the characters to be replaced and the number of characters specified in the soft element starting number S2: The starting character position of the string to be replaced S2+1: The number of characters to be replaced	BIN16 bits

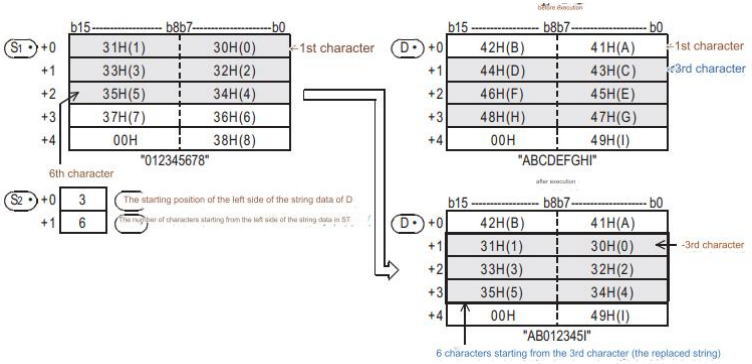
Available operand component types

Operands	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
S1								*	*	*	*	*	*	*	*			*					
D									*	*	*	*	*	*	*			*					
S2								*	*	*	*	*	*	*	*			*					

2. Function and action description

For the character string data stored in the device starting with S1, the character data starting from the left side (the beginning of the character string) (the number of character

data is the number of characters specified by S2+1) is saved to the character string stored in the device starting with D, and the saving position starts from the position specified by S2 from the left of the character string data.

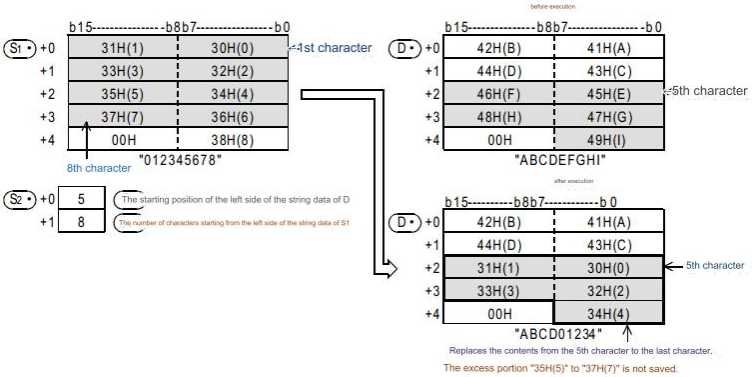


• The character string (data) designated by D in S1 refers to the data from the designated device to the detection head and the first [00H].

• If the number of characters to be replaced designated by S2+1 is " 00 " , no processing is performed.

• When the number of characters specified in S2+1 is "-1", all the Contents up to the last character data specified by S1 are stored in the soft element specified by D and later.

• If the number of characters to be replaced specified by S2+1 exceeds the last character of the character string data starting with D, the data up to the last character will be stored.



1. Program Example

Slightly .

2. Precautions

Error 6706 is reported in the following situations:

• When [00H] is not set in the corresponding device range starting from the device specified by S1 and D.

• When the value of S2 exceeds the number of characters of D.

• When S2 is a negative value.

• When S2+1 is a value less than -2.

• When S2+1 exceeds the number of characters in S1.

2.15.9 INSTR/string search command

Retrieves the specified string from the specified string.

1. Instruction format

FNC208	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
16-bit	9	INSTR INSTRP	┤[INSTR(P) S1 S2 D n]

Operand Description

Operands	Content	Type
S1	The starting number of the device that stores the character string to be searched	String
S2	The device start number that stores the search source character string	String
D	The starting number of the device where the search results are stored	BIN16 bits
n	Start search position	BIN16 bits

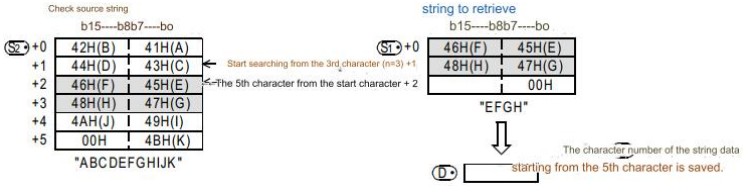
Available operand component types

Operands	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
S1												*	*	*	*			*				*	
S2												*	*	*	*			*					
D												*	*	*	*			*					
n														*	*				*	*			

2. Function and action description

(1) Starting from the nth character from the left (starting character) of the search source string stored in the device starting with S2, search for the same string as the string stored in the device starting with S1, and then store the string position information of the search

result in D. The search result is the character position information of the nth character from the left (starting character) of the search source string.



(2) If there is no matching string, "0" is stored in D.

(1) If the search start position n is a negative number or "0", no processing is performed.

(2) String S1 can directly specify string type data

3. Program example

Slightly.

4. Notes

Error 6706 is reported in the following situations:

• When the search start position n exceeds the number of characters of S2.

• When there is no 00H (NULL) in the device range of the corresponding device starting from S1.

• When there is no 00H (NULL) in the device range of the corresponding device starting from S2.

2.15.10 $MOV/string transfer instruction

Instructions for transferring string data.

1. Instruction format

FNC209	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
16-bit	5	$MOV	-[$MOV(P) S D]
		$MOVP	

Operand Description

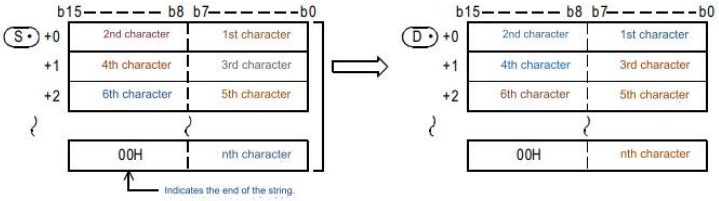
Operands	Content	Type
S	Source string (maximum 32 characters),	String
D	Destination string	String

Available operand component types

Operands	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
S								*	*	*	*	*	*	*	*			*				*	
D									*	*	*	*	*	*	*			*					

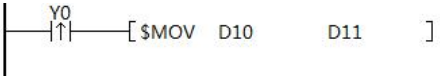
2. Function and action description

The character string data stored in the device specified by S is transferred to the device starting with D. When transferring the character string, the data from the device number specified by S to the device whose high byte or low byte contains "00H" are transferred at once.



3. Program example

123456 stored in D10 to D13 is transferred to D11 to D14 .



After the command is executed, you can see that the string in D10 has not changed after the transmission.

Display Format 16I 32I 32E ASC 10 16

soft components	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0	value
D10	0	0	1	1	0	0	1	0	0	0	1	1	0	0	0	1	12
D11	0	0	1	1	0	0	1	0	0	0	1	1	0	0	0	1	12
D12	0	0	1	1	0	1	0	0	0	0	1	1	0	0	1	1	34
D13	0	0	1	1	0	1	1	0	0	0	1	1	0	1	0	1	56

The transfer can be performed even when the device range (S to S+n) storing the character string data to be transferred and the device range (D to D+n/2) storing the character string data to be transferred are used in duplicate.

4. Notes

(1) When "00H" is stored in the low byte of S+n, "00H" is stored in both the high byte and the low byte of D+n.

(2) Error 6706 is reported in the following situations:

• When "00H" does not exist between the device number specified in S and the last device number specified.

• When all the specified character strings cannot be stored in the points between the device number specified in D and the last device number specified.

2.16 Data processing 3 instructions

Used to read the data entered later and control the left and right shift instructions with carry.

2.16.1 FDEL/Data table data deletion instruction

A command to delete any data in the data table.

1. Instruction format

FNC210	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
16-bit	7	FDEL	-[FDEL(P) S D n]
		FDELP	

Operand Description

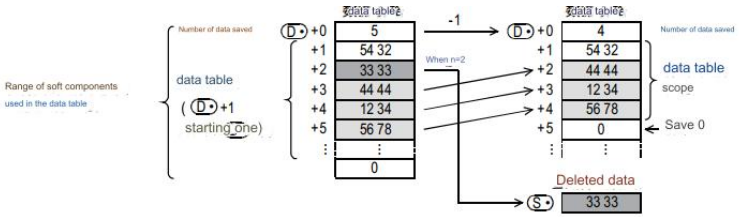
Operands	Content	Type
S	Save deleted data	BIN16 bits
D	Data table start soft element	BIN16 bits
n	The data to be deleted is located in the table	BIN16 bits

Available operand component types

Operands	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
S												*	*	*	*			*					
D												*	*	*	*			*					
n														*	*				*	*			

2. Function and action description

Delete the nth data in the data table (starting from D), and save the deleted data to S. The data starting from the n+1th data in the data table are moved forward one by one , and the number of data saved is reduced by 1.



3. Program example

Slightly .

4. Notes

(1) The range of the data table is D starting from the next soft element (D+1) of the data storage number D.

(2) Error 6706 is reported in the following situations:

• When the table position n of the data to be deleted is larger than the number of stored data.

• When the value of n exceeds the device range of the data table D.

• When the instruction is executed when n ≤ 0.

• When the value of the number of stored data D is 0.

• When the range of the data table exceeds the corresponding device range.

2.16.2 FINS/Data table data insertion instructions

A command to insert data at any position in a data table.

1. Instruction format

FNC211	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
16-bit	7	FINS	[FINS(P) S D n]
		FINSP	

Operand Description

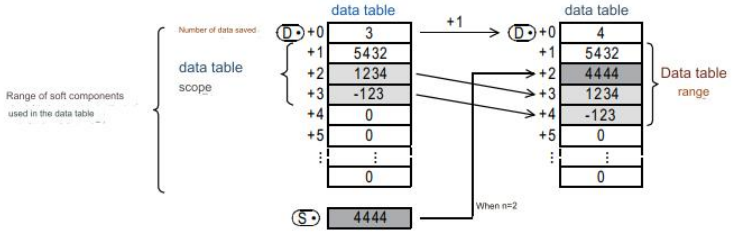
Operands	Content	Type
S	The device number to store the inserted data	BIN16 bits
D	The starting device number of the data table	BIN16 bits
n	Insert data into the table	BIN16 bits

Available operand component types

Operands	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
S												*	*	*	*			*					
D												*	*	*	*			*					
n														*	*				*	*			

2. Function and action description

Insert the 16-bit data S into the nth number of the data table (after D). The data after the nth number of the data table are shifted back one by one, and the number of data stored is increased by 1.



3. Program example

Slightly .

4. Notes

(1) The range of the data table is D starting from the next soft element (D+1) of the data storage number D.

(2) Error 6706 is reported in the following situations:

• When the table position n of the data to be inserted is greater than the number of stored data + 1.

• When the value of n exceeds the device range of the data table D.

• When the instruction is executed when n ≤ 0.

• When the range of the data table exceeds the corresponding device range.

2.16.3 POP/read the last input data instruction

Data is read out in a first-in, last-out manner.

1. Instruction format

FNC212	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
16-bit	7	POP POPP	[POP(P) S D n]

Operand Description

Operands	Content	Type
S	The starting word element of the data to be read (S is the number of data points, and the data starts from S + 1)	BIN16 bits
D	Read data	BIN16 bits
n	The number of data points to save + 1 (2 ≦ n ≦ 512)	BIN16 bits

Available operand component types

Operands	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
S								*	*	*	*	*	*	*	*			*					
D								*	*	*	*	*	*	*	*	*	*	*					
n																			*	*			

2. Function and action description

First-in, last-out control data

value	Content
S	Pointer data (the number of data to be saved)
S+1	Data area (data first written using the unique write instruction SFWR)
S+2	
...	
S+n-2	
S+n-1	

For word soft elements (S to S+n-1), each time an instruction is executed, the soft element (S + pointer data S) is read out and saved in D (the last data written using the shift write instruction (SFWR) for first-in first-out control is read into D), and the value of pointer data S is subtracted by 1 once.

3. Program example

Slightly.

4. Notes

(1) The action of S M8020, when pointer S=0, turns ON after executing the instruction and does not process the instruction.

(2) When the current value of the pointer is 1, 0 is written to S and S M8020 turns ON.

(3) Error 6706 is reported in the following situations:

• When S > n-1.

• When S < 0.

2.16.4 SFR/16-bit data n-bit right shift (with carry) instruction

This instruction shifts the 16 bits in a word device to the right by n bits .

1. Instruction format

FNC213	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
16-bit	5	SFR	[SFR(P) D n]
		SFRP	

Operand Description

Operands	Content	Type
D	The starting word device number for storing the data to be moved	BIN16 bits
n	Number of moves 0≦n≦15	BIN16 bits

Available operand component types

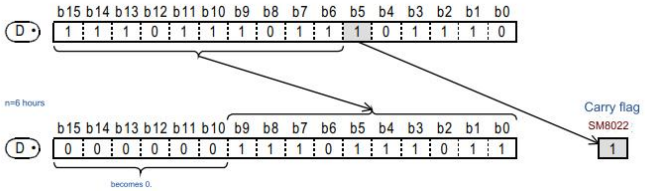
Operands	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
D									*	*	*	*	*	*	*	*	*	*					
n								*	*	*	*	*	*	*	*	*	*		*	*			

2. Function and action description

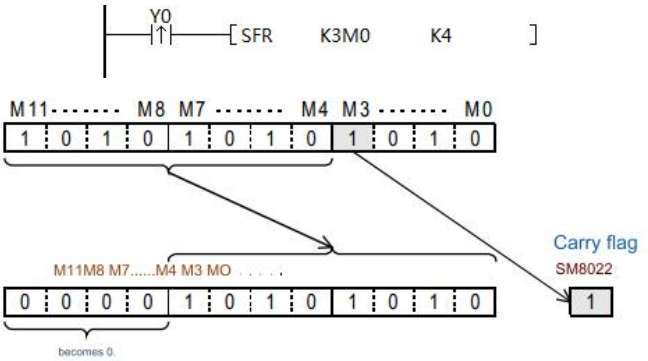
(1) Shift the 16 bits in word soft element D right by n bits. n specifies a number from 0 to 15. When n specifies a number greater than 16, the shift is based on the remainder of n/16. For example, if n=18, 18/16=1 with a remainder of 2, so it is shifted right by 2 bits.

(2) Transfer the ON(1)/OFF(0) state of the nth bit (n-1 bit) in word soft element D to the carry flag M8022.

(3) The n bits starting from the most significant bit become 0.



(4) When specifying a bit device by digit designation, data is shifted by the specified number of digits (4*K coefficient) .



3. Program example

Slightly .

4. Notes

(1) The action of the carry flag of S M8022 is turned on or off according to the status of shift n-1.

(2) If n is a negative value, error 6706 is reported.

2.16.5 SFL/16-bit data n-bit left shift (with carry) instruction

Instruction to shift the 16 bits in a word soft element to the left by n bits

1. Instruction format

FNC214	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
16-bit	5	SFL	[SFL(P) D n]
		SFLP	

Operand Description

Operands	Content	Type
D	The starting word device number for storing the data to be moved	BIN16 bits
n	Number of moves 0≦n≦15	BIN16 bits

Available operand component types

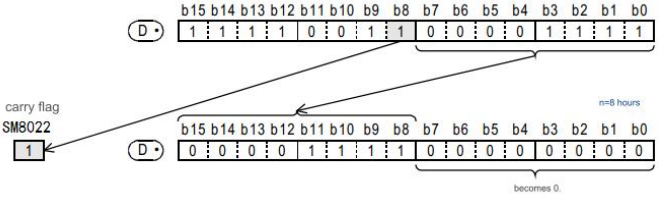
Operands	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
D									*	*	*	*	*	*	*	*	*	*					
n								*	*	*	*	*	*	*	*	*	*		*	*			

2. Function and action description

(1) Shift the 16 bits in word soft element D left by n bits. n specifies a number from 0 to 15. When n specifies a number greater than 16, shift according to the remainder of n/16. For example, when n=18, 18/16=1 with a remainder of 2, so shift right by 2 bits.

(2) Transfer the ON(1)/OFF(0) state of the n+1th bit (nth bit) in word soft element D to the carry flag M8022.

(3) The n bits starting from the lowest bit are changed to0.



(4) When specifying a bit device by digit designation, data of the specified digit (4×K□) is shifted.

3. Program example

Slightly .

4. Notes

(1) The action of the carry flag of S M8022 is turned on or off according to the status of shift n.

(2) If n is a negative value, error 6706 is reported.

2.17 Contact comparison instructions

Provides instructions for data comparison using contact symbols , and the contact comparison symbols are displayed in UNISYS.

2.17.1 LD=, >, <, <>, <=, >= / Contact comparison instructions

This instruction starts the contact comparison operation by comparing numerical values and turning on the contact when the condition is met.

1. Instruction Format

FNC NO	Number of instruction bits	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
224	16-bit	5	LD=	⊣[LD(D)= S1 S2]⊢
	32-bit	9	LDD=	
225	16-bit	5	LD>	⊣[LD(D)> S1 S2]⊢
	32-bit	9	LDD>	
226	16-bit	5	LD<	⊣[LD(D)< S1 S2]⊢
	32-bit	9	LDD<	
228	16-bit	5	LD<>	⊣[LD(D)<> S1 S2]⊢
	32-bit	9	LDD<>	
229	16-bit	5	LD<=	⊣[LD(D)<= S1 S2]⊢
	32-bit	9	LDD<=	
230	16-bit	5	LD>=	⊣[LD(D)>= S1 S2]⊢
	32-bit	9	LDD>=	

Operand Description

Operands	Content	Type
S1	Comparison data 1	BIN16/32 bits
S2	Comparison data 2	BIN16/32 bits

Available operand component types

Operands	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
S1								*	*	*	*	*	*	*	*	*	*	*	*	*			
S2								*	*	*	*	*	*	*	*	*	*	*	*	*			

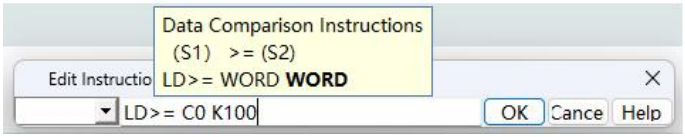
2. Function and action description

The contact comparison instruction connected to the bus performs a BIN comparison on the Contents of S1 and S2, and controls the conduction or non-conduction of the contact according to the result.

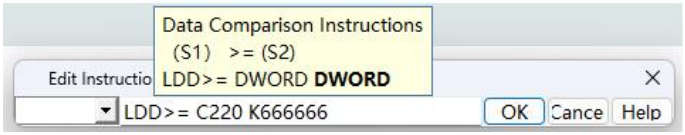
FNC No	16-bit instructions	32-bit instructions	Conductivity conditions	Non-conduction condition
224	LD=	LDD=	S1=S2	S1≠S2
225	LD>	LDD>	S1>S2	S1≤S2
226	LD<	LDD<	S1<S2	S1≥S2
228	LD<>	LDD<>	S1≠S2	S1=S2
229	LD<=	LDD<=	S1≤S2	S1>S2
230	LD>=	LDD>=	S1≥S2	S1<S2

3. Program Example

(1) 16-bit operation input



(2) 32-bit operation input



(3) Examples

When C0 value is >= 100, Y0 is driven, and when C220 value is >= 6666 6 6, Y1 is driven.

4. Precautions

(1) Note that 32-bit instructions are selected when the comparison source is a 32-bit counter (C200~C255).

(2) Pay attention to how the instructions are written.

2.17.2 AND=, >, <, <>, <=, >= / Contact comparison instructions

This instruction starts the contact comparison operation by comparing numerical values and turning on the contact when the condition is met.

1. Instruction format

FNC NO	Number of instruction bits	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
232	16-bit	5	AND=	⊣[AND(D)= S1 S2]
	32-bit	9	ANDD=	
233	16-bit	5	AND>	⊣[AND(D)> S1 S2]
	32-bit	9	ANDD>	
234	16-bit	5	AND<	⊣[AND(D)< S1 S2]
	32-bit	9	ANDD<	
236	16-bit	5	AND<>	⊣[AND(D)<> S1 S2]
	32-bit	9	ANDD<>	
237	16-bit	5	AND<=	⊣[AND(D)<= S1 S2]
	32-bit	9	ANDD<=	
238	16-bit	5	AND>	⊣[AND(D)>= S1 S2]
	32-bit	9	ANDD>=	

Operand Description

Operands	Content	Type
S1	Comparison data 1	BIN16/32 bits
S2	Comparison data 2	BIN16/32 bits

Available operand component types

Operands	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
S1								*	*	*	*	*	*	*	*	*	*	*	*	*			
S2								*	*	*	*	*	*	*	*	*	*	*	*	*			

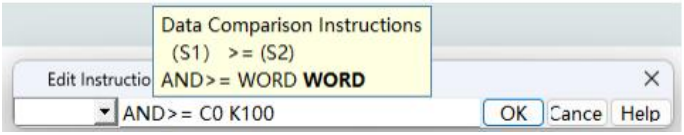
2. Function and action description

Contact comparison instruction connected in series with other contacts. Perform BIN comparison on the Contents of S1 and S2, and control the conduction or non-conduction of the contacts according to the result.

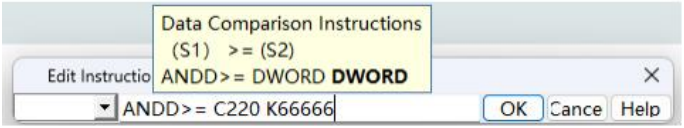
FNC No	16-bit instructions	32-bit instructions	Conductivity conditions	Non-conduction condition
232	AND=	ANDD=	S1=S2	S1≠S2
233	AND>	ANDD>	S1>S2	S1≤S2
234	AND<	ANDD<	S1<S2	S1≥S2
236	AND<>	ANDD<>	S1≠S2	S1=S2
237	AND<=	ANDD<=	S1≤S2	S1>S2
238	AND>=	ANDD>=	S1≥S2	S1<S2

3. Program Example

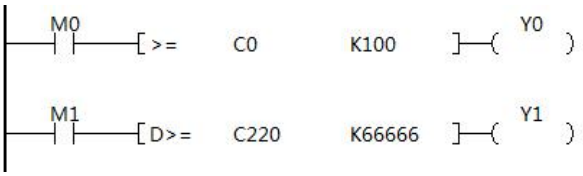
(1) 16-bit operation input



(2) 32-bit operation input



(3) Examples



When M0 is ON and C0 value >= 100, Y0 is driven; when M1 is ON and C220 value >= 66666, Y1 is driven.

4. Notes

(1) Note that when the comparison source is a 32-bit counter (C200 to C255), select a 32-bit instruction.

(2) Pay attention to the way instructions are written.

2.17.3 OR=, >, <, <>, <=, >= / Contact comparison instructions

This instruction starts the contact comparison operation by comparing numerical values and turning on the contact when the condition is met.

1. Instruction format

FNC NO	Number of instruction bits	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
240	16-bit	5	OR=	[OR(D)= S1 S2]
	32-bit	9	ORD=	
241	16-bit	5	ORD>	[OR(D)> S1 S2]
	32-bit	9	ORD>	
242	16-bit	5	OR<	[OR(D)< S1 S2]
	32-bit	9	ORD<	
244	16-bit	5	OR<>	[OR(D)<> S1 S2]
	32-bit	9	ORD<>	
245	16-bit	5	OR<=	[OR(D)<= S1 S2]
	32-bit	9	ORD<=	
246	16-bit	5	OR>	[OR(D)>= S1 S2]
	32-bit	9	ORD>=	

Operand Description

Operands	Content	Type
S1	Comparison data 1	BIN16/32 bits
S2	Comparison data 2	BIN16/32 bits

Available operand component types

Operands	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
S1								*	*	*	*	*	*	*	*	*	*	*	*	*			
S2								*	*	*	*	*	*	*	*	*	*	*	*	*			

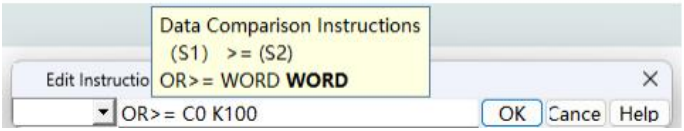
2. Function and action description

A contact comparison instruction connected in parallel with other contacts. Perform a BIN comparison on the Contents of S1 and S2, and control the conduction or non-conduction of the contacts based on the result.

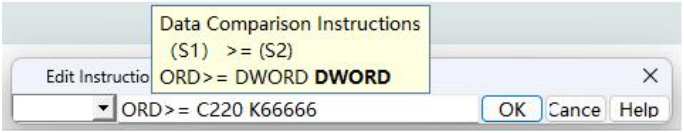
FNC No	16-bit instructions	32-bit instructions	Conductivity conditions	Non-conduction condition
240	OR=	ORD=	S1=S2	S1≠S2
241	OR>	ORD>	S1>S2	S1≤S2
242	OR<	ORD<	S1<S2	S1 ≥ S2
244	OR<>	ORD<>	S1≠S2	S1=S2
245	OR<=	ORD<=	S1≤S2	S1>S2
246	OR>=	ORD>=	S1 ≥ S2	S1<S2

3. Program example

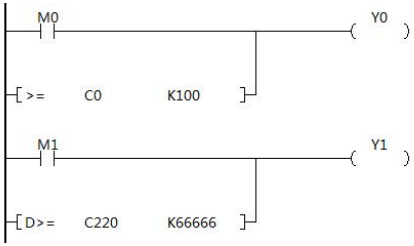
(1) 16-bit operation input



(2) 32-bit operation input



(3) Examples



When M0 is ON or C0 value>=100, Y0 is driven; when M1 is ON or C220 value>=66666, Y1 is driven.

4. Notes

(1) Note that when the comparison source is a 32-bit counter (C200 to C255), select a 32-bit instruction.

(2) Pay attention to the way instructions are written.

2.18 Data table processing instructions

Mainly used to control output data when input data passes through group data.

2.18.1 LIMIT/Upper and lower limit control instructions

This command sets the upper and lower limits of input values and outputs them.

1. Instruction format

FNC256	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
16-bit	9	LIMIT	-[(D)LIMIT(P) S1 S2 S3 D]
		LIMITP	
32-bit	17	DLIMIT	
		DLIMITP	

Operand Description

Operands	Content	Type
S1	Lower limit value	BIN16/32 bits
S2	Upper limit value	BIN16/32 bits
S3	Input value to be controlled by upper and lower limits	BIN16/32 bits
D	Output value after upper and lower limit control	BIN16/32 bits

Available operand component types

Operands	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
S1								*	*	*	*	*	*	*	*			*	*	*			
S2								*	*	*	*	*	*	*	*			*	*	*			
S3								*	*	*	*	*	*	*	*			*					
D									*	*	*	*	*	*	*			*					

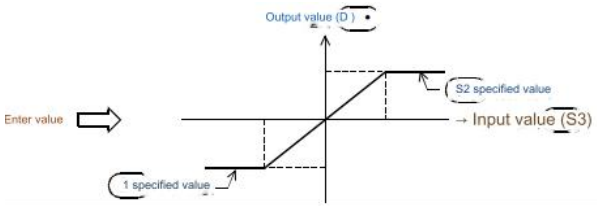
2. Function and action description

By judging whether the input value specified in S3 is within the upper and lower limit values specified in S1 and S2, the output value stored in the soft element specified by D is controlled.

the lower limit value > input value, the output value = lower limit value;

When the upper limit value < input value, the output value = upper limit value;

When the lower limit value ≤ input value ≤ upper limit value, the output value = input value.



3. Program example

Slightly .

4. Notes

After executing the command, if the set lower limit value > upper limit value, error 6706 will be reported.

2.18.2 BAND/Dead Band Control Instructions

The output value is controlled by judging whether the input value is within the upper and lower limits of the specified dead zone.

1. Instruction format

FNC257	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
16-bit	9	BAND	[(D)BAND(P) S1 S2 S3 D]
		BANDP	
32-bit	17	DBAND	
		DBANDP	

Operand Description

Operands	Content	Type
S1	Lower limit of dead zone (no output area)	BIN16/32 bits
S2	Upper limit of dead zone (no output area)	BIN16/32 bits
S3	Input value to be controlled by deadband	BIN16/32 bits
D	Output value after dead-band control	BIN16/32 bits

Available operand component types

Operands	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
S1								*	*	*	*	*	*	*	*			*	*	*			
S2								*	*	*	*	*	*	*	*			*	*	*			
S3								*	*	*	*	*	*	*	*			*					
D									*	*	*	*	*	*	*			*					

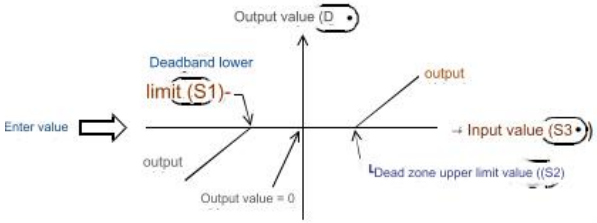
2. Function and action description

By judging whether the input value specified by S3 is within the dead zone specified by S1 and S2, the output value stored in the soft element specified by D is controlled.

the lower limit value > input value, the output value = input value - lower limit value;

the upper limit value < input value, the output value = input value - upper limit value;

When the lower limit value ≤ input value ≤ upper limit value, the output value = 0.



3. Program example

Slightly.

4. Notes

(1) When the output value overflows, the output value = input value - dead zone lower limit value

16-bit operation: When the operation result exceeds -32768~32767

Example: S1 = 10, S3 = -32768, output D = -32768 - 10 = 8000H - AH = 7FF6H = 32758

32-bit operation: When the operation result exceeds -2147483648 to 2147483647

Example: (S1+1, S1) = 1000, (S3+1, S3) = -2147483648, output (D+1, D) = -2147483648 - 1000 = 80000000H - 000003E8H = 7FFFFC18H = 2147482648

(2) When the command is running, if the lower limit of the dead zone is greater than the upper limit of the dead zone, error 6706 will be reported.

2.18.3 ZONE/Zone Control Instructions

An instruction that controls the output value by a specified bias value, depending on whether the input value is positive or negative.

1. Instruction format

FNC258	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
16-bit	9	ZONE	[(D)ZONE(P) S1 S2 S3 D]
		ZONEP	
32-bit	17	DZONE	
		DZONEP	

Operand Description

Operands	Content	Type
S1	Negative deviation	BIN16/32 bits
S2	Positive deviation	BIN16/32 bits
S3	Input value to be controlled by region	BIN16/32 bits
D	Output value controlled by region	BIN16/32 bits

Available operand component types

Operands	Bit Components							Word component								Address Indexing		constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
S1								*	*	*	*	*	*	*	*			*	*	*			
S2								*	*	*	*	*	*	*	*			*	*	*			
S3								*	*	*	*	*	*	*	*			*					
D									*	*	*	*	*	*	*			*					

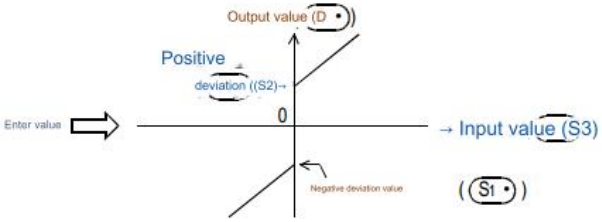
2. Function and action description

Add the deviation value specified by S1 or S2 to the input value specified by S3, and then save it to the soft element number specified by D.

the input value is < 0, the output value = input value + negative deviation value;

When input value = 0, output value = 0;

the input value > 0, the output value = input value + positive bias value.



3. Program example

Slightly.

4. Notes

(1) When the output value overflows, the output value = input value + negative deviation value

16-bit operation: When the operation result exceeds -32768~32767

Example: S1 = -100, S3 = -32768, output D = -32768 + (-100) = 8000H + FF9CH = 7F9CH = 32668

32-bit operation: When the operation result exceeds -2147483648 to 2147483647

Example: (S1+1, S1) = -1000, (S3+1, S3) = -2147483648, output (D+1, D) = -2147483648 + (-1000) = 80000000H - FFFFFC18H = 7FFFFC18H = 2147482648

(2) When the command is running, if the negative deviation value is greater than the positive deviation value, error 6706 will be reported.

2.18.4 SCL/Coordinate setting (coordinate data of different points) instruction

the coordinate system composed of the specified data table , the input X value executes the coordinate command and then outputs the Y value .

1. Instruction format

FNC259	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
16-bit	7	SCL	[(D)SCL(P) S1 S2 D]
		SCLP	
32-bit	13	DSCL	
		DSCLP	

Operand Description

Operands	Content	Type
S1	X coordinate point	BIN16/32 bits
S2	Data table that makes up the coordinate system	BIN16/32 bits
D	Output Y coordinate point	BIN16/32 bits

Available operand component types

Operands	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
S1								*	*	*	*	*	*	*	*			*	*	*			
S2														*	*			*					
D									*	*	*	*	*	*	*			*					

2. Function and action description

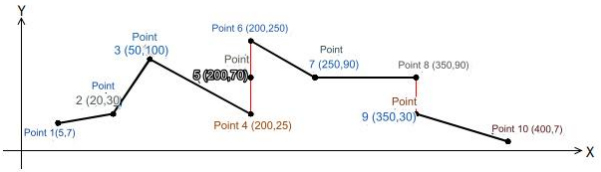
The input value specified by S1 is coordinate-mapped according to the specified conversion characteristics, and then stored in the soft element number specified by D. The conversion for coordinate-mapping is performed based on the data table stored starting from the soft element specified by S2. However, if the output data is not an integer value, the first decimal place is rounded off and then output.

3. Program example

(1) Setting the conversion table for coordinate determination

Setting Project		Set the device allocation of the data table	
		16-bit instruction operation	32-bit instruction operation
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	X coordinate	S2+2n-1	(S2+4n-1, S2+4n-2)
	Y coordinate	S2+2n	(S2+4n+1, S2+4n)

(2) Taking 16-bit operation as an example, when performing 32-bit operation, set the table data according to the 32-bit data format



Setting Project		Setting software and setting Contents			Remark
		When R0 is specified in S2		Setting Content	
Coordinate points		S2	R0	K10	
Point 1	X coordinate	S2+1	R1	K5	
	Y coordinate	S2+2	R2	K7	
Point 2	X coordinate	S2+3	R3	K20	
	Y coordinate	S2+4	R4	K30	
Point 3	X coordinate	S2+5	R5	K50	
	Y coordinate	S2+6	R6	K100	
Point 4	X coordinate	S2+7	R7	K200	If you specify the coordinates of three points like this, the output value is the middle value. In this example, the y coordinate of point 5 is specified as the output value (middle value). In addition, if the x coordinates of more than three points are the same,
	Y coordinate	S2+8	R8	K25	
Point 5	X coordinate	S2+9	R9	K200	
	Y coordinate	S2+10	R10	K70	
Point 6	X coordinate	S2+11	R11	K200	
	Y coordinate	S2+12	R12	K250	
Point 7	X coordinate	S2+13	R13	K250	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+14	R14	K90	
Point 8	X coordinate	S2+15	R15	K350	
	Y coordinate	S2+16	R16	K90	
Point 9	X coordinate	S2+17	R17	K350	
	Y coordinate	S2+18	R18	K30	
Point 10	X coordinate	S2+19	R19	K400	
	Y coordinate	S2+20	R20	K7	

4. Notes

(1) When the X coordinates or Y coordinates are the same, the results are output as written in the table above.

(2) Error 6706 is reported in the following situations:

• When the Xn data in the data table is not arranged in ascending order. However, since the operation is retrieved from the lower side of the soft element number in the data table, even if a part of the data table is not arranged in ascending order, the operation up to this part will not cause an operation error and the instruction will be executed.

• When S1 is outside the range set in the data table.

• When the value during the operation exceeds the range of 32-bit data. In this case, please confirm that the distance between each point does not exceed 65535. If it exceeds 65535, shorten the distance between each point.

2.18.5 DABIN/Decimal ASCII→BIN conversion command

This command converts data displayed in the form of decimal ASCII code (30H to 39H) into BIN data.

1. Instruction format

FNC260	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
16-bit	5	DABIN	[(D)DABIN(P) S D]
		DABINP	
32-bit	9	DDABIN	
		DDABINP	

Operand Description

Operands	Content	Type
S	The starting number of the device that stores the data (ASCII code) to be converted into BIN value	String
D	The device number where the conversion result is stored	BIN16/32 bits

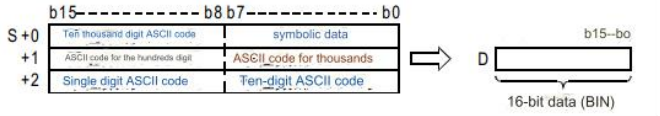
Available operand component types

Operands	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
S1												*	*	*	*			*					
D									*	*	*	*	*	*	*	*	*	*					

2. Function and action description

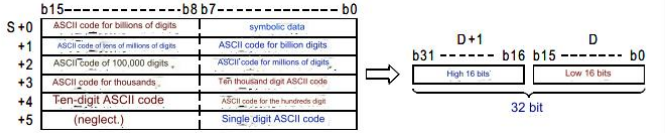
(1) 16-bit instructions

The data from D to D+2 displayed in decimal ASCII code (30H to 39H) is converted into 16-bit data (BIN) and stored in D.



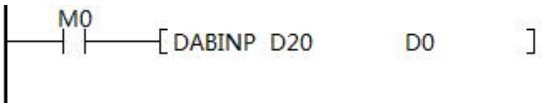
(2) 32-bit instructions

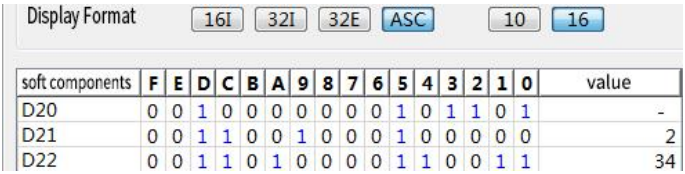
The data from D to D+5 displayed in decimal ASCII code (30H to 39H) is converted into 32-bit data (BIN) and stored in (D+1, D).



3. Program example

When M0 is ON, the program converts the symbol set in D20 to D22 and the 5-digit decimal ASCII code data into BIN values and saves them in D0.





soft components	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0	value
D20	0	0	1	0	0	0	0	0	0	0	1	0	1	1	0	1	-
D21	0	0	1	1	0	0	1	0	0	0	1	0	0	0	0	0	2
D22	0	0	1	1	0	1	0	0	0	0	1	1	0	0	1	1	34

M0 is set , D0 = -234

4. Notes

(1) When the ASCII code of each digit is "20H" or "00H (NULL)", it is treated as "30H".

(2) Error 6706 is reported in the following situations:

• In the sign data (low byte of S), if the data to be converted is positive, set "20H (space)", if it is negative, set "2DH (-)", and an error will be reported if any other value is set.

• When the ASCII code of each digit of S to S+2(5) is a value other than "30H" to "39H", "20H (space)" or "00H (NULL)".

• When the numerical range of S to S+2(5) is outside the following range:

16-bit operation: -32768 to 32767; 32-bit operation: -2147483648 to 2147483647.

• When S to S+2(5) exceeds the soft element range.

2.18.6 BINDA/BIN→Decimal ASCII Conversion Instructions

This command converts BIN data into ASCII code (30H to 39H).

1. Instruction format

FNC261	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
16-bit	5	BINDA	[(D)BINDA(P) S D]
		BINDAP	
32-bit	9	DBINDA	
		DBINDAP	

Operand Description

Operands	Content	Type
S	The device number that stores the BIN data to be converted into ASCII code	BIN16/32 bits
D	starting number of the device to store the conversion results	String

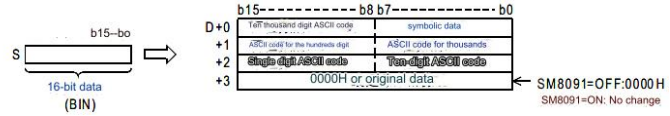
Available operand component types

Operands	Bit Components							Word component								Address Indexing		constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
S								*	*	*	*	*	*	*	*	*	*	*	*	*			
D												*	*	*	*			*					

2. Function and action description

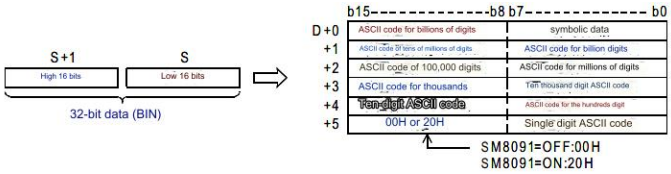
(1) 16-bit instructions

Convert the 16-bit data (BIN) of S into ASCII code (30H~39H) according to the decimal digits, and then save it to D.



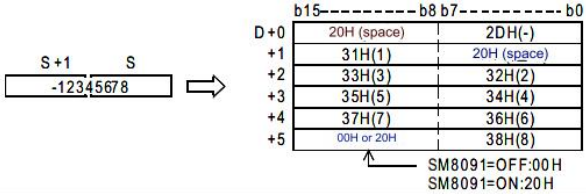
(2) 32-bit instructions

Convert the 32-bit data (BIN) of (S+1, S) into ASCII code (30H to 39H) according to the decimal digits, and then save it to the soft element starting with D.



3. Program example

For example, when (S+1, S) is -12,345,678, the following is stored in the devices starting with D.



4. Notes

(1) Actions of related software components:

Software	name	Content	
		16-bit operation	32-bit operations
M8091	Output character number switching signal	When M8091=OFF, D+3 is 0000H (NULL). When M8091=ON, D+3 does not change.	When M8091=OFF, the high byte of D+5 is 00H (NULL). When M8091=ON, the high byte of D+5 is 20H (space).

It should be noted that when M8091 is OFF during 16-bit operation, 4 word soft elements will be occupied.

(2) In the sign data (low byte of D), "20H (space)" is stored when the data to be converted is positive, and "2DH (-)" is stored when it is negative.

(3) When the total number of digits in S is less than the total number of digits that can be stored in D, "20H (space)" is stored to the left of the effective digit , as in the example above.

(4) The number of points occupied by D cannot exceed the range of the modified soft element. If it exceeds the range, error 6706 will be reported.

2.18.7 SCL2/Set coordinate 2 (X/Y coordinate data) instruction

the coordinate system composed of the specified data table , the input X value executes the coordinate command and then outputs the Y value .

1. Instruction format

FNC269	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
16-bit	7	SCL2	[(D)SCL2(P) S1 S2 D]
		SCL2P	
32-bit	13	DSCL2	
		DSCL2P	

Operand Description

Operands	Content	Type
S1	X coordinate point	BIN16/32 bits
S2	Data table that makes up the coordinate system	BIN16/32 bits
D	Output Y coordinate point	BIN16/32 bits

Available operand component types

Operands	Bit Components							Word component								Address Indexing		constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
S1								*	*	*	*	*	*	*	*			*	*	*			
S2														*	*			*					
D									*	*	*	*	*	*	*			*					

2. Function and action description

The input value specified by S1 is coordinate-mapped according to the specified conversion characteristics, and then stored in the soft element number specified by D. The conversion for coordinate-mapping is performed based on the data table stored starting from the soft element specified by S2. However, if the output data is not an integer value, the first decimal place is rounded off and then output.

3. Program example

(1) Setting the conversion table for coordinate determination

Setting Project		Set the device allocation of the data table	
		16-bit instruction operation	32-bit instruction operation
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+n+2	(S2+2n+5, S2+2n+4)
	...	...	...
	Point n (last)	S2+2n	(S2+4n+1, S2+4n)

(2) Taking 16-bit operation as an example, when performing 32-bit operation, set the table data according to the 32-bit data format

Setting Project		Setting software and setting Contents			Remark
		When R0 is specified in S2		Setting Content	
Coordinate points		S2	R0	K10	
X coordinate	Point 1	S2+1	R1	K5	
	Point 2	S2+2	R2	K20	
	Point 3	S2+3	R3	K50	
	Point 4	S2+4	R4	K200	*1
	Point 5	S2+5	R5	K200	
	Point 6	S2+6	R6	K200	
	Point 7	S2+7	R7	K250	
	Point 8	S2+8	R8	K350	*2
	Point 9	S2+9	R9	K350	
	Point 10	S2+10	R10	K400	
Y coordinate	Point 1	S2+11	R11	K7	
	Point 2	S2+12	R12	K30	
	Point 3	S2+13	R13	K100	
	Point 4	S2+14	R14	K25	*1
	Point 5	S2+15	R15	K70	
	Point 6	S2+16	R16	K250	
	Point 7	S2+17	R17	K90	
	Point 8	S2+18	R18	K90	*2
	Point 9	S2+19	R19	K30	
	Point 10	S2+20	R20	K7	

*1. If the coordinates of three points are specified as points 4, 5, and 6, the output value is the middle value. In this example, the y coordinate of point 5 is specified as the output value (middle value). In addition, even if the X coordinates of three or more points are the same, the value of the second point is output.

*2. If the coordinates of two points are specified as points 8 and 9, the output value is the y coordinate value of the next point. In this example, the y coordinate of point 9 is specified as the output value.

4. Notes

(1) When the X coordinates or Y coordinates are the same, the results are output as written in the routine.

(2) Error 6706 is reported in the following situations:

• When the Xn data in the data table is not arranged in ascending order. However, since the operation is retrieved from the lower side of the soft element number in the data table,

even if a part of the data table is not arranged in ascending order, the operation up to this part will not cause an operation error and the instruction will be executed.

• When S1 is outside the range set in the data table.

• When the value during the operation exceeds the range of 32-bit data. In this case, please confirm that the distance between each point does not exceed 65535. If it exceeds 65535, shorten the distance between each point.

2.19 Convenient instructions

Convenient instructions for realizing complex control with minimal sequence programs.

2.19.1 ABSD/Cam control absolute mode command

Generate multiple output mode instructions based on the current value of the counter.

1. Instruction format

FNC62	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
16-bit	9	ABSD	[(D)ABSD S1 S2 D n]
		\	
32-bit	17	DABSD	
		\	

Operand Description

Operands	Content	Type
S1	Data table start device	BIN16
S2	counter	BIN16
D	Output start bit element	Bit
n	Number of output bit element points/number of data table data groups (1≤n≤64)	BIN16

Available operand component types

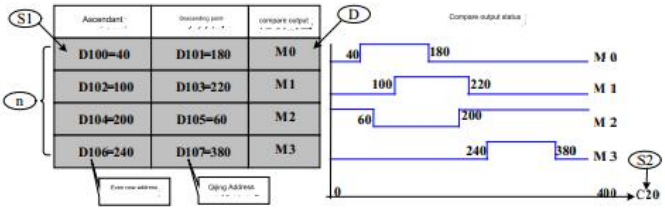
Operands	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
S1								*	*	*	*	*	*	*	*			*					
S2													*					*					
D		*	*			*	*											*					
n																			*	*			

2. Function and action description

Compare the data table data stored in S1 as the starting address with the current data in counter S2, and control the bit element with D as the starting address to be ON/OFF according to the comparison result. The operation completed by this instruction is multi-segment comparison, which is used to realize cam control. The comparison table, counter, etc. are all set in absolute mode.

3. Program Example

When X10 is ON, the execution results are as follows:



4. Notes

(1) This instruction is scanned and executed in the main program, and the comparison result is affected by the lag of the scan time.

(2) When a high-speed counter is used in the DABSD instruction, the comparison output D is also affected by the lag of the user program scan time. For applications that require timely response, the HSZ high-speed comparison instruction can be used.

2.19.2 INCD/Cam control relative mode instruction

Use two consecutive counters to generate instructions for multiple output patterns.

1. Instruction format

FNC63	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
16-bit	9	INCD \	[INCD S1 S2 D n]

Operand Description

Operands	Content	Type
S1	Data table start device	BIN16
S2	Counter, occupies 2, the counter corresponding to the adjacent S2+1 is used to count the number of times the counter is reset after the comparison match	BIN16
D	Output start bit element	Bit
n	Number of output bit element points/number of data table data groups (1≤n≤64)	BIN16

Available operand component types

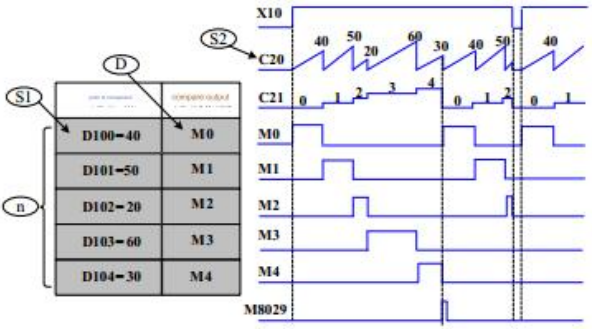
Operands	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
S1								*	*	*	*	*	*	*	*			*					
S2													*					*					
D		*	*			*	*											*					
n																			*	*			

2. Function and action description

Compare the data table data stored at the starting address S1 with the current data in the counter S2, and control the bit element at the starting address D to be ON/OFF according to the comparison result. The operation completed by this instruction is multi-segment comparison, which is used to realize cam control. The comparison table, counter, etc. are all set in increments.

3. Program example

When X10 is ON, the execution results are as follows:



4. Notes

(1) This instruction is scanned and executed in the main program. The comparison result is affected by the lag of the scan time. For applications that require timely response, the HSZ high-speed comparison instruction can be used.

(2) After the comparison of the number of groups of n is completed, the instruction execution completion flag SM8029 will be set.

2.19.3 ALT/Alternate output command

Instruction for reversing the status of bit devices.

1. Instruction format

FNC66	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
16-bit	3	ALT	-[ALT(P) D]
		ALTP	

Operand Description

Operands	Content	Type
D	Alternate output bit device	Bit

Available operand component types

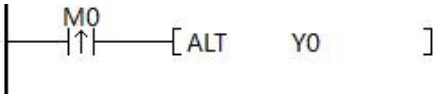
Operands	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
D		*	*			*	*											*					

2. Function and action description

Each time the command input changes from OFF to ON, the bit device specified in D reverses its status.

3. Program example

Each time the rising edge of M0 is triggered, the state of Y0 will be reversed



4. Notes

When programming with the ALT instruction, a reversal action is performed in each operation cycle, and a contact instruction is used to control the output state reversal.

2.19.4 SORT/Data sorting instructions

This instruction is used to rearrange the data table consisting of data rows and data columns in ascending order based on the specified column and row unit.

1. Instruction format

FNC69	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
16-bit	11	SORT \	SORT S m1 m2 D n

Operand Description

Operands	Content	Type
S	Source data table [occupies m1×m2 points]	BIN16 bits
m1	Number of rows [1～32]	
m2	Number of columns [1～6]	
D	Save the sorted table [occupies m1×m2 points]	
n	Column number used as sorting standard [1～m2]	

Available operand component types

Operands	Bit Components							Word component							Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
S														*	*								
m1														*	*				*	*			
m2														*	*				*	*			
D																							
n														*	*				*	*			

2. Function and action description

In the data table (before sorting) at (m1×m2) points starting from S , rearrange the data rows in ascending order based on the group data of n columns, and then save them in the data table (after sorting) at (m1×m2) points starting from D. After the instruction is executed, SM8029 will be set, and this flag can be used to reset the input execution switch in front of SORT.

3. Program example

In the case of "n=K2 (column number 2)" and "n=K3 (column number 3)", the data before sorting shown below is sorted, and the operation is as follows. If a sequential number such as a management number is entered in the first column, the original row number can be determined from its Content.

(1) Data before sorting:

Column Number Line Number		Number of groups: m2 (when m2=K4)			
		1	2	3	4
		Management Number	height	weight	age
When the number of data m1=K5	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

(2) Sorting results when executing instructions based on n=K2 (column number 2)

Column Number Line Number		Number of groups: m2 (when m2=K4)			
		1	2	3	4
		Management Number	height	weight	age
When the number of data m1=K5	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+1 4	D+19
		2	180	50	40

(3) Sorting results when executing instructions based on n=K3 (column number 3)

Column Number Line Number		Number of groups: m2 (when m2=K4)			
		1	2	3	4
		Management Number	height	weight	age
When the number of data m1=K5	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+1 3	D+1 8
		5	150	50	45
	5	D+ 4	D+ 9	D+1 4	D+19
		3	1 60	70	30

4. Notes

(1) Operation of related software components

S M8029: instruction end flag, set when the sequence is completed;

(2) The following operations will cause incorrect sorting, please note

• Do not change the operands and data Contents during the operation.

• When executing again, please input OFF the instruction once.

• It can only be used once in the program .

• The original data and the data after sorting should be stored in different positions to avoid overlap.

2.20 Interpolation instructions

Interpolation instructions support absolute position and relative position interpolation. The supported interpolation trajectory types include: two-axis linear interpolation, two-axis clockwise circular interpolation, and two-axis counterclockwise circular interpolation.

(1) Instruction introduction

Descriptors	illustrate	Remark
G90G01	Two-axis linear absolute position interpolation	Version 2118 (32-bit instructions)
G91G01	Two-axis linear relative position interpolation	Version 2118 (32-bit instructions)
G90G02	Two-axis circular absolute position interpolation	Version 2119 and above (32-bit instructions)
G91G02	Relative position interpolation of two axes along circular arc	Version 2119 and above (32-bit instructions)
G90G03	Two-axis reverse arc absolute position interpolation	Version 2119 and above (32-bit instructions)
G91G03	Relative position interpolation of two axes in reverse arc	Version 2119 and above (32-bit instructions)
ABSLINE	Three-axis linear absolute position interpolation	2137 and above (32-bit instructions)
RELLINE	Three-axis linear relative position interpolation	2137 and above (32-bit instructions)

(2) X3 interpolation related software components

Y0 terminal corresponding register	Y1 terminal corresponding register	effect	Initial Value
SM8029		Instruction execution end flag	
SM8329		Abnormal end of instruction flag	
SM8340	SM8350	Pulse output monitoring	
SM8343	SM8353	Forward limit	
SM8344	SM8354	Reversal Limit	
SM8348	SM8358	Positioning command driving	
SM8349	SM8359	Pulse stop command (emergency stop switch)	
SD8340	SD8350	Current value register	0
SD8341	SD8351		
SD8342	SD8352	Base speed	800
SD8343	SD8353	Top speed	200,000
SD8344	SD8354		
SD8348	SD8358	Acceleration time [ms]	100
SD8349	SD8359	Deceleration time [ms]	100

2.20.1 G90G01 Two-axis linear absolute position interpolation

The two axes output pulses at the set synthesis frequency to complete the set interpolation trajectory. Based on absolute position.

1. Instruction Format

FNC281	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
32-bit	twenty one	G90G01 \	G90G01 S1 S2 S D1 D2

Operand Description

Operands	Content	Type
S1	Number of X-axis pulses, absolute value of X-axis target position	BIN32 bits
S2	Number of Y-axis pulses, absolute value of Y-axis target position	BIN32 bits
S	Output frequency, synthesis frequency	BIN32 bits
D1	Output port, occupies D1 and D1+1	Bit
D2	Output direction, occupies D2 and D2+1	Bit

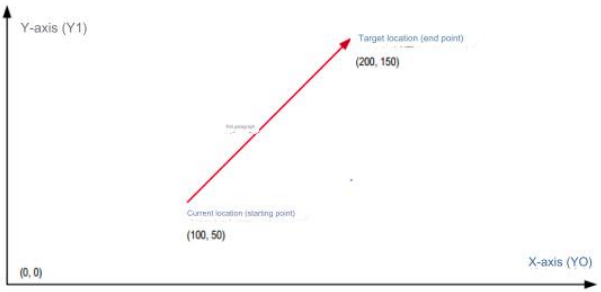
Available operand component types

Operands	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
S1														*	*				*	*			
S2														*	*				*	*			
S														*	*				*	*			
D1		*																					
D2		*																					

2. Function and action description

- S1 specifies the absolute position of the X-axis target, ranging from -2,147,483,648 to 2,147,483,647.
- S2 specifies the absolute position of the Y-axis target, ranging from -2,147,483,648 to 2,147,483,647.
- S specifies the XY synthesis frequency, ranging from 50 to 280,000 Hz; and the frequency after decomposition must not be greater than 200K.
- D1 specifies the pulse output port, occupying two consecutive high-speed terminals.
- D2 specifies the output direction and occupies two consecutive body terminals.

3. Program example



Assuming the current position is (100K, 50K), it means linear interpolation from the current position to the position (200K, 150K), the synthesis speed is 100KHz. Use Y0/Y1 as the pulse direction output port respectively.

4. Notes

(1) When using the interpolation command, the parameter setting is based on the X-axis (Y0).

(2) When the pulses are not completely sent and the command is disconnected, the machine will decelerate and stop at the set deceleration rate.

(3) When encountering any limit switch or emergency stop switch of the two axes, it will stop immediately.

2.20.2 G91G01 Two-axis linear relative position interpolation

The two axes output pulses at the set synthesis frequency to complete the set interpolation trajectory. Based on relative position.

1. Instruction format

FNC282	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
32-bit	twenty one	G91G01 \	G91G01 S1 S2 S D1 D2

Operand Description

Operands	Content	Type
S1	Number of X-axis pulses, relative value of X-axis target position	BIN32 bits
S2	Number of Y-axis pulses, relative value of Y-axis target position	BIN32 bits
S	Output frequency, synthesis frequency	BIN32 bits
D1	Output port, occupies D1 and D1+1	BIN32 bits
D2	Output direction, occupies D2 and D2+1	BIN32 bits

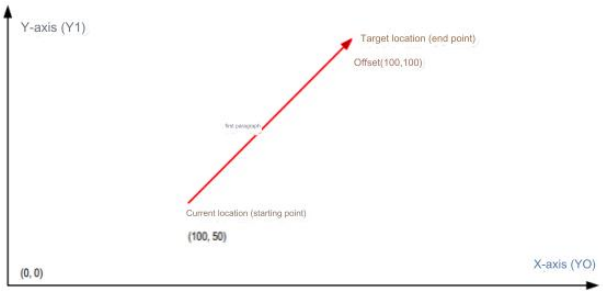
Available operand component types

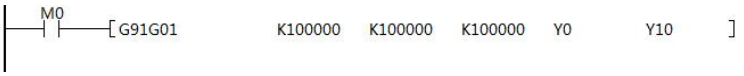
Operands	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
S1														*	*				*	*			
S2														*	*				*	*			
S														*	*				*	*			
D1		*																					
D2		*																					

2. Function and action description

- S1 specifies the relative position of the X-axis target. The range is -2,147,483,648 to 2,147,483,647.
- S2 specifies the relative position of the Y-axis target, ranging from -2,147,483,648 to 2,147,483,647.
- S specifies the XY synthesis frequency, ranging from 50 to 280,000 Hz; and the frequency after decomposition must not be greater than 200K.
- D1 specifies the pulse output port, occupying two consecutive high-speed terminals.
- D2 specifies the output direction and occupies two consecutive body terminals .

3. Program example





Assuming the current position is (100K, 50K), it means linear interpolation from the current position, offset (100K, 100K), that is, the position of (200K, 150K), the synthetic speed is 100KHz. Use Y0/Y1 as pulse output ports, and Y10/Y11 as pulse direction output ports.

4. Notes

(1) When using the interpolation command, the parameter setting is based on the X-axis (Y0).

(2) When the pulses are not completely sent and the command is disconnected, the machine will decelerate and stop at the set deceleration rate.

(3) When encountering any limit switch or emergency stop switch of the two axes, it will stop immediately.

2.20.3 G90G02 Two-axis circular absolute position interpolation

The two axes output pulses at the set synthesis frequency to complete the set interpolation trajectory. Based on absolute position.

1. Instruction format

FNC283	16bit		32bit 29 steps		Instruction Format
G90G02	\	\	G90G02	\	G90G02 S1 S2 S3 S4 S D1 D2 1D

FNC283	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
32-bit	29	G90G02 \	G90G02 S1 S2 S3 S4 S D1 D2

Operand Description

Operands	Content	Type
S1	Number of X-axis pulses, absolute value of X-axis target position	BIN32 bits
S2	Number of Y-axis pulses, absolute value of Y-axis target position	BIN32 bits
S3	X center coordinate, absolute pulse number of X axis of the center	BIN32 bits

Operands	Content	Type
S4	Y center coordinate, absolute pulse number of Y axis of the center	BIN32 bits
S	Output frequency, synthetic frequency	BIN32 bits
D1	Output port, occupies D1 and D1+1	BIN32 bits
D2	Output direction, occupies D2 and D2+1	BIN32 bits

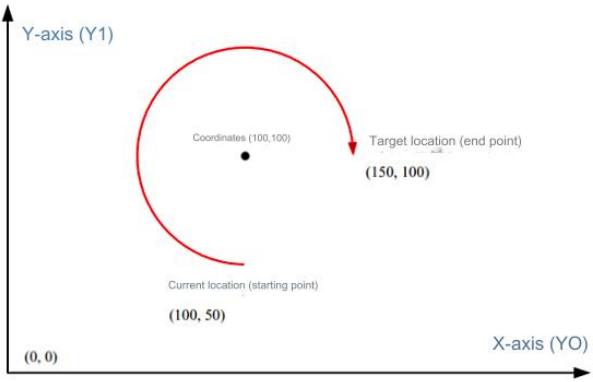
Available operand component types

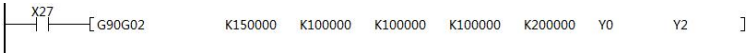
Operands	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
S1														*	*				*	*			
S2														*	*				*	*			
S3														*	*				*	*			
S4														*	*				*	*			
S														*	*				*	*			
D1		*																					
D2		*																					

2. Function and action description

- S1 specifies the absolute position of the X-axis target, ranging from -2,147,483,648 to 2,147,483,647.
- S2 specifies the absolute position of the Y-axis target, ranging from -2,147,483,648 to 2,147,483,647.
- S3 specifies the absolute X- axis target position of the circle center. The range is -2,147,483,648 to 2,147,483,647.
- S4 specifies the absolute Y-axis target position of the circle center, ranging from -2,147,483,648 to 2,147,483,647.
- S specifies the XY synthesis frequency, ranging from 50 to 200,000 Hz; and the frequency after decomposition must not be greater than 200K.
- D1 specifies the pulse output port, occupying two consecutive high-speed terminals.
- D2 specifies the output direction and occupies two consecutive body terminals .

3. Program example





Assuming the current position is (100K, 50K), it means that the arc interpolation is from the current position, the target point is (150K, 100K), and the coordinates (100k , 100K) are used to draw a circle clockwise, and the synthesis speed is 200KHz. Use Y0/Y1 as the pulse output port, and Y2/Y3 as the pulse direction output port.

4. Notes

(1) When using the interpolation command, the parameter setting is based on the X-axis (Y0).

(2) When the pulses are not completely sent and the command is disconnected, the machine will decelerate and stop at the set deceleration rate.

(3) When encountering any limit switch or emergency stop switch of the two axes, it will stop immediately.

2.20.4 G91G02 Two-axis circular relative position interpolation

The two axes output pulses at the set synthesis frequency to complete the set interpolation trajectory. Based on relative position.

1. Instruction format

FNC284	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
32-bit	29	G9 1 G02 \	G91G02 S1 S2 S3 S4 S D1 D2

Operand Description

Operands	Content	Type
S1	Number of X-axis pulses, relative value of X-axis target position	BIN32 bits
S2	Number of Y-axis pulses, relative value of Y-axis target position	BIN32 bits
S3	X center coordinate, relative pulse number of X axis of center	BIN32 bits
S4	Y center coordinate, relative pulse number of Y axis of center	BIN32 bits
S	Output frequency, synthetic frequency	BIN32 bits
D1	Output port, occupies D1 and D1+1	BIN32 bits
D2	Output direction, occupies D2 and D2+1	BIN32 bits

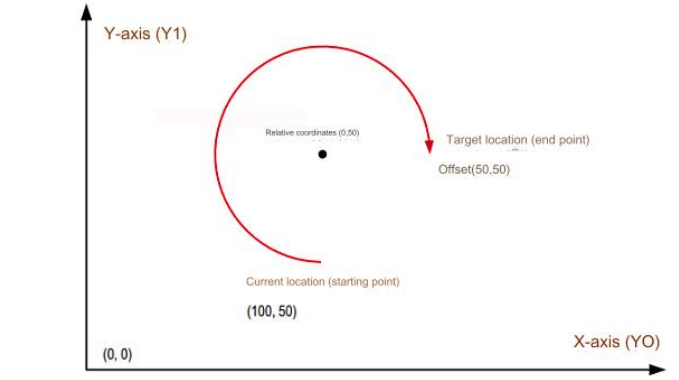
Available operand component types

Operands	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
S1														*	*				*	*			
S2														*	*				*	*			
S3														*	*				*	*			
S4														*	*				*	*			
S														*	*				*	*			
D1		*																					
D2		*																					

2. Function and action description

- S1 specifies the relative position of the X-axis target. The range is -2,147,483,648 to 2,147,483,647.
- S2 specifies the relative position of the Y-axis target, ranging from -2,147,483,648 to 2,147,483,647.
- S3 specifies the relative position of the X- axis target of the circle center. The range is -2,147,483,648~2,147,483,647.
- S4 specifies the relative position of the Y axis target of the circle center. The range is -2,147,483,648~2,147,483,647.
- S specifies the XY synthesis frequency, ranging from 50 to 200,000 Hz; and the frequency after decomposition must not be greater than 200K.
- D1 specifies the pulse output port, occupying two consecutive high-speed terminals.
- D2 specifies the output direction and occupies two consecutive body terminals .

3. Program example



Assuming the current position is (100K, 50K), it means that the arc interpolation from the current position is relative to the target point (50K, 50K), and the circle is drawn clockwise with the relative coordinates (0, 50K), and the synthesis speed is 200KHz. Use Y0/Y1 as the pulse output port, and Y2/Y3 as the pulse direction output port.

4. Notes

(1) When using the interpolation command, the parameter setting is based on the X-axis (Y0).

(2) When the pulses are not completely sent and the command is disconnected, the machine will decelerate and stop at the set deceleration rate.

(3) When encountering any limit switch or emergency stop switch of the two axes, it will stop immediately.

(4) When the target point is not on the circle, error 6706 will be reported.

(5) When the target point is (0, 0), a full circle will be drawn.

2.20.5 G90G03 Two-axis counter-circular absolute position interpolation

The two axes output pulses at the set synthesis frequency to complete the set interpolation trajectory. Based on absolute position.

1. Instruction format

FNC285	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
32-bit	29	G9 0 G0 3 \	-[G90G03 S1 S2 S3 S4 S D1 D2]

Operand Description

Operands	Content	Type
S1	Number of X-axis pulses, absolute value of X-axis target position	BIN32 bits
S2	Number of Y-axis pulses, absolute value of Y-axis target position	BIN32 bits
S3	X center coordinate, absolute pulse number of X axis of the center	BIN32 bits
S4	Y center coordinate, absolute pulse number of Y axis of the center	BIN32 bits
S	Output frequency, synthetic frequency	BIN32 bits
D1	Output port, occupies D1 and D1+1	BIN32 bits
D2	Output direction, occupies D2 and D2+1	BIN32 bits

Available operand component types

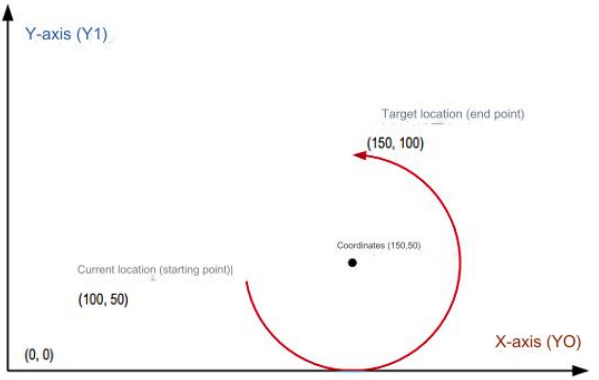
Operands	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
S1														*	*				*	*			
S2														*	*				*	*			
S3														*	*				*	*			
S4														*	*				*	*			
S														*	*				*	*			
D1		*																					
D2		*																					

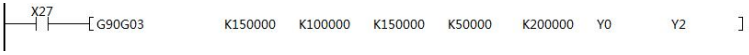
2. Function and action description

- S1 specifies the absolute position of the X-axis target, ranging from -2,147,483,648 to 2,147,483,647.
- S2 specifies the absolute position of the Y-axis target, ranging from -2,147,483,648 to 2,147,483,647.
- S3 specifies the absolute X- axis target position of the circle center. The range is -2,147,483,648 to 2,147,483,647.
- S4 specifies the absolute Y-axis target position of the circle center, ranging from -2,147,483,648 to 2,147,483,647.

- S specifies the XY synthesis frequency, ranging from 50 to 200,000 Hz; and the frequency after decomposition must not be greater than 200K.
- D1 specifies the pulse output port, occupying two consecutive high-speed terminals.
- D2 specifies the output direction and occupies two consecutive body terminals .

3. Program example





Assuming the current position is (100K, 50K), it means that the arc interpolation is from the current position, the target point is (150K, 100K), and the coordinates (150K , 50K) are used to draw a circle counterclockwise, and the synthesis speed is 200KHz. Use Y0/Y1 as the pulse output port, and Y2/Y3 as the pulse direction output port.

4. Notes

(1) When using the interpolation command, the parameter setting is based on the X-axis (Y0).

(2) When the pulses are not completely sent and the command is disconnected, the machine will decelerate and stop at the set deceleration rate.

(3) When encountering any limit switch or emergency stop switch of the two axes, it will stop immediately.

(4) When the target point is not on the circle, error 6706 will be reported.

(5) When the target point is (0, 0), a full circle will be drawn.

2.20.6 G91G03 Two-axis counter-circular relative position interpolation

The two axes output pulses at the set synthesis frequency to complete the set interpolation trajectory. Based on relative position.

1. Instruction format

FNC286	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
32-bit	29	G9 1 G0 3 \	[G91G03 S1 S2 S3 S4 S D1 D2]

Operand Description

Operands	Content	Type
S1	Number of X-axis pulses, relative value of X-axis target position	BIN32 bits
S2	Number of Y-axis pulses, relative value of Y-axis target position	BIN32 bits
S3	X center coordinate, relative pulse number of X axis of center	BIN32 bits
S4	Y center coordinate, relative pulse number of Y axis of center	BIN32 bits
S	Output frequency, synthetic frequency	BIN32 bits
D1	Output port, occupies D1 and D1+1	BIN32 bits
D2	Output direction, occupies D2 and D2+1	BIN32 bits

Available operand component types

Operands	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
S1														*	*				*	*			
S2														*	*				*	*			
S3														*	*				*	*			
S4														*	*				*	*			
S														*	*				*	*			
D1		*																					
D2		*																					

2. Function and action description

- S1 specifies the relative position of the X-axis target. The range is -2,147,483,648 to 2,147,483,647.
- S2 specifies the relative position of the Y-axis target, ranging from -2,147,483,648 to 2,147,483,647.
- S3 specifies the relative position of the X- axis target of the circle center. The range is -2,147,483,648~2,147,483,647.

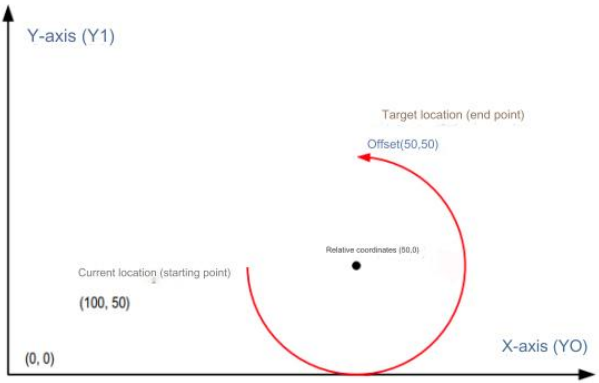
●S4 specifies the relative position of the Y axis target of the circle center. The range is -2,147,483,648~2,147,483,647.

●S specifies the XY synthesis frequency, ranging from 50 to 200,000 Hz; and the frequency after decomposition must not be greater than 200K.

●D1 specifies the pulse output port, occupying two consecutive high-speed terminals.

●D2 specifies the output direction and occupies two consecutive body terminals .

3. Program example



Assuming the current position is (100K, 50K), it means that the arc interpolation from the current position is relative to the target point (50K, 50K), and the relative coordinate (50K, 0) is used to draw a circle counterclockwise, and the synthesis speed is 200KHz. Use Y0/Y1 as the pulse output port, and Y2/Y3 as the pulse direction output port.

4. Notes

(1) When using the interpolation command, the parameter setting is based on the X-axis (Y0).

(2) When the pulses are not completely sent and the command is disconnected, the machine will decelerate and stop at the set deceleration rate.

(3) When encountering any limit switch or emergency stop switch of the two axes, it will stop immediately.

(4) When the target point is not on the circle, error 6706 will be reported.

(5) When the target point is (0, 0), a full circle will be drawn.

2.20.7 ABSLINE Three-axis linear absolute position interpolation

The three axes output pulses at the set synthesis frequency to complete the set interpolation trajectory. Based on absolute position.

1. Instruction format

FNC306	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
32-bit	29	ABSLINE \	ABSLINE S1 S2 S3 S D1 D2 D3

Operand Description

Operands	Content	Type
S1	Number of X-axis pulses, absolute value of X-axis target position	BIN32 bits
S2	Number of Y-axis pulses, absolute value of Y-axis target position	BIN32 bits
S3	Z -axis pulses, the absolute value of the Z -axis target position	BIN32 bits
S	Output frequency, synthesis frequency	BIN32 bits
D1	X-axis pulse output port , direction is fixed to D1+output pulse number	Bit
D2	Y-axis pulse output port , direction is fixed to D2+output pulse number	Bit
D3	Z-axis pulse output port , direction is fixed to D3+output pulse number	Bit

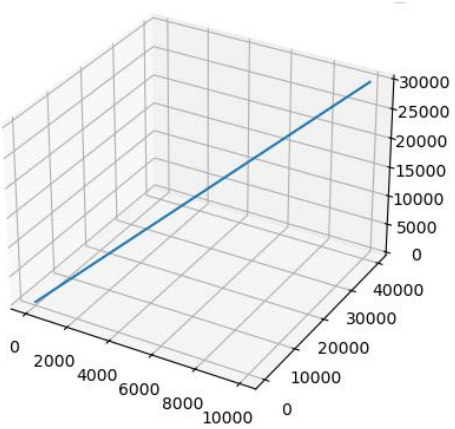
Available operand component types

Operands	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
S1														*	*				*	*			
S2														*	*				*	*			
S3														*	*				*	*			
S4														*	*				*	*			
S														*	*				*	*			
D1		*																					
D2		*																					
D3		*																					

2. Function and action description

- S1 specifies the absolute position of the X-axis target, ranging from -2,147,483,648 to 2,147,483,647.
- S2 specifies the absolute position of the Y-axis target, ranging from -2,147,483,648 to 2,147,483,647.
- S3 specifies the absolute target position of the Z axis. The range is -2,147,483,648 to 2,147,483,647.
- S specifies the XY Z composite frequency, ranging from 50 to 346410 Hz; and the frequency after decomposition must not be greater than 200K.
- D1 specifies the X-axis pulse output port, which can only be Y0 -Y (maximum number of pulse paths - 1) .
- D2 specifies the Y-axis pulse output port, which can only be Y0 -Y (maximum number of pulse paths - 1) .
- D3 specifies the Z-axis pulse output port, which can only be Y0 -Y (maximum number of pulse paths - 1). You can fill in Y255 to omit this parameter .

3. Program example



4. Notes

(1) When using interpolation instructions, the parameter setting is based on the X axis.

(2) When the pulses are not completely sent and the command is disconnected, the machine will decelerate and stop at the set deceleration rate.

(3) When encountering any limit switch or emergency stop switch of the three axes, it will stop immediately.

2.20.8 RELLINE Three-axis linear relative position interpolation

The three axes output pulses at the set synthesis frequency to complete the set interpolation trajectory. Based on relative position.

1. Instruction format

FNC307	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
32-bit	29	RELLINE \	RELLINE S1 S2 S3 S D1 D2 D3

Operand Description

Operands	Content	Type
S1	Number of X-axis pulses, relative value of X-axis target position	BIN32 bits
S2	Number of Y-axis pulses, relative value of Y-axis target position	BIN32 bits
S3	Z -axis pulses, the relative value of the Z -axis target position	BIN32 bits
S	Output frequency, synthesis frequency	BIN32 bits
D1	X-axis pulse output port , direction is fixed to D1+output pulse number	Bit
D2	Y-axis pulse output port , direction is fixed to D2+output pulse number	Bit
D3	Z-axis pulse output port , direction is fixed to D3+output pulse number	Bit

Available operand component types

Operands	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
S1														*	*				*	*			
S2														*	*				*	*			
S3														*	*				*	*			
S4														*	*				*	*			
S														*	*				*	*			
D1		*																					
D2		*																					
D3		*																					

2. Function and action description

- S1 specifies the absolute position of the X-axis target, ranging from -2,147,483,648 to 2,147,483,647.
- S2 specifies the absolute position of the Y-axis target, ranging from -2,147,483,648 to 2,147,483,647.
- S 3 specifies the absolute target position of the Z axis. The range is -2,147,483,648 to 2,147,483,647.
- S specifies the XY Z composite frequency, ranging from 50 to 346410 Hz; and the frequency after decomposition must not be greater than 200K.
- D1 specifies the X-axis pulse output port, which can only be Y0 -Y (maximum number of pulse paths - 1) .
- D 2 specifies the Y-axis pulse output port, which can only be Y0 -Y (maximum number of pulse paths - 1) .
- D 3 specifies the Z-axis pulse output port, which can only be Y0 -Y (maximum number of pulse paths - 1). You can fill in Y255 to omit this parameter .

3. Program example

4. Notes

(1) When using interpolation instructions, the parameter setting is based on the X axis.

(2) When the pulses are not completely sent and the command is disconnected, the machine will decelerate and stop at the set deceleration rate.

(3) When encountering any limit switch or emergency stop switch of the three axes, it will stop immediately.

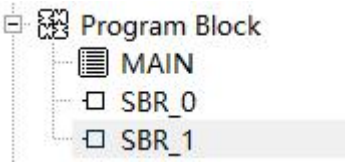
3. Application

3.1 Subroutines

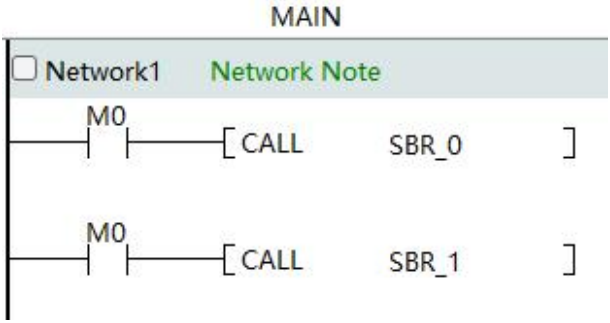
Action description: When the instruction input is ON, the CALL instruction is executed and jumps to the step marked P. Then, the subroutine marked P is executed, and after executing SRET, it returns to the next step of the CALL instruction . An example is shown below.

Subroutine programming method:

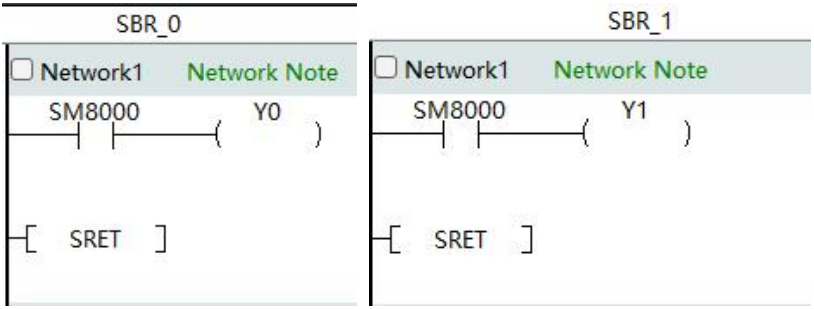
①Right click on the program block to insert a subroutine



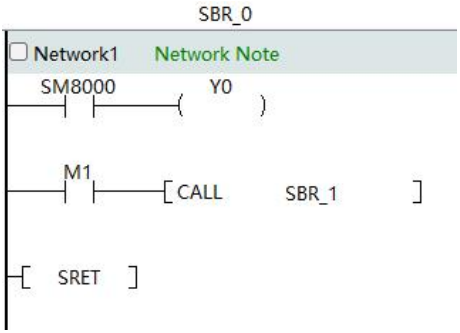
② Write the subroutine name after CALL, where the label after SBR corresponds to the P label



③Subroutine, with SRET subroutine return



other subroutines to the subroutines



Note:

(1) Program with FEND instruction at the end of the main program;

(2) When writing a subroutine in label form, the label P used for the CALL instruction must be programmed after the FEND instruction;

(3) A maximum of 6 nesting levels are allowed. If the number exceeds 6, a shutdown error 6634 will be reported.

(4) There is no SRET instruction in the subroutine, and XPLC reports 6637 and stops running.

Regarding the process ladder diagram, XPLC will find the error during self-check and shut down the machine.

3.2 Interrupt procedure

Action description: EI is the interrupt enable instruction, and DI is the interrupt disable instruction. After the interrupt source triggers the interrupt, the PLC jumps to the subroutine specified by the interrupt pointer number for execution. IRET is the interrupt return instruction, which is the end instruction of the interrupt handler . Its function is to make the PLC return to the next instruction when it is interrupted and continue to execute. If there is no interrupt handling subroutine, although the interrupt is triggered, nothing is processed.

3.2.1 Interrupt event classification

(1) External input interrupt: 6 points

The external interrupt pointers are labeled as follows:

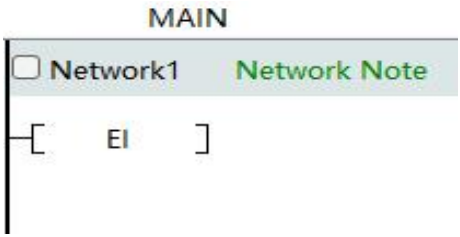
Receive input signals from specific input numbers without being affected by the PLC operation cycle. Trigger the input signal and execute the interrupt subroutine. Since input interrupts can process signals shorter than the operation cycle, they can be used in sequential control when priority processing or short-time pulse processing control is required.

enter	Input interrupt pointer		Disable interrupt flag
	Rising edge interrupt	Falling edge interrupt	
X00	I001	I000	SM8050
X01	I101	I100	SM8051
X02	I201	I200	SM8052
X03	I301	I300	SM8053
X04	I401	I400	SM8054
X05	I501	I500	SM8055

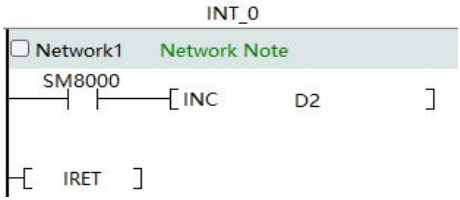
The rising edge and falling edge of the same input in the same ladder diagram can trigger the corresponding external interrupt.

Program example:

Main program:

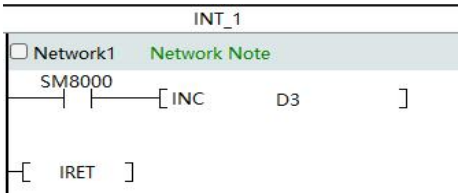


I000 interrupt subroutine:



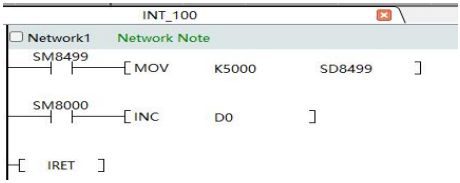
Interrupt description: When a falling edge interrupt occurs on X0, the D2 count increases by 1 immediately.

I001 interrupt subroutine:



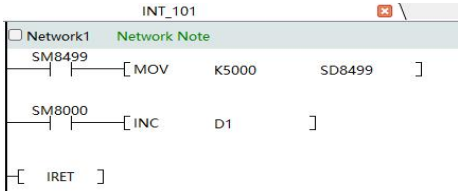
When a rising edge interrupt occurs on X0, the D3 count increases by 1 immediately.

I100 interrupt subroutine:



When a falling edge interrupt occurs on X1, due to the external interrupt delay of SM8499, there will be a delay of SD8499, which is 5S here, and then the D0 count will increase by 1.

I101 interrupt subroutine:



When X1 has a rising edge interrupt, due to the SM8499 external interrupt delay, there will be a delay of SD8499, which is 5S here. After 5S, the D1 count increases by 1.

(2) Timer interrupt: 3 points

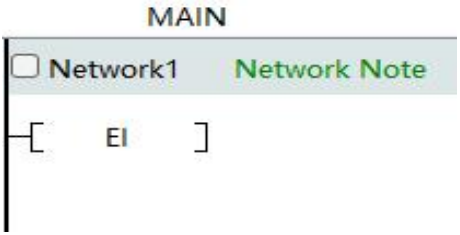
The timer interrupt pointers are numbered as follows:

The interrupt subroutine is executed every specified interrupt cycle time (10ms to 99ms). It is used in controls that require cyclic interrupt processing outside the PLC operation cycle.

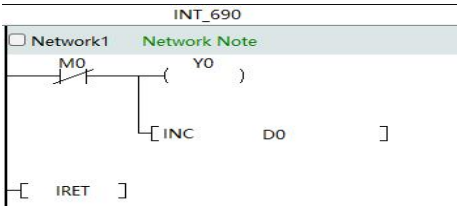
Input number	Interrupt cycle (ms)	Interrupt disable flag
I6XX	In the XX field of the pointer name, enter an integer from 10 to 99. For example: I610 = timer interrupt every 10ms	SM8056
I7XX		SM8057
I8XX		SM8058

Program example:

Main program:

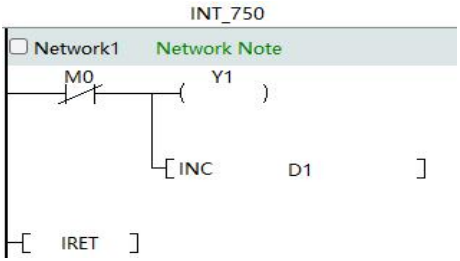


I690 interrupt subroutine:



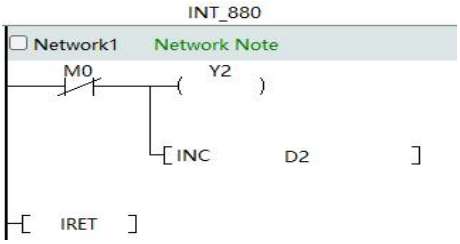
Interrupt description: After timer 1 counts for 90ms, a timer interrupt occurs and the D0 count is immediately increased by 1.

I750 interrupt subroutine:



Interrupt description: After timer 2 counts for 50ms, a timer interrupt occurs and the D1 count is immediately increased by 1.

I880 interrupt subroutine:



Interrupt description: After timer 3 counts for 80ms, a timer interrupt occurs and the D2 count is immediately increased by 1.

(3) Counter interrupt: 6 points

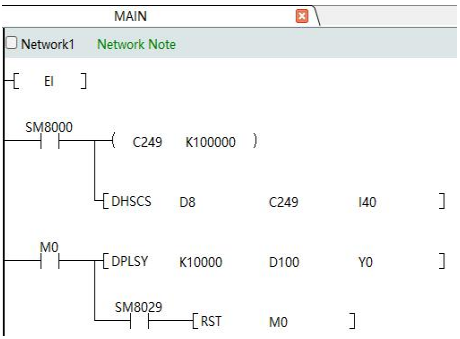
The counter interrupt pointers are labeled as follows:

Executes the interrupt subroutine according to the comparison result of the high-speed counter comparison set instruction (DHSCS instruction). It is used to control the counting result with priority processing using the high-speed counter.

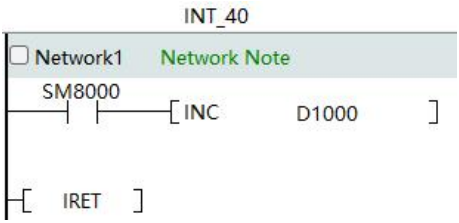
Program example:

Pointer number	Interrupt disable flag
I010	SM8059
I020	
I030	
I040	
I050	
I060	

Main program:



I40 interrupt subroutine:

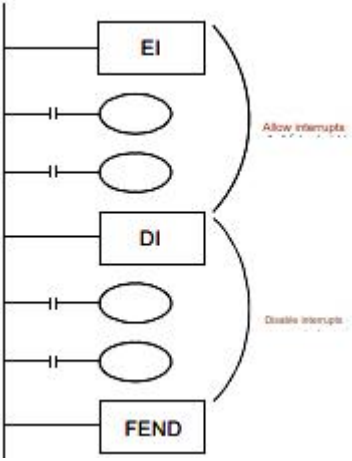


Interrupt description: When the count value of high-speed counter C249 is equal to D8, the counter interrupt I400 will be triggered immediately, and then the value of D1000 will increase by 1.

(4) Interrupt range limitation

The DI instruction can be used to set the interrupt prohibition area. When an interrupt occurs between DI and EI, it will not be executed immediately but the interrupt will be stored and executed after EI is executed.

Program example:

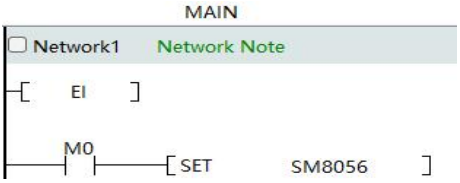


(5) Disable interrupts

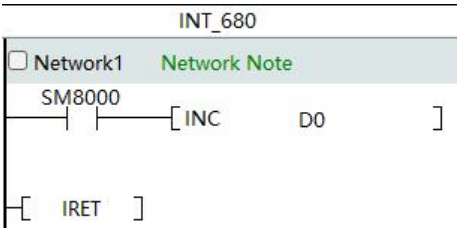
Each interrupt subroutine has its own interrupt disable bit. When the corresponding interrupt disable bit is set, the interrupt will not be executed even if an interrupt occurs.

Program example:

Main program:



I680 interrupt subroutine:



Under normal circumstances, a timer interrupt will occur every 50ms, and the D0 value will increase by 1. When SM8056 is set, the D0 value will no longer increase. After resetting SM8056, an interrupt will occur every 50ms to increase D0 by 1.

3.2.2 Interrupt priority

External input interrupt > Timer interrupt > Counter interrupt

I000 > I100 > I200 > I300 > I400 > I500 > I001 > I101 > I201 > I301 > I401 > I501 > I6xx > I7xx > I8xx > I010 > I020 > I030 > I040 > I050 > I060

3.2.3 Interrupt redirection

When input terminals X00-X07 are occupied by other functions, SM8480~SM8487 can be used to redirect the interrupt function to input terminals X10~X17. Redirection means connecting the input signal to X10~X17 to use the corresponding function. The functions affected by redirection are: external input interrupt, input capture, DVIT

Original Terminal	Redirected terminal	Enable switch
X00	X10	SM8480
X01	X11	SM8481
X02	X12	SM8482
X03	X13	SM8483
X04	X14	SM8484
X05	X15	SM8485
X06	X16	SM8486
X07	X17	SM8487

3.2.4 Notes

(1) If the corresponding interrupt disable flag is set, the function will not be executed even if an interrupt occurs.

(2) Note the action of the SM8499 register . The interrupt program is followed by an LD SM8499 , which means that the interrupt will not be acted upon immediately, but will be delayed for the delay time of SD8499 .

3.3 High-speed counting

Depending on the number of points of PLC, there are different scopes of application.

Points	Specification
16:00	3 AB phase 100K or 3 single phase 200K (up to 3 counters can be used simultaneously)
24 o'clock	4 AB phase 100K or 4 single phase 200K (up to 4 counters can be used simultaneously)
32 points	4 AB phase 100K or 4 single phase 200K (up to 4 counters can be used simultaneously)
40 points	4 AB phase 100K or 4 single phase 200K (up to 4 counters can be used simultaneously)
60 points	4 AB phase 100K or 8 single phase 200K (up to 8 counters can be used simultaneously)

(Note: X2PLC and relay type PLC can use up to 3 counters)

According to the terminals of each point, select the counter of the terminal corresponding mode from the table below.

3.3.1 Classification of high-speed counters supported

U: up count input; D: down count input; R reset input; S: start input;

A: A phase pulse input; B: B phase pulse input

	C235	C236	C237	C238	C239	C240	C241	C242	C243	C244	C245	C246	C247	C248	C249	C250	C251	C252	C253	C254	C255
X0	U/D								U/D			U			U		A	A			
X1		U/D							R			D			D		B	B			
X2			U/D							U/D			U		R		R		A		
X3				U/D						R			D		S				B		
X4					U/D					S				U		U				A	
X5						U/D					U/D			D		D				B	
X6							U/D				R					R					A
X7								U/D			S					S					B

(Note: For X2PLC and relay PLC terminals, only X0~X5 terminals support high-speed counters)

3.3.2 Control special registers

Single-phase single count up /down count control special SM components:

	One-way single count										
	C235	C236	C237	C238	C239	C240	C241	C242	C243	C244	C245
Increase and decrease count control	SM8235	SM8236	SM8237	SM8238	SM8239	SM8240	SM8241	SM8242	SM8243	SM8244	SM8245

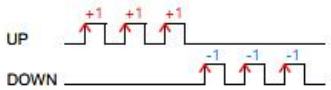
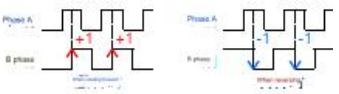
Single-phase dual count increase /AB phase count increase and decrease state special SM components:

	Single phase double counting					AB phase counting				
	C246	C247	C248	C249	C250	C251	C252	C253	C254	C255
Up and down counting status	SM8246	SM8247	SM8248	SM8249	SM8250	SM8251	SM8252	SM8253	SM8254	SM8255

Special SM components for frequency multiplication control:

	AB phase counting				
	C251	C252	C253	C254	C255
4x frequency enable	SM8195	SM8196	SM8197	SM819 8	SM819 9

3.3.3 Counting mode description

Counting Mode		Input signal form	Counting direction
Single phase single count			The count-up or count-down is specified by turning on/off SM8235 to SM8245. ON: count-down OFF: count-up
Single phase double counting			Check the count-up or count-down status by turning on/off SM82 46 to SM8250. ON: count-down status OFF: count-up status
Dual phase dual counting	1x		Check the count-up or count-down status by turning on/off SM8251 to SM8255. ON: count-down status OFF: count-up status By setting SM8195 to SM8199 , C251 to C255 can be switched to 4x counting mode. Note: 1x and 4x counting must be set before the high-speed counter is activated, otherwise the change of multiplication will be invalid.
	4x		

3.3.4 Use of high-speed instructions

Please refer to section 2.7.

Note: After power failure, the reset state of the high-speed counter will be saved. If there is no RST instruction in the ladder diagram to reset the high-speed counter, the high-speed counter cannot count normally. In addition, when using the RST instruction to reset the high-speed counter, a normally closed switch such as SM8000 cannot be used before the RST instruction, otherwise, the counter will be in the reset state and cannot count normally.

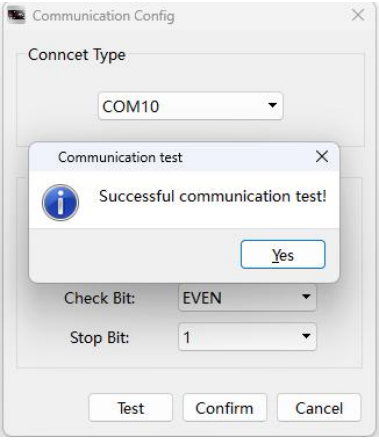
3.4 Serial communication part

X series PLC contains three available serial ports, 232 port, 485 port, and one BD board port. When the BD board is a 485BD board, its function is the same as the other two serial ports, and both use ladder diagrams to implement RS2 or MODBUS communication of 232 port and 485 port. When COM0, COM1, and COM2 are connected to the 485BD board, monitoring download, RS2 communication, and MODBUS communication can all be performed. COM0, COM1, and COM2 are the order of the three serial port channels of the PLC board. The upper computer displays the COM number detected by the actual computer, which cannot be confused.

3.4.1 Monitoring downloads

The RS485 interface and RS232 interface support downloading monitoring protocols.

Connect the corresponding port of PLC to PC through USB-485 communication converter or USB-232 communication converter, open UniSYS software on PC, and then download ladder diagram and monitor PLC through UniSYS software. It is worth noting that before communication starts, it is necessary to test the communication of PLC on UniSYS software. Click Tools->Communication Configuration in the menu bar, select the corresponding port, set the baud rate, and click Test. If the pop-up box shows that the communication test is successful, it means that UniSYS software can communicate normally with PLC. If other pop-up boxes are displayed, it means that there is a communication failure and UniSYS software cannot communicate normally with PLC. At this time, you need to find the cause of the failure (check whether the wiring is correct and whether PLC is normal). The figure below shows the pop-up box in normal communication status.



Download monitoring protocol is a master-slave protocol, in which UniSYS software is the master and PLC is the slave. The slave generally does not actively send communication data. When the master has a communication request, the slave will send communication data to the master as a response.

Download monitoring protocol is a general term for a class of protocol commands. This protocol includes most of the operations of UniSYS software on PLC, including downloading, uploading, verification, monitoring, forcing and other functions.

Downloading the communication protocol can set multiple baud rates, the most common ones are 9600baud, 19.2kbaud and 115.2kbaud.

The USB-485 or USB-232 communication converter is a plug-and-play device that supports USB V1.1 and USB2.0 PCs. This device provides data conversion from the PC USB port to RS485 or RS232 type data so that the corresponding port on the PLC side can receive normal communication data. If necessary, the use of this device requires the installation of the corresponding driver on the PLC side. The driver installation software can be used for automatic detection and one-click installation of the corresponding driver.

3.4.2 RS2 Communications

Free port communication is performed through the COM0/COM1/COM2 (connected to 485BD board) ports on the X3-PLC body to execute data sending and receiving instructions.

1. RS2 instruction format

FNC87	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
16-bit	11	RS2 \	RS2 S m D n n1

Operand Description

Operands	Content	Type
S	The starting device of the data register that stores the transmitted data	BIN16 bits
m	Number of bytes of sent data [setting range: 1 to 512]	BIN16 bits
D	When data reception is completed, the starting soft element of the data register that stores the received data	BIN16 bits
n	Number of bytes of received data [setting range: 1 to 512]	BIN16 bits
n1	Use channel number [setting Content: K0: COM0 ; K1: COM1; K2: COM2]	BIN16 bits

Available operand component types

Operands	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
S														*	*			*					
m														*	*				*	*			
D														*	*			*					
n														*	*				*	*			
n1																			*	*			

2. Function and action description

of the RS-232C or RS-485 serial communication port on the basic unit to perform data transmission and reception. To use this instruction, you must first use the ladder diagram to configure the communication-related registers .

(1) Communication format register configuration table

Communication format configuration	Bit15~13	Bit12~11	Bit10~7	Bit6~4	Bit3	Bit2~1	Bit0
protocol	Protocol Type	Hardware Type	RS2 frame format	Baud rate	Stop bits	Check digit	Data bits
RS2	010: RS2	00:232 mouth 01:485 mouth	0001~1100	001:4800 010:9600 011:19200 100:38400	0:1 stop bit 1:2 stop bits	00: No verification 01: Odd parity 10: Even parity	0: 7-bit data bit (does not support 7-bit mode without parity check) 1: 8bit data bit

By configuring the RS2 frame format, select the message frame to be used in communication:

0001	data
0010	Data+CR+LF
0011	Data + trailer
0100	Data+end+CR+LF
0101	Data + trailer + checksum
0110	Data+end+checksum+CR+LF
0111	Header + Data
1000	Header+Data+CR+LF
1001	Header + data + trailer
1010	Header+Data+Trailer+CR+LF
1011	Header + data + trailer + checksum
1100	Header+data+tailer+checksum+CR+LF

(2) COM0 (232 port) related software components when using RS2 communication

Bit device	Name
SM8062	Serial communication error
SM8120	Send wait flag
SM8121	Send Request
SM8122	Receive end flag
SM8129	Timeout flag

Word software	Name
SD8062	Serial Communication Error Codes
SD8120	Setting the communication format
SD8122	Remaining points for sending data
SD8123	Monitoring of receiving points
SD8129	Set the response timeout (unit: 10ms). When set to 0, the default value is 100ms.
SD8124	Header 1, 2
SD8125	Header 3, 4
SD8126	End of report 1, 2
SD8127	End of report 3, 4
SD8405	Receiving and verifying (receiving data)
SD8418	Receive and verify (calculation result)
SD8419	Send and check

(3) Related software components when COM1 (485 port) uses RS2 communication

Bit device	Name
SM8063	Serial communication error
SM8401	Send wait flag
SM8402	Send Request
SM8403	Receive end flag

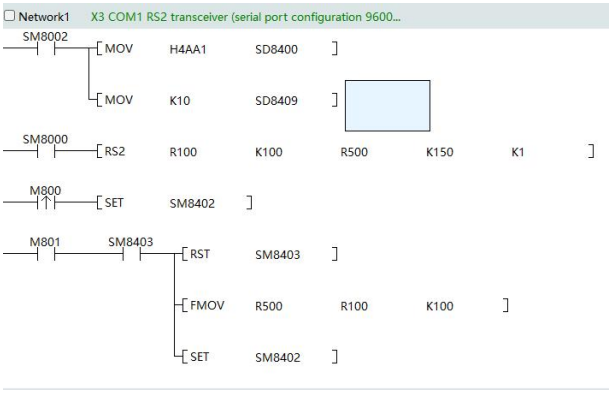
Bit device	Name
SM8409	Timeout flag
Word software	name
SD8063	Serial Communication Error Codes
SD8400	Setting the communication format
SD8402	Remaining points for sending data
SD8403	Monitoring of receiving points
SD8409	Set the response timeout (unit: 10ms). When set to 0, the default value is 100ms.
SD8410	Header 1, 2
SD8411	Header 3, 4
SD8412	End of report 1, 2
SD8413	End of report 3, 4
SD8414	Receiving and verifying (receiving data)
SD8415	Receive and verify (calculation result)
SD8416	Send and check

(4) COM2 (connected to 485BD board) related software components when using RS2 communication

Bit device	name
SM8438	Serial communication error
SM8421	Send wait flag
SM8422	Send Request
SM8423	Receive end flag
SM8429	Timeout flag

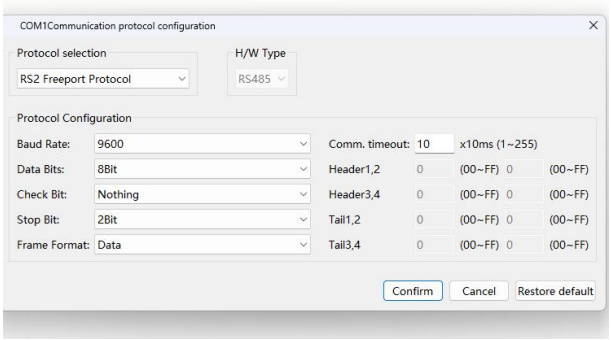
Word software	name
SD8438	Serial Communication Error Codes
SD8420	Setting the communication format
SD8422	Remaining points for sending data
SD8423	Monitoring of receiving points
SD8429	Set the response timeout (unit: 10ms). When set to 0, the default value is 100ms.
SD8430	Header 1, 2
SD8431	Header 3, 4
SD8432	End of report 1, 2
SD8433	End of report 3, 4
SD8434	Receiving and verifying (receiving data)
SD8435	Receive and verify (calculation result)
SD8436	Send and check

3. Ladder diagram to complete RS2 communication method

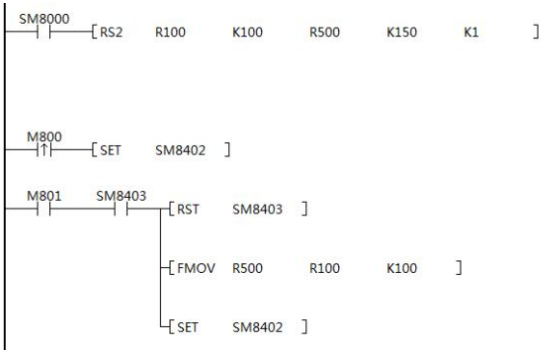


Write a value to the special SD8 register to initialize the RS2 communication protocol configuration (serial port configuration, RS2 frame format, etc.). When the serial port configuration of the device communicating with it is consistent with this, set SM8402 (send request) in the ladder diagram, and the PLC will send 100 bytes of data from R100 to R149 . After sending, SM8402 will be automatically reset . When SM8403 (receive end flag) is set, the monitoring can find that the 150 bytes after R500 store the data sent by the communication device. At this time, set M801 again, and the PLC will send the first 100 bytes of the received data to the communication device.

4. Configuration table to complete RS2 communication method



You can use the host computer to complete the protocol configuration initialization of the COM port (serial port configuration, RS2 frame format, etc.), but the actual communication data still needs to use RS2 instructions, as follows:



The ladder diagram no longer needs to initialize the SD8 communication protocol configuration register of RS2. This method is for the convenience of quickly configuring the RS2 communication protocol parameters.

5. Notes

(1) Header: up to 4 headers, placed in the corresponding special registers, data before 00H is valid, arranged in the order of low, high, low, high. For example, if the header of COM1 is set to SD8410 = 1234H, SD8411 = 5600H, the actual header is H34 (SD8410 low) H12 (SD8410 high).

(2) Trailer: There are up to four trailers, and their usage is similar to the header.

(3) Sum check: When selecting sum check, the message tail must be set. When sending data, the sum of data + message tail is calculated and attached to the sent data; when receiving data, check whether the received sum is the same as the one calculated in the PLC.

(4) CR+LF: If CR+LF is selected in the communication format setting, the character code CR+LF will be added after the data to be sent. If CR+LF is received continuously when receiving data, it means that the reception is completed. In addition, CR should not be included in the message. If this data appears, the reception will end abnormally.

(5) If the header, trailer and checksum are configured, the number of bytes n of the received data should be greater than or equal to the actual received data length plus 8.

(6) In general, it is best not to set the send request flag immediately after receiving the data. Allow some time for the serial port status to change.

(7) The configuration method of the protocol for initializing RS2 by the host computer has a higher priority than the ladder diagram method

3.4.3 Serial port MODBUS RTU communication

1. Parameter description required for communication configuration in ladder diagram

(1) Communication format configuration table

Communication format configuration	Bit15~13	Bit12~11	Bit10~7	Bit6~4	Bit3	Bit2~1	Bit0
protocol	Protocol Type	Hardware Type	Master and slave stations	Baud rate	Stop bits	Check digit	Data bits
MODBUS	011: MODBUS	00:232 mouth 01:485 mouth	0000: Slave 0001: Master	001:4800 010:9600 011:19200 100:38400 101:57600 110:115200	0:1 stop bit 1:2 stop bits	00: No verification 01: Odd parity 10: Even parity	0:7bit data bit 1: 8bit data bit

RTU mode data transmission method:

Coding system: 8-bit binary, hexadecimal 0-9, AF

Bits per byte: 1 start bit; 8 data bits; 1 stop bit with parity; 2 stop bits without parity

Error checking area: Cyclic redundancy check (CRC)

(2) Communication protocol configuration table

Bit	Name	Content	
		0 (bit = OFF)	1 (bit = ON)
b0	Select protocol	Other communication protocols	MODBUS Protocol
b1~b3	Unavailable		
b4	Master-slave station settings	MODBUS Master	MODBUS Slave
b5~b7	Unavailable		
b8	RTU/ASCII mode	RTU	ASCII (not supported)
b9~b15	Unavailable		

(3) Configurable communication formats for SD8120/SD8400/SD8420

Main Station			Slave			Communications
Data bits	Check digit	Stop bits	Data bits	Check digit	Stop bits	
8bit	none	1bit	8bit	none	1bit	yes
8bit	none	2bit	8bit	none	2bit	yes
8bit	strange	1bit	8bit	strange	1bit	yes
8bit	strange	2bit	8bit	strange	2bit	yes
8bit	even	1bit	8bit	even	1bit	yes
8bit	even	2bit	8bit	even	2bit	yes

2. Description of relevant special registers and error codes

(1) MODBUS related special registers

Among them: COM0 is port 232, COM1 is port 485, COM2 is port of 485BD board, the next communication refers to the execution of the next ADPRW instruction or polling the next line of MODBUS configuration table. Under normal circumstances, all related registers will be cleared when PLC STOPs.

Special Data Registers			Valid Station	Details
COM0	COM1	COM2		
SM8029	SM8029	SM8029	Main Station	Command end flag (used for ADPRW command in MODBUS)
SM8121	SM8401	SM8421	Master /Slave	MODBUS communication
SM8122	SM8402	SM8422	Master /Slave	An error occurred in MODBUS communication , which will be cleared during the next communication
SM 8123	SM8403	SM8423	Master /Slave	MODBUS communication error latch
SM8 062	SM8063	SM8438	Master /Slave	Serial communication error latch
SM8128	SM8408	SM8428	Main Station	The master station has a retransmission event judgment flag , which is cleared at the next communication.
SM8129	SM8409	SM8429	Main Station	timeout flag is set when the number of retransmissions is reached and cleared at the next communication.
SD8120	SD8400	SD8420	Master /Slave	Communication format setting
SD8121	SD8401	SD8421	Master /Slave	protocol
SD8122	SD8402	SD8422	Master /Slave	Communication error register, showing the latest error
SD8123	SD8403	SD8423	Master /Slave	Communication error latch register, latches the first error
SD8062	SD8063	SD8438	Master /Slave	Serial port error latch register, when MODBUS communication error occurs, 6321 is written into it
SD8128	SD8408	SD8428	Main Station	Display the current retransmission times of the master station
SD8129	SD8409	SD8429	Main Station	Set the response timeout (unit: ms) . If you enter 0, the default value is 3s.
SD8093	SD8411	SD8431	Main Station	Inter-request delay (inter-frame delay , unit: ms), baud rate greater than 19200, delay time defaults to 4ms, others are 10ms, cannot be changed in the slave station , when the number of slave stations connected to the master station is relatively large, it is recommended to set it to >=10ms
SD8094	SD8412	SD8432	Main Station	Set the number of retransmissions (0 ~20), if more than 20, set it to 20
SD8095	SD8414	SD8434	Slave	Slave station number (range 1~247)

(2) MODBUS error code description

Error Code	Error Content	Valid Station	Related component actions	Solution
202	MODBUS parameter setting error	Master /Slave	(1) When using COM0: SM8122, SM8123, and SM8062 will be turned on, the current error code will be stored in SD8122, SD8123 will latch the first error code, and SD8062 will store it in 6321. (2) When using COM1: SM8402, SM8403, and SM8063 will be turned on, the current error code will be stored in SD8402, SD8403 will latch the first error code, and SD8063 will store it in 6321. (3) When using COM2: SM8422, SM8423, and SM8438 will be turned on, the current error code will be stored in SD8422, SD8423 will latch the first error code, and SD8438 will store it in 6321.	Check whether the MODBUS related communication configuration ladder diagram or configuration parameter table is correct.
205	CRC/LCR checksum error	Master /Slave		Check whether the data protocol frame is correct
207	The response text length is incorrect	Master /Slave		Determine if the length of the response text received is correct
210	Illegal soft element address reading and writing	Master/Slave		Confirm whether the master station has accessed a valid soft component range (whether the address is within the permitted range); Confirm whether the MODBUS address of the slave is assigned correctly or includes the read and write address range.
212	Abnormal text	Master/Slave		Determine whether the address + number of points read by the master station exceeds the specified range; Check whether the ADPRW instruction parameter settings are correct
214	No function code	Master /Slave		Confirm whether MODBUS supports this function code
215	bad Request	Slave		Read the X coil
217	Incorrect use of ADPRW instruction	Master/Slave		Confirm whether the MODBUS communication in the master station is configured successfully; Check whether the ADPRW instruction is used in the slave station, and delete it if it is used.

Error Code	Error Content	Valid Station	Related component actions	Solution
211	Slave response timeout	Main Station	In addition to the above actions, the corresponding serial port timeout flags (SM8129, SM8409, SM8429) will be set	Determine whether the slave station number and communication parameters are correct; Determine whether the communication line connection is stable; Determine whether the slave starts normally

3. List of MODBUS standard functions supported

Function Code	Sub-function code	Function Name	Text Accessibility Points	Is this feature allowed?
0x01		Coil readout	1~2000 points	√
0x02		Input Readout	1~2000 points	√
0x03		Holding register read	1~125 points	√
0x04		Input register read	1~125 points	√
0x05		1 coil write	0Fixed	√
0x06		1 Register write	0Fixed	√
0x0F		Batch coil writing	1~1968	√
0x10		Batch Register Write	1~123 points	√
0x17		Bulk register read/write	Readout: 1~125 points Write: 1~121 points	√

4. Special instruction ADPRW for communication between master and slave

The command used to communicate with the slave station corresponding to the MODBUS master station (read/write data). The parameters corresponding to the command can be

Refer to the table above for configuration. The instruction format is shown in the table below.

FNC276	The number of program steps occupied by the instruction	Instruction Classification	Instruction Format
16-bit	11	ADPRW \	ADPRW S S1 S2 S3 S4/D

Operand Description

Operands	Content	Type
S	Specify the station number and the communication serial port number. The slave station number is (1~247), S+1 is the serial port number (range 0~2). The serial port number must be filled in first in the first cycle of the program.	BIN16 bits
S1	Function Code	BIN16 bits
S2	MODBUS address (0H~FFFFH)	BIN16 bits
S3	Read and write MODBUS address points or sub-function data	BIN16 bits
S4/D	Read / write data buffer	Bit/BIN 16 bits

Available operand component types

Operands	Bit Components							Word component								Address Indexing			constant		Real Numbers	String	pointer
	X	Y	M	T	C	S	Dx.y	KnX	KnY	KnM	KnS	T	C	D	R	V	Z	Modification	K	H	E	""	P
S														*	*			*					
S1														*	*			*	*	*			
S2														*	*			*	*	*			
S3														*	*			*	*	*			
S4/D	*	*	*			*								*	*			*	*	*			

5. MODBUS soft component address allocation

(1) Bit soft element

MODBUS software		X series software
Input (read only)	Coil (read/write)	
0x0000～0x1DFF	0x0000～0x1DFF	M0～M7679
0x1E00～0x1FFF	0x1E00～0x1FFF	SM8000～SM8511
0x2000～0x2FFF	0x2000～0x2FFF	S0～S4095
0x3000～0x31FF	0x3000～0x31FF	TS0～TS511
0x3200～0x32FF	0x3200～0x32FF	CS0～CS255
0x3300～0x33FF	0x3300～0x33FF	Y0～Y377
0x3400～0x34FF	-	X0～X377

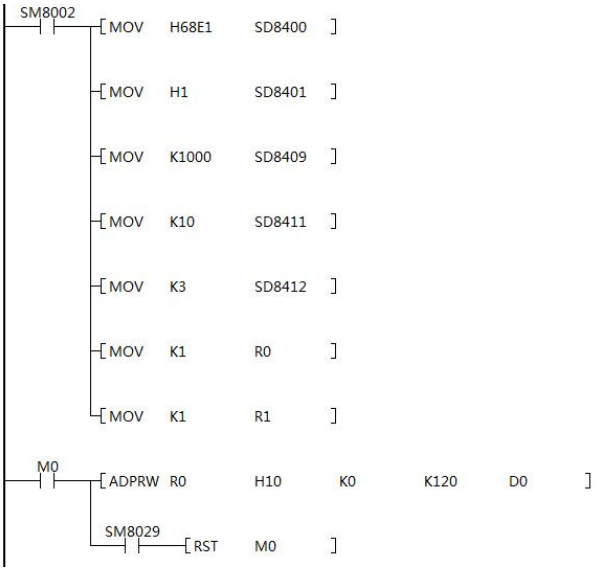
(2) Word software

MODBUS software		X series software
Input register (read only)	Holding registers (read/write)	
0x0000～0x1F3F	0x0000～0x1F3F	D0～D7999
0x1F40～0x213F	0x1F40～0x213F	SD8000～ SD8511
0x2140～0xA13F	0x2140～0xA13F	R0～R32767
0xA140～0xA33F	0xA140～0xA33F	T0～T511

MODBUS software		X series software
Input register (read only)	Holding registers (read/write)	
0xA340～0xA407	0xA340～0xA407	C0～C199
0xA408～0xA477	0xA408～0xA477	C200～C255
0xA478～0xA657	0xA478～0xA657	M0～M7679
0xA658～0xA677	0xA658～0xA677	SM8000～SM8511
0xA678～0xA777	0xA678～0xA777	S0～S4095
0xA778～0xA797	0xA778～0xA797	TS0～TS511
0xA798～0xA7A7	0xA798～0xA7A7	CS0～CS255
0xA7A8～0xA7B7	0xA7A8～0xA7B7	Y0～Y377
0xA7B8～0xA7C7	-	X0～X377

6. Ladder diagram to complete MODBUS communication method

(1) Example of COM1 master station configuration



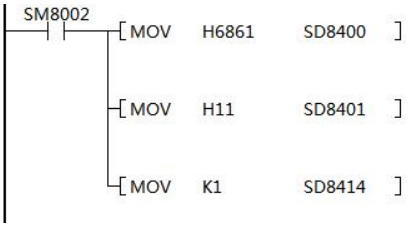
Among them: the master communication configuration of COM1 is baud rate 115200, 8 data bits, no check, 1 stop bit, timeout time 1s, inter-frame delay 10ms, and retry times 3 times. When M0 is set, the batch register write function is executed to write the 120 register data of D0~D199 to the slave.

Note:

① The actual meaning of the ADPRW parameter, S1, S1+1 parameters must be initialized in the first cycle;

② Generally, the master station can complete the communication configuration by configuring 5 registers according to the register corresponding to the serial port number;
③ The action cycle of SM8029, one ADPRW instruction ends and is set, and the next one is reset when the ADPRW instruction begins. This flag can be used to perform MODBUS polling operations.

(2) COM1 slave configuration example



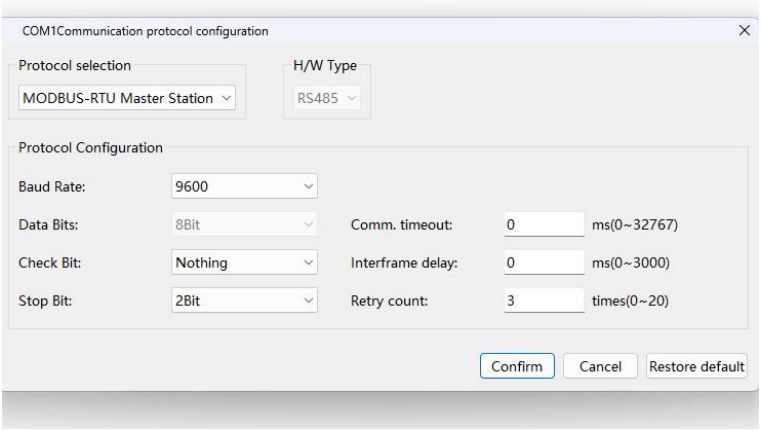
Among them: the slave communication configuration of COM1 is baud rate 115200, 8 data bits, no parity, 1 stop bit, and station number 1.

Note: Generally, the slave can complete the communication configuration by configuring 3 registers according to the registers corresponding to the serial port number.

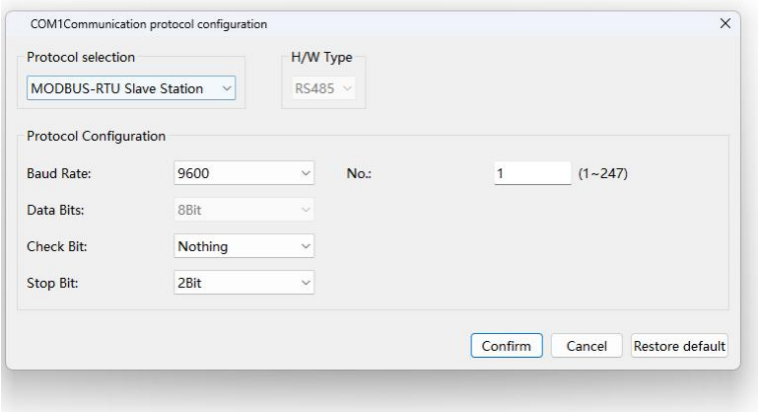
7. Configuration table completes MODBUS communication method

(1) Protocol configuration

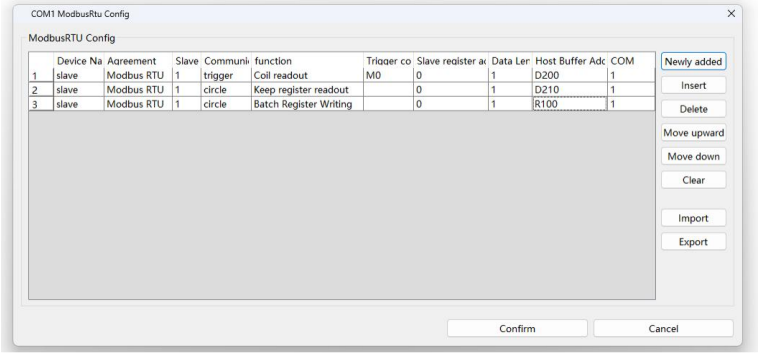
Master station configuration:



Slave configuration:



(2) Communication configuration



(3)Instructions

① When configuring MODBUS communication, you can directly use the configuration table method. Download the configuration table and the program to the PLC together to execute MODBUS communication. There is no need to initialize the communication parameters in the ladder diagram and write the ADPRW instruction. You only need to control the communication method in the configuration table to let the PLC complete the communication. The priority of the configuration table is higher than the ladder diagram configuration. This method is to quickly configure the MODBUS communication protocol parameters and communication data.

②MODBUS RTU configuration table is mainly used for master station communication. It is divided into trigger communication mode and cyclic communication mode. The trigger

mode requires the trigger bit to be set before sending, and it will automatically reset after sending. The cyclic mode is to send all the time.

③The suggestions for using the loop mode and trigger mode are as follows

• Cyclic communication: When you need to repeatedly and quickly read and write data that changes quickly in the slave station, such as reading the operating frequency, operating status, and input port status of the inverter; in process control application systems, real-time communication rewrites the operating frequency of the inverter, the status of the output port, etc., you can choose the "cyclic" communication mode. When the PLC executes the user program, it will repeatedly scan all the "cyclic" configuration items in the execution communication configuration table .

• Trigger communication: If you need to read or write data with a slower refresh rate from the slave at a regular interval, such as reading the inverter's output current, output power, current fault alarm information, etc., you can select the "trigger" communication mode. In the user program, each time the trigger condition is set , it will cause a communication operation of the corresponding communication item in the communication configuration table; by regularly setting the trigger condition in the user program , you can achieve the required frequency of communication read and write operations .

• Suggestions for communication mode settings: Reasonable selection based on the characteristics of the interactive parameter refresh required can greatly improve communication performance. Do not set all communication items as "loop" items for the sake of programming simplicity. This may reduce the timeliness of data interaction due to too many loop items, affecting the control effect of the system. Arrange some unimportant data access as "trigger" items, and trigger communication according to priority, which can greatly improve the timeliness of communication.

④ There are only 4 functions available in the configuration table method, coil read, batch coil write, holding register read, batch register write.

8. Special instructions

(1) The communication format configuration of X3 must refer to this document for configuration.

(2) MODBUS communication is only available when the PLC is in RUN state. In STOP state, the PLC returns to monitoring status.

(3) X3 currently supports RTU mode data transmission but does not support ASCII mode.

(4) The station number can be set in the range of 1-247. To ensure stable communication, a maximum of 32 slave stations are allowed to be connected. When the number of slave stations connected to the master station is relatively large, it is recommended to set it to >=10ms.

(5) Supports the use of multiple COM ports for MODBUS communication at the same time. MODBUS communication starts in the second cycle. Therefore, in the first cycle of PLC

operation, please initialize the communication configuration parameters and the serial port parameters used by the ADPRW instruction before the ADPRW instruction.

(6) It is not allowed to change the communication parameter configuration and the serial port parameters in the ADPRW instruction. If changed, it will only take effect the next time the PLC changes from stop to run.

(7) The main station does not support the playback function.

(8) The use of the MODBUS RTU configuration table has a higher priority than the register Contents configured in the ladder diagram and the ADPRW communication instruction.

3.5 Password function

It is divided into primary encryption and advanced encryption. Primary encryption mainly restricts the reading and writing of programs through passwords to prevent users from uploading, downloading or viewing programs at will. Advanced encryption mainly restricts the use of the program through secret keys. Only when the secret key is injected into the PLC can the matching extension program be used. After the extension program is encrypted, it always has the attributes of being unviewable and unchangeable, which can well protect one's intellectual property rights.

3.5.1 PLC primary encryption function

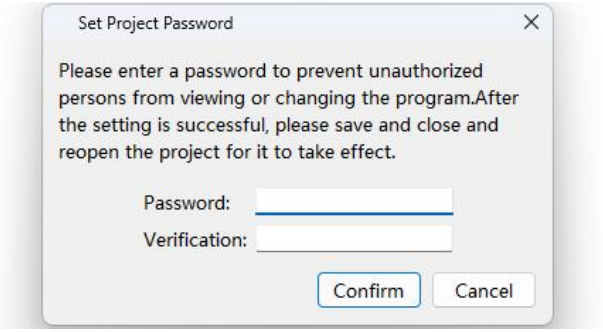
The host computer can set the read and write (modification) permissions of the PLC program through the PLC password setting . The primary encryption is divided into two parts: project encryption and program encryption. The program encryption is further divided into prohibiting upload, prohibiting download, shielding the main program, and shielding the subroutine .

1. Engineering encryption

Prevent unauthorized persons from viewing and changing the program. Only downloading is allowed. After downloading, the password is stored in the PLC, and the program will no longer be allowed to be uploaded. It can only be uploaded after the password is cleared in the PLC password setting. When the encrypted project is downloaded to the PLC, other programs can be downloaded, but after downloading, other programs will also be changed to the upload-prohibited state. Only by clearing the memory or clearing the password in the PLC password setting can they be uploaded normally. This function is mainly used to protect the customer's original program from being uploaded.

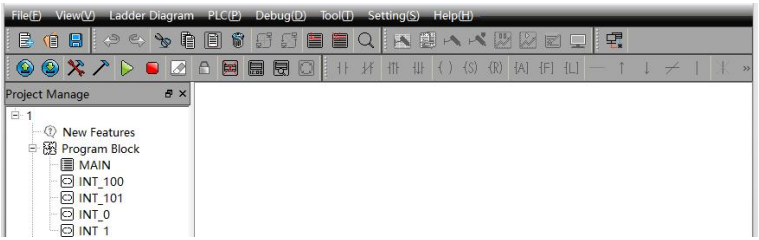
The operation method and effects are as follows:

① Click File->Set Project Password in the UNISYS host computer. After the password is set, open the project again and the project will be encrypted.



② When opening an encrypted project file, if you enter the correct project password, you can view and modify the program Content.

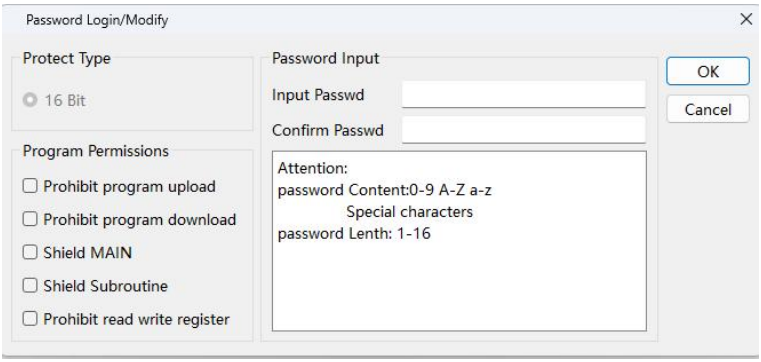
③ Skip the password verification process above, or do not go to File->Clear Project Password, then all programs in the program block part will be blocked and cannot be viewed or modified. The effect is as follows.



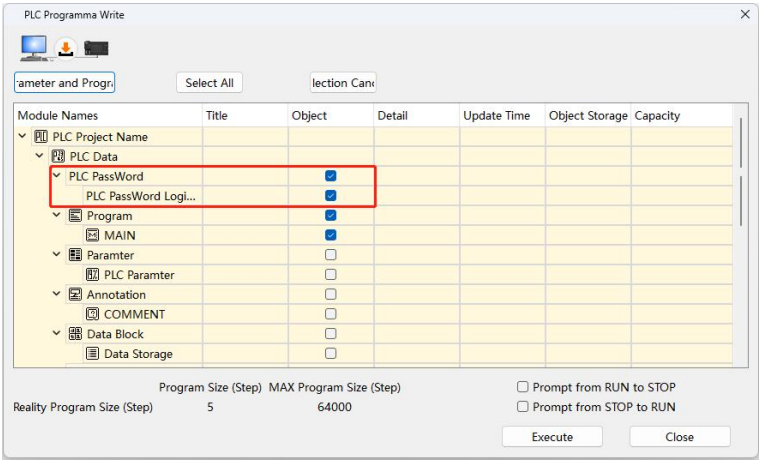
④ If the project is downloaded to the PLC without decrypting or clearing the password, the PLC will save the project password and change the program permission to prohibit uploading.

2. Program encryption

Encrypting the user program with corresponding permissions can prevent important parts of the ladder diagram program from being viewed or changed. It is divided into prohibiting upload, prohibiting download, shielding the main program, and shielding the subprogram.

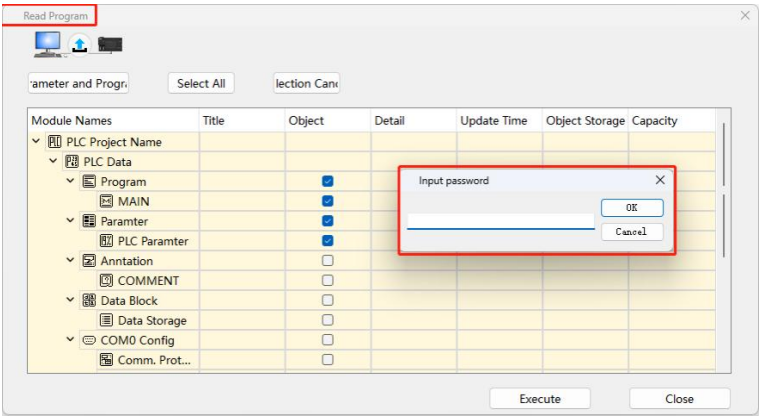


Check the corresponding permissions to limit the program's viewing or change permissions. When downloading, the password will be downloaded to the PLC along with the program, configuration, etc. If the PLC has a password, you need to enter the original password first, and then the PLC will change to the currently downloaded password.

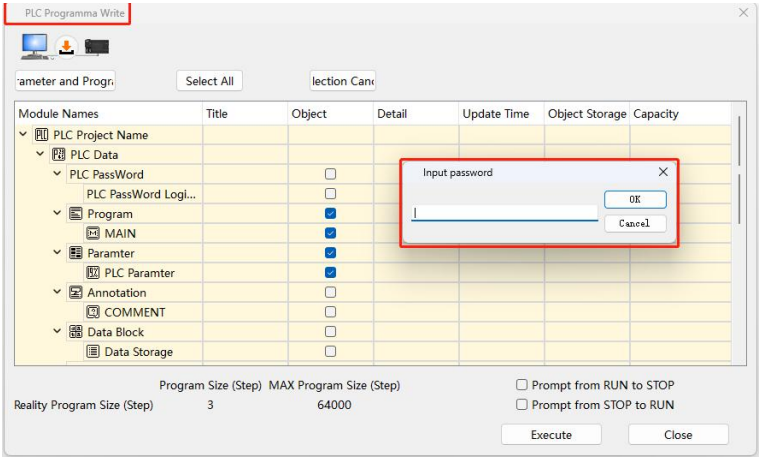


The effect is as follows (the premise is that the PLC has been encrypted with corresponding program permissions):

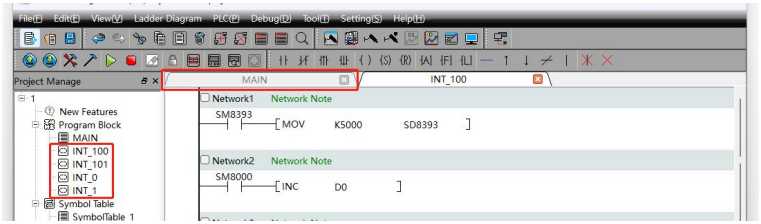
(1)Upload prohibited: When uploading



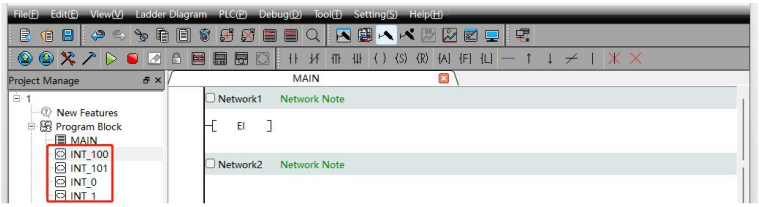
(2)Prohibit download: You need to enter a password to download the program



(3)Shield the main program: the main program part is invisible and cannot be changed, but the subprogram part can be changed

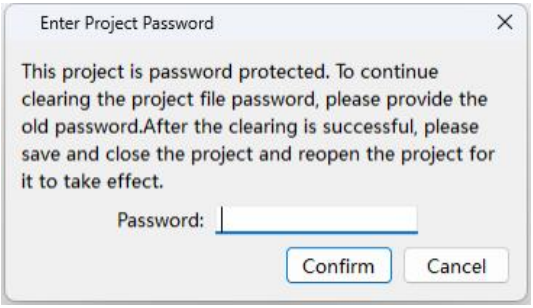


(4)Shielding subroutine:



3. Password Clear

(1) Clear project password: Click File->Clear project password in UNISYS. You need to enter the correct password when clearing the password . After clearing, all programs in the project become visible and can be changed.



(2) Clear the PLC password: Click PLC->PLC Password Setting->Clear in UNISYS. You need to enter the correct password when clearing the password. After the password is cleared, the program permissions stored in the PLC will disappear. In addition, the password stored in the PLC will be cleared when the PLC memory is cleared.

4. Notes

(1) The upload prohibition permission cannot coexist with the main program shielding permission or the subprogram shielding permission. Other permissions can coexist.

(2) The password function is mainly used to limit the read and write permissions of the program. When you forget the password, you can use the memory clear function to clear the password, but the program stored in the PLC will also be cleared.

3.5.2 PLC advanced encryption function

Application Scenario 1

A and B cooperate to develop a ladder diagram, and each protects their own intellectual property rights. A gives the encrypted extension program project to B for use (provided that A downloads the secret key to the PLC), but B cannot view and change this part of the ladder diagram. At the same time, B can edit his own ladder diagram based on A's project and can also perform corresponding encryption operations (download the secret key of B's extension program to the PLC and encrypt B's extension program), so that A and B can protect their intellectual property rights.

Application Scenario 2

A downloads the secret key to the PLC in advance and gives the PLC to B. Then B can use the encrypted extension program provided by A (B has the secret key corresponding to this extension program). However, B cannot give this project to C's PLC because C's PLC does not have this secret key.

The applications of the above scenarios can all be protected by generating extension blocks and 32-bit keys to protect the ladder diagram. The host computer can add an extension block by right-clicking the program block. To make the extension effective, the corresponding 32-bit key must be injected into the PLC through the key injection operation under the host computer->PLC. Then the extension will take effect after it is downloaded to the PLC. The extension can be effectively protected by this 32-bit key. The extension block can be written in three blocks, and the additional functions include extension encryption, password clearing, key generation and export, key import, and resource usage table.

1. Extension Encryption

If you use the advanced encryption function and allow the host computer to randomly generate a key, you need to perform the key generation and export operation before encryption. If you use your own key composed of characters, perform the key import operation before encryption. It is mainly used to protect the extension block so that it cannot be changed or viewed.

2. Password Clear

Mainly used by developers to view and modify extension blocks. After the password is cleared, the extension becomes visible and modifiable.

3. Key generation and export

It is mainly used to generate a 32-bit key that matches the expansion program block, and then inject this key into the PLC. The expansion program block can be recognized and used only after it is downloaded to the PLC.

4. Key import

Mainly used for developers to write their own keys.

5. Resource usage table

It can be edited and saved, and is mainly used by developers to provide resource usage information to facilitate subsequent developers to continue programming in projects with extensions. Of course, it can also be not provided to prevent other customers from knowing which resources the first-time developer used.

6. Specific usage

After the export (or import) key is generated in the extension, the extension can only be used by the PLC with this key. By downloading this key to the PLC, the PLC obtains the permission to use the extension.

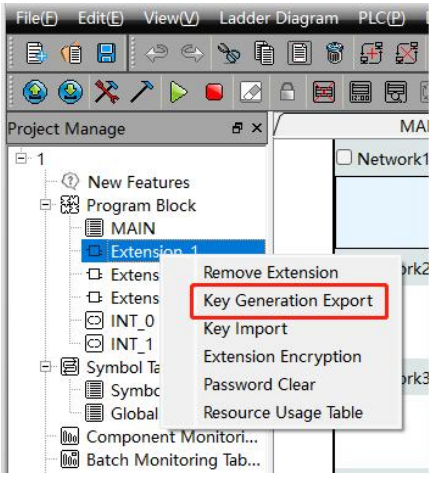
First-time developers write extension blocks according to the following process and inject the 32-bit key into the PLC, so that the PLC can make the corresponding extension blocks effective.

(1) Writing an extension program

Open the program block in UNISYS, right-click the program block to insert an extended program, insert as many blocks as you need, up to three blocks. Right-click the extended program block to insert a subroutine or interrupt subroutine. The total capacity of all programs is related to the system parameter settings, and the programming rules are the same as the MAIN program.

(2) Secret key generation and export

Use the host computer to match a random 32-bit key to the extension program and export it to a text file *.txt to facilitate developers to view the specific key information.

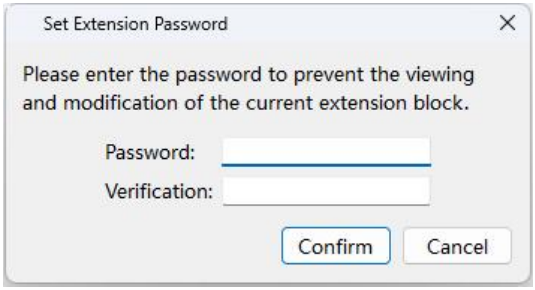


(3) Importing secret keys

If you use your own 32-bit key, skip step (2). The developer creates a .txt file and writes the 32-bit key information (valid characters 0-9, a-z, A-Z). You can then use your own key in the extension.

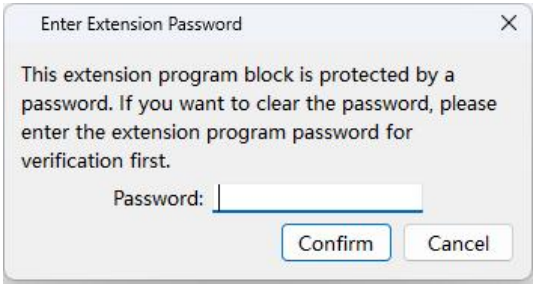
(4) Encryption extension block

The extension program block can be downloaded to the PLC only after encryption, otherwise the host computer will not download the extension program. After encryption, the shielding function of the extension program will take effect and will not allow viewing and modification.



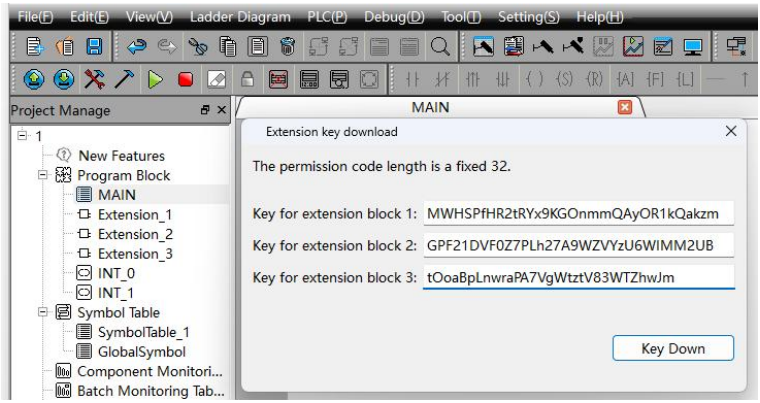
(5) Password clearing

This allows developers to view and modify extensions.



(6) PLC key injection

Before using the extension block, first inject the corresponding key into the PLC through the extension key download operation under the host computer->PLC. This key is the key that the developer has downloaded to the PLC in advance and matches the extension.



(8) Check the program download checkbox to download the corresponding project of the extended program block to the PLC.

7. Notes

(1) The extended program block is subordinate to the program block. Writing a ladder diagram is writing the entire program block. Here, only the Content to be encrypted is placed in the extended program block. However, the programming principle is the same as programming the MAIN program separately. It is just equivalent to dividing the previous ladder diagram program into multiple blocks. The MAIN program is the front part of the ladder diagram, and the extended program is the back part of the ladder diagram. Of course, you can choose to write only the extended program, which is equivalent to encrypting the entire ladder diagram data.

(2) After the extension program is written, you must first generate and export the key or import the key before encrypting it in order to obtain the usage key of the extension program. This key is used to download it to the PLC to make the PLC effective for the extension program block.

(3) When the PLC has an extended program block, the new key injection corresponding to the extended program block number will not take effect, that is, it is not allowed to change the key when there is an extended program block. The new key can only be downloaded when the memory is cleared or the program is cleared.

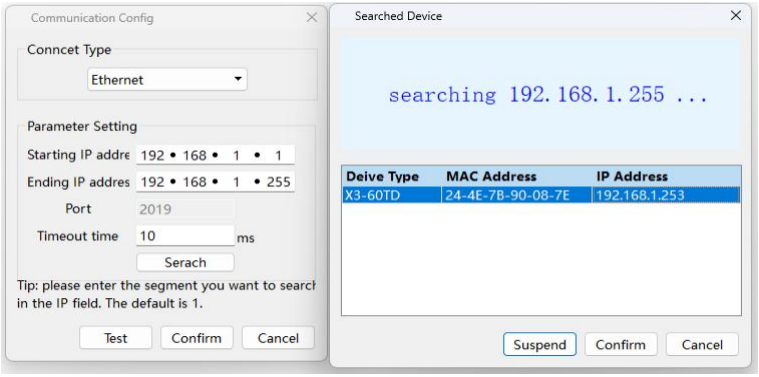
(4) When the secret key in the extension program does not match the secret key in the PLC, the extension program will fail to download, and only the MAIN program and the successfully downloaded extension program will be effective in the PLC.

(5) When deleting an extension block, you need to enter a password for verification. Only when it is correct can it be deleted. At the same time, the program, key, and original password inside will be deleted.

3.6 Ethernet Function

3.6.1 Monitoring download in Ethernet mode

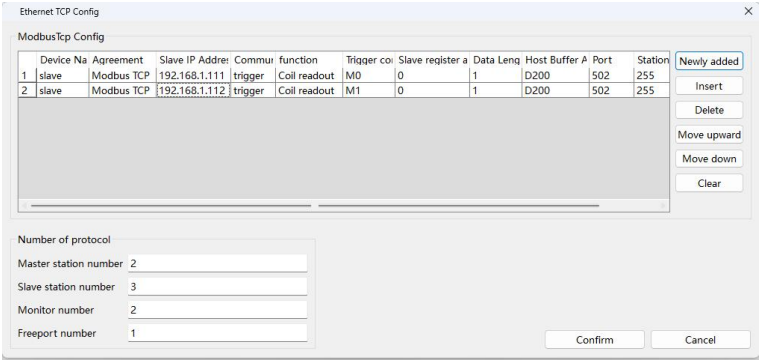
Select the communication configuration of the host computer, change the connection type to Ethernet, and display a default IP address. Click Search to search for the PLC IP address and its corresponding MAC address connected to Ethernet in the same network segment. Change the network segment to search for the PLC IP address of other network segments. Then select the PLC IP to be connected, click Test, and you can monitor and download after the test is successful. It supports monitoring and downloading of multiple computers with UNISYS software installed or monitoring and downloading of multiple UNISYS on one computer, but the number is limited. Please refer to the following connection protocol allocation introduction for details.



3.6.2 Download the PLC IP address

The host computer can reconfigure the PLC's IP address parameters. The PLC's default IP is 192.168.1.254.

3.6.3 Ethernet communication protocol configuration



(1) Number of protocols

protocols is allocated according to the configuration table . The number of master stations is related to the IP. The number of slave stations, the number of monitoring stations and the number of free ports can be allocated by yourself. The total number of connections cannot exceed eight, that is, a maximum of 8 different IPs are allowed.

①master stations: Assigned according to different IPs in the configuration parameters. As many master stations as there are IPs, because PLC meets eight function codes, it is allowed that the same IP configuration parameter can have eight rows of configuration tables, and configuration parameters can be selected arbitrarily when entering the table, but the total number of configuration table rows for the same IP cannot exceed 8 rows.

②of slave stations: Indicates how many master stations (with different IP addresses) are allowed to connect to the current PLC.

③monitoring: Indicates how many host computer software are allowed to monitor and download the PLC.

④Number of free ports: indicates how many free ports the PLC is allowed to connect to communicate. For details, see Section 2.8.

(2) MODBUS TCP communication configuration

①Serial number: corresponds to the row number of the configuration table

②Protocol: MODBUS TCP

③Device Name: Used to note the slave device name connected to the PLC

④Slave IP address: the IP address of the server to which the PLC is connected

⑤Communication mode: Trigger means that the master station must set the trigger condition to send data to access the slave station; Cycle means that the master station keeps sending data to access the slave station

⑥Functions: including coil read, input read, holding register read, input register read, single coil write, single (hold) register write, batch coil write, batch (hold) register write

⑦Trigger condition: When the communication mode is triggered, the corresponding condition flag needs to be set to communicate. The trigger condition will be reset after the communication timeout or completion.

⑧Slave station register address: The soft element address corresponding to the slave station to be accessed by the master station. The address allocation is consistent with the serial port MODBUS soft element address allocation

⑨Data length: The data length is limited according to the selected function. Coil readout, input readout: 1~2000; holding register readout, input register readout: 1~125; single coil write, single (hold) register write: fixed 0; batch coil write: 1~1968; batch (hold) register write: 1~123

⑩Master buffer address: stores the data information sent back by the slave , which can only be D or R register

⑪Port number: For MODBUS TCP communication, fixed at 502

(3) Special instructions

①When the PLC loses power, the last MODBUS TCP parameters will be saved and communication will be performed according to the configuration next time;

②When the number of allocated monitoring units is 0 and the number of allocated protocols is downloaded to the lower computer, the monitoring download function will not be available in the Ethernet mode of the upper computer;

③The MODBUS TCP parameter configuration table can be configured with a maximum of 64 lines, only 8 different IPs are allowed, and each IP configuration table cannot exceed 8 lines. The number of master stations must correspond to the configuration table, otherwise such configuration is not allowed.

④The trigger conditions of different slave stations (with different IP addresses) must be different, and the host will impose restrictions.

⑤When the PLC memory clear function is used, the MODBUS TCP parameter configuration table stored in the PLC will be cleared.

3.6.4 Ethernet related special registers

Element	Function
SM8500 (read and write)	Slave station is not online 0: All slaves are online, and the SM8500 will be reset. 1: A slave is offline. This bit is set when any slave is offline.
SM8501 (Readable and writable)	PLC network cable is not connected or disconnected 0: The network cable connection is normal 1: The network cable is not connected or disconnected
SM8502 (Readable and writable)	Ethernet function control bit 0: Enable Ethernet function

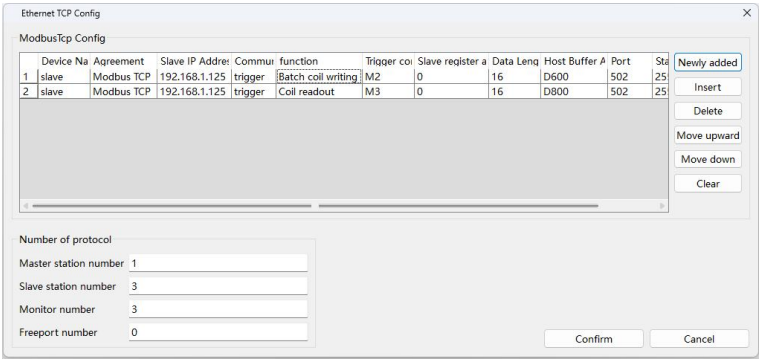
Element	Function
	1: Disable Ethernet function
SM8503 (Readable and writable)	MODBUS TCP function control bits 0: Enable MODBUS TCP function 1: Disable MODBUS TCP function
SM8080~SM8087 (readable and writable)	slave IP where the timeout occurs corresponds to the timeout of IP1~IP8 in the MODBUS TCP configuration table. After setting, it means the corresponding IP timeout. If multiple timeout judgments are required, manual reset can be performed.
SD8500 (read only)	Display the number of PLC currently assigned protocols 0: means there are 3 monitors, 4 slaves, and 0 masters Hexadecimal number: 11-8bit, 7-4bit, 3-0bit four-digit number represents the number of monitoring, slave station, and master station respectively (SD8500 = H340 means 3 monitors, 4 slaves, and 0 masters)
SD8501 (Readable and writable)	The IP corresponding to the offline slave (the fourth paragraph) is checked when SM8500 is set. At this time, the server (slave device) needs to be reopened to establish communication again. After the communication is successfully established, the register will be cleared. If the value is not 0, there may be other devices that are not connected normally or the device fails to restart. The default value is 0
SD8502 (Readable and writable)	Display MODBUS TCP error codes 0: No error -1: The SD8500 value is incorrectly set, and the default connection configuration will be used. Error when PLC is a slave station (error code composition: the upper eight bits are the station number, the lower eight bits are the error type): This situation indicates that the data frame sent by the master station to the PLC is wrong Error type: 0x01 indicates illegal function 0x02 indicates illegal data address, slave address setting error 0x03 indicates illegal data value, data length setting error or master buffer data setting error Error when PLC is the master station (error code composition: the upper eight bits are IP, the lower eight bits are error type): This situation indicates that the data frame returned by the slave station to the PLC is wrong Error type: 0x11 indicates illegal function 0x12 indicates illegal data address 0x13 indicates an illegal data value
SD8503 (Readable and writable)	Set the timeout period (in ms). The default value of 0 means the timeout period is 1000ms.

Element	Function
SD8504 (read only)	Display the current number of TCP connections (the total number of master + slave + monitoring)
SD8505 (not allowed to use)	Burn mac address dedicated register, when it is -1, it means that the mac address needs to be burned.
SD8506 (Readable and writable)	Display the latest slave station IP with timeout and the table position of the timeout function code corresponding to the IP (the upper eight bits are the timeout IP, and the lower eight bits are the function code position. For example: 0x4901 means the timeout IP is 73, and the timeout function code is the first read or write function of the IP)
SD8507 (Readable and writable)	The reconnection interval (the interval between each attempt of the master to reconnect to the slave after the slave is offline, in ms). When it is 0, the default is 100ms
SD8508 (read only)	The first segment of the local IP
SD8509 (read only)	of the local IP
SD8510 (read only)	of the local IP
SD8511 (read only)	of the local IP

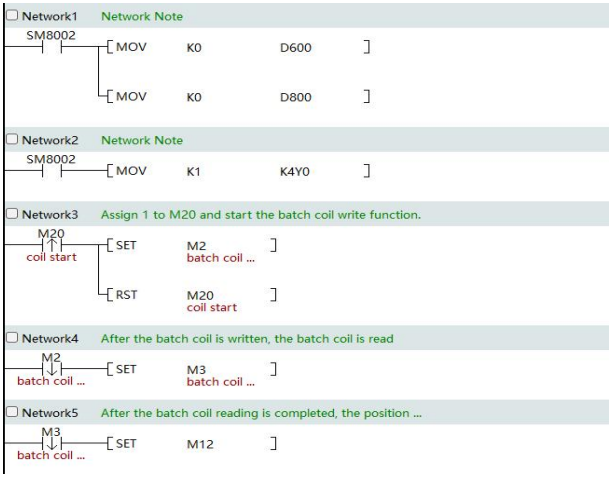
3.6.5 MODBUS TCP trigger mode protocol configuration example

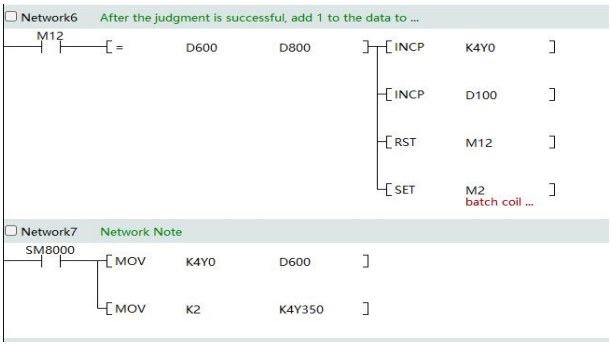
Use trigger polling to read and write coils to the slave station with IP 192.168.1.125.

(1) Download protocol configuration



(2) Write the trigger conditions for ladder diagram operations so that they can communicate according to the program flow





(3) Start M20, and you can see that the output LED on the master station PLC turns on the marquee mode.

(4) Communication process analysis

① PLC will execute the configuration functions in the configuration table line by line. After executing the entire table, it will execute again from the first line of configuration.

② The function of the configuration line will be executed when the trigger condition is enabled, and the execution order in ① is also followed.

③ After the trigger function is executed or a timeout occurs, the trigger bit will be reset. The reset of this trigger bit can be used to determine the completion of the current configuration line execution, and then the edge jump method can be used to execute the next function configuration line. In this way, the trigger polling of the entire table can be completed.

④If the loop function is selected in the configuration table, the PLC will always execute in sequence according to the configuration table and perform the communication function. The trigger method can choose when to communicate based on the user's process requirements.

3.6.6 MODBUS soft component address allocation

(1) Bit soft element

MODBUS software		X series software
Input (read only)	Coil (read/write)	
0x0000～0x1DFF	0x0000～0x1DFF	M0～M7679
0x1E00～0x1FFF	0x1E00～0x1FFF	SM8000～SM8511
0x2000～0x2FFF	0x2000～0x2FFF	S0～S4095
0x3000～0x31FF	0x3000～0x31FF	TS0～TS511
0x3200～0x32FF	0x3200～0x32FF	CS0～CS255
0x3300～0x33FF	0x3300～0x33FF	Y0～Y377
0x3400～0x34FF	-	X0～X377

(2) Word software

MODBUS software		X series software
Input register (read only)	Holding registers (read/write)	
0x0000～0x1F3F	0x0000～0x1F3F	D0～D7999
0x1F40～0x213F	0x1F40～0x213F	SD8000～ SD8511
0x2140～0xA13F	0x2140～0xA13F	R0～R32767
0xA140～0xA33F	0xA140～0xA33F	T 0 ～T511
0xA340～0xA407	0xA340～0xA407	C0～C199
0xA408～0xA477	0xA408～0xA477	C200～C255
0xA478～0xA657	0xA478～0xA657	M0～M7679
0xA658～0xA677	0xA658～0xA677	SM8000～SM8511
0xA678～0xA777	0xA678～0xA777	S0～S4095
0xA778～0xA797	0xA778～0xA797	TS0～TS511
0xA798～0xA7A7	0xA798～0xA7A7	CS0～CS255
0xA7A8～0xA7B7	0xA7A8～0xA7B7	Y0～Y377
0xA7B8～0xA7C7	-	X0～X377

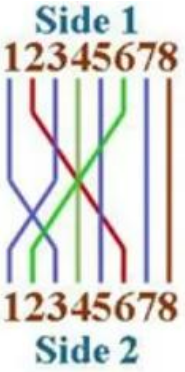
3.6.7 Ethernet free port communication

Please refer to Chapter 2.8 for instructions on how to use the Ethernet free port commands.

3.6.8 Precautions for using Ethernet function

(1) When the PLCs cannot communicate normally with each other through MODBUS TCP, first check the PLC IP problem to see if there is an IP conflict or incorrect IP configuration. Then check whether the network cable is normal and the network is stable. Then check whether the configuration table is incorrect.

(2) PLCs communicate directly via network cables, with switches or cross-over cables in between. The cross-over cable wiring method is as follows



(3) The following are the recommendations for using the loop mode and trigger mode:

• Cyclic communication: When you need to repeatedly and quickly read and write data that changes quickly in the slave station, such as reading the operating frequency, operating status, and input port status of the inverter; in process control application systems, real-time communication rewrites the operating frequency of the inverter, the status of the output port, etc., you can choose the "cyclic" communication mode. When the PLC executes the user program, it will repeatedly scan all the "cyclic" configuration items in the execution communication configuration table .

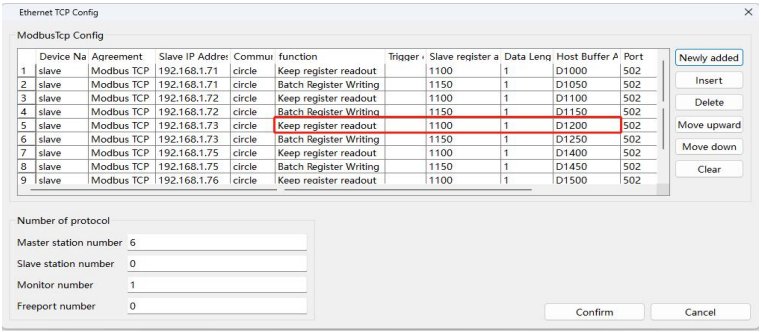
• Trigger communication: If you need to read or write data with a slower refresh rate from the slave at a regular interval, such as reading the inverter's output current, output power, current fault alarm information, etc., you can select the "trigger" communication mode. In the user program, each time the trigger condition is set , it will cause a communication operation of the corresponding communication item in the communication configuration table; by regularly setting the trigger condition in the user program , you can achieve the required frequency of communication read and write operations .

• Suggestions for communication mode settings: Reasonable selection based on the characteristics of the interactive parameter refresh required can greatly improve communication performance. Do not set all communication items as "loop" items for the sake of programming simplicity. This may reduce the timeliness of data interaction due to too many loop items, affecting the control effect of the system. Arrange some unimportant data access as "trigger" items, and trigger communication according to priority, which can greatly improve the timeliness of communication.

(4) Exception handling when communication timeout occurs

Communication errors will cause timeouts. If users use Ethernet communication, errors that occur will cause major accidents to the operation of the equipment. Examples of avoidance methods are as follows:

The data read out by the first function of IP73, the register read function, is particularly important data. Timeout is not allowed to cause data not to be refreshed and cause equipment operation failure. The processing method is as follows.



IP73 corresponds to the third IP, so the timeout flag is SM8082. When the timeout occurs, the high eight bits of SD8506 display the IP, and the low eight bits display the location of the timeout function data under the IP. The exception handling ladder diagram is as follows

3.7 Use of extension modules

3.7.1 Extension module ID

1. Module ID composition rules

Byte 1 (H)								Byte 2							
t	t	t	t	N	N	A	A	I	I	I	I	Q	Q	Q	Q
t: module type identifier 0000 - Normal module 0001 - Module (RTD) 0010 - Module (TC) 0011 - Module (PID) ... Others - Reserved N: Disable A: I/O type identification 00 - None 01 - Numbers 10 - Simulation 11 - (no mixed modules)								I: Input ID 0000 - No input 0001 - 1AI or 4DI 0010 - 2AI or 8DI 0011 - 4AI or 16DI 0100 - 8AI or 32DI 0101 - 16AI or 64DI Q: Output ID 0000 - No output 0001 - 1AO or 4DO 0010 - 2AO or 8DO 0011 - 4AO or 16DO 0100 - 8AO or 32DO 0101 - 16AO or 64DO							

2. Module classification and corresponding ID

Modules	ID (hexadecimal)	illustrate
8DI	0120	8-point digital input
16DI	0130	16-point digital input
32DI	0140	32-point digital input
8DO	0102	8-point digital output
16DO	0103	16-point digital output
32DO	0104	32-point digital output
2AI	0220	2-channel analog input
4AI	0230	4-channel analog input
8AI	0240	8-channel analog input
2AO	0202	2-channel analog output
4AO	0203	4-channel analog output
8AO	0204	8-channel analog output
8DI/8DO	0122	Already available, 8-point input/8-point output digital quantity
16DI/16DO	0133	16-point input/16-point output digital
4AI/2AO	0232	Already available, 4-channel input/2-channel output analog
4RTD	1230	4-input RTD analog module
8TC	2240	TC module, 8-channel thermocouple input analog
8TC-PID	3240	Special modules, 8-channel thermocouple input analog, PID module
...	...	...

3. Special registers related to module ID

S D8260	S D8261	S D8262	S D8263
Module 1 ID	Module 2 ID	Module 3 ID	Module 4 ID
S D8264	S D8265	S D8266	
Module 5 ID	Module 6 ID	Module 7 ID	

ID Display:

Under normal circumstances, the module ID is displayed. Under abnormal circumstances, H0000 (0) is displayed if it is not connected, HFFFF (-1) is displayed if it is disconnected, and HFFFE (-2) is displayed if the configuration is wrong .

The bus always tries to obtain the maximum number of connections that have ever existed, that is, when there are 7 modules after power-on, removing 7 will cause 7 to be considered disconnected.

3.7.2 Digital expansion module

Digital modules with less than one byte (such as 4DI) are counted as one byte. And the port number of the expansion module always starts from 0 in octal. The digital input of the XPLC mounting module is counted from the address behind the main body (X370~X377 is reserved for the BD board). When power is turned on, the number of digital bytes corresponding to each slot can be obtained according to the ID. Therefore, the X/Y number corresponding to each slot can be known.

3.7.3 Analog expansion module

The analog expansion module is stored in the storage area of the PLC, fixed in D7500~D7999. The analog data includes status register, data register, configuration register , and additional register (generally, digital quantity does not have configuration register and additional register). The configuration register is used to configure the range, gain, polarity and other information of each channel of the module. The additional register is used for some special purposes, such as configuration saving, calibration, etc. Each module occupies 50 D (more are reserved) registers. And the first one in each group is the status register. The specific addresses are as follows:

Status Register	R area (analog input data area)	W area (analog output data area)	Area C (Configuration Registers)
D7500	D7501~D7508 (①No R zone ②No W zone ③Both R and W)		D7509~D7510
D7550	D7551~D7558 (①No R zone ②No W zone ③Both R and W)		D7559~D7560
D7600	D7601~D7608 (①No R zone ②No W zone ③Both R and W)		D7609~D7610
D7650	D7651~D7658 (①No R zone ②No W zone ③Both R and W)		D7659~D7660
D7700	D7701~D7708 (①No R zone ②No W zone ③Both R and W)		D7709~D7710
D7750	D7751~D7758 (①No R zone ②No W zone ③Both R and W)		D7759~D7760
D7800	D7801~D7808 (①No R zone ②No W zone ③Both R and W)		D7809~D7810

Notice:

(1) When using a module, do not reuse the registers used in the corresponding module in the ladder diagram, otherwise problems will occur.

(2) The R area and W area of the data area should be distinguished according to the module type and the number of input and output points, including the three situations described in the table. ① For example, 8AO, there are 8 W areas and no R area; ② For example, 8AI, there are 8 R areas and no W area; ③ For example, 4AI2AO, there are 4 registers in the R area and 2 registers in the W area.

3.7.4 Module register reference table

Register comparison table:

Offset	Effect	illustrate
0	Status Register	Refer to the status register description
1	AI 0	A I Channel 0
2	AI 1	AI channel 1
...	...	...
n	AI n	A I channel n
n+1	AO 0	A O Channel 0
n+2	AO 1	A O Channel 1
...	...	...
8	8	A O Channel 8
9	Configuration	Choose to use 1 or 2 configuration registers according to the number of AI channels. 4 channels occupy 1 configuration register.
10		

Note: 1 status register, 8 data registers (1~n represents AI channel data , n+1~8 represents AI channel data) , 2 configuration registers

The status register is described as follows:

Bit	Bontent
Bit15-Bit8	/
Bit7	Internal wire breakage or bottom board breakage
Bit6-Bit4	/
Bit3	Uncalibrated
Bit2	Command Error
Bit1	Configuration Error
Bit0	Internal communication error

Note: This status word is invalid for 8DI8DO module

Configuration register details:

Bit	Effect	illustrate
15	A I (x+3) polarity control	0: bipolar; 1: unipolar
14-12	A I (x+3) gain control	000 , 00 1, 010 correspond to 1, 2, 4 times gain respectively
11	A I (x+2) polarity control	0: bipolar; 1: unipolar
10-8	A I (x+2) gain control	000 , 001, 010 correspond to 1, 2, 4 times gain respectively
7	A I (x+1) polarity control	0: bipolar; 1: unipolar
6-4	A I (x+1) gain control	000 , 00 1, 010 correspond to 1, 2, 4 times gain respectively
3	A I x polarity control	0: bipolar; 1: unipolar
2-0	A I x Gain Control	000 , 00 1, 010 correspond to 1, 2, 4 times gain respectively

Notice:

(1) The configuration register is allocated according to the number of AI channels. If it is a 4AI2AO module, there is only one configuration register, then x = 0, and the configuration register is the configuration of AI 3-AI 0 ; if it is an 8AI module, there are two configuration registers, x is 0 and 4 respectively, the first configuration register is the configuration of AI 3-AI 0 , and the second register is the configuration of AI 7- AI 4.

(2) writing the configuration register, you need to enable the configuration switch

(3) If it is an RTD module, the configuration register is used to configure the RTD module type. All channels share the type of channel 1. The configuration register is written with H0~H7 to represent PT100 , PT200 , PT500 , PT1000 , NI100 , NI120, NI1000, and CU10 respectively.

R area data register data description:

The register value is -32000~+32000 , and the corresponding voltage range is as follows

Gain multiple	Corresponding voltage
1	-10V~10V
2	-5V~5V
4	-2.5V~2.5V

W area data register data description:

Register value -32000~+32000, corresponding to -10V~10V

3.7.5 Module configuration enable switch

address	Module 1	Module 2	Module 3	Module 4	Module 5	Module 6	Module 7
Configuration Enable	SM8261	SM8264	SM8267	SM8270	SM8273	SM8276	SM8279

After the configuration data is written to the module configuration register, the corresponding configuration enable flag must be set to make the configuration effective. It will automatically reset after being set and cannot be set all the time.

3.7.6 Bus error analog module indicator light

Error code register S D8449, error flag S M8449

Error code format A00E

A: Address number, for example, if module 7 is disconnected, it is 700 2

E: Error code (2 : disconnection , 3: module cannot be recognized , 4 : configuration error , B: configuration does not match)

Analog module indicator light:

PW: Power

COM: Communication error within the module

ERR: Read/write abnormality (incorrect command or parameter configuration)

3.7.7 Disconnection recovery

Hot swapping is not supported yet. To replace a module, you must power off and restart.

3.7.8 Module Configuration Example

Example 1 (manually configure module configuration registers):

The following takes 4AI2AO configuration as an example to illustrate the module configuration method. Assume that the 4AI2AO module is the first module . This module has 4 analog input channels (corresponding to A I 3-A I 0) and 2 analog output channels. The corresponding configuration enable flag is S M826 1 .

Module related operation enable switch

address	Module 1	Module 2	Module 3	Module 4	Module 5	Module 6	Module 7
Configuration Enable	SM8261	SM8264	SM8267	SM8270	SM8273	SM8276	SM8279

(1) Status register D7 5 00 can observe the current module status

Bit	Content
Bit15-Bit8	/
Bit7	Internal disconnection
Bit3	Uncalibrated
Bit2	Command Error
Bit1	Configuration Error
Bit0	Internal communication error

(2) Data registers D7501 ~ D7504 can observe the current 4 analog input values, and D7505 ~ D7506 can observe the current 2 analog output values.

In the default configuration (bipolar, 1x gain), -32000~32000 corresponds to -10V~+10V

Gain multiple	Corresponding voltage
1	-10V~10V
2	-5V~5V
4	-2.5V~2.5V

(3) Configuration register D7509

D7509 corresponds to the polarity and gain control of AI3-AI0, 4 bits correspond to one channel

Bit	Effect	illustrate
15	A I 3 polarity control	0: bipolar; 1: unipolar
14-12	AI 3 gain control	0 00 , 00 1, 010 correspond to 1, 2, 4 times gain respectively
11	A I 2 polarity control	0: bipolar; 1: unipolar
10-8	AI 2 gain control	0 00 , 00 1, 010 correspond to 1, 2, 4 times gain respectively
7	A I 1 polarity control	0: bipolar; 1: unipolar

Bit	Effect	illustrate
6-4	A I 1 gain control	0 00 , 00 1, 010 correspond to 1, 2, 4 times gain respectively
3	A I 0 polarity control	0: bipolar; 1: unipolar
2-0	A I 0 gain control	0 00 , 00 1, 010 correspond to 1, 2, 4 times gain respectively

If you want to configure AI0 to bipolar 2x gain and leave other channels as default, write 0001 to bit 3-0 of D7509 and 0000 0000 0000 for other bits, that is, write H1 to D7509 , and then set the configuration enable flag, here SM8261 . You can see that the data D7501 = 16000 corresponding to the input voltage of 5V will become 32000, which is doubled.

Note: After enabling the configuration, the new configuration will be automatically saved. After writing the configuration register, the corresponding configuration enable flag must be enabled.

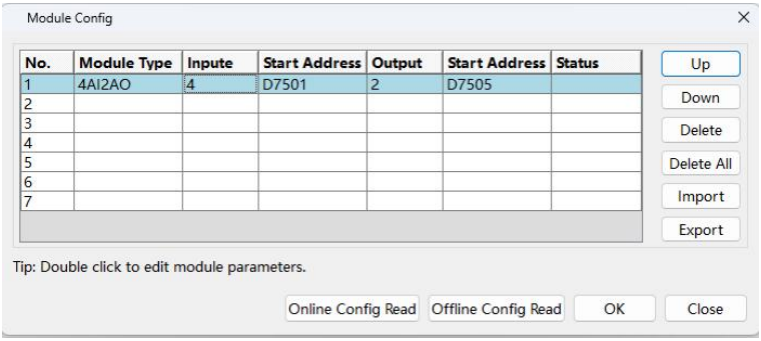
(4) Write output

A00 and A01 of the 4AI2AO module correspond to D7505 and D7506 . If it is a digital quantity module, the write output is realized through the Y relay.

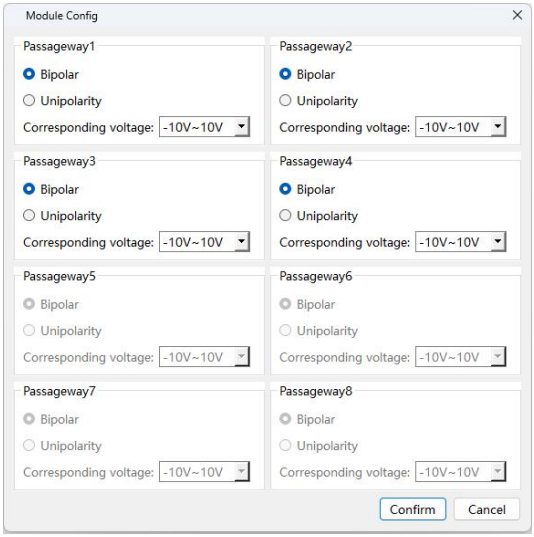
Example 2 (offline configuration of the host computer UNISYS module)

First download this configuration to the PLC, and then connect the module, or perform corresponding configuration according to the module already connected to the PLC, collectively referred to as module offline configuration.

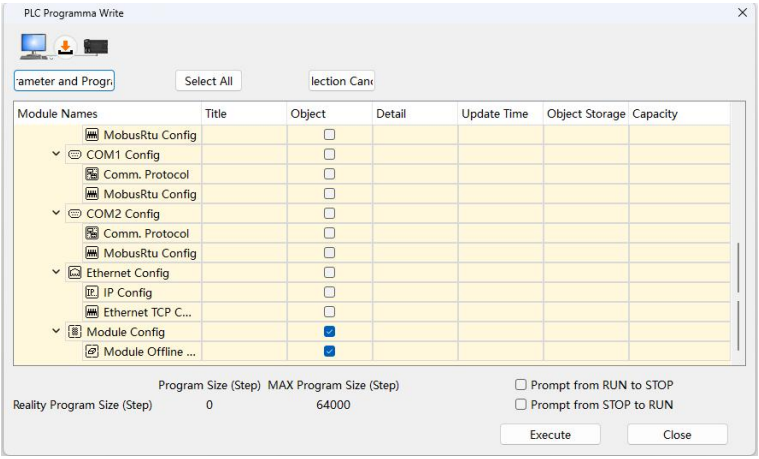
(1) Open the module configuration in the host computer UNISYS and select the module to be connected to the PLC



(2) Double-click the corresponding module to edit the parameters. Here, double-click the first line of configuration.

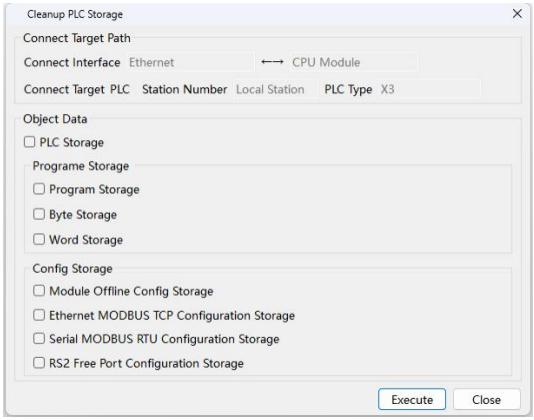


(3) Download module configuration to PLC to complete module offline configuration



(4) If all the above operations are successful, you can use the module normally.

(5) After configuring the module through the host computer, if you want to manually configure it again by writing the module register, you need to clear the current offline configuration first.



(6) UNISYS has both online module readout and offline module readout functions. The online readout is of the module actually connected to the PLC, and the offline readout is of the configuration downloaded from the host computer offline configuration.

(7) If the offline configuration does not match the configuration currently connected to the module, an error will be reported (the error code part of SD8449 is B: configuration does not match), and then the current module configuration will use the previously stored configuration.

3.7.9 Module Configuration Notes

(1) Online configuration refers to updating the configuration directly to the module FLASH by ladder diagram or register method; offline configuration is saved in the PLC and updated to the module FLASH after downloading or when power is restarted.

(2) Pay attention to the priority of online configuration and offline configuration. It is not recommended to use online configuration and offline configuration at the same time. Offline configuration updates the module configuration when power is turned on or after the offline configuration is downloaded, and online configuration updates the module configuration during configuration. The specific configuration is based on the last update.

3.8 Use of BD board module

3.8.1 BD board ID and version description

1. Available BD board modules and corresponding IDs are as follows.

Special registers	Numeric	illustrate
SD8219	H1240	Digital quantity, 2DI4DO
	H1600	Digital quantity, 6DI
	H1060	Digital quantity, 6DO
	H2401	Analog (current), 4AI
	H2202	Thermal resistor, 2RTD
	H2210	Analog (voltage), 2AI1AO
	H230A	(current + RTD), 1AI1RTD
	H2020	Analog 2AO

2. BD board version register

SD8220	100	Decimal 100 means the current BD board software version number is V1.00

3.8.2 Digital BD Board User Manual

Digital DI0-DI7 points correspond to PLC X370-X377, DO0-DO7 points correspond to PLC Y370-Y377. When powered on, the X/Y numbers corresponding to each slot can be obtained according to the ID.

DI0 - DI7	DO0 - DO7
X 370 - X 377	Y370 - Y371

3.8.3 Analog BD Board User Manual

1. BD board configuration command description

Command format : 0x DCBA		illustrate
15~12bit	0-1	0000: Configure the second channel as two-wire / four-wire 0001: Configure the three-wire connection method for channel 2
11~8bit	0-1	0000: Configure channel 1 as two-wire / four-wire 0001: Configure the three-wire connection method for channel 1
7~4bit	0-3	0000: Configure the second channel as PT100 type 0001: Configure the second channel as PT200 four types 0010: Configure the second channel as PT1000 four types 0011: Configure the second channel as Ni1000 four types
3~0bit	0-3	0000: Configure channel 1 as PT100 type 0001: Configure channel 1 as PT200 four types 0010: Configure channel 1 to PT1000 four types 0011: Configure channel 1 to be Ni1000 four types

For example: 0x 0011 means configuring the 1st and 2nd channels as PT200 inputs . Note that the configuration enable flag SM8179 needs to be set after writing the configuration.

2. Description of BD board status register SD8221

Address	Respectively indicate the operating status of the two BD boards
bit0	1: Communication error 0: Normal
bit1	1: Configuration error 0: Normal
bit2	1: Command error 0: Normal
bit3	1: Uncalibrated 0: Normal
Bit4~15	Reserve

3.8.4 Analog BD board address comparison table

1. Address correspondence table of 4AI BD board

Address	illustrate
SD8219	H2401, indicating 4AI BD board
SD8220	100, indicating version number V1.00
SD8221	BD board status
SD8222	Display the value of input channel 1
SD8223	Display the value of input channel 2
SD8224	Display the value of input channel 3
SD8225	Display the value of input channel 4

2. Address correspondence table of 2RTD BD board

Address	illustrate
SD8219	H2202, indicates 2RTD BD board
SD8220	100, indicating version number V1.00
SD8221	BD board status
SD8222	Display the value of RTD input channel 1
SD8223	Displays the value of the second channel of RTD input
SD8224	Configuration Registers
SM8179	Configuration command enable

3. Address correspondence table of 2AI1AO BD board

Address	illustrate
SD8219	H2210, indicating 2AI1AO BD board
SD8220	100, indicating version number V1.00
SD8221	BD board status
SD8222	Display the value of input channel 1
SD8223	Display the value of input channel 2
SD8224	Display output channel 1 value

4. 1 AI1 RTD BD board address table

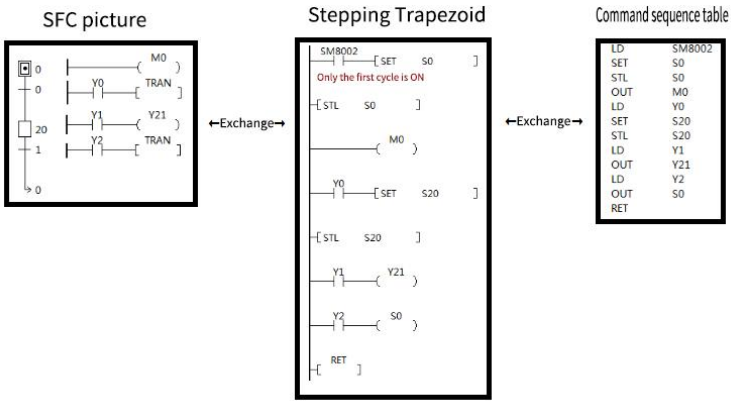
Address	illustrate
SD8219	H230A, indicates 1AI1RTD BD board
SD8220	100, indicating version number V1.00
SD8221	BD board status
SD8222	Display the value of input channel 1
SD8223	Display the value of RTD input channel 1
SD8224	Configuration Registers
SM8179	Configuration command enable

3.9 SFC Program and Step Ladder Diagram

This chapter mainly explains the essentials of " SFC " and "step ladder diagram" programming and sequential control actions using UNISYS .

3.9.1 Overview

In the X series programmable controller, you can use SFC diagram (sequential function chart) or step ladder diagram to realize sequential control, treat state S as a control process, and write the sequential control program of input conditions and output control in it. Since the previous process will be inactive when the process advances, the machine can be controlled in a simple order of each process, making the design of sequential control simple. In addition, SFC program and step ladder diagram instructions are programmed according to established rules, so they can be converted to each other. Therefore, the substantive Content is completely the same and can also be used as a familiar relay ladder diagram. The following are examples of conversion between SFC diagram, step ladder diagram, and instruction table:



3.9.2 Related command action description

1. STL instruction: After the state is ON, the ladder diagram (internal ladder diagram) connected to it is activated through the STL contact.

2. TRAN instruction: used for state transfer, transfer to the next state.

3. RET instruction: put it at the end of the step sequence program.

4. OUT instruction: After Y and M are set as output in the process (state step), they will be reset at the next state transfer, but can still be set in the next process, and then reset at the next state transfer; if the T element is connected after OUT, then after the state switch, the data, contacts, and coils of T0~T245 and T256~T511 are all reset, and T246~T255 maintains the data value and contact state before the transfer; if the C element is connected after OUT, after the state switch, the data value and contact state before the transfer are maintained; in the step ladder diagram, OUT S means jumping to the corresponding state.

5. SET instruction: After Y and M are set in the process (state step), the original ON output will remain at the next state transfer; SET S in the step ladder diagram indicates transfer to the corresponding state.

6. RST instruction: After Y and M are reset by RST in the process (status step), they can still be set in the following processes; reset T and C, data values, contacts, and coils will be reset, but the reset status of T246~T255 and C200~C255 will be maintained; RST S in the step ladder diagram indicates the corresponding state of reset.

3.9.3 Related special auxiliary registers

Register	illustrate	Content
SM8046	STL status action (processed when executing END instruction)	When SM8047 is turned on, any one of S0 ~ S899, S1000 ~ S4095 is turned on.
SM8047	STL monitoring is effective (processed when the END instruction is executed)	SM8047 is connected, SD8040~SD8047 are effective
SM8048	Signal alarm action	When SM8049 is on, any one of S900~S999 is on.
SM8409	Signal alarm is effective	After SM8049 is turned on, the action of SD8049 is effective
SD8040	ON state number 1	The smallest number of the active state in states S0~S899, S1000~S4095 is saved to SD8040. The next ON state is saved to the next register, with a maximum of eight points, and the default is -1.
SD8041	ON state number 2	
SD8042	ON state number 3	
SD8043	ON state number 4	
SD8044	ON state number 5	
SD8045	ON state number 6	
SD8046	ON state number 7	
SD8047	ON state number 8	
SD8049	Minimum number in ON state (when SM8049 is valid)	When SM8049 is ON, the minimum number of the signal alarm relays S900~S999 that are ON is saved. The default is -1.

3.9.4 SFC Programming Instructions

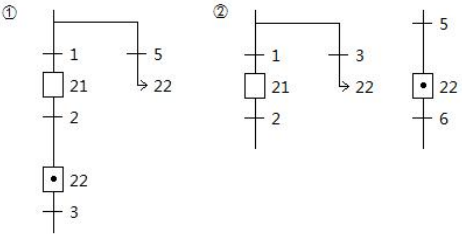
1. SFC programming related symbols

Symbol	illustrate	Remark
	Initial process	The status in the initial process can only be S0~S9
	Process	The status in non-initial process can only be S10~S4095
	Transfer Symbol	If the conditions are met, jump to the next process (status step)
	Select branch	Each branch is limited to 8 circuits or less
	Select Confluence	
	Parallel Branches	Each branch is limited to 8 circuits or less
	Parallel merging	
	Jump symbol	Jump to the corresponding state step
	Reset Symbol	Reset the corresponding status bit

2. The form of SFC process

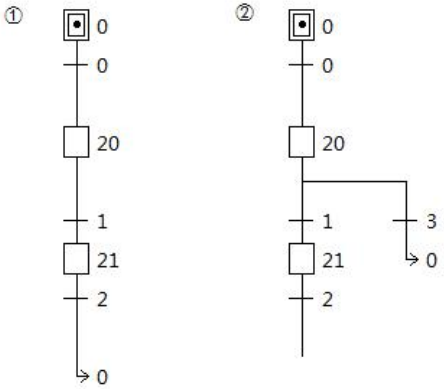
(1) Process jump

Jump to the state below, or transfer to a state outside the process, using a jump symbol to indicate the target state to be transferred.



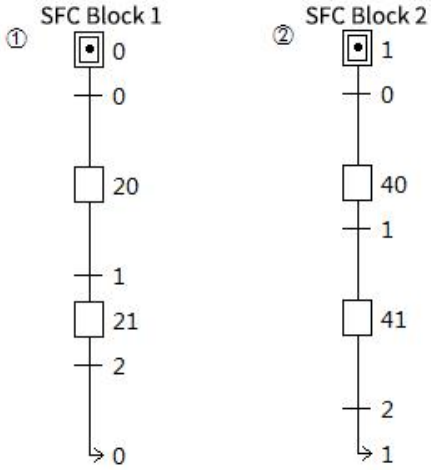
(2) Process duplication

Transferring to the top and re-executing this process is called repetition, and the jump symbol is used to indicate the target state to be transferred.



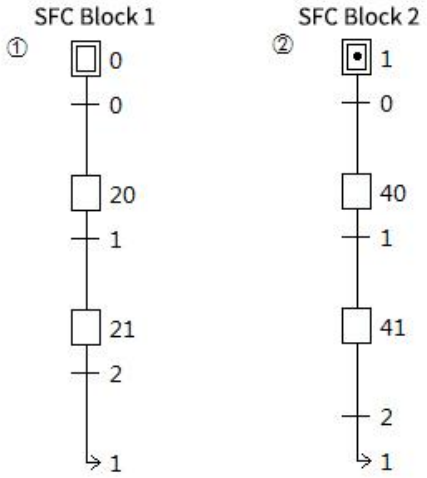
(3) Process separation

An SFC program with multiple initialization states is programmed in multiple blocks.



(4) Jump out of the process

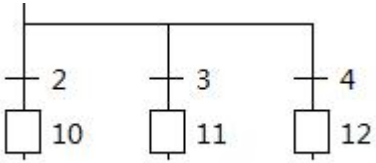
Status transfer can also be performed between SFC programs created by separating program blocks.



3. Branch and merge status of SFC

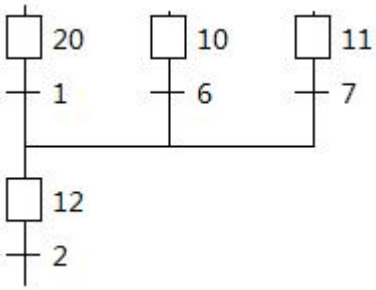
(1) Select a branch

After the branch, write the transfer condition.



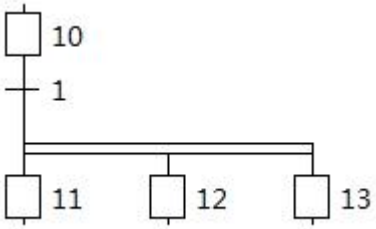
(2) Select Confluence

Write the transition conditions first, then merge them.



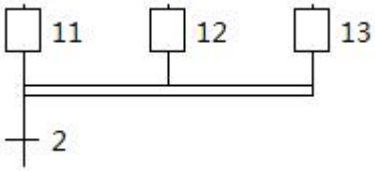
(3) Parallel branches

Write the transfer condition first, then branch.



(4) Parallel merging

After the convergence, write the transition conditions.

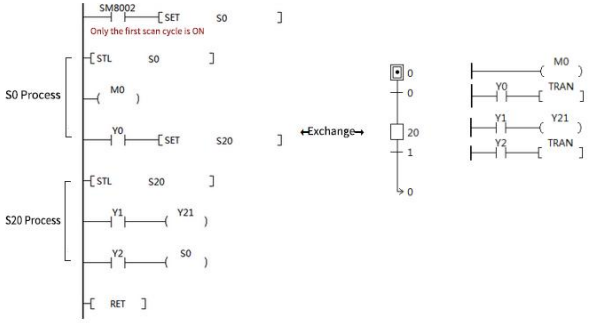


3.9.5 Step ladder diagram programming instructions

1. Conversion relationship with SFC diagram

The essence of step ladder diagram is the same as SFC, but it is expressed in the style of relay ladder diagram.

Down:



In:

SET S means transfer to the corresponding state S;

STL S means executing the process corresponding to S (internal ladder diagram);

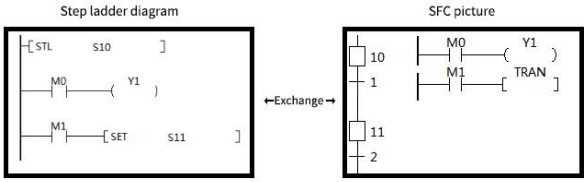
OUT S means jumping to the corresponding state S;

RET is placed at the end of the step ladder diagram, mainly to distinguish it from the ordinary ladder diagram.

2. The process form of step ladder diagram

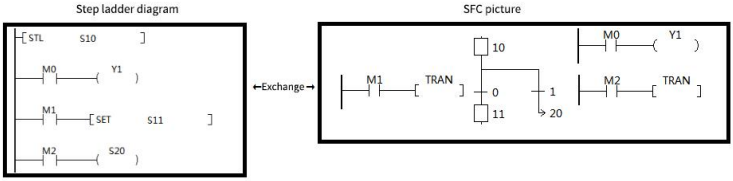
(1) Transfer target

When the condition in front of SET S is met, it will transfer to the next corresponding state.



(2) Jump/Repeat

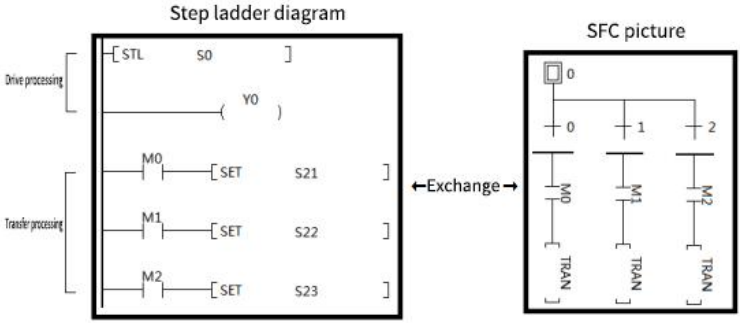
Use OUT S to indicate the state to jump/repeat.



3. Branch and merge status of step ladder diagram

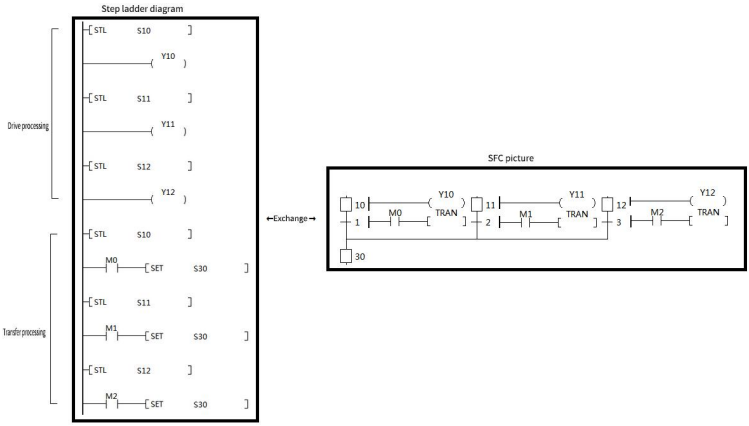
(1) Select a branch

The drive process is executed first, followed by the transfer process, and all transfer processes are executed in sequence.



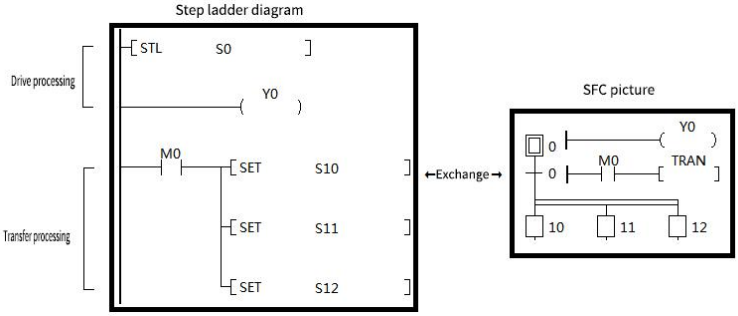
(2) Select Confluence

The driver program of the pre-merge state is executed first, and then the transition processing of the merge state is executed in sequence.



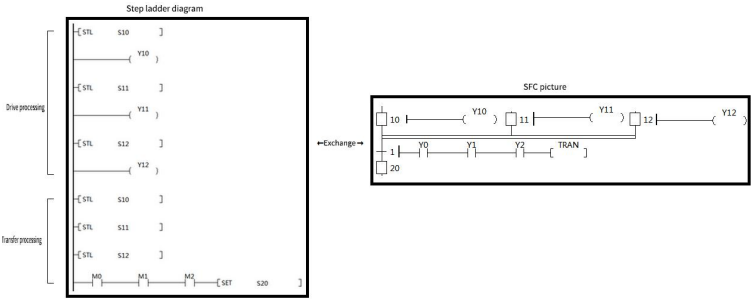
(3) Parallel branches

The drive process is executed first, followed by the transfer process, and all transfer processes are executed in sequence.



(4) Parallel merging

The driver program of the pre-merge state is executed first, and then the transition processing of the merge state is executed in sequence.



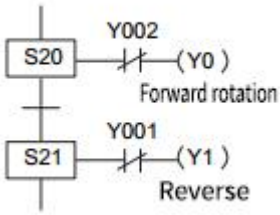
Notes on SFC or step ladder programming

(1) The initial process status number can only be S0~S9.

(2) Status S numbers cannot be reused.

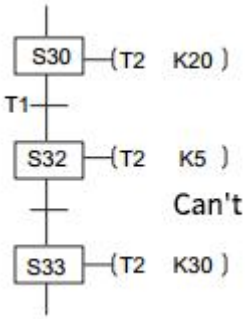
(3) Output interlock

During the state transition process, two states will be ON at the same time for only a moment (one operation cycle). Therefore, in order to prevent a pair of outputs that cannot be turned on at the same time from being ON at the same time, set an interlock outside the programmable controller. In addition, please also perform the mutual interlock as shown in the figure in the program.



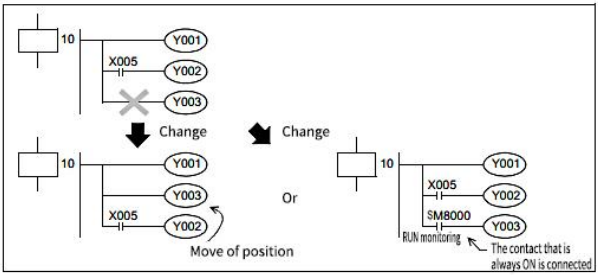
(4) Reuse of timers

The timer coil is the same as the output coil. The same soft element can be programmed in different states, but it cannot be programmed in adjacent states. If it is programmed in adjacent states, the timer coil will not be disconnected when the process is transferred, and the current value will not be reset .



(5) Output driver

As shown in the figure below, starting from the bus in the state, once the LD or LDI instruction is written, instructions that do not require contacts can no longer be written . Please make changes as shown in the figure below .



(6) Sequence control instructions that can be processed within the state

Order / State		LD/LDI/LDP/LDF, AND/ANI/ ANDPIANDF, OR/ORIORF, INV, OUT, SET/RST, PLS/PLF	ANB/ORB MPS/MRD/MPP	MC/MCR
Initial state/normal state		Available	Available	Unavailable
Branching, merging state	Output processing	Available	Available	Unavailable
	Transfer processing	Available	Unavailable	Unavailable

(7) STL instructions cannot be used in interrupt programs and subroutines .

(8) The use of jump instructions is not prohibited within STL instructions, but their actions are complex and are not recommended.

3.10 PID calculation

3.10.1 Instruction operands

Operands	Content
S1	Target value (SV)
S2	Feedback value (PV)
S3	Parameter data register start location
D	Output value (MV)

3.10.2 Functional action description

After executing the program to set the target value S1 , measured value S2 , and parameters S3 to S+6 , the calculation result (MV) is saved in the output value D every sampling time S3.

3.10.3 PID register list

Address	Function	illustrate
S3+0	Sampling period (T)	The setting range is 1~32767 (ms), but it must be greater than the PLC program scanning cycle
S3+1	Mode Configuration (ctl_state)	Bit0: 0 = forward action; 1 = reverse action Bit1: 0 = Input change alarm is invalid; 1 = Input change alarm is valid Bit2: 0 = Output change alarm is invalid; 1 = Output change alarm is valid Bit3: 0 = PID not in action; 1 = PID in action (read only) Bit4: 0 = manual PID; 1 = self-tuning PID Bit5: 0 = Output value upper and lower limit settings are invalid; 1 = Output value upper and lower limit settings are valid Bit6: 0 = auto-tuning is not running; 1 = auto-tuning is in progress (read only) Bit7: 0 = Self-tuning is not completed; 1 = Self-tuning is completed (read only) Bit8~Bit15: 0x00=incremental; 0x01=positional; 0x02=with integral optimization mode (temperature control adaptation);
S3+2	Input filter constant (a)	0 ~ 99 (%), 0 = no input filtering
S3+3	Proportional gain (KP)	1 ~ 32767 (%)
S3+4	Integration time (TI)	0 ~ 32767 (100ms unit), 0 = no integration processing
S3+5	Differential gain (KD)	0~32767 (%)
S3+6	Derivative time (TD)	0 ~ 32767 (10ms unit), 0 = no differential processing
S3+ (7~19)	PID operation is occupied by internal processing, please do not change the data	
S3+20	Input change (increase side) alarm setting value	0 ~ 32767, (valid when bit1 of ctl_state = 1)

Address	Function	illustrate
S3+21	Input change (minus side) alarm setting value	0 ~ 32767, (valid when bit1 of ctl_state = 1)
S3+22	Output change (increase side) alarm setting value	0 ~ 32767, (valid when bit2=1 and bit5=0 of ctl_state)
	Output upper limit setting value	-32768 ~ 32767, (valid when bit2=0 and bit5=1 of ctl_state)
S3+23	Output change (minus side) alarm setting value	0 ~ 32767, (valid when bit2=1 and bit5=0 of ctl_state)
	Output lower limit setting value	-32768 ~ 32767, (valid when bit2=0 and bit5=1 of ctl_state)
S3+24	Alarm output	Bit0 input change (increase side) overflow Bit 1 input change (decrement side) overflow Bit2 output change (increase side) overflow Bit3 output change (decrement side) overflow (Effective when bit1=1 and bit2=1 of ctl_state)
S3+25	No definition yet	Used when setting parameters
S3+26	Upper limit of output value (ULV)	Maximum output value of output value (MV) (ULV) setting (used when setting parameters) 0-100 (%)
S3+27	Output value lower limit (LLV)	Minimum output value (MV) (0-100) (%) (LLV) setting (used when setting parameters)
S3+28	No definition yet	Used when setting parameters
S3+29	Auto-tuning method selection Auto-tuning parameter configuration selection	bit1-bit0 value selects the auto-tuning method 0: Ziegler-Nichols method/Relay method 1, 2, 3 are reserved temporarily, the default value is 0 Bit3-bit2 Select the output parameter type 3: Fast Mode 2: Medium speed mode 1: Slow mode 0: Very slow mode (default) Bit4: When it is 1, the fast mode is selected for the heating process, and the slow mode parameters are used for the insulation process. When it is 0, it is operated with the output parameter type (not supported yet)

3.10.4 Parameter error code

Setting conditions	Error Code
The number of PID instructions exceeds the range	K6746 (need to be tested)
The sampling time (Ts) is outside the applicable range (Ts ≦ 0)	K6730(need to be tested)
PID instruction type out of range	K6731(need to be tested)
The input filter constant (α) is outside the applicable range (α<0 or 100≦α)	K6732((Need to be tested))

Setting conditions	Error Code
The proportional gain (KP) is outside the target range (KP<0)	K6733 (need to be tested)
The integral time (TI) is outside the applicable range (TI<0)	K6734 (need to be tested)
The differential gain (KD) is outside the target range (KD<0)	K6735 (need to be tested)
The derivative time (TD) is outside the target range (TD<0)	K6736 (need to be tested)
Integral gain (KI is outside the target range (KI<0)	K6737 (temporarily reserved)
PV exceeds (PV<-32768 or 32767<PV)	K6742 (Normally no, but unforeseen unexpected situations)
Deviation exceeded (EV<-32768 or 32767<EV)	K6743 (Normally no, but unexpected situations are difficult to predict)
PID calculation result exceeds (outside -32768 to 32767)	K6747 (Normally no, but unexpected situations are difficult to predict)
PID output upper limit setting value < output lower limit setting value	K6748 (need to be tested)
PID input change alarm setting value, output change alarm setting value abnormal (setting value < 0)	K6749 (need to be tested)

3.10.5 Notes

(1) For the two configuration registers S3+17 and S3+18, if the PLC does not have a power-off retention function or does not use self-tuning, MOV assignment is required when temperature integral optimization is required.

(2) If the operation cycle T<=0, error 6730 will be reported. If the incremental and position modes are set to T<100ms, the range is limited to T=100ms. If the temperature control mode is set to T<1000ms, the range is limited to T=1000ms. T can be modified during PID operation.

(3) A maximum of 16 PID operations are supported. However, please note that the soft element numbers used in the operation cannot be repeated.

4. Appendix

4.1 Error code Content

Error handling process:

(1) First read SD8060 to find the error type. The error level can be determined based on the error type, error or warning. Communication errors (serial communication, module bus light) will not be written.

(2) Check the error code in the corresponding error register according to the error type, and check the error step of the corresponding error. Only operation errors have error steps.

(3) Error description or diagnostic information can be displayed based on the error code in the error register.

(4) Specific error issues are shown in the table below.

Error Type	Error type code	Error code number	Error description	Error Level	Diagnostic Information
PLC hardware error	8061	6101	RAM Errors	Error	It may be that the external flash fails to read or write when powered on, or is damaged.
		6105	Watchdog timer timeout error		The sampling (computation time) exceeds the value of SD8000. Please check the program.
		6116	Stack Error		There is a problem with the storage of P tags in the program
Parameter error	8064	6402	Memory capacity setting error		Check whether the system parameter area settings are consistent with the parameter range supported by the current PLC
		6403	The setting of the hold area is incorrect.		
		6404	Error in setting the comment area		
		6409	RUN terminal setting error		
Syntax Error	8065	6501	Incorrect combination of instruction, software symbol and software number		This error occurs when the instruction is used incorrectly. Please check the instruction usage. It may also be that the program is incomplete due to a sudden power outage during the program download.
		6504	Duplicate tag numbers		
		6505	Soft element number out of range		
		6506	Undefined instruction used		
		6508	Interrupt input (I) definition error		
		6509	Program Error		
		6510	The size relationship of MC's nested numbers is wrong		

Error Type	Error type code	Error code number	Error description	Error Level	Diagnostic Information
Circuit/process error	8066	6610	LD and LDI are used more than 9 times in a row		This error occurs when there is an incorrect combination of instructions as a whole circuit block, or when the relationship between paired instructions is incorrect. Check the relationship between the instructions.
		6617	The command that should start from the bus is not connected to the bus		
		6618	Instructions that can only be used in the main program are outside the main program : STL, MC, MCR , RET		
		6619	There are instructions that cannot be used between FOR-NEXT: STL, RET, MC, MCR, I, IRET		
		6620	FOR-NEXT nesting exceeded (maximum 5 levels)		
		6621	FOR NEXT quantity relation error		
		6623	No MC instruction		
		6624	No MCR instruction		
		6625	STL has been used more than 9 times in a row		
		6626	There are instructions that cannot be used between STL-RET: MC, MCR, I, SRET, IRET		
		6627	No STL instructions		
		6629	No P, I		
		6634	The number of nested CALLs exceeds 6.		
		6637	Branch Error		
Operation error	8067	6701	① There is no jump target address of CJ or CALL ② The result of the address modification is out of bounds or incorrect	Warning	These are errors that occur during the execution of operations. Please modify the program or check the Contents of the operands of the application instructions. Even if there are no syntax or loop errors, for example, the soft element address exceeds its maximum range.
		6705	The operand in the application instruction is a soft element outside the specified range.		
		6706	The device number range or data value of the operand of the application instruction exceeds the range.		

Error Type	Error type code	Error code number	Error description	Error Level	Diagnostic Information
		6710	Mismatch between parameters		In shift instructions, the source operand and the destination operand may be duplicated.
		6730	The sampling time (Ts) is outside the applicable range (Ts ≦ 0)		A data error occurs in the setting value of the control parameter or in the PID operation. Please check the Content of the parameter.
		6732	The input filter constant (α) is outside the applicable range (α<0 or 100≦α)		
		6733	The proportional gain (KP) is outside the target range (KP<0)		
		6734	The integral time (TI) is outside the applicable range (TI<0)		
		6735	The differential gain (KD) is outside the applicable range (KD<0 or 201≦KD)		
		6736	The derivative time (TD) is outside the target range (TD<0)		
		6742	The measured value change exceeds (ΔPV<-32768 or 32767<ΔPV)		Each parameter continues to operate at the maximum or minimum value
		6743	Deviation exceeded (EV<-32768 or 32767<EV)		
		6747	PID calculation result exceeds (outside -32768 to 32767)		
		6748	PID output upper limit setting value < output lower limit setting value		Please confirm whether the object settings are correct.
		6749	PID input change alarm setting value, output change alarm setting value abnormal (setting value < 0)		
		6764	The input or input terminal is occupied by other functions		It may be that other counters or pulse output instructions occupy the terminal.
		6770	File register instruction write error		The file register is not initialized before writing instructions. Use the relevant instructions to initialize it first.

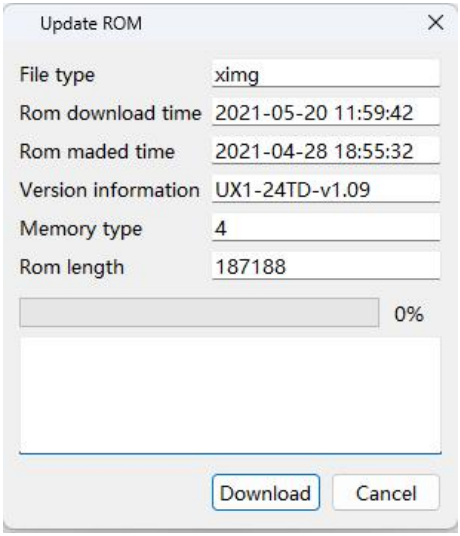
Error Type	Error type code	Error code number	Error description	Error Level	Diagnostic Information
		6774	The corresponding high-speed counter is not enabled		
		6775	A high-speed counter can use up to 6 comparison instructions.		
		6776	Only one of each table comparison instruction can be used.		
Serial communication error	COM0	6201	Serial communication parity, overflow, frame error		
		6203	Serial communication data check error		
		6204	Serial communication data format error		
	COM1	6301	Serial communication parity, overflow, frame error		
		6303	Serial communication data check error		
		6304	Serial communication data format error		
	COM2	3801	Serial communication parity, overflow, frame error		
		3803	Serial communication data check error		
		3804	Serial communication data format error		
Module bus error	/	N*1000+2 (N is the module position number 1-7)	Module N connection error		It may be that the PLC cannot allocate memory for the module or the PLC does not support the module
		N*1000+3 (N is the module position number 1-7)	Module N connection is disconnected		It may be that the working environment is too disturbed or the power supply is unstable.
		N*1000+4 (N is the module position number 1-7)	Module N configuration error		The current module connection status does not match the offline configuration Content

4.2 PLC Firmware Upgrade

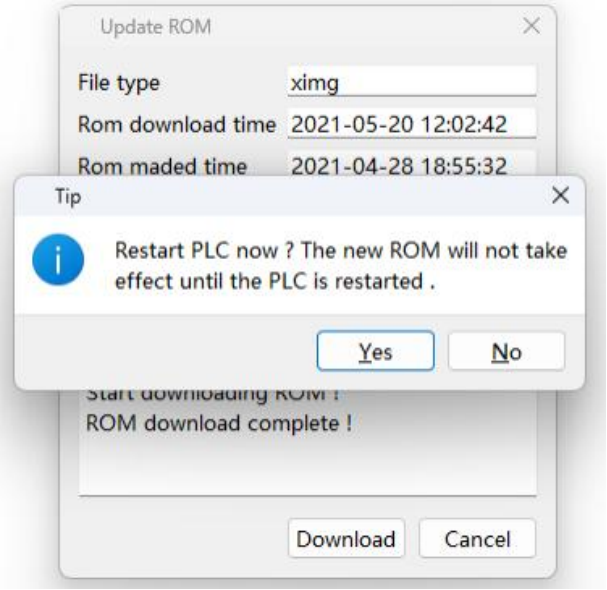
The firmware upgrade function can only be used for PLC versions after V1.37 of the X1, X2, and X3 series. The specific upgrade method is as follows.

1. Before upgrading, power off and restart the PLC. This step is optional and is to save the previous PLC data.

2. Open the PLC upgrade function in the PLC of the host computer, select the .ximg file (here we take X1 as an example, the upgrade method for all firmware of the X series is the same), and then click Download



3. After downloading, click Restart and then observe the upgraded phenomenon.



4. Description of the phenomenon after the upgrade is completed

During the upgrade process, the RUN/STOP=>BATTERY=>ERROR indicators flash in sequence. After the upgrade is completed, there will be three phenomena

①The upgrade is successful, RUN/STOP/BATTERY/ERRO flashes once at the same time, and the new PLC firmware is started normally;

②Firmware check fails, RUN/STOP/BATTERY/ERRO flashes three times at the same time, and the old PLC firmware is started normally;

③Firmware upgrade failed, RUN/STOP/BATTERY/ERRO flashes repeatedly at the same time, unable to enter PLC firmware, restart crashes, at this time you can only use "XPLC Firmware Upgrade Program" to re-update the firmware.