PLC **SOFTWARE MANUAL**

Soft Components Functions
Basic Program Instructions
Applied Instructions
High Speed Counter (HSC)
Pulse Output
Communication Function
PID Control Function
C Language Function Block
Sequential Function BLOCK
Special Function Instructions
Program Application Samples

XC Series PLC Software Manual

Index

Chapter 4 **Applied Instructions**

Chapter 5 **High Speed Counter (HSC)**

Chapter 6 **Pulse Output**

Chapter 7 **Communication Function**

Chapter 8 **PID Control Function**

Chapter 9 **C Language Function Block**

Chapter 10 **Sequential Function BLOCK**

Chapter 11 **Special Function Instructions**

Chapter 12 **Program Application Samples**

1 Program Summary

XC Series PLCs differ from the controllers in that the signal and execution of the program occur in the controller. In this chapter, we begin with the program forms, introduce the main features, the supported two program languages etc.

1-1 Program Controller Features

Program Language

XC series PLCs support two kinds of programming language; Instruction List and Ladder, the two languages can convert to each other.

Program Security

The program is encrypted to prevent unlawful copying or modification. When uploading the encrypted program, you will be asked to input a password. This maintains the user's Copyright.

Program Comments

When the user program becomes too long, adding comments to the program and its soft components may be necessary.

Offset Function

Adding offset appendix (like X3[D100], M10[D100], D0[D100]) behind coils, data registers can realize indirect addressing. For example, when D100=9, X3[D100]=X14; M10[D100]=M19, D0[D100]=D9

Rich Basic Functions

- With enough basic instructions XC Series PLCs can fulfill basic sequential control; data moving and comparing; arithmetic operation; logic control; data loop and shift etc.
- XC Series PLCs also support special comparisons; high speed pulse; frequency testing; precise time; PID control: position control etc. for interruption, high speed counter (HSC).

C Language Function Block

XC Series PLCs support C language function block. Users can call the edited function block freely. This function reduces the program size greatly.

Stop when Power ON Function

XC Series PLCs support "Stop when Power ON PLC" function. With this function, if there is a serious problem whilst the PLC is running, this function will allow the system to stop all output immediately.

Communication Function

XC series PLCs support many communication formats, for example, Modbus communication, CAN-Bus communication and Free Format communication. Via a special network module PLCs can also be connected to Ethernet or GPRS net.

1-2 Programming Language

1-2-1 Types of Language Available

XC Series PLCs support two types of program language:

Instruction List

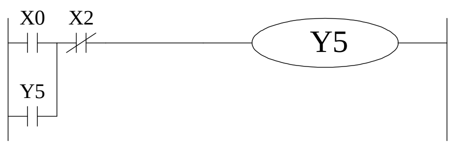
Instruction list inputs in the form of “LD”, “AND”, “OUT” etc. This is the basic input form of the programs, but it’s hard to read and understand;

E.g.:	Step	Instruction	Soft Components
	0	LD	X000
	1	OR	Y005
	2	ANI	X002
	3	OUT	Y005

Ladder List

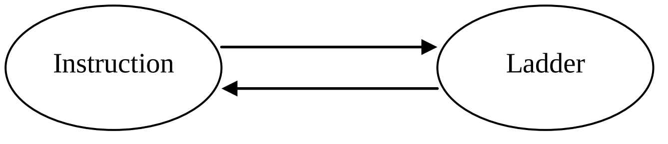
With sequential control signal and soft components, it is possible to draw the sequential control graph on the program interface, this method is called “Ladder”. This method uses coil signs etc. to represent sequential circuits, so it’s easier to understand the program. Meantime, it allows monitoring of the PLC showing the circuit’s status.

E.g.:



1-2-2 Alternation

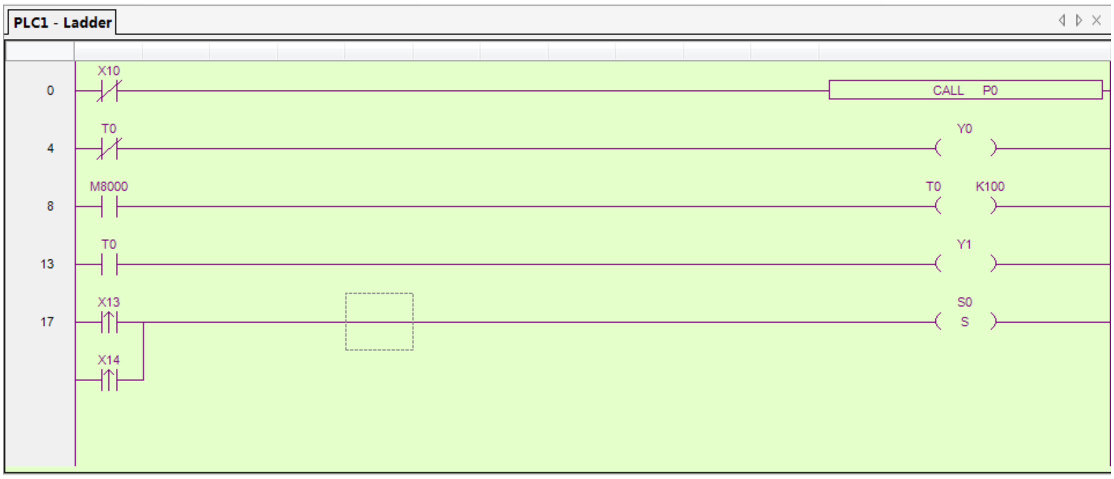
The above two methods can convert to ech other freely:



1-3 Programming Formats

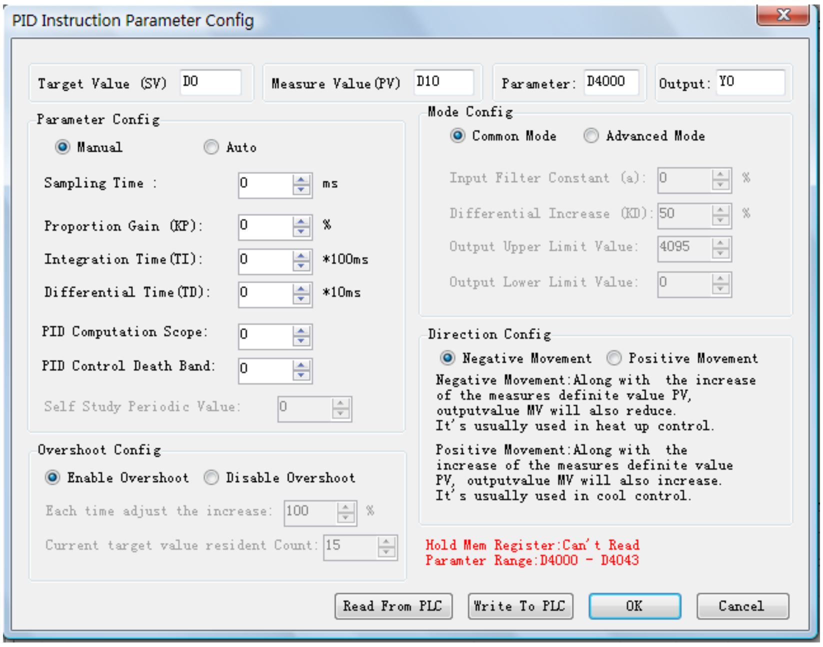
Direct Input

The above two program methods allow input in the corresponding interface separately, however, in the ladder window, there is an instruction hint function, this improves the program efficiency greatly.



Panel Configuration

Some of the functions, like PID anf high speed counters, have a faceplate wizard which help guide the user when inputing the configuration and settings.



2 Soft Component's Functions and Actions

In chapter 1, we briefly covered the program languages of XC Series PLCs. However, the most important element to a program is the operands. These elements relate to the relays and registers inside the controller. In this chapter, we will describe the functions and methods of using these.

2-1 Summary of the Soft Components

There are many relays, timers and counters inside PLCs. They all have countless NO (Normally ON) and NC (Normally Closed) contactors. Connecting these contactors with the coils will make a sequential control circuit. Below, we will introduce these soft components briefly;

Input Relay (X)

- Usage of the input relays

  The input relays are used to accept the external ON/OFF signal, we use **X** to state.

- Address Specify Principle
  - In each basic unit, specify the ID of input relay, output relay in the form of X000~X007，X010~X017…,Y000~Y007，Y010~Y017… (octal form).
  - The expansion module's ID obeys the principle of channel 1 starts from X100/Y100, channel 2 starts from X200/Y200… 7 expansions can be connected in total.
- Points to pay attention to when using:
  - For the input relay's input filter, we use digital filter. Users can change the filter parameters via relate settings.
  - PLCs are equipped with with more relays than are required for the input/output points, these can be utilized as auxiliary relays, program as normal contactors/coils.

Output Relay (Y)

- Usage of the output relays

Output relays are the interface of drive external loads, represent with sign Y;

- Address Assignment Principle
  - In each basic unit，assign the ID of output relays in the form of Y000~Y007, Y010~Y017… this octal format.
  - The ID of expansion obeys the principle of: channel 1 starts from Y100, channel 2 starts from Y200… 7 expansions could be connected totally.

Auxiliary Relays (M)

Auxiliary relays are equipped inside PLC, represent with the sign of M;

- Address assignment principle

In basic units, assign the auxiliary address in decimal form.

- Points to note:
  - This type of relay differs from the input/output relay, it can't be used to take an external load, it can only use in program.
  - A retentive relay can keep its ON/OFF status in case of PLC power OFF.

Status Relays (S)

- Usage of status relays

  Used as relays in Ladder, represent with "S"

- Address assignment principle

  In basic units, assign the ID in decimal form.

- Points to note:

  If not used as operation number, they can be used as auxiliary relays, program as normal contactors/coils. They can also be used as signal alarms, for external diagnosis.

Timer (T)

- Usage of the timers

Timers are used to calculate the time pulse like 1ms, 10ms, 100ms etc. when the set value is reached, the output contactor acts, represent with "T"

- Address assignment principle

In basic units, assign the timer's ID in decimal form, but divide ID into several parts according to the clock pulse, accumulate or not. Please refer to chapter 2-2 for details.

- Time pulse

There are three specifications for the timer's clock pulse: 1ms, 10ms, 100ms. If 10ms timer is selected, then timing is carried out in 10ms pulses.

- Accumulation/not accumulation

The times are divided into two modes: accumulation time means even the timer coil's driver is OFF, the timer will still keep the current value; while the not accumulation time means when the count value reaches the set value, the output contact acts, the count value clears to 0.

Counter (C)

To facilitate different application and purposes, we can divide the counters to different types as detailed below:

- For internal count (for general use/Power OFF retentive usage)
  - 16 bits counter: for increment count, the count range is 1~32,767
  - 32 bits counter: for increment count, the count range is 1~2,147,483,647
  - These counters can be used by PLC's internal signal. The response speed is one scan cycle or longer.
- For High Speed Count (Power OFF retentive)
  - 32 bits counter: for increment/decrement count, the count range is -2,147,483,648~ +2,147,483,647

  (single phase increment count, single phase increment/decrement count, AB phase cont) The counters are tied to specific digital input channels.
  - The high speed counter can count 80KHz frequency, it synchronizes with the PLC's scan cycle.

Data Register (D)

- Use of Data Registers

  Data Registers are used to store data, represented by "D"
- Addressing Form

  The data registers in XC Series PLCs are all 16 bits (the highest bit is the sign bit), by combining two data registers together 32 bit operationcan be achieved (the highest bit is the sign bit) data process.
- Points to note:

  As with other soft components, data registers also have common usage type and Power OFF retentive type.

FlashROM Register (FD)

- Usage of FlashROM registers

  FlashROM registers are used to store data soft components, represent with “FD”
- Addressing Form

  In basic units, FlashROM registers are addressed in decimal form.
- Points to note:

  Even if the battery power is OFF, this area can retain data. So this area is used to store important parameters. FlashROM can write about 1,000,000 times, and it takes time at every write. Too many write instructions can cause permanent damage of the FD address.

Constant (B) (K) (H)

- In every type of data in PLC, B represents Binary, K represents Decimal, H represents Hexadecimal. They are used to set timers and counters values, or operands of application instructions.

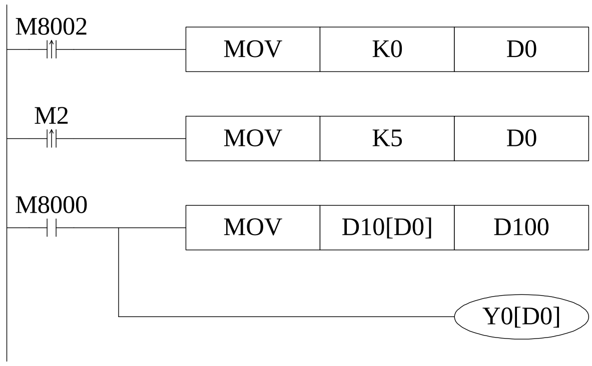
2-2 Structure of Soft Components

2-2-1 Memory Structure

There are many registers in XC Series PLCs. In addition to the common data registers D and FlashROM registers, we can also make registers by combining bit soft components.

Data Register (D)

- For common use, 16 bits
- For common use, 32 bits (via combine two sequential 16 bits registers)
- For power off retentive usage, the retentive zone can be modified
- For special usage, occupied by the system, these are special function registers used by the system
- For offset usage (indirect specifies)



- Form: Dn[Dm]、Xn[Dm] 、Yn[Dm] 、Mn[Dm] etc.

In the above sample, if D0=0, then D100=D10, Y0 is ON.

If M2 turns from OFF to be ON, D0=5, then D100=D15, Y5 is ON.

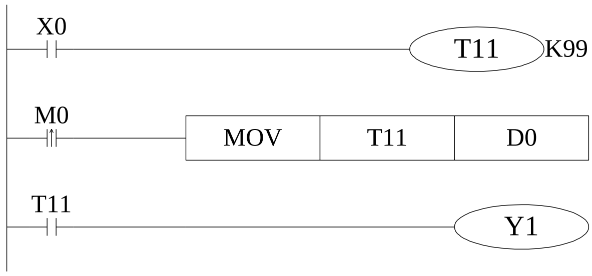
Therein, D10[D0]=D[10+D0]，Y0[D0]=Y[0+D0]。

- The word offset combined by bit soft components: DXn[Dm] represents DX[n+Dm]。
- The soft components with offset, the offset can be represented by soft component D.

Timer (T)

- For common usage, 16 bits, represent the current value of timer/counter;
- For common usage, 32 bits, (via combine two sequential 16 bits registers)
- To represent them, just use the letter+ID method, such as T10, C11.

E.g.



FlashROM Register (FD)

- For power off retentive usage, 16 bits
- For power off retentive usage, 16 bits, (via combine two sequential 16 bits registers)
- For special usage, occupied by the system, these are special function registers used by the system

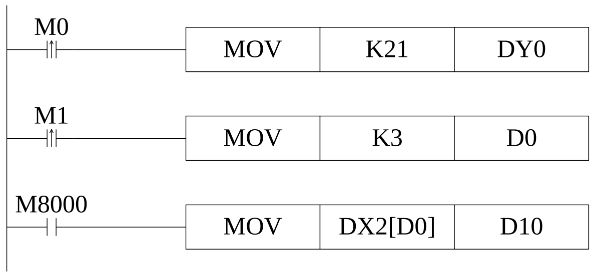
Expansion's Internal Register

- For common usage, 16 bits,
- For common usage, 32 bits, (via combine two sequential 16 bits registers)

Bit Soft Components Combined Register

- For common usage, 16 bits, (via combine two sequential 16 bits registers).
- The soft components which can be combined to be words are: X, Y, M, S, T, C.
- Format: add “D” in front of soft components, like DM10, represents a 16 bits data from M10~M25.
- Get 16 points from DXn, but not beyond the soft components range.

  E.g.:



  - When M0 changes from OFF to be ON, the value in the word which is combined by Y0~Y17 equals 21, i.e. Y0, Y2, Y4 becomes to be ON

2-2-2 BitSoft Components’ Structure

Bit soft components structure is simple, the common ones are X, Y, M, S, T, C however, a bit of a register can also represent:

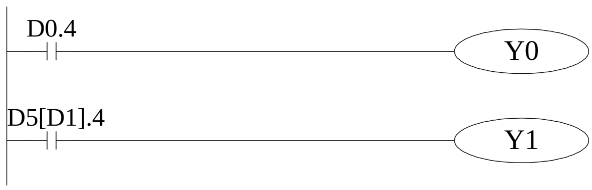
Relay

- Input Relay X, octal type
- Output Relay Y, octal type
- Auxiliary Relay M, S, decimal type
- Auxiliary Relay T, C, decimal type, as the representative method is as with registers, we need to clarify if it’s a word register or bit register according within the register.

Register’s Bit

- Made up by register’s bit, support register D
- Represent method: Dn.m (0≤m≤15): the Nr.m bit of Dn register
- The represent method of word with offset: Dn[Dm].x
- Bit of Word can’t compose to be word again;

  E.g.:



- ➢ D0.4 means when the Nr.4 bit of D0 is 1, set Y0 ON .
- ➢ D5[D1].4 means bit addressing with offset, if D1=5, then D5[D1]

2-3 Soft Components List

2-3-1 Soft Components List

XC1 Series

Mnemonic	Name	Range				points			
		10I/O	16 I/O	24 I/O	32 I/O	10 I/O	16 I/O	24 I/O	32 I/O
I/O points※1	Input Points	X0~X4	X0~X7	X0~X13	X0~X17	5	8	12	16
	Output Points	Y0~Y4	Y0~Y7	Y0~Y13	Y0~Y17	5	8	12	16
X※2	Internal Relay	X0~X77				64			
Y※3	Internal Relay	Y0~Y77				64			
M	Internal Relay	M0~M199【M200~M319】※4				320			
		For Special Usage ※5M8000~M8079				128			
		For Special Usage ※5M8120~M8139							
		For Special Usage ※5M8170~M8172							
		For Special Usage ※5M8238~M8242							
		For Special Usage ※5M8350~M8370							
S	Flow	S0~S31				32			
T	Timer	T0~T23: 100ms not accumulation				80			
		T100~T115: 100ms accumulation							
		T200~T223: 10ms not accumulation							
		T300~T307: 10ms accumulation							
		T400~T403: 1ms not accumulation							
		T500~T503: 1ms accumulation							
C	Counter	C0~C23: 16 bits forward counter				48			
		C300~C315: 32 bits forward/backward counter							
		C600~C603: single-phase HSC							
		C620~C621							
		C630~C631							
D	Data Register	D0~D99【D100~D149】※4				150			
		For Special Usage ※5D8000~D8029				138			
		For Special Usage ※5D8060~D8079							
		For Special Usage ※5D8120~D8179							
		For Special Usage ※5D8240~D8249							
		For Special Usage ※5D8306~D8313							
		For Special Usage ※5D8460~D8469							
FD	FlashROM Register※6	FD0~FD411				412			
		For Special Usage ※5FD8000~FD8011				98			
		For Special Usage ※5FD8202~FD8229							
		For Special Usage ※5FD8306~FD8315							

		For Special Usage [※5]FD8323~FD8335	
		For Special Usage [※5]FD8350~FD8384	

XC2 Series

Mnemonic	Name	Range				Points			
		14 I/O	16 I/O	24/32 I/O	48/60 I/O	14 I/O	16 I/O	24/32 I/O	48/60 I/O
I/O Points[※1]	Input Points	X0~X7	X0~X7	X0~X15 X0~X21	X0~X33 X0~X43	8	8	14/18	28/36
	Output Points	Y0~Y5	Y0~Y7	Y0~Y11 Y0~Y15	Y0~Y23 Y0~Y27	6	8	10/14	20/24
X[※2]	Internal Relay	X0~X1037				544			
Y[※3]	Internal Relay	Y0~Y1037				544			
M	Internal Relay	M0~M2999 【M3000~M7999】[※4]				8000			
		For Special Usage[※5]M8000~M8767				768			
S	Flow	S0~S511 【S512~S1023】[※4]				1024			
T	Timer	T0~T99: 100ms not accumulation				640			
		T100~T199: 100ms accumulation							
		T200~T299: 10ms not accumulation							
		T300~T399: 10ms accumulation							
		T400~T499: 1ms not accumulation							
		T500~T599: 1ms accumulation							
		T600~T639: 1ms precise time							
C	Counter	C0~C299: 16 bits forward counter				640			
		C300~C599: 32 bits forward/backward counter							
		C600~C619: single-phase HSC							
		C620~C629: double-phase HSC							
		C630~C639: AB phase HSC							
D	Data Register	D0~D999 【D4000~D4999】[※4]				2000			
		For Special Usage[※5]D8000~D8511				612			
		For Special Usage[※5]D8630~D8729							
FD	FLASH Register	FD0~FD127				128			
		For Special Usage[※5]FD8000~FD8383				384			

XC3 Series

Mnemonic	Name	Range: 14 I/O	Range: 24/32 I/O	Range: 48/60 I/O	Points: 14 I/O	Points: 24/32 I/O	Points: 48/60 I/O
I/O Points※1	Input Points	X0~X7	X0~X15 X0~X21	X0~X33 X0~X43	8	14/18	28/36
	Output Points	Y0~Y5	Y0~Y11 Y0~Y15	Y0~Y23 Y0~Y27	6	10/14	20/24
X※2	Internal Relay	X0~X1037			544		
Y※3	Internal Relay	Y0~Y1037			544		
M	Internal Relay	M0~M2999 【M3000~M7999】※4			8000		
		For Special Usage※5M8000~M8767			768		
S	Flow	S0~S511 【S512~S1023】※4			1024		
T	TIMER	T0~T99: 100ms not accumulation			640		
		T100~T199: 100ms accumulation					
		T200~T299: 10ms not accumulation					
		T300~T399: 10ms accumulation					
		T400~T499: 1ms not accumulation					
		T500~T599: 1ms accumulation					
		T600~T639: 1ms precise time					
C	COUNTER	C0~C299: 16 bits forward counter			640		
		C300~C599: 32 bits forward/backward counter					
		C600~C619: single-phase HSC					
		C620~C629: double-phase HSC					
		C630~C639: AB phase HSC					
D	DATA REGISTER	D0~D3999 【D4000~D7999】※4			8000		
		For Special Usage※5D8000~D9023			1024		
FD	FlashROM REGISTER※6	FD0~FD1535			1536		
		For Special Usage※5FD8000~FD8511			512		

ED※7	EXPANSION'S INTERNAL REGISTER	ED0~ED16383	16384

XC5 Series

<table>
<tr><th rowspan="2">Mnemonic</th><th rowspan="2">Name</th><th colspan="2">I/O RANGE</th><th colspan="2">POINTS</th></tr>
<tr><th>24/32 I/O</th><th>48/60 I/O</th><th>24/32 I/O</th><th>48/60 I/O</th></tr>
<tr><td rowspan="2">I/O Points※1</td><td>Input Points</td><td>X0~X15
X0~X21</td><td>X0~X33
X0~X43</td><td>14/18</td><td>28/36</td></tr>
<tr><td>Output Points</td><td>Y0~Y11
Y0~Y15</td><td>Y0~Y23
Y0~Y27</td><td>10/14</td><td>20/24</td></tr>
<tr><td>X※2</td><td>Internal Relay</td><td colspan="2">X0~X1037</td><td colspan="2">544</td></tr>
<tr><td>Y※3</td><td>Internal Relay</td><td colspan="2">Y0~Y1037</td><td colspan="2">544</td></tr>
<tr><td rowspan="2">M</td><td rowspan="2">Internal Relay</td><td colspan="2">M0~M3999
【M4000~M7999】※4</td><td colspan="2">8000</td></tr>
<tr><td colspan="2">For Special Usage※5M8000~M8767</td><td colspan="2">768</td></tr>
<tr><td>S</td><td>Flow</td><td colspan="2">S0~S511
【S512~S1023】※4</td><td colspan="2">1024</td></tr>
<tr><td rowspan="7">T</td><td rowspan="7">TIMER</td><td colspan="2">T0~T99: 100ms not accumulation</td><td colspan="2" rowspan="7">640</td></tr>
<tr><td colspan="2">T100~T199: 100ms accumulation</td></tr>
<tr><td colspan="2">T200~T299: 10ms not accumulation</td></tr>
<tr><td colspan="2">T300~T399: 10ms accumulation</td></tr>
<tr><td colspan="2">T400~T499: 1ms not accumulation</td></tr>
<tr><td colspan="2">T500~T599: 1ms accumulation</td></tr>
<tr><td colspan="2">T600~T639: 1ms precise time</td></tr>
<tr><td rowspan="5">C</td><td rowspan="5">COUNTER</td><td colspan="2">C0~C299: 16 bits forward counter</td><td colspan="2" rowspan="5">640</td></tr>
<tr><td colspan="2">C300~C599: 32 bits forward/backward counter</td></tr>
<tr><td colspan="2">C600~C619: single-phase HSC</td></tr>
<tr><td colspan="2">C620~C629: double-phase HSC</td></tr>
<tr><td colspan="2">C630~C639: AB phase HSC</td></tr>
<tr><td rowspan="2">D</td><td rowspan="2">DATA REGISTER</td><td colspan="2">D0~D3999
【D4000~D7999】※4</td><td colspan="2">8000</td></tr>
<tr><td colspan="2">For Special Usage※5D8000~D9023</td><td colspan="2">1024</td></tr>
<tr><td rowspan="2">FD</td><td rowspan="2">FlashROM REGISTER※6</td><td colspan="2">FD0~FD5119</td><td colspan="2">5120</td></tr>
<tr><td colspan="2">For Special Usage※5FD8000~FD9023</td><td colspan="2">1024</td></tr>
<tr><td>ED※7</td><td>EXPANSION'S INTERNAL REGISTER</td><td colspan="2">ED0~ED36863</td><td colspan="2">36864</td></tr>
</table>

XCM Series

Mnemonic	Name	I/O Range		Points	
		24/32 I/O	48 I/O	24/32 I/O	48 I/O
I/O Points[※1]	Input Points	X0~X15 X0~X21	X0~X33	14/18	28
	Output Points	Y0~Y11 Y0~Y15	Y0~Y23	10/14	20
X[※2]	Internal Relay	X0~X1037		544	
Y[※3]	Internal Relay	Y0~Y1037		544	
M	Internal Relay	M0~M2999 【M3000~M7999】[※4]		8000	
		For Special Usage[※5]M8000~M8767		768	
S	Flow	S0~S511 【S512~S1023】[※4]		1024	
T	TIMER	T0~T99: 100ms not accumulation		640	
		T100~T199: 100ms accumulation			
		T200~T299: 10ms not accumulation			
		T300~T399: 10ms accumulation			
		T400~T499: 1ms not accumulation			
		T500~T599: 1ms accumulation			
		T600~T639: 1ms precise time			
C	COUNTER	C0~C299: 16 bits forward counter		640	
		C300~C599: 32 bits forward/backward counter			
		C600~C619: single-phase HSC			
		C620~C629: double-phase HSC			
		C630~C639: AB phase HSC			
D	DATA REGISTER	D0~D2999 【D4000~D4999】[※4]		4000	
		For Special Usage[※5]D8000~D9023		1024	
FD	FlashROM REGISTER[※6]	FD0~FD63		64	
		For Special Usage[※5]FD8000~FD8349		460	
		For Special Usage[※5]FD8890~FD8999			
ED[※7]	EXPANSION'S INTERNAL REGISTER	ED0~ED36863		36864	

※1: I/O points, means the terminal number that users can use to wire the input s/outputs;

※2: X, means the internal input relay, the X beyond Input points can be used as middle relay;
※3: Y, means the internal output relay, the Y beyond Output points can be used as middle relay;
※4: The memory zone in【 】 is power off retentive zone, soft components D、M、S、T、C can change the retentive area via setting. Please refer to 2-3-2 for details;
※5: for special use, means the special registers occupied by the system, can't be used for other purpose. Please refer to Appendix 1.
※6: FlashROM registers needn't set the power off retentive zone, when power is off (no battery), the data will not be lost;
※7: Expansion's internal register ED, requires PLC hardware V3.0 or above;
※8: Input coils、output relays are in octal form, the other registers are in decimal form;
※9: I/Os that are not connected to external devices can be used as fast internal relays;
※10: for the soft components of expansion devices, please refer to related manuals;

2-3-2 Power-off Retentive Zone

The power off retentive area of XC Series PLCs are set as below, this area can be re-set by user:

	Soft components	SET AREA	FUNCTION	System's default value	Retentive Zone
XC1 Series	D	FD8202	Start tag of D power off retentive zone	100	D100~D149
	M	FD8203	Start tag of M power off retentive zone	200	M200~M319
	T	FD8204	Start tag of T power off retentive zone	640	Not set
	C	FD8205	Start tag of C power off retentive zone	320	C320~C631
	S	FD8206	Start tag of S power off retentive zone	512	S0~S31
XC2 Series	D	FD8202	Start tag of D power off retentive zone	4000	D4000~D4999
	M	FD8203	Start tag of M power off retentive zone	3000	M3000~M7999
	T	FD8204	Start tag of T power off retentive zone	640	Not set
	C	FD8205	Start tag of C power off retentive zone	320	C320~C639
	S	FD8206	Start tag of S power off retentive zone	512	S512~S1023
XC3 Series	D	FD8202	Start tag of D power off retentive zone	4000	D4000~D7999
	M	FD8203	Start tag of M power off retentive zone	3000	M3000~M7999
	T	FD8204	Start tag of T power off retentive zone	640	Not set
	C	FD8205	Start tag of C power off retentive zone	320	C320~C639
	S	FD8206	Start tag of S power off retentive zone	512	S512~S1023
	ED	FD8207	Start tag of ED power off retentive zone	0	ED0~ED16383
XC5 Series	D	FD8202	Start tag of D power off retentive zone	4000	D4000~D7999
	M	FD8203	Start tag of M power off retentive zone	4000	M4000~M7999
	T	FD8204	Start tag of T power off retentive zone	640	Not set

	C	FD8205	Start tag of C power off retentive zone	320	C320~C639
	S	FD8206	Start tag of S power off retentive zone	512	S512~S1023
	ED	FD8207	Start tag of ED power off retentive zone	0	ED0~ED36863
XCM Series	D	FD8202	Start tag of D power off retentive zone	4000	D4000~D4999
	M	FD8203	Start tag of M power off retentive zone	3000	M3000~M7999
	T	FD8204	Start tag of T power off retentive zone	640	Not set
	C	FD8205	Start tag of C power off retentive zone	320	C320~C639
	S	FD8206	Start tag of S power off retentive zone	512	S512~S1023
	ED	FD8207	Start tag of ED power off retentive zone	0	ED0~ED36863

For timer T, we can set not only retentive zone, but also set certain timer's retentive zone

Soft Components	Set area	Function	Retentive Zone
T	FD8323	Set the start tag of 100ms not accumulation timer's retentive zone	The set value ~T99
	FD8324	Set the start tag of 100ms accumulation timer's retentive zone	The set value~T199
	FD8325	Set the start tag of 10ms not accumulation timer's retentive zone	The set value~T299
	FD8326	Set the start tag of 10ms accumulation timer's retentive zone	The set value~T399
	FD8327	Set the start tag of 1ms not accumulation timer's retentive zone	The set value~T499
	FD8328	Set the start tag of 1ms accumulation timer's retentive zone	The set value~T599
	FD8329	Set the start tag of 1ms precise timer's retentive zone	The set value~T639

For counter C, we can set not only retentive zone, but also set certain counter's retentive zone

Soft Components	Set area	Function	Retentive Zone
C	FD8330	Set the start tag of 16 bits positive counter's retentive zone	The set value~C299
	FD8331	Set the start tag of 32 bits positive/negative counter's retentive zone	The set value~C599
	FD8332	Set the start tag of single phase HSC's retentive zone	The set value~C619
	FD8333	Set the start tag of dual direction HSC's retentive zone	The set value~C629
	FD8334	Set the start tag of AB phase HSC's retentive zone	The set value~C639

※1：if the whole power off retentive zone is smaller than the segment's retentive area, then the segment's area is invalid. If the total counter's set range is T200~T640, FD8324 value is 150, then the 100ms accumulate timer's retentive area T150~T199 is invalid.

2-4 Input / Output Relays (X, Y)

Number List

XC Series PLC's inputs/outputs are all in octal form, each series numbers are listed below:

Series	Name	Range				Points			
		10I/O	16 I/O	24 I/O	32 I/O	10 I/O	16 I/O	24 I/O	32 I/O
XC1	X	X0~X4	X0~X7	X0~X13	X0~X17	5	8	12	16
	Y	Y0~Y4	Y0~Y7	Y0~Y13	Y0~Y17	5	8	12	16

Series	Name	Range				Points			
		14 I/O	16 I/O	24/32 I/O	48/60 I/O	14 I/O	16 I/O	24/32 I/O	48/60 I/O
XC2	X	X0~X7	X0~X7	X0~X15 X0~X21	X0~X33 X0~X43	8	8	14/18	28/36
	Y	Y0~Y5	Y0~Y7	Y0~Y11 Y0~Y15	Y0~Y23 Y0~Y27	6	8	10/14	20/24

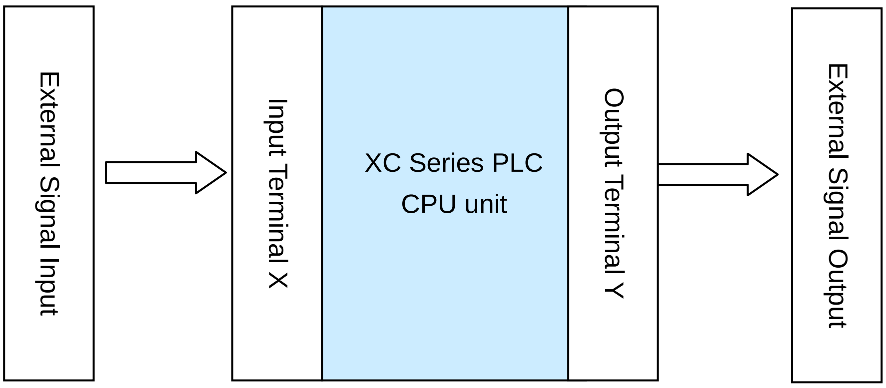
Series	Name	Range			Points		
		14 I/O	24/32 I/O	48/60 I/O	14 I/O	24/32 I/O	48/60 I/O
XC3	X	X0~X7	X0~X15 X0~X21	X0~X33 X0~X43	8	14/18	28/36

	Y	Y0~Y5	Y0~Y11 Y0~Y15	Y0~Y23 Y0~Y27	6	10/14	20/24

Series	Name	Range		Points	
		24/32 I/O	48/60 I/O	24/32 I/O	48/60 I/O
XC5	X	X0~X15 X0~X21	X0~X33 X0~X43	14/18	28/36
	Y	Y0~Y11 Y0~Y15	Y0~Y23 Y0~Y27	10/14	20/24

Series	Name	Range			Points		
		24 I/O	32 I/O	48 I/O	24 I/O	32 I/O	48 I/O
XCM	X	X0~X15	X0~X21	X0~X33	14	18	28
	Y	Y0~Y11	Y0~Y15	Y0~Y23	10	14	20

Function



Input Relay X

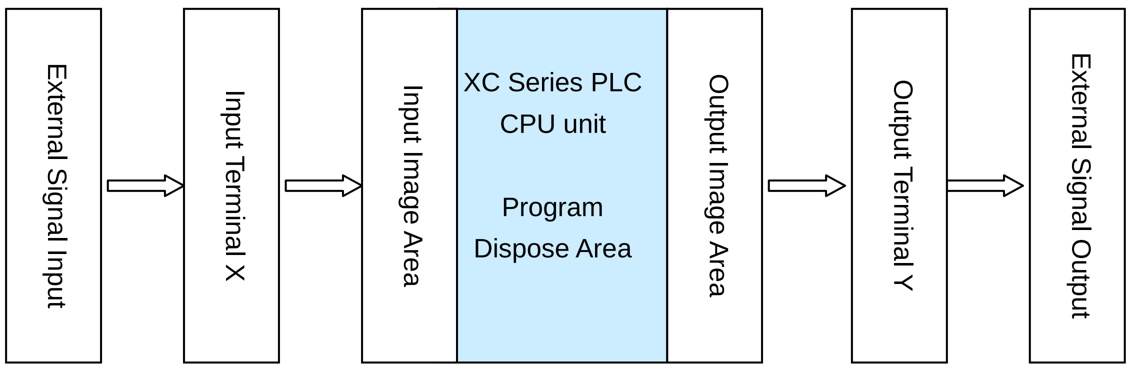
- PLC's input terminals are used to accept the external signal input, while the input relays are a type of optical relays to connect PLC inside and input terminals;
- The input relays have countless normally ON/OFF contactors, they can be used freely;
- The input relays which are not connected with external devices can be used as fast internal relays;

Output Relay Y

- PLC's output terminals can be used to send signals to external loads. Inside PLC, output relay's external output contactors (including relay contactors, transistor's contactors) connect with output terminals.

- The output relays have countless normally ON/OFF contactors, they can be used freely;
- The output relays which are not connected with external devices can be used as fast internal relays;

Execution Order



- Input Disposal
  - ➢ Before PLC executing the program, read every input terminal's ON/OFF status of PLC to the image area.
  - ➢ In the process of executing the program, if the input is changed, the content in the input image area will not change. However, in the next scan cycle, the status of the input will change.

- Output Disposal
  - ➢ Once finished executing all the instructions, transfer the ON/OFF status of output Y image area is set. This will be the actual output of the PLC.
  - ➢ The contacts used for the PLC's external output will act according to the device's response delay time.

2-5 Auxiliary Relay (M)

Number List

The auxiliary relays M in XC Series PLCs are all in decimal form, please refer the details from tables below:

SERIES	NAME	RANGE		
		FOR COMMON USE	FOR POWER-OFF RETENTIVE USE	FOR SPECIAL USE
XC1	M	M000~M199	M200~M319	M8000~M8079
				M8120~M8139
				M8170~M8172
				M8238~M8242
				M8350~M8370

SERIES	NAME	RANGE		
		FOR COMMON USE	FOR POWER-OFF RETENTIVE USE	FOR SPECIAL USE
XC2	M	M000~M2999	M3000~M7999	M8000~M8767

SERIES	NAME	RANGE		
		FOR COMMON USE	FOR POWER-OFF RETENTIVE USE	FOR SPECIAL USE

XC3	M	M000~M2999	M3000~M7999	M8000~M8767

SERIES	NAME	RANGE		
		FOR COMMON USE	FOR POWER-OFF RETENTIVE USE	FOR SPECIAL USE
XC5	M	M000~M3999	M4000~M7999	M8000~M8767

SERIES	NAME	RANGE		
		FOR COMMON USE	FOR POWER-OFF RETENTIVE USE	FOR SPECIAL USE
XCM	M	M000~M2999	M3000~M7999	M8000~M8767

Function

In PLC, auxiliary relays M are used frequently. This type of relay's coil is same with the output relay. They are driven by soft components in PLCs;
auxiliary relays M have countless normally ON/OFF contactors. They can be used freely, but this type of contactors can't drive external loads.

- For common use
  - This type of auxiliary relays can be used only as normal auxiliary relays. i.e. if power supply suddenly stops during running, the relays will disconnect.
  - Common usage relays can't be used for power off retentive, but the zone can be modified;

- For Power Off Retentive Use
  - The auxiliary relays for power off retentive usage, if power is lost to the PLC, the ON/OFF satus is retained;
  - Power off retentive zone can be modified by the user;
  - Power off retentive relays are usually used to retain memory of the status before power is lost, when power is restored to the PLC, the current status will resume;

- For Special Usage
  - Special relays refer some relays which are defined with special meanings or functions, start from M8000.
  - There are two types of usages for special relays, one type is used to drive the coil, the other type is used to the specified execution;
    E.g.: M8002 is the initial pulse, activates only at the moment of start
    M8033 is "all output disabled"
  - Special auxiliary relays can't be used as a normal relay M;

2-6 Status Relay (S)

Address List

XC Series PLCs' status relays S are addressed in decimal form; each subfamily's ID are listed below:

SERIES	NAME	RANGE	
		FOR COMMON USE	FOR POWER-OFF RETENTIVE USE
XC1	S	S000~S031	-

SERIES	NAME	RANGE	
		FOR COMMON USE	FOR POWER-OFF RETENTIVE USE
XC2	S	S000~S511	S512~S1023

SERIES	NAME	RANGE	
		FOR COMMON USE	FOR POWER-OFF RETENTIVE USE
		S000~S511	S512~S1023

SERIES	NAME	RANGE	
		FOR COMMON USE	FOR POWER-OFF RETENTIVE USE
XC5	S	S000~S511	S512~S1023

SERIES	NAME	RANGE	
		FOR COMMON USE	FOR POWER-OFF RETENTIVE USE
XCM	S	S000~S511	S512~S1023

Function

Status relays are very import in ladder programming; usually use them with instruction "STL". In the form on flow, this can make the program's structure much clearer and easy to modify;

- For common use
  If the PLC loses power, this type of relay will revert to OFF status;
- For Power Off Retentive Use
  - The auxiliary relays for power off retentive usage, if power is lost to the PLC, the ON/OFF satus is retained;
  - Power off retentive zone can be modified by the user;
- The status relays also have countless “normally ON/OFF” contactors. So users can use them freely in the program;

2-7 Timer (T)

Address List

XC Series PLCs’ timers T are addressed in decimal form; each subfamily’s ID are listed below:

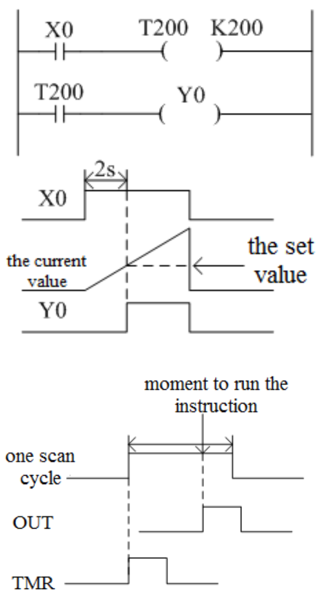
SERIES	NAME	RANGE	
		FOR COMMON USE	POINTS
XC1	T	T0~T23: 100ms not accumulation	80
		T100~T115: 100ms accumulation	
		T200~T223: 10ms not accumulation	
		T300~T307: 10ms accumulation	
		T400~T403: 1ms not accumulation	
		T500~T503: 1ms accumulation	
XC2 XC3 XC5 XCM	T	T0~T99: 100ms not accumulation	640
		T100~T199: 100ms accumulation	
		T200~T299: 10ms not accumulation	
		T300~T399: 10ms accumulation	
		T400~T499: 1ms not accumulation	
		T500~T599: 1ms accumulation	
		T600~T639: 1ms with precise time	

Function

The timers accumulate the 1ms, 10ms, 10ms clock pulse, the output contactor activates when the accumulation reaches the set value;

We use OUT or TMR instruction to time for the **normal** timers. We use constant (K) to set the value, or use data register (D) to indirect point the set value;

Normal Type



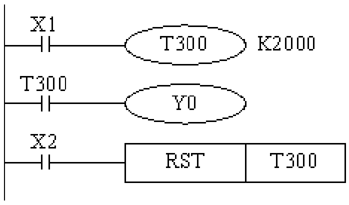
- If X0 is ON, then T200 accumulate 10ms clock pulse based on the current value; when the accumulation value reaches the set value K200, the timer's output contact activates. I.e. the output contact activates 2s later. If X0 breaks, the timer resets, the output contact resets;

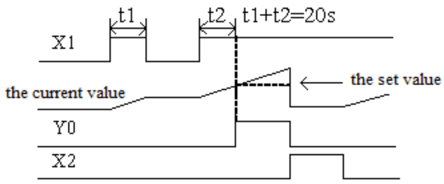
- Both OUT and TMR can realize the time function. But if use OUT, the start time is 0; if use TMR, the start time is 1 scan cycle

Accumulation Type

- If X001 is ON, then T300 accumulate 10ms clock pulse based on the current value; when the accumulation value reaches the set value K2000, the timer's output contact activates. I.e. the output contact activates 2s later.

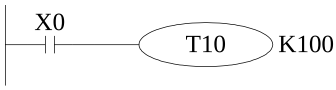
Even if X0 breaks, the timer will continue to accumulate on re-starting. The accumulation time is 20ms;



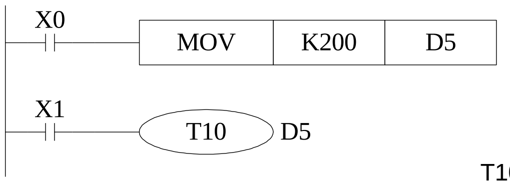


Specify the set value

《Constant (K)》



《Register (D)》



Write the indirect data register the contents of the data memory indirect pre-written program or through the switch input values.

In keeping with the register specified as a power outage, please pay attention to the battery voltage, if less than the value set will result in an unstable situation.

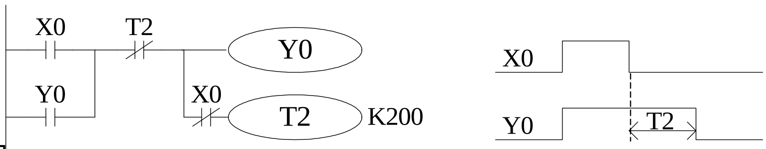
T10 is the timer with 100ms as the unit. Specify 100 as the constant, then 0.1s*100=10s timer works;

Timer Value

Timer T0~T599 is 16 bits linear increment mode (0~K32,767), when the timer's value reaches the max value K32767, it stops timing. The timer's status keeps still;

(Output Delay off timer)

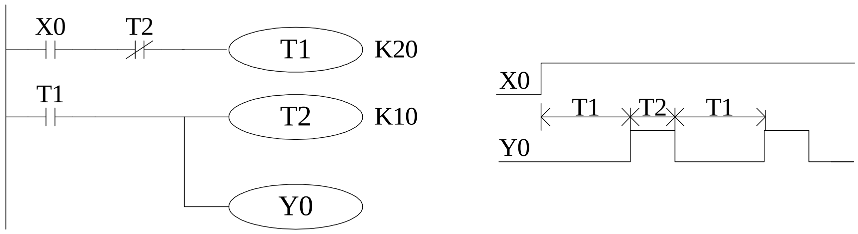
Action Example



Number List

X000 is ON, the output Y000;

When the X000 by the ON → OFF, it will delay T2 (20 seconds) time, the output Y000 was disconnected. (Flicker)



2-8 Counter (C)

Counter

XC Series PLCs - all decimal counter C to be addressed, for series of numbers see the table below:

SERIES	NAME	RANGE	
		FOR COMMON USE	POINTS
XC1	C	C0~C23: 16 bits forward counter	48
		C300~C315: 32 bits forward/backward counter	
		C600~C603: single-phase HSC	
		C620~C621	
		C630~C631	
XC2 XC3	C	C0~C299: 16 bits forward counter	640
		C300~C599: 32 bits forward/backward counter	
		C600~C619: single-phase HSC	
		C620~C629: double-phase HSC	
		C630~C639: AB phase HSC	

The number of counters on the following principles:

TYPE	DESCRIPTION
16 bits forward counter	C0~C299
32 bits forward/backward counter	C300~C599 (C300,C302...C598)(each occupies 2 counters number) the number should be even
HSC (High Speed Counter)	C600~C634(C600,C602...C634)((each occupies 2 counters number) the number should be even

※1：On high-speed counter usage, see Chapter 5.

Function

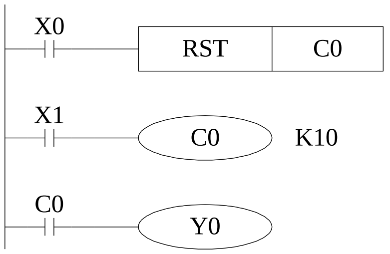
16-bit counter and 32-bit counter is characterized as follows:

Items	16 bits counter	32 bits counter
Count direction	Positive	Positive/negative
The set value	1~32,767	-2,147,483,648~+2,147,483,647
The assigned set value	Constant K or data register	Same as the left, but data register must be in a couple
Changing of the current value	Change after positive count	Change after positive count (Loop counter)
Output contact	Hold the action after positive count	Hold the action after positive count, reset if negative count
Reset activates	When executing RST command, counter's current value is 0, output contacts recover	
The current value register	16 bits	32 bits

The assignment of common use counters and power off retentive counters can me changed via FD parameters from peripheral devices;

Sixteen counter for general use \ l

16 bits binary increment counters, the valid value is K1~K32,767 (decimal type constant). The set value K0 and K1 has the same meaning. i.e. the output contact works on the first count starts



If you cut off the power programmable controller, the general count of the counter is cleared, and the latched counter can be used to store the count value before the power outage, so the last time the counter value according to the cumulative count.

- X001 count input C0 of each drive coil once the counter current value plus 1, the coil in the implementation of the tenth command, the output contact action. Enter the X001 again after the counter movement, counter current value will continue to add 1.

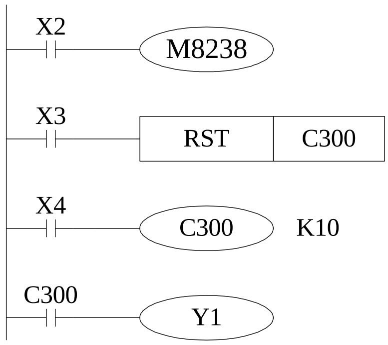
- If the reset input X000 is ON, the RST instruction is executed, the counter's current value is 0, reset input contact.

- Counter set value, in addition to the constant K set, but also by the data register number specified. For example, specify the D10, if the contents of D10 to 123, then set the K123 with the same time.

- In a MOV instruction to set the value of such data is written above the current value register, then the next input, the output coil connected to the current value into a register set value.

Thirty-two counter for general use \ Latched

32-bit binary up / down counter set value range for the +2,147,483,648 ~ -2,147,483,647 (decimal constant). The use of special auxiliary relay M8238 specified by the count of all 32-bit up / down counter (C300 ~ C498) direction.



- If the X2 driver M8238, was counting down; was not driven by the count.

- According to constant K D of the content or data register, setting the value is positive. The even number data register as a pair, as 32-bit data processing. Thus, when the designated D0, D1 and D0 two 32-bit settings as a treatment. C300 X004 driver using the input coil count when the up / down counting.

- If the reset input X3 is ON, the RST instruction is executed, the current value of the counter becomes 0, the output contact is reset.

- Use for Latched counter, the counter's current value, the output contacts reset state action and latched.

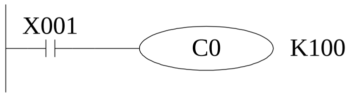
- 32-bit counter can also be used as a 32-bit data register.

Settings

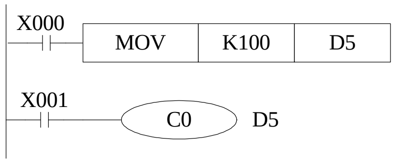
Count of the specified 16-bit and 32 bits is divided into two cases discussed.

➢ 16-bit counter

"Constant specified (K)"

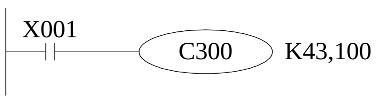


"Indirect designated (D)

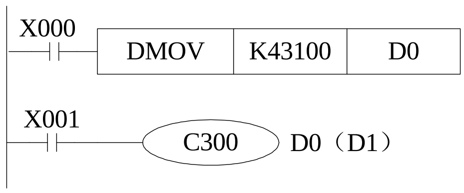


➢ 32-bit counter

"Constant specified (K)"



"Indirect designated (D)



Number List

~ C299 counting mode is 16-bit linear increment mode (0 ~ K32, 767), when ches the maximum count K32, 767 will stop the clock, the counter remains unchanged.

Counter C300 ~ C599 counting mode is 32-bit linear add / drop mode (-2,147,483,648 +2,147,483,647), when the counter reaches its maximum count value increment K2, 147,483,647 will become K-2, 147,483,648, when the counter counts down to minimum K-2, 147,483,648 will become K2, 147,483,647, the state of the counter with the count should change.

2-9 Data Register (D)

XC Series PLCs - all data register D to be addressed in decimal, for series of numbers see the table below:

Structure

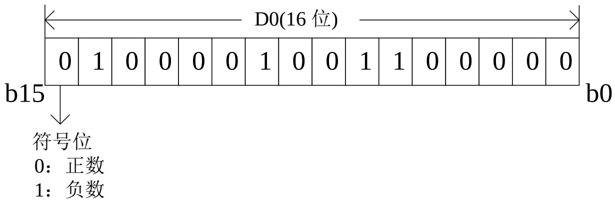
SERIES	NAME	RANGE			
		FOR COMMON USE	FOR POWER OFF RETENTIVE USE	FOR SPECIAL USE	
		D0~D99	D100~D149	D8000~D8029 D8060~D8079 D8120~D8179 D8240~D8249 D8306~D8313 D8460~D8469	138
XC2	D	D0~D999	D4000~D4999	D8000~D8511 D8630~D8729	612
XC3 XC5	D	D0~D3999	D4000~D7999	D8000~D9023	1024
XCM	D	D0~D2999	D3000~D4999	D8000~D9023	1024

Count

Data register is used to store data devices, including 16-bit (MSB is sign bit), 32 (a combination of two data registers, the MSB is sign bit) of two types.

16-bit data register's value is within the range of -32,768 to +32,767

Sixteen

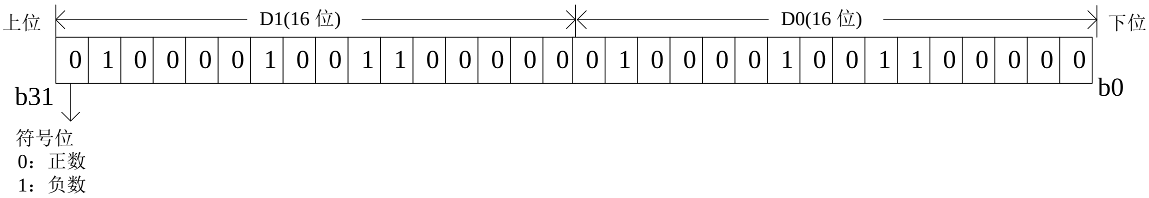


Read and write data register values commonly used application instructions. In addition, through other devices, such as man-machine interface to the PLC to write or read values.

Function

Thirty-two

The data from the two adjacent 32-bit data registers (high word in the post, the low word first, as D1D0 composition, D0 for the next bit, D1 is upper). Processing range is -2,147,483,648 to 2,147,483,647 values.



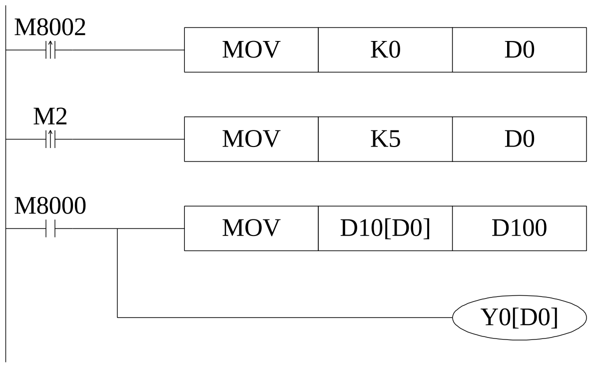
In the specified 32-bit register, if specified low as D0, the default of its high for the subsequent D1. Low can be odd or even any of the device to specify, but for the convenience, we recommend the use of even lower device number.

- **General Use**
  - ➢ When the data register to write successfully, just not re-write, then the data in the register will remain unchanged.
  - ➢ When the PLC goes from RUN to STOP or STOP to RUN, all data will be cleared.
- **Latched**
  - ➢ Latched area of data registers in the PLC from RUN to STOP or power failure, the data remains unchanged.
  - ➢ Latched area range, can be set by the user.

- **Special Use**
  - Special register is used to write with the specific purpose of data, or specific content is written by the system data.
  - Some special registers in the data, the PLC is powered on, is initialized.

- **As the offset (indirect specify)**
  - D data register can be used as an offset the device, making the device easier to use and easy to control.
  - Format: Dn [Dm], Xn [Dm], Yn [Dm], Mn [Dm] and so on.
  - Bit device composed of the word offset: DXn [Dm] said DX [n + Dm].
  - Device with offset, the offset is only available device D said.

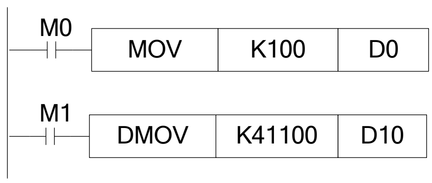
Example Action



The above example, when D0 = 0, the point D100 = D10, Y0 is ON;
When the M2 the OFF → ON,, D0 = 5, then D100 = D15, Y5 is ON.
Which D10 [D0] = D [10 + D0], Y0 [D0] = Y [0 + D0].

Data register D can handle a variety of data, the data register can be achieved through a variety of control.

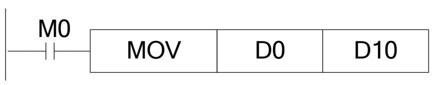
- **Data Storage**



M0 is turned on, write to the D0 16-bit, decimal number 100.

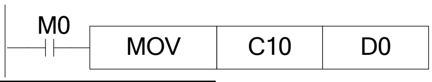
M1 is turned on, to D11D10 write 32-bit decimal number 41100. As the value of 41100 is 32 bits (over 32,767), and therefore store data, although designated as D10, but D11 is also automatically occupied.

- **Data Transfer**



M0 is switched on, the D0 of the data transfer to the D10.

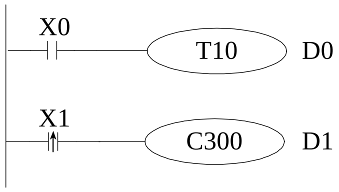
- **Read Timer or Counter**



M0 is switched on, the counter current value of C10 in the D0 in reading.

Data Processing

or Counter Set Value



X0 is switched on, T10 start time, regular time determined by the value in D0.

X1 is switched on every time, C300 starts counting, the count is determined by the D1.

2-10 Constant (K, H)

XC Series programmable controllers can be utilized for different uses and purposes, they use of five types of number system, each role and functions are as follows:

- **10 decimal (DEC: DECIMAL NUMBER)**
  - timer and counter set value (K constant)
  - Auxiliary relay (M), timer (T), counter (C), state (S) such number (device number)
  - Application of the instruction operands specifying the values and command action (K constant)

- **16 Hexadecimal (HEX: HEXADECIMAL NUMBER)**
  - and 10 hexadecimal numbers, as used to specify the application of the instruction operands and instruction moves the value (H constant)

- **2 binary number (BIN: BINARY NUMBER)**
  - As mentioned earlier, to decimal or hexadecimal number for the timer, counter values or data register specified in its internal programmable control, these figures are the number of binary processing. Moreover, in the external device monitoring, these devices will be automatically converted to decimal (which can also switch to hexadecimal).

- **8 binary numbers (OCT: OCTAL NUMBER)XC**
  - Series programmable controller input relay, output relay device number to octal values to assign, therefore, can be [0-7,10-17,. . . 70-77,100-107] into the position.

Representation

: BINARY CODE DECIMAL)BCD

decimal number you from 0 to 9 numerical method. The processing of each bit is easy, therefore, can be used for BCD output switch or the shape of seven segment digital display controls and so on.

- **Other values (floating point)**
  - XC programmable controller can be precision floating point functions.
    Binary floating-point floating-point operations, while monitoring the implementation of decimal floating-point values.

Value of the PLC program processing, you must use a constant K, H. Generally used to refer to decimal K, H refer to the hexadecimal number, but the PLC input and output relays with octal numbers.

- **Constant K**
  - K is the symbol that a decimal integer, such as K10, expressed in decimal 10. It is used for the specified timer, counter settings, and application instructions and number of operations.

- **Constant H**
  - H is the hex number of symbols, such as H10, is the hex number 10. Mainly used to specify the application instruction operand values.

2-11 Program Principle

- **Tag P、I**

  Tag P、I are used in branch division and interruption.

  Tag for branch (P) is used in condition jump or subroutine’s jump target;

  Tag for interruption (I) is used to specify the e input interruption, time interruption;

  The tags P, I are both in decimal form, each coding principle is listed below:

SERIES	NAME	RANGE
XC1、XC2、XC3、XC5、XCM	P	P0~P9999

<table>
<tr><th rowspan="3">SERIES</th><th rowspan="3">NAME</th><th colspan="4">RANGE</th></tr>
<tr><th colspan="3">FOR EXTERNAL INTERRUPTION</th><th rowspan="2">For time interruption</th></tr>
<tr><th>Input terminals</th><th>Rising edge interruption</th><th>Falling edge interruption</th></tr>
<tr><td rowspan="3">XC2</td><td rowspan="3">I</td><td>X2</td><td>I0000</td><td>I0001</td><td rowspan="3">There are 10 channels time interruption, the represent method is: I40**~I49**. (“**” represents interruption time, the unit is mm)</td></tr>
<tr><td>X5</td><td>I0100</td><td>I0101</td></tr>
<tr><td>X10</td><td>I0200</td><td>I0201</td></tr>
</table>

<table>
<tr><th rowspan="3">SERIES</th><th rowspan="3">NAME</th><th rowspan="3">I/O</th><th colspan="4">RANGE</th></tr>
<tr><th colspan="3">FOR EXTERNAL INTERRUPTION</th><th rowspan="2">For time interruption</th></tr>
<tr><th>Input terminals</th><th>Rising edge interruption</th><th>Falling edge interruption</th></tr>
</table>

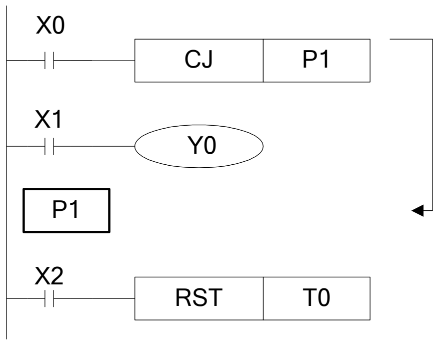
XC3	I	14	X7	I0000	I0001	There are 10 channels time interruption, the represent method is: I40**~I49**. ("**" represents interruption time, the unit is mm)
		24 32	X2	I0000	I0001	
			X5	I0100	I0101	
			X10	I0200	I0201	
		19 48 60	X10	I0000	I0001	
			X7	I0100	I0101	
			X6	I0200	I0201	

SERIES	NAME	I/O	RANGE			
			FOR EXTERNAL INTERRUPTION			For time interruption
			Input terminals	Rising edge interruption	Falling edge interruption	
XC5	I	24 32	X2	I0000	I0001	There are 10 channels time interruption, the represent method is: I40**~I49**. ("**" represents interruption time, the unit is mm)
			X5	I0100	I0101	
			X10	I0200	I0201	
			X11	I0300	I0301	
			X12	I0400	I0401	
		48 60	X2	I0000	I0001	
			X5	I0100	I0101	
			X10	I0200	I0201	

SERIES	NAME	I/O	RANGE			
			FOR EXTERNAL INTERRUPTION			For time interruption
			Input terminals	Rising edge interruption	Falling edge interruption	
XCM	I	24 32	X2	I0000	I0001	There are 10 channels time interruption, the represent method is: I40**~I49**. ("**" represents interruption time, the unit is mm)
			X5	I0100	I0101	
			X10	I0200	I0201	
			X11	I0300	I0301	
			X12	I0400	I0401	

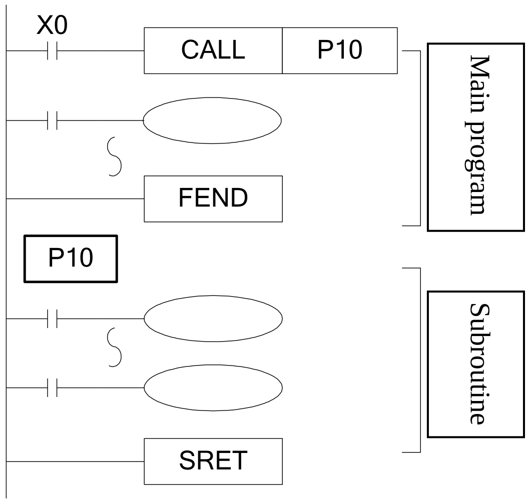
Tag P is usually used in flow, it is used with CJ (condition jump), CALL (subroutine call) etc.

- Condition Jump CJ



If coil X0 gets ON, jump to the step behind tag P1;
If the coil X0 is not ON, do not execute jump action, but run with the original program;

- Call the subroutine (CALL)



If X0 becomes ON, jump to the subroutine from the main program;
If the coil is not ON, run with the original program;

After executing the subroutine, return to the main program;

Tag I

Tag I is usually used in interruption, including external interruption, time interruption etc. use with IRET (interruption return), EI (enable interruption), DI (disable interruption);

- External interruption
  - ➢ Accepts input signal from the special input terminals, not effected by the scan cycle. Activates the input signal, executes the interruption subroutine.
  - ➢ With external interruption, PLC can dispose the signal shorter than scan cycle; so it can be used as essential priority disposal in sequence control, or used in short time pulse control.
- Time interruption
  - ➢ Execute the interruption subroutine at each specified interruption loop time. Use this interruption in the control which requires it to be different with PLC's operation cycle.

- Action order of input/output relays and response delay

- Input disposal
  Before PLC executing the program, read all the input terminal's ON/OFF status of PLC to the image area. In the process of executing the program, even the input changed, the content in the input image area will not change. However, in the input disposal of next scan cycle, read out the change.

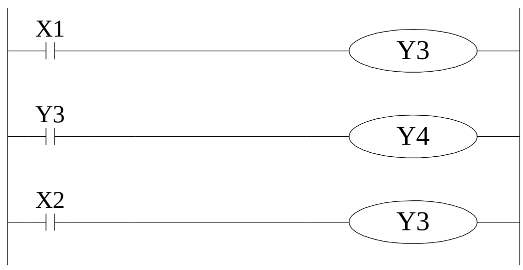
- Output disposal
  Once finished executing all the instructions, transfer the ON/OFF status of output Y image area to the output lock memory area. This will be the actual output of the PLC. The contacts used for the PLC's exterior output will act according to the device's response delay time.

When using this input/output format in a batch, the drive time and operation cycle of input filter and output device will also appear as per the response delay.

- **Not accept narrow input pulse signal**

PLC's input ON/OFF time should be longer than its loop time. E.g. if input filter's response delay 10ms, loop time is 10ms，then ON/OFF time needs 20 ms separately. So, up to 1，000/(20+20)=25Hz input pulse can't be disposed. But, this condition could be improved when use PLC's special function and applied instructions.

- **Dual output（Dual coils）action**



When executing dual output (use dual coil), the back side act in prior.

As shown in the left map, please consider the things of using the same coil Y003 at many positions:
E.g. X001=ON，X002=OFF

At first, X001 is ON, its image area is ON, output Y004 is also ON.

But, as input X002 is OFF, the image area of Y003 is OFF.

So, the actual output is: Y003=OFF, Y004= ON.

3 Basic Program Instructions

In this chapter, we give the basic instructions and their functions.

3-13． [OUT], [RST] (Aim at counter device)

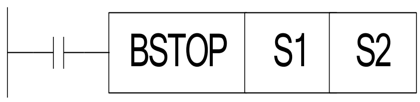
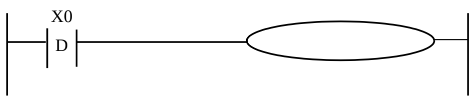
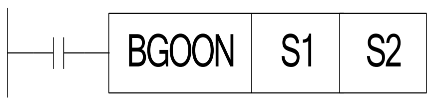
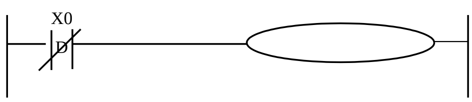
3-14． [NOP], [END]

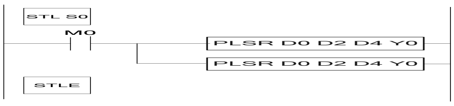
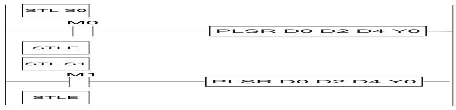
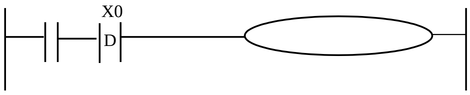
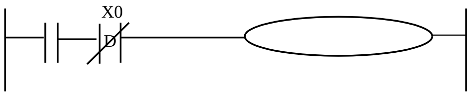
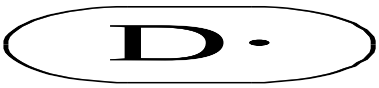
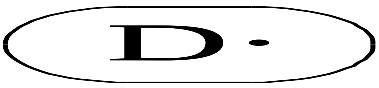
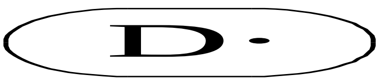
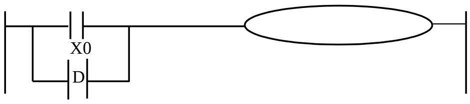
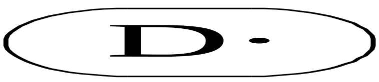
3-15． [GROUP], [GROUPE]

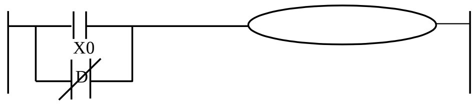
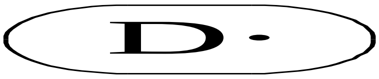
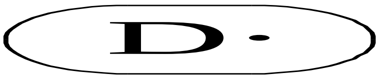
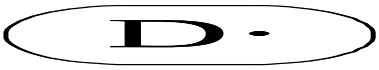
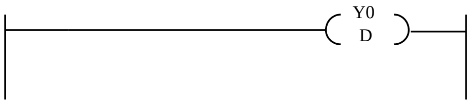
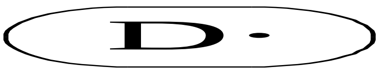
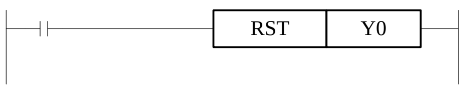
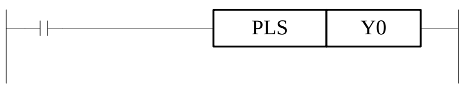
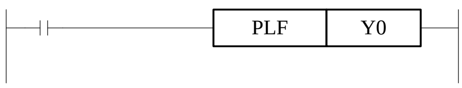
3-16． Programming Notes

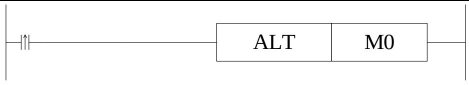
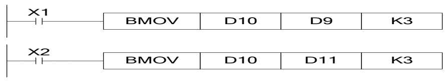
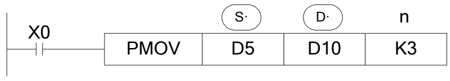
3-1 Basic Instructions List

All XC1, XC2, XC3, XC5, XCM series support the instructions below:

Mnemonic	Function	Format and Device	**Chapter**
LD (LoaD)	Initial logical operation contact type NO (normally open)	 X、Y、M、S、T、C、Dn.m、FDn.m	3-2
LDD (LoaD Directly)	Read the status from the contact directly	 X	3-6
LDI (LoaD Inverse)	Initial logical operation contact type NC (normally closed)	 X、Y、M、S、T、C、Dn.m、FDn.m	3-2
LDDI	Read the normally closed contact directly	 X	3-6

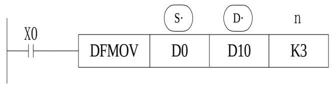
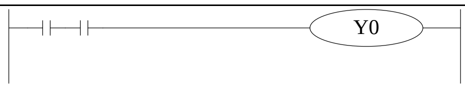
LDP (LoaD Pulse)	Initial logical operation-Rising edge pulse	 X、Y、M、S、T、C、Dn.m、FDn.m	3-5
LDF (LoaD Falling Pulse)	Initial logical operation-Falling /trailing edge pulse	 X、Y、M、S、T、C、Dn.m、FDn.m	3-5
AND (AND)	Serial connection of NO (normally open) contacts	 X、Y、M、S、T、C、Dn.m、FDn.m	3-3
ANDD	Read the status from the contact directly	 X	3-6
ANI (AND Inverse)	Serial connection of NC (normally closed) contacts	 X、Y、M、S、T、C、Dn.m、FDn.m	3-3
ANDDI	Read the normally closed contact directly	 X	3-6
ANDP (AND Pulse)	Serial connection of rising edge pulse	 X、Y、M、S、T、C、Dn.m、FDn.m	3-5
ANDF (AND Falling pulse)	Serial connection of falling/trailing edge pulse	 X、Y、M、S、T、C、Dn.m、FDn.m	3-5
OR (OR)	Parallel connection of NO (normally open) contacts	 X、Y、M、S、T、C、Dn.m、FDn.m	3-4
ORD	Read the status from the contact directly	 X	3-6
ORI (OR Inverse)	Parallel connection of NC (normally closed) contacts	 X、Y、M、S、T、C、Dn.m、FDn.m	3-4

ORDI	Read the normally closed contact directly	 X	3-6
ORP (OR Pulse)	Parallel connection of rising edge pulse	 X、Y、M、S、T、C、Dn.m、FDn.m	3-5
ORF (OR Falling pulse)	Parallel connection of falling/trailing edge pulse	 X、Y、M、S、T、C、Dn.m、FDn.m	3-5
ANB (ANd Block)	Serial connection of multiply parallel circuits	 None	3-8
ORB (OR Block)	Parallel connection of multiply parallel circuits	 None	3-7
OUT (OUT)	Final logic operation type coil drive	 Y、M、S、T、C、Dn.m	3-2
OUTD	Output to the contact directly	 Y	3-6
SET (SET)	Set a bit device permanently ON	 Y、M、S、T、C、Dn.m	3-12
RST (ReSeT)	Reset a bit device permanently OFF	 Y、M、S、T、C、Dn.m	3-12
PLS (PuLSe)	Rising edge pulse	 X、Y、M、S、T、C、Dn.m	3-11
PLF (PuLse Falling)	Falling/trailing edge pulse	 X、Y、M、S、T、C、Dn.m	3-11

MCS (New bus line start)	Connect the public serial contacts	 None	3-9
MCR (Bus line return)	Clear the public serial contacts	 None	3-9
ALT (Alternate state)	The status of the assigned device is inverted on every operation of the instruction	 X、Y、M、S、T、C、Dn.m	3-10
END (END)	Force the current program scan to end	 None	3-14
GROUP	Group	 None	3-15
GROUPE	Group End	 None	3-15
TMR	Time		2-7

3-2 [LD] , [LDI] , [OUT]

Mnemonic and Function

Mnemonic	Function	Format and Operands
LD (LoaD)	Initial logic operation contact type NO (Normally Open)	 Operands: X、Y、M、S、T、C、Dn.m、FDn.m
LDI (LoaD Inverse)	Initial logic operation contact type NC (Normally Closed)	 Devices：X、Y、M、S、T、C、Dn.m、FDn.m
OUT (OUT)	Final logic operation type drive coil	 Operands: X、Y、M、S、T、C、Dn.m

Statements

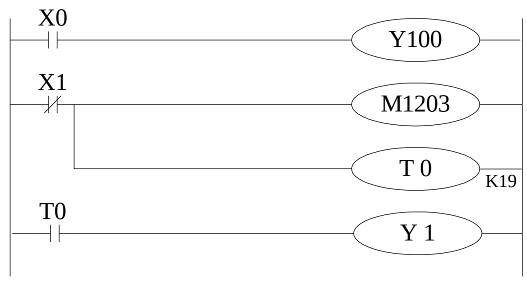
- Connect the LD and LDI instructions directly to the left bus bar, or use them to define a new block of program when using ANB instruction.
- OUT instruction is the coil drive instruction for the output relays, auxiliary relays、status, timers, counters. But this instruction can't be used for the input relays
- Can not sequentially use parallel OUT command for many times.
- For the timer's time coil or counter's count coil, after using OUT instruction, set constant K is necessary.

- For the constant K's setting range, actual timer constant, program's step relative to OUT instruction (include the setting value), See table below:

Timer, Counter	Setting Range of constant K	The actual setting value
1ms Timer	1～32,767	0.001～32.767 sec
10ms Timer		0.01～327.67 sec
100ms Timer		0.1～3276.7 sec
16 bits counter	1～32,767	Same as the left
32 bits counter	1～2,147,483,647	Same as the left

Mnemonic	Function	Format and Operands

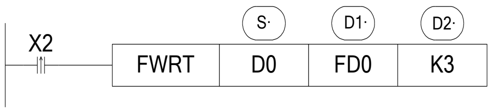
Program




```
LD    X0
OUT   Y100
LDI   X1
OUT   M1203
OUT   T0     K19
LD    T0
OUT   Y1
```

3-3 [AND] , [ANI]

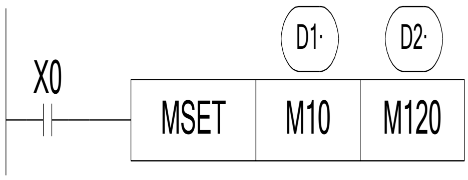
Mnemonic and Function

AND (AND)	Serial connection of NO (Normally Open) contacts	 Operands: X、Y、M、S、T、C、Dn.m、FDn.m
ANI (ANd Inverse)	Serial connection of NC (Normally Closed) contacts	 Operands: X、Y、M、S、T、C、Dn.m、FDn.m

Statements

- Use the AND and the ANI instruction for serial connection of contacts. As many contacts as required can be connected in series. They can be used for many times.
- The output processing to a coil, through writing the initial OUT instruction is called a “follow-on” output (For an example see the program below: OUT M2 and OUT Y003). Follow-on outputs are permitted repeatedly as long as the output order is correct. There’s no limit for the serial connected contacts’ Nr. and follow-on outputs’ number.

Program




```
LD    X2
AND   M1
OUT   Y2
LD    Y2
ANI   X3
OUT   M2
AND   T1
OUT   Y3
```

3-4 [OR], [ORI]

Mnemonic and Function

Mnemonic	Function	Format and Operands

Mnemonic	Function	Format and Operands
OR (OR)	Parallel connection of NO (Normally Open) contacts	D2· Operands: X、Y、M、S、T、C、Dn.m、FDn.m
ORI	Parallel connection of ... contacts	Operands: X、Y、M、S、T、C、Dn.m、FDn.m

Statements

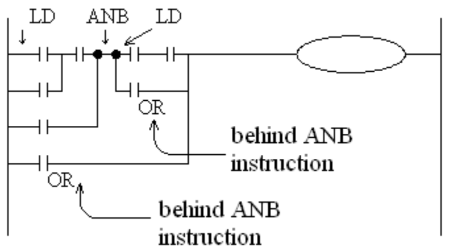
- Use the OR and ORI instructions for parallel connection of contacts. To connect a block that contains more than one contact connected in series to another circuit block in parallel, use an ORB instruction, which will be described later;
- OR and ORI start from the instruction's step, parallel connect with the LD and LDI instruction's step said before. There is no limit for the parallel connect times.

Program

D2·

LD	X5
OR	X6
OR	M11
OUT	Y6
LDI	Y6
AND	M4
OR	M12
ANI	X7
OR	M13
OUT	M100

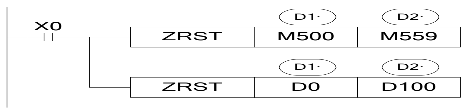
Relationship with ANB



The parallel connection with OR, ORI instructions should connect with LD, LDI instructions in principle. But behind the ANB instruction, it's still ok to add a LD or LDI instruction.

3-5 [LDP], [LDF], [ANDP], [ANDF], [ORP], [ORF]

Mnemonic and Function

LDP (LoaD Pulse)	Initial logical operation-Rising edge pulse	D1· Operands: X、Y、M、S、T、C、Dn.m、FDn.m
LDF (LoaD Falling pulse)	Initial logical operation Falling/trailing edge pulse	D2· Operands: X、Y、M、S、T、C、Dn.m、FDn.m
ANDP (AND Pulse)	Serial connection of Rising edge pulse	D1· Operands: X、Y、M、S、T、C、Dn.m、FDn.m
ANDF (AND Falling pulse)	Serial connection of Falling/trailing edge pulse	 Operands: X、Y、M、S、T、C、Dn.m、FDn.m
ORP (OR Pulse)	Parallel connection of Rising edge pulse	D2· Operands: X、Y、M、S、T、C、Dn.m、FDn.m
ORF (OR Falling pulse)	Parallel connection of Falling/trailing edge pulse	D1· Operands: X、Y、M、S、T、C、Dn.m、FDn.m

Statements

- LDP, ANDP, ORP are active for one program scan after the associated devices switch from OFF to ON.
- LDF, ANDF, ORF are active for one program scan after the associated devices switch from ON to OFF.

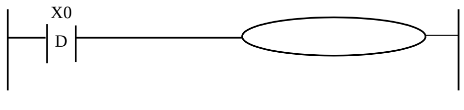
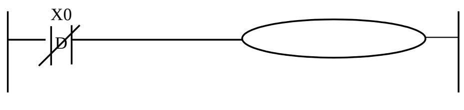
Program

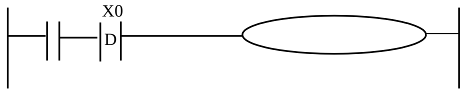
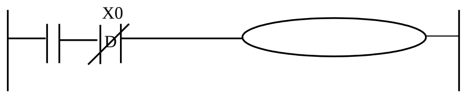
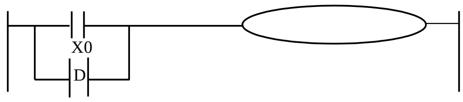
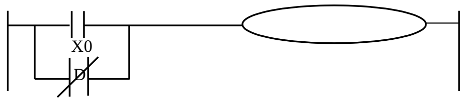
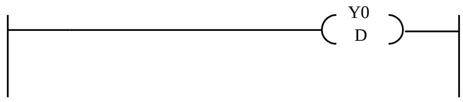
D2·

LDP	X5
ORP	X6
OUT	M13
LD	M8000
ANDP	X7

3-6 [LDD], [LDDI], [ANDD], [ANDDI], [ORD], [ORDI], [OUTD]

Mnemonic and Function

Mnemonic	Function	Format and Operands
LDD	Read the status from the contact directly	 Devices: X
LDDI	Read the normally closed contact directly	 Devices: X

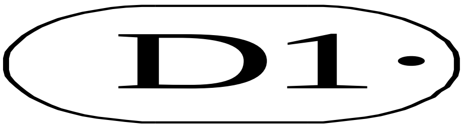
ANDD	Read the status from the contact directly	 Devices: X
ANDDI	Read the normally closed contact directly	 Devices: X
ORD	Read the status from the contact directly	 Devices: X
ORDI	Read the normally closed contact directly	 Devices: X
OUTD	Output to the contact directly	 Devices: Y

Statements

- The function of LDD, ANDD, ORD instructions are similar with LD, AND, OR;
- LDDI, ANDDI, ORDI instructions are similar with LDI, ANDI, ORI; but if the operand is X, the LDD, ANDD, ORD commands read the signal from the terminals directly, this is the only difference.
- OUTD and OUT are output instructions. But if OUTD is used, output immediately if the condition comes true, needn't wait the next scan cycle.

```
LDD     X0
LDDI    X2
ORD     X2
ANB
OUTD    Y0
```

Program



3-7 [ORB]

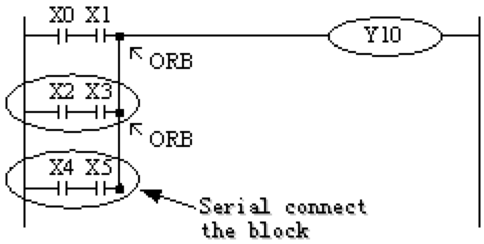
Mnemonic and Function

Mnemonic	Function	Format and Devices
ORB (OR Block)	Parallel connection of multiply parallel circuits	 Devices: none

Statements

- The serial connection with two or more contacts is called "serial block". If parallel connect the serial block, use LD, LDI at the branch start place, use ORB at the stop place;
- As the ANB instruction，an ORB instruction is an independent instruction and is not associated with any device number.
- There are no limitations to the number of parallel circuits when using an ORB instruction in the sequential processing configuration.

Program



Recommended good programming method：

```
LD    X0
AND   X1
LD    X2
AND   X3
ORB
LD    X4
AND   X5
ORB
OUT   Y10
```

Non-preferred batch programming method：

```
LD    X0
AND   X1
LD    X2
AND   X3
LD    X4
AND   X5
ORB
ORB
```

3-8 [ANB]

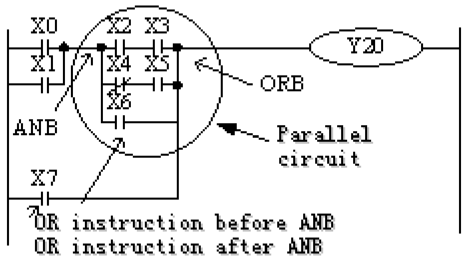
Mnemonic and Function

Mnemonic	Function	Format and Devices
ANB (And Block)	Serial connection of multiply parallel circuits	D2- Devices: none

Statements

(1) To declare the starting point of the circuit block, use a LD or LDI instruction. After completing the parallel circuit block, connect it to the preceding block in series using the ANB instruction.
(2) It is possible to use as many ANB instructions as necessary to connect a number of parallel circuit blocks to the preceding block in series.

Program



LD	X0	
OR	X1	
LD	X2	Start of a branch
AND	X3	
LDI	X4	
AND	X5	
ORB		End of a parallel circuit block
OR	X6	
ANB		
OR	X7	Serial connect with the preceding circuit
OUT	Y20	

3-9 [MCS], [MCR]

Mnemonic and Function

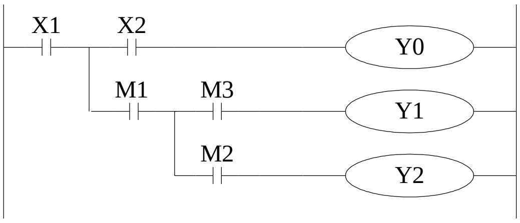
Mnemonic	Function	Format and Devices
MCS (Master control)	Denotes the start of a master control block	 Devices：None

MCR (Master control Reset)	Denotes the end of a master control block	 Devices：None

Statements

- After the execution of an MCS instruction, the bus line（LD, LDI）shifts to a point after the MCS instruction. An MCR instruction returns this to the original bus line.
- MCS, MCR instructions should use in pair.
- The bus line could be used nesting. Between the matched MCS, MCR instructions use matched MCS, MCR instructions. The nest level increase with the using of MCS instruction. The max nest level is 10. When executing MCR instruction, go back to the upper bus line.
- When use flow program, bus line management could only be used in the same flow. When end some flow, it must go back to the main bus line.

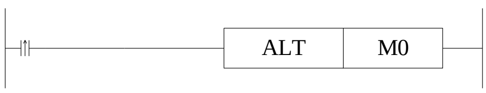
Program



LD	X1	
MCS		Bus line starts
LD	X2	
OUT	Y0	
LD	M1	
MCS		Bus line nest
LD	M3	
OUT	Y1	
LD	M2	
OUT	Y2	
MCR		
MCR		Bus line back

3-10 [ALT]

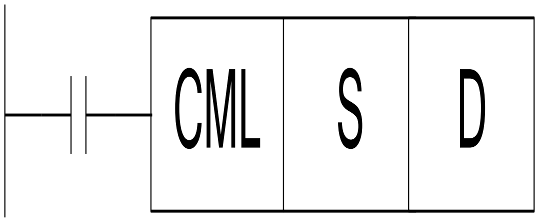
Mnemonic and Function

Mnemonic	Function	Format and Devices
ALT (Alternate status)	The status of the assigned devices inverted on every operation of the instruction	 Devices： Y、M、S、T、C、Dn.m

Statements

The status of the destination device is alternated on every operation of the ALT instruction.

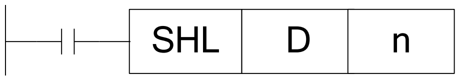
Program




```
LDP    M100
ALT    M0
LD     M0
OUT    Y0
LDI    M0
OUT    Y1
```

3-11 [PLS], [PLF]

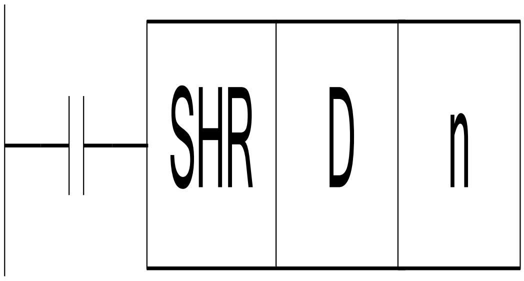
Mnemonic and Function

Mnemonic	Function	Format and Devices
PLS (Pulse)	Rising edge pulse	 Devices： Y、M、S、T、C、Dn.m
PLF (Pulse Falling)	Falling/trailing edge pulse	 Devices： Y、M、S、T、C、Dn.m

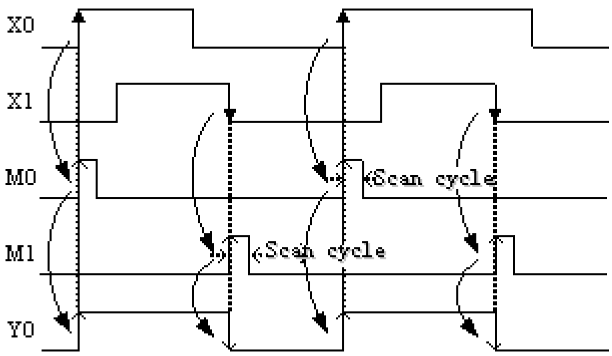
Statements

1. When a PLS instruction is executed, object devices Y and M operate for one operation cycle after the drive input signal has turned ON.
2. When a PLF instruction is executed, object devices Y and M operate for one operation cycle after the drive input signal has turned OFF.

Program




```
LD    X0
PLS   M0
LD    M0
SET   Y0
---------------------
LD    X1
PLF   M1
LD    M1
RST   Y0
```



3-12 [SET], [RST]

Mnemonic and Function

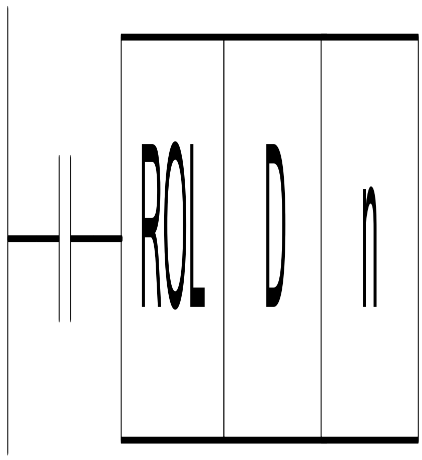
Mnemonic	Function	Format and Devices
SET（Set）	Set a bit device permanently ON	LSL D n Devices： Y、M、S、T、C、Dn.m
RST(Reset)	Reset a bit device permanently OFF	LSR D n Devices： Y、M、S、T、C、Dn.m

Statements

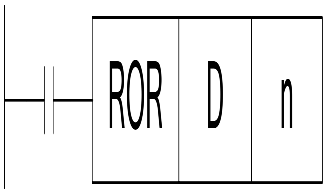
Turning ON X010 causes Y000 to turn ON. Y000 remains ON even after X010 turns OFF. Turning ON X011 causes Y000 to turn OFF. Y000 remains OFF even after X011 turns OFF. It's the same with M, S.

- SET and RST instructions can be used for the same device as many times as necessary. However, the last instruction activated determines the current status.
- It is also possible to use RST instruction to reset the current contents of timer, counter and contacts.
- When use SET, RST commands, avoid to use the same ID with OUT command.

Program

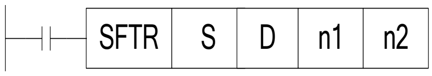


LD	X10	
SET	Y0	
LD	X11	
RST	Y0	
LD	X12	
SET	M50	
LD	X13	
RST	M50	
LD	X14	
SET	S0	
LD	X15	
RST	S0	
LD	X10	
OUT	T250	K10
LD	X17	
RST	T250	

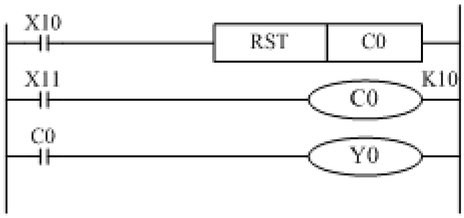


3-13 [OUT], [RST] for the counters

Mnemonic and Function

Mnemonic	Function	Format and Devices
OUT	Final logic operation type coil drive	 Device：K、D
RST	Reset a bit device permanently OFF	 Device：C

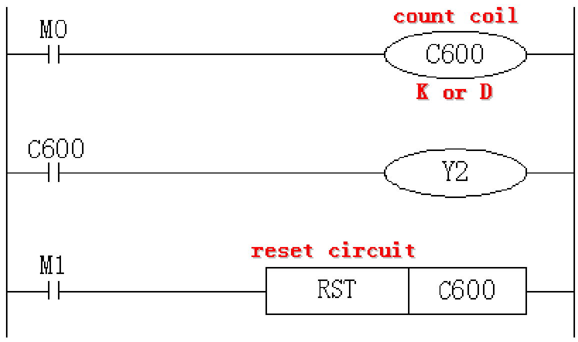
Programming of interior counter



Counter used for power cut retentive. Even when power is cut, hold the current value and output contact's action status and reset status.

C0 carries on increase count for the OFF→ON of X011. When the set value K10 is reached, output contact C0 activates. Afterwards, even X011 turns from OFF to ON, counter's current value will not change, output contact keep on activating. To clear this, let X010 be the activate status and reset the output contact. It's necessary to assign constant K or indirect data register's ID behind OUT instruction.

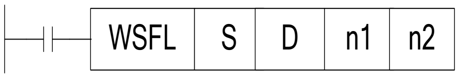
Programming of high speed



- In the preceding example, when M0 is ON, carry on positive count with OFF→ON of X0.
- Counter's current value increase, when it reaches the set value (K or D), the output contact is reset.
- When M1 is ON, counter's C600 output contact is reset, counter's current value turns to be 0.

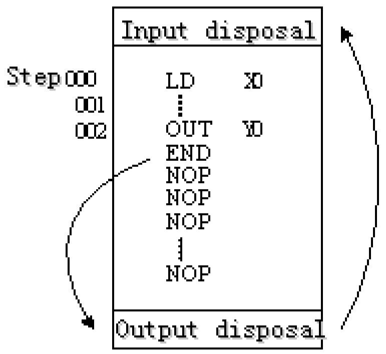
3-14 [END]

Mnemonic and Function

Mnemonic	Function	Format and Devices：None
END (END)	Force the current program scan to end	 Devices: None

Statements

PLC repeatedly performs input disposal, program executing and output disposal. If write END instruction at the end of the program, then the instructions behind END instruction won't be executed. If there's no END instruction in the program, the PLC executes the end step and



When executing END instruction, refresh monitor timer. (Check if scan cycle is a long timer.)

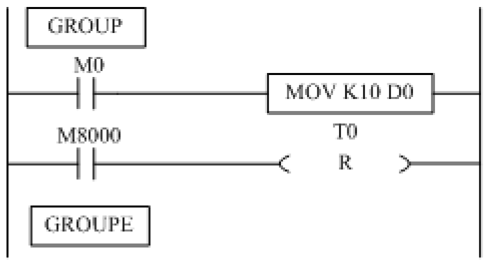
3-15 [GROUP], [GROUPE]

Mnemonic and Function

Mnemonic	Function	Format and Device
GROUP	GROUP	WSFR S D n1 n2 Devices: None
GROUPE	GROUP END	WTD S D Devices: None

Statements

- GROUP and GROUPE should used in pairs.
- GROUP and GROUPE don't have practical meaning, they are used to optimize the program structure. So, add or delete these instructions doesn't effect the program's running.
- The using method of GROUP and GROUPE is similar with flow instructions; enter GROUP instruction at the beginning of group part; enter GROUPE instruction at the end of group part.



Generally, GROUP and GROUPE instruction can be programmed according to the group's function. The programmed instructions can be FOLDED or UNFOLDED. To a redundant project, these two instructions are quite useful.

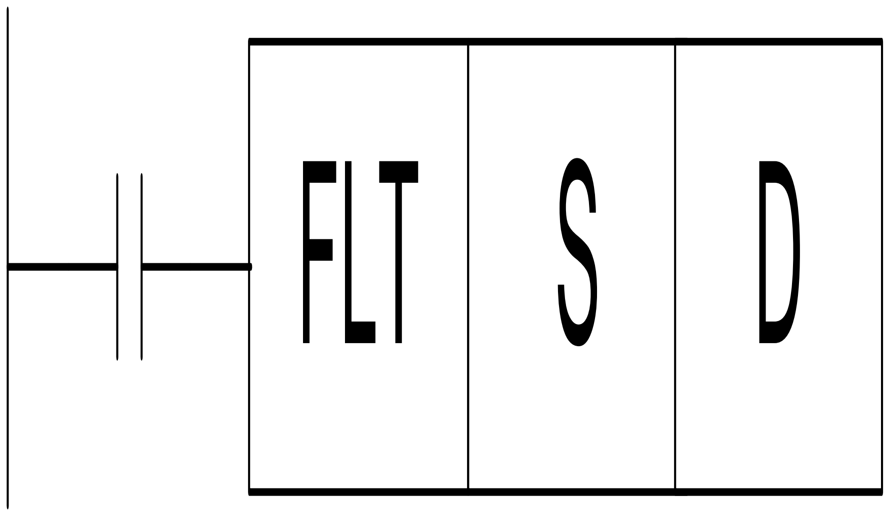
3-16 Programming Notes

1: Program's executing sequence

The program control flow is processed from【From top to bottom】and【From left to right】Sequencial control instructions also encode following this flow.

2: Calling outputs multiple times

See the below example on how to stop this occuring

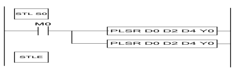
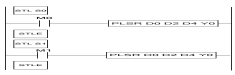


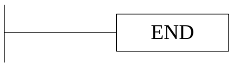
There are other methods. E.g. jump instructions or step ladder. However, when use step ladder, if the main program's output coil is programmed, then the disposal method is the same with dual coil, please note this.

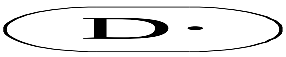
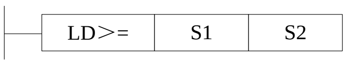
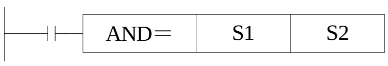
4 Applied Instructions

In this chapter, we describe the applied instruction's function of XC Series PLC.

4-1 Applied Instruction List

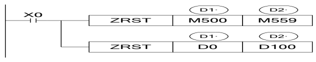
Mnemonic	Function	**Ladder chart**	**Chapter**
Program Flow			
CJ	Condition jump		4-3-1
CALL	Call subroutine		4-3-2
SRET	Subroutine return		4-3-2
STL	Flow start		4-3-3
STLE	Flow end		4-3-3
SET	Open the assigned flow, close the current flow		4-3-3

ST	Open the assigned flow, not close the current flow		4-3-3
FOR	Start a FOR-NEXT loop		4-3-4
NEXT	End of a FOR-NEXT loop		4-3-4
FEND	Main program END		4-3-5
END	Program END		4-3-5

Data Compare			
LD＝	LD activates if (S1) = (S2)		4-4-1
LD＞	LD activates if (S1) > (S2)		4-4-1
LD＜	LD activates if (S1) =< (S2)		4-4-1
LD＜＞	LD activates if（S1）≠（S2）		4-4-1
LD＜＝	LD activates if（S1）≤（S2）		4-4-1
LD＞＝	LD activates if（S1）≥（S2）		4-4-1
AND＝	AND activates if（S1）＝（S2）		4-4-2

AND＞	AND activates if（S1）＞（S2）	AND> S1 S2	4-4-2
AND＜	AND activates if（S1）＜（S2）	AND< S1 S2	4-4-2
AND＜＞	AND activates if（S1）≠（S2）	AND<> S1 S2	4-4-2
AND＜＝	AND activates if（S1）≤（S2）	AND<= S1 S2	4-4-2
AND＞＝	AND activates if（S1）≥（S2）	X1 BMOV D10 D9 K3 X2 BMOV D10 D11 K3	4-4-2
OR＝	OR activates if（S1）＝（S2）	OR= S1 S2	4-4-3
OR＞	OR activates if（S1）＞（S2）	X0 PMOV S· D5 D· D10 n K3	4-4-3
OR＜	OR activates if（S1）＜（S2）	OR＜ S1 S2	4-4-3
OR＜＞	OR activates if（S1）≠（S2）	OR＜＞ S1 S2	4-4-3
OR＜＝	OR activates if（S1）≤（S2）	X0 DFMOV S· D0 D· D10 n K3	4-4-3
OR＞＝	OR activates if（S1）≥（S2）	OR＞= S1 S2	4-4-3

Data Move			
CMP	Compare the data	CMP S1 S D	4-5-1
ZCP	Compare the data in certain area	X2 FWRT S· D0 D1· FD0 D2· K3	4-5-2
MOV	Move	MOV S D	4-5-3
BMOV	Block move	X0 MSET D1· M10 D2· M120	4-5-4
PMOV	Transfer the Data block	D2·	4-5-5
FMOV	Multi-points repeat move	D1·	4-5-6
FWRT	Flash ROM written	D2·	4-5-7

MSET	Zone set	D1·	4-5-8
ZRST	Zone reset	D2·	4-5-9
SWAP	Swap the high and low byte	D1·	4-5-10
XCH	Exchange two values		4-5-11

Data Operation			
ADD	Addition	D2·	4-6-1
SUB	Subtraction	D1·	4-6-2
MUL	Multiplication	D2·	4-6-3
DIV	Division	D1·	4-6-4
INC	Increment	D1·	4-6-5
DEC	Decrement	D2·	4-6-5
MEAN	Mean	D1·	4-6-6

WAND	Word And	WAND | S1 | S2 | D	4-6-7
WOR	Word OR	WOR | S1 | S2 | D	4-6-7
WXOR	Word exclusive OR	WXOR | S1 | S2 | D	4-6-7
CML	Compliment	CML | S | D	4-6-8
NEG	Negative	NEG | D	4-6-9

Data Shift			
SHL	Arithmetic Shift Left	SHL | D | n	4-7-1
SHR	Arithmetic Shift Right	SHR | D | n	4-7-1
LSL	Logic shift left	LSL | D | n	4-7-2
LSR	Logic shift right	LSR | D | n	4-7-2
ROL	Rotation shift left	ROL | D | n	4-7-3
ROR	Rotation shift right	ROR | D | n	4-7-3
SFTL	Bit shift left	SFTL | S | D | n1 | n2	4-7-4

SFTR	Bit shift right	SFTR S D n1 n2	4-7-5
WSFL	Word shift left	WSFL S D n1 n2	4-7-6
WSFR	Word shift right	WSFR S D n1 n2	4-7-7

Data Convert			
WTD	Single word integer converts to double word integer	WTD S D	4-8-1
FLT	16 bits integer converts to float point	FLT S D	4-8-2
DFLT	32 bits integer converts to float point	DFLT S D	4-8-2
FLTD	64 bits integer converts to float point	FLTD S D	4-8-2
INT	Float point converts to integer	INT S D	4-8-3
BIN	BCD converts to binary	BIN S D	4-8-4
BCD	Binary converts to BCD	BCD S D	4-8-5

ASCI	Hex. converts to ASCII	ASCI S D n	4-8-6
HEX	ASCII converts to Hex.	HEX S D n	4-8-7
DECO	Coding	DECO S D n	4-8-8
ENCO	High bit coding	ENCO S D n	4-8-9
ENCOL	Low bit coding	ENCOL S D n	4-8-10

Float Point Operation			
ECMP	Float compare	ECMP S1 S2 D	4-9-1
EZCP	Float Zone compare	EZCP S1 S2 D1 D2	4-9-2
EADD	Float Add	EADD S1 S2 D	4-9-3
ESUB	Float Subtract	ESUB S1 S2 D	4-9-4
EMUL	Float Multiplication	EMUL S1 S2 D	4-9-5
EDIV	Float division	EDIV S1 S2 D	4-9-6
ESQR	Float Square Root	ESQR S D	4-9-7

SIN	Sine	SIN S D	4-9-8
COS	Cosine	COS S D	4-9-9
TAN	Tangent	TAN S D	4-9-10
ASIN	Floating Sine	ASIN S D	4-9-11
ACOS	Floating Cosine	ACOS S D	4-9-12
ATAN	Floating Tangent	ATAN S D	4-9-13
Clock Operation			
TRD	Read RTC data	TRD D	4-10-1
TWR	Write RTC data	TWR D	4-10-2

4-2 Reading Method of Applied Instructions

In this manual, the applied instructions are described in the following manner:

1: Summary

ADDITION [ADD]			
16 bits	ADD	32 bits	DADD
Execution condition	Normally ON/OFF, Rising/Falling edge	Suitable Models	XC1.XC2.XC3.XC5.XCM
Hardware requirement	-	Software requirement	-

2: Operands

Operands	Function	Data Type
S1	Specify the augend data or register	16 bits/32 bits, BIN
S2	Specify the summand data or register	16 bits/32 bits, BIN

D	Specify the register to store the sum	16 bits/32 bits, BIN

3: .Suitable Soft Components

Word

operands	System									Constant	Module	
	D	FD	ED	TD	CD	DX	DY	DM	DS	K/H	ID	QD
S1	●	●		●	●	●	●	●	●	●		
S2	●	●		●	●	●	●	●	●	●		
D	●	●		●	●		●	●	●			

Bit

Operands	System					
	X	Y	M	S	T	C	Dn.m

Description

<16 bits instruction>

（D10）＋（D12）→（D14）

<32 bits instruction>

（D11D10）＋（D13D12）→（D15D14）

1. The data contained within the two source devices are combined and the total is stored in the specified destination device. Each data’s highest bit is the sign bit, 0 stands for positive, 1 stand for negative. All calculations are algebraic processed. (5+(-8)= -3).

2. If the result of a calculations is “0”, the “0’ flag acts. If the result exceeds 323,767(16 bits limit) or 2,147,483,648 (32 bits limit), the carry flag acts. (refer to the next page). If the result exceeds -323,768 (16 bits limit) or -2,147,483,648 (32 bits limit) , the borrow flag acts (Refer to the next page).

3. When carry on 32 bits operation, word device’s 16 bits are assigned, the device follow closely the preceding device’s ID will be the high bits. To avoid ID repetition, we recommend you assign device’s ID to be even ID.

4. The same device may be used a source and a destination. If this is the case then the result changes after every scan cycle. Please note this point.

Related Flag

Flag	Name	Function
M8020	Zero	ON：the calculate result is zero OFF：the calculate result is not zero
M8021	Borrow	ON：the calculate result is over 32767(16bits) or 2147483647(32bits) OFF：the calculate result is not over 32767(16bits) or 2147483647(32bits)
M8022	Carry	ON：the calculate result is over 32767(16bits) or 2147483647(32bits) OFF：the calculate result is not over 32767(16bits) or 2147483647(32bits)

Related Description

- The assignment of the data

  The data register of XC series PLC is a single word (16 bit) data register, single word data only engross one data register which is assigned by single word object instruction. The disposal bound is: Dec. –327,68~327,67, Hex. 0000~FFFF.

Single word object instruction　　　　D(NUM)

Instruction	D(NUM)	→	Object data

Double word（32 bit）engrosses two data register, it's composed by two consecutive data registers, the first one is assigned by double word object instruction.　The dispose bound is: Dec.　-214,748,364,8~214,748,364,7, Hex. 00000000~FFFFFFFF.

Double word object instruction　　　　D(NUM+1)　　D(NUM)

Instruction	D(NUM)	→	Object data	Object data

- The denote way of 32 bits instruction

  If an instruction can not only be 16 bits but also be 32 bits, then the denote method for 32 bits instruction is to add a "D" before 16 bits instruction.

  E.g：ADD　D0　D2　D4 denotes two 16 bits data adds；

  　　DADD　D10　D12　D14 denotes two 32 bits data adds

※1：Flag after executing the instruction. Instructions without the direct flag will not display.

※2： S· Source operand, its content won't change after executing the instruction.

※3： D· Destinate operand, its content changes with the execution of the instruction.

※4：Tell the instruction's basic action, using way, applied example, extend function, note items etc.

4-3 Program Flow Instructions

Mnemonic	Instruction's name	Chapter
CJ	Condition Jump	4-3-1
CALL	Call subroutine	4-3-2
SRET	Subroutine return	4-3-2
STL	Flow start	4-3-3
STLE	Flow end	4-3-3
SET	Open the assigned flow, close the current flow (flow jump)	4-3-3
ST	Open the assigned flow, not close the current flow (Open the new flow)	4-3-3
FOR	Start of a FOR-NEXT loop	4-3-4
NEXT	End of a FOR-NEXT loop	4-3-4
FEND	First End	4-3-5
END	Program End	4-3-5

4-3-1 Condition Jump [CJ]

1: Summary

As used to run a part of program, CJ shorten the operation cycle and using the dual coil

Condition Jump [CJ]			
16 bits	CJ	32 bits	-
Execution condition	Normally ON/OFF coil	Suitable Models	XC1.XC2.XC3.XC5.XCM
Hardware requirement	-	Software requirement	-

2.Operands

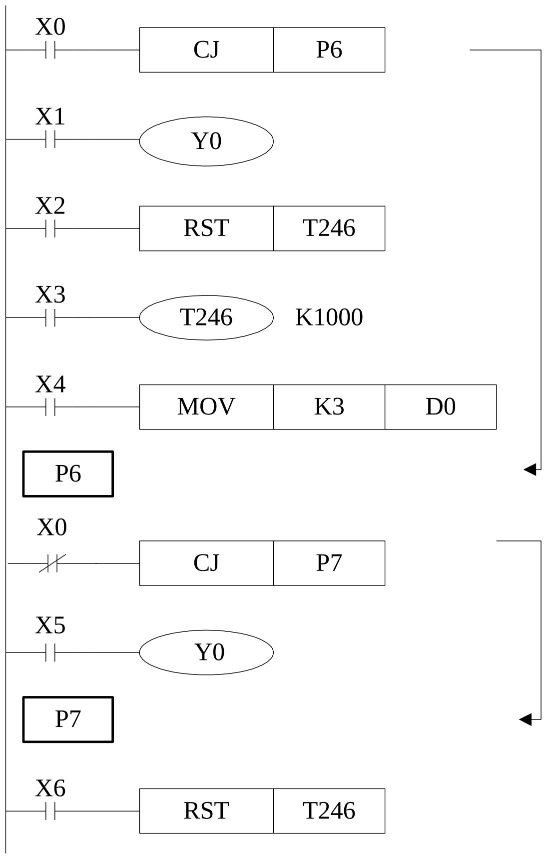
Operands	Function	Data Type
Pn	Jump to the target (with pointer Nr.) P (P0~P9999)	Pointer's Nr.

3.Suitable Soft Components

Other	Pointer	
	P	I
	●	

Description

In the below graph, if X000 is "ON", jump from the first step to the next step behind P6 tag. If X000 "OFF", do not execute the jump construction;



- In the left graph, Y000 becomes to be dual coil output, but when X000=OFF, X001 activates; when X000=ON, X005 activates
- CJ can't jump from one STL to another STL;
- After driving time T0~T640 and HSC C600~C640, if execute CJ, continue to work, the output activates.

4-3-2. Call subroutine [CALL] and Subroutine return [SRET]

1: Summary

Call the programs which need to be executed together, decrease the program's steps

Subroutine Call [**CALL**]			
16 bits	CALL	32 bits	-

Execution condition	Normally ON/OFF, Rising/Falling edge	Suitable Models	XC1.XC2.XC3.XC5.XCM
Hardware requirement	-	Software requirement	-
Subroutine Return [**SRET**]			
16 bits	SRET	32 bits	-
Execution condition	-	Suitable Models	XC1.XC2.XC3.XC5.XCM
Hardware requirement	-	Software requirement	-

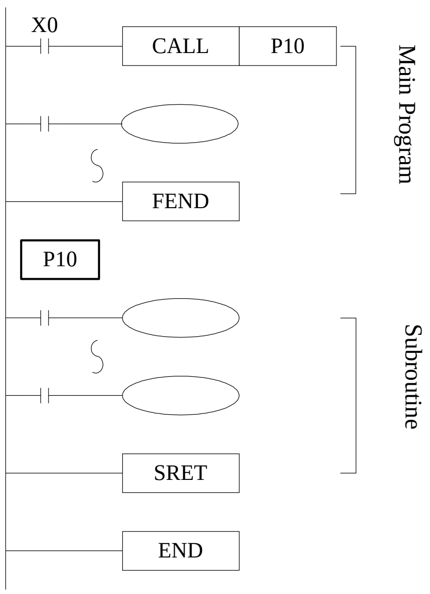
2.Operands

Operands	Function	Data Type
Pn	Jump to the target (with pointer Nr.) P (P0~P9999)	Pointer's Nr.

3.Suitable Soft Components

	Pointer	
Others	P	I
	●	

Description



- If X000= "ON", execute the call instruction and jump to the step tagged by P10. after executing the subroutine, return the original step via SRET instruction.Program the tag with FEND instruction (will describe this instruction later)
- In the subroutine 9 times call is allowed, so totally there can be 10 nestings.

4-3-3 Flow [SET], [ST], [STL], [STLE]

1: Summary

Instructions to specify the start, end, open, close of a flow;

Open the specified flow, close the local flow [**SET**]			
16 bits	SET	32 bits	-

Execution condition	Normally ON/OFF, Rising/Falling edge	Suitable Models	XC1.XC2.XC3.XC5.XCM
Hardware requirement	-	Software requirement	-
Open the specified flow, not close the local flow [**ST**]			
16 bits	ST	32 bits	-
Execution condition	Normally ON/OFF, Rising/Falling edge	Suitable Models	XC1.XC2.XC3.XC5.XCM
Hardware requirement	-	Software requirement	-
Flow starts [**STL**]			
16 bits	STL	32 bits	-
Execution condition	-	Suitable Models	XC1.XC2.XC3.XC5.XCM
Hardware requirement	-	Software requirement	-
Flow ends [**STLE**]			
16 bits	STLE	32 bits	-
Execution condition	-	Suitable Models	XC1.XC2.XC3.XC5.XCM
Hardware requirement	-	Software requirement	-

2: Operands

Operands	Function	Data Type
Sn	Jump to the target flow S	Flow ID

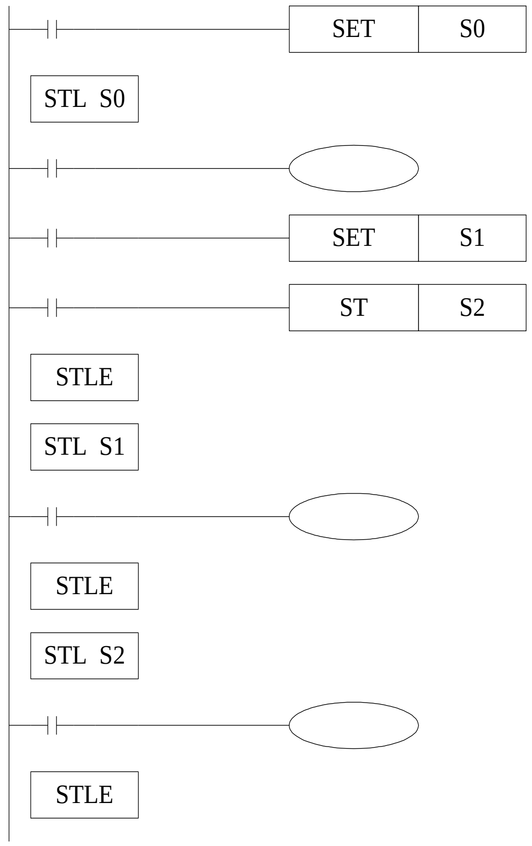
3: Suitable Soft Components

Bit

Operands	System						
	X	Y	M	S	T	C	Dn.m
Sn				●			

Description

- STL and STLE should be used in pairs. STL represents the start of a flow, STLE represents the end of a flow.
- After executing of **SET Sxxx** instruction, the flow specified by these instructions is ON.
- After executing **RST Sxxx** instruction, the specified flow is OFF.
- In flow S0, SET S1 close the current flow S0, open flow S1.
- In flow S0, ST S2 open the flow S2, but don’t close flow S0.
- When flow turns from ON to be OFF, reset OUT、PLS、PLF、not accumulate timer etc. which belongs to the flow.



4-3-4 [FOR] and [NEXT]

1: Summary

Loop execute the program between **FOR** and **NEXT** with the specified times;

Loop starts [**FOR**]			
16 bits	FOR	32 bits	-

Execution condition	Rising/Falling edge	Suitable Models	XC1.XC2.XC3.XC5.XCM
Hardware requirement	-	Software requirement	-
Loop ends [**NEXT**]			
16 bits	NEXTs	32 bits	-
Execution condition	Normally ON/OFF, Rising/Falling edge	Suitable Models	XC1.XC2.XC3.XC5.XCM
Hardware requirement	-	Software requirement	-

2: Operands

Operands	Function	Data Type
S	Program’s loop times between FOR~NEXT	16 bits, BIN

3: Suitable Soft Components

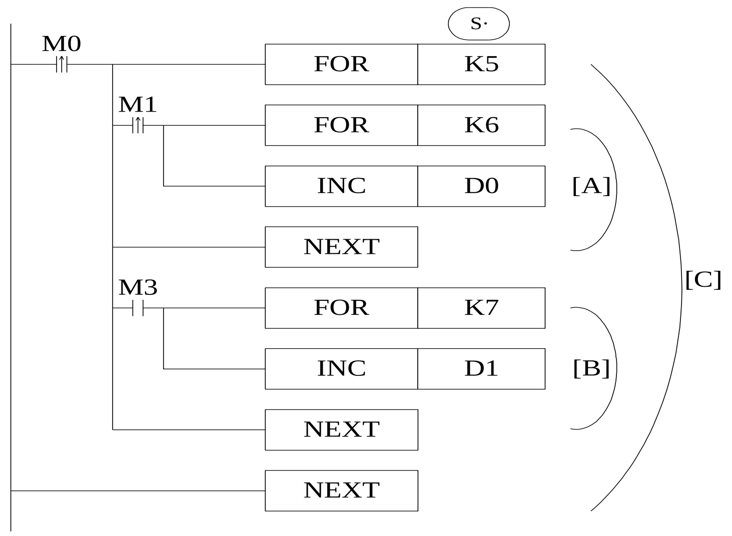
Word

Operands	System									Constant	Module	
	D	FD	ED	TD	CD	DX	DY	DM	DS	K/H	ID	QD
S	●									●		

Description

- FOR.NEXT instructions must be programmed as a pair. Nesting is allowed, and the nesting level is 8.

- Between FOR/NEXT, LDP.LDF instructions are effective for one time. Every time when M0 turns from OFF to ON, and M1 turns from OFF to ON, [A] loop is executed 6 times.
- Every time if M0 turns from OFF to ON and M3 is ON, [B] loop is executed 5×7=35 times.
- If there are many loop times, the scan cycle will be prolonged. Monitor timer error may occur, please note this.
- If NEXT is before FOR, or no NEXT, or NEXT is behind FENG,END, or FOR and NEXT number is not equal, an error will occur.
- Between FOR~NEXT, CJ nesting is not allowed, also in one STL, FOR~NEXT must be programmed as a pair.



4-3-5 [FEND] and [END]

1: Summary

FEND means the main program ends, while END means program ends;

main program ends [**FEND**]			
Execution condition	-	Suitable Models	XC1.XC2.XC3.XC5.XCM
Hardware	-	Software	-

requirement		requirement	
program ends [**END**]			
Execution condition	-	Suitable Models	XC1.XC2.XC3.XC5.XCM
Hardware requirement	-	Software requirement	-

2: Operands

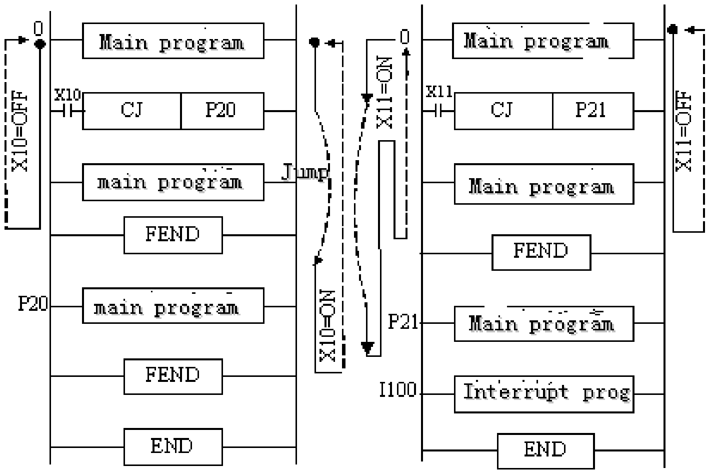
Operands	Function	Data Type
None	-	-

3: Suitable Soft Components

None

Description

Even though [FEND] instruction represents the end of the main program, if execute this instruction, the function is same with END. Execute the output/input disposal, monitor the refresh of the timer, return to the 0th step.



- If program the tag of CALL instruction behind FEND instruction, there must be SRET instruction. If the interrupt pointer program behind FEND instruction, there must be IRET instruction.
- After executing CALL instruction and before executing SRET instruction, if execute FEND instruction; or execute FEND instruction after executing FOR instruction and before executing NEXT, then an error will occur.
- In the condition of using many FEND instruction, please compile routine or subroutine between the last FEND instruction and END instruction.

4-4 Data Compare Function

Mnemonic	Function	Chapter
LD＝	LD activates when（S1)＝（S2)	4-4-1
LD＞	LD activates when（S1)＞（S2)	4-4-1

LD＜	LD activates when（S1)＜（S2)	4-4-1
LD＜＞	LD activates when（S1)≠（S2)	4-4-1
LD＜＝	LD activates when（S1)≤（S2)	4-4-1
LD＞＝	LD activates when（S1)≥（S2)	4-4-1
AND＝	AND activates when（S1)＝（S2)	4-4-2
AND＞	AND activates when（S1)＞（S2)	4-4-2
AND＜	AND activates when（S1)＜（S2)	4-4-2
AND＜＞	AND activates when（S1)≠（S2)	4-4-2
AND＜＝	AND activates when（S1)≤（S2)	4-4-2
AND＞＝	AND activates when（S1)≥（S2)	4-4-2
OR＝	OR activates when（S1)＝（S2)	4-4-3
OR＞	OR activates when（S1)＞（S2)	4-4-3
OR＜	OR activates when（S1)＜（S2)	4-4-3
OR＜＞	OR activates when（S1)≠（S2)	4-4-3
OR＜＝	OR activates when（S1)≤（S2)	4-4-3
OR＞＝	OR activates when（S1)≥（S2)	4-4-3

4-4-1 LD Compare [LD]

1: Summary

LD is the point compare instruction connected with the generatrix.

LD Compare [LD]			
16 bits	As below	32 bits	As below
Execution condition	-	Suitable Models	XC1.XC2.XC3.XC5.XCM
Hardware requirement	-	Software requirement	-

2: Operands

Operands	Function	Data Type
S1	Specify the Data (to be compared) or soft component's address code	16/32bits, BIN
S2	Specify the comparand's value or soft component's address code	16/32 bits, BIN

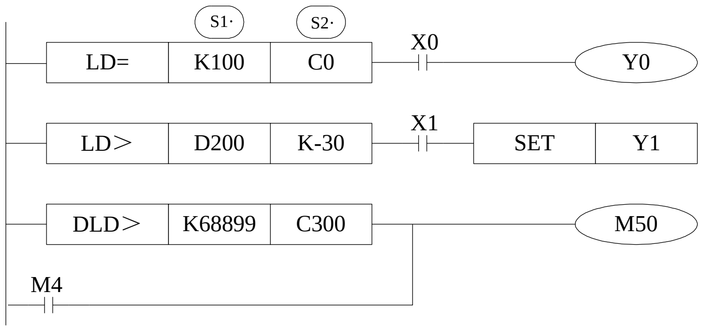
3: Suitable soft components

Word

Operands	System									Constant	Module	
	D	FD	ED	TD	CD	DX	DY	DM	DS	K/H	ID	QD
S1	●	●		●	●	●	●	●	●	●		
S2	●	●		●	●	●	●	●	●	●		

Description

16 bits instruction	32 bits instruction	Activate Condition	Not Activate Condition
LD＝	DLD＝	（S1)＝（S2)	（S1)≠（S2)
LD＞	DLD＞	（S1)＞（S2)	（S1)≤（S2)
LD＜	DLD＜	（S1)＜（S2)	（S1)≥（S2)
LD＜＞	DLD＜＞	（S1)≠（S2)	（S1)＝（S2)
LD＜＝	DLD＜＝	（S1)≤（S2)	（S1)＞（S2)
LD＞＝	DLD＞＝	（S1)≥（S2)	（S1)＜（S2)



Notes

- When the source data's highest bit (16 bits：b15，32 bits：b31) is 1，use the data as a negative.
- The comparison of 32 bits counter (C300~) must be 32 bits instruction. If assigned as a 16 bits instruction, it will lead the program error or operation error.

4-4-2．AND Compare [AND]

1: Summary

AND: The compare instruction to serial connect with the other contactors.

AND Compare [AND]			
16 bits	As Below	32 bits	As Below
Execution condition	Normally ON/OFF coil	Suitable Models	XC1.XC2.XC3.XC5.XCM
Hardware requirement	-	Software requirement	-

2: Operands

Operands	Function	Data Type

S1	Specify the Data (to be compared) or soft component's address code	16/32bit,BIN
S2	Specify the comparand's value or soft component's address code	16/32bit,BIN

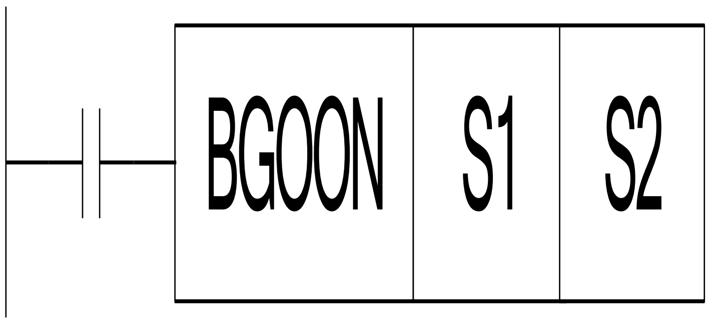
3: Suitable soft components

Word

Operands	System									Konstant	Module	
	D	FD	ED	TD	CD	DX	DY	DM	DS	K/H	ID	QD
S1	●	●		●	●	●	●	●	●	●		
S2	●	●		●	●	●	●	●	●	●		

Description

16 bits instruction	32 bits instruction	Activate Condition	Not Activate Condition
AND＝	DAND＝	（S1)＝（S2)	（S1)≠（S2)
AND＞	DAND＞	（S1)＞（S2)	（S1)≤（S2)
AND＜	DAND＜	（S1)＜（S2)	（S1)≥（S2)
AND＜＞	DAND＜＞	（S1)≠（S2)	（S1)＝（S2)
AND＜＝	DAND＜＝	（S1)≤（S2)	（S1)＞（S2)
AND＞＝	DAND＞＝	（S1)≥（S2)	（S1)＜（S2)



Notes

- When the source data's highest bit (16 bits：b15，32 bits：b31) is 1，use the data as a negative.
- The comparison of 32 bits counter (C300~) must be 32 bits instruction. If assigned as a 16 bits instruction, it will lead the program error or operation error.

4-4-3. Parallel Compare [OR]

1: Summary

OR The compare instruction to parallel connect with the other contactors

Parallel Compare [OR]			
16 bits	As below	32 bits	As below
Execution condition	-	Suitable Models	XC1.XC2.XC3.XC5.XCM
Hardware requirement	-	Software requirement	-

2: Operands

Operands	Function	Data Type
S1	Specify the Data (to be compared) or soft component's address code	16/32 bit,BIN
S2	Specify the comparand's value or soft component's address code	16/32 bit,BIN

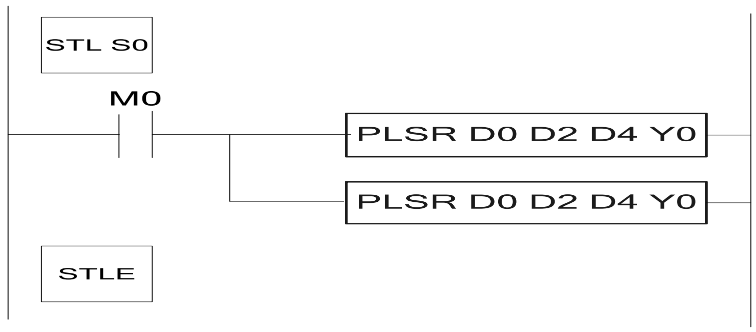
3: Suitable soft components

Word

Operands	System									Constant	Module	
	D	FD	ED	TD	CD	DX	DY	DM	DS	K/H	ID	QD
S1	●	●		●	●	●	●	●	●	●		
S2	●	●		●	●	●	●	●	●	●		

Description

16 bits instruction	32 bits instruction	Activate Condition	Not Activate Condition
OR＝	DOR＝	（S1)＝（S2)	（S1)≠（S2)
OR＞	DOR＞	（S1)＞（S2)	（S1)≤（S2)
OR＜	DOR＜	（S1)＜（S2)	（S1)≥（S2)
OR＜＞	DOR＜＞	（S1)≠（S2)	（S1)＝（S2)
OR＜＝	DOR＜＝	（S1)≤（S2)	（S1)＞（S2)
OR＞＝	DOR＞＝	（S1)≥（S2)	（S1)＜（S2)



Notes

- When the source data's highest bit (16 bits：b15，32 bits：b31) is 1，use the data as a negative.
- The comparison of 32 bits counter (C300~) must be 32 bits instruction. If assigned as a 16 bits instruction, it will lead the program error or operation error.

4-5 Data Move

Mnemonic	Function	Chapter
CMP	Data compare	4-5-1
ZCP	Data zone compare	4-5-2
MOV	Move	4-5-3
BMOV	Data block move	4-5-4
PMOV	Data block move (with faster speed)	4-5-5
FMOV	Fill move	4-5-6
FWRT	FlashROM written	4-5-7
MSET	Zone set	4-5-8
ZRST	Zone reset	4-5-9
SWAP	The high and low byte of the destinated devices are exchanged	4-5-10
XCH	Exchange	4-5-11

4-5-1 Data Compare [CMP]

1. Summary

Compare the two specified Data, output the result.

Data compare [CMP]

16 bits	CMP	32 bits	DCMP
Execution condition	Normally ON/OFF, rising/falling edge	Suitable Models	XC1.XC2.XC3.XC5.XCM
Hardware requirement	-	Software requirement	-

2: Operands

Operands	Function	Data Type
S1	Specify the data (to be compared) or soft component's address code	16 bit,BIN
S	Specify the comparand's value or soft component's address code	16 bit,BIN
D	Specify the compare result's address code	bit

3: Suitable soft component

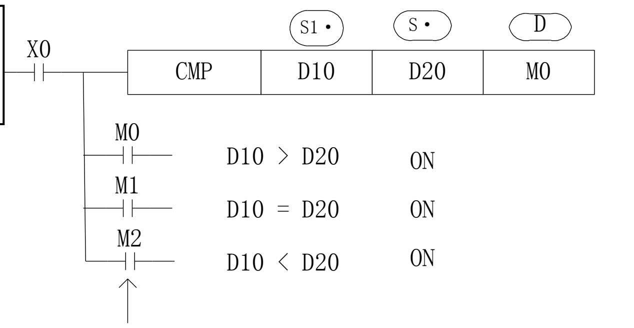
Word

Operands	System									Constant	Module	
	D	FD	ED	TD	CD	DX	DY	DM	DS	K/H	ID	QD
S1	●	●		●	●	●	●	●	●	●		
S	●	●		●	●	●	●	●	●	●		

Bit

Operands	System						
	X	Y	M	S	T	C	Dn.m
D		●	●	●			

Description



Even X000=OFF to stop ZCP instruction, M0~M2 will keep the original status

- Compare data S1· and S·, output the three points' ON/OFF status (start with D·
- D·, D·+1, D·+2 : the three point's on/off output according to the valve

4-5-2 Data zone compare [ZCP]

1: Summary

Compare the two specify Data with the current data, output the result.

Data Zone compare [ZCP]			
16 bits	ZCP	32 bits	DZCP
Execution condition	Normally ON/OFF, rising/falling edge	Suitable Models	XC1.XC2.XC3.XC5.XCM
Hardware requirement	-	Software requirement	-

2: Operands

Operands	Function	Data Type
S1	Specify the down-limit Data (of the compare stand) or soft component's address code	16 bit, BIN
S2	Specify the Up-limit Data (of the compare stand) or soft component's address code	16 bit, BIN
S	Specify the current data or soft component's address code	16 bit, BIN
D	Specify the compare result's data or soft component's address code	bit

3: Suitable soft components

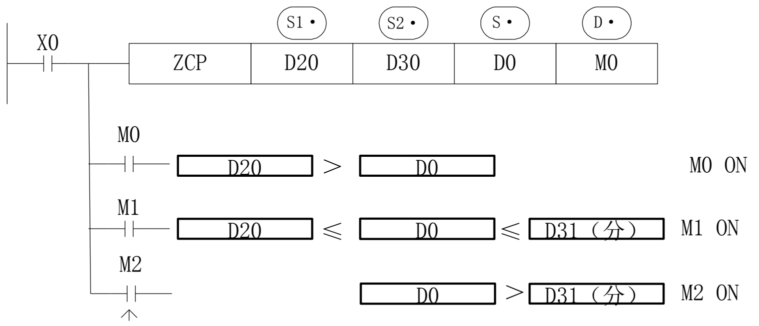
Word

Operands	System									Constant	Module	
	D	FD	ED	TD	CD	DX	DY	DM	DS	K/H	ID	QD
S1	●	●		●	●	●	●	●	●	●		
S2	●	●		●	●	●	●	●	●	●		
S	●	●		●	●	●	●	●	●	●		

Bit

Operands	System						
	X	Y	M	S	T	C	Dn.m
D		●	●	●			

Description



Even X000=OFF stop ZCP instruction，M0~M2 will keep the original status

- Compare S· data with S1 and S2 , D· output the three point's ON/OFF status according to the zone size.
- ， ＋1， ＋2 : the three point's ON/OFF output according to the result

4-5-3 MOV [MOV]

1: Summary

Move the specified data to the other soft components

MOV [MOV]			
16 bits	MOV	32 bits	DMOV
Execution condition	Normally ON/OFF, rising/falling edge	Suitable Models	XC1.XC2.XC3.XC5.XCM
Hardware requirement	-	Software requirement	-

2: Operands

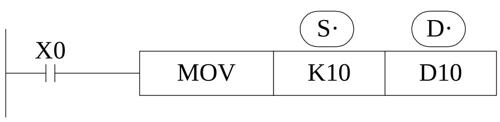
Operands	Function	Data Type
S	Specify the source data or register's address code	16 bit/32 bit, BIN
D	Specify the target soft component's address code	16 bit/32 bit, BIN

3: Suitable soft component

Word

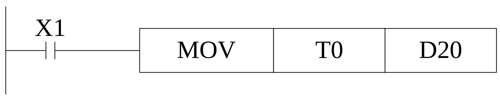
Operands	System									Constant	Module	
	D	FD	ED	TD	CD	DX	DY	DM	DS	K/H	ID	QD
S	●	●	●	●	●	●	●	●	●	●		
D	●		●	●	●		●	●	●			

Description

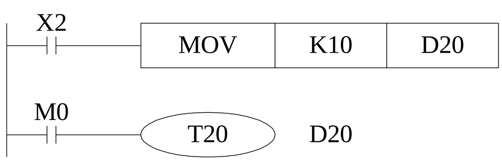


- Move the source data to the target
- When X000 is off, the data keeps same
- Convert constant K10 to be BIN code automatically

<read the counter's or time's current value>



<indirectly specify the counter's ,time's set value>



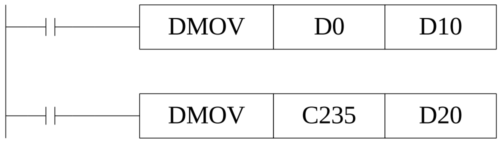
(The current value of T0)→(D20)

The same as counter

（K10)（D10)

D20=K10

< Move the 32bits data >



Please use DMOV when the value is 32 bits, such as MUL instruction, high speed counter…

(D1，D0)→(D11，D10)

(the current value of C235)→(D21，D20)

4-5-4. Data Block Move [BMOV]

1: Summary

Move the specified data block to

Data block move [BMOV]			
16 bits	BMOV	32 bits	-
Execution condition	Normally ON/OFF coil	Suitable Models	XC1.XC2.XC3.XC5.XCM
Hardware requirement	-	Software requirement	-

2: Operands

Operands	Function	Data Type
S	Specify the source data block or soft component address code	16 bits, BIN; bit
D	Specify the target soft components address code	16 bits, BIN; bit
n	Specify the move data's number	16 bits, BIN;

3: Suitable soft components

Word

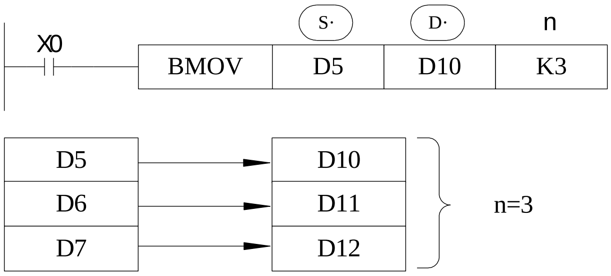
Operands	System									Constant	Module	
	D	FD	ED	TD	CD	DX	DY	DM	DS	K/H	ID	QD
S	●	●	●	●	●	●	●	●	●			
D	●		●	●	●		●	●	●			
n	●			●	●	●		●	●	●		

Bit

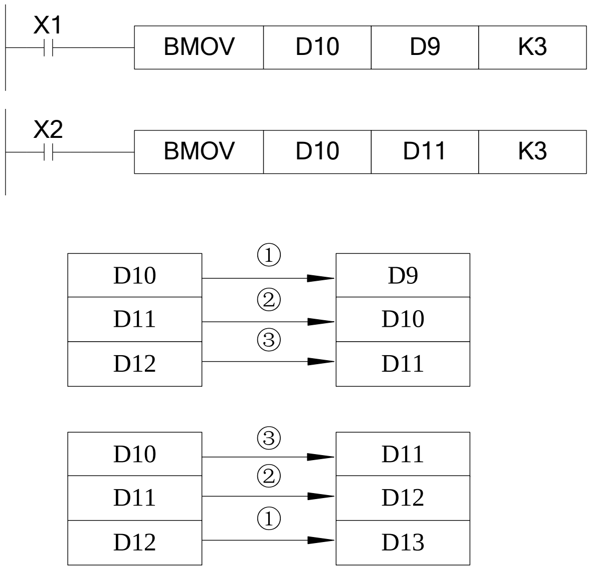
Operands	System						
	X	Y	M	S	T	C	Dn.m
S	●	●	●				
D	●	●	●				

Description

(1) Move the specified "n" data to the specified "n" soft components in the form block.



(2) As the following picture, when the data address overlapped, the instruction will do from 1 to 3.



4-5-5 Data Block Move [PMOV]

1: Summary

Move the specified data block to the other soft components

Data block mov[PMOV]			
16 bits	PMOV	32 bits	-
Execution condition	Normally ON/OFF coil	Suitable Models	XC1.XC2.XC3.XC5.XCM
Hardware requirement	-	Software requirement	-

2: Operands

Operands	Function	Data Type
S	Specify the source data block or soft component address code	16 bits, BIN; bit
D	Specify the target soft components address code	16 bits, BIN; bit
n	Specify the move data's number	16 bits, BIN;

3: Suitable soft components

Word

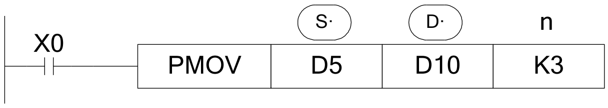
Operands	System									Constant	Module	
	D	FD	ED	TD	CD	DX	DY	DM	DS	K/H	ID	QD
S	●	●	●	●	●	●	●	●	●			
D	●		●	●	●		●	●	●			
n	●			●	●		●	●	●	●		

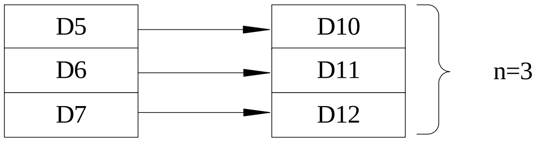
Bit

Operands	system						
	X	Y	M	S	T	C	Dn.m
S	●	●	●				
D	●	●	●				

Description

(3) Move the specifed "n" data to the specified "n" soft components in form of block





- The function of PMOV and BMOV is mostly the same, but the PMOV has the faster speed
- PMOV finish in one scan cycle, when executing PMOV , close all the interruptions
- Mistake many happen, if there is a repeat with source address and target address

4-5-6 Fill Move [FMOV]

1: Summary

Move the specified data block to the other soft components

Fill Move [FMOV]			
16 bits	FMOV	32 bits	DFMOV
Execution condition	Normally ON/OFF, rising/falling edge	Suitable Models	XC1.XC2.XC3.XC5.XCM
Hardware requirement	DFMOV need above V3.0	Software requirement	-

2: Operands

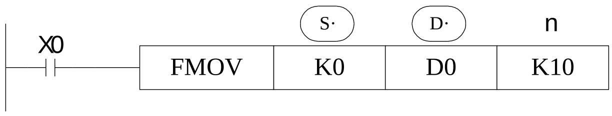
Operands	Function	Data Type
S	Specify the source data block or soft component address code	16 bits, BIN; bit
D	Specify the target soft components address code	16 bits, BIN; bit
n	Specify the move data’s number	16 bits, BIN;

3: Suitable soft component

Word	Operands	System									Constant	Module	
		D	FD	ED	TD	CD	DX	DY	DM	DS	K/H	ID	QD
	S	●	●	●	●	●	●	●	●	●	●		
	D	●		●	●	●		●	●	●			
	n	●			●	●		●	●	●	●		

Description

<16 bits instruction>

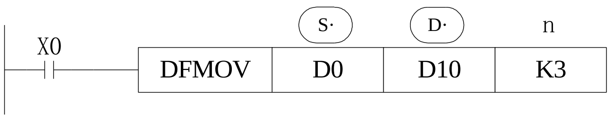


(4) Move K0 to D0~D9, copy a single data device to a range of destination device.

(5) The data stored in the source device (S) is copied to every device within the destination range, The range is specified by a device head address (D) and a quantity of consecutive elements (n).

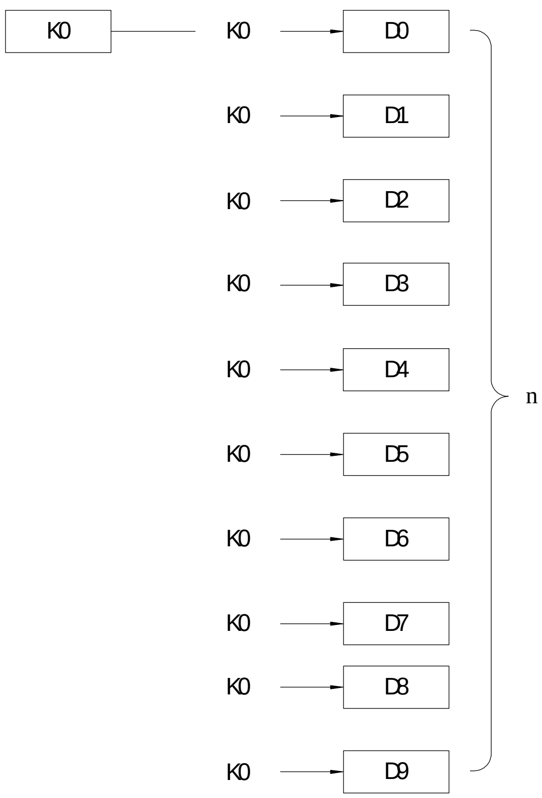
(6) If the specified number of destination devices (n) exceeds the available space at the destination location, then only the available destination devices will be written to.

<32 bits instruction >

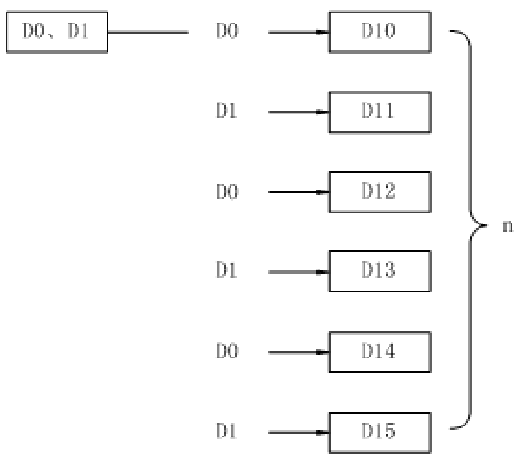


- Move D0.D1 to D10.D11:D12.D13:D14.D15.

<16 bits Fill Move >



<32 bits Fill move>



4-5-7 FlashROM Write [FWRT]

1: Summary

Write the specified data to other soft components

FlashROM Write [FWRT]			
16 bits	FWRT	32 bits	DFWRT
Execution condition	rising/falling edge	Suitable Models	XC1.XC2.XC3.XC5.XCM
Hardware requirement	-	Software requirement	-

2: Operands

Operands	Function	Data Type
S	The data write in the source or save in the soft element	16 bits/32 bits, BIN
D	Write in target soft element	16 bits/32 bits, BIN
D1	Write in target soft element start address	16 bits/32 bits, BIN
D2	Write in data quantity	bit

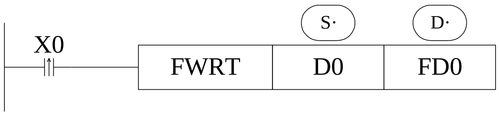
3: Suitable soft components

Word

Operands	System									Constant	Module	
	D	FD	ED	TD	CD	DX	DY	DM	DS	K/H	ID	QD
S	●	●		●	●	●	●	●	●	●		
D		●										
D1		●										
D2	●			●	●	●	●	●	●	●		

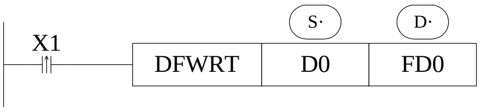
Description

< Written of a word >



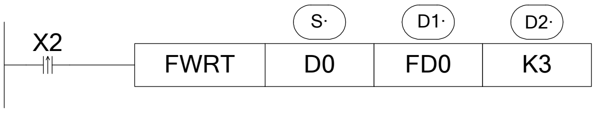
Write value in D0 into FD0

<Written of double word>



Write value in D0,D1 into FD0,FD1

<Written of multi-word>



Write value in D0,D1,D2 into FD0,FD1,FD2

※1：FWRT instruction only allow to write data into FlashRom register. In this storage, even battery drop, data could be used to store important technical parameters

※2：Written of FWRT needs a long time, about 150ms, so frequently operate this operate this operate operation is recommended

※3：The written time of Flshrom is about 1,000,000 times. So we suggest using edge signal (LDP, LDF etc.) to trigger.

※4：Frequently written of FlashROM

4-5-8 Zone set [MSET]

1: Summary

Set or reset the soft element in certain range

Multi-set [MSET]			
16 bits	MSET.ZRST	32 bits	-
Execution condition	Normally ON/OFF	Suitable Models	XC1.XC2.XC3.XC5.XCM
Hardware requirement	-	Software requirement	-

2: Operands

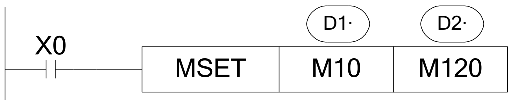
Operands	Function	Data Type
D1	Start soft element address	bit
D2	End soft element address	bit

3: Suitable soft components

Bit

Operands	System						
	X	Y	M	S	T	C	Dn.m
D1	●	●	●	●	●	●	
D2	●	●	●	●	●	●	

Description



Zone set unit M10~M120

- D1· D2· Are specified as the same type of soft units, and D1· < D2·
- When D1· > D2·, will not run Zone set, set M8004.M8067, and D8067=2。

4-5-9 Zone reset [ZRST]

1: Summary

Reset the soft element in the certain range

Multi-reset [ZRST]			
16 bits	ZRST	32 bits	-
Execution condition	Normally ON/OFF	Suitable Models	XC1.XC2.XC3.XC5.XCM
Hardware requirement	-	Software requirement	-

2: Operands

Operands	Function	Data Type
D1	Start address of soft element	Bit:16 bits,BIN
D2	End address of soft element	Bit:16 bits,BIN

3: Suitable soft components

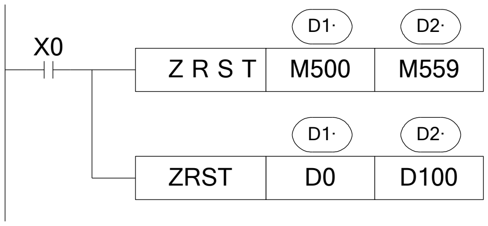
Word

Operands	System									Constant	Module	
	D	FD	ED	TD	CD	DX	DY	DM	DS	K/H	ID	QD
D1	●					●	●	●				
D2	●				●	●	●	●				

Bit

Operands	System						
	X	Y	M	S	T	C	Dn.m
D1	●	●	●	●	●	●	
D2	●	●	●	●	●	●	

Description



Zone reset bits M5 00~M559。

Zone reset words D0~D100

- D1· D2· Are specified as the same type of soft units, and D1· < D2·
- When D1· > D2· only reset the soft unit specified in D1·, and set M8004, M8067, D8067=2.

Other Reset Instruction

1. As soft unit's separate reset instruction, RST instruction can be used to bit unit Y, M, S and word unit T, C, D
2. As fill move for constant K0, 0 can be written into DX, DY, DM, DS, T, C, D.

4-5-10 Swap the high and low byte [SWAP]

1: Summary

Swap the high and low byte

High and low byte swap [SWAP]			
16 bits	SWAP	32 bits	-
Execution condition	Normally ON/OFF	Suitable Models	XC1.XC2.XC3.XC5.XCM
Hardware requirement	-	Software requirement	-

2: Operands

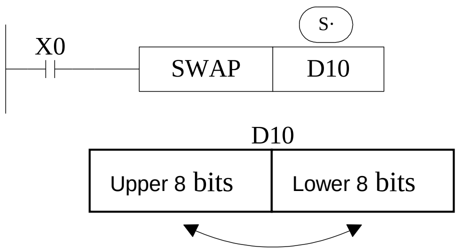
Operands	Function	Data Type
S	The address of the soft element	16 bits: BIN

3: Suitable soft components

Word

Operands	System									Constant	Module	
	D	FD	ED	TD	CD	DX	DY	DM	DS	K/H	ID	QD
S	●			●	●							

Description



- Low 8 bits and high 8 bits change when it is 16 bits instruction.
- If the instruction is a consecutive executing instruction, each operation cycle should change.

4-5-11 Exchange [XCH]

1: Summary

Exchange the data in two soft element

Exchange [XCH]			
16 bits	XCH	32 bits	DXCH
Execution condition	Normally ON/OFF	Suitable Models	XC1.XC2.XC3.XC5.XCM
Hardware requirement	-	Software requirement	-

2: Operands

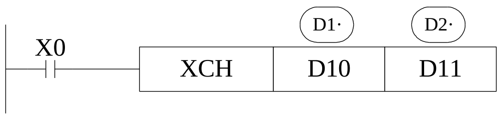
Operands	Function	Data Type
D1	The soft element address	16 bits, BIN
D2	The soft element address	16 bits, BIN

3: Suitable soft component

Word	Operands	System									Constant	Module	
		D	FD	ED	TD	CD	DX	DY	DM	DS	K/H	ID	QD
	D1	●			●	●		●	●	●			
	D2	●			●	●		●	●	●			

Description

<16 bits instruction>

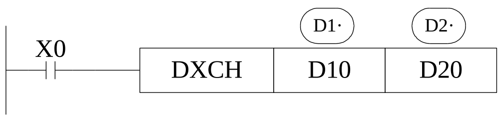


Before（D10）=100 →After （D10）=101

（D11）=101 （D11）=100

- The contents of the two destination devices D1 and D2 are swapped,
- When drive input X0 is ON, each scan cycle should carry on data exchange, please note.

<32 bits instruction >



- 32 bits instruction [DXCH] swaps value composed by D10、D11 and the value composed by D20、D21.

4-6 Data Operation Instructions

Mnemonic	Function	Chapter
ADD	Addition	4-6-1
SUB	Subtraction	4-6-2
MUL	Multiplication	4-6-3
DIV	Division	4-6-4
INC	Increment	4-6-5
DEC	Decrement	4-6-5
MEAN	Mean	4-6-6
WAND	Logic Word And	4-6-7
WOR	Logic Word Or	4-6-7
WXOR	Logic Exclusive Or	4-6-7
CML	Compliment	4-6-8
NEG	Negation	4-6-9

4-6-1 Addition [ADD]

1: Summary

Add two numbers and store the result

Add [ADD]			
16 bits	ADD	32 bits	DADD
Execution condition	Normally ON/OFF	Suitable Models	XC1.XC2.XC3.XC5.XCM
Hardware requirement	-	Software requirement	-

2: Operands

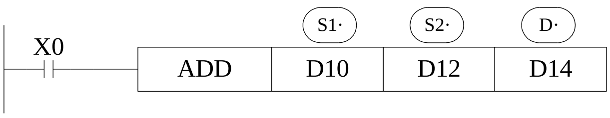
Operands	Function	Data Type
S1	The number address	16 bit/32 bit, BIN
S2	The number address	16 bit/32bit, BIN
D	The result address	16 bit/32bit, BIN

3: Suitable soft components

Word

Operands	System									Constant	Module	
	D	FD	ED	TD	CD	DX	DY	DM	DS	K/H	ID	QD
S1	●	●		●	●	●	●	●	●	●		
S2	●	●		●	●	●	●	●	●	●		
D	●			●	●		●	●	●			

Description



（D10）＋（D12）→（D14）

3. The data contained within the two source devices are combined and the total is stored in the specified destination device. Each data’s highest bit is the sign bit, 0 stands for positive、1 stands for negative. All calculations are algebraic processed.（5+（-8）=-3）
4. If the result of a calculation is “0”, the “0” flag acts. If the result exceeds 323，767（16 bits limit）or 2,147,483,647（32 bits limit）, the carry flag acts.（refer to the next page）. If the result exceeds –323,768（16 bits limit）or –2,147,483,648（32 bits limit）, the borrow flag acts（Refer to the next page。
5. When carry on 32 bits operation, word device’s low 16 bits are assigned, the device following closely the preceding device’s ID will be the high bits. To avoid ID repetition, we recommend you assign device’s ID to be even ID.
6. The same device may be used as a source and a destination. If this is the case then the result changes after every scan cycle. Please note this point.

Related Flag

Flag meaning:

Flag	Name	Function
M8020	Zero	ON：the calculate result is zero OFF：the calculate result is not zero
M8021	Borrow	ON：the calculate result is less than -32768(16 bit) or -2147483648(32bit) OFF：the calculate result is over -32768(16 bit) or -2147483648(32bit)
M8022	Carry	ON：the calculate result is over 32768(16 bit) or 2147483648(32bit) OFF：the calculate result is less than 32768(16 bit) or 2147483648(32bit)

4-6-2 Subtraction [SUB]

1: Summary

Sub two numbers, store the result

Subtraction [SUB]			
16 bits	SUB	32 bits	DSUB

Execution condition	Normally ON/OFF	Suitable Models	XC1.XC2.XC3.XC5.XCM
Hardware requirement	-	Software requirement	-

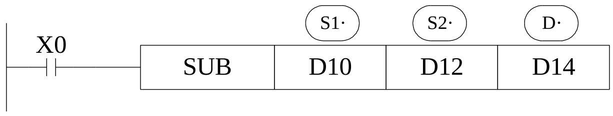
2: Operands

Operands	Function	Data Type
S1	The number address	16 bits /32 bits,BIN
S2	The number address	16 bits /32 bits,BIN
D	The result address	16 bits /32 bits,BIN

3: Suitable soft component

Word	Operands	System									Constant	Module	
		D	FD	ED	TD	CD	DX	DY	DM	DS	K/H	ID	QD
	S1	●	●		●	●	●	●	●	●	●		
	S2	●	●		●	●	●	●	●	●	●		
	D	●			●	●		●	●	●			

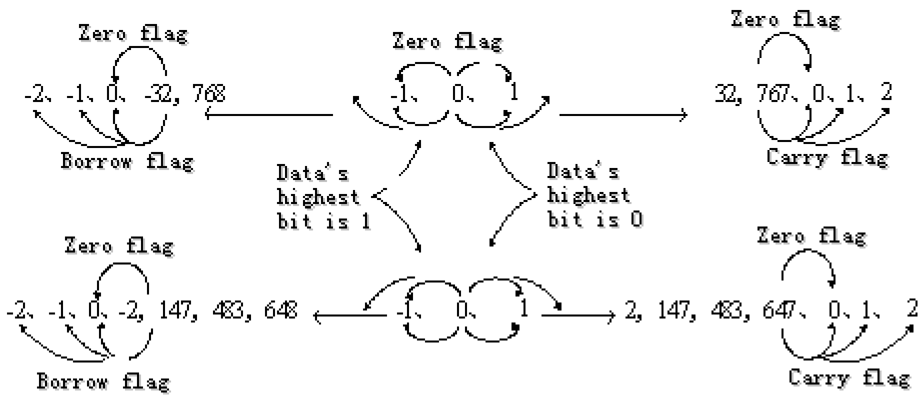
Description



（D10）—（D12）→（D14）

7. S1· appoint the soft unit's content, subtract the soft unit's content appointed by S2· in the format of algebra. The result will be stored in the soft unit appointed by D·. (5-(-8)=13)
8. The action of each flag, the appointment method of 32 bits operation's soft units are both the same with the preceding ADD instruction.
9. The importance is: in the preceding program, if X0 is ON, SUB operation will be executed every scan cycle

The relationship of the flag's action and vale's positive/negative is shown below:



4-6-3 Multiplication [MUL]

1: Summary

Multiply two numbers, store the result

Multiplication [MUL]			
16 bits	MUL	32 bits	DMUL
Execution condition	Normally ON/OFF	Suitable Models	XC1.XC2.XC3.XC5.XCM
Hardware requirement	-	Software requirement	-

2: Operands

Operands	Function	Data Type
S1	The number address	16 bits/32bits,BIN
S2	The number address	16 bits/32bits,BIN
D	The result address	16 bits/32bits,BIN

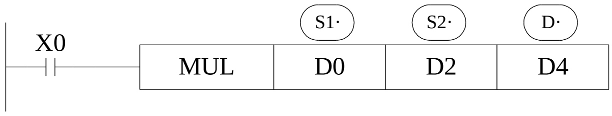
3: Suitable soft component

Word

Operands	System									Constant	Module	
	D	FD	ED	TD	CD	DX	DY	DM	DS	K/H	ID	QD
S1	●	●		●	●	●	●	●	●	●		
S2	●	●		●	●	●	●	●	●	●		
D	●			●	●		●	●	●			

Description

<16 bits Operation>



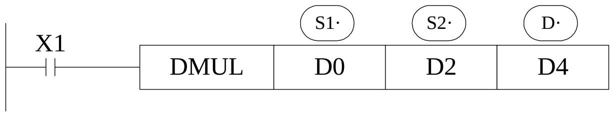
BIN BIN BIN

（D0） × (D2) → （D5，D4）

16 bits 16 bits → 32 bits

10. The contents of the two source devices are multiplied together and the result is stored at the destination device in the format of 32 bits. As in the upward chart: when (D0)=8,(D2)=9, (D5, D4) =72.
11. The result's highest bit is the symbol bit: positive (0), negative (1).
12. When be bit unit, it can carry on the bit appointment of K1~K8. When appoint K4, only the result's low 16 bits can be obtained.

<32 bits Operation >



BIN BIN BIN

（D1，D0）× （D3，D2） → （D7，D6，D5，D4）

32 bits 32 bits → 64 bits

13. When use 2 bits Operation, the result is stored at the destination device in the format of 64 bits.
14. Even when utilizing word device, 64 bits results can't be monitored at once.

4-6-4 Division [DIV]

1: Summary

Divide two numbers and store the result

Division [DIV]			
16 bits	DIV	32 bits	DDIV
Execution condition	Normally ON/OFF, rising/falling edge	Suitable Models	XC1.XC2.XC3.XC5.XCM
Hardware requirement	-	Software requirement	-

2: Operands

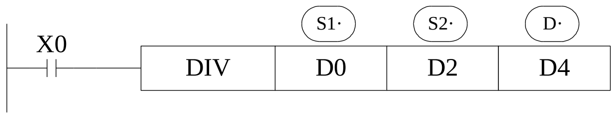
Operands	Function	Data Type
S1	The number address	16 bits / 32 bits, BIN
S2	The number address	16 bits /32 bits, BIN
D	The result address	16 bits /32 bits, BIN

3: Suitable soft components

Word	Operands	System									Constant	Module	
		D	FD	ED	TD	CD	DX	DY	DM	DS	K/H	ID	QD
	S1	●	●		●	●	●	●	●	●	●		
	S2	●	●		●	●	●	●	●	●	●		
	D	●			●	●		●	●	●			

Description

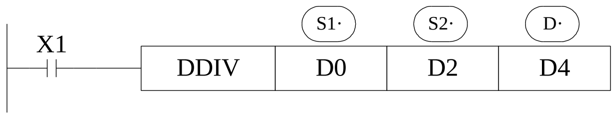
<**16 bits Operation** >



Dividend		Divisor		Result		Remainder
BIN		BIN		BIN		BIN
(D0)	÷	(D2)	→	D4）	---	(D5)
16 bits		16 bits		16 bits		16 bits

15. S1· appoints the device's content be the dividend, S2· appoints the device's content be the divisor, D· appoints the device and the next one to store the result and the remainder.
16. In the above example, if input X0 is ON, devision operation is executed every scan cycle.

<**32 bits Operation** >



Dividend		Divisor	Result		Remainder
BIN		BIN	BIN		BIN
(D1,D0)	÷	(D3,D2)	（D5,D4）	---	(D7,D6)
32 bits		32 bits	32 bits		32 bits

17. The dividend is composed by the device appointed by S1· and the next one. The divisor is composed by the device appointed by S2· and the next one. The result and the remainder are stored in the four sequential devices, the first one is appointed by D·
18. If the value of the divisor is 0, then an operation error is executed and the operation of the DIV instruction is cancelled
19. The highest bit of the result and remainder is the symbol bit (positive:0, negative: 1). When any of the dividend or the divisor is negative, then the result will be negative. When the dividend is negative, then the remainder will be negative.

4-6-5 Increment [INC] & Decrement [DEC]

1: Summary

Increase or decrease the number

Increment 1[INC]			
16 bits	INC	32 bits	DINC
Execution condition	Normally ON/OFF, rising/falling edge	Suitable Models	XC1.XC2.XC3.XC5.XCM
Hardware requirement	-	Software requirement	-
Increment 1[DEC]			
16 bits	DEC	32 bits	DDEC
Execution condition	Normally ON/OFF, rising/falling edge	Suitable Models	XC1.XC2.XC3.XC5.XCM
Hardware requirement	-	Software requirement	-

2: Operands

Operands	Function	Data Type
D	The number address	16 bits / 32bits,BIN

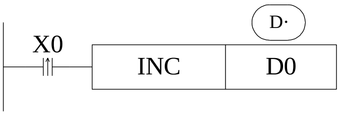
3: Suitable soft components

Word

Operands	System									Constant	Module	
	D	FD	ED	TD	CD	DX	DY	DM	DS	K/H	ID	QD
D	●			●	●		●	●	●			

Description

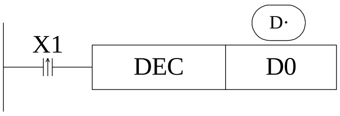
< Increment [INC]>



（D0）＋1→(D0)

20. On every execution of the instruction the device specified as the destination D· has its current value incremented (increased) by a value of 1.
21. In 16 bits operation, when +32,767 is reached, the next increment will write -32,767 to the destination device. In this case, there’s no additional flag to identify this change in the counted value.

<Decrement [DEC]>



（D0）－1 →（D0）

23. On every execution of the instruction the device specified as the destination D· has its current value decremented (decreased) by a value of 1.
24. When -32，768 or -2，147，483，648 is reached, the next decrement will write +32，767 or +2，147，483，647 to the destination device.

4-6-6 Mean [MEAN]

1: Summary

Get the mean value of numbers

Mean [MEAN]			
16 bits	MEAN	32 bits	DMEAN
Execution condition	Normally ON/OFF, rising/falling edge	Suitable Models	XC1.XC2.XC3.XC5.XCM
Hardware requirement	-	Software requirement	-

2: Operands

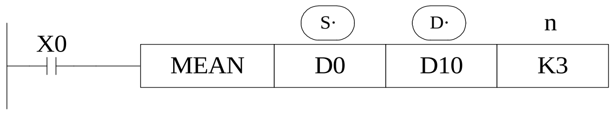
Operands	Function	Data Type
S	The head address of the numbers	16 bits, BIN
D	The mean result address	16 bits, BIN
n	The number quantity	16 bits, BIN

3: Suitable soft components

Word

Operands	System									Constant	Module	
	D	FD	ED	TD	CD	DX	DY	DM	DS	K/H	ID	QD
S	•	•		•	•		•	•	•			
D	•			•	•		•	•	•			
n										•		

Description



$$\frac{(D0) + (D1) + (D2)}{3} \longrightarrow (D10)$$

25. The value of all the devices within the source range is summed and then divided by the number of devices summed, i.e. n.. This generates an integer mean value which is stored in the destination device (D) The remainder of the calculated mean is ignored.
26. If the value of n is specified outside the stated range (1 to 64) an error is generated.

4-6-7 Logic AND [WAND], Logic OR[WOR], Logic Exclusive OR [WXOR]

1: Summary

Do logic AND, OR, XOR for numbers

Logic AND [WAND]			
16 bits	WAND	32 bits	DWAND
Execution condition	Normally ON/OFF, rising/falling edge	Suitable Models	XC1.XC2.XC3.XC5.XCM
Hardware requirement	-	Software requirement	-
Logic OR[WOR]			
16 bits	WOR	32 bits	DWOR
Execution condition	Normally ON/OFF, rising/falling edge	Suitable Models	XC1.XC2.XC3.XC5.XCM
Hardware requirement	-	Software requirement	-
Logic Exclusive OR [WXOR]			
16 bits	WXOR	32 bits	DWXOR
Execution condition	Normally ON/OFF, rising/falling edge	Suitable Models	XC1.XC2.XC3.XC5.XCM
Hardware requirement	-	Software requirement	-

2: Operands

Operands	Function	Data Type
S1	The soft element address	16bit/32bit,BIN
S2	The soft element address	16bit/32bit,BIN
D	The result address	16bit/32bit,BIN

3: Suitable soft components

Word

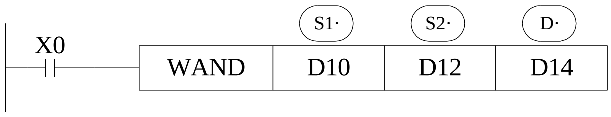
Operands	System									Constant	Module	
	D	FD	ED	TD	CD	DX	DY	DM	DS	K/H	ID	QD
S1	●	●		●	●	●	●	●	●			
S2	●	●		●	●	●	●	●	●			
D	●			●	●		●	●	●			

Description

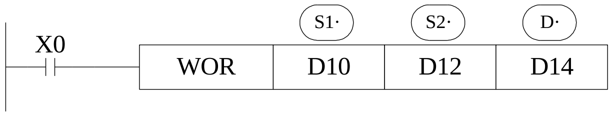
- **< Execute logic AND operation with each bit>**

0&0=0 0&1=0

1&0=0 1&1=1

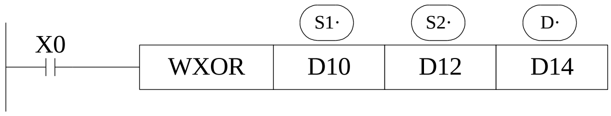


< Execute logic OR operation with each bit >



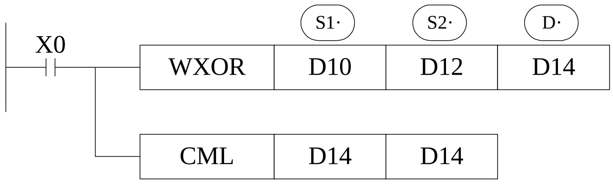
0 or 0=0 0 or 1=1
1 or 0=1 1 or 1=1

< Execute logic Exclusive OR operation with each bit >



0 xor 0=0 0 xor 1=1
1 xor 0=1 1 xor 1=0

If use this instruction along with CML instruction, XOR NOT operation could also be executed.



4-6-8 Converse [CML]

1: Summary

Converse the phase of the numbers

Converse [CML]			
16 bits	CML	32 bits	DCML
Execution condition	Normally ON/OFF, rising/falling edge	Suitable Models	XC1.XC2.XC3.XC5.XCM
Hardware requirement	-	Software requirement	-

2: Operands

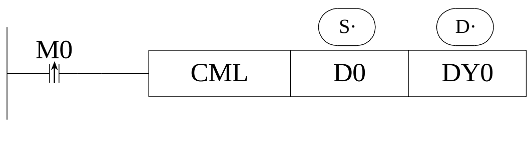
Operands	Function	Data Type
S	Source number address	16 bits/32 bits, BIN
D	Result address	16 bits/32 bits, BIN

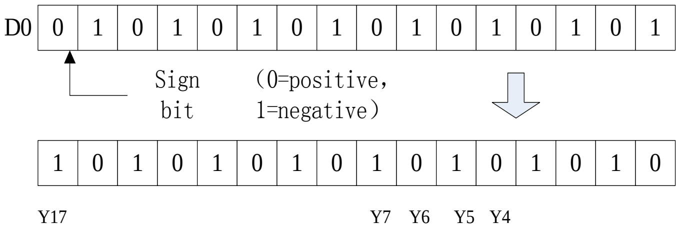
3: Suitable soft components

Word

Operands	System									Constant	Module	
	D	FD	ED	TD	CD	DX	DY	DM	DS	K/H	ID	QD
S1	●	●		●	●	●	●	●	●	●		
D	●			●	●		●	●	●			

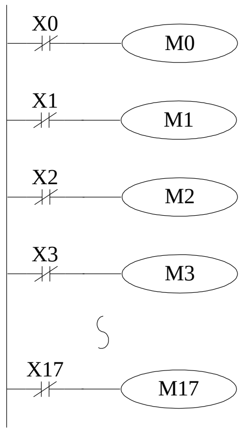
Description



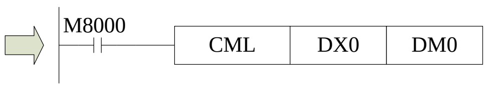


27. Each data bit in the source device is inverted （1→0，0→1） and sent to the destination device. If use constant K in the source device, it can be auto convert to be binary.
28. It’s available when you want to inverted output the PLC’s output

< Reading of inverted input >



The sequential control instruction in the left could be denoted by the following CML instruction.



4-6-9 Negative [NEG]

1: Summary

Get the negative number

Negative [NEG]			
16 bits	NEG	32 bits	DNEG
Execution condition	Normally ON/OFF, rising/falling edge	Suitable Models	XC1.XC2.XC3.XC5.XCM
Hardware requirement	-	Software requirement	-

2: Operands

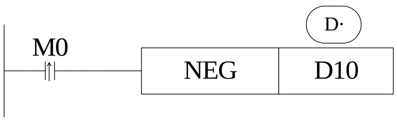
Operands	Function	Data Type
D	The source number address	16 bits/ bits, BIN

3: Suitable soft components

Word

Operands	System									Constant	Module	
	D	FD	ED	TD	CD	DX	DY	DM	DS	K/H	ID	QD
D	●			●	●		●	●	●			

Description



$(\overline{D10})\ +1 \longrightarrow (D10)$

29. The bit format of the selected device is inverted, I.e. any occurrence of a “1’ becomes a “0” and any occurrence of “0” becomes “1”, when this is complete, a further binary 1 is added to the bit format. The result is the total logic sigh change of the selected devices contents.

30.

4-7 Shift Instructions

Mnemonic	Function	Chapter

SHL	Arithmetic shift left	4-7-1
SHR	Arithmetic shift right	4-7-1
LSL	Logic shift left	4-7-2
LSR	Logic shift right	4-7-2
ROL	Rotation left	4-7-3
ROR	Rotation right	4-7-3
SFTL	Bit shift left	4-7-4
SFTR	Bit shift right	4-7-5
WSFL	Word shift left	4-7-6
WSFR	Word shift right	4-7-7

4-7-1 Arithmetic shift left [SHL], Arithmetic shift right [SHR]

1: Summary

Do arithmetic shift left/right for the numbers

Arithmetic shift left [SHL]

16 bits	SHL	32 bits	DSHL
Execution condition	Normally ON/OFF, rising/falling edge	Suitable Models	XC2.XC3.XC5.XCM
Hardware requirement	-	Software requirement	-
Arithmetic shift right [SHR]			
16 bits	SHR	32 bits	DSHR
Execution condition	Normally ON/OFF, rising/falling edge	Suitable Models	XC2.XC3.XC5.XCM
Hardware requirement	-	Software requirement	-

2: Operands

Operands	Function	Data Type
D	The source data address	16bit/32bit,BIN
n	Shift left or right times	16bit/32bit,BIN

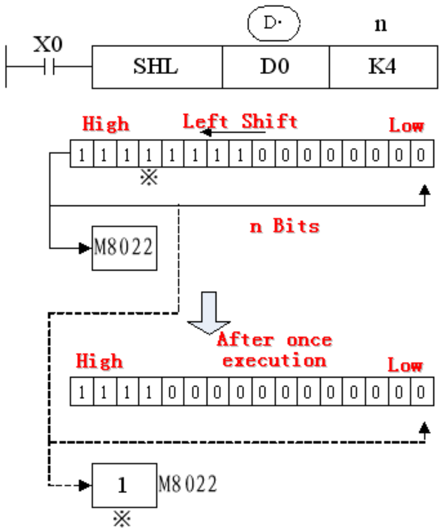
3: Suitable soft components

Word	Operands	System									Constant	Module	
		D	FD	ED	TD	CD	DX	DY	DM	DS	K/H	ID	QD
	D	●			●	●		●	●	●			
	n										●		

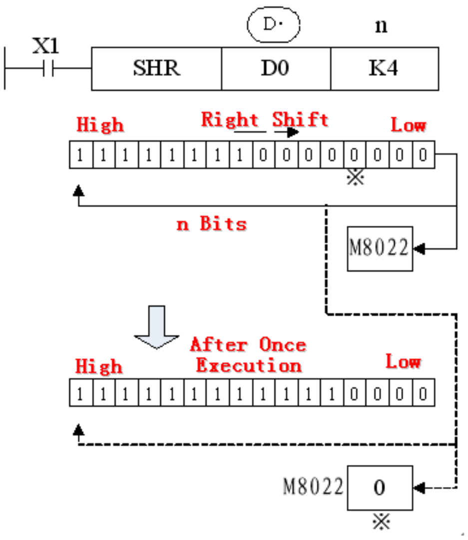
Description

- After once execution, the low bit is filled in 0, the final bit is stored in carry flag.
- After once execution, the high bit is same with the bit before shifting, the final bit is stored in carry flag.

< Arithmetic shift left >



< Arithmetic shift right >



4-7-2 Logic shift left [LSL], Logic shift right [LSR]

1: Summary

Do logic shift right/left for the numbers

Logic shift left [LSL]			
16 bits	LSL	32 bits	DLSL

Execution condition	Normally ON/OFF, rising/falling edge	Suitable Models	XC2.XC3.XC5.XCM
Hardware requirement	-	Software requirement	-
Logic shift right [LSR]			
16 bits	LSR	32 bits	DLSR
Execution condition	Normally ON/OFF, rising/falling edge	Suitable Models	XC2.XC3.XC5.XCM
Hardware requirement	-	Software requirement	-

2. Operands

Operands	Function	Data Type
D	Source data address	16 bits/32 bits, BIN
n	Arithmetic shift left/right times	16 bits/32bits, BIN

3. Suitable soft components

Word

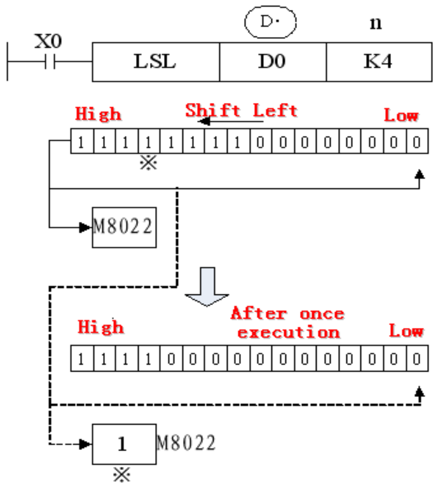
Operands	System									Constant	Module	
	D	FD	ED	TD	CD	DX	DY	DM	DS	K/H	ID	QD
D	•			•	•		•	•	•			
n										•		

Description

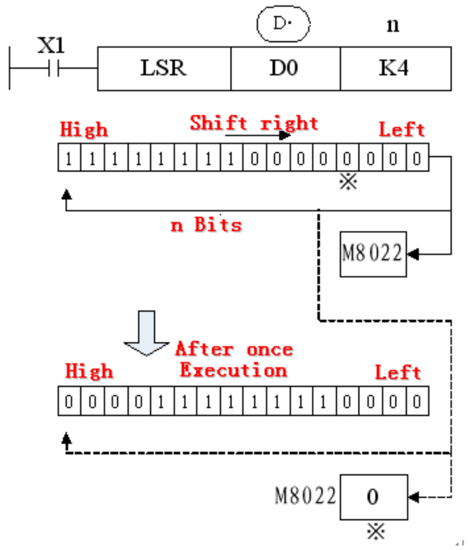
- After once execution, the low bit is filled in 0, the final bit is stored in carry flag.
- LSL meaning and operation are the same as SHL.
- After once execution, the high bit is same with the bit before shifting, the final bit is stored in carry flag。

- LSR and SHR is different, LSR add 0 in high bit when moving, SHR all bits are moved.

< Logic shift left >



< Logic shift right >



4-7-3. Rotation shift left **[ROL]** , Rotation shift right **[ROR]**

1: Summary

Continue and cycle shift left or right

Rotation shift left [ROL]			
16 bits	ROL	32 bits	DROL
Execution condition	Normally ON/OFF, rising/falling edge	Suitable Models	XC2.XC3.XC5.XCM

Hardware requirement	-	Software requirement	-
Rotation shift right [ROR]			
16 bits	ROR	32 bits	DROR
Execution condition	Normally ON/OFF, rising/falling edge	Suitable Models	XC2.XC3.XC5.XCM
Hardware requirement	-	Software requirement	-

2: Operands

Operands	Function	Data Type
D	Source data address	16 bits/32 bits, BIN
n	Shift right or left times	16 bits/32 bits, BIN

3: Suitable soft components

Word

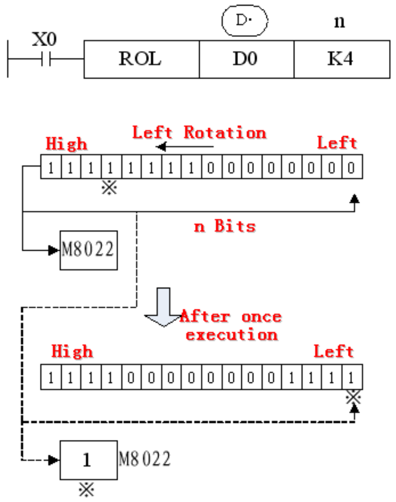
Operands	System									Constant	Module	
	D	FD	ED	TD	CD	DX	DY	DM	DS	K/H	ID	QD
D	•			•	•		•	•	•			
n										•		

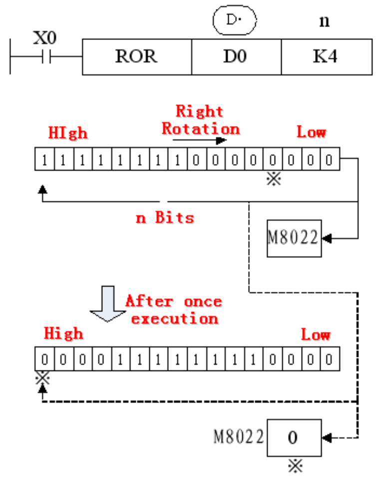
Description

- The bit format of the destination device is rotated in bit places to the left on every operation of the instruction.

< Rotation shift left >

< Rotation shift right >





4-7-4 Bit shift left [SFTL]

1: Summary

Bit shift left

Bit shift left [SFTL]			
16 bits	SFTL	32 bits	DSFTL

Execution condition	Normally ON/OFF, rising/falling edge	Suitable Models	XC2.XC3.XC5.XCM
Hardware requirement	-	Software requirement	-

2: Operands

Operands	Function	Types
S	Source soft element head address	bit
D	Target soft element head address	bit
n1	Source data quantity	16 bits /32 bits, BIN
n2	Shift left times	16 bits/32 bits, BIN

3. Suitable soft components

Word

Operands	System									Constant	Module	
	D	FD	ED	TD	CD	DX	DY	DM	DS	K/H	ID	QD
n1	●			●	●	●	●	●	●	●		
n2	●			●	●	●	●	●	●	●		

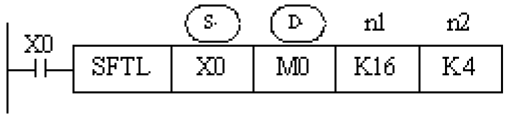
Bit

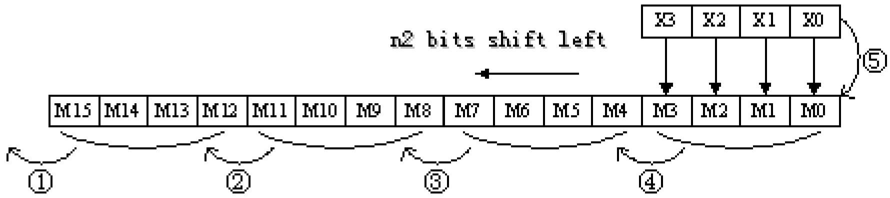
Operands	System						
	X	Y	M	S	T	C	Dn.m
S	●	●	●	●	●	●	
D		●	●	●	●	●	

Description

(2) The instruction copies n2 source devices to a bit stack of length n1. For every new addition of n2 bits, the existing data within the bit stack is shifted n2 bits to the left/right. Any bit data moving to the position exceeding the n1 limit is diverted to an overflow area.

(3) In every scan cycle, loop shift left action will be executed





① M15~M12→Overflow

② M11~M 8→M15~M 12

③ M 7~M 4→M11~M8

④ M 3~M 0→M7~M4

⑤ X 3~X 0→M3~M0

4-7-5 Bit shift right [SFTR]

1: Summary

Bit shift right

Bit shift right [SFTR]			
16 bits	SFTR	32 bits	DSFTR
Execution condition	rising/falling edge	Suitable Models	XC2.XC3.XC5.XCM

Hardware requirement	-	Software requirement	-

2: Operands

Operands	Function	Data Type
S	Source soft element head address	bit
D	Target soft element head address	bit
n1	Source data quantity	16 bits/32 bits, BIN
n2	Shift right times	16 bits/32 bits, BIN

3: Suitable soft components

Word

Operands	System									Constant	Module	
	D	FD	ED	TD	CD	DX	DY	DM	DS	K/H	ID	QD
n1	●			●	●	●	●	●	●	●		
n2	●			●	●	●	●	●	●	●		

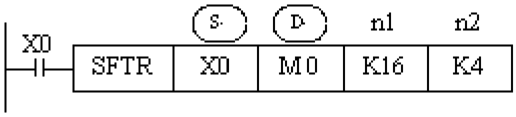
Bit

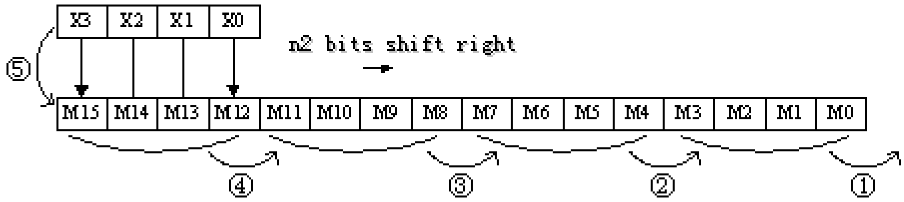
Operands	System						
	X	Y	M	S	T	C	Dn.m
S	●	●	●	●	●	●	
D		●	●	●	●	●	

Description

(4) The instruction copies n2 source devices to a bit stack of length n1. For every new addition of n2 bits, the existing data within the bit stack is shifted n2 bits to the left/right. Any bit data moving to the position exceeding the n1 limit is diverted to an overflow area.

(5) In every scan cycle, loop shift right action will be executed





① M 3~M 0→Overflow

② M 7~M 4→M3~M0

③ M11~M 8→M7~M4

④ M15~M12→M11~M8

⑤ X 3~X 0→M15~M12

4-7-6 Word shift left [WSFL]

1: Summary

Word shift left

Word shift left [[WSFL]			
16 bits	WSFL	32 bits	-
Execution condition	rising/falling edge	Suitable Models	XC2.XC3.XC5.XCM
Hardware	-	Software	-

requirement		requirement	

2: Operands

Operands	Function	Data Type
S	Source soft element head address	16 bits/32 bits, BIN
D	Target soft element head address	16 bits /32 bits, BIN
n1	Source data quantity	16 bits /32 bits, BIN
n2	Word shift left times	16 bits /32 bits, BIN

3: Suitable soft components

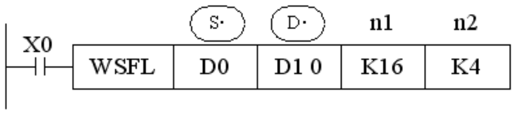
Word

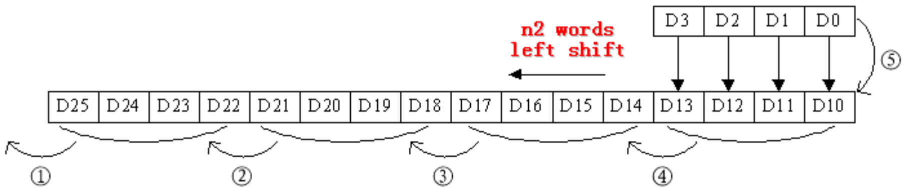
Operands	System									Constant	Module	
	D	FD	ED	TD	CD	DX	DY	DM	DS	K/H	ID	QD
S	●	●		●	●	●	●	●	●			
D	●			●	●		●	●	●			
n1	●			●	●		●	●	●	●		
n2	●			●	●		●	●	●	●		

Description

- The instruction copies n2 source devices to a word stack of length n1. For each addition of n2 words, the existing data within the word stack is shifted n2 words to the left. Any word data moving to a position exceeding the n1 limit is diverted to an overflow area.
- In every scan cycle, loop shift left action will be executed.

n2 word shift left





① D25~D22→Overflow

② D21~D18→D25~D22

③ D17~D14→D21~D18

④ D13~D10→D17~D14

⑤ D 3~D 0→D13~D10

4-7-7 Word shift right[WSFR]

1: Summary

Word shift right

Word shift right [WSFR]			
16 bits	WSFR	32 bits	-
Execution condition	rising/falling edge	Suitable Models	XC2.XC3.XC5.XCM
Hardware requirement	-	Software requirement	-

2: Operands

Operands	Function	Data Type
S	Source soft element head address	16 bits/32 bits, BIN
D	Target soft element head address	16 bits/32 bits, BIN
n1	Source data quantity	16 bits/32 bits, BIN
n2	Shift right times	16 bits/32 bits, BIN

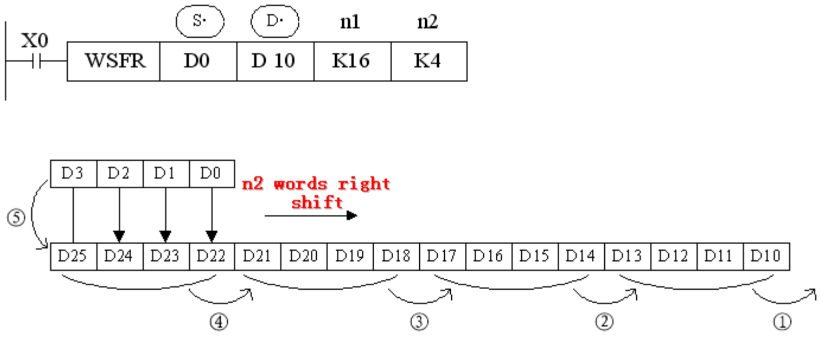
3: Suitable soft components

Word

Operands	System									Constant	Module	
	D	FD	ED	TD	CD	DX	DY	DM	DS	K/H	ID	QD
S	●	●		●	●	●	●	●	●			
D	●			●	●		●	●	●			
n1	●			●	●		●	●	●	●		
n2	●			●	●		●	●	●	●		

Description

- The instruction copies n2 source devices to a word stack of length n1. For each addition of n2 words, the existing data within the word stack is shifted n2 words to the right. Any word data moving to a position exceeding the n1 limit is diverted to an overflow area.
- In every scan cycle, loop shift right action will be executed



① D13~D10→Overflow

② D17~D14→D13~D10

③ D21~D18→D17~D14

④ D25~D22→D21~D18

⑤ D 3~D 0→D25~D22

4-8 Data Convert

Mnemonic	Function	Chapter
WTD	Single word integer converts to double word integer	4-8-1
FLT	16 bits integer converts to float point	4-8-2
DFLT	32 bits integer converts to float point	4-8-2
FLTD	64 bits integer converts to float point	4-8-2
INT	Float point converts to integer	4-8-3

BIN	BCD convert to binary	4-8-4
BCD	Binary converts to BCD	4-8-5
ASCI	Hex. converts to ASCII	4-8-6
HEX	ASCII converts to Hex.	4-8-7
DECO	Coding	4-8-8
ENCO	High bit coding	4-8-9
ENCOL	Low bit coding	4-8-10

4-8-1 Single word integer converts to double word integer [WTD]

1: Summary

Single word integer converts to double word integer [WTD]			
16 bits	WTD	32 bits	-
Execution condition	Normally ON/OFF, rising/falling edge	Suitable Models	XC2.XC3.XC5.XCM
Hardware requirement	-	Software requirement	-

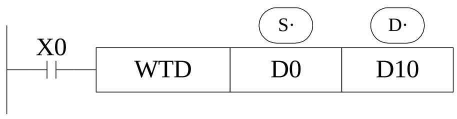
2: Operands

Operands	Function	Data Type
S	Source soft element address	16 bits, BIN
D	Target soft element address	32 bits, BIN

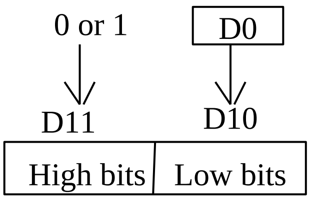
3: Suitable soft components

Word	Operands	System								Constant	Module	
		D	FD	ED	TD	CD	DX	DY	DM	DS	K/H	ID	QD
	S	●	●		●	●	●	●	●	●			
	D	●			●	●		●	●	●			

Description



（D0） → （D11，D10）
Single Word Double



- When single word D0 is positive integer, after executing this instruction, the high bit of double word D10 is 0.
- When single word D0 is negative integer, after executing this instruction, the high bit of double word D10 is 1.

4-8-2 16 bits integer converts to float point [FLT]

1: Summary

16 bits integer converts to float point [FLT]					
16 bits	FLT	32 bits	DFLT	64 bits	FLTD
Execution condition	Normally ON/OFF, rising/falling edge		Suitable Models	XC2.XC3.XC5.XCM	
Hardware requirement	-		Software requirement	-	

2: Operands

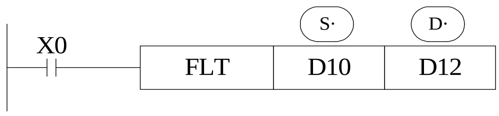
Operands	Function	Data Type
S	Source soft element address	16 bits/32 bits/64 bits,BIN
D	Target soft element address	32 bits/64 bits,BIN

3: Suitable soft components

Word	Operands	System									Constant	Module	
		D	FD	ED	TD	CD	DX	DY	DM	DS	K/H	ID	QD
	S	●	●								●		
	D	●											

Description

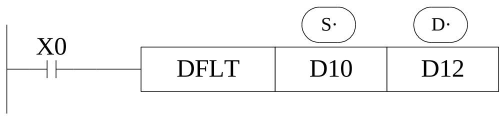
<16 bits>



（D10） → (D13,D12)

BIN integer　Binary float point

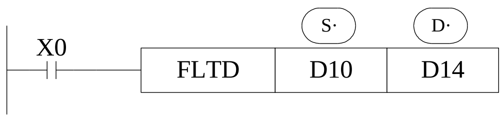
<32 bits >



（D11,D10）→ （D13,D12）

BIN integer　Binary float point

<64 bits>



（D13,D12,D11,D10）→ （D17,D16,D15,D14）

BIN integer　Binary float point

- Convert BIN integer to binary float point. As the constant K ,H will auto convert by the float operation instruction, so this FLT instruction can’t be used.
- The instruction is contrary to INT instruction

4-8-3　Float point converts to integer [INT]

1: Summary

Float point converts to integer [INT]			
16 bits	INT	32 bits	DINT
Execution condition	Normally ON/OFF, rising/falling edge	Suitable Models	XC2.XC3.XC5.XCM
Hardware requirement	-	Software requirement	-

2: Operands

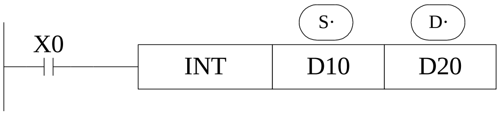
Operands	Function	Data Type
S	Source soft element address	16 bits/32 bits, BIN
D	Target soft element address	16 bits/32 bits, BIN

3: Suitable soft components

Word	Operands	System									Constant	Module	
		D	FD	ED	TD	CD	DX	DY	DM	DS	K/H	ID	QD
	S	●	●										
	D	●											

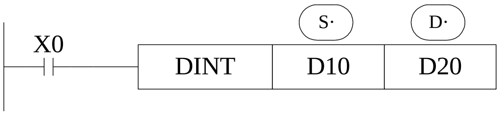
Description

<16 bits>



（D11,D10） → (D20)
Binary Float BIN integer
Give up the data after the decimal dot

<32 bits>



（D11,D10） → (D20,D21)
Binary Float BIN integer
Give up the data after the decimal dot

- The binary source number is converted into a BIN integer and stored at the destination device. Abandon the value behind the decimal point.
- This instruction is contrary to FLT instruction.
- When the result is 0, the flag bit is ON

  When converting, less than 1 and abandon it, zero flag is ON.

  The result is over below data, the carry flag is ON.

  16 bits operation: -32,768~32,767

  32 bits operation: -2,147,483,648~2,147,483,647

4-8-4 BCD convert to binary [BIN]

1: Summary

BCD convert to binary [BIN]			
16 bits	BIN	32 bits	-
Execution condition	Normally ON/OFF, rising/falling edge	Suitable Models	XC2.XC3.XC5.XCM
Hardware requirement	-	Software requirement	-

2: Operands

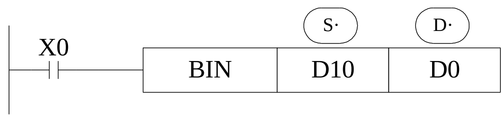
Operands	Function	Data Type
S	Source soft element address	BCD
D	Target soft element address	16 bits/32 bits, BIN

3: Suitable soft components

Word	Operands	System									Constant	Module	
		D	FD	ED	TD	CD	DX	DY	DM	DS	K/H	ID	QD
	S	●	●		●	●	●	●	●	●			
	D	●			●	●		●	●	●			

Description

Convert and move instruction of Source (BCD) → destination (BIN)



- When source data is not BCD code, M8067（Operation error）, M8004 (error occurs)
- As constant K automatically converts to binary, so it's not suitable for this instruction.

4-8-5 Binary convert to BCD [BCD]

1: Summary

Binary convert to BCD [BCD]			
16 bits	BCD	32 bits	-
Execution condition	Normally ON/OFF, rising/falling edge	Suitable Models	XC2.XC3.XC5.XCM
Hardware requirement	-	Software requirement	-

2: Operands

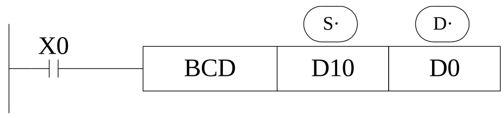
Operands	Function	Data Type
S	Source soft element address	16 bits/32 bits, BIN
D	Target soft element address	BCD code

3: Suitable soft components

	Operands	System									Constant	Module	
Word		D	FD	ED	TD	CD	DX	DY	DM	DS	K/H	ID	QD
	S	●	●		●	●	●	●	●	●			
	D	●			●	●		●	●	●			

Description

Convert and move instruction of source (BIN)→destination (BCD)



- This instruction can be used to output data directly to a seven-segment display.

4-8-6 Hex. converts to ASCII [ASCI]

1: Summary

Hex. convert to ASCII [ASCI]			
16 bits	ASCI	32 bits	-
Execution condition	Normally ON/OFF, rising/falling edge	Suitable Models	XC2.XC3.XC5.XCM
Hardware requirement	-	Software requirement	-

2: Operands

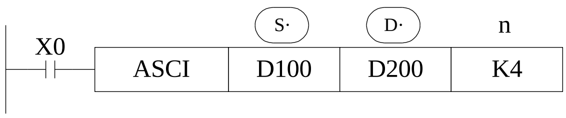
Operands	Function	Data Type
S	Source soft element address	2 bits, HEX
D	Target soft element address	ASCII code
n	Transform character quantity	16 bits, BIN

3: Suitable soft components

Word

Operands	System									Constant	Module	
	D	FD	ED	TD	CD	DX	DY	DM	DS	K/H	ID	QD
S	●	●		●	●	●	●	●	●			
D	●			●	●		●	●	●			
n	●			●	●		●	●	●	●		

Description



Convert each bit of source’s (S) Hex. format data to be ASCII code, move separately to the high 8 bits and low 8 bits of destination (D). The convert alphanumeric number is assigned with n.

D· is low 8 bits, high 8 bits, store ASCII data.

The converted result is this

Assign start device：
(D100)=0ABCH
(D101)=1234H
(D102)=5678H

[0]=30H [1]=31H
[5]=35H [A]=41H
[2]=32H [6]=36H
[B]=42H [3]=33H
[7]=37H [C]=43H
[4]=34H [8]=38H

n D	K1	K2	K3	K4	K5	K6	K7	K8	K9
D200 down	[C]	[B]	[A]	[0]	[4]	[3]	[2]	[1]	[8]
D200 up		[C]	[B]	[A]	[0]	[4]	[3]	[2]	[1]
D201 down			[C]	[B]	[A]	[0]	[4]	[3]	[2]
D201 up				[C]	[B]	[A]	[0]	[4]	[3]
D202 down					[C]	[B]	[A]	[0]	[4]
D202 up						[C]	[B]	[A]	[0]
D203 down							[C]	[B]	[A]
D203 up								[C]	[B]
D204 down									[C]

4-8-7 ASCII convert to Hex.[HEX]

1: Summary

ASCII converts to Hex. [HEX]			
16 bits	HEX	32 bits	-
Execution condition	Normally ON/OFF, rising/falling edge	Suitable Models	XC2.XC3.XC5.XCM
Hardware requirement	-	Software requirement	-

2: Operands

	Function	Date type

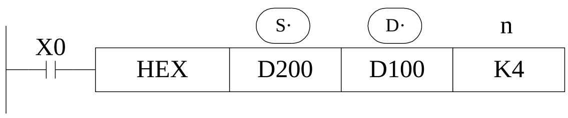
Operands		
S	Source soft element address	ASCII
D	Target soft element address	2 bits, HEX
n	Character quantity	16 bits, BIN

3: Suitable soft components

Word

Operands	System									Constant	Module	
	D	FD	ED	TD	CD	DX	DY	DM	DS	K/H	ID	QD
S	•	•		•	•	•	•	•	•			
D	•			•	•		•	•	•			
n										•		

Description



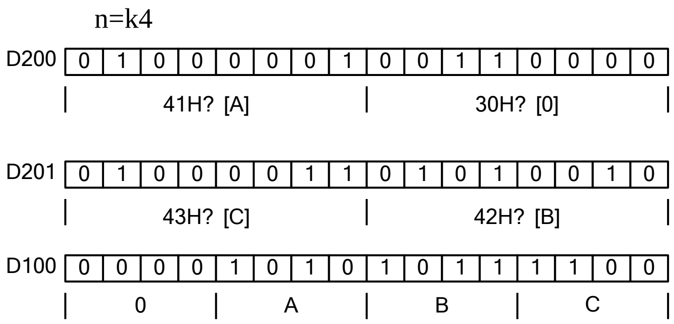
Convert the high and low 8 bits in source (S·) to HEX data. Move 4 bits every time to destination (D·) . The convert alphanumeric number is assigned by n.

The completed conversion of the above program is the following：

(S·)	ASCII Code	HEX Convert
D200 down	30H	0
D200 up	41H	A
D201	42H	B

n (D·)	D102	D101	D100
1	Not change to be 0		···0H
2			··0AH
3			·0ABH
4			0ABC

时



4-8-8 Coding [DECO]

1: Summary

Transform the ASCII code to Hex numbers.

Coding [DECO]			
16 bits	DECO	s	-
Execution condition	Normally ON/OFF, rising/falling edge	Suitable Models	XC2.XC3.XC5.XCM
Hardware requirement	-	Software requirement	-

2: Operands

Operands	Function	Data Type
S	Source soft element address	ASCII

D	Target soft element address	2 bits HEX
n	The coding soft element quantity	16bits, BIN

3: Suitable soft components

Word

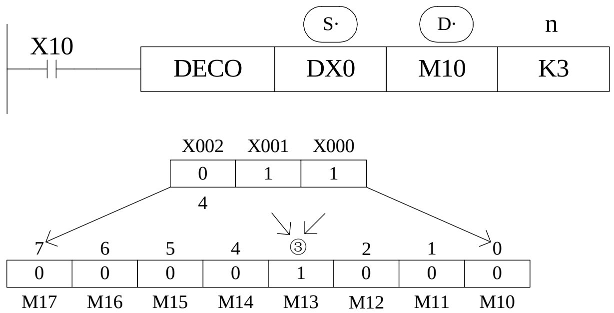
Operands	System									Constant	Module	
	D	FD	ED	TD	CD	DX	DY	DM	DS	K/H	ID	QD
S	●	●		●	●	●	●	●	●			
n										●		

Bit

Operands	System						
	X	Y	M	S	T	C	Dn.m
D	●	●	●	●	●	●	

Description

< When D· is bit unit > n≤16



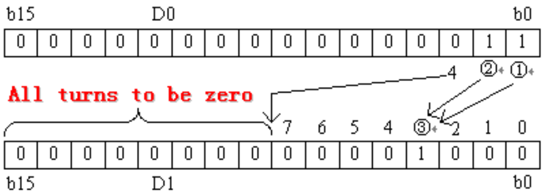
- The source address is 1+2=3，starts from M10, the number 3 bit (M13) is 1. If the source are all 0, M10 is 1.
- When n=0, no operation, beyond n=0~16, don’t execute the instruction.
- When n=16, if coding command is soft unit, it’s point is 2^16=65536.
- When drive input is OFF, instructions are not executed, the activate coding output keep on activate.

D·

< When D· is word device > n≤4

X0

	S·	D·	n
DECO	D0	D1	K3



- Low n bits(n≤4) of source address is decoded to target address. n≤3, the high bit of target address all become 0.
- When n=0, no operation, beyond n=0~14, don't execute the instruction.

4-8-9 High bit coding [ENCO]

1: Summary

Transform the ASCII code to hex numbers

High bit coding [ENCO]			
16 bits	ENCO	32 bits	-
Execution condition	Normally ON/OFF, rising/falling edge	Suitable Models	XC2.XC3.XC5.XCM
Hardware requirement	-	Software requirement	-

2: Operands

Operands	Function	Data Type
S	data address need coding	16 bits, BIN; bit
D	Coding result address	16 bits, BIN

n	soft element quantity to save result	16 bits, BIN

3: Suitable soft components

Word

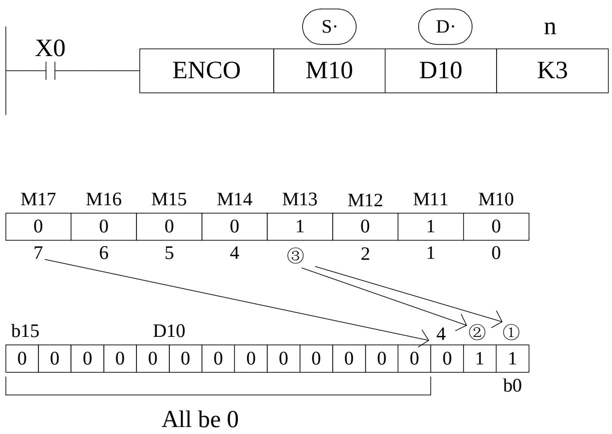
Operands	System									Constant	Module	
	D	FD	ED	TD	CD	DX	DY	DM	DS	K/H	ID	QD
S	●	●		●	●	●	●	●	●			
D	●			●	●		●	●	●			
n										●		

Bit

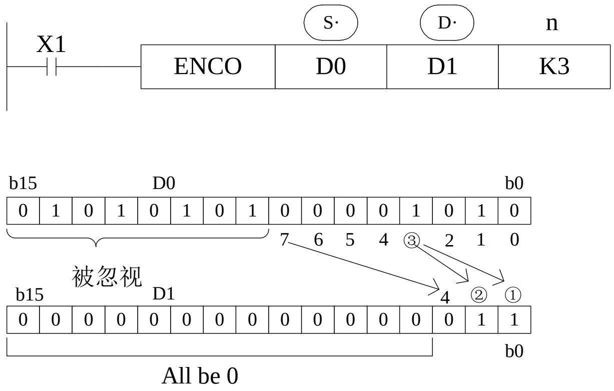
Operands	System						
	X	Y	M	S	T	C	Dn.m
S	●	●	●	●	●	●	

Description

< When S· is bit device > n≤16



< When S· is word device > n≤4



- If many bits in the source ID are 1, ignore the low bits. If source ID are all 0, don't execute the instructions.
- When drive input is OFF, the instruction is not executed, encode output doesn't change.
- When n=8, if encode instruction's "S" is bit unit, it's point number is 2^8=256

4-8-10 Low bit coding [ENCOL]

1: Summary

Transform the ASCII to hex numbers.

Low bit coding [ENCOL]			
16 bits	ENCOL	32 bits	-
Execution condition	Normally ON/OFF, rising/falling edge	Suitable Models	XC2.XC3.XC5.XCM
Hardware requirement	-	Software requirement	-

2: Operands

Operands	Function	Data Type
S	Soft element address need coding	16bit,BIN；bit
D	Soft element address to save coding result	16bit,BIN
n	The soft element quantity to save result	16bit,BIN

3: Suitable soft components

Word

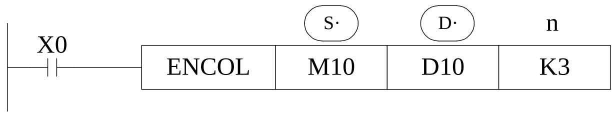
Operands	System									Constant	Module	
	D	FD	ED	TD	CD	DX	DY	DM	DS	K/H	ID	QD
S	●	●		●	●	●	●	●	●			
D	●			●	●		●	●	●			
n										●		

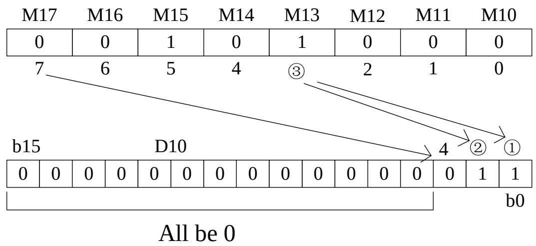
Bit

Operands	System						
	X	Y	M	S	T	C	Dn.m
S	●	●	●	●	●	●	

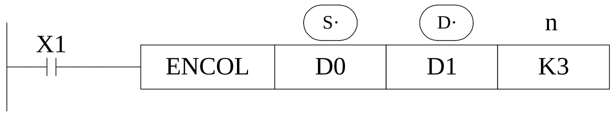
Description

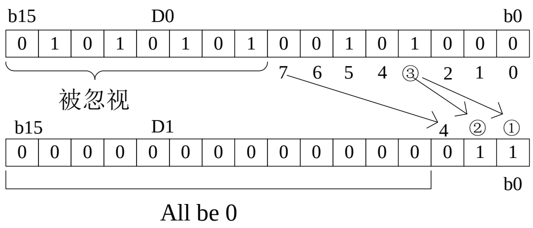
<if S· is bit device > n≤16





< if S· is word device> n≤4





- If many bits in the source ID are 1, ignore the high bits. If source ID are all 0, don't execute the instructions。
- When drive input is OFF, the instruction is not executed, encode output don't change
- When n=8, if encode instruction's S· is bit unit, it's point number is 2^8=256

4-9 Floating Operation

Mnemonic	Function	Chapter
ECMP	Float Compare	4-9-1
EZCP	Float Zone Compare	4-9-2
EADD	Float Add	4-9-3
ESUB	Float Subtract	4-9-4
EMUL	Float Multiplication	4-9-5
EDIV	Float Division	4-9-6
ESQR	Float Square Root	4-9-7
SIN	Sine	4-9-8
COS	Cosine	4-9-9
TAN	Tangent	4-9-10
ASIN	ASIN	4-9-11
ACOS	ACOS	4-9-12
ATAN	ATAN	4-9-13

4-9-1 Float Compare [ECMP]

1: Summary

Float Compare [ECMP]			
16 bits	-	32 bits	ECMP
Execution condition	Normally ON/OFF, rising/falling edge	Suitable Models	XC2.XC3.XC5.XCM
Hardware requirement	-	Software requirement	-

2: Operands

Operands	Function	Data Type
S1	Soft element address need compare	32 bits, BIN
S2	Soft element address need compare	32 bits, BIN
D	Compare result	bit

3: Suitable soft components

Word

F

Operands	System									Constant	Module	
	D	FD	ED	TD	CD	DX	DY	DM	DS	K/H	ID	QD
S1	●	●				●	●	●	●	●		
S2	●	●				●	●	●	●	●		

V1

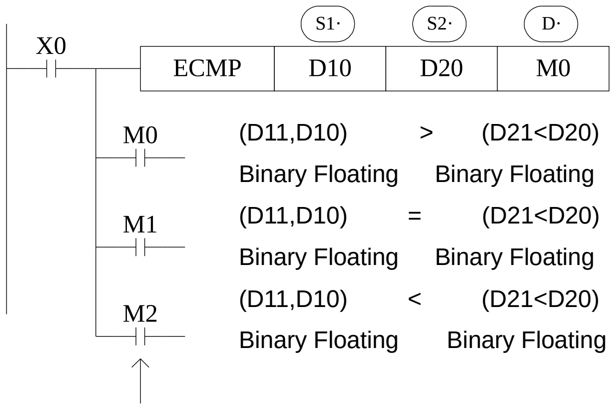
Bit

Operands	System						
	X	Y	M	S	T	C	Dn.m
D		●	●	●			

Description

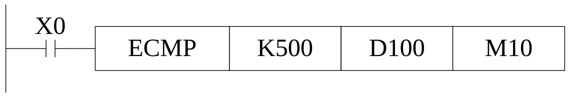
（D11,D10） ： （D21,D20）→ M0,M1,M2

Binary Floating Binary Floating



The status of the destination device will be kept even if the ECMP instruction is deactivated.

- The binary float data of S1 is compared to S2. The result is indicated by 3 bit devices specified with the head address entered as D
- If a constant K or H used as source data, the value is converted to floating point before the addition operation.



（K500）:　　（D101，D100）→M10,M11,M12

Binary converts to floating　　Binary floating

4-9-2　Float Zone Compare [EZCP]

1: Summary

Float Zone Compare [EZCP]			
16 bits	-	32 bits	EZCP
Execution condition	Normally ON/OFF, rising/falling edge	Suitable Models	XC2.XC3.XC5.XCM
Hardware requirement	-	Software requirement	-

2: Operands

Operands	Function	Data Type
S1	Soft element address need compare	32 bits, BIN
S2	Upper limit of compare data	32 bits, BIN
S3	Lower limit of compare data	32 bits, BIN
D	The compare result soft element address	bit

3: Suitable soft components

Word

Operands	System									Constant	Module	
	D	FD	ED	TD	CD	DX	DY	DM	DS	K/H	ID	QD
S1	●	●				●	●	●	●	●		
S2	●	●				●	●	●	●	●		
S3	●	●				●	●	●	●	●		

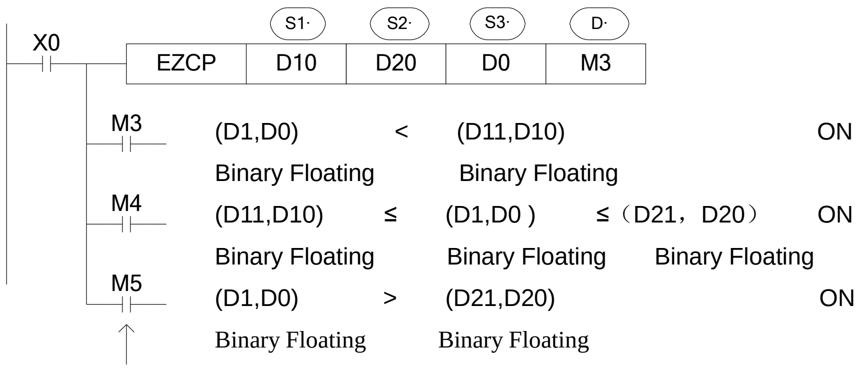
V1

Bit

Operands	System					
	X	Y	M	S	T	C	Dn.m
D		●	●	●			

Description

Compare a float range with a float value:



The status of the destination device will be kept even if the EZCP instruction is deactivated.

- The data of S1 is compared to the data of S2. The result is indicated by 3 bit devices specified with the head address entered as D.
- If a constant K or H used as source data, the value is converted to floating point before the addition operation.

X0	EZCP	K10	K2800	D5	M0

（K10）:　　[D6,D5]:　　（K2800）→　M0，M1，M2

Binary converts to Floating　　Binary Floating　　Binary converts to Floating

Please set S1<S2, when S2>S1, see S2 as the same with S1 and compare them

4-9-3 Float Add[EADD]

1: Summary

Float Add [EADD]			
16 bits	-	32 bits	EADD
Execution condition	Normally ON/OFF, rising/falling edge	Suitable Models	XC2.XC3.XC5.XCM
Hardware requirement	-	Software requirement	-

2: Operands

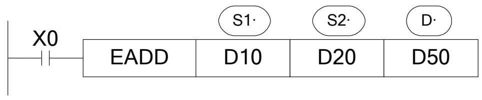
Operands	Function	Data Type
S1	Soft element address need to add	32 bits, BIN
S2	Soft element address need to add	32 bits, BIN
D	Result address	32 bits, BIN

3: Suitable soft components

Word

Operands	System									Constant	Module	
	D	FD	ED	TD	CD	DX	DY	DM	DS	K/H	ID	QD
S1	●	●				●	●	●	●	●		
S2	●	●				●	●	●	●	●		
D	●						●	●	●			

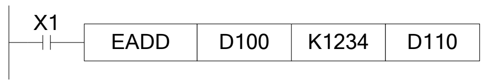
Description



（D11,D10） + (D21,D20) → (D51,D50)

Binary Floating Binary Floating Binary Floating

- The floating point values stored in the source devices S1 and S2 are algebraically added and the result stored in the destination device D.
- If a constant K or H used as source data, the value is converted to floating point before the addition operation.



（K1234） + (D101,D100) → (D111,D110)

Binary converts to Floating Binary Floating Binary Floating

- The same device may be used as a source and as the destination. If this is the case then, on continuous operation of the EADD instruction, the result of the previous operation will be used as a new source value and a new result calculated. This will happen in every program scan unless the pulse modifier or an interlock program is used.

4-9-4 Float Sub[ESUB]

1: Summary

Float Sub [ESUB]			
16 bits	-	32 bits	ESUB
Execution condition	Normally ON/OFF, rising/falling edge	Suitable Models	XC2.XC3.XC5.XCM
Hardware requirement	-	Software requirement	-

2: Operands

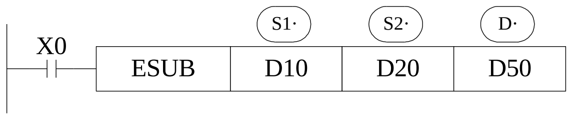
Operands	Function	Data Type
S1	Soft element address need to subtract	32 bits, BIN
S2	Soft element address need to subtract	32 bits, BIN
D	Result address	32 bits, BIN

3: Suitable soft components

Word

Operands	System									Constant	Module	
	D	FD	ED	TD	CD	DX	DY	DM	DS	K/H	ID	QD
S1	●	●				●	●	●	●	●		
S2	●	●				●	●	●	●	●		
D	●						●	●	●			

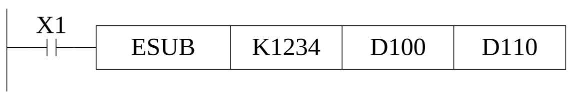
Description



(D11,D10) — (D21,D20) → (D51,D50)

Binary Floating Binary Floating Binary Floating

- The floating point value of S2 is subtracted from the floating point value of S1 and the result stored in destination device D.
- If a constant K or H used as source data, the value is converted to floating point before the addition operation.



（K1234） — (D101,D100) → (D111,D110)

Binary converts to Floating Binary Floating Binary Floating

- The same device may be used as a source and as the destination. If this is the case then, on continuous operation of the EADD instruction, the result of the previous operation will be used as a new source value and a new result calculated. This will happen in every program scan unless the pulse modifier or an interlock program is

4-9-5. Float Mul[EMUL]

1: Summary

Float Multiply [EMUL]			
16 bits	-	32 bits	EMUL
Execution condition	Normally ON/OFF, rising/falling edge	Suitable Models	XC2.XC3.XC5.XCM
Hardware requirement	-	Software requirement	-

2: Operands

Operands	Function	Data Type
S1	Soft element address need to multiply	32 bits, BIN
S2	Soft element address need to multiply	32 bits, BIN
D	Result address	32 bits, BIN

3: Suitable soft components

Word

Operands	System									Constant	Module	
	D	FD	ED	TD	CD	DX	DY	DM	DS	K/H	ID	QD
S1	●	●				●	●	●	●	●		
S2	●	●				●	●	●	●	●		
D	●						●	●	●			

Description

X0	EMUL	D10 (S1·)	D20 (S2·)	D50 (D·)

（D11，D10） × （D21,D20）→ （D51,D50）

Binary Floating　Binary Floating　Binary Floating

- The floating value of S1 is multiplied with the floating value point value of S2. The result of the multiplication is stored at D as a floating value.
- If a constant K or H used as source data, the value is converted to floating point before the addition operation.

X1	EMUL	K100	D100	D110

（K100） × （D101,D100） → （D111,D110）

Binary converts to Floating　Binary Floating　Binary Floating

4-9-6 Float Div[EDIV]

1: Summary

Float Divide [EDIV]			
16 bits	-	32 bits	EDIV
Execution condition	Normally ON/OFF, rising/falling edge	Suitable Models	XC2.XC3.XC5.XCM
Hardware requirement	-	Software requirement	-

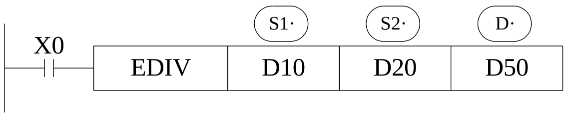
2: Operands

Operands	Function	Data Type
S1	Soft element address need to divide	32 bits, BIN
S2	Soft element address need to divide	32 bits, BIN
D	Result address	32 bits, BIN

3: Suitable soft components

word	Operands	System									Constant	Module	
		D	FD	ED	TD	CD	DX	DY	DM	DS	K/H	ID	QD
	S1	●	●				●	●	●	●	●		
	S2	●	●				●	●	●	●	●		
	D	●						●	●	●			

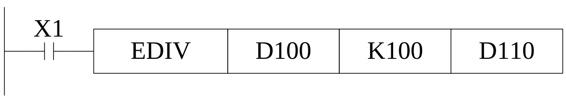
Description



（D11,D10） ÷ （D21,D20）→ （D51,D50）

Binary Floating Binary Floating Binary Floating

- The floating point value of S1 is divided by the floating point value of S2. The result of the division is stored in D as a floating point value. No remainder is calculated.
- If a constant K or H used as source data, the value is converted to floating point before the addition operation



（D101,D100） ÷ （K100） →（D111,D110）

Binary converts to Floating Binary Floating Binary Floating

NB: If S2 is 0, the calculate is error, the instruction can not work

4-9-7 Float Square Root [ESQR]

1: Summary

Float Square Root [ESQR]			
16 bits	-	32 bits	ESQR
Execution condition	Normally ON/OFF, rising/falling edge	Suitable Models	XC2.XC3.XC5.XCM
Hardware requirement	-	Software requirement	-

2: Operands

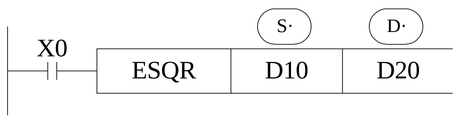
Operands	Function	Data Type
S	The soft element address need to do square root	32 bits, BIN
D	The result address	32 bits, BIN

3: Suitable soft components

Word

Operands	System									Constant	Module	
	D	FD	ED	TD	CD	DX	DY	DM	DS	K/H	ID	QD
S	●	●				●	●	●	●	●		
D	●						●	●	●			

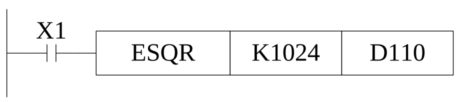
Description



（D11,D10） →（D21,D20）

Binary Floating Binary Floating

- A square root is performed on the floating point value in S the result is stored in D
- If a constant K or H used as source data, the value is converted to floating point before the addition operation.



（K1024） → （D111，D110）

Binary converts to Floating Binary Floating

- When the result is zero, zero flag activates.
- Only when the source data is positive will the operation be effective. If S is negative then an error occurs and error flag M8067 is set ON, the instruction can’t be executed.

4-9-8 Sine[SIN]

1: Summary

Float Sine[SIN]			
16 bits	-	32 bits	SIN
Execution condition	Normally ON/OFF, rising/falling edge	Suitable Models	XC2.XC3.XC5.XCM
Hardware requirement	-	Software requirement	-

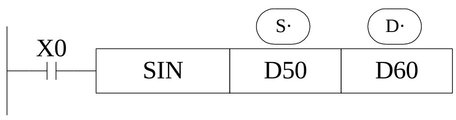
2: Operands

Operands	Function	Data Type
S	The soft element address need to do sine	32 bits, BIN
D	The result address	32 bits, BIN

3: Suitable soft components

Word	Operands	System									Constant	Module	
		D	FD	ED	TD	CD	DX	DY	DM	DS	K/H	ID	QD
	S	●	●				●	●	●	●	●		
	D	●						●	●	●			

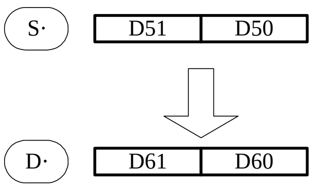
Description



(D51,D50) → (D61,D60)SIN

Binary Floating   Binary Floating

- This instruction performs the mathematical SIN operation on the floating point value in S (angle RAD). The result is stored in D.



RAD value (angle×π/180)

Assign the binary floating value

SIN value

Binary Floating

4-9-9 Cosine[SIN]

1: Summary

Float Cosine[COS]			
16 bits	-	32 bits	COS
Execution condition	Normally ON/OFF, rising/falling edge	Suitable Models	XC2.XC3.XC5.XCM
Hardware requirement	-	Software requirement	-

2: Operands

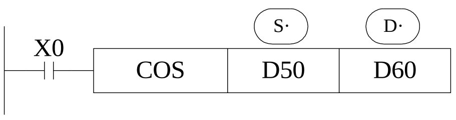
Operands	Function	Data Type
S	Soft element address need to do cos	32 bits, BIN
D	Result address	32 bits, BIN

3: Suitable soft components

Word

Operands	System									Constant	Module	
	D	FD	ED	TD	CD	DX	DY	DM	DS	K/H	ID	QD
S	●	●				●	●	●	●	●		
D	●						●	●	●			

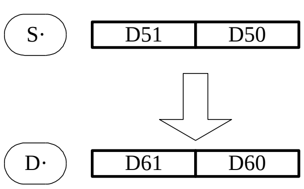
Description



(D51,D50)RAD → (D61,D60)COS

Binary Floating Binary Floating

- This instruction performs the mathematical COS operation on the floating point value in S (angle RAD). The result is stored in D.



RAD value (angle×π/180)

Assign the binary floating value

COS value

Binary Floating

4-9-10 TAN [TAN]

1: Summary

TAN [TAN]			
16 bits	-	32 bits	TAN
Execution condition	Normally ON/OFF, rising/falling edge	Suitable Models	XC2.XC3.XC5.XCM
Hardware requirement	-	Software requirement	-

2: Operands

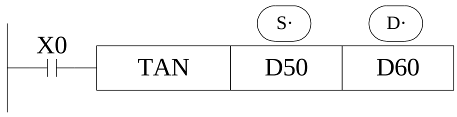
Operands	Function	Data Type
S	Soft element address need to do tan	32bit,BIN
D	Result address	32bit,BIN

3: Suitable soft components

Word

Operands	System									Constant	Module	
	D	FD	ED	TD	CD	DX	DY	DM	DS	K/H	ID	QD
S	●	●				●	●	●	●	●		
D	●						●	●	●			

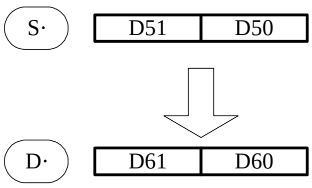
Description



(D51,D50)RAD → (D61,D60)TAN

Binary Floating Binary Floating

- This instruction performs the mathematical TAN operation on the floating point value in S. The result is stored in D.



RAD value (angle×π/180)

Assign the binary floating value

TAN value

Binary Floating

4-9-11 ASIN [ASIN]

1: Summary

ASIN [ASIN]			
16 bits	-	32 bits	ASIN
Execution condition	Normally ON/OFF, rising/falling edge	Suitable Models	XC2.XC3.XC5.XCM
Hardware requirement	V3.0 and above version	Software requirement	-

2: Operands

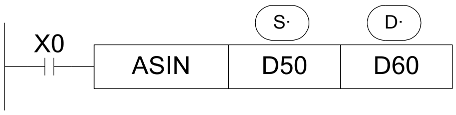
Operands	Function	Data Type
S	Soft element address need to do arcsin	32 bits, BIN
D	Result address	32 bits, BIN

3: Suitable soft components

Word

Operands	System									Constant	Module	
	D	FD	ED	TD	CD	DX	DY	DM	DS	K/H	ID	QD
S	●	●				●	●	●	●	●		
D	●						●	●	●			

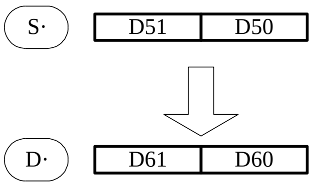
Description



(D51,D50)ASIN → (D61,D60)RAD

Binary Floating Binary Floating

- This instruction performs the mathematical ASIN operation on the floating point value in S. The result is stored in D.



ASIN value

Binary Floating

RAD value (angle×π/180)

Assign the binary floating value

4-9-12 ACOS [ACOS]

1: Summary

ACOS [ACOS]			
16 bits	-	32 bits	ACOS
Execution condition	Normally ON/OFF, rising/falling edge	Suitable Models	XC2.XC3.XC5.XCM
Hardware requirement	V3.0 and above	Software requirement	-

2: Operands

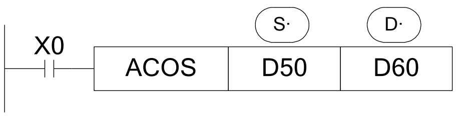
Operands	Function	Data Type
S	Soft element address need to do arccos	32 bits, BIN
D	Result address	32 bits, BIN

3: Suitable soft components

Word

Operands	System									Constant	Module	
	D	FD	ED	TD	CD	DX	DY	DM	DS	K/H	ID	QD
S	●	●				●	●	●	●	●		
D	●						●	●	●			

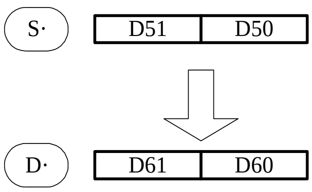
Description



(D51,D50)ACOS → (D61,D60)RAD

Binary Floating Binary Floating

- Calculate the arcos value(radian), save the result in the target address



TCOS value

Binary Floating

RAD value (angle×π/180)

Assign the binary floating value

4-9-13 ATAN [ATAN]

1: Summary

ATAN [ATAN]			
16 bits	-	32 bits	ACOS
Execution condition	Normally ON/OFF, rising/falling edge	Suitable Models	XC2.XC3.XC5.XCM
Hardware requirement	V3.0 and above	Software requirement	-

2: Operands

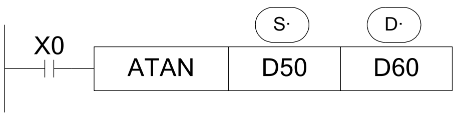
Operands	Function	Data Type
S	Soft element address need to do arctan	32 bit, BIN
D	Result address	32 bit, BIN

3: Suitable soft components

Word

Operands	System									Constant	Module	
	D	FD	ED	TD	CD	DX	DY	DM	DS	K/H	ID	QD
S	●	●				●	●	●	●	●		
D	●						●	●	●			

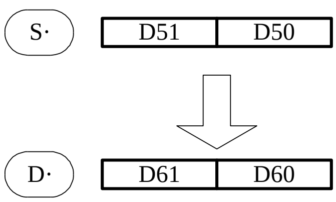
Description



(D51,D50)ATAN → (D61,D60)RAD

Binary Floating Binary Floating

- Calculate the arctan value (radian), save the result in the target address



ATAN value

Binary Floating

RAD value (angle×π/180)

Assign the binary floating value

4-10 RTC Instructions

Mnemonic	Function	Chapter
TRD	Clock data read	4-10-1
TWR	Clock data write	4-10-2

※1: Only available on models equipped with RTC function.

4-10-1 Read the clock data [TRD]

1: Instruction Summary

Read the clock data:

Read the clock data: [TRD]			
16 bits	TRD	32 bits	-
Execution condition	Normally ON/OFF, rising/falling edge	Suitable Models	XC2.XC3.XC5.XCM
Hardware requirement	V2.51 and above	Software requirement	-

2: Operands

Operands	Function	Data Type
D	Register to save clock data	16 bits, BIN

3: Suitable Soft Components

Word	Operands	System									Constant	Module	
		D	FD	ED	TD	CD	DX	DY	DM	DS	K/H	ID	QD
	D	•			•	•							

Functions and Actions

The current time and date of the real time clock are read and stored in the 7 data devices specified by the head address D.

- Read PLC's real time clock according to the following format.
  The reading source is the special data register (D8013~D8019) which save clock data.

	Unit	Item	Clock data		Unit	Item
Special data register for real time clock t	D8018	Year	0-99	→	D0	Year
	D8017	Month	1-12	→	D1	Month
	D8016	Date	1-31	→	D2	Date
	D8015	Hour	0-23	→	D3	Hour
	D8014	Minute	0-59	→	D4	Minute
	D8013	Second	0-59	→	D5	Second
	D8019	Week	0 (Sun.)-6 (Sat.)	→	D	Week

4-10-2 Write Clock Data [TWR]

1: Instruction Summary

Write the clock data:

Write clock data [TRD]			
16 bits	-	32 bits	TRD
Execution condition	Normally ON/OFF, rising/falling edge	Suitable Models	XC2.XC3.XC5.XCM
Hardware requirement	V2.51 and above	Software requirement	-

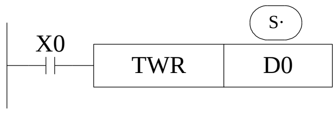
2: Operands

Operands	Function	Data Type
S	Write the clock data to the register	16 bits, BIN

3: Suitable Soft Components

Word	Operands	System									Constant	Module	
		D	FD	ED	TD	CD	DX	DY	DM	DS	K/H	ID	QD
	S	●	●		●	●	●	●	●	●			

Functions and Actions



The 7 data devices specified with the head address S are used to set a new current value of the real time clock.

(3) Write the set clock data into PLC's real time clock.

In order to write real time clock, the 7 data devices specified with the head address S· should be pre-set.

Data for clock setting

Unit	Item	Clock data		Unit	Item
D10	Year	0-99	→	D8018	Year
D11	Month	1-12	→	D8017	Month
D12	Date	1-31	→	D8016	Date
D13	Hour	0-23	→	D8015	Hour
D14	Minute	0-59	→	D8014	Minute
D15	Second	0-59	→	D8013	Second
D16	Week	0 (Sun.)-6	→	D8019	Week

Special data register for real time clock t

After executing TWR instruction, the time in real time clock will immediately change to be the new set time. So, when setting the time it is a good idea to set the source data to a time a number of minutes ahead and then drive the instruction when the real time reaches this value.

5 High Speed Counter (HSC)

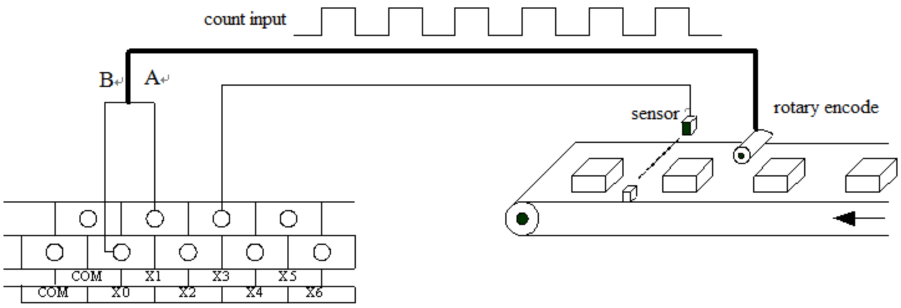
In this chapter we explore high speed counter's functions, including high speed count model, wiring method, read/write HSC value, reset etc.

Instructions List for HSC

MNEMONIC	FUNCTION	CIRCUIT AND SOFT COMPONENTS	CHAPTER
READ/WRITE HIGH SPEED COUNTER			
HSCR	Read HSC	HSCR S D	5-6-1
HSCW	Write HSC	HSCW S D	5-6-2
OUT	HSC (High Speed Counter)	BSTOP S1 S2	3-13
OUT	24 segments HSC Interruption	Cn Kn D ( )	5-10
RST	HSC Reset	BGOON S1 S2	3-13

5-1 Functions Summary

XC series PLCs have an HSC (High Speed Counter) function which is independent of the scan cycle. By choosing different counters, the high speed input signals can be tested with detect sensors and rotary encoders. The highest testing frequency can reach 80KHz.

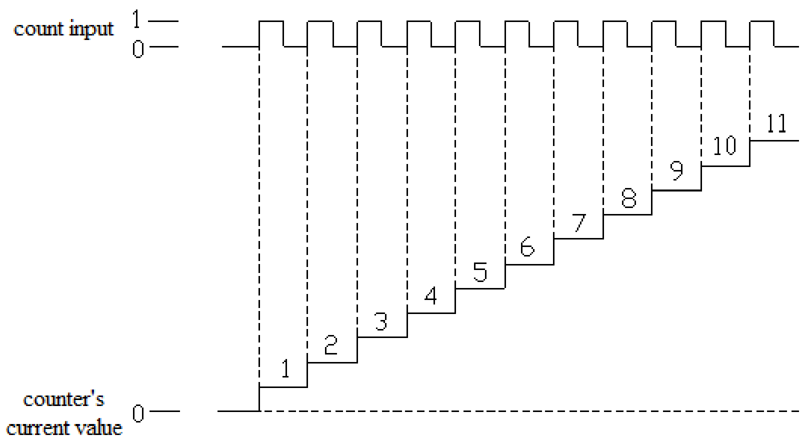


5-2 HSC Mode

The XC Series' high speed counter function has three count modes: Increment Mode, Pulse + Direction Mode and AB phase Mode;

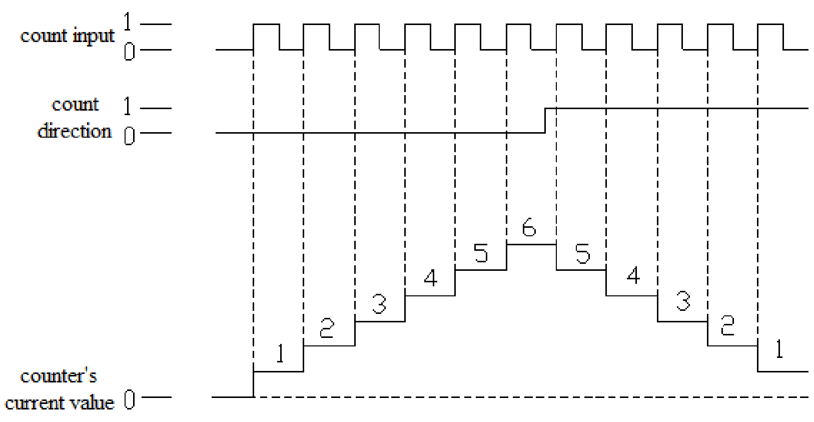
Increment Mode

Under this mode, count and input the pulse signal, the count value increase at each pulse's rising edge;



Pulse + Direction Mode

Under this mode, the pulse signal and direction signal are inputted, the count value increases or decreases with the direction signal's status. When the count signal is OFF, the count input's rising edge carry on plus count; When the count signal is ON, the count input's rising edge carry on minus count;

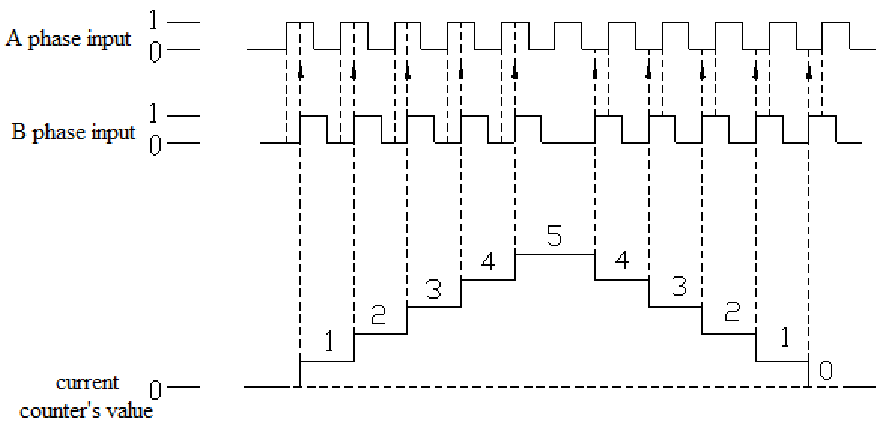


AB Phase Mode

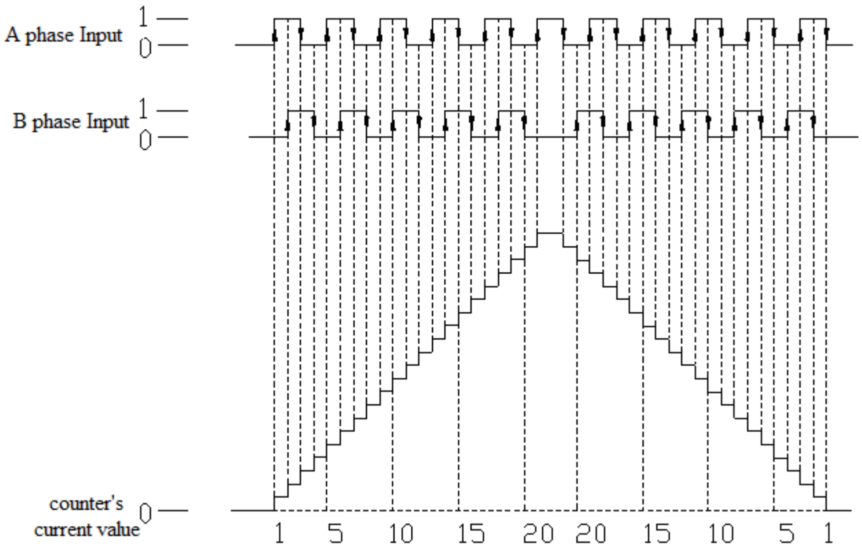
Under this mode, the HSC value increases or decreases according to two differential signals (A phase and B phase). There are two frequyency modes available: 1-time frequency and 4-time frequency. The default count mode is 4-time mode.

1-time frequency and 4-time frequency modes are shown below:

- **1-time Frequency**



- **4-time Frequency**



5-3 HSC Range

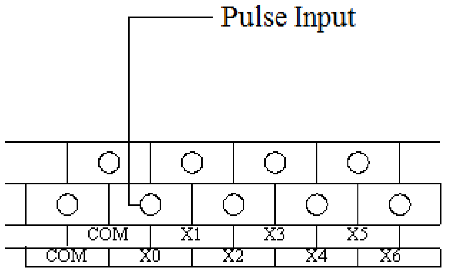
HSC’s count range is: K-2, 147, 483, 648 ~ K+2, 147, 483, 647. If the count value overflows this range, then up flow or down flow appears;

For “up flow”, it means the count value jumps from K+2, 147, 483, 647 to be K-2, 147, 483, 648, then continues to count; For “down flow”, it means the count value jumps from K-2, 147, 483, 648 to be K+2, 147, 483, 647 then continues to count.

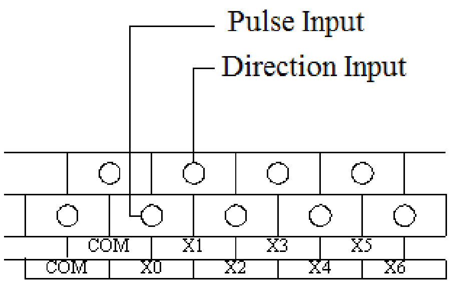
5-4 HSC Input Wiring

For the counter’s pulse input wiring, things differ with different PLC models and counter models; several typical input wiring methods are shown below: (take XC3-48 as the example):

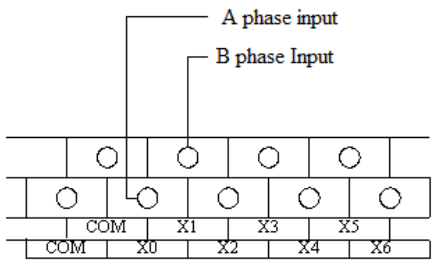
increment mode (Counter C600)



Pulse+Direction Mode (C620)



AB phase Mode (C630)



5-5 HSC Ports Assignment

Description of Letters:

U	Dir	A	B
Pulse input	Count Direction Judgment (OFF=increment, ON=decrement)	A phase input	B phase input

Normally, X0 and X1 can accept 80KHz frequency under single phase mode and AB phase mode. Other terminals can accept only 10KHz under single phase mode, 5KHz under AB phase mode. X can use as normal input terminals when they are not used as high speed input. The detailed assignment is shown as below:

XC2 Series PLC																		
	Increment										Pulse+Dir Input					AB Phase Mode		
	C600	C602	C604	C606	C608	C610	C612	C614	C616	C618	C620	C622	C624	C626	C628	C630	C632	C634
Max.F	80K	80K	10K	10K	10K						80K	10K				80K	5K	
4-times F																√		
Count Interrupt	√	√	√	√	√						√					√		
X000	U										U					A		
X001		U									Dir					B		
X002																		
X003			U									U					A	
X004												Dir					B	
X005																		
X006				U														
X007					U													
X010																		
X011																		
X012																		

	Increment										Pulse+Dir Input					AB Phase Mode		
	C600	C602	C604	C606	C608	C610	C612	C614	C616	C618	C620	C622	C624	C626	C628	C630	C632	C634
*Max.F	10K	10K	10K	10K							10K	10K				5K		
4-times F																		
Count Interrupt	√	√	√	√								√						
X000	U										U					A		
X001											Dir					B		
X002		U																
X003			U															
X004																		
X005				U														

* C600、C620、C630 can support 80KHz with special requirement

XC3-19AR-E																		
	Increment										Pulse+Dir Input					AB Phase Mode		
	C600	C602	C604	C606	C608	C610	C612	C614	C616	C618	C620	C622	C624	C626	C628	C630	C632	C634
Max.F	10K	10K	10K	10K							10K	10K				5K	5K	
4-times F																	√	
Count Interrupt	√	√	√	√								√					√	
X000	U										U					A		
X001											Dir					B		
X002		U										U					A	
X003												Dir					B	
X004			U															
X005				U														

XC3-24、32 PLC and XC5-48、60 PLC			
	Increment	Pulse+Dir Input	AB Phase Mode

	C600	C602	C604	C606	C608	C610	C612	C614	C616	C618	C620	C622	C624	C626	C628	C630	C632	C634
Max.F	80K	80K	10K	10K	10K	10K					80K	10K	10K			80K	5K	5K
4-times F																√		√
Count Interrupt	√	√	√	√	√	√					√					√		
X000	U										U					A		
X001		U									Dir					B		
X002																		
X003			U									U					A	
X004												Dir					B	
X005																		
X006				U									U					A
X007													Dir					B
X010																		
X011					U													
X012						U												

XC3-48、60 PLC																		
	Increment										Pulse+Dir Input					AB Phase Mode		
	C600	C602	C604	C606	C608	C610	C612	C614	C616	C618	C620	C622	C624	C626	C628	C630	C632	C634
Max.F	80K	80K	10K	10K							80K	80K				80K	80K	
4-times F																	√	
Count Interrupt	√	√	√	√								√					√	
X000	U										U					A		
X001											Dir					B		
X002		U										U					A	
X003												Dir					B	
X004			U															
X005				U														

XC5-24/32 PLC、XCM-24/32 PLC																		
	Increment										Pulse+Dir Input					AB Phase Mode		
	C600	C602	C604	C606	C608	C610	C612	C614	C616	C618	C620	C622	C624	C626	C628	C630	C632	C634
Max.F	80K	10K									80K					80K		

4-times F																√		
Count Interrupt	√	√									√					√		
X000	U										U					A		
X001											Dir					B		
X002																		
X003		U																
X004																		
X005																		
X006																		

5-6 Read/Write HSC value

All high speed counters support read instruction [HSCR] and write instruction [HSCW].

Hardware must be V3.1c and above.

5-6-1 Read HSC value [HSCR]

1: Instruction Summary

Read HSC value to the specified register;

Read from HSC [HSCR]/ write to HSC [HSCW]			
16 bits Instruction	-	32 bits Instruction	HSCR
Execution condition	Normally ON/OFF, rising/falling edge	Suitable models	XC2、XC3、XC5、XCM
Hardware requirement	V3.1c and above	Software requirement	-

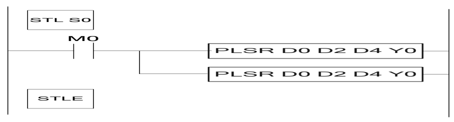
2: Operands

Operands	Function	Type
S	Specify HSC code	32 bits, BIN
D	Specify the read/written register	32 bits, BIN

3: Suitable Soft Components

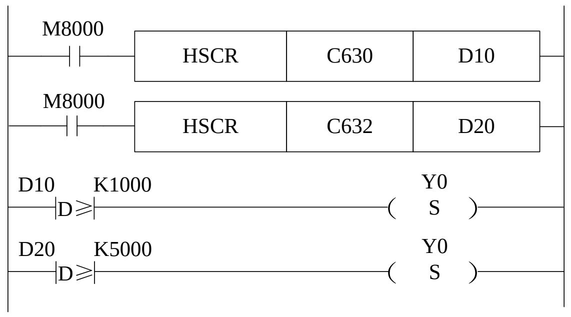
word	operands	system									constant	module	
		D	FD	ED	TD	CD	DX	DY	DM	DS	K/H	ID	QD
	S					●							

Functions and Actions



- When the activate condition is true, read the HSC value in C630 (DWORD) into D10 (DWORD)
- Instruction HSCR reads the HSC value into the specified register, improve HSC value’s precision.

Sample Program:



5-6-2 Write HSC Value [HSCW]

1: Instruction Summary

Write the specified register value into HSC;

Write HSC value [HSCW]			
16 bits Instruction	-	32 bits Instruction	HSCW
Execution condition	Normally ON/OFF, rising/falling edge	Suitable models	XC2、XC3、XC5、XCM
Hardware requirement	V3.1c and above	Software requirement	-

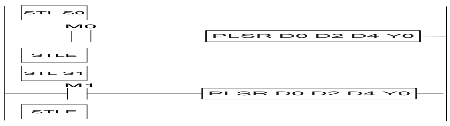
2: Operands

Operands	Function	Type
S	Specify HSC code	32 bits, BIN
D	Specify the read/written register	32 bits, BIN

3: Suitable soft components

Word	operands	system									constant	module	
		D	FD	ED	TD	CD	DX	DY	DM	DS	K/H	ID	QD
	S					●							
	D	●											

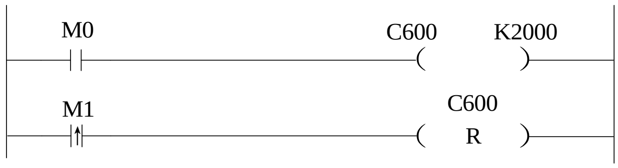
Functions and Actions



- When the activated condition is true, write the value in D20 (DWORD) into C630 (DWORD), the original value is replaced;
- We suggest users to apply high speed counter only with HSCR and HSCW, not with other instructions like DMOV, LD>, DMUL etc. and users must run after converting HSC to be other registers.

5-7 HSC Reset Mode

Reset HSC via software:



In the above graph, when M0 is ON, C600 starts to count the input pulse on X0; when M1 changes from OFF to be ON, reset C600, clears the count value

5-8 AB Phase Counter Multiplication Setting

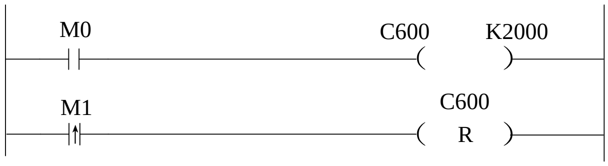
About AB phase counter, modify the frequency multiplication value via setting FLASH data register FD8241, FD8242, FD8243. If the value is 1, it is 1-time frequency, if it is 4, it is 4-time frequency.

Register	Function	Set Value	Meaning
FD8241	Frequency multiplication of C630	1	1-time frequency
		4	4-time frequency
FD8242	Frequency multiplication of C632	1	1-time frequency
		4	4-time frequency
FD8243	Frequency multiplication of C634	1	1-time frequency
		4	4-time frequency

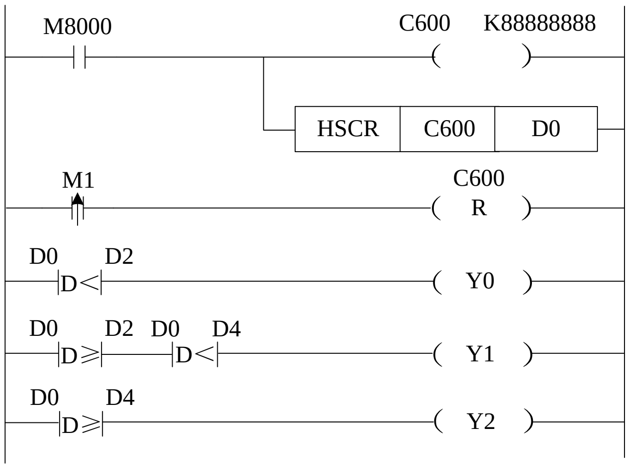
5-9 HSC Examples

Below, we take XC3-60 PLC as the example, to introduce HSC's program form;

Increment Mode

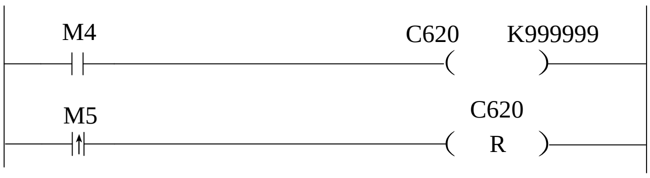


- When M0 is ON, C600 starts the HSC with the OFF→ON of X000;
- When comes the rising edge of M1, reset HSC C600

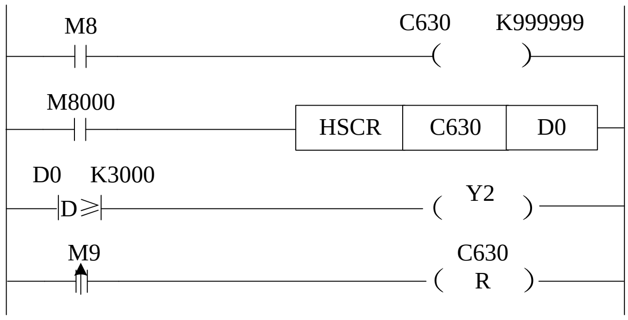


- When normally ON coil M8000 is ON, set the value of C600, the set value is K888888888, read the HSC value (DWORD) into data register D0 (DWORD).
- If the value in C600 is smaller than value in D2, set the output coil Y0 ON; If the value in C600 equals or be larger than value in D2, and smaller than value in D4, set the output coil Y1 ON; If the value in C600 equals or be larger than value in D4, set the output coil Y2 ON;
- When comes the rising edge of M1, resets HSC C600 and stops counting.

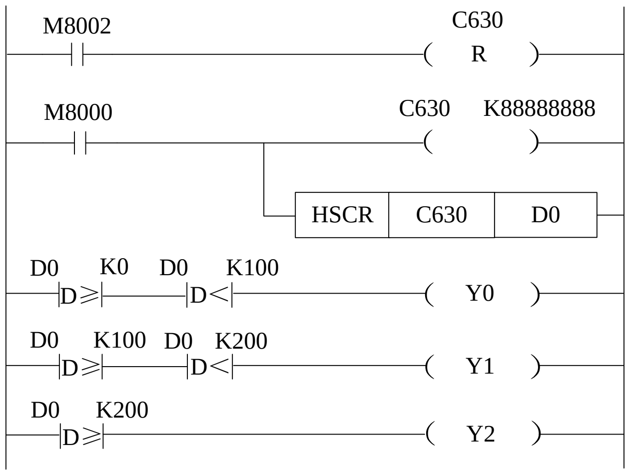
Pulse+Direction Mode



- When M4 is ON, C620 starts the HSC with the OFF→ON of X000; judge the count direction according to the input X001 status (OFF or ON). If X001 is OFF, it's increment count; if X001 is ON, it's decrement count;
- When it reaches the rising edge of M5, it will reset HSC C620 and stop counting.



- When M8 is ON, C630 starts to count immediately. Count input via X000 (B Phase)、X001 (A Phase)
- When the count value exceeds K3000, output coil Y2 is ON;
- When comes the rising edge of M9, it resets HSC C630



- When the rising edge of initial positive pulse coil M8002 comes, i.e. Each scan cycle starts, HSC C630 reset and clear the count value.
- When set coil M8000 ON, C630 starts to count, the count value is set to be K8888888。
- If the count value is greater than K0 but smaller than K100, the output coil Y0 set ON; If the count value is greater thanK100 but smaller than K200 时，the output coil Y1 set ON; If the count value is greater thanK200, the output coilY2 set ON;

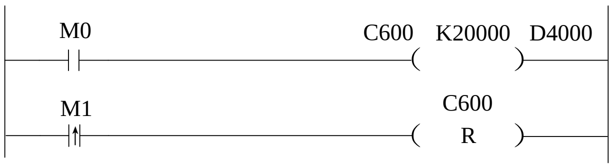
5-10 HSC Interruption

To XC series PLC, each HSC channels has 24 segments 32-bit pre-set value. When the HSC difference value equals the correspond 24-segment pre-set value, then interruption occurs according to the interruption tag;

To use this function, please use hardware V3.1c or above;

5-10-1 Instruction Description

(for Interruption program instructions, please refer chapter 5-10-4)



LD	M0			//HSC activate condition M0 (interruption count condition)
OUT	C600	K20000	D4000	//HSC value and set the start ID of 24-segment
LDP	M1			//activate condition reset
RST	C600			//HSC and 24-segment reset (interruption reset)

As shown in the above graph, data register D4000 is the start ID of 24-segment pre-set value area. As a back-up, save each pre-set value in DWORD form. Please pay attention when using HSC:

- If certain pre-set value is 0, it means count interruption stops at this segment;
- Set the interruption pre-set value but not write the correspond interruption program is not allowed;
- 24-segment interruption of HSC occurs in order. I.e. If the first segment interruption doesn't happen, then the second segment interruption will not happen;
- 24-segment pre-set value can be specified to be relative value or absolute value. Meantime, users can specify the set value to be loop or not. But the loop mode can't be used together with absolute value.

5-10-2 Instruction tags to HSC

In the below table, we list each counter's 24-segment pre-set value to its interruption tag.

E.g.: 24-segment pre-set value of counter C600 correspond with the interruption pointer: I1001、I1002、I1003、…I1024.

Increment Mode

Counter	Interruption tag
C600	I1001~I1024
C602	I1101~I1124
C604	I1201~I1224
C606	I1301~I1324
C608	I1401~I1424
C610	I1501~I1524
C612	I1601~I1624
C614	I1701~I1724
C616	I1801~I1824
C618	I1901~I1924

Define the preset value

Pulse + Direction Mode

Counter	Interruption tag
C620	I2001~I2024
C622	I2101~I2124
C624	I2201~I2224
C626	I2301~I2324
C628	I2401~I2424

AB Phase Mode

Counter	Interruption tag
C630	I2501~I2524
C632	I2601~I2624
C634	I2701~I2724
C636	I2801~I2824
C638	I2901~I2924

HSC 24-segment pre-set value is the difference value, the count value equals the counter's current value plus the preset value, self-generating the interruption. N interruption tags correspond with N interruption preset values. The (N+1) preset value is 0;

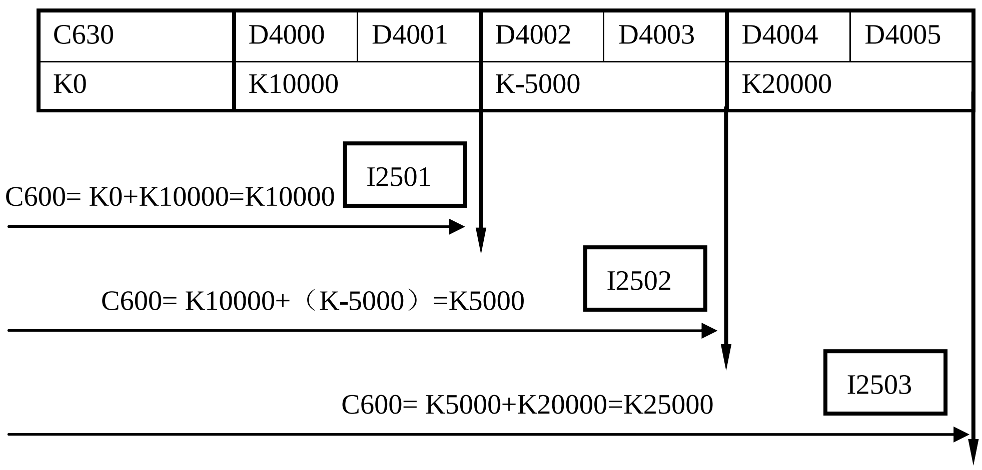
E.g. 1, the current value is C630 is 0, the first preset value is 10000, the preset value in segment 2 is －5000, and the preset value in segment 3 is 20000.

When counting begins: if the counter's current value is 10000, the first interruption I2501 will be generated.

When counting begins: if the counter's current value is 5000, the first interruption I2502 will be generated.

When counting begings: if the counter's current value is 25000, the first interruption I2503 will be generated.

See graph below:

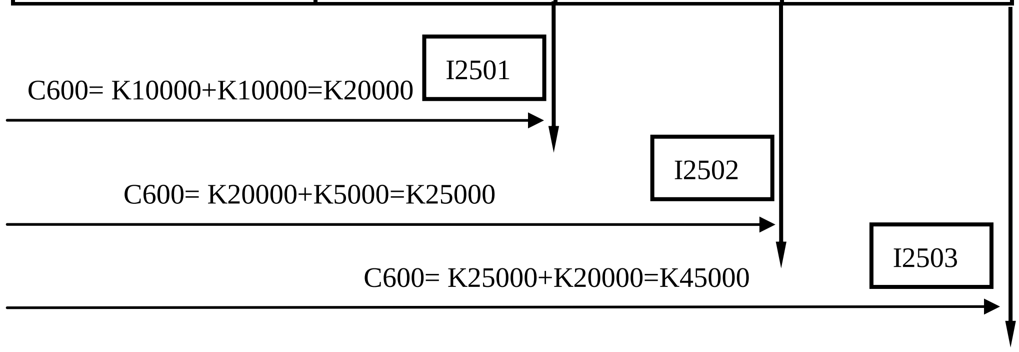


E.g. 2, the current value is C630 is 10000, the first preset value is 10000, the preset value in segment 2 is 5000, the preset value in segment 3 is 20000.

When count begins, if the counter's current value is 20000, this generates first interruption at I2501;
When count begins, if the counter's current value is 25000, this generates first interruption at I2502;
When count begins, if the counter's current value is 45000, this generates first interruption at I2503.

See graph below:

C630	D4000	D4001	D4002	D4003	D4004	D4005
K10000	K10000		K5000		K20000	



5-10-3 Loop Mode of HSC Interruption

Mode 1: Unicycle (normal mode)

Not happen after HSC interruption ends. The conditions below can re-start the interruption:

(1) reset the HSC

(2) Reboot the HSC activate condition

Mode 2: Continuous loop

Restart after HSC interruption ends. This mode is especially suitable for the following application:

(7) continuous back-forth movement

(8) Generate cycle interruption according to the defined pulse

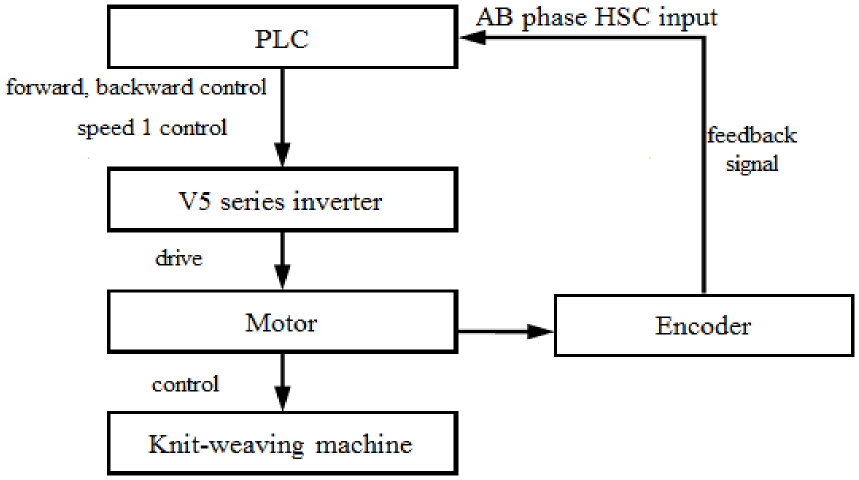
With setting the special auxiliary relays, users can set the HSC interruption to be unicycle mode or continuous loop mode. The loop mode is only suitable with the relative count. The detailed assignment is show below:

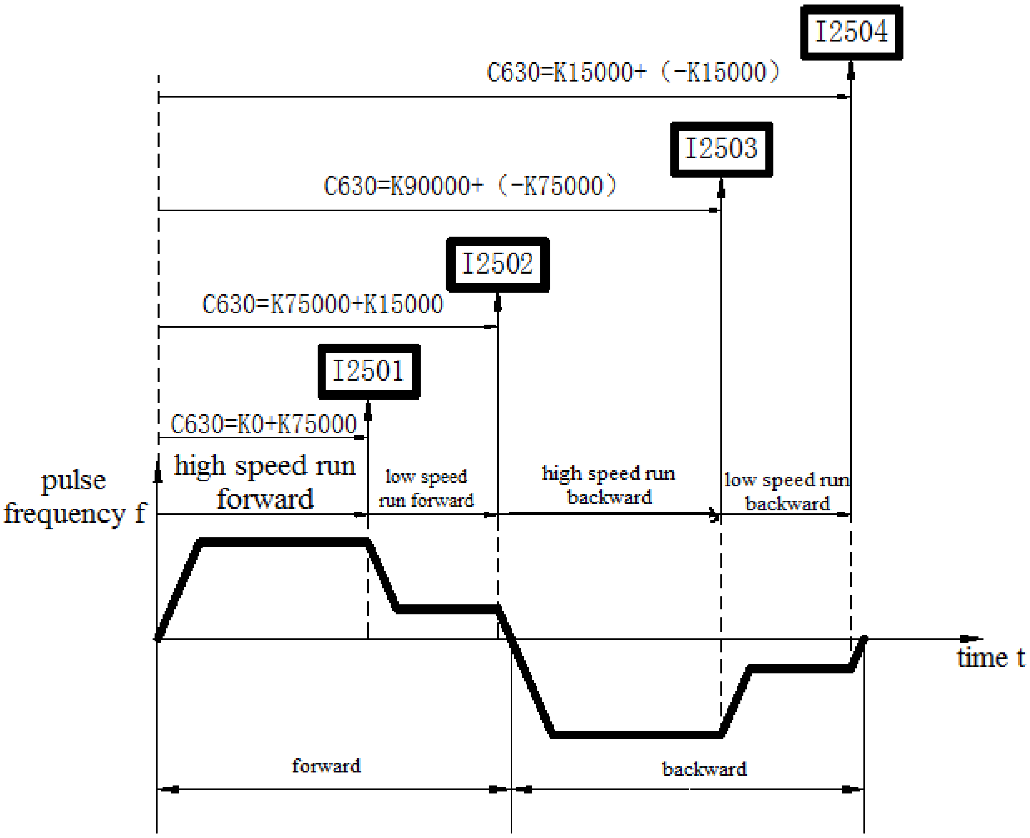
ID	HSC ID	Setting
M8270	24 segments HSC interruption loop (C600)	OFF: unicycle mode ON: continuous loop mode
M8271	24 segments HSC interruption loop (C602)	
M8272	24 segments HSC interruption loop (C604)	
M8273	24 segments HSC interruption loop (C606)	
M8274	24 segments HSC interruption loop (C608)	
M8275	24 segments HSC interruption loop (C610)	
M8276	24 segments HSC interruption loop (C612)	
M8277	24 segments HSC interruption loop (C614)	
M8278	24 segments HSC interruption loop (C616)	
M8279	24 segments HSC interruption loop (C618)	
M8280	24 segments HSC interruption loop (C620)	
M8281	24 segments HSC interruption loop (C622)	
M8282	24 segments HSC interruption loop (C624)	
M8283	24 segments HSC interruption loop (C626)	
M8284	24 segments HSC interruption loop (C628)	
M8285	24 segments HSC interruption loop (C630)	
M8286	24 segments HSC interruption loop (C632)	
M8287	24 segments HSC interruption loop (C634)	

5-10-4 Example of HSC Interruption

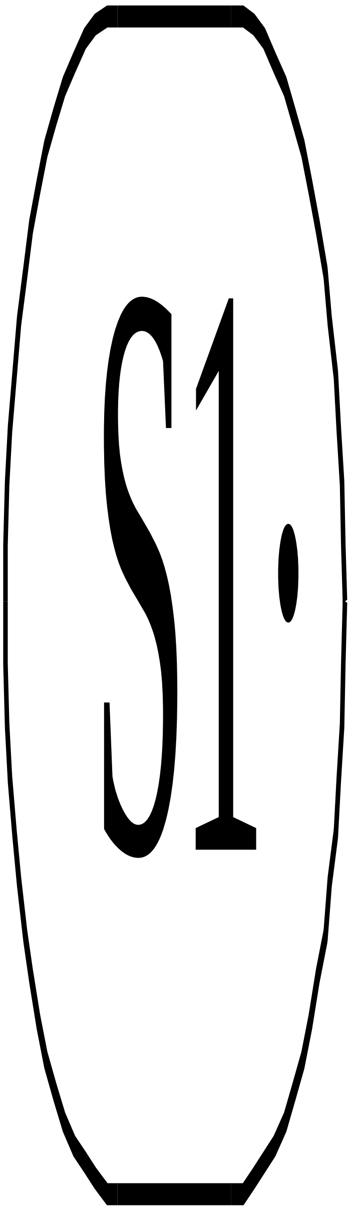
E.g.2：Application on knit-weaving machine (continuous loop mode)

The system theory is shown as below: Control of the inverter via PLC, Processing the movement, via the feedback signal from encoder, control the knit-weaving machine and realize the precise position.





Below is PLC program: Y2 represents forward output signal; Y3 represents backward output signal; Y4 represents output signal of speed 1; C340: Back-forth times accumulation counter; C630: AB phase HSC;



Instruction List Form:

LD M8002 //M8002 is initial positive pulse coil

```
SET     M8285                          //special auxiliary relay set ON, to enable C630
                                                                    continuous loop

SET     Y2                             //set output coil Y2 (i.e. Start run forth)
LDP     Y2                             //knit-weaving machine back-forth times counter's
                                           activate condition Y2(forth rising edge activate)

OUT     C340     K1000000              //counter C340 starts to count
LD      M8000                          //M8000 is normally ON coil
DMOV    K75000   D4000                 //set segment-1 ID D4000 to be K75000
DMOV    K15000   D4002                 //set segment-2 D4002 to be K15000
DMOV    K-75000  D4004                 //set segment-3 D4004 to be K-75000
DMOV    K-15000  D4006                 //set segment-4 D4004 to be K-15000
LD      M8000                          //M8000 is normally ON coil
OUT     C630     K30000000   D4000     //HSC and start ID of 24-segment
LD      M8000                          //M8000 is normally ON coil
HSCR    C630     D200                  //read the HSC value of C630 to D200
FEND                                   //main program end
I2501                                  //interruption tag of segment 1
LD      M8000                          //M8000 is normally ON coil
SET     Y4                             //output coil Y4 set (low-speed run with speed 1)
IRET                                   //interruption return tag
I2502                                  ///interruption tag of segment 2
LD      M8000                          //M8000 is normally ON coil
RST     Y4                             //output coil Y4 reset (low-speed run stop)
RST     Y2                             //output coil Y2 reset (run forward stops)
SET     Y3                             //output coil Y3 set (back running)
IRET                                   //interruption return tag
I2503                                  ///interruption tag of segment 3
LD      M8000                          //M8000 is normally ON coil
SET     Y4                             //output coil Y4 set (low-speed run with speed 1)
IRET                                   //interruption return tag
I2504                                  ///interruption tag of segment 4
LD      M8000                          //M8000 is normally ON coil
RST     Y3                             //output coil Y3 reset (back running stop)
RST     Y4                             //output coil Y4 reset (low-speed run stop)
SET     Y2                             //output coil Y2 set (run forward)
IRET                                   //interruption return tag
```

6 Pulse Output

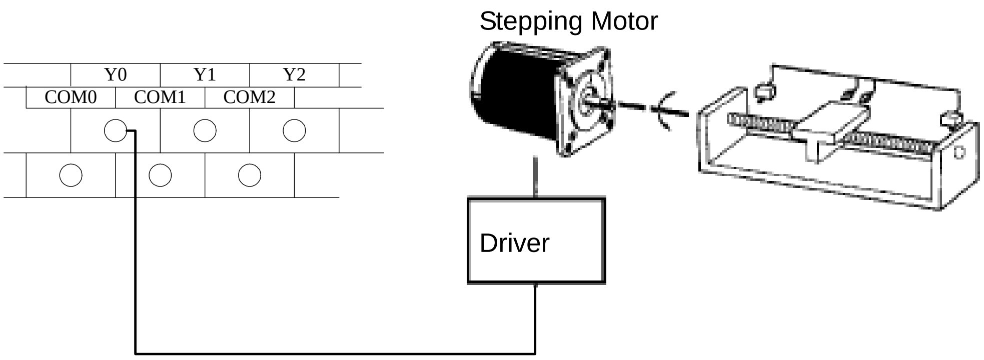
In this chapter we explain the pulse function of XC series PLCs. The content includes pulse output instructions, input/output wiring, items to note in relation to coils and registers etc.

Pulse Output Instructions List

Mnemonic	Function	Circuit And Soft Device	Chapter
PULSE OUTPUT			
PLSY	Unidirectional ration pulse output without ACC/DEC time change	PLSY S1 S2 D	6-2-1
PLSF	Variable frequency pulse output	PLSF S D	6-2-2
PLSR	Ration pulse output with ACC/DEC speed	PLSR S1 S2 S3 D	6-2-3
PLSNEXT/ PLSNT	Pulse Section Switch	PLSNT S	6-2-4
STOP	Pulse Stop	STOP S	6-2-5
PLSMV	Refresh Pulse Nr. immediately	PLSMV S D	6-2-6
ZRN	Original Return	ZRN S1 S2 S3 D	6-2-7
DRVI	Relative Position Control	DRVI S1 S2 S3 D1 D2	6-2-8
DRVA	Absolute Position Control	DRVA S1 S2 S3 D1 D2	6-2-9
PLSA	Absolute Position multi-section pulse control	PLSA S1 S2 D	6-2-10

6-1 Functions Summary

Generally, XC3 and XC5 series PLC are equipped with 2CH pulse output function. Via different instructions, users can realize unidirectional pulse output without ACC/DEC speed; unidirectional pulse output with ACC/DEC speed; multi-segments, positive/negative output etc., the output frequency can reach 400K Hz.



※1: To use pulse output, please choose PLC with transistor output, like XC3-14T-E or XC3-60RT-E etc.
※2: XC5 series 32I/O PLC has 4CH (Y0, Y1, Y2, Y3) pulse output function.

6-2 Pulse Output Types and Instructions

6-2-1 Unidirectional ration pulse output without ACC/DEC time change [PLSY]

1: Instruction Summary

Instruction to generate ration pulse with the specified frequency;

Unidirectional ration pulse output without ACC/DEC time change [PLSY]			
16 bits instruction	PLSY	32 bits instruction	DPLSY
Execution condition	Normally ON/OFF coil	Suitable models	XC2、XC3、XC5、XCM
Hardware requirement	-	Software requirements	-

2: Operands

Operands	Function	Type
S1	Specify the frequency's value or register ID	16 bits/32 bits, BIN
S2	Specify the pulse number or register's ID	16 bits /32 bits, BIN
D	Specify the pulse output port	bit

3: Suitable soft components

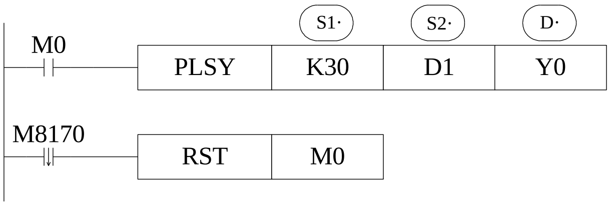
Word

operands	system									constant	module	
	D	FD	ED	TD	CD	DX	DY	DM	DS	K/H	ID	QD
S1	•	•		•	•					•		
S2	•	•		•	•					•		

Bit

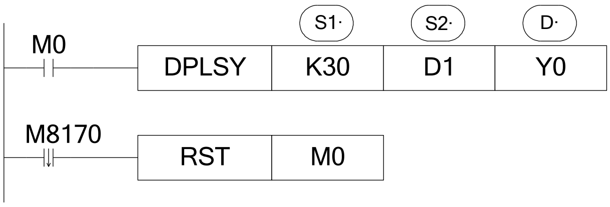
operands	system						
	X	Y	M	S	T	C	Dn.m
D		•					

《**16 bits Instruction**》



- Frequency Range: 0~400KHz；
- Pulse Quantity Range: 0~K32767；
- Pulse output from Y000 or Y001 only;
- When M0 is ON, PLSY instruction output 30Hz pulse at Y0, the pulse number is decided by D1, M8170 is set ON only when sending the pulse. When the output pulse number reaches the set value, stop sending the pulse, M8170 is set to be OFF, reset M0;

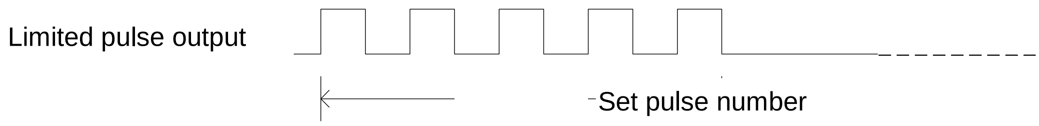
《**32 bits Instruction**》



- Frequency Range: 0~400KHz；
- Pulse Quantity Range: 0~K2147483647；
- Pulse output from Y000 or Y001 only;
- When M0 is ON, DPLSY instruction output 30Hz pulse at Y0, the pulse number is decided by D2D1, M8170 is set ON only when sending the pulse. When the output pulse number reaches the set value, stop sending the pulse, M8170 is set to be OFF, reset M0;

Output Mode

《**continuous or limited pulse number**》



When finish sending the set pulse number, stop outputting automatically

Items to Note

If the control object is stepping/servo motor, we recemend users not use this instruction, to avoid the motor losing synchronism. PLSR is available.

6-2-2 Variable Pulse Output [PLSF]

1: Instruction Summary

Instruction to generate continuous pulse in the form of variable frequency

Variable Pulse Output [PLSF]			
16 bits Instruction	PLSF	32 bits Instruction	DPLSF
Execution condition	Normally ON/OFF coil	Suitable Models	XC2、XC3、XC5、XCM
Hardware requirement	-	Software requirement	-

2: Operands

Operands	Function	Type
S	Specify the frequency or register ID	16 bits/32 bits, BIN
D	Specify pulse output port	bit

3: Suitable soft components

Word

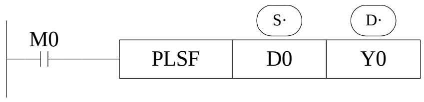
operands	system									constant	module	
	D	FD	ED	TD	CD	DX	DY	DM	DS	K/H	ID	QD
S	●	●		●	●					●		

Bit

operands	system						
	X	Y	M	S	T	C	Dn.m
D		●					

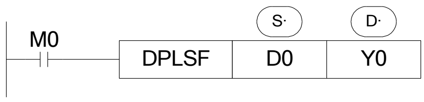
Functions and Actions

《16 bit instruction form》



- Frequency range: 6Hz~400KHz (when the set frequency is lower than 200Hz, output 200Hz)
- Pulse can only be output at Y000 or Y001.
- With the changing of setting frequency in D0, the output pulse frequency changes at Y0

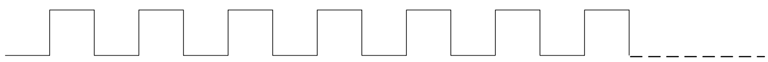
《32 bit instruction form》



- Frequency range: 6Hz~400KHz (when the set frequency is lower than 200Hz, output 200Hz)
- Pulse can only be output at Y000 or Y001.
- With the changing of setting frequency in D0, the output pulse frequency changes at Y0
- Accumulate pulse number in register D8170 (DWord)

Output Mode

Sequential pulse output



Sequential output pulse with the set frequency till stop output via the instruction

6-2-3 Multi-segment pulse control at relative position [PLSR]

PLSR/DPLSR instruction has two control modes. Below we will introduce one by one;

- **Mode 1: segment uni-directional pulse output PLSR**

1: Instruction Summary

Generate certain pulse quantity (segmented) with the specified frequency and acceleration/deceleration time

Segmented uni-directional pulse output [PLSR]			
16 bits Instruction	PLSR	32 bits Instruction	DPLSR
Execution condition	Normally ON/OFF coil	Suitable Models	XC2、XC3、XC5、XCM
Hardware requirement	-	Software requirement	-

2: Operands

Operands	Function	Type
S1	Specify the soft component's start ID of the segmented pulse parameters	16 bit/ 32 bit, BIN
S2	Specify acceleration/deceleration time or soft component's ID	16 bit/ 32 bit, BIN
D	Specify the pulse output port	Bit

3: Suitable soft components

Word

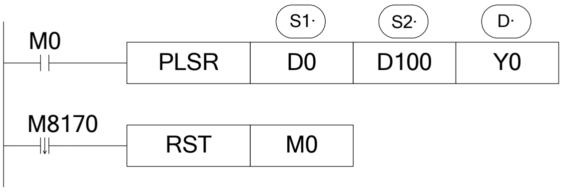
operands	system									constant	module	
	D	FD	ED	TD	CD	DX	DY	DM	DS	K/H	ID	QD
S1	●	●		●	●							
S2	●	●		●	●					●		

Bit

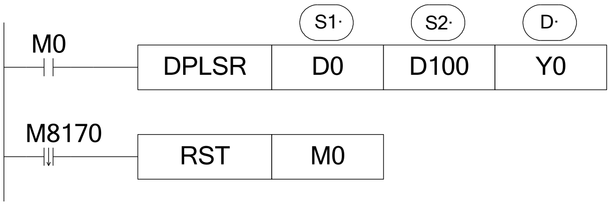
operands	system						
	X	Y	M	S	T	C	Dn.m
D		●					

Functions and Actions

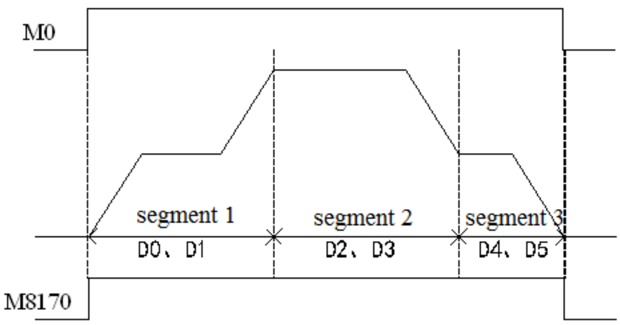
《16 bit instruction form》



《32 bit instruction form》



- The parameters' address is a section starts from **Dn** or **FDn**. In the above example (16bit instruction form): **D0** shows the first segment pulse's highest frequency; **D1** shows the first segment's pulse number; **D2** shows the second segment pulse's highest frequency; **D3** shows the second segment's pulse number，…… if the set value in **Dn**、**Dn+1** is 0, this represents the end of segment, the segment number is not limited.
- To 32 bit instruction **DPLSR**, **D0, D1** set the first segment pulse's highest frequency; **D2, D3** set the first segment's pulse number; **D4, D5** set the second segment pulse's highest frequency; **D6, D7** set the second segment's pulse number……
- Acceleration/deceleration time is the time from the start to the first segment's highest frequency. Meantime, it defines the slope of all segment's frequency to time. In this way the following acceleration/deceleration will perform according to this slope.
- Pulse can be output at only Y000 or Y001
- Frequency range: 0~400KHz;
- Pulse number range: 0~K32,767 (16 bits instruction)、0~K2,147,483,647 (32 bits instruction)
- Acceleration/deceleration time : below 65535 ms



➢ **Mode 2: segmented dual-directional pulse output PLSR**

1: Instruction Summary Generate certain pulse quantity with the specified frequency、acceleration/deceleration time and pulse direction ;

Segmented dual-directional pulse output [PLSR]

16 bits Instruction	PLSR	32 bits Instruction	DPLSR
Execution condition	Normally ON/OFF coil	Suitable Models	XC2、XC3、XC5、XCM
Hardware requirement	-	Software requirement	-

2: Operands

Operands	Function	Type
S1	Specify the soft component's start ID of the segmented pulse parameters	16 bit/ 32 bit, BIN
S2	Specify acceleration/deceleration time or soft component's ID	16 bit/ 32 bit, BIN
D1	Specify the pulse output port	Bit
D2	Specify the pulse output direction's port	Bit

3: Suitable soft components

Word

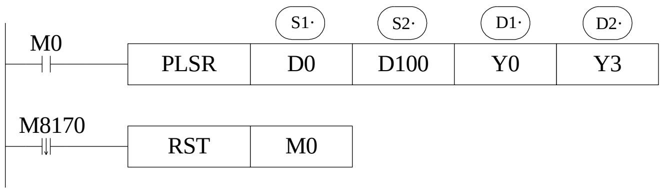
operands	system									constant	module	
	D	FD	ED	TD	CD	DX	DY	DM	DS	K/H	ID	QD
S1	●	●		●	●							
S2	●	●		●	●					K		

Bit

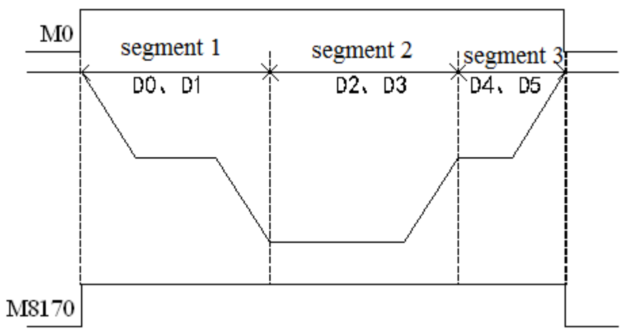
operands	system						
	X	Y	M	S	T	C	Dn.m
D1		●					
D2		●					

Functions and Actions

《16 bit instruction form》



- The parameters' address is a section starts from **Dn** or **FDn**. In the above example: **D0** set the first segment pulse's highest frequency; **D1** sshows the first segment's pulse number; **D2** shows the second segment pulse's highest frequency; **D3** shows the second segment's pulse number，…… if the set value in **Dn**、**Dn+1** is 0, this represents the end of segment, the number of segments available is not limited.
- Acceleration/deceleration time is the time from the start to the first segment's highest frequency. Meantime, it defines the slope of all segment's frequency to time. In this way the following acceleration/deceleration will perform according to this slope.
- Pulse can be output at only Y000 or Y001
- Y for Pulse direction can be specified freely. E.g.: if in S1 (the first segment) the pulse number is positive, Y output is ON; if the pulse number is negative, Y output is OFF; Note: in the first segment's pulse output, the pulse direction is only decided by the pulse number's nature (positive or negative) of the first segment.
- Frequency range: 0~400KHz;
- Pulse number range: 0~K32,767 (16 bits instruction)、0~K2,147,483,647 (32 bits instruction)
- Acceleration/deceleration time : below 65535 ms



6-2-4 Pulse Segment Switch [PLSNEXT]/[PLSNT]

1: Instruction Summary

Enter the next pulse output;

Pulse segment switch [PLSNEXT]/[PLSNT]

16 bits Instruction	PLSNEXT/PLSNT	32 bits Instruction	-
Execution condition	Rising/falling edge	Suitable Models	XC2、XC3、XC5、XCM
Hardware requirement	-	Software requirement	-

2: Operands

Operands	Function	Type
D	Specify the pulse output port	Bit

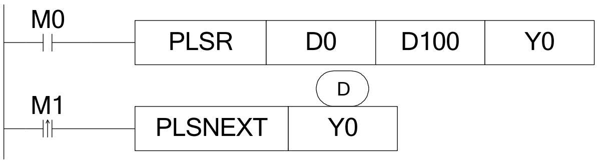
3: Suitable soft components

Bit

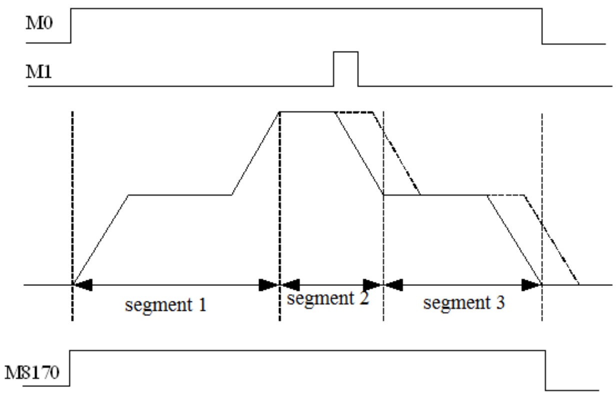
operands	system						
	X	Y	M	S	T	C	Dn.m
D		●					

Functions and Actions

《16 bit instruction form》



- If the pulse output reaches the highest frequency at the current segment, and output steadily at this frequency; when M1 changes from OFF to ON, then enter the next pulse output with the acceleration/deceleration time;
- Run the instruction within the acceleration/deceleration time is invalid;
- Instruction PLSNT is the brief of PLSNEXT, the functions are same;



--------(the dashed line represents the original pulse output

6-2-5 Pulse Stop [STOP]

1: Instruction Summary

Stop pulse output immediately;

Pulse stop [STOP]

16 bits Instruction	STOP	32 bits Instruction	-
Execution condition	Rising/falling edge	Suitable Models	XC2、XC3、XC5、XCM
Hardware requirement	-	Software requirement	-

2: Operands

Operands	Function	Type
D	Specify the port to stop pulse output	Bit

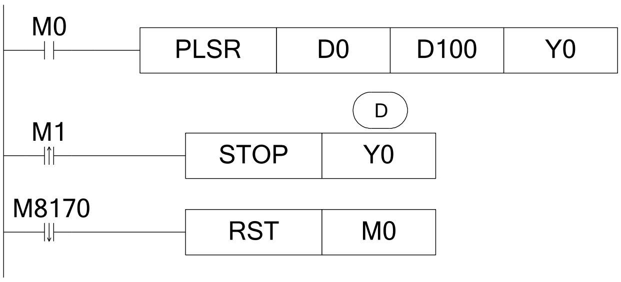
3: Suitable soft components

Bit

operands	system						
	X	Y	M	S	T	C	Dn.m
D		•					

Functions and Actions

《16 bit instruction form》



- When M000 changes from OFF to be ON, PLSR output pulse at Y000. D0 specifies the frequency, D001 specifies the pulse number, D100 specifies the acceleration/deceleration time; when the output pulse number reaches the set value, stop outputting the pulse; on the rising edge of M001, STOP instruction stops outputting the pulse at Y000.

6-2-6 Refresh the pulse number at the port [PLSMV]

1: Instruction Summary

Refresh the pulse number at the port;

Refresh the pulse number at the port [PLSMV]

16 bits Instruction	-	32 bits Instruction	PLSMV
Execution condition	Normally ON/OFF coil	Suitable Models	XC2、XC3、XC5、XCM
Hardware requirement	-	Software requirement	-

2: Operands

Operands	Function	Type
S	Specify the pulse number or soft components' ID	32bit, BIN
D	Specify the port to refresh the pulse	Bit

3: Suitable soft components

Word

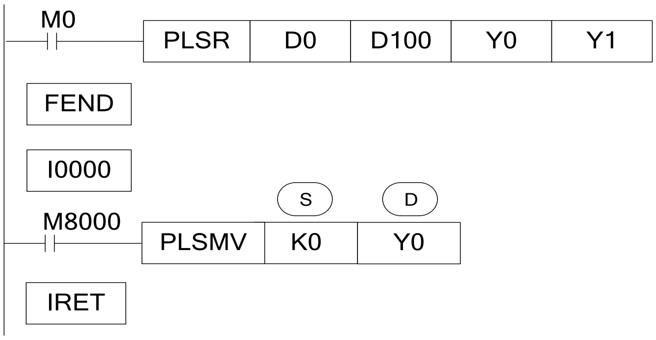
operands	system									constant	module	
	D	FD	ED	TD	CD	DX	DY	DM	DS	K/H	ID	QD
S	●	●		●	●					●		

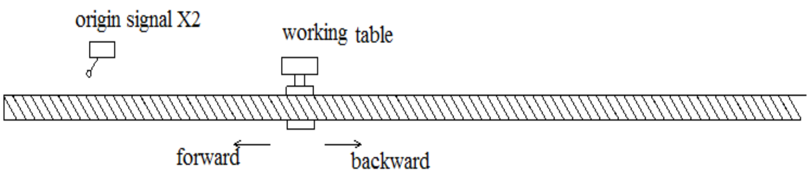
Bit

operands	system						
	X	Y	M	S	T	C	Dn.m
D		●					

Functions and Actions

《32 bit instruction form》





- When the working table is moving backward, it gets the origin signal X2, executes the external interruption, PLSMV command run immediately, this is not effected by the scan cycle. Refresh the pulse number from Y0 and send to D8170.
- This instruction is used remove the accumulation difference caused in pulse control.

6-2-7 Back to the Origin [ZRN]

1: Instruction Summary

Back to the Origin

Back to the Origin [ZRN]			
16 bits Instruction	ZRN	32 bits Instruction	DZRN

Execution condition	Normally ON/OFF coil	Suitable Models	XC2、XC3、XC5、XCM
Hardware requirement	-	Software requirement	-

2: Operands

Operands	Function	Type
S1	Specify the backward speed or soft components' ID	16/32bit, BIN
S2	Specify the creeping speed or soft components' ID	16/32 bit, BIN
S3	Specify the soft components' ID of the close point's signal	Bit
D	Specify the pulse output port	Bit

3: Suitable soft components

Word

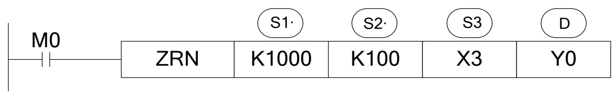
operands	system									constant	module	
	D	FD	ED	TD	CD	DX	DY	DM	DS	K/H	ID	QD
S1	●	●		●	●					●		
S2	●	●		●	●					●		

Bit

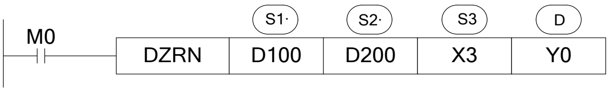
operands	system						
	X	Y	M	S	T	C	Dn.m
S3	●		●				
D		●					

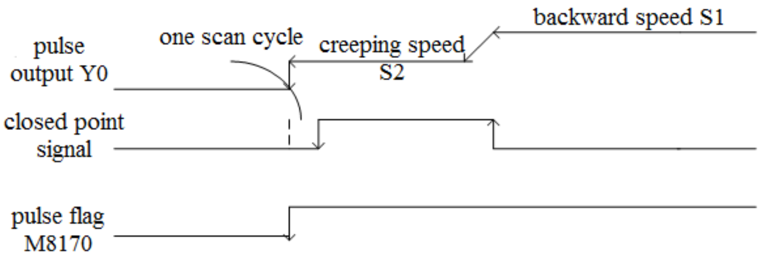
Functions and Actions

《16 bit instruction form》



《32 bit instruction form》





- Pulse output address: Y0 or Y1 only.
- S1 and S2 direction is same and the absolute value of S1 is greater than S2.
- After driving the instruction, move with the origin return speed S1.
- When the closed point signal turns from OFF to be ON, decrease the speed to be S2.
- When the closed point signal turns from ON to be OFF, write to registers (Y0:[D8171,D8170],Y1:[D8174,D8173]) when stopping pulse output.
- The decrease time can be specified by D8230~D8239; please refer to chapter 6-6 for details.

6-2-8 Relative position uni-segment pulse control [DRVI]

1:Instruction Summary

Relative position uni-segment pulse control;

Relative position uni-segment pulse control [DRVI]			
16 bits Instruction	DRVI	32 bits Instruction	DDRVI
Execution condition	Normally ON/OFF coil	Suitable Models	XC2、XC3、XC5、XCM
Hardware	-	Software	-

requirement		requirement	

2:Operands

Operands	Function	Type
S1	Specify the output pulse value or soft components ID	16/32bit, BIN
S2	Specify the output pulse frequency or soft components ID	16/32 bit, BIN
D1	Specify the pulse output port	Bit
D2	Specify the pulse output direction port	Bit

3: Suitable soft components

Word

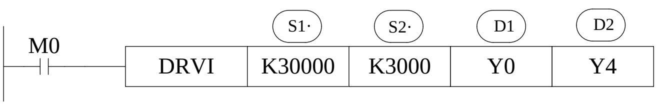
operands	system									constant	module	
	D	FD	ED	TD	CD	DX	DY	DM	DS	K/H	ID	QD
S1	●	●		●	●					●		
S2	●	●		●	●					●		

Bit

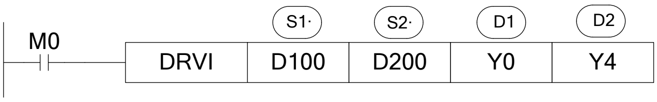
operands	system						
	X	Y	M	S	T	C	Dn.m
D1		●					
D2		●					

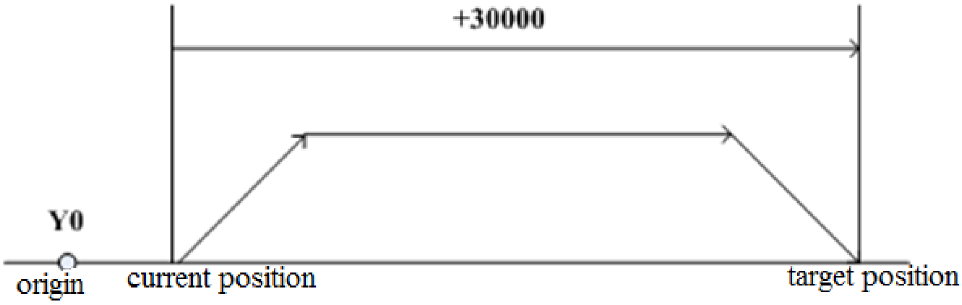
Functions and Actions

《16 bit instruction form》



《32 bit instruction form》





- Pulse output ID: only Y0 or Y1.
- Pulse output direction can specify any Y.
- Acceleration/deceleration time is specified by D8230 (single word).
- The relative drive form means: move from the current position.

6-2-9 Absolute position uni-segment pulse control [DRVA]

1:Instruction Summary

Absolute position uni-segment pulse control

Absolute position uni-segment pulse control [DRVA]			
16 bits Instruction	DRVA	32 bits Instruction	DDRVA
Execution condition	Normally ON/OFF coil	Suitable Models	XC2、XC3、XC5、XCM
Hardware requirement	-	Software requirement	-

2: Operands

Operands	Function	Type
S1	Specify the output pulse value or soft components ID	16/32bit, BIN
S2	Specify the output pulse frequency or soft components ID	16/32 bit, BIN
D1	Specify the pulse output port	Bit
D2	Specify the pulse output direction port	Bit

3: Suitable soft components

Word

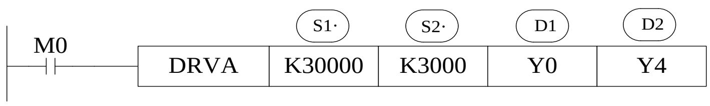
operands	system									constant	module	
	D	FD	ED	TD	CD	DX	DY	DM	DS	K/H	ID	QD
S1	•	•		•	•					•		
S2	•	•		•	•					•		

Bit

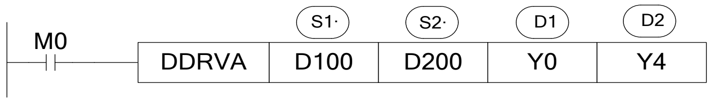
operands	system						
	X	Y	M	S	T	C	Dn.m
D1		•					
D2		•					

Functions and Actions

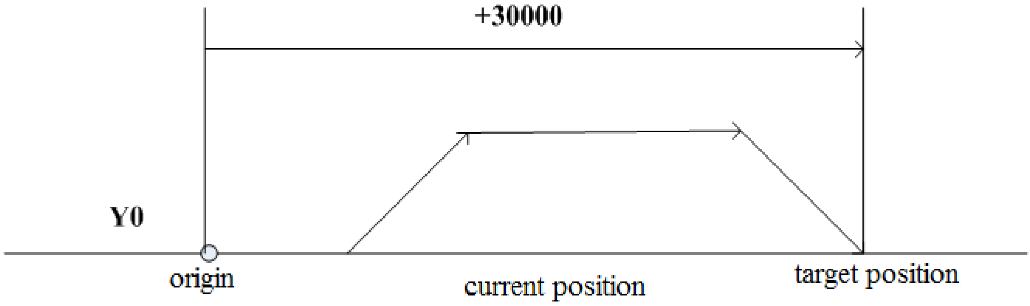
《16 bit instruction form》



《32 bit instruction form》



(Y0:[D8171,D8170],Y1:[D8174,D8173])



- Pulse output ID: only Y0 or Y1.
- Pulse output direction can specify any Y.
- Acceleration/deceleration time is specified by D8230 (single word).
- The relative drive form means: move from the origin position.
- Target position means S1, correspond with the following current value register as the absolute position.

6-2-10 Absolute position multi-segment pulse control [PLSA]

PLSA/DPLSA has two control modes, below we will introduce one by one;

➢ **Mode 1: uni-directional pulse output PLSA**

1: Instruction Summary

Generate absolute position segmented pulse with the specified frequency, acceleration/deceleration time and pulse direction;

Absolute position multi-segment pulse control [PLSA]			
16 bits Instruction	PLSA	32 bits Instruction	DPLSA

Execution condition	Normally ON/OFF coil	Suitable Models	XC2、XC3、XC5、XCM
Hardware requirement	-	Software requirement	-

2: Operands

Operands	Function	Type
S1	Specify the soft component's number to output the pulse parameters	16/32bit, BIN
S2	Specify the acceleration/deceleration time or soft component's number	16/32 bit, BIN
D	Specify the pulse output port	Bit

3: Suitable soft components

Word

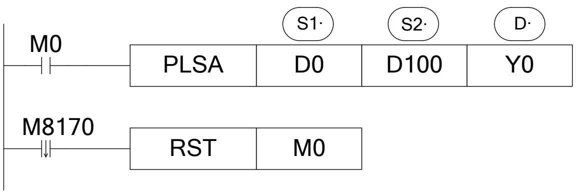
operands	system									constant	module	
	D	FD	ED	TD	CD	DX	DY	DM	DS	K/H	ID	QD
S1	●	●		●	●							
S2	●	●		●	●					K		

Bit

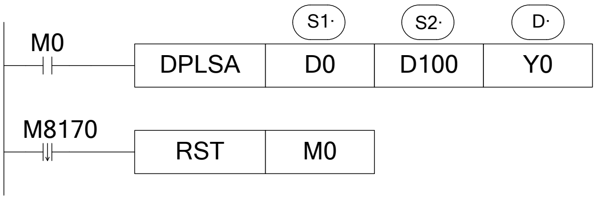
operands	system						
	X	Y	M	S	T	C	Dn.m
D1		●					

Functions and Actions

《16 bit instruction form》



《32 bit instruction form》



- The parameters' address is a section starts from **Dn** or **FDn**. In the above example: **D0** shows the first segment pulse's highest frequency; **D1** shows the first segment's absolute position; **D2** shows the second segment pulse's highest frequency; **D3** shows the second segment's absolute position，...... if the set value in **Dn, Dn+1** is 0, this represents the end of segment. Up to a maximum of 24 segments can be set.
- Acceleration/deceleration time is the time from the start to the first segment's highest frequency. Meantime, it defines the slope of all segment's frequency to time. In this way the following acceleration/deceleration will perform according to this slope.
- Pulse can be output at only Y000 or Y001

➢ **Mode 2: dual-directional pulse output PLSA**

1: Instruction Summary

Generate absolute position pulse with the specified frequency, acceleration/deceleration time and pulse direction;

Absolute position multi-segment pulse control [PLSA]			
16 bits Instruction	PLSA	32 bits Instruction	DPLSA
Execution condition	Normally ON/OFF coil	Suitable Models	XC2、XC3、XC5、XCM
Hardware	-	Software	-

requirement		requirement	

2: Operands

Operands	Function	Type
S1	Specify the soft component's number to output the pulse parameters	16/32bit, BIN
S2	Specify the acceleration/deceleration time or soft component's number	16/32 bit, BIN
D1	Specify the pulse output port	Bit
D2	Specify the pulse direction port	Bit

3、suitable soft components

Word

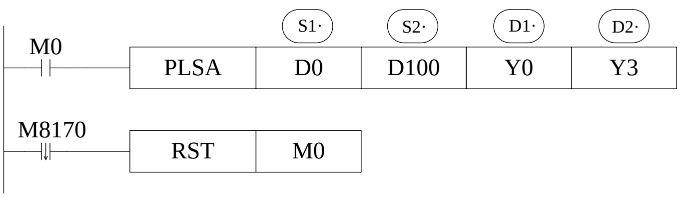
operands	system									constant	module	
	D	FD	ED	TD	CD	DX	DY	DM	DS	K/H	ID	QD
S1	●	●		●	●							
S2	●	●		●	●					K		

Bit

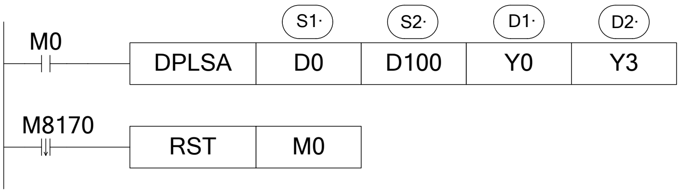
operands	system						
	X	Y	M	S	T	C	Dn.m
D1		●					
D2		●					

Functions and Actions

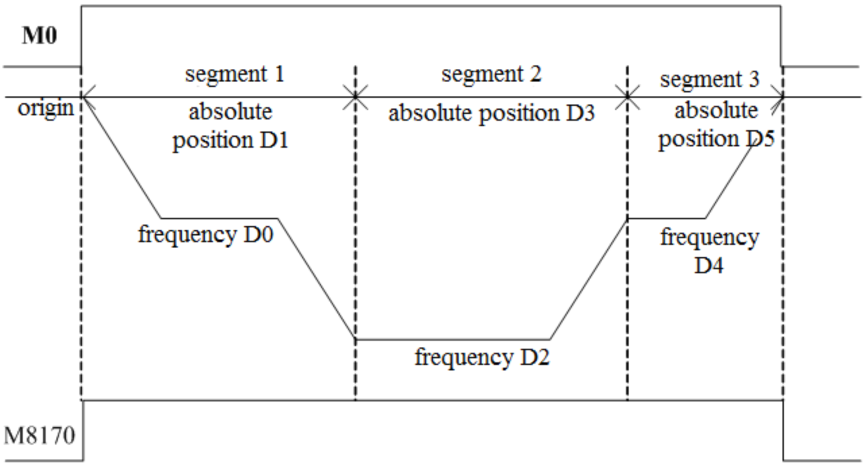
《16 bit instruction form》



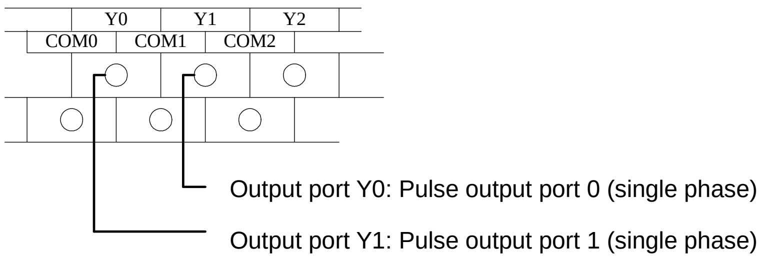
《32 bit instruction form》



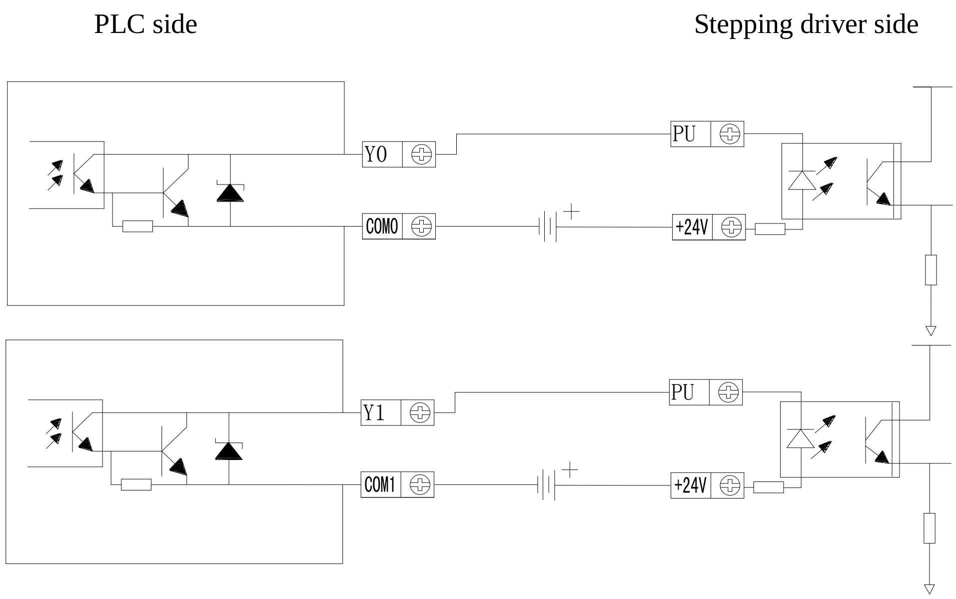
- The parameters' address is a section starts from **Dn** or **FDn**. In the above example: **D0** shows the first segment pulse's highest frequency; **D1** sshows the first segment's absolute position; **D2** shows the second segment pulse's highest frequency; **D3** shows the second segment's absolute position，…… if the set value in **Dn, Dn+1** is 0, this represents the end of segment. Up to a mximum of 24 segments can be set.
- Acceleration/deceleration time is the time from the start to the first segment's highest frequency. Meantime, it defines the slope of all segment's frequency to time. In this way the following acceleration/deceleration will perform according to this slope.
- Pulse can be output at only Y000 or Y001
- The Y port to output the pulse direction can be set freely;



6-3 Output Wiring

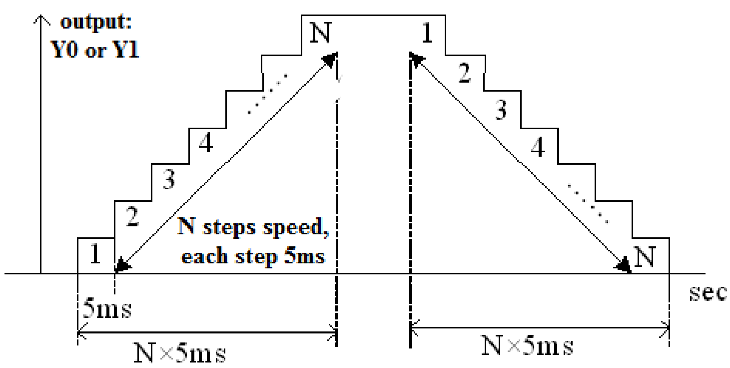


Below is the graph to show the output terminals and stepping driver wiring:



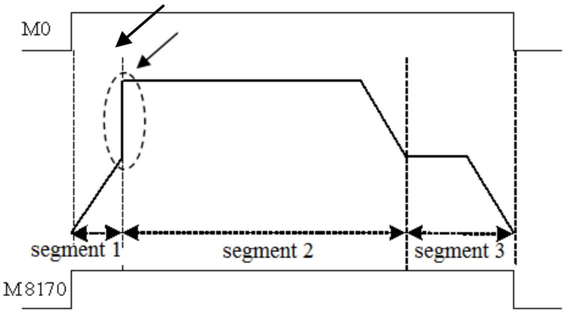
6-4 Items to Note

1: Concept of Step Frequency



- During ACC/DEC, each step time is 5ms, this time is fixed and not changeable.
- The minimum step frequency (each step’s rising/falling time) is 10Hz. If the frequency is lower than 10Hz, calculate as 10Hz; the maximum step frequency is 15Hz. If the frequency is larger than 15Hz, calculate as 15Hz.
- In case of frequency larger than 200Hz, please make sure each segment’s pulse

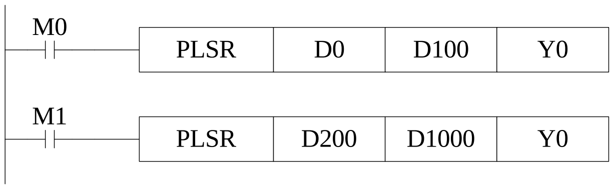
2、frequency jump in segment pulse output



- When outputting the segmented pulse, if the current segment's pulse has been set out, while meantime it doesn't reach the highest frequency, then from the current segment to the next pulse output segment, pulse jump appears, see graph above;

3: Dual pulse output is invalid

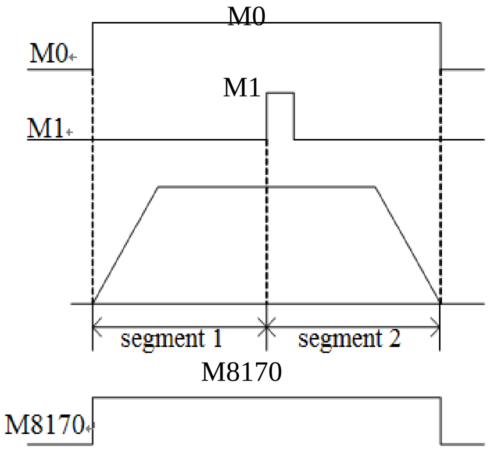
- In one main program, users can't write two or more pulse output instructions with one output port Y;
- Therefore the sample below is wrong;



6-5 Sample Programs

E.g.1: Stop at certain length

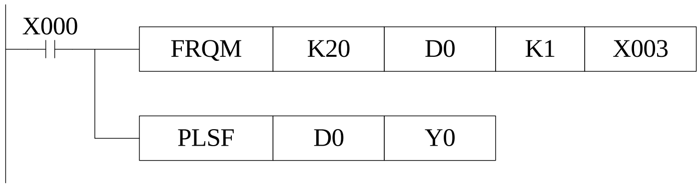
With instruction [PLSR] and [PLSNEXT], realize this "stop at certain length" function;



Take the sample program as the example, set two segments pulse output in D0, D1 and D2，D3, with the same frequency value; In second segment pulse output, set pulse number D3 as the output pulse number after receive M1 signal. This will realize “stop at certain length” function. See graph on the left.

E.g.2: follow function

In this sample, the pulse frequency from Y0 equals with the frequency tested from X003. If the frequency tested from X003 changes, the pulse frequency from Y0 changes;



6-6 Relative coils and registers of pulse output

Some flags of pulse output are listed below:

ID	Pulse ID	Function	Specification
M8170	PULSE_1	“sending pulse” flag	Being ON when sending the pulse,

M8171		overflow flag of "32 bits pulse sending"	When overflow, Flag is on
M8172		Direction flag	1 is positive direction, the correspond direction port is on
M8173	PULSE_2	"sending pulse" flag	Being ON when sending the pulse,
M8174		overflow flag of "32 bits pulse sending"	When overflow, Flag is on
M8175		Direction flag	1 is positive direction, the correspond direction port is on
M8176	PULSE_3	"sending pulse" flag	Being ON when sending the pulse,
M8177		overflow flag of "32 bits pulse sending"	When overflow, Flag is on
M8178		Direction flag	1 is positive direction, the correspond direction port is on
M8179	PULSE_4	"sending pulse" flag	Being ON when sending the pulse,
M8180		overflow flag of "32 bits pulse sending"	When overflow, Flag is on
M8181		Direction flag	1 is positive direction, the correspond direction port is on
M8210	PULSE_1	Pulse alarm flag (frequency change suddenly)	**1** is alarm, **0** is correct
M8211		Neglect the alarm or not	When flag is 1, stop sending alarm
M8212	PULSE_2	Pulse alarm flag (frequency change suddenly)	**1** is alarm, **0** is correct
M8213		Neglect the alarm or not	When flag is 1, stop sending alarm
M8214	PULSE_3	Pulse alarm flag (frequency change suddenly)	**1** is alarm, **0** is correct
M8215		Neglect the alarm or not	When flag is 1, stop sending alarm
M8216	PULSE_4	Pulse alarm flag (frequency change suddenly)	**1** is alarm, **0** is correct
M8217		Neglect the alarm or not	When flag is 1, stop sending alarm
M8218	PULSE_5	Pulse alarm flag (frequency change suddenly)	**1** is alarm, **0** is correct
M8219		Neglect the alarm or not	When flag is 1, stop sending alarm

Some special registers of pulse output are listed below:

ID	Pulse ID	Function	Specification
D8170	PULSE_1	The low 16 bits of accumulated pulse number	
D8171		The high 16 bits of accumulated pulse number	
D8172		The current segment (means Nr.n segment)	
D8173	PULSE_2	The low 16 bits of accumulated pulse number	

D8174		The high 16 bits of accumulated pulse number	
D8175		The current segment (means Nr.n segment)	
D8176	PULSE_3	The low 16 bits of accumulated pulse number	
D8177		The high 16 bits of accumulated pulse number	
D8178		The current segment (means Nr.n segment)	
D8179	PULSE_4	The low 16 bits of accumulated pulse number	
D8180		The high 16 bits of accumulated pulse number	
D8181		The current segment (means Nr.n segment)	
D8190	PULSE_1	The low 16 bits of the current accumulated current pulse number	
D8191		The high 16 bits of the current accumulated current pulse number	
D8192	PULSE_2	The low 16 bits of the current accumulated current pulse number	
D8193		The high 16 bits of the current accumulated current pulse number	
D8194	PULSE_3	The low 16 bits of the current accumulated current pulse number	Only XC5-32RT-E (4PLS) model has
D8195		The high 16 bits of the current accumulated current pulse number	
D8196	PULSE_4	The low 16 bits of the current accumulated current pulse number	
D8197		The high 16 bits of the current accumulated current pulse number	
D8210	PULSE_1	The error pulse segment’s position	
D8212	PULSE_2	The error pulse segment’s position	
D8214	PULSE_3	The error pulse segment’s position	
D8216	PULSE_4	The error pulse segment’s position	
D8218	PULSE_5	The error pulse segment’s position	

Absolute position/relative position/back to origin;

ID	Pulse	Function	Description
D8230	PULSE_1	Rising time of the absolute/relation position instruction (Y0)	
D8231		Falling time of the origin return instruction (Y0)	
D8232	PULSE_2	Rising time of the absolute/relation position instruction (Y1)	

D8233		Falling time of the origin return instruction (Y1)	
D8234	PULSE_3	Rising time of the absolute/relation position instruction (Y2)	
D8235		Falling time of the origin return instruction (Y2)	
D8236	PULSE_4	Rising time of the absolute/relation position instruction (Y3)	
D8237		Falling time of the origin return instruction (Y3)	
D8238	PULSE_5	Rising time of the absolute/relation position instruction	
D8239		Falling time of the origin return instruction	

7 Communication Function

This chapter includes: basic concepts of communication, Modbus communication, free communication and CAN-bus communication;

Relative Instructions:

Mnemonic	Function	Circuit and Soft Components	Chapter
MODBUS Communication			
COLR	Coil Read	COLR S1 S2 S3 D1 D2	7-2-3
INPR	Input coil read	INPR S1 S2 S3 D1 D2	7-2-3
COLW	Single coil write	COLW D1 D2 S1 S2	7-2-3
MCLW	Multi-coil write	MCLW D1 D2 D3 S1 S2	7-2-3
REGR	Register read	REGR S1 S2 S3 D1 D2	7-2-3
INRR	Input register read	INRR S1 S2 S3 D1 D2	7-2-3
REGW	Single register write	REGW D1 D2 S1 S2	7-2-3
MRGW	Multi-register write	MRGW D1 D2 D3 S1 S2	7-2-3
Free Communication			
SEND	Send data	SEND S1 S2 n	7-3-2
RCV	Receive data	RCV S1 S2 n	7-3-2
CAN-bus Communication			
CCOLR	Read coil	CCOLR S1 S2 S3 D	7-4-4
CCOLW	Write coil	CCOLW D1 D2 D3 S	7-4-4
CREGR	Read register	CREGR S1 S2 S3 D	7-4-4
CREGW	Write register	CREGW D1 D2 D3 S	7-4-4

7-1 Summary

XC2-PLC, XC3-PLC, XC5-PLC main units can fulfill your requirements for communication and networking. They not only support simple networks (Modbus protocol, Free Communication protocol), but also support complicated networks.

XC2-PLC, XC3-PLC, XC5-PLC offer communication accessthat enables communication with peripheral devices (such as printers, instruments etc.) that have their own communication protocol.

XC2-PLC, XC3-PLC, XC5-PLC all support Modbus protocol and Free protocol however, the XC5-PLC also supports CAN-Bus functions.

7-1-1 COM Port

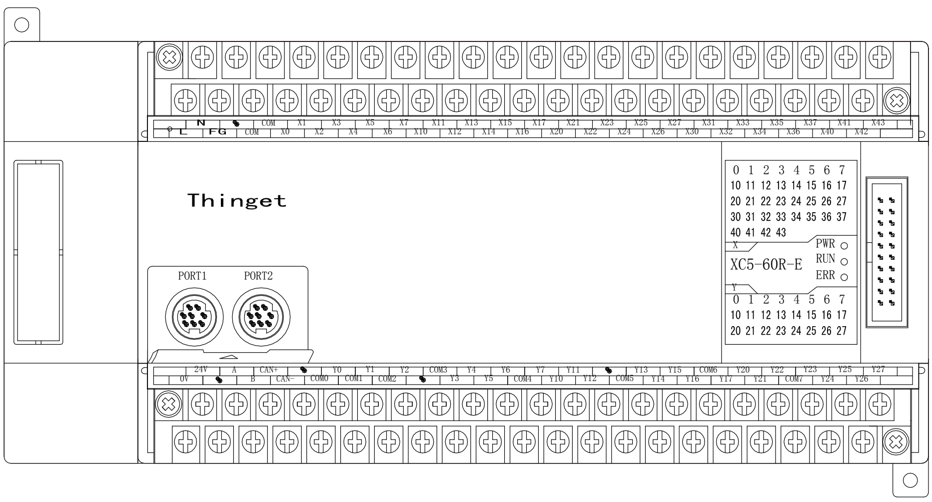
COM Port

There are 2 COM ports (Port1、Port2) on XC3 Series PLC basic units, while there are 3 COM ports on XC5 Series PLC main units. In addition to the same COM ports (COM1、COM2), they have also CAN COM port.

COM 1 (Port1) is the program port, it can be used to download the program and connect with the other devices. The parameters (baud rate, data bit etc.) of this COM port are fixed, can't be re-set.

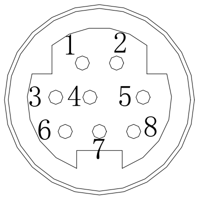
COM 2 (Port2) is communication port, it can be used to download a program and connect with the other devices. The parameters (baud rate, data bit etc.) of this COM port can be re-set via software.

Via BD cards, XC Series PLCs can accommodate other COM ports. These COM ports can be RS232 and RS485.



1: RS232 COM Port

- **COM1** **Pin Definition:**

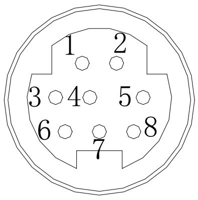


2：PRG
4：RxD
5：TxD

6：VCC
8：GND

Mini Din 8 pin female

COM2 **Pin Definition:**



4：RxD
5：TxD
8：GND

Mini Din 8 pin female

2: RS485 COM port:

For the RS485 COM port, A is “+” signal、B is “-“ signal.
The A, B terminals (RS485) on XC Series PLCs come from COM2, so, you cannot connect a device to the COM2 plug socket and also to the A & B terminals.

3: CAN COM port:

CAN port can be used to realize CAN-Bus communication. The pin terminals are “CAN+”, “CAN-“
For the detailed CAN communication functions, please refer to “6-8. CAN-Bus function (XC5 series)”

7-1-2 Communication Parameters

Communication Parameters

Station	Modbus Station number: 1~254、255 (FF) is free format communication
Baud Rate	300bps~115.2Kbps
Data Bit	8 bits data、7 bits data
Stop Bit	2 stop bits、1 stop bit
Parity	Even、Odd、No check

The default parameters of COM 1:
Station number is 1, baud rate is 19200bps, 8 data bit, 1 stop bit, Even

Parameters Setting

Set the parameters with the COM ports on XC series PLC;

	Number	Function	Description
COM 1	FD8210	Communication mode	255 is free format， **1~254 bit is Modbus station number**
	FD8211	Communication format	Baud rate, data bit, stop bit, parity
	FD8212	ASC timeout judgment time	Unit: ms，if set to be 0, it means no timeout waiting
	FD8213	Reply timeout judgment time	Unit: ms，if set to be 0, it means no timeout waiting
	FD8214	Start symbol	High 8 bits invalid
	FD8215	End symbol	High 8 bits invalid
	FD8216	Free format setting	8/16 bits cushion, with/without start bit, with/without stop bit
COM 2	FD8220	Communication mode	255 is free format， **1~254 bit is Modbus station number**
	FD8221	Communication format	Baud rate, data bit, stop bit, parity
	FD8222	ASC timeout judgment time	Unit: ms，if set to be 0, it means no timeout waiting
	FD8223	Reply timeout judgment time	Unit: ms，if set to be 0, it means no timeout waiting

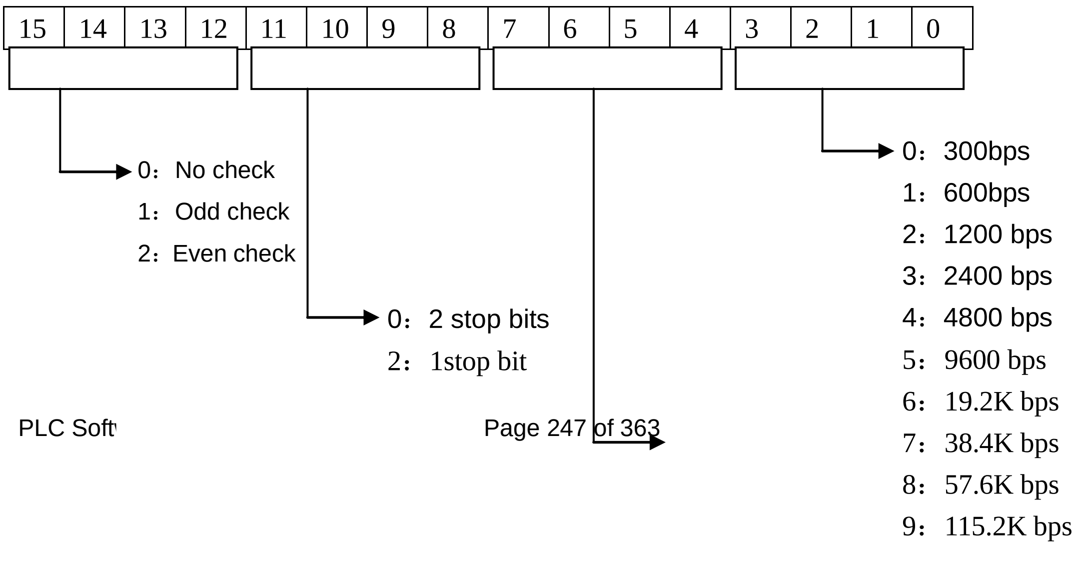
	FD8224	Start symbol	High 8 bits invalid
	FD8225	End symbol	High 8 bits invalid
	FD8226	Free format setting	8/16 bits cushion, with/without start bit, with/without stop bit
COM 3	FD8230	Communication mode	255 is free format， **1~254 bit is Modbus station number**
	FD8231	Communication format	Baud rate, data bit, stop bit, parity
	FD8232	ASC timeout judgment time	Unit: ms，if set to be 0, it means no timeout waiting
	FD8233	Reply timeout judgment time	Unit: ms，if set to be 0, it means no timeout waiting
	FD8234	Start symbol	High 8 bits invalid
	FD8235	End symbol	High 8 bits invalid
	FD8236	Free format setting	8/16 bits cushion, with/without start bit, with/without stop bit

※1: The PLC will be offline after changing the communication parameters, use “stop when reboot” function to keep PLC online.

※2: After modifying the data with special FLASH data registers, the new data will come into effect after reboot.

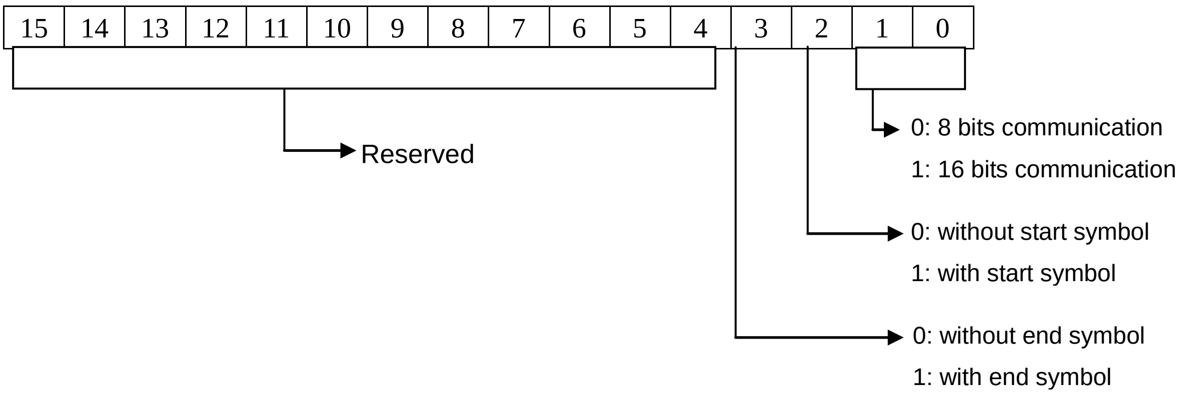
Set Communication Parameters

FD8211 (COM1)/FD8221 (COM2)/FD8231 (COM3)



0：8bits data
1：7bits data

FD8216 (COM1)/FD8226 (COM2)/FD8236 (COM3)



7-2 Modbus Communication

7-2-1 Function

XC Series PLCs support both Modbus master and Modbus slave.

MASTER FORMAT: When PLC is set to be master, PLC sends request to other slave devices via Modbus instructions, other devices respond to the master unit.

SLAVE FORMAT: when PLC is set to be slave, it can only communicate with master devices.

The default status of XC-PLC is Modbus slave.

7-2-2 Address

For the soft component's number in PLC which corresponds with Modbus address number, please see the following table:

Coil Space: (Modbus ID prefix is "0x")

Bit ID	ModbusID (decimal K)	Modbus ID (Hex. H)
M0~M7999	0~7999	0~1F3F
X0~X1037	16384~16927	4000~421F
Y0~Y1037	18432~18975	4800~4A1F
S0~S1023	20480~21503	5000~53FF
M8000~M8511	24576~25087	6000~61FF
T0~T618	25600~26218	6400~666A
C0~C634	27648~28282	6C00~6E7A

Register Space: (Modbus ID prefix is "4x")

Word ID	ModbusID (decimal K)	Modbus ID (Hex. H)
D0~D7999	0~7999	0~1F3F
TD0~TD618	12288~12906	3000~326A
CD0~CD634	14336~14970	3800~3A7A
D8000~D8511	16384~16895	4000~41FF
FD0~FD5000	18432~23432	4800~5B88
FD8000~FD8511	26624~27135	6800~69FF

※1: Bit soft components X、Y are in Octal form, the left are in decimal form.

7-2-3 Communication Instructions

Modbus instructions include coil read/write, register read/write; below, we describe these instructions in details:

➢ **Coil Read [COLR]**

1: Instruction Summary

Read the specified station's specified coil status to the local PLC;

Coil read [COLR]			
16 bits instruction	COLR	32 bits instruction	-
Execution Condition	Normally ON/OFF coil	Suitable Models	XC2、XC3、XC5、XCM
Hardware Requirement	-	Software Requirement	-

2: Operands

Operands	Function	Type
S1	Specify the remote communication station or soft component's ID	16bits, BIN
S2	Specify the remote coil's start ID or soft component's ID	16bits, BIN
S3	Specify the coil number or soft component's ID	16bits, BIN
D1	Specify the start ID of the local receive coils	bit
D2	Specify the serial port's number	16bits, BIN

3: Suitable soft components

Word

Operands	System									constant	module	
	D	FD	ED	TD	CD	DX	DY	DM	DS	K/H	ID	QD
S1	●	●		●	●					●		
S2	●	●		●	●					●		
S3	●	●		●	●					●		
D2										K		

Bit

Operands	Operands						
	X	Y	M	S	T	C	Dn.m
D1	●	●	●	●	●	●	

Function

X0		S1·	S2·	S3·	D1·	D2·
┤├	COLR	K1	K500	K3	M1	K2

- Read coil instruction, Modbus code is 01H。
- Serial Port: K1~K3

➢ **Input Coil Read [INPR]**

1: Instruction

Read the specified station's specified input coils into local coils:

Input coil read [INPR]			
16 bits instruction	INPR	32 bits instruction	-
Execution Condition	Normally ON/OFF、rising edge	Suitable Models	XC2、XC3、XC5、XCM
Hardware Requirement	-	Software Requirement	-

2: Operands

Operands	Function	Type
S1	Specify the remote communication station or soft component's ID	16bits, BIN
S2	Specify the remote coil's start ID or soft component's ID	16bits, BIN
S3	Specify the coil number or soft component's ID	16bits, BIN

D1	Specify the start ID of the local receive coils	bit
D2	Specify the serial port's number	16bits, BIN

3: Suitable Soft Components

Word

Operands	System									constant	module	
	D	FD	ED	TD	CD	DX	DY	DM	DS	K/H	ID	QD
S1	●	●		●	●					●		
S2	●	●		●	●					●		
S3	●	●		●	●					●		
D2										K		

Bit

Operands	System						
	X	Y	M	S	T	C	Dn.m
D1	●	●	●	●	●	●	

Function

X0		S1·	S2·	S3·	D1·	D2·
─┤├─	INPR	K1	K500	K3	M1	K2

- Instruction to read the input coil, Modbus code is 02H
- Serial port: K1~K3
- When X0 is ON, execute COLR or INPR instruction, set communication flag after execution of the instruction; when X0 is OFF, no operation. If error happens during communication, it resends automatically. If 3 errors are noted, the communication error flag will be set. The user can check the relative registers to judge the error.

➢ **Single Coil Write [COLW]**

1: Summary

Write the local coil status to the specified station's specified coil;

Single coil write [COLW]			
16 bits instruction	COLW	32 bits instruction	-
Execution Condition	Normally ON/OFF、rising edge	Suitable Models	XC2、XC3、XC5、XCM
Hardware Requirement	-	Software Requirement	-

2: Operands

Operands	Function	Type
D1	Specify the remote communication station or soft component's ID	16bits, BIN
D2	Specify the remote coil's start ID or soft component's ID	16bits, BIN
S1	Specify the start ID of the local receive coils	bit

S2	Specify the serial port's number	16bits, BIN

3: Suitable soft components

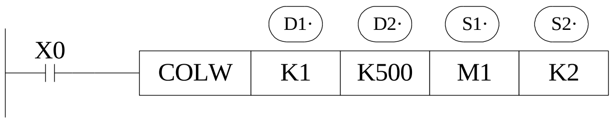
Word

Operands	System									constant	module	
	D	FD	ED	TD	CD	DX	DY	DM	DS	K/H	ID	QD
D1	●	●		●	●					●		
D2	●	●		●	●					●		
S2										K		

Bit

Operands	System						
	X	Y	M	S	T	C	Dn.m
S1	●	●	●	●	●	●	

Function



- Write the single coil, Modbus code is 05H
- Serial port: K1~K3

➢ Multi-coil Write [MCLW]

1:Summary

Write the local multi-coil status into the specified station's specified coil;

Multi-coil write [MCLW]			
16 bits instruction	MCLW	32 bits instruction	-
Execution Condition	Normally ON/OFF、rising edge	Suitable Models	XC2、XC3、XC5、XCM
Hardware Requirement	-	Software Requirement	-

2: Operands

Operands	Function	Type
D1	Specify the remote communication station or soft component's ID	16bits, BIN
D2	Specify the remote coil's start ID or soft component's ID	16bits, BIN

D3	Specify the coil number or soft component's ID	16bits, BIN
S1	Specify the start ID of the local receive coils	bit
S2	Specify the serial port's number	16bits, BIN

3: Suitable soft components

Word

Operands	System									constant	module	
	D	FD	ED	TD	CD	DX	DY	DM	DS	K/H	ID	QD
D1	●	●		●	●					●		
D2	●	●		●	●					●		
D3	●	●		●	●					●		
S2										K		

Bit

Operands	System						
	X	Y	M	S	T	C	Dn.m
S1	●	●	●	●	●	●	

Function

X0

	D1·	D2·	D3·	S1·	S2·
MCLW	K1	K500	K3	M1	K2

- Instruction to write the multiply coils, Modbus code is 0FH
- Serial port: K1~K3
- When X0 is ON, execute COLW or MCLW instruction, set communication flag after execution the instruction; when X0 is OFF, no operation. If error happens during communication, it resends automatically. If 4 errors are noted, the communication error flag will be set. The user can check the relative registers to judge the error.

➢ Register Read [REGR]

1: Summary

Read the specified station's specified register to the local register;

Register read [REGR]			
16 bits instruction	REGR	32 bits instruction	-
Execution Condition	Normally ON/OFF、rising edge	Suitable Models	XC2、XC3、XC5、XCM
Hardware Requirement	-	Software Requirement	-

2: Operands

Operands	Function	Type
S1	Specify the remote communication station or soft component's ID	16bits, BIN
S2	Specify the remote coil's start ID or soft component's ID	16bits, BIN

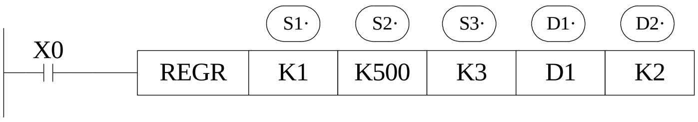
S3	Specify the coil number or soft component's ID	16bits, BIN
D1	Specify the start ID of the local receive coils	bit
D2	Specify the serial port's number	16bits, BIN

3: Suitable soft components

Word

Operands	System									constant	module	
	D	FD	ED	TD	CD	DX	DY	DM	DS	K/H	ID	QD
S1	●	●		●	●					●		
S2	●	●		●	●					●		
S3	●	●		●	●					●		
D1	●											
D2										K		

Function



- Instruction to read the REGISTERS, Modbus code is 03H
- Serial port: K1~K3

➢ Register Input Read [INNR]

1: Summary

Read the specified station's specified input register to the local register

Read Input Register [INRR]			
16 bits instruction	INRR	32 bits instruction	-
Execution Condition	Normally ON/OFF、rising edge	Suitable Models	XC2、XC3、XC5、XCM
Hardware Requirement	-	Software Requirement	-

2:Operands

Operands	Function	Type
S1	Specify the remote communication station or soft component's ID	16bits, BIN
S2	Specify the remote coil's start ID or soft component's ID	16bits, BIN

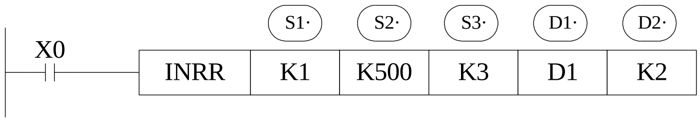
S3	Specify the coil number or soft component’s ID	16bits, BIN
D1	Specify the start ID of the local receive coils	bit
D2	Specify the serial port’s number	16bits, BIN

3: Suitable soft components

Word

Operands	System									constant	module	
	D	FD	ED	TD	CD	DX	DY	DM	DS	K/H	ID	QD
S1	●	●		●	●					●		
S2	●	●		●	●					●		
S3	●	●		●	●					●		
D1	●											
D2										K		

Function



- Instruction to read the input registers, Modbus code is 04H
- Serial port: K1~K3
- When X0 is ON, execute REGR or INRR instruction, set communication flag after execution the instruction; when X0 is OFF, no operation. If error happens during communication, it resends automatically. If 4 errors are noted, the communication error flag will be set. The user can check the relative registers to judge the error.

➢ **Single Register Write [REGW]**

1: Summary

Instruction to write the local specified register into the specified station’s specified register;

Single register write [REGW]			
16 bits instruction	REGW	32 bits instruction	-
Execution Condition	Normally ON/OFF、rising edge	Suitable Models	XC2、XC3、XC5、XCM
Hardware Requirement	-	Software Requirement	-

2: Operands

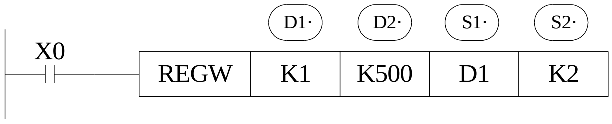
Operands	Function	Type
D1	Specify the remote communication station or soft component’s ID	16bits, BIN

D2	Specify the remote coil's start ID or soft component's ID	16bits, BIN
S1	Specify the start ID of the local receive coils	16bits, BIN
S2	Specify the serial port's number	16bits, BIN

3: Suitable soft components

Word	Operands	System									constant	module	
		D	FD	ED	TD	CD	DX	DY	DM	DS	K/H	ID	QD
	D1	●	●		●	●					●		
	D2	●	●		●	●					●		
	S1	●											
	S2										K		

Function



- Write the single register, Modbus code is 06H
- Serial port: K1~K3

➢ Multi-register write [MRGW]

1:Summary

Instruction to write the local specified register to the specified station's specified register;

Multi-register write [MRGW]			
16 bits instruction	MRGW	32 bits instruction	-
Execution Condition	Normally ON/OFF 、 rising edge	Suitable Models	XC2、XC3、XC5、XCM
Hardware Requirement	-	Software Requirement	-

2: Operands

Operands	Function	Type
D1	Specify the remote communication station or soft component's ID	16bits, BIN

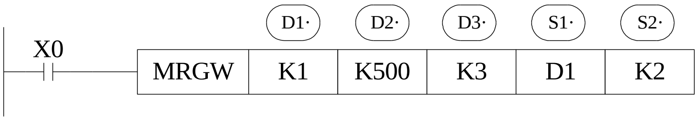
D2	Specify the remote coil’s start ID or soft component’s ID	16bits, BIN
D3	Specify the coil number or soft component’s ID	16bits, BIN
S1	Specify the start ID of the local receive coils	bit
S2	Specify the serial port’s number	16bits, BIN

3: Suitable soft components

Word

Operands	System									constant	module	
	D	FD	ED	TD	CD	DX	DY	DM	DS	K/H	ID	QD
D1	●	●		●	●					●		
D2	●	●		●	●					●		
S1	●											
S2										K		

Function



- Instruction to write the multiply registers, Modbus code is 10H
- Serial port: K1~K3

When X0 is ON, execute REGW or MRGW instruction, set communication flag after execution the instruction; when X0 is OFF, no operation. If error happens during communication, it resends automatically. If 4 errors are noted, the communication error flag will be set. The user can check the relative registers to judge the error.

7-3 Free Format Communication

7-3-1 Communication Mode

Free format communication transfer data in the form of data block, each block can transfer a maximum of 128 bytes. Each block can set a start symbol and stop symbol, or not set.

Communication Mode:

Start Symbol (1 byte)	Data Block (max. 128 bytes)	End Symbol (1 byte)

- Port1, Port2 or Port3 can realize free format communication
- Under free format form, FD8220 or FD8230 should set to be 255 (FF)
- Baud Rate: 300bps~115.2Kbps
- Data Format
  Data Bit: 7bits、8bits
  Parity: Odd, Even, No Check
  Stop bit: 1 bit,2 bits
- Start Symbol: 1 bit
  Stop Symbol: 1 bit
  User can set a start/stop symbol, after set the start/stop symbol, PLC will automatically add this start/stop symbol when sending data; remove this start/stop symbol when receiving data.
- Communication Format: 8 bits,16 bits
  If utilizing 8 bits buffer format to communicate, within the communication process, the high bytes are invalid, PLCs only use the low bytes to send and receive data.
  If utilizing 16 bits buffer format to communicate, when PLC is sending data, PLC will send low bytes before sending higher bytes

7-3-2 Instruction Form

➢ **Send Data [SEND]**

1: Summary

Write the local specified data to the specified station's specified ID;

Send data [SEND]			
16 bits instruction	SEND	32 bits instruction	-
Execution Condition	Normally ON/OFF 、 rising edge	Suitable Models	XC2、XC3、XC5、XCM
Hardware Requirement	-	Software Requirement	-

2: Operands

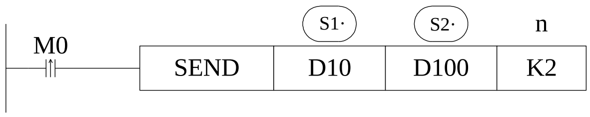
Operands	Function	Type
S1	Specify the start ID of local PLC	16bits, BIN
S2	Specify the ASC number to send or soft component's ID	16bits, BIN
n	Specify the COM port Nr.	16bits, BIN

3: Suitable soft components

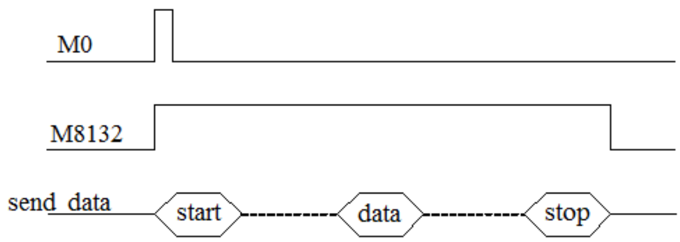
Word

Operands	System									constant	module	
	D	FD	ED	TD	CD	DX	DY	DM	DS	K/H	ID	QD
S1	●	●		●	●							
S2	●	●		●	●					●		
n	●									K		

Function



- Data send instruction, send data on the rising edge of M0;
- Serial port: K2~K3
- When sending data, set "sending" flag M8132 (COM2) ON



➢ **Receive Data [RCV]**

1: Summary

Write the specified station's data to the local specified ID;

Receive data [RCV]			
16 bits instruction	RCV	32 bits instruction	-
Execution Condition	Normally ON/OFF 、 rising edge	Suitable Models	XC2、XC3、XC5、XCM
Hardware Requirement	-	Software Requirement	-

2: Operands

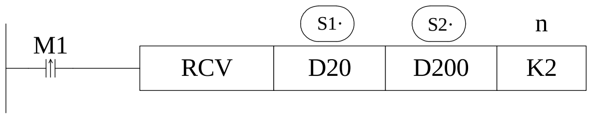
Operands	Function	Type
S1	Specify the start ID of local PLC	16bits, BIN
S2	Specify the ASC number to receive or soft component's ID	16bits, BIN
n	Specify the COM port Nr.	16bits, BIN

3: Suitable soft components

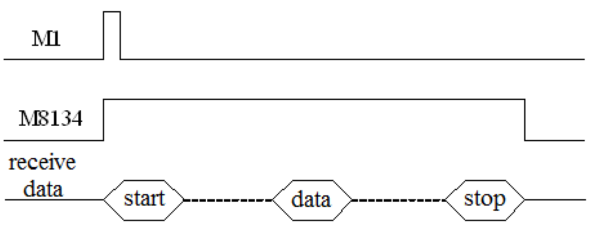
Word

Operands	System									constant	module	
	D	FD	ED	TD	CD	DX	DY	DM	DS	K/H	ID	QD
S1	●	●		●	●							
S2	●	●		●	●					●		
n										●		

Function



- Data receive instruction, receive data on the rising edge of M0;
- Serial port: K2~K3
- When receiving data, set "receiving" flag M8134(COM2) ON

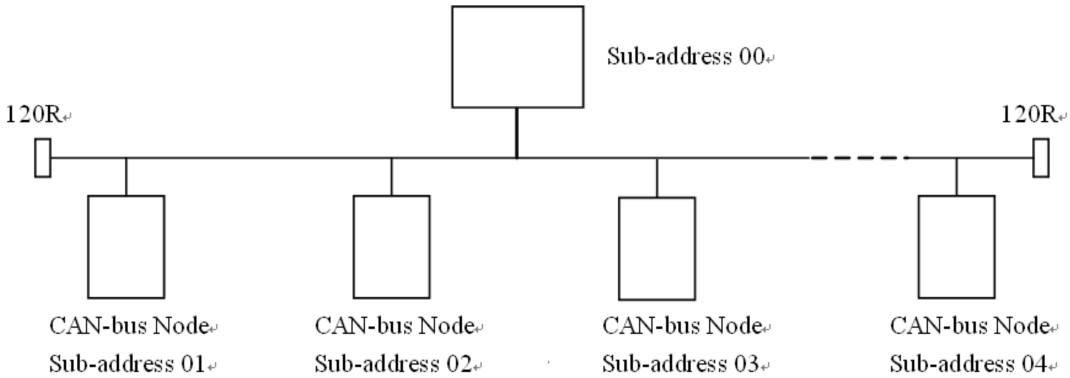


※1: If you require PLC to receive but not send, or receive before send, you need to set the communication timeout as 0ms

7-4 CAN-Bus Format

7-4-1 Brief Introduction of CAN-Bus

XC5 Series PLCs support CAN-Bus functions. Below we will give some basic concept on CAN-Bus;



CAN (Controller Area Network) belongs to the industrial area bus category. Compared with common communication bus, CAN-Bus data communication has performance of outstanding dependability, real time ability and flexibility.

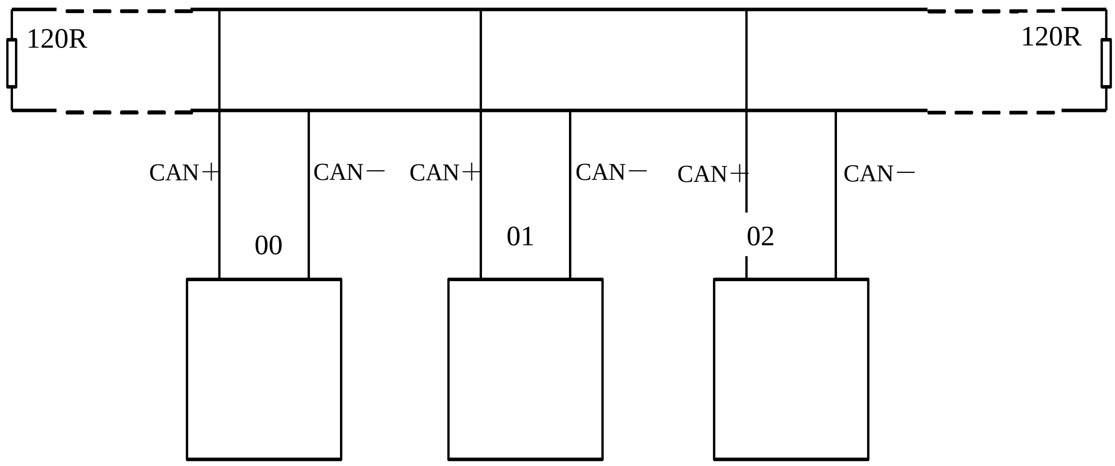
CAN controller works under multi-master format. In the network, each node can send data to the bus according to the bus visit priority. These characters enable each node in the CAN-Bus network to have stronger data communication real time performance, and easy to construct a redundant structure, improving the system's dependability and flexibility.

In CAN-Bus networks, any node can initiatively send message at any time to any other node, no master and no slave. Enabling flexible communication; it's easy to compose multi-device backup system, distributing format monitor, control system. To fulfill different real time requirements, the nodes can be divided to be different priority levels. With non-destroy bus arbitrament technology, when two nodes send message to the network at the same time, the low level priority node intuitively stops data sending, while high level priority node can continue transferring data without any influence. This gives functions of node to node, node to multi-node, bureau broadcasting sending/receiving data. Each frame's valid byte number is 8, so the transfer time is short, the probability ratio is low.

7-4-2 External Wiring

CAN-Bus Communication Port: CAN＋、CAN－

The wiring among each node of CAN-Bus is shown in the following graph; at the two ends, add 120 ohm middle-terminal resistors.



7-4-3 CAN-Bus Network Form

There are two forms of CAN-Bus network: one is instructions communication format; the other is internal protocol communication format. These two forms can work at the same time

- Instructions communication format

This format means, in the local PLC program, via CAN-Bus instructions, execute bit or word reading/writing with the specified remote PLC.

- Internal protocol communication format

This format means, via setting of special register, via configure table format, realize allude with each other among PLC's certain soft component's space. In this way, realize PLC source sharing in CAN-Bus network.

7-4-4 CAN-Bus Instructions

- **Read Coil [CCOLR]**

1:Instruction Description

Function：Read the specified station’s specified coil status into the local specified coil.

Read Coil [CCOLR]			
16 bits instruction	CCOLR	32 bits instruction	-
Execution Condition	Normally ON/OFF, rising edge activates	Suitable Models	XC5
Hardware Requirement	-	Software Requirement	-

2: Operands

Operands	Function	Type
S1	Specify remote communication station ID or soft component’s number;	16bits, BIN
S2	Specify the remote coil’s start ID or soft component’s number;	16bits, BIN
S3	Specify the coil number or soft component’s number;	16bits, BIN
D	Specify the local receive coil’s start ID	bit

3: Suitable Soft Components

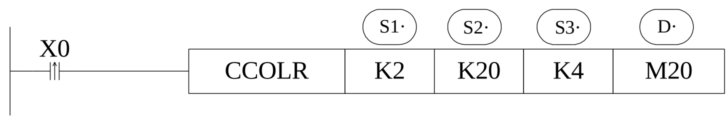
Word

Operands	System									Constant	Module	
	D	FD	ED	TD	CD	DX	DY	DM	DS	K/H	ID	QD
S1	●	●		●	●					●		
S2	●	●		●	●					●		
S3	●	●		●	●					●		

Bit

Operands	System						
	X	Y	M	S	T	C	Dn.m
D	●	●	●	●	●	●	

Function



- Execute CCOLR instruction when X0 changes from OFF to ON; read the four coils data of remote station at address 2, coil’s start ID K20 to local M20～M23.

➢ **Write the Coil [CCOLW]**

1: Summary

Write the local specified multi-coils status into the specified station’s specified coils;

Write the coil [CCOLW]			
16 bits	CCOLW	32 bits	-

instruction		instruction	
Execution Condition	Normally ON/OFF 、 rising edge	Suitable Models	XC5
Hardware Requirement	-	Software Requirement	-

2: Operands

Operands	Function	Type
D1	Specify remote communication station ID or soft component's number;	16 BIN
D2	Specify the remote coil's start ID or soft component's number;	16 BIN
D3	Specify the coil number or soft component's number;	16 BIN
S	Specify the local receive coil's start ID	Position

3: Suitable soft components

Word

Operands	System									constant	module	
	D	FD	ED	TD	CD	DX	DY	DM	DS	K/H	ID	QD
S1	●	●		●	●					●		
S2	●	●		●	●					●		
S3	●	●		●	●					●		

Bit

Operands	System					
	X	Y	M	S	T	C	Dn.m
D	●	●	●	●	●	●	

Function

X0		D1·	D2·	D3·	S·
─┤↑├─	CCOLW	K2	K20	K4	M20

- Execute CCOLW instruction when X0 changes from OFF to ON; write the local M20～M23 to the remote station 20th, coil's start ID K20.

➢ Read Register [CREGR]

1: Summary

Read the specified station's specified register to the local specified register;

Read register [CREGR]			
16 bits instruction	CREGR	32 bits instruction	-
Execution	Normally ON/OFF、rising edge	Suitable Models	XC5

Condition			
Hardware Requirement	-	Software Requirement	-

2: Operands

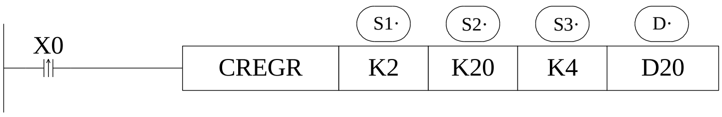
Operands	Function	Type
D1	Specify remote communication station ID or soft component's number;	16bits, BIN
D2	Specify the remote register's start ID or soft component's number;	16bits, BIN
D3	Specify the register number or soft component's number;	16bits, BIN
S	Specify the local receive coil's start ID	16bits, BIN

3: Suitable soft components

Word

Operands	System									constant	module	
	D	FD	ED	TD	CD	DX	DY	DM	DS	K/H	ID	QD
S1	●	●		●	●					●		
S2	●	●		●	●					●		
S3	●	●		●	●					●		
D	●			●	●							

Function



- Execute CREGR instruction when X0 changes from OFF to ON; read the remote station 2th, coil's start ID K20 to the local D20～D23

➢ **Write the Register [CREGW]**

1: Summary

Write the specified local input register to the specified station's specified register;

Write the register [CREGW]			
16 bits instruction	CREGW	32 bits instruction	-
Execution Condition	Normally ON/OFF、rising edge	Suitable Models	XC5

Hardware Requirement	-	Software Requirement	-

2: Operands

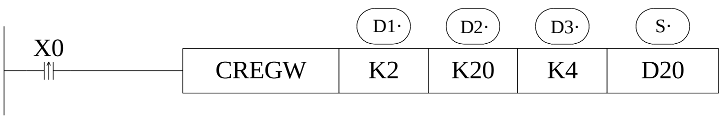
Operands	Function	Type
D1	Specify remote communication station ID or soft component's number;	16bits, BIN
D2	Specify the remote register's start ID or soft component's number;	16bits, BIN
D3	Specify the register number or soft component's number;	16bits, BIN
S	Specify the local receive coil's start ID	16bits, BIN

3: Suitable soft components

Word

Operands	System									constant	module	
	D	FD	ED	TD	CD	DX	DY	DM	DS	K/H	ID	QD
S1	•	•		•	•					•		
S2	•	•		•	•					•		
S3	•	•		•	•					•		
D	•			•	•							

Function



- Execute CREGW instruction when X0 changes from OFF to ON; write the local D20～D23 to the remote station 2th, coil's start ID K20.

7-4-5 Communication Form of Internal Protocol

Function

- Open/close the internal protocol communication function
  Set the value in register FD8350:
  0: do not use CAN internal protocol communication;
  1: use CAN internal protocol communication
  CAN internal protocol communication is default to be closed

- Set the communication parameters

See the setting methods with baud rate, station number, sending frequency etc. in the below table:

Define the configure items:

Internal protocol communication is to communicate via setting the configure items;

The configure items include: read the bit, read the word, write the bit, write the word;

The configure form:

Step 1: add the four configure items numbers separately: FD8360—read the bit items; FD8361—read the word items; FD8362—write the bit items; FD8363—write the word items.

Step 2: set each configure item's communication object, each item includes four parameters: remote node's station; remote node's object ID; local object's ID; number; the corresponding register ID is: FD8370~FD8373 represents Nr.1 item; FD8374~FD8377 represents Nr.2 item, ……FD9390~FD9393 represents Nr.256 item. A maximum of 256 items can be set;

see tables below:

Communication Setting

Nr.	Function	Description
FD8350	CAN communication mode	0 represents **not use**; 1 represents internal protocol
FD8351	CAN baud rate	See CAN baud rate setting table
FD8352	Self CAN station	For CAN protocol use (the default value is 1)

FD8354	Configured sending frequency	The set value's unit is **ms**, represents "send every **ms**" if set to be 0, it means send every cycle, the default value is 5ms
FD8360	Read bit number	-
FD8361	Read word number	
FD8362	write bit number	
FD8363	write word number	
FD8370	Remote node's ID	The Nr.1 item's configuration
FD8371	Remote node's object ID	
FD8372	Local object's ID	
FD8373	Number	
……	……	……
FD9390	Remote node's ID	The Nr.256 item's configuration
FD9391	Remote node's object ID	
FD9392	Local object's ID	
FD9393	Number	

Status Flag

M8240	CAN self check error flag	Set 1 if error; set 0 if correct
M8241	Error flag of CAN configure	Set 1 if error; set 0 if correct
M8242	Automatically recover the control	If set to be 1, then recover after error happens; If set to be 1, then CAN stops working

Baud Rate Setting

FD8351 value	Baud Rate (BPS)
0	1K
1	2K
2	5K
3	10K
4	20K
5	40K

Register Status

D8240	CAN error information	0: no error 2: initialize error 30: bus error 31: error alarm 32: data overflow
D8241	The configure item's Nr. which has error	Show the first number of error configure item
D8242	Data package number sent every second	-
D8243	Data package number received every second	-
D8244	CAN communication error count	-

7-4-6 CAN Free Format Communication

- **CAN Sending [CSEND]**

1: Instructions Summary

Write the specified data from the unit to a specified address (data transfer in one unit)

CAN Sending [CSEND]

16bits instruction	CSEND	32bits instruction	-
Executing Condition	Normally ON/OFF、Rising edge	Suitable Models	XC5
Hardware Requirement	-	Software Requirement	-

2: Operands

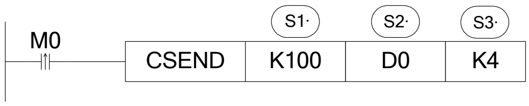
Operands	Function	Type
S1	specify the ID number to send the data package	16bits, BIN
S2	specify the first ID number of sent data or soft component locally	16bits, BIN
S3	specify the byte number of sent data	16bits, BIN

3: Suitable soft components

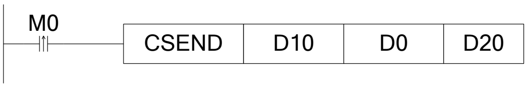
Word type

Operands	System									constant	module	
	D	FD	ED	TD	CD	DX	DY	DM	DS	K/H	ID	QD
S1	●	●		●	●					●		
S2	●	●		●	●							
S3	●	●		●	●					●		

Functions and Actions



- Instruction to enable data sending, send data at every rising edge of M0
- ID number of sending data package is 100, 4 bytes data, the first ID is in D0
- 8 bits data transfer: the transferred data is: D0L, D1L, D2L, D3L (D0L means the low byte of D0)
- 16 bits data transfer: the transferred data is: D0L, D0H, D1L, D1H (D0H means the high byte of D0)



- The ID of sending data package is specified by D10, the data number is specified by D20, the first ID is in D0;
- 8 bits data transfer: the transferred data is: D0L, D1L, D2L, D3L (D0L means the low byte of D0)
- 16 bits data transfer: the transferred data is: D0L, D0H, D1L, D1H (D0H means the high byte of D0)
- Standard Frame: the valid bits of the data package ID number that is specified by D10 is the low 11 bits, the left bits are invalid;
- The expansion frame: the valid bits of the data package ID number that is specified by D10 is the low 29 bits, the left bits are invalid;
- The maximum data bits specified by D20 is 8, if exceeds 8, the instruction will send only 8 bits;

➢ CAN Receive [CRECV]

1: Instructions Summary

Write the specified data in one unit to a specified address in another unit (data transfers between different units)

CAN Receive [CRECV]			
16 bits instruction	CRECV	32 bits instruction	-
Executing Condition	Normally ON/OFF 、 Rising edge	Suitable Models	XC5
Hardware Requirement	-	Software Requirement	-

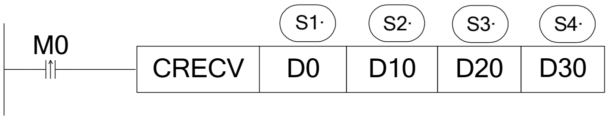
2: Operands

Operands	Function	Type
S1	specify the ID number to receive the data package	16bits, BIN
S2	specify the first ID number of received soft component locally	16bits, BIN
S3	specify the byte number of received data	16bits, BIN
S4	specify the soft component's start ID number of ID filter code	16bits, BIN

3: Suitable soft components

Word Type	Operands	System									Constant	Module	
		D	FD	ED	TD	CD	DX	DY	DM	DS	K/H	ID	QD
	S1	●	●		●	●							
	S2	●	●		●	●							
	S3	●	●		●	●							
	S4	●											

Functions and Actions



- The 32 bits memory combined by [D1, D0] (D0 is low byte, D1 is high byte) is used to stock ID number of the received data package. The received data length is stored in D20. The data content is stored in registers start from D10. D30 specifies the received ID filter code; if the received data doesn't fit the filter codes, then it will keep the RECV status;
- ID filter code: D30 specifies the start address of ID filter codes; the instruction specifies two groups of filter codes, occupy D30~D37 zone;

Filter Code	Memory	Description	Example
The first group	D31, D30	D30 low bytes, D31 high bytes, they compose a 32 bits mask code	D30=0xFFFF, D31=0x0000, then the mask code is 0x0000FFFF D30=0x1234, D31=0x0000, then filter value is 0x00001234 If ID and 0x0000FFFF equals 0x00001234, the pass the first group of filter. If the ID pass any of two groups, the allow the reception
	D33, D32	D32 low bytes, D33 high bytes, they compose a 32 bits filter value	
The first group	D35, D34	D34 low bytes, D35 high bytes, they compose a 32 bits mask code	
	D37, D36	D36 low bytes, D37 high bytes, they compose a 32 bits filter value	

- Standard/ expansion frame: the setting of FD8358 has no effect to reception. If the data frame fulfills ID mask codes, the standard frame and the expansion frames can be all received. When receive the standard frame, the ID bits is 11, but will still occupy the 32 bits memory combined by [D1,D0]
- 8 bits data transfer: the transfer data is: D0L, D1L, D2L, D3L……(D0L means the low byte of D0)
- 16 bits data transfer: the transfer data is: D0L, D0H, D1L, D1H……(D0H means the high byte of D0)

➢ **Relate Special Soft Components List**

ID	Function	Description
FD8350	CAN Mode	0: not usable 1: XC-CAN network 2: Free format **FREE**
FD8351	CAN baud rate	0, 1KBPS initial value, actual is 5KBPS. 1, 2KBPS initial value, actual is 5KBPS. 2, 5KBPS initial value 3, 10KBPS initial value 4, 20KBPS initial value 5, 40KBPS initial value 6, 50KBPS initial value 7, 80KBPS initial value 8, 100KBPS initial value 9, 150KBPS initial value 10, 200KBPS initial value 11, 250KBPS initial value 12, 300KBPS initial value 13, 400KBPS initial value 14, 500KBPS initial value 15, 600KBPS initial value 16, 800KBPS initial value 17, 1000KBPS initial value
FD8358	CAN free format mode	low 8 bits: 0-standard frame . low 8 bits: 1-expansion frame high 8 bits: 0-8 bits data store high 8 bits: 1-16 bits data store
FD8359	CAN accept timeout time	for free format using, unit: ms
	CAN send timeout time	fixed to be 5ms

2: System M8000 flag

ID	Function	Description

M8240	CAN error flag	ON: error happens OFF: normal if set M8242 as ON, and manually set M8240 as ON, this will enable CAN reset
M8241	CAN node dropped off flag	XC-CAN mode valid ON: certain node/nodes are dropped off OFF: Normal
M8242	do reset or not if CAN error happens	ON: CAN reset automatically when error happens OFF: take no operation when error happens
M8243	CAN send/accept finished flag	FREE mode valid ON: receive/accept finish reset ON automatically when starting to send/accept
M8244	CAN send/accept timeout flag	FREE mode valid ON: send/accept timeout Set OFF automatically when starting to send/accept

3: System D8000

ID	Function	Description
D8240	CAN error information	0: no error 2: initializing error 30: CAN bus error 31: error alarm 32: data overflow
D8241	configure item number when error happens	XC-CAN valid
D8242	data package number sent every second	both XC-CAN and FREE modes are valid
D8243	data package number accepted every second	both XC-CAN and FREE modes are valid
D8244	CAN communication error counter	correspond with M8240 at every CAN error, M8240 will be set ON one time, D8244 increase 1

8 PID Control Function

In this chapter, we mainly introduce the applications of PID instructions for XC Series PLC basic units, including: call the instructions, set the parameters, items to note, sample programs etc.

8-1 Brief Introduction of the Functions

PID instructions and auto-tune functions are added into XC Series PLC basic units (Version 3.0 and above). Via auto-tune method, users can achive the best sampling time and PID parameters and improve the control precision.

The previous versions cannot support PID function on basic units unless they extend with analog modules or BD cards. PID instruction has brought many facilities to the users.

1. The output can be data form **D** and on-off quantity **Y**, user can choose them freely when programming.

2. Via auto-tune, users can achive the best sampling time and PID parameters and improve the control precision.

3. User can choose positive or negative movement via software setting. The former is used in heating control; the later is used in cooling control.

4. PID control separates the basic units with the expansions; this improves the flexibility of this function.

8-2 Instruction Forms

1: Brief Introductions of the Instructions

Execute PID control instructions with the data in specified registers.

PID control [PID]			
16 bits instruction	PID	32 bits instruction	-
Executing Condition	Normally ON/normally closed coil activates	Suitable Models	XC2、XC3、XC5、XCM
Hardware Condition	V3.0 or above	Software Condition	V3.0 or above

2: Operands

Operands	Usage	Type
S1	set the ID Nr. of the target value (SV)	16bits, BIN
S2	set the ID Nr. of the tested value (PV)	16 bits, BIN
S3	set the first ID Nr. of the control parameters	16 bits, BIN
D	the ID Nr. of the operation resule (MV) or output port	16 bits, BIN

3: Suitable soft components

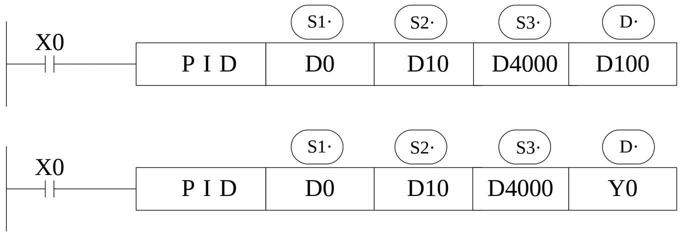
Word Type

Operands	System									Constant	Module	
	D	FD	ED	TD	CD	DX	DY	DM	DS	K/H	ID	QD
S1	●									●		
S2	●										●	
S3	●											
D	●											●

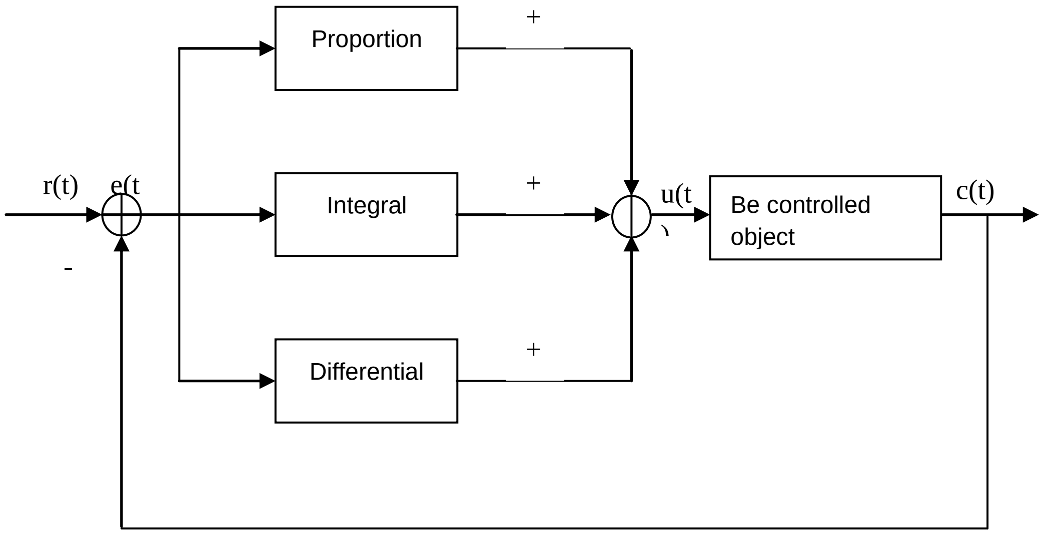
Bit Type

Operands	System						
	X	Y	M	S	T	C	Dn.m
D		●	●	●	●	●	

Functions and Actions



- S3~ S3+ 43 will be occupied by this instruction, do not use them as the common data registers.
- This instruction executes with each sampling time interval.
- To the operation result **D**, the data registers are used to store PID output values; the output points are used to output the occupy ratio in the form of ON/OFF.
- PID control rules are shown as below:



e(t) = r (t) –c (t) (1-1)

u(t) = Kp [e (t) + 1/Ti∫e(t)dt + TD de(t)/dt] (1-2)

Here, e(t) is warp, r(t) is the given value, c(t) is the actual output value, u(t) is the control value;

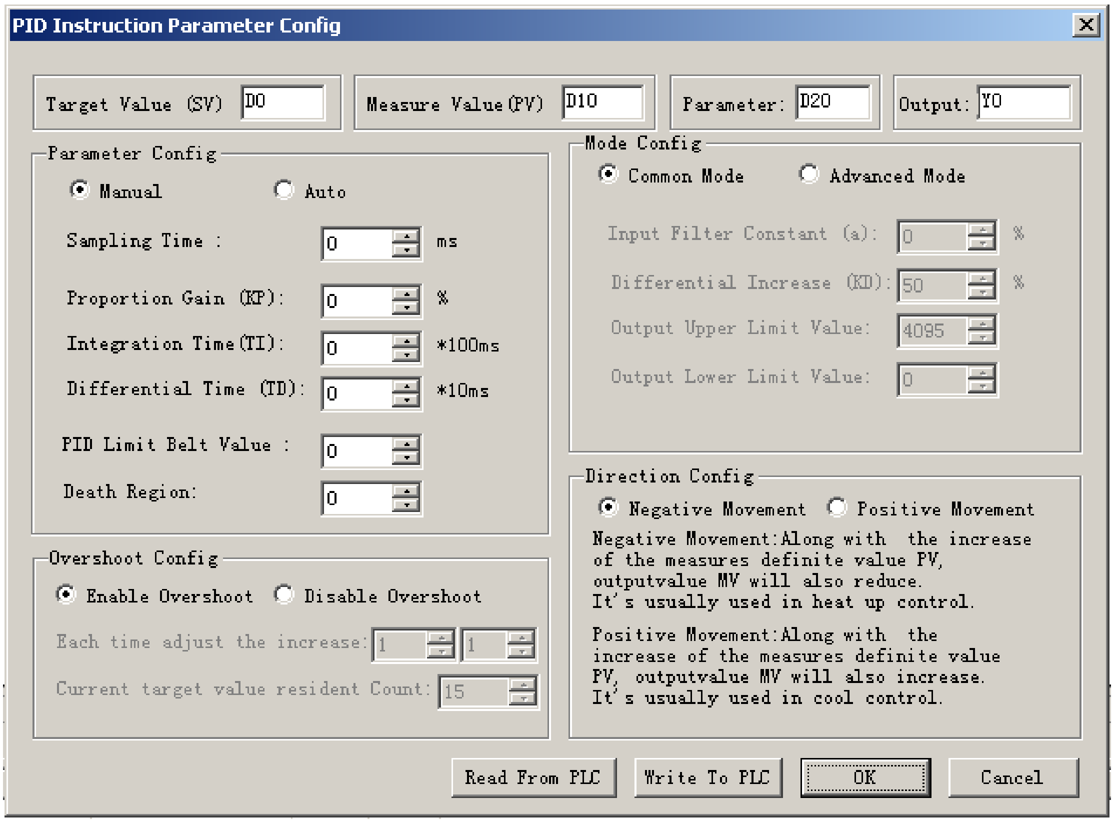
In function (1-2), Kp is the proportion coefficient, Ti is the integration time coefficient, and TD is the differential time coefficient.

The result of the operation:

5. Analog output: MV= digital form of u (t), the default range is 0 ~ 4095.
6. Digital output: Y=T*[MV/PID output upper limit]. Y is the output's activation time within the control cycle. T is the control cycle, equals to the sampling time. PID output upper limit default value is 4095.

8-3 Parameters Setting

Users can call PID instructions in XCP Pro software directly and set the parameters in the window (see graph below), for the details please refer to XCP Pro user manual. Users can also write the parameters into the specified registers by MOV instructions before PID operation.



8-3-1 Register and their Functions

For PID control instruction's relative parameters ID, please refer to the below table:

ID	Function	Description	Memo
S3	sampling time	32 bits without sign	Unit: ms
S3+1	sampling time	32 bits without sign	Unit: ms
S3+2	mode setting	bit0: 0: Negative; 1 Negative; bit1～bit6 not usable bit7: 0: Manual PID; 1: Auto-tune PID bit8: 1: Auto-tune successful flag bit9～bit14 not usable bit15: 0: regular mode; 1: advanced mode	
S3+3	Proportion Gain (Kp)	Range: 1～32767[%]	
S3+4	Integration time (TI)	0～32767[*100ms]	0 is taken as no integral.
S3+5	Differential time (TD)	0～32767[*10ms]	0 is taken as no differential.
S3+6	PID operation zone	0～32767	PID adjustment band width value.
S3+7	control death zone	0～32767	PID value keeps constant in death zone
S3+8	PID Auto-tune cycle varied value	full scale AD value *（0.3~1%）	
S3+9	PID Auto-tune overshoot permission	0: enable overshoot 1:disable overshoot	
S3+10	current target value adjustment percent in auto-tune finishing transition stage		
S3+11	current target value resident count in auto-tune finishing transition stage		
S3+12~ S3+39	occupied by PID operation's internal process		
Below is the ID of advanced PID mode setting			
S3+40	Input filter constant (a)	0～99[%]	0: no input filter
S3+41	Differential gain (KD)	0～100[%]	0: no differential gain
S3+42	Output upper limit value	-32767～32767	
S3+43	Output lower limit value	-32767～32767	

8-3-2 Parameters Description

- **Movement Direction:**

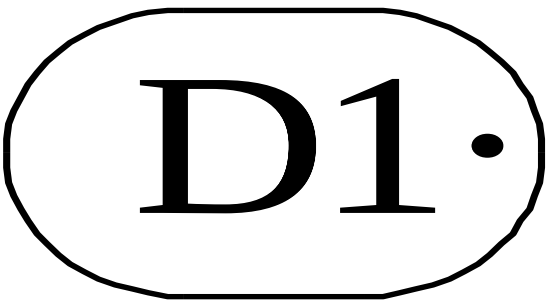
- Positive movement: the output value MV will increase with the increasing of the detected value PV, usually used for cooling control.
- Negative movement: the output value MV will decrease with the increasing of the detected value PV, usually used for heating control.

- **Mode Setting**

- Common Mode:
  The parameter's register zone is from **S3** to **S3**+43, **S3** to **S3**+11 and needs to be set by users. **S3+12** to **S3**+43+12 are occupied by the system and are not available to users.
- Advanced Mode:
  The parameter's register zone is from **S3** to **S3**+43, **S3** to (**S3**+11) and (**S3+40)** to (**S3**+43) need to be set by users. (**S3+12)** to (**S3**+**39)** are occupied by the system and are not availableto users.
- **Sample Time [S3]**
  The system samples the current value according to certain time interval and compare them with the output value. This time interval is the sample time **T**. There is no requirement for **T** during **AD** output. **T** should be larger than one PLC scan period during port output. **T** value should be chosen among 100~1000 times of PLC scan periods.

- **PID Operation Zone [S3+6]**

  PID control is entirely opened at the beginning and close to the target value with the highest speed (the defaulted value is 4095), when it entered into the PID computation range, parameters Kp, Ti, TD will be effective.
  See graph below:



If the target value is 100, PID operation zone is 10, then the real PID's operation zone is from 90 to 110.

- **Death Region [S3+7]**

  Within this region the PID value will not vary. This stops the system from making small changes which will imbalance the system.

Suppose: we set the death region value to be 10. Then in the above graph, the difference is only 2 comparing the current value with the last value. The PID control will not change value. The difference is 13 (more than death region 10) comparing the current value with the next value, this difference value is larger than control death region value, the PID control will start to vary.

8-4 Auto-tune Mode

If users do not know how to set the PID parameters, they can choose auto-tune mode which can find the optimal control parameters (sampling time, proportion gain **Kp**, integral time **Ti**, differential time **TD**) automatically.

I. Auto-tune mode is suitable for these objectives: temperature, pressure; but is not suitable for liquid level and flow.

II. Users can set the sampling cycle to be 0 at the beginning of the auto-tune process then modify the value manually in terms of practical needs after the auto-tune process is completed.

III. Before selecting auto-tune, the system should be under the no-control steady state. If the function is to 'Take the temperature' for example: the detected temperature should be the same as the environment temperature.

To enter the auto-tune mode, please set bit7 of (**S3+ 2)** to be 1 and turn on PID working condition. If bit8 of (**S3+ 2)** turns to 1, it means the auto-tune is successful.

- PID auto-tune period value **[S3+ 8]**

Set this value in **[S3+ 8]** during auto-tune.
This value decides the auto-tune performance, in a general way, set this value to be the AD result corresponding to one standard detected unit. The default value is 10. The suggested setting range:
full-scale AD result × 0.3 ~ 1%.
This value does not normally need altering, however, if the system is interfered greatly by outside, this value should be increased modestly to avoid wrong judgment for positive or negative movement. If this value is too large, the PID control period (sampling time) set by the auto-tune process will be too long.

※1: if users have no experience, please use the defaulted value 10, set PID sampling time (control period) to be 0ms then start the auto-tune.

- PID auto-tune overshooting permission setting **[S3+ 9]**

If set 0, overshooting is permitted, the system can study the optimal PID parameters all the time. But in self-study process, detected value may be lower or higher than the target value, safety factor should be considered here.
If set 1, overshooting is not permitted. For these objectives which have strict safety demand such as pressure vessel, set **[S3+ 9]** to be 1 to prevent from detected value being seriously over the target value. In this process, if **[S3+ 2]** bit8 changes from 0 to 1, it means the auto-tune is successful and the optimal parameters are set; if **[S3+ 2]** is always 0 until **[S3+ 2]** bit7 changes from 1 to 0, it means the auto-tune is completed but the parameters are not the best and need to be modified by users.

- Every adjustment percent of current target value at auto-tune process finishing transition stage **[S3+10]**

This parameter is effective only when **[S3+ 9]** is 1.

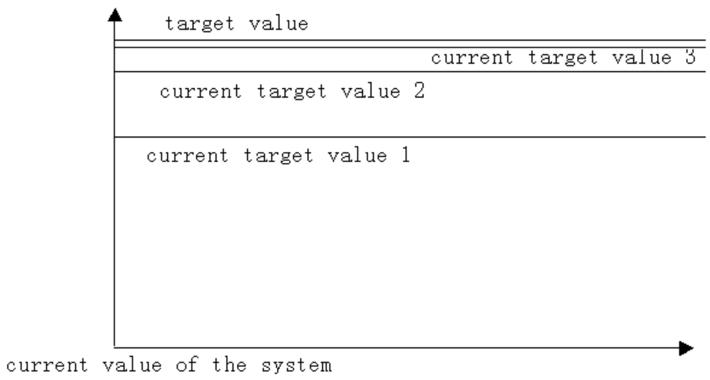
If setting PID control after auto-tune, small range of overshooting may be occurred. It is better to decrease this parameter to control the overshooting. But response delay may occur if this value is too small. The defaulted value is 100% which means the parameter is not effective. The recommended range is 50~80%.

Cutline Explanation:

Current target value adjustment percent is 2/3 (**S3 + 10** = 67%), the original temperature of the system is 0 ºC, target temperature is 100 ºC, the current target temperature adjustment situation is shown as below:

Next current target value = current target value + (final target value – current target value) × 2/3;

So the changing sequence of current target is 66 ºC, 88 ºC, 96 ºC, 98 ºC, 99 ºC, 100 ºC.



- The stay times of the current target value at auto-tune process finishing transition stage **[S3+11]**

This parameter is valid only when **[S3+9]** is 1;

If entering into PID control directly after auto-tune, small range of overshoot may occur. Overshoot can be prevented if increasing this parameter properly, but it will cause response lag if this value is too large. The default value is 15 times. The recommended range is from 5 to 20.

8-5 Advanced Mode

Users can set some parameters in advanced mode in order to get the better effect of PID control. Enter into the advanced mode, please set **[S3+2]** bit 15 to be 1, or set it in the XCP Pro software.

- Input Filter constant

  It will smooth the sampling value. The default value is 0% which means no filter.

- Differential Gain

The low pass filtering process will relax the sharp change of the output value. The default value is 50%, the relaxing effect will be more obviously if increasing this value. Users do not need to change it.

- Upper-limit and lower-limit value

Users can choose the analog output range via setting this value.

Default value: lower- limit output= 0

Upper -limit= 4095

8-6 Application Outlines

- Under continuous output, the system whose effectability will die down with the change of the feedback value can do self-study, such as temperature or pressure. It is not suitable for flux or liquid level.
- Under the condition of overshoot permission, the system will get the optimal PID parameters from self-study.
- Under the condition of overshoot not allowed, the PID parameters got from self-study is up to the target value, it means that different target value will produce different PID parameters which are not the optimal parameters of the system and for reference only.
- If the self-study is not available, users can set the PID parameters according to practical experience. Users need to modify the parameters when debugging. Below are some experience values of the control system for your reference:

◆ Temperature system:

P (%) 2000 ~ 6000, I (minutes) 3 ~ 10, D (minutes) 0.5 ~ 3

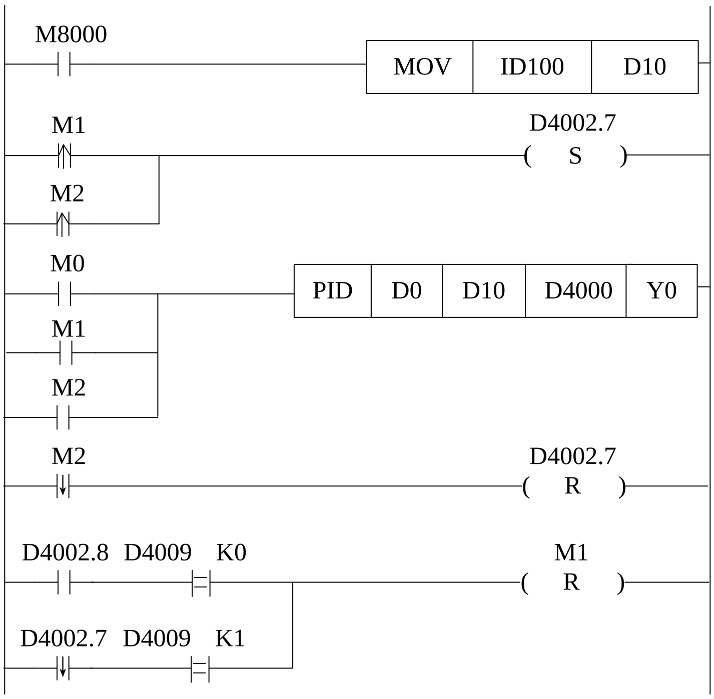
◆ Flux system: P (%) 4000 ~ 10000, I (minutes) 0.1 ~ 1

◆ Pressure system: P (%) 3000 ~ 7000, I (minutes) 0.4 ~ 3

◆ Liquid level system: P (%) 2000 ~ 8000, I (minutes) 1 ~ 5

8-7 Example Program

PID Control Program is shown below:



// Move ID100 content into D10

// convert PID mode to be auto tune at the beginning of auto tune control starts or auto tune finish

// start PID, D0 is target value, D10 is detected value, from D4000 the zone is PID parameters area; output PID result via Y0

// PID control finish, close auto tune PID mode

// if auto tune is successful, and overshoot is permitted, close auto tune control bit, auto tune finish;
If auto tune turns to be manual mode, and auto tune is not permitted, close auto tune control bit

Soft components function comments:
D4000.7: auto-tune bit
D4002.8: auto-tune successful sign
M0: normal PID control
M1: auto-tune control
M2: enter into PID control after auto-tune

9 C Language Function Block

In this chapter, we focus on C language function block's specifications; edition; instruction calling; application points etc. We end the chapter with the common functions list.

9-1 Functions Summary

This is the new added function in XCP Pro software. This function enables the customers to write function blocks with C language in XCP Pro and call the function blocks at any necessary place. This function supports most of C language functions, strength the program's security. As users can call the function at many places and call different functions, this function increases the programmer's efficiency greatly.

9-2 Instruction Format

1:Instruction Summary

Call the C language Function Block at the specified place

Call the C language Function Block [NAME_C]			
16 bits Instruction	NAME_C	32 bits Instruction	-
Execution Condition	Normally ON/OFF, Rising/Falling Edge activation	Suitable Models	XC1、XC2、XC3、XC5、XCM
Hardware Requirement	V3.0C and above	Software Requirement	V3.0C and above

2: Operands

Operands	Function	Type
S1	name of C Function Block, defined by the user	String
S2	Correspond with the start ID of word **W** in C language Function	16bits, BIN
S3	Correspond with the start ID of word **B** in C language Function	16bits, BIN

3: Suitable Soft Components

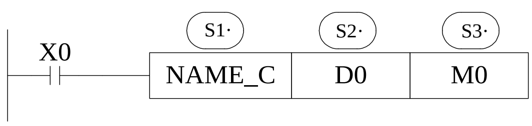
Word

Operands	System									Constant	Module	
	D	FD	ED	TD	CD	DX	DY	DM	DS	K/H	ID	QD
S2	•											

Bit

Operands	System						
	X	Y	M	S	T	C	Dn.m
S3			•				

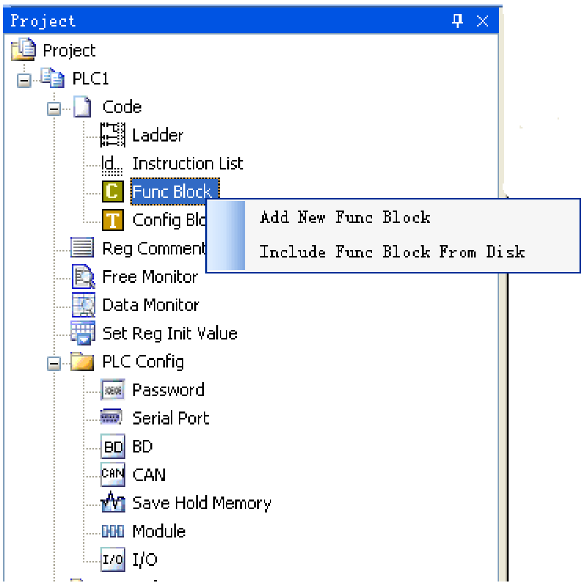
Functions and Actions



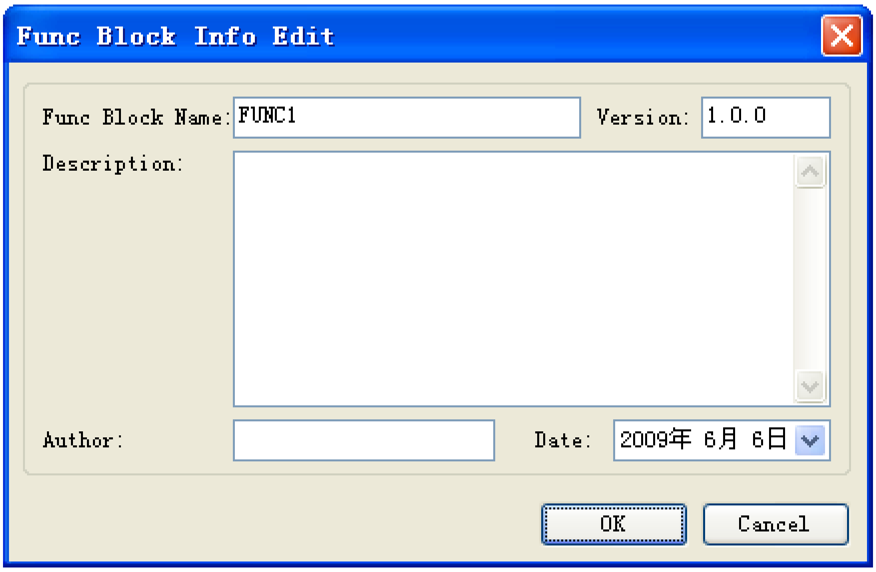
- The name is composed by numbers, letters and underscores, the first character must not be a number and the name shouldn't be longer than 8 ASC.
- The name can't be same with PLC's internal instructions e.g. LD, ADD, SUB, PLSR etc.
- The name can't be same as any function blocks already existing in the PLC.

9-3 Operation Steps

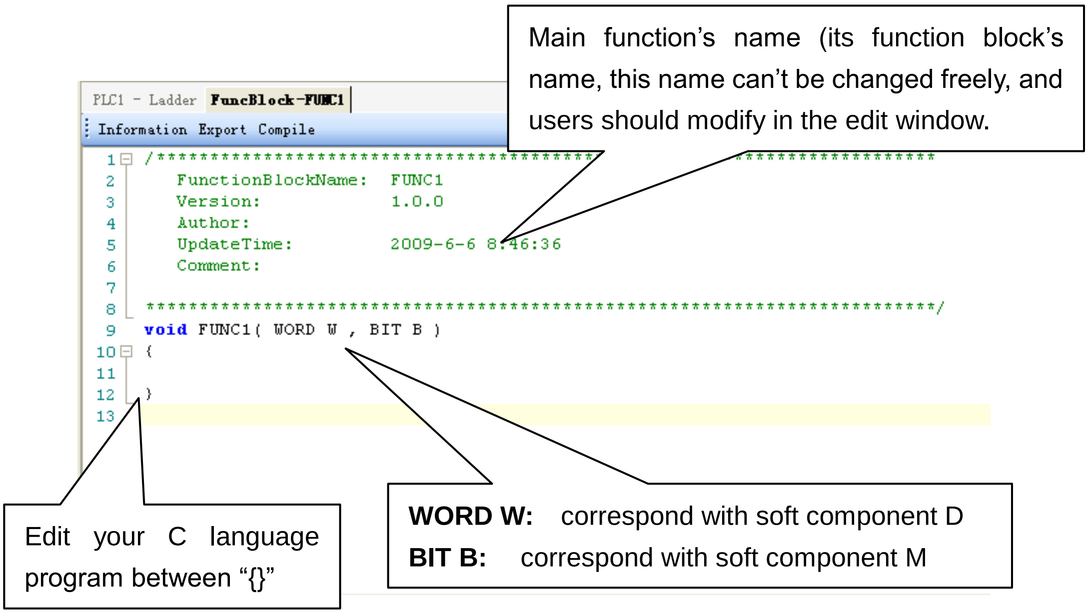
1: Open PLC edit tool, in the left “Project” toolbar, choose “Function Block”, right click it and choose “Add New Function Block”



2: See graph below, fill in the information of your function;



3: After creating the new Function Block, you can see the edit interface as shown below:



- Parameters' transfer format: if **Function Block** is called in ladder format, the transferred D and M is the start ID of W and B. Take the above graph as the example, start with D0 and M0, then W[0] is D0, W[10] is D10, B［0 is M0, B［10］is M10. If in the ladder the used parameters are D100, M100, then W[0] is D100, B［0］is M100. So, word and bit component's start address is defined in PLC program by the user.

- Parameter W: represent **Word** soft component, use in the form of data group. E.g. W[0]=1;W[1]=W[2]+W[3]; in the program, use according to standard C language rules.

- Parameter B: represents **Bit** soft component, use in the form of data group. Supports **SET** and **RESET**. E.g: B[0]=1;B[1]=0; And assignment, for example B[0]=B[1].

- Double-word operation: add **D** in front of **W**, e.g. DW[10]=100000, it means assignment to the double-word W[10]W[11]

- Floating Operation: Supports the definition of floating variable in the function, and executes floating operation;

- Function Library: In **Function Block**, users can use the Functions and Variables in function library directly. For the Functions and Variables in function library, see the C Language Function List at the end of this chapter.

- The other data type supported:

```
BOOL;      //BOOL Quantity
INT8U;     //8 bits unsigned integral
INT8S;     //8 bits signed integral
INT16U     //16 bits unsigned integral
INT16S     //8 bits signed integral
INT32U     //32 bits unsigned integral
INT32S     //32 bits signed integral
FP32;      //Single precision Floating
FP64;      // Double precision Floating
```

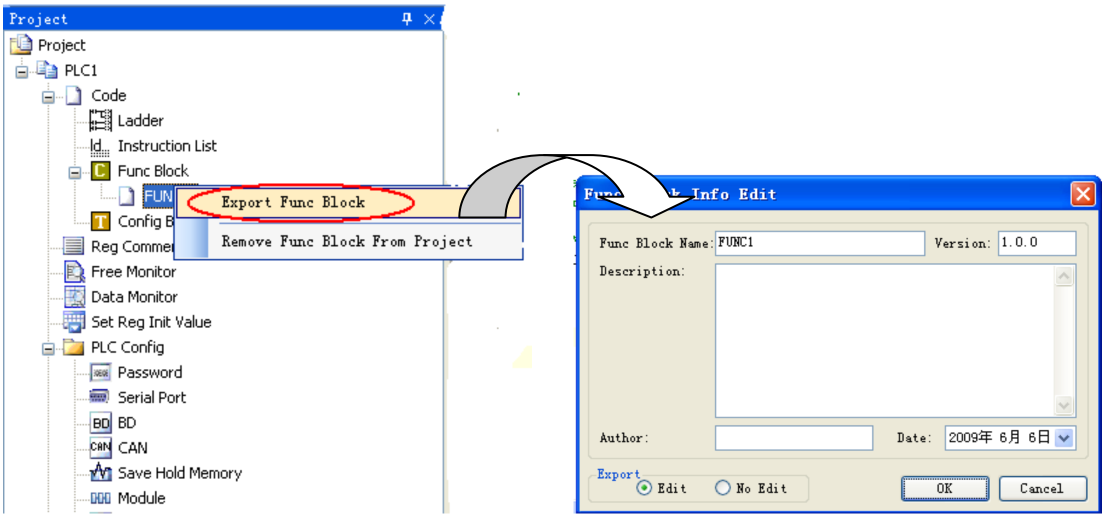
- Predefined Marco

```
#define   true    1
#define   false   0
#define   TRUE    1
#define   FALSE   0
```

9-4 Import and Export the Functions

1: Export

(1) Function: export the function as the file, then other PLCs program can import to use;



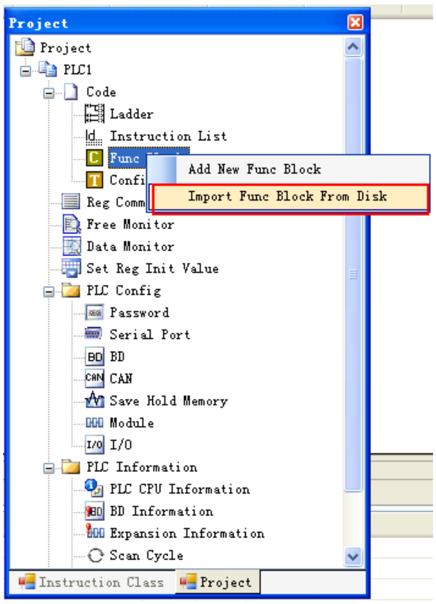
(2) Export Format

a) Editable; export the source codes and save as a file. If imported again, the file is editable.

b) Not editable: if the source code is not exported the file will be read-only by third parties.

2: Import

Function; Import the existing **Function Block** file, to use in the PLC program;



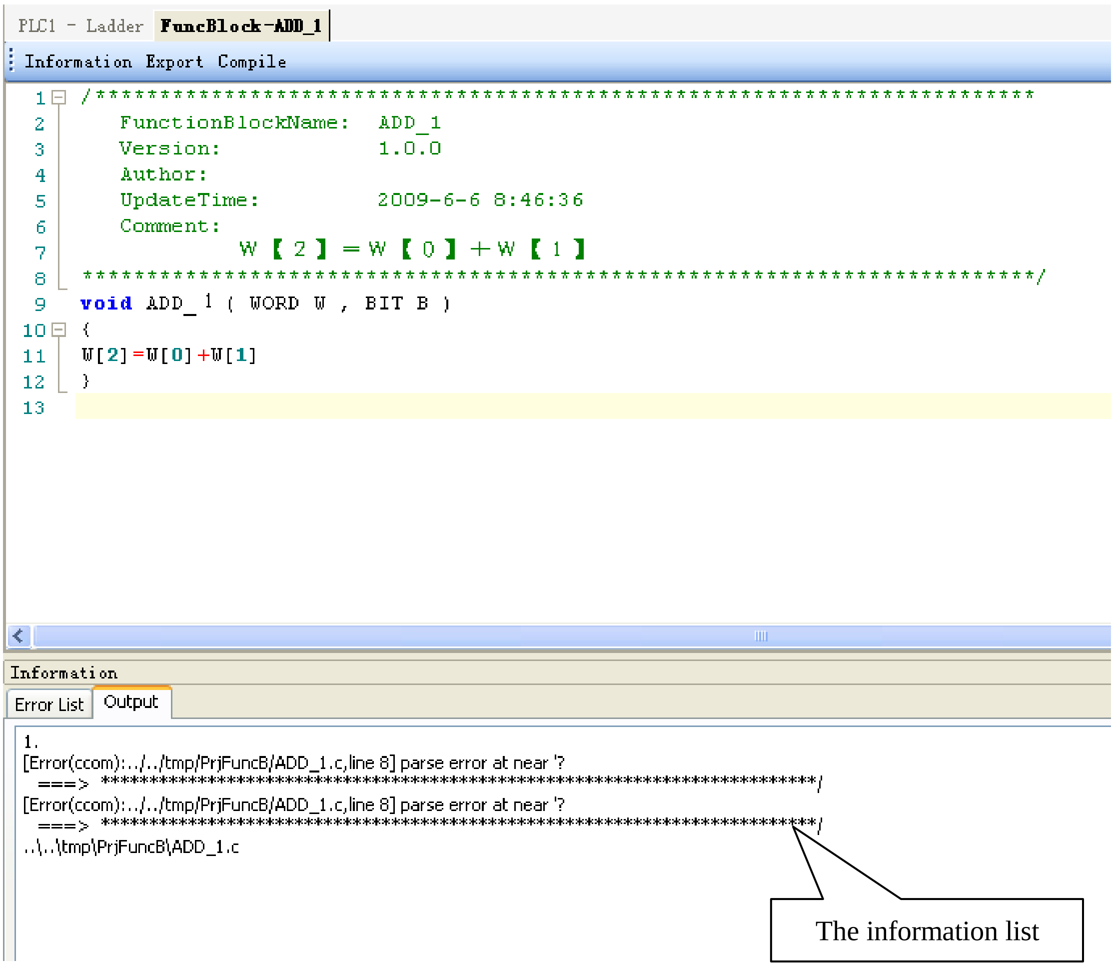
Choose the **Function Block**, right click “Import Function Block From Disk”, choose the correct file, then click OK.

9-5 Edit the Function Blocks

Example: Add D0 and D1 in the PLC's registers, then assign the value to D2;

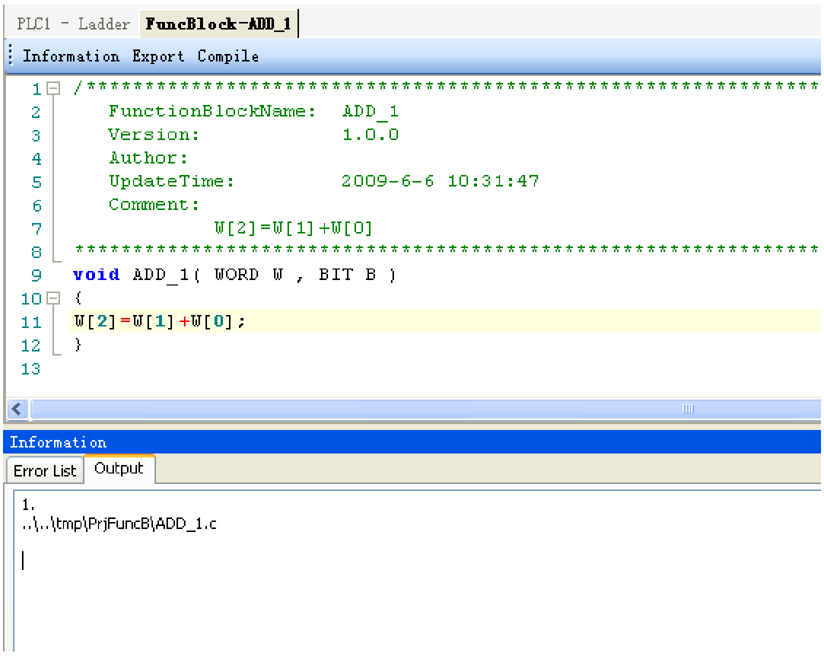
(1) In "Project" toolbar, new create a **Function Block**, here we name the **Function Block** as **ADD_2**, then edit C language program;

(2) Click **compile** after edition



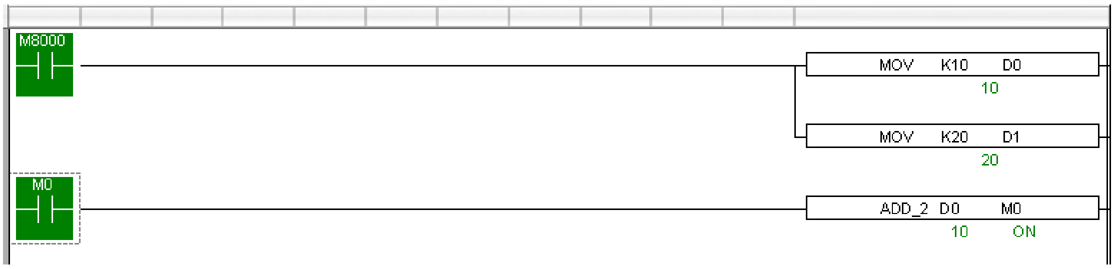
According to the information shown in the output blank, we can search and modify the grammar error in C language program. Here we can see that in the program there is no "**;**" sign behind W[2]=W[0]+W[1];

Compile the program again after modify the program. In the information list, we can confirm that there is now no grammar error in the program.

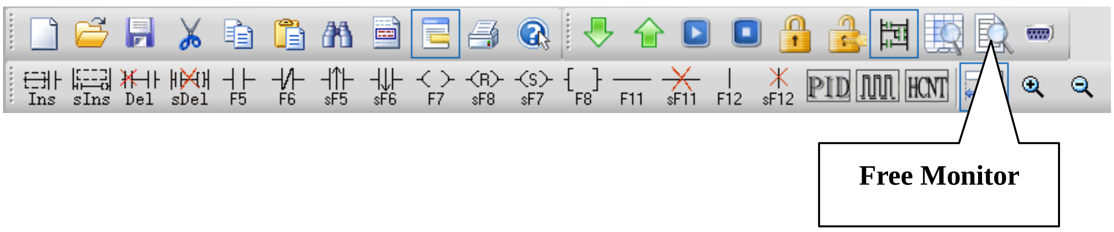


(3) Write PLC program, assign value 10 and 20 into registers D0, D1 separately, then call Function Block ADD_2, see graph below:

(4) Download program into PLC, run PLC and set M0.



(5) From Free Monitor in the toolbar, we can see that D2 changes to be 30, it means the assignment is successful.



9-6 Example Program

- Function: calculate CRC parity value via Function Block
- CRC calculation rules:

(1) Set 16 bits register (CRC register) = FFFF H

(2) XOR (Exclusive OR) 8 bits information with the low byte of the 16 bits CRC register.

(3) Right shift 1 bit of CRC register, fill 0 in the highest bit.

(4) Check the right shifted value, if it is 0, save the new value from step3 into CRC register; if it is not 0, XOR the CRC register value with A001 H and save the result into the CRC register.

(5) Repeat step3&4 until all the 8 bits have been calculated.

(6) Repeat step2~5, then calculate the next 8 bits information. Until all the information has been calculated, the result will be the CRC parity code in CRC register.

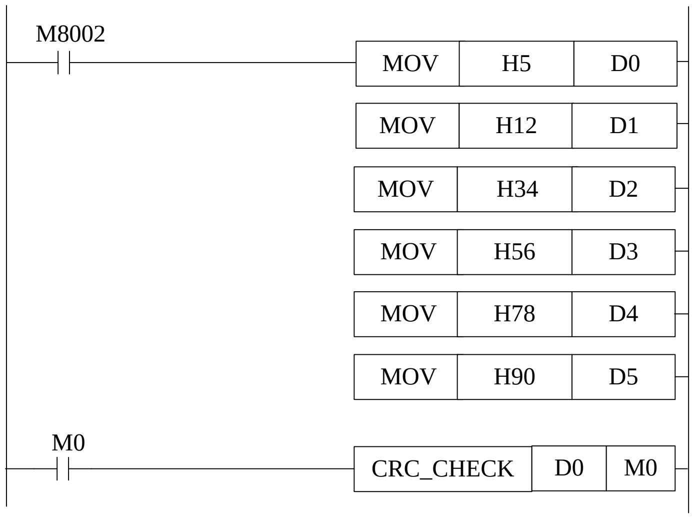
- Edit C language Function Block program, see graph below:

```
void CRC_CHECK( WORD W , BIT B )
{
        int i,j,m,n;
        unsigned int reg_crc=0xffff,k;

        for( i = 0 ; i < W[0] ; i++ )
                {
                reg_crc^=W[i+1];
                for(j=0;j<8;j++)
                {
                if(reg_crc&0x01)
                        reg_crc=(reg_crc>>1)^0xa001;
                else
                        reg_crc=reg_crc>>1;
                }
                }

                m=W[0]+1;
                n=W[0]+2;
                k=reg_crc&0xff00;
                W[m] = k>>8;
                W[n]=reg_crc&0xff;
              }
```

- Edit PLC ladder program,

D0: Parity data byte number;

D1~D5: Parity data's content, see graph below:



- Download to PLC, then RUN PLC, set M0, via Free Monitor, we can find that values in D6 and D7 are the highest and lowest bit of CRC parity value.

9-7 Application Points

- When uploading a PLC program which contains some Function Blocks, the Function Blocks can't be uploaded, there will be an error say: There is an unknown instruction;
- In one Function Block file, you can write many subsidiary functions, can call each other;
- Each Function Block files are independent, they can't call its owned functions;
- Function Block files can call C language library functions in form of floating, arithmetic like sin, cos, tan etc.

9-8 C Language Function List

The default function library

Constant	Data	Description
_LOG2	(double)0.69314718055994530941723212145 8	Logarithm of 2
_LOG10	(double)2.30258509299404590109361379290 93	Logarithm of 10
_SQRT2	(double)1.41421356237309504880168872421	Radical of 2
_PI	(double)3.14159265358979323846264338327 95	PI
_PIP2	(double)1.57079632679489661923132169163 975	PI/2
_PIP2x3	(double)4.71238898038468985769396507491 925	PI*3/2

String Function	Description
void * memchr(const void *s, int c, size_t n);	Return the first **c** position among **n** words before **s** position
int memcmp(const void *s1, const void *s2, size_t n);	Compare the first **n** words of position **s1** and **s2**
void * memcpy(void *s1, const void *s2, size_t n);	Copy **n** words from position **s2** to s1and return **s1**
void * memset(void *s, int c, size_t n);	Replace the **n** words start from **s** position with word **c**, and return position **s**
char * strcat(char *s1, const char *s2);	Connect string **ct** behind string **s**
char * strchr(const char *s, int c);	Return the first word **c** position in string **s**
int strcmp(const char *s1, const char *s2);	Compare string s1 and s2
char * strcpy(char *s1, const char *s2);	Copy string s1 to string s2

Double-precision math function	Single-precision math function	Description
double acos(double x);	oat acosf(float x);	Inverse cosine function.
double asin(double x);	float asinf(float x);	Inverse sine function
double atan(double x);	float atanf(float x);	Inverse tangent function
double atan2(double y, double x);	float atan2f(float y, float x);	Inverse tangent value of parameter (y/x)
double ceil(double x);	float ceilf(float x);	Return the smallest double integral which is greater or equal with parameter **x**
double cos(double x);	float cosf(float x);	Cosine function
double cosh(double x);	float coshf(float x);	Hyperbolic cosine function cosh(x)=(e^x+e^(-x))/2.
double exp(double x);	float expf(float x);	Exponent (e^x) of a nature data
double fabs(double x);	float fabsf(float x);	Absolute value of parameter x
double floor(double x);	float floorf(float x);	Return the largets dounble integral which is smaller or equals with **x**
double fmod(double x, double y);	float fmodf(float x, float y);	If **y** is not zero, return the reminder of floating **x/y**
double frexp(double val, int _far *exp);	float frexpf(float val, int _far *exp);	Break floating data **x** to be mantissa and exponent **x** = m*2^exp, return the mantissa of m, save the logarithm into **exp**.
double ldexp(double x, int exp);	float ldexpf(float x, int exp);	X multipy the (two to the power of n) is x*2^n.
double log(double x);	float logf(float x);	Nature logarithm logx
double log10(double x);	float log10f(float x);	logarithm (log10x)
double modf(double val, double *pd);	float modff(float val, float *pd);	Break floating data X to be integral part and decimal part, return the decimal part, save the integral part into parameter ip.
double pow(double x, double y);	float powf(float x, float y);	Power value of parameter **y** (x^y)
double sin(double x);	float sinf(float x);	sine function
double sinh(double x);	float sinhf(float x);	Hyperbolic sine function, sinh(x)=(e^x-e^(-x))/2.
double sqrt(double x);	float sqrtf(float x);	Square root of parameter X
double tan(double x);	float tanf(float x);	tangent function.
double tanh(double x);	float tanhf(float x);	Hyperbolic tangent function, tanh(x)=(e^x-e^(-x))/(e^2+e^(-x)).

10 Sequential Function BLOCK

This chapter describes the basic concepts; internal instruction manipulation; relative instructions; executing form and application points of Sequential Function Blocks.

Relative Instructions:

Mnemonic	Function	Circuit and soft components	chapter
SEQUENTIAL FUNCTION BLOCK			
BSTOP	Pause the execution of **BLOCK**	BSTOP S1 S2	10-6-1
BGOON	Continue to execute **BLOCK**	BGOON S1 S2	10-6-1

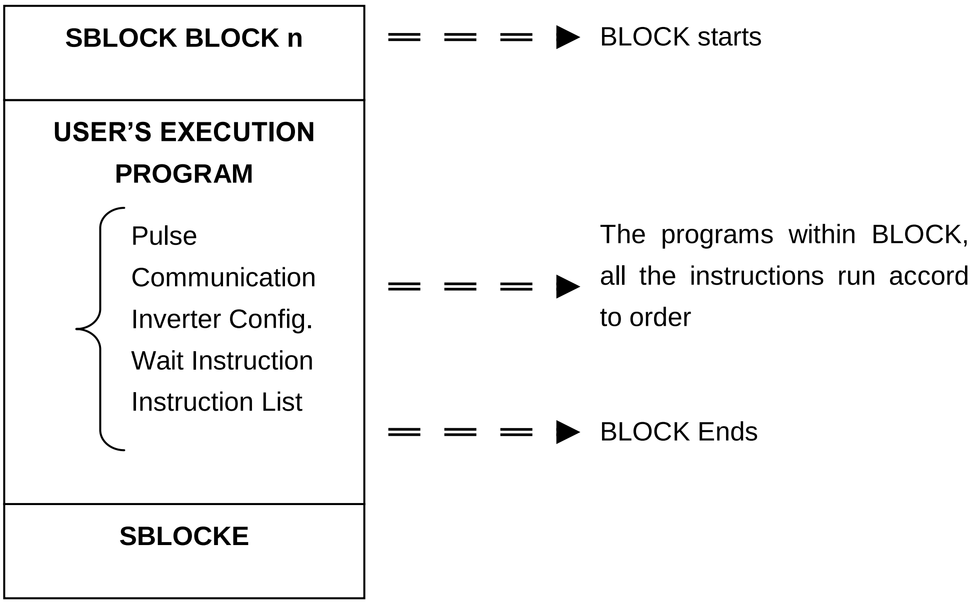
10-1 BLOCK Basic Concept

10-1-1 BLOCK Summary

Sequential function block, in short BLOCK, is a program block to realize certain functions. We can treat the block as a special flow, in this special flow, all the programs run according to one principle, i.e. sequential execution principle; this is how BLOCK differs from other programs.

BLOCK starts with SBLOCK, ends with SBLOCKE, the programmer writes programs between them. If in one BLOCK there are many “send pulse” instructions (also same with other type of instructions), then the pulse instructions will run according to the time order of the activate conditions; the next pulse instruction runs only after the previous instruction finishes.

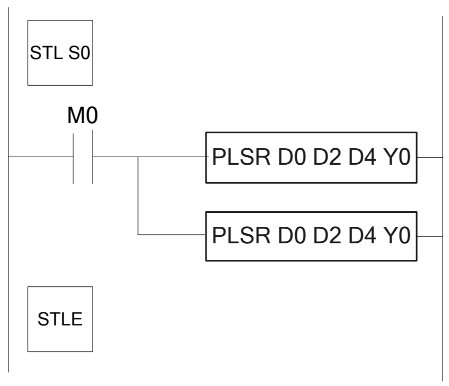
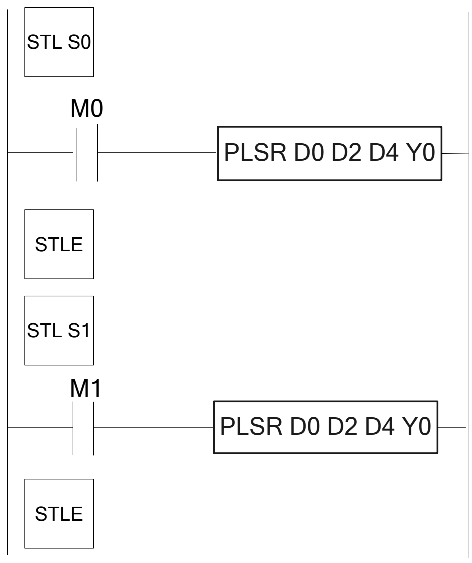
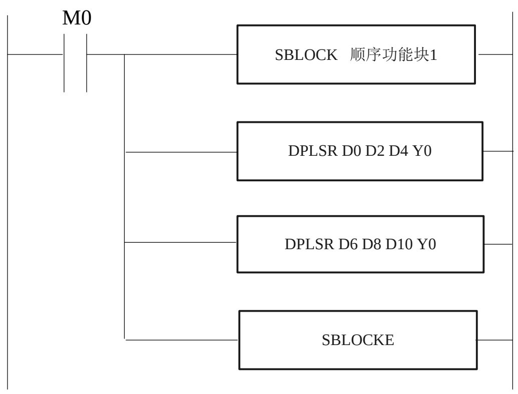
See a whole BLOCK structure below:



10-1-2 Reason to introduce BLOCK

How to write instructions to optimize the original pulse, communication in flows;

As in XCP Pro, we don't support to run many pulse, communication instructions in one flow, it's troublesome to write the program. With BLOCK, we support writing many pulse, communication instructions, all the instructions run accord to sequential principle;

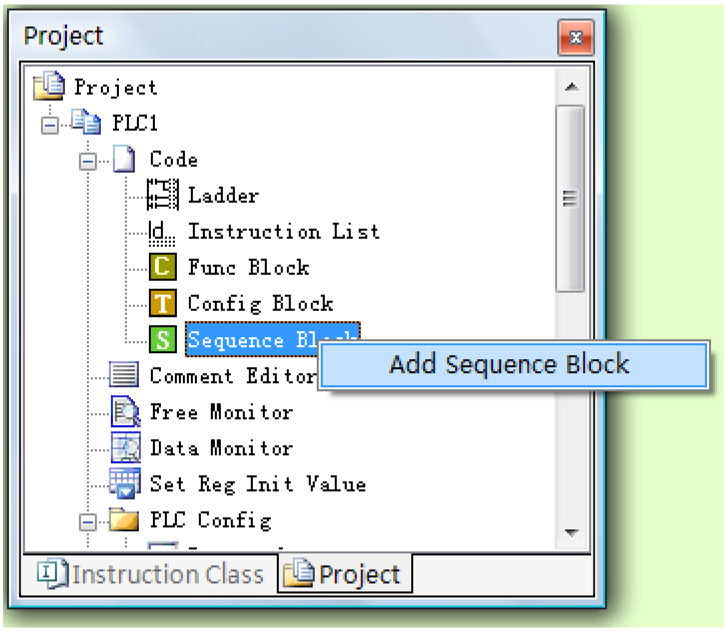
<table>
<tr><th></th><th>Wrong （×）</th><th>Correct （√）</th></tr>
<tr><td>WITHOUT BLOCK</td><td>
</td><td>
</td></tr>
<tr><td>WITH BLOCK</td><td></td><td>
</td></tr>
</table>

10-2 Call the BLOCK

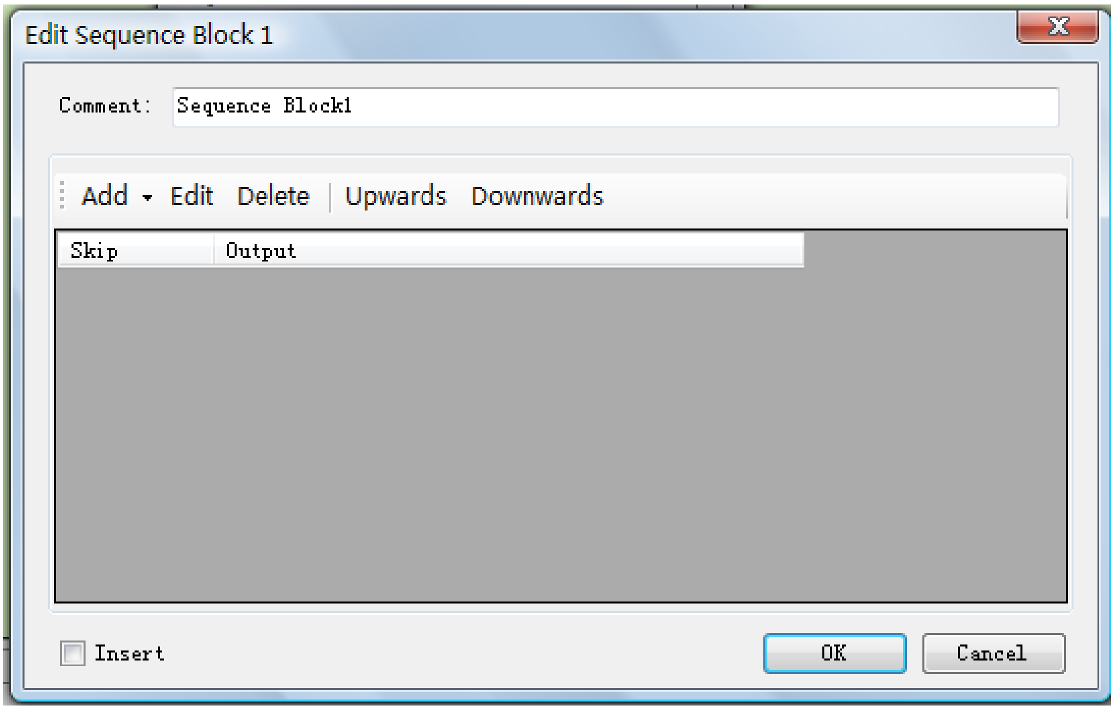
In one program, you can call many BLOCKs. Call BLOCK via XCP Pro. See method below:

10-2-1 Add a BLOCK

Open XCP Pro, in the left toolbar, find “Sequence Block”, right click it, you can see “Add Sequence Block”. See graph below:

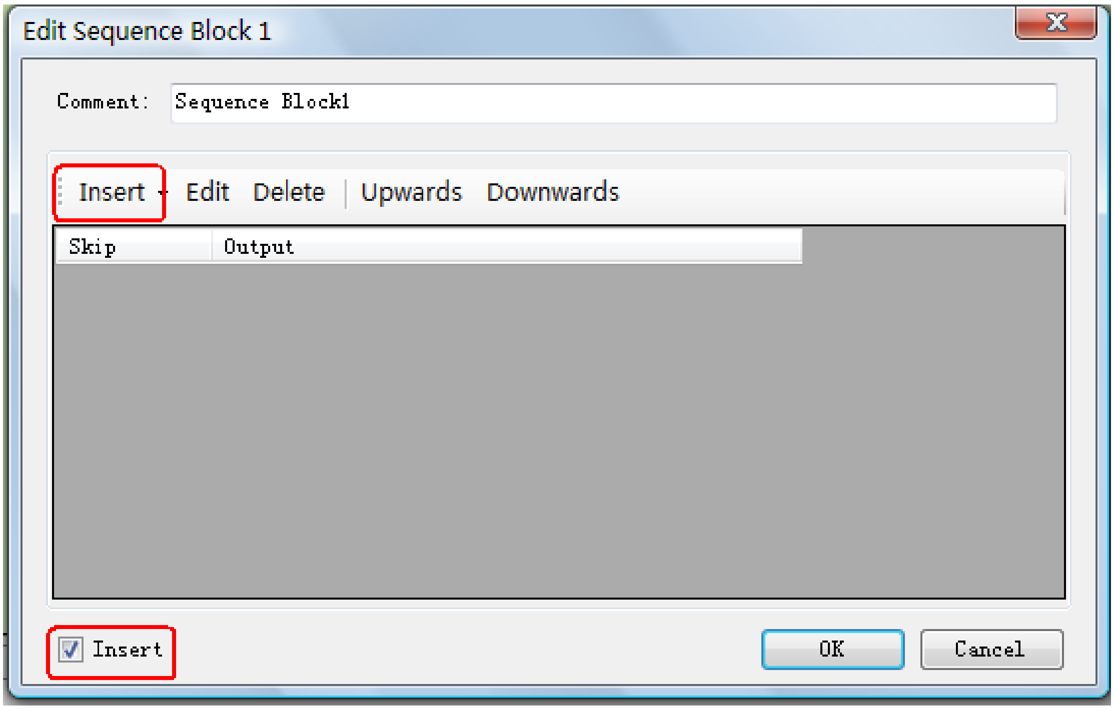


Click this command, see the configure interface below:



The above interface is used to edit one BLOCK, in that interface you can add many program sections, modify and delete the correspond sections, including pulse, communication, motion control etc; upwards/downwards is used to up/down shift the instructions in BLOCK.

Please note: in the left bottom there is a “inset” item, if you choose it, the “Add” button will change to be “Insert:, see screenshot below:

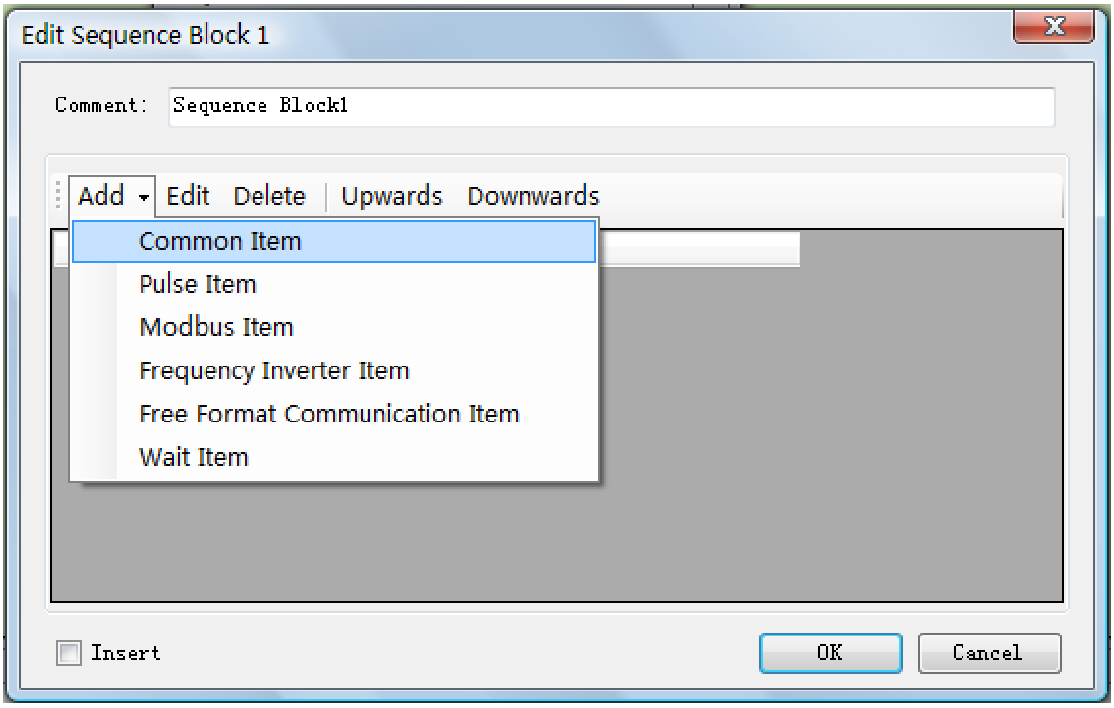


The difference between “Add” and “Insert”:

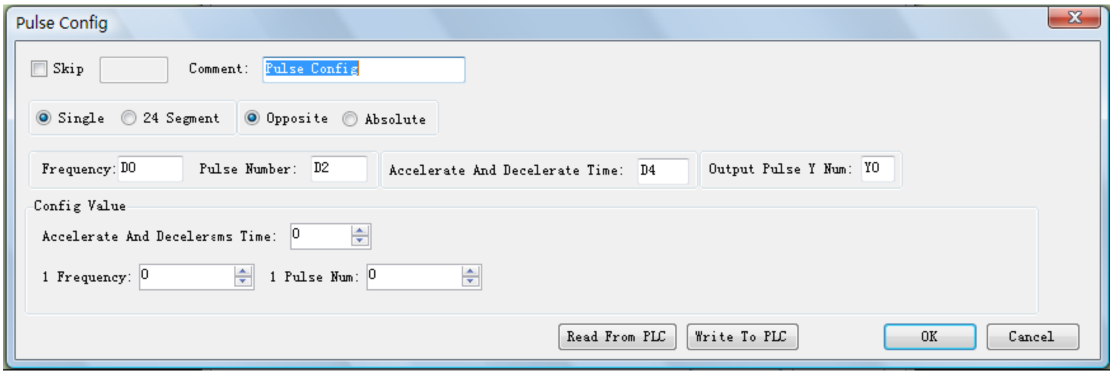
Add: add the specified content at **the end** of BLOCK;

Insert: add the specified content at **any place** of BLOCK;

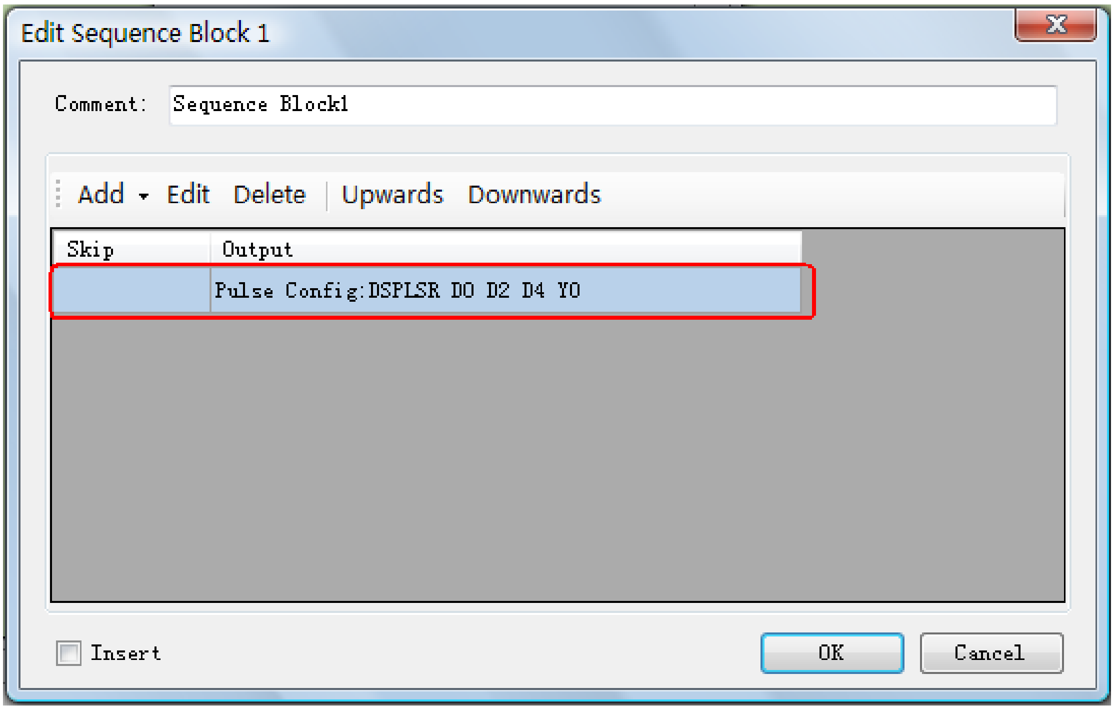
Click “Add”, you can see that the system lists all the instruction types you may use, including instruction list, pulse configure, Modbus instruction, Wait instruction, inverter read/write, free format communication; see screenshot below:



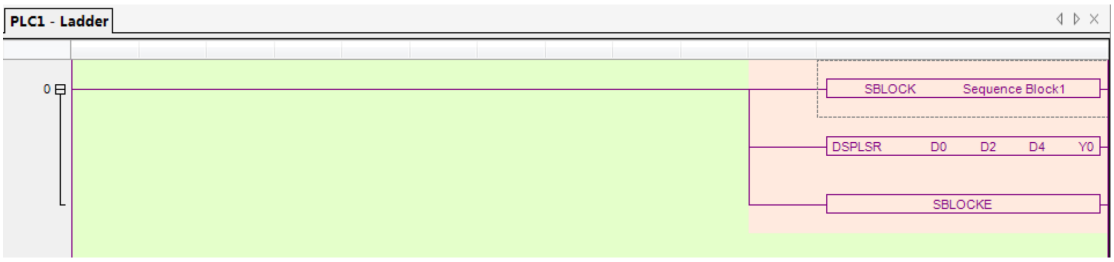
For example, add a “Pulse Item” in the BLOCK and set it:



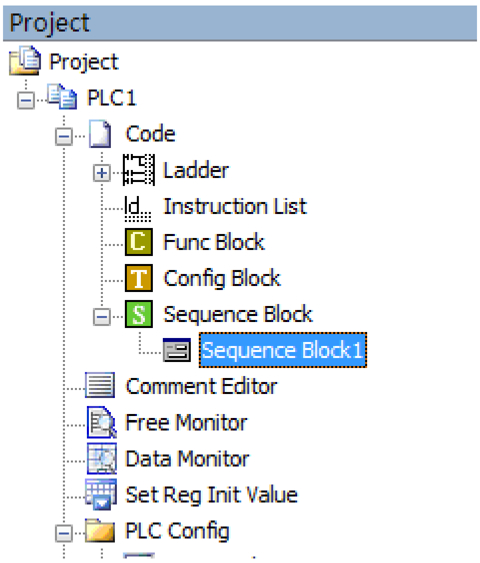
Click OK, we can see that in the configure interface, the corresponding information also been added, see screenshot below:



Click “OK”, in the Ladder interface, you can see the instructions section as below:

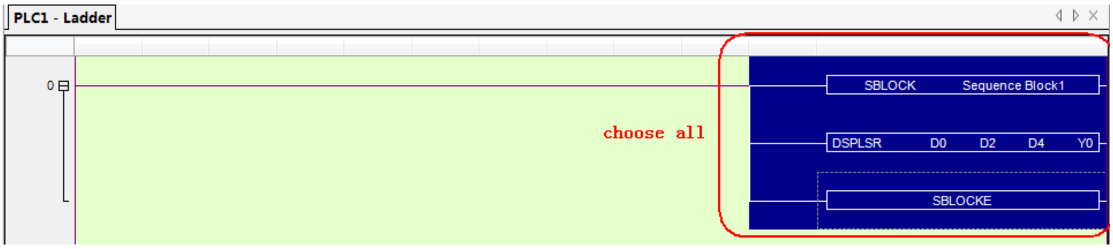


Meantime, in the left toolbar, you can see the new added block, see graph below:

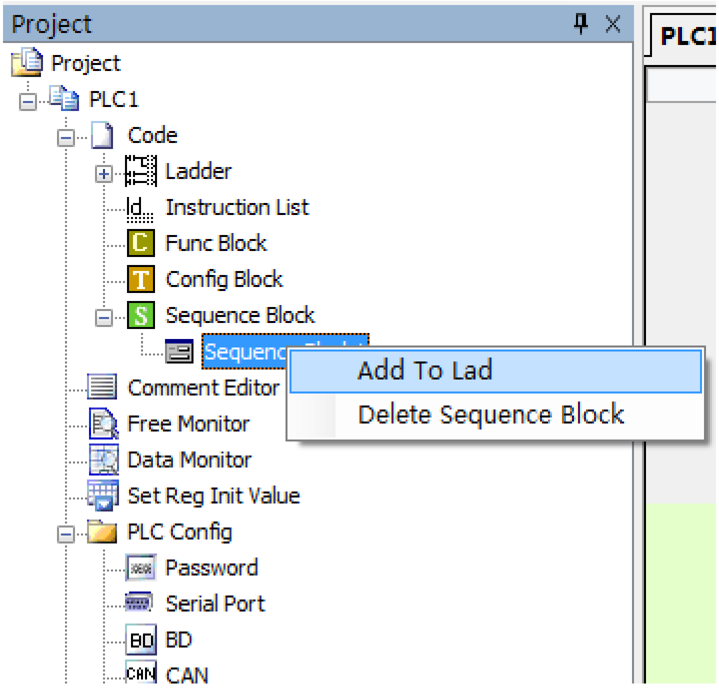


10-2-2 Move the BLOCK

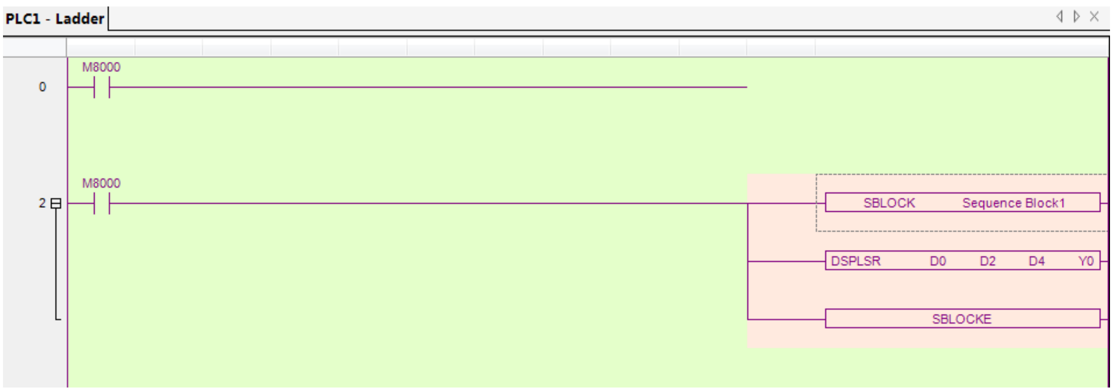
If you want to move the created BLOCK elsewhere, you should delete the original BLOCK (choose all and delete), see graph below:



Then move the mouse to the required place, activate this place; right click the created BLOCK, in the pop-up menu, choose "Add To Lad", see graph below:



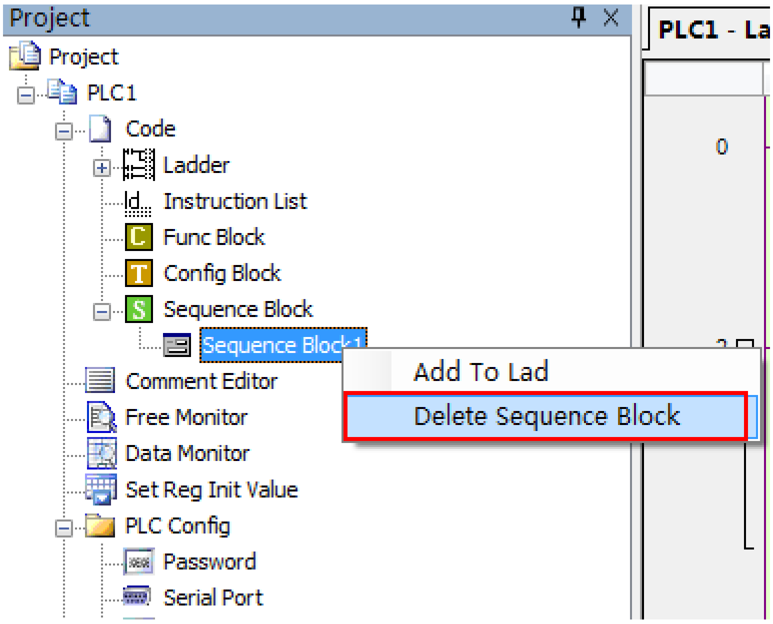
Here we can see that the BLOCK appears at the activate place, see graph below



10-2-3 Delete the BLOCK

If just delete the BLOCK called in the program, you can choose the BLOCK area and delete (refer the previous method).

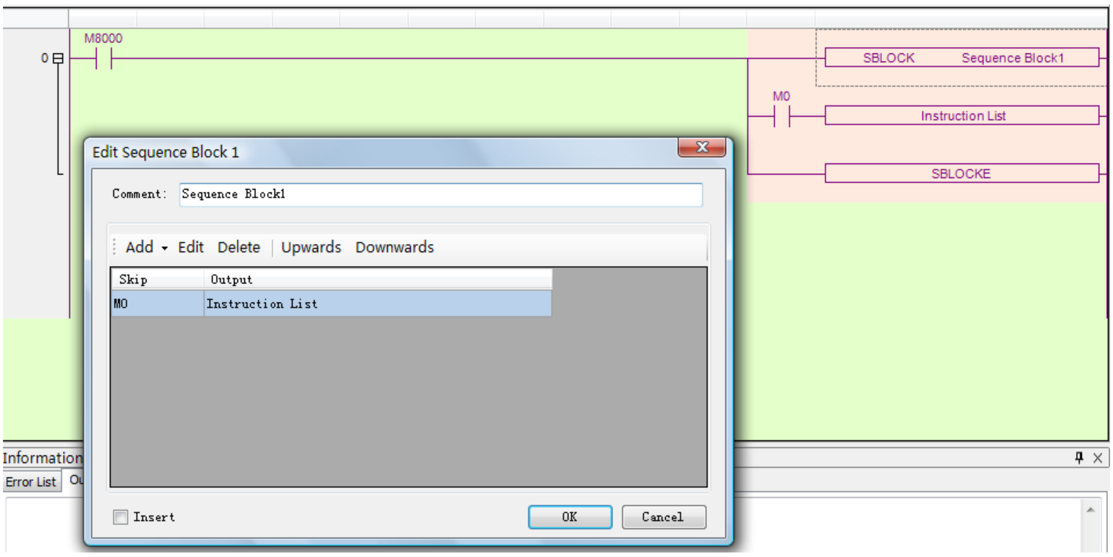
If you want to delete one BLOCK thoroughly, choose “Delete Sequence Block”. After this, you can’t call it any more, the only method is to add it again; see graph below:



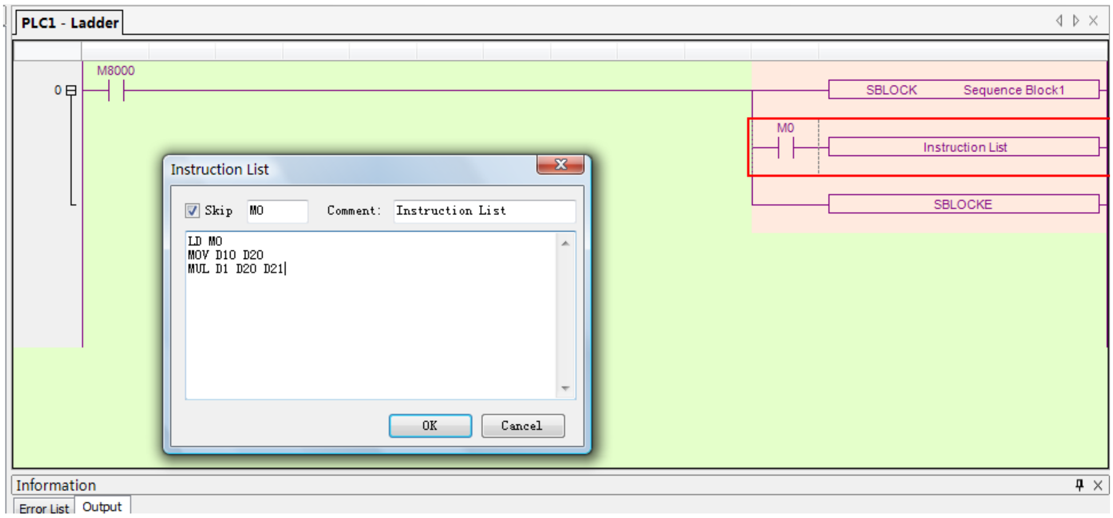
10-2-4 Modify the BLOCK

After adding the BLOCK, if you want to modify it totally, you just click the start and end segments in the ladder window; if you just want to modify a certain program segment, you just double-click the instruction. The two methods are shown below:

（A）Double click the start/end segment of BLOCK:



（B）Double click certain instruction:

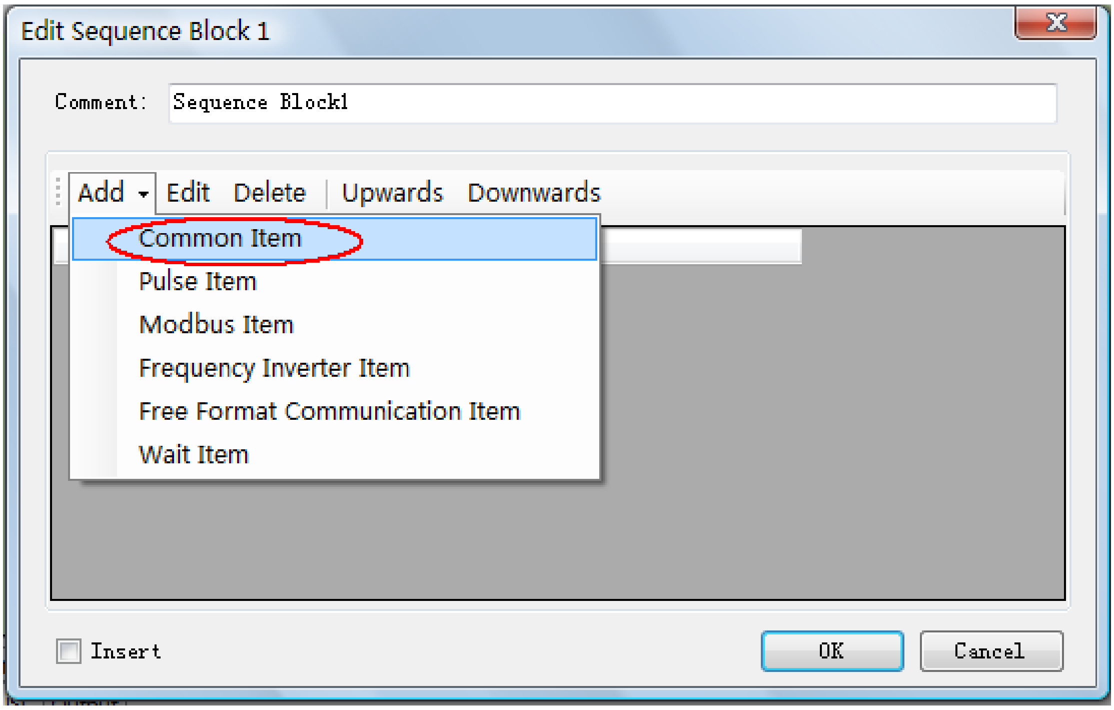


10-3 Edit the internal instructions in BLOCK

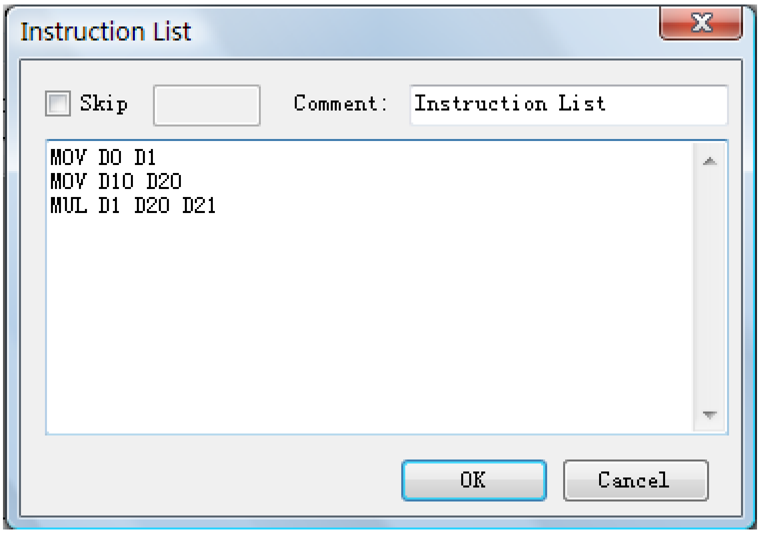
10-3-1 Common Item

In order to add the programs to BLOCK freely, we enable the user to write instructions in form of instruction list.

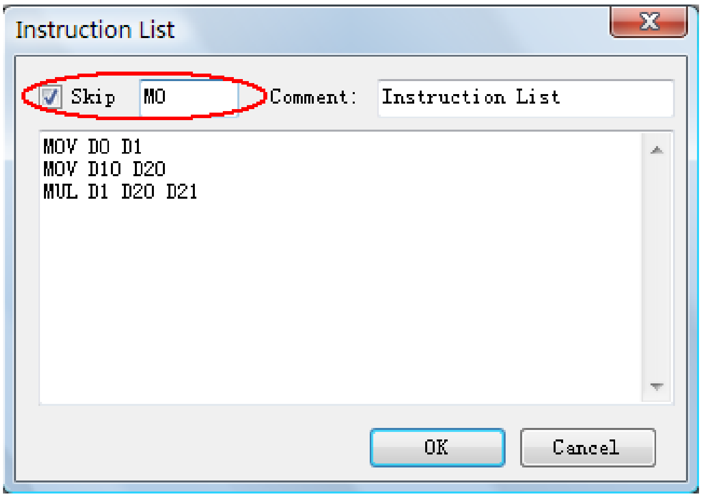
Open the edit interface, click "Add", see graph below:



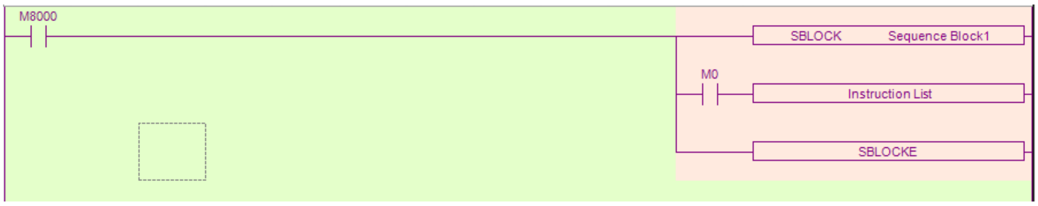
Click "Common Item", a new interface will pop up, see below:



In the interface, user can add the required programs freely. The point to note is that, "Skip" is used to control the run or not on the instructions. If not fill it in, it default to run; if choose "Skip", and fill in the control coil, then when the coil activates, the instructions will not be executed. See below:



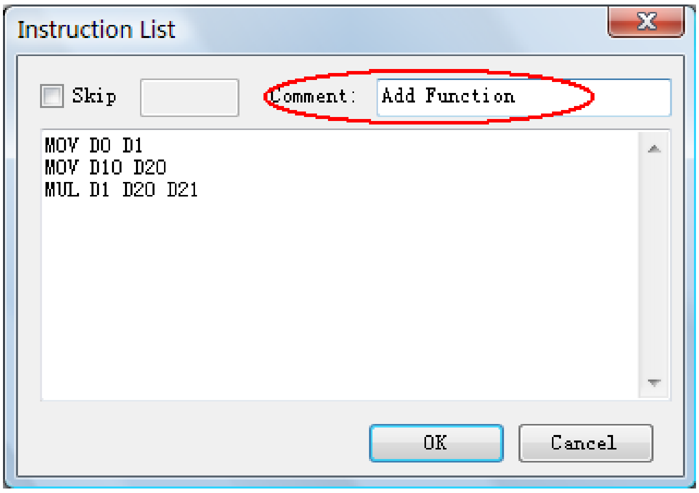
Click "OK", in the ladder you can see program as shown below:



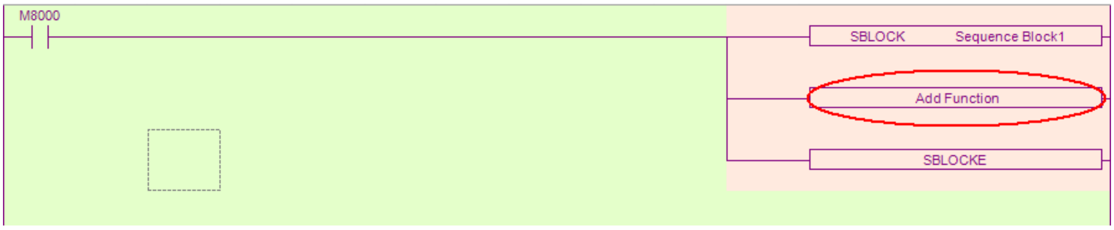
The M0 before "Instruction List" is the condition to run the instruction or not.

Note: In one BLOCK, user can add many program segments, each segment is controlled by "SKIP". If the condition is true, then skip to run the instruction; if the condition is false or vacant, execute the instruction.

In the above graph, the instruction list is not shown in details, but you can add the comments according to the program's function. See below:

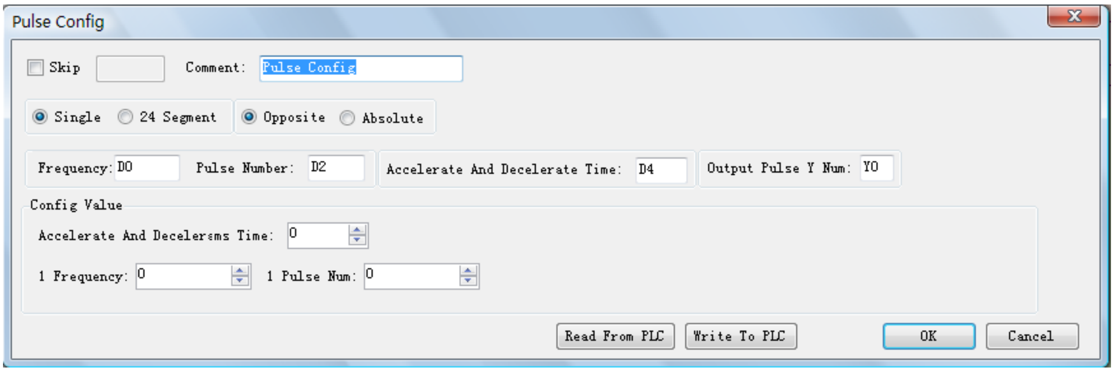


After adding the comment, BLOCK changes in the ladder, see graph below:



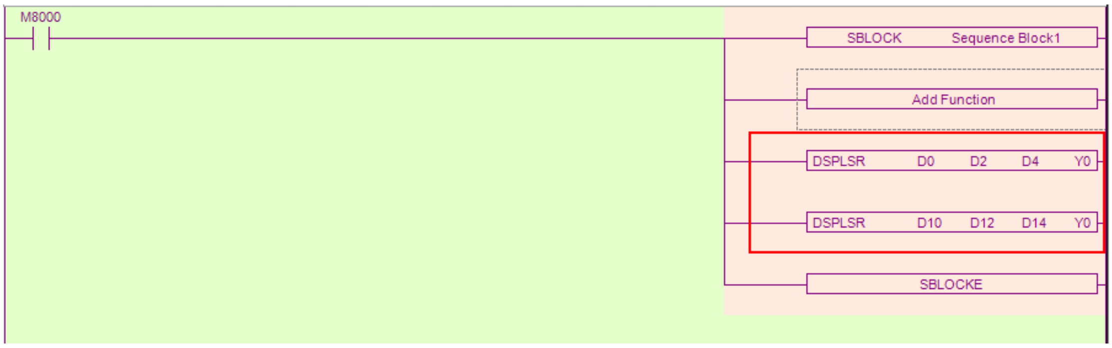
10-3-2 Pulse Configure

Open “Pulse Config” interface with the same method, see below:



In this configure interface, you can set pulse output form, single or 24 segments, opposite or absolute. Write the other parameters in the corresponding blanks, like frequency, pulse, acceleration and deceleration time, pulse number etc.

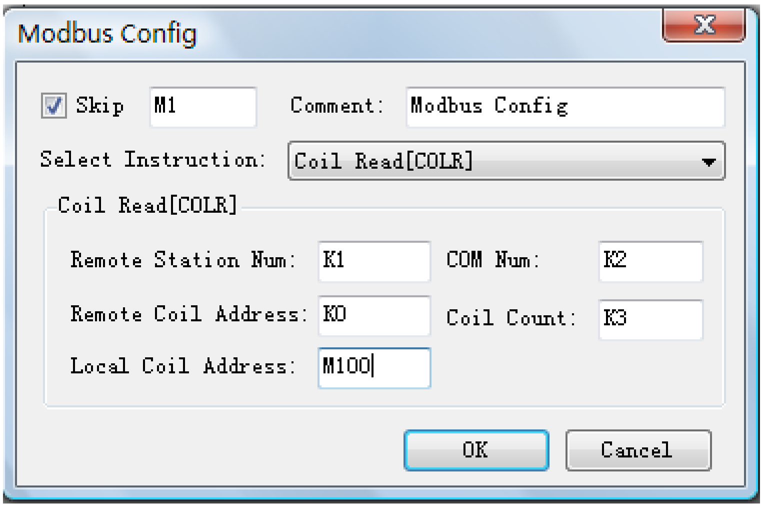
Add two sending pulse instruction into “BLOCK”, see below:



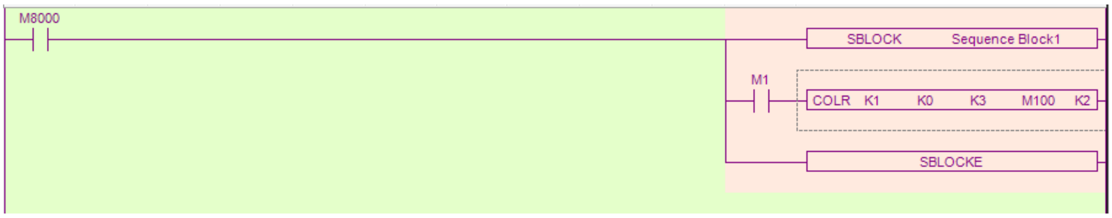
※1: In BLOCK, the pulse output instructions are both in 32 bits form;

10-3-3 Modbus Instruction

As before, open Modbus instructions configure interface, see below:



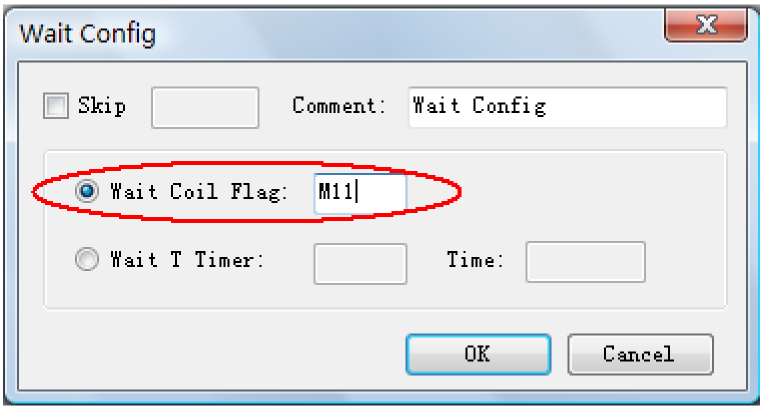
Modbus instructions configuration is easy, just choose "Modbus Item" from the draw down menu, fill in the remote station Nr., COM Nr., local coil ID, coil Nr., the system will generate the instruction automatically. See below:



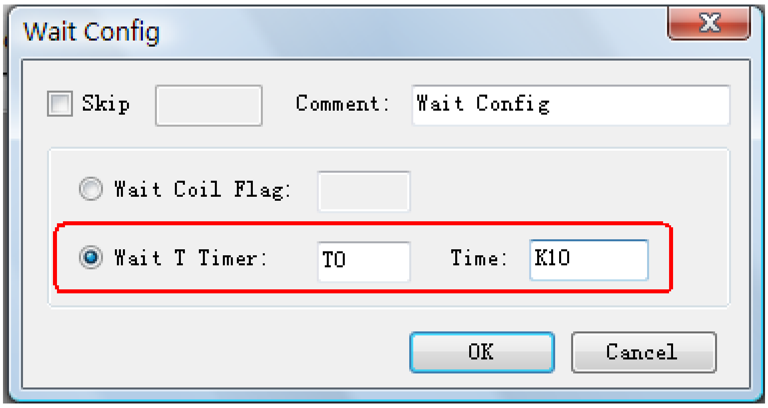
10-3-4 Wait Instruction

Same asthe previous method, open Wait configure interface. Wait instruction is used to wait the flag bit or time. There are two wait forms in the configure interface, one is the flg bit, the other is timer. See the configure method as below:

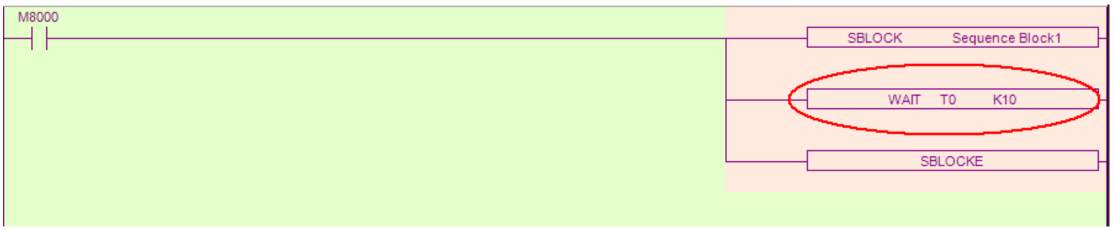
（A）Flag



（B）Timer Wait

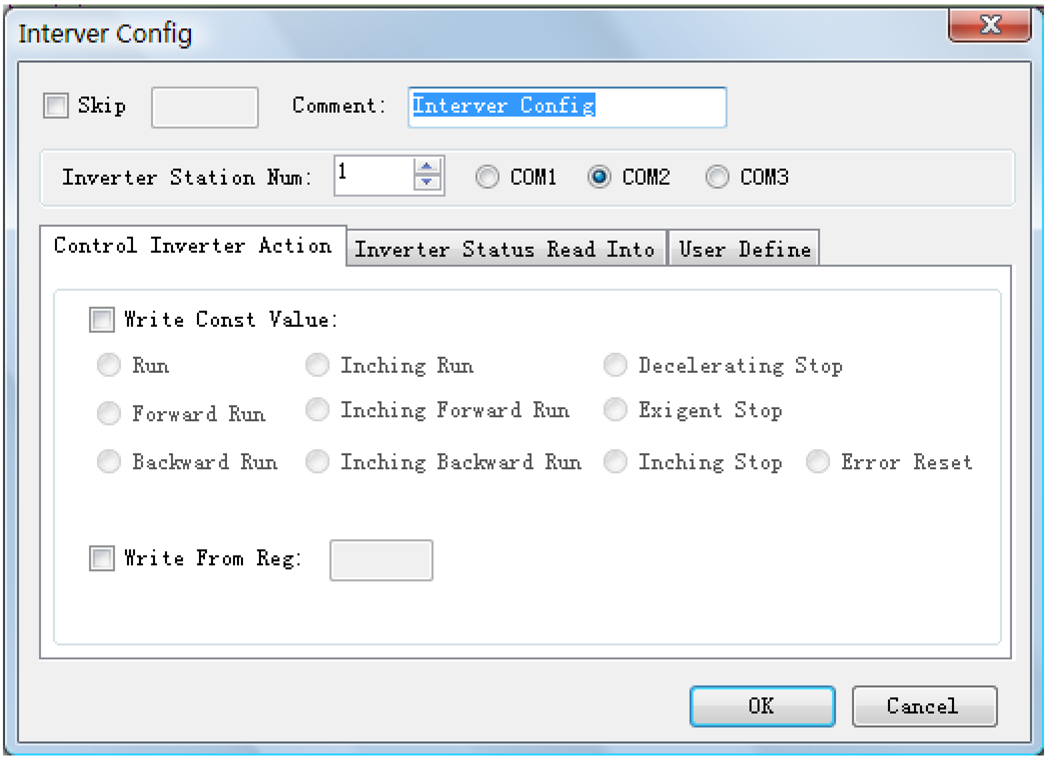


（C）See the result in the ladder



10-3-5 Frequency Inverter Configure

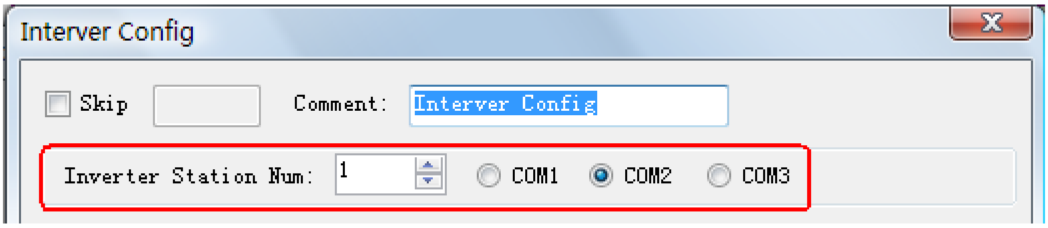
This time is applied for PLCs with XINJE inverters. By changing this interface, user can read/write the inverters. See below:



The interface includes four parts, they are: inverter station number, COM port number, control inverter action, monitor inverter's status, user define etc. Below we introduce the four parts one by one:

（A）inverter's station number and COM port

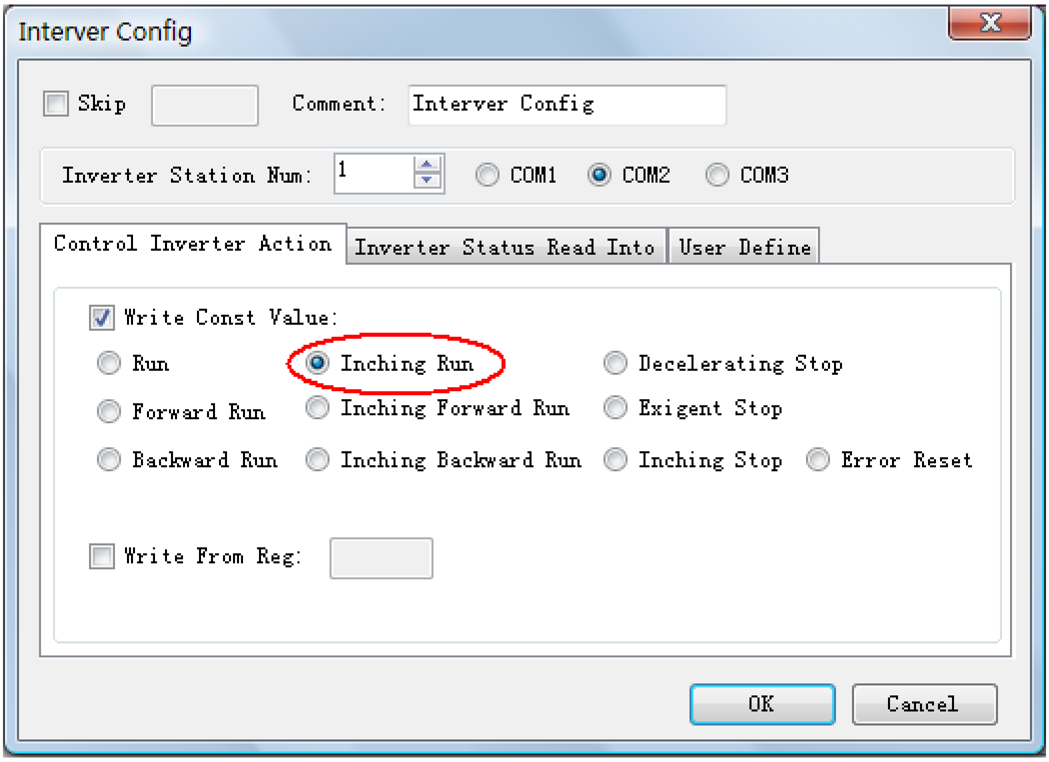
The station number is used to specify the inverter's station number, the COM port is PLC's COM port, see the configuration below:



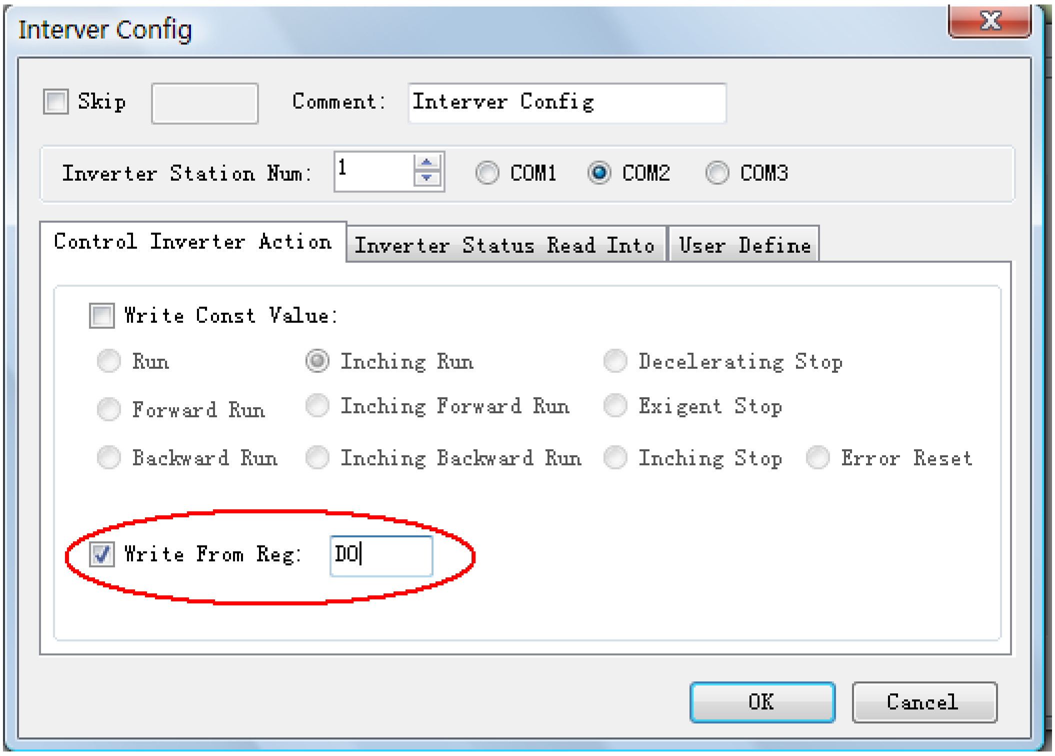
（B）Control Inverter's Action

This item includes "write constant value" and "write from register". "write constant value" specify the inverter's running manner directly; "write from register" decide the inverter's running manner according to register's value:

The first form is very easy, choose the required operation directly, see graph below:

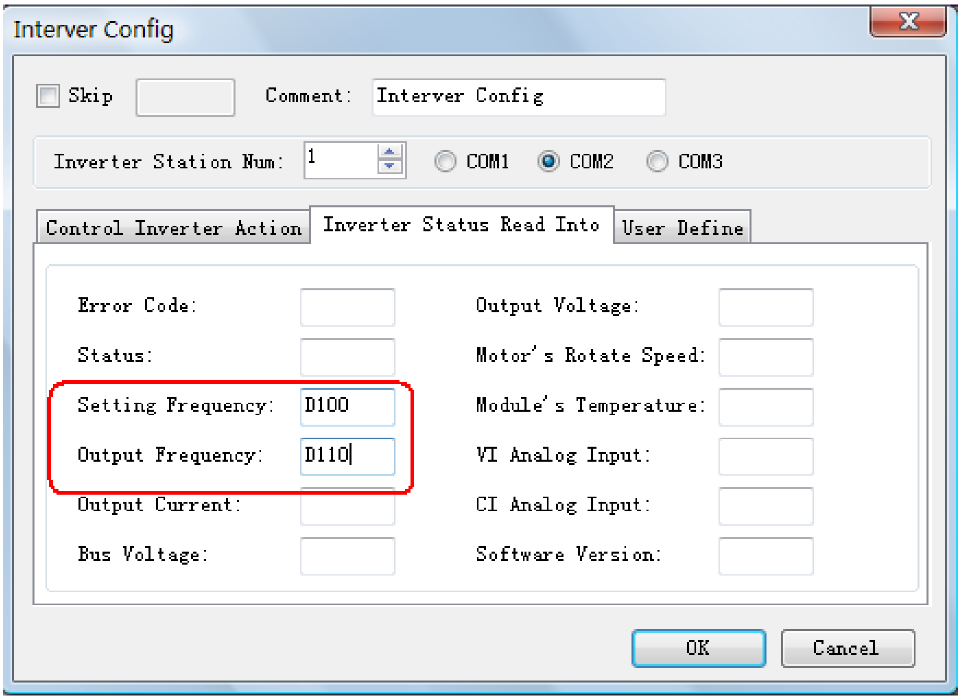


For the second form, we take an example to show: write D0 into inverter:



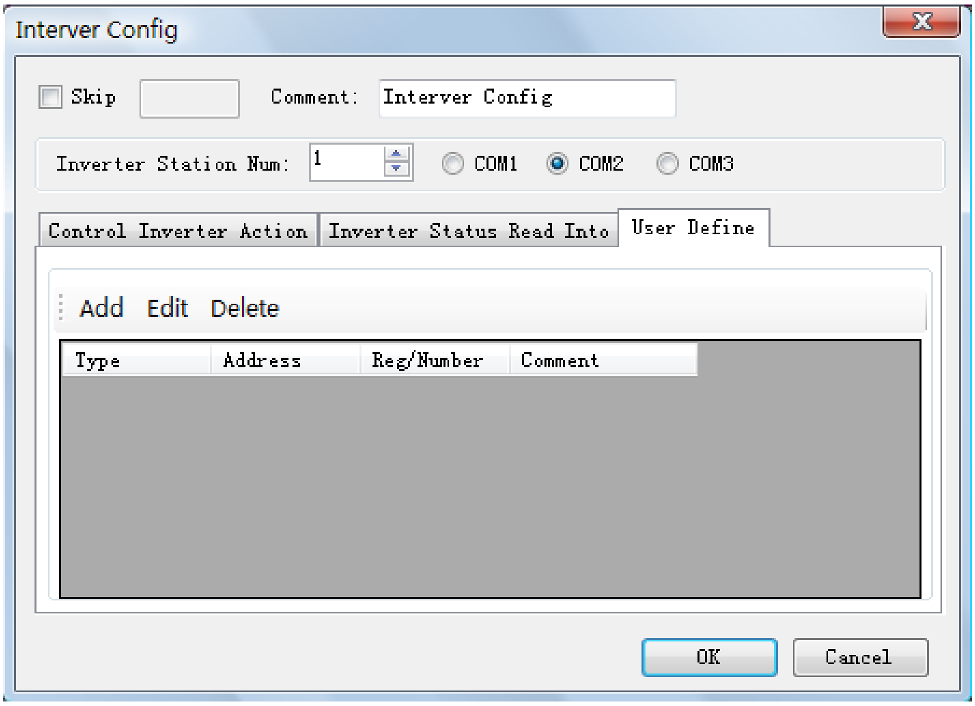
（C）Inverter Status Read Into

This is used to read inverter's status. According to the object shown on interface, insert the value into the specified register in PLC, see below:

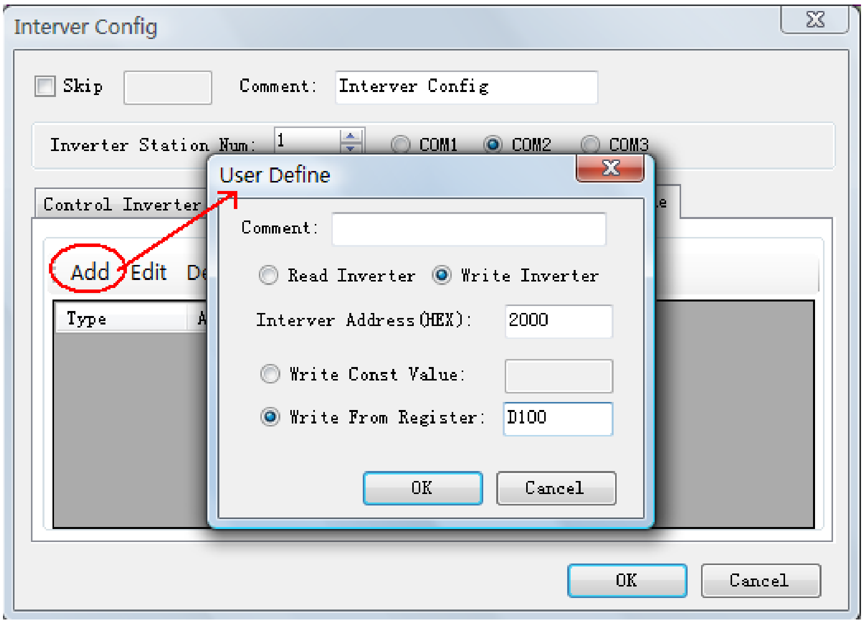


（D）User Define

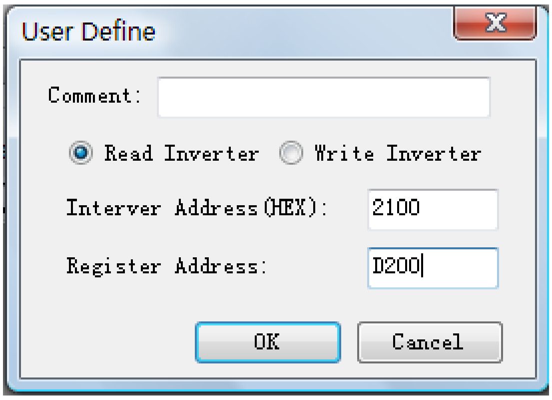
Set the inverter via user define mode, read from and write into inverter directly. The configure interface is shown below:



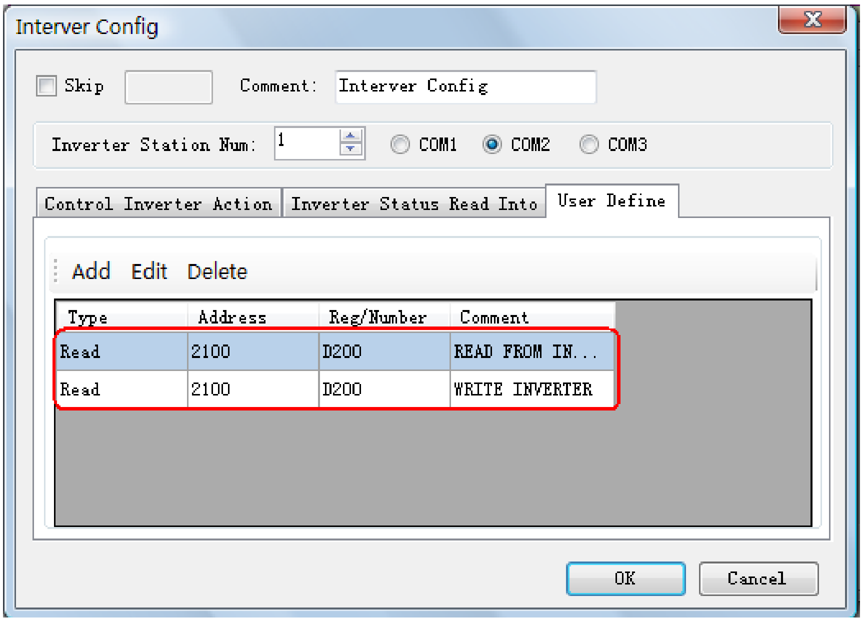
Add a write instruction, see configuration below:



Add a read instruction:



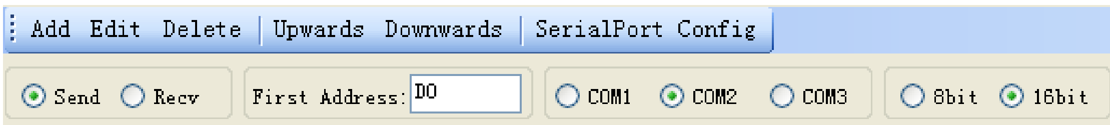
See the result below:



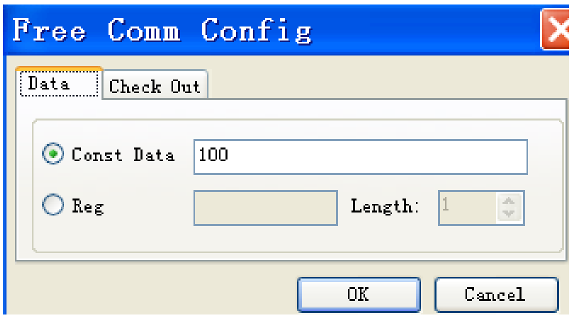
10-3-6 Free Format Communication

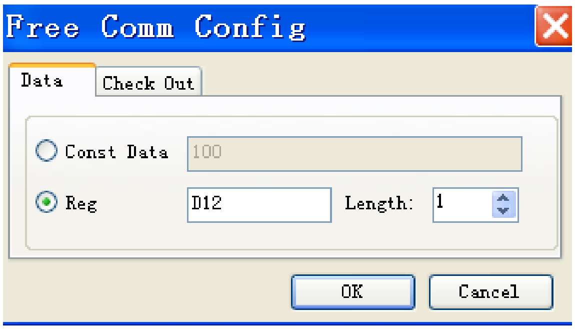
Add free format communication instructions in the block.

For example, select “send” instruction, first address set to D0, serial port is 2, 16 bits.

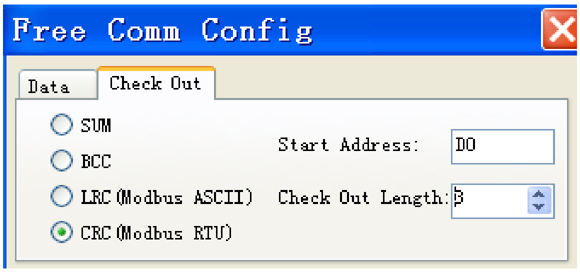


There are two methods to set the data. Const data is to set the value directly. Reg is to set the value via register.

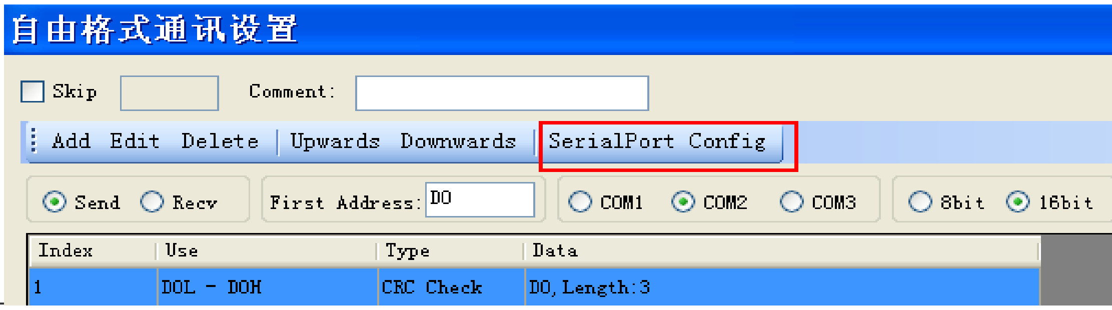




Change to check out tab, select the checking mode.



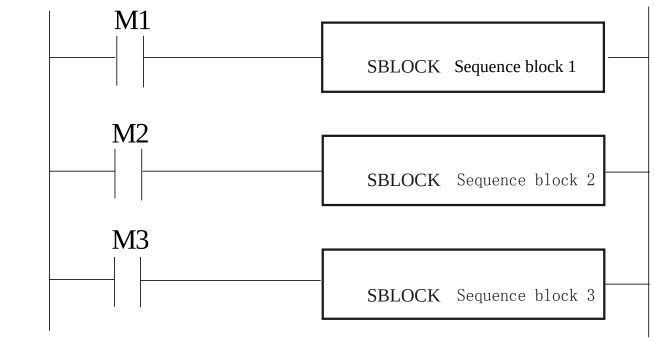
The communication parameters also need to be set. Click “serial port config”:

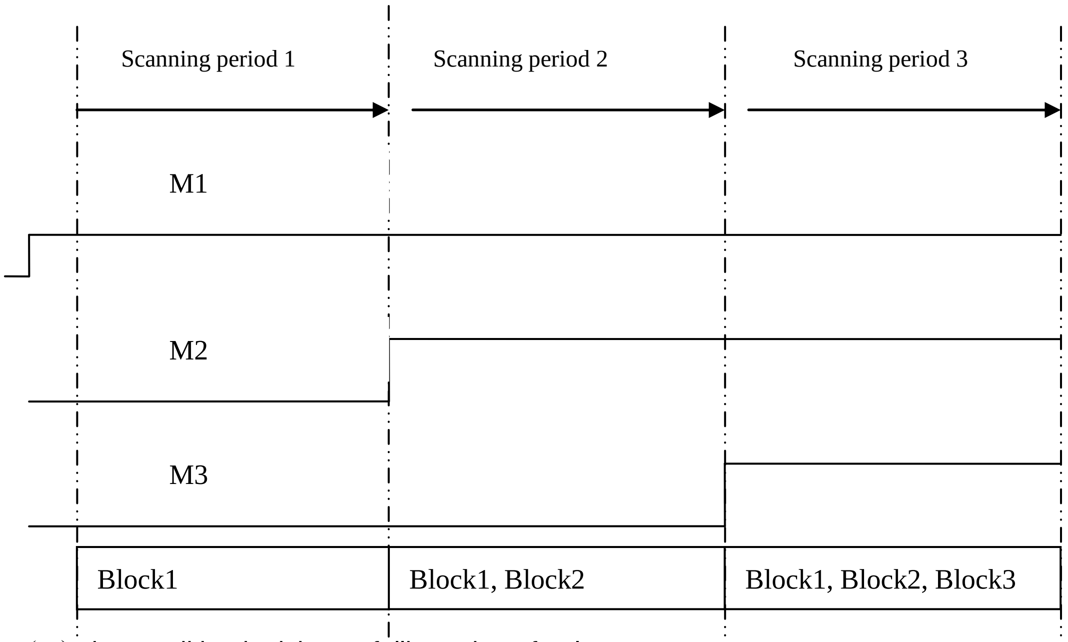


10-4 Running Form of the BLOCK

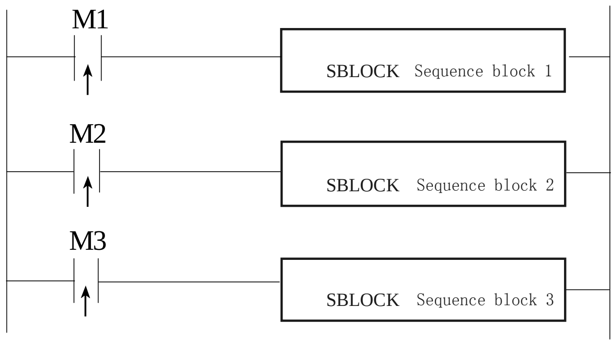
1: If there are many blocks, they run as the normal program. The block is running when the condition is ON.

（A）the condition is normal ON, normal OFF coil





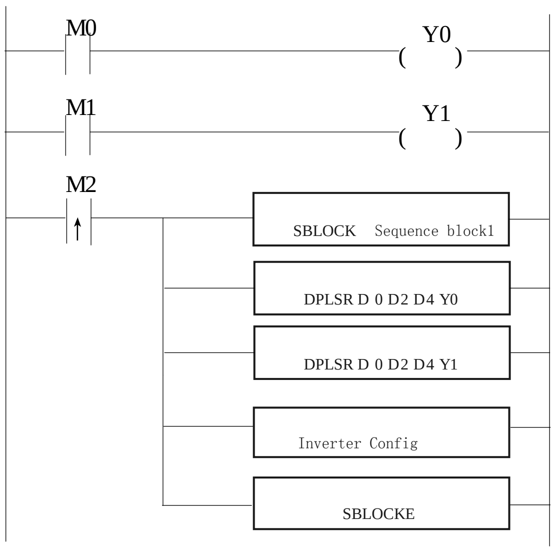
（B）the condition is rising or falling edge of pulse



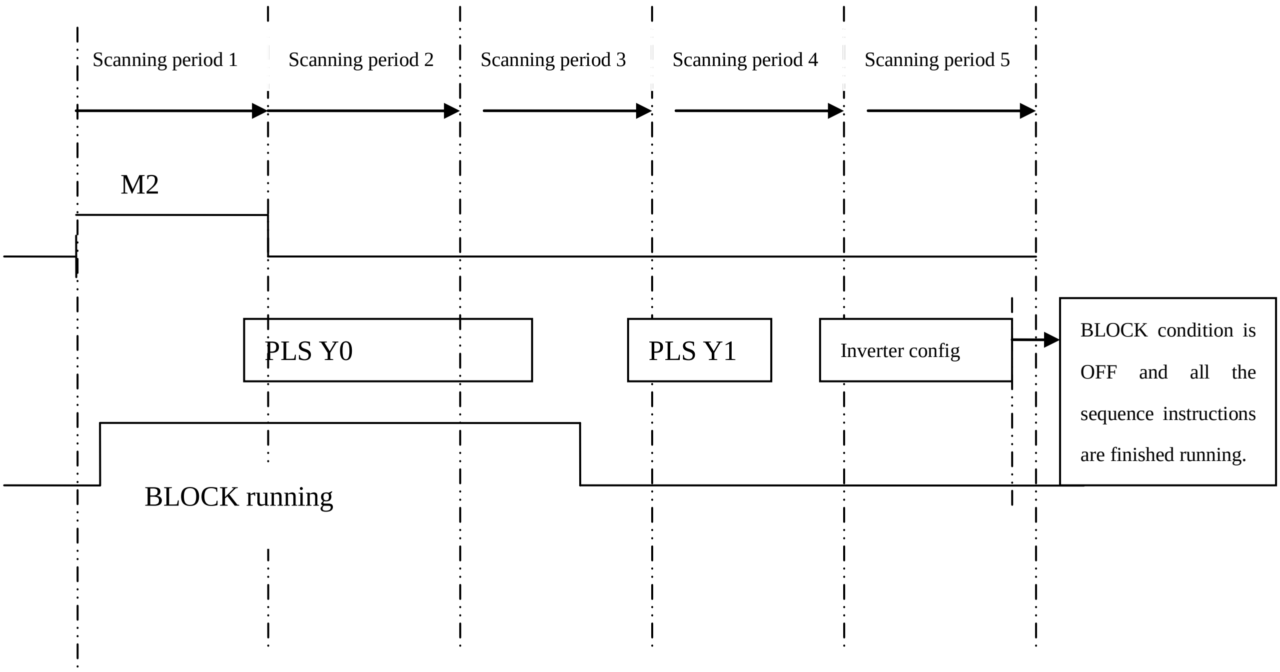
When M1, M2, M3 is from OFF to ON, all these blocks will run once.

2: The instructions in the block run in sequence according to the scanning time. They run one after another when the condition is ON.

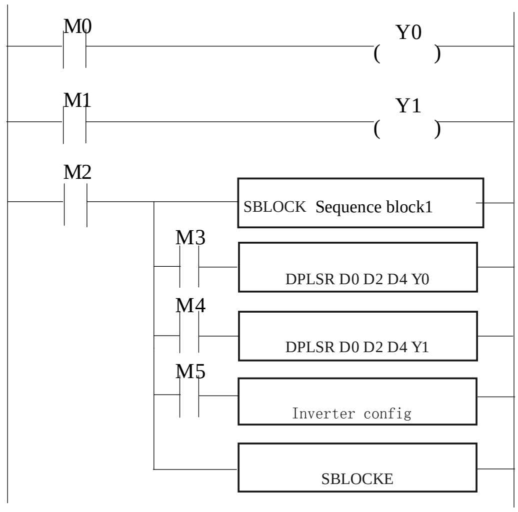
（A）Without SKIP condition



The instructions running sequence in block 1 is shown as below:



（B）With SKIP condition



Explanation:

A) When M2 is ON, block 1 is running.

B) All the instructions run in sequence in the block.

C) M3, M4, M5 are the sign of SKIP, when they are ON, this instruction will not run.

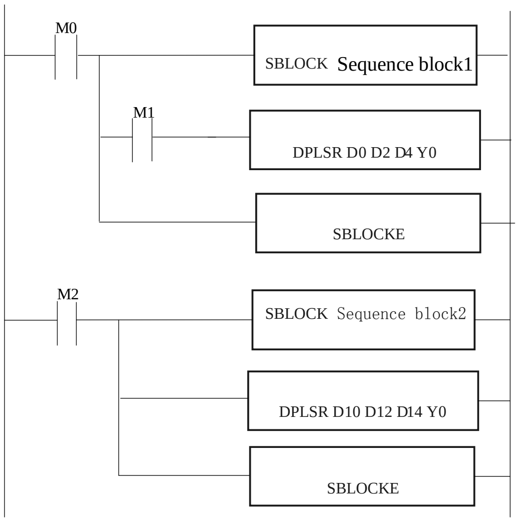
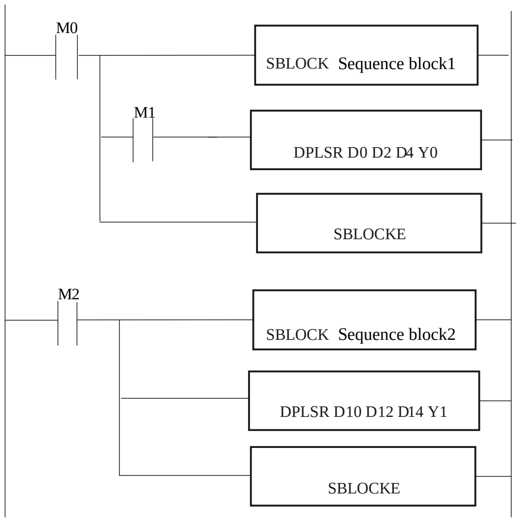
D) When M3 is OFF, if no other instructions use this Y0 pulse , DPLSR D0 D2 D4 Y0 will run; if not, the DPLSR D0 D2 D4 Y0 will run after it is released by other instructions.

E) After “DPLSR D0 D2 D4 Y0” is over, check M4. If M4 is OFF, check “DPLSR D0 D2 D4 Y1”, if M4 is ON, check M5. If M5 is OFF, “inverter config” will run.

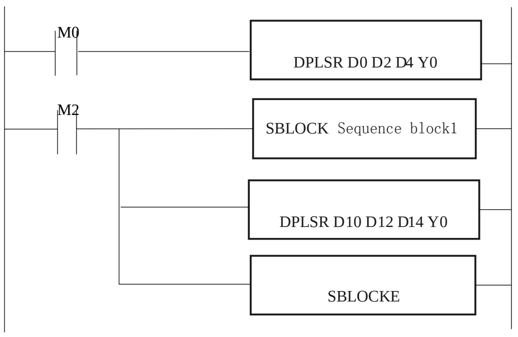
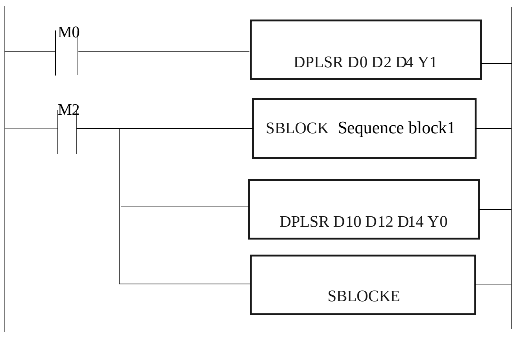
10-5 BLOCK instruction editing rules

In the BLOCK, when Instruction Editing follow the rules below:

1:Do not use the same pulse output terminal in different BLOCK.

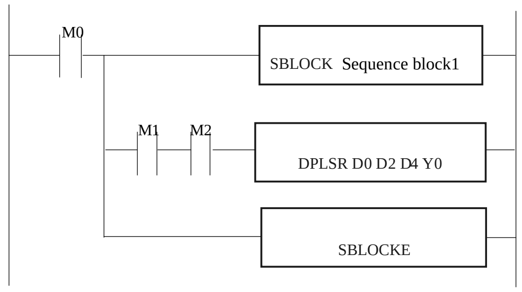
NO（×）	YES（√）
	

2: Do not use the same pulse output terminal in BLOCK and main program.

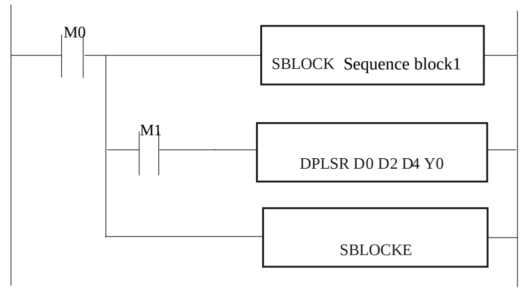
NO（×）	YES（√）
	

3: There only can be one SKIP condition for one BLOCK instruction.

NO（×）

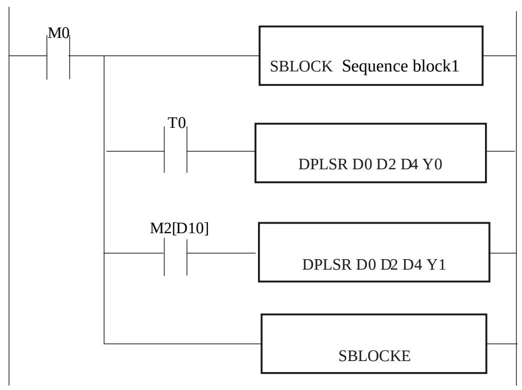


YES（√）

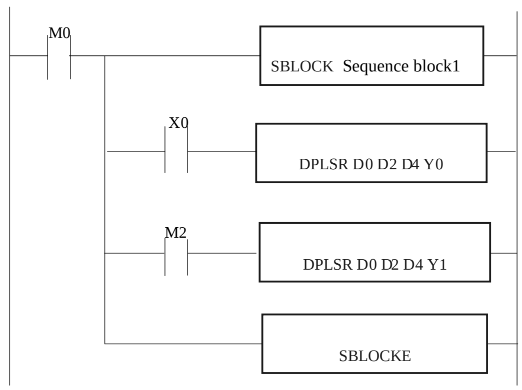


4: The SKIP condition only can use M, X, can not use other coil or register.

NO（×）

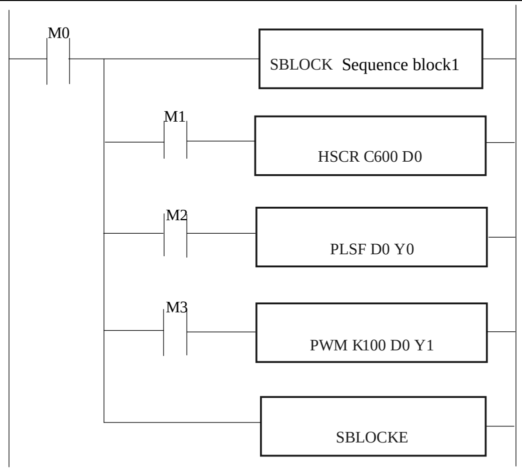


YES（√）

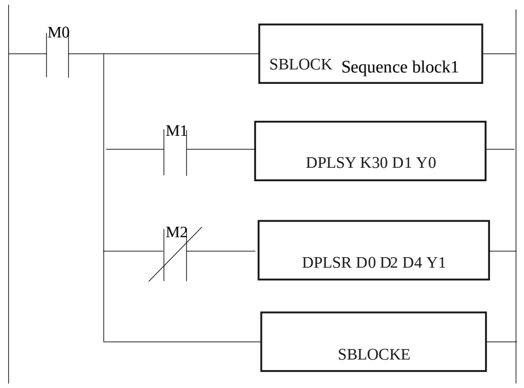


5: The output instructions can not be HSC, PLSF, PWM, FRQM.

NO（×）



YES（√）



6、LabelKind type can not be used in the block. Sign P, I can not be used in block. (they can be added to the block but the program does not support this).

10-6-1 Instruction Explanation

➢ Stop Running the BLOCK [BSTOP]

1: Summarization

Stop the instructions running in the block

[BSTOP]			
16 bits	BSTOP	32 bits	-
Condition	NO,NC coil and pulse edge	Suitable types	XC1、XC2、XC3、XC5、XCM
Hardware	V3.1i and above	Software	V3.1h and above

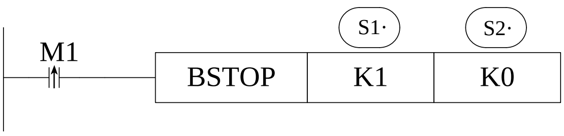
2: Operand

Operand	Function	Type
S1	The number of the BLOCK	16 bits, BIN
S2	The mode to stop the BLOCK	16 bits, BIN

3: Suitable component

Word component	Operand	Register									Constant	Module	
		D	FD	ED	TD	CD	DX	DY	DM	DS	K/H	ID	QD
	S1	●									●		
	S2										K		

Function



- S2 is the mode to stop BLOCK, operand K1, K2

K0: stop the BLOCK slowly, if the pulse is outputting, the BLOCK will stop after the pulse outputting is finished.

K1: stop the BLOCK immediately; stop all the instructions running in the BLOCK.

➢ Continue Running the BLOCK [BGOON]

1: Summarization

This instruction is opposite to BSTOP. To continue running the BLOCK.

[BGOON]			
16 bits	BGOON	32 bits	-
Condition	Pulse edge	Suitable types	XC1、XC2、XC3、XC5、XCM
Hardware	V3.1i and above	Software	V3.1h and above

2: Operand

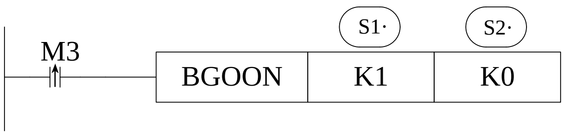
Operand	Function	Type
S1	The number of the BLOCK	16 bits, BIN
S2	The mode to continue running the BLOCK	16 bits, BIN

3: Suitable component

Word Component

Operand	Register									Constant	Module	
	D	FD	ED	TD	CD	DX	DY	DM	DS	K/H	ID	QD
S1	●									●		
S2										K		

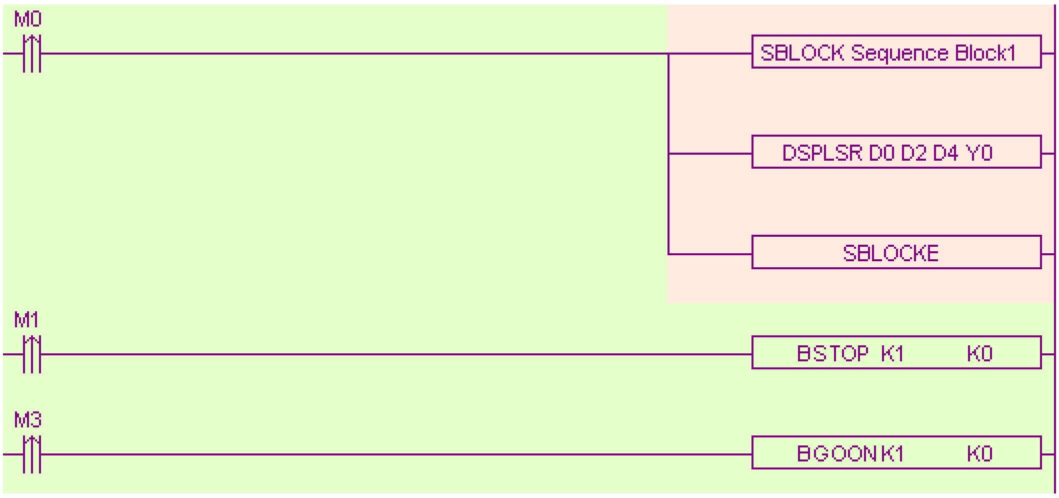
Function

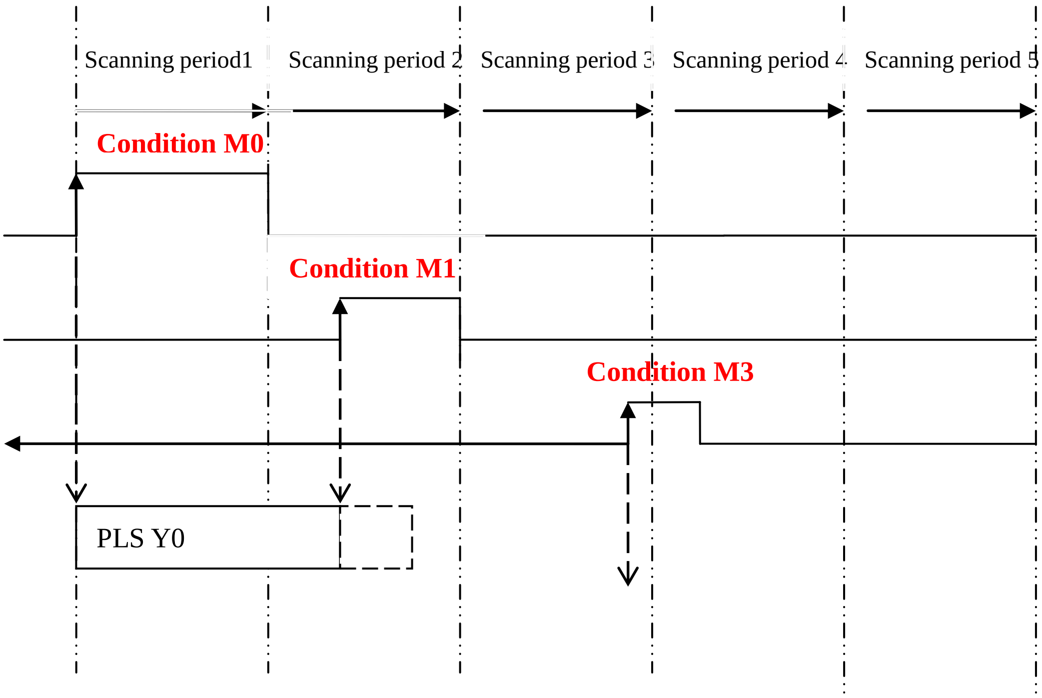


- S2 is the mode to continue running the BLOCK. Operand: K0, K1. K0: continue running the instructions in the BLOCK. For example, if pulse outputting stopped last time, BGOON will continue outputting the rest pulse. K1: continue running the BLOCK, but abandon the instructions have not finished last time. Such as the pulse output instruction, if the pulse has not finished last time, BGOON will not continue outputting this pulse but go to the next instruction in the BLOCK.

10-6-2 The timing sequence of the instructions

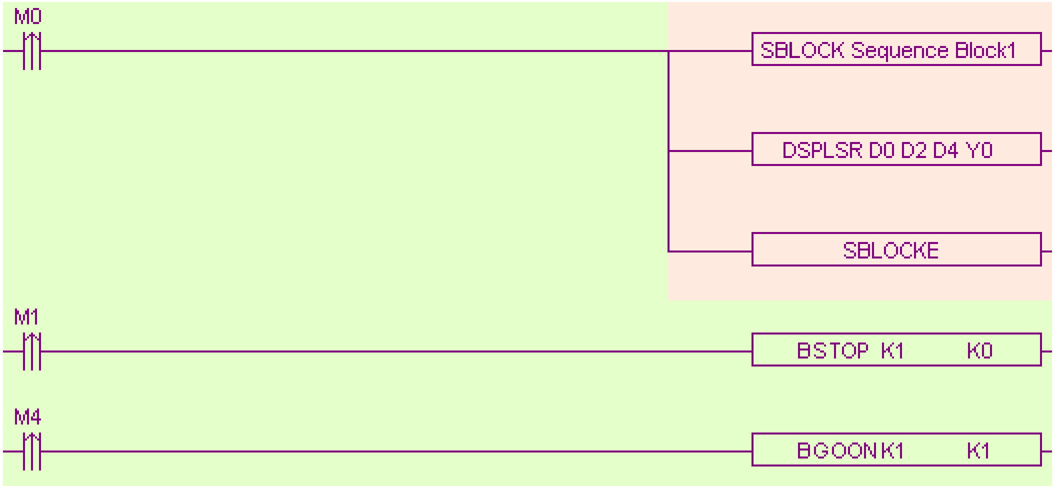
1: BSTOP（K1 K0）+BGOON（K1 K0）

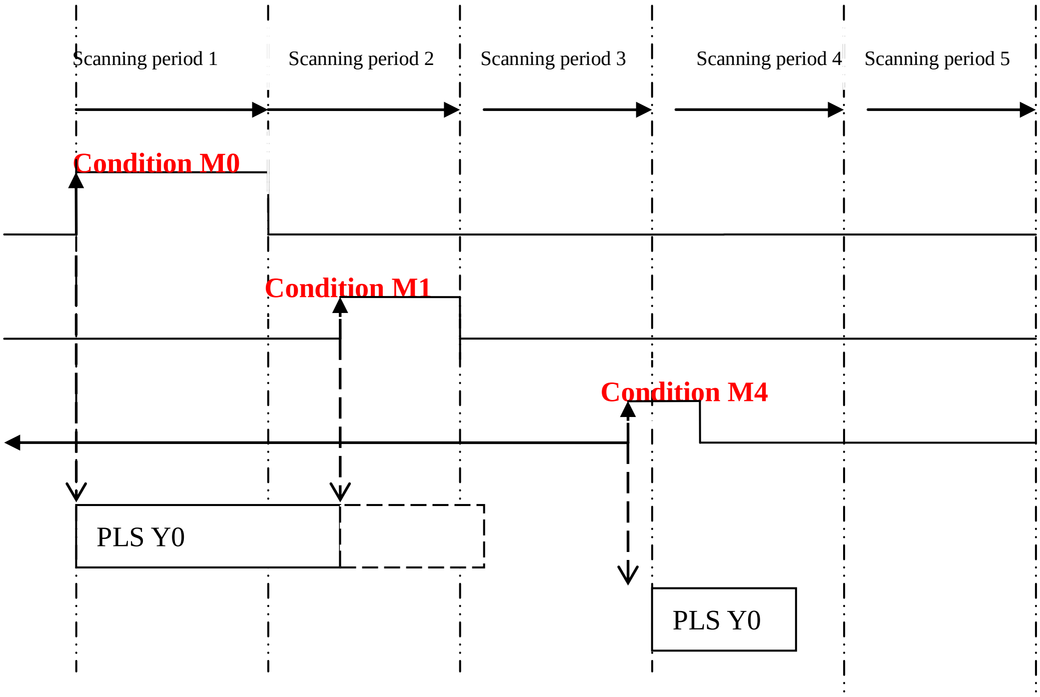




When M0 is from OFF→ON, run “DSPLSR D0 D2 D4 Y0” in the BLOCK to output the pulse; when M1 is from OFF→ON, the BLOCK stops running, pulse outputting stops at once; when M3 is from OFF→ON, abandon the rest pulse.

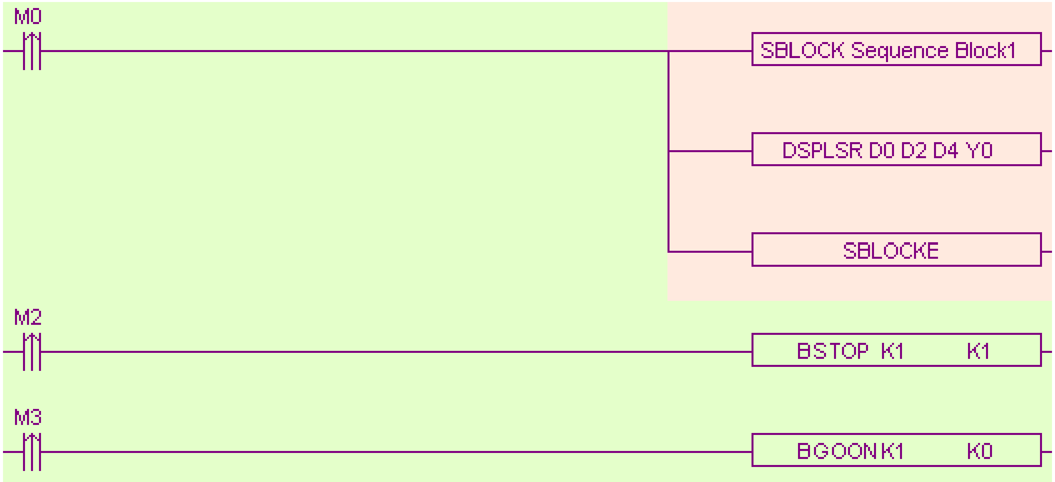
2: BSTOP（K1 K0）+BGOON（K1 K1）

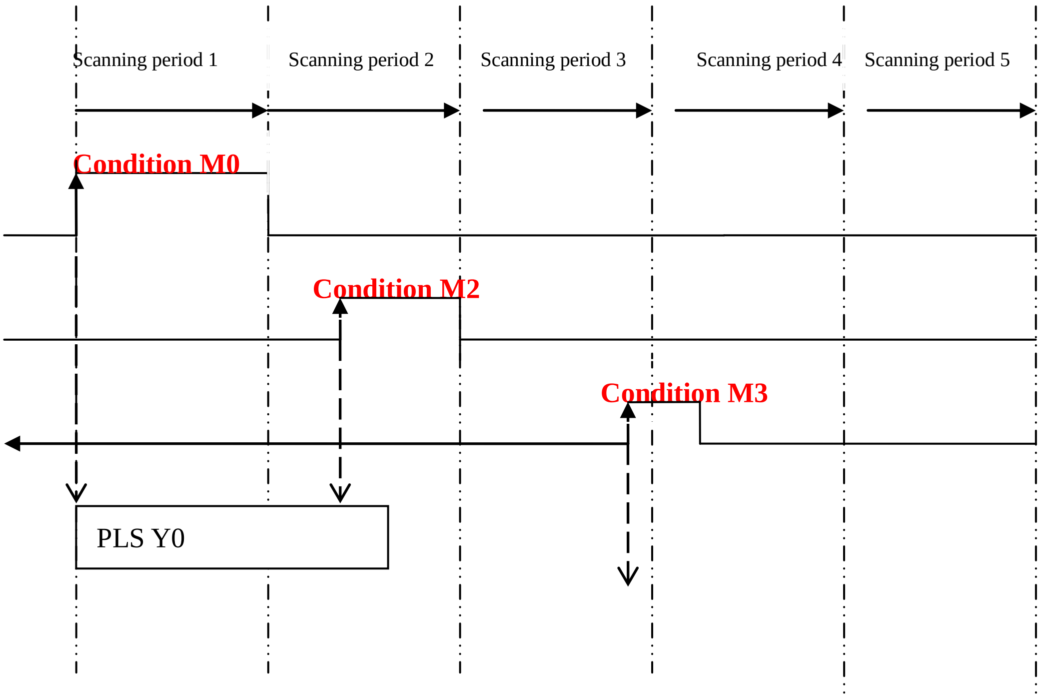




When M0 is from OFF→ON, run “DSPLSR D0 D2 D4 Y0” in the BLOCK to output the pulse; when M1 is from OFF→ON, the BLOCK stops running, the pulse outputting stops at once; when M4 is from OFF→ON, output the rest pulses.

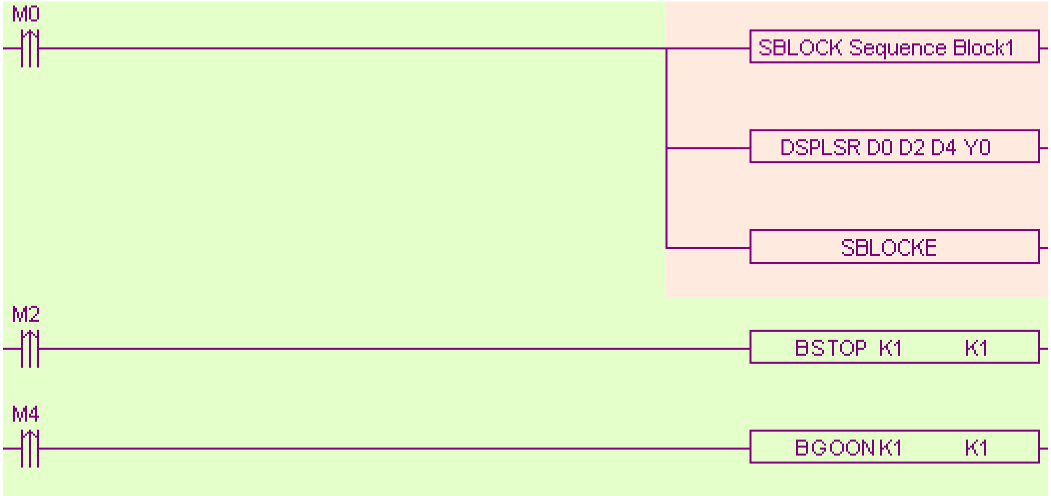
3: BSTOP（K1 K1）+BGOON（K1 K0）

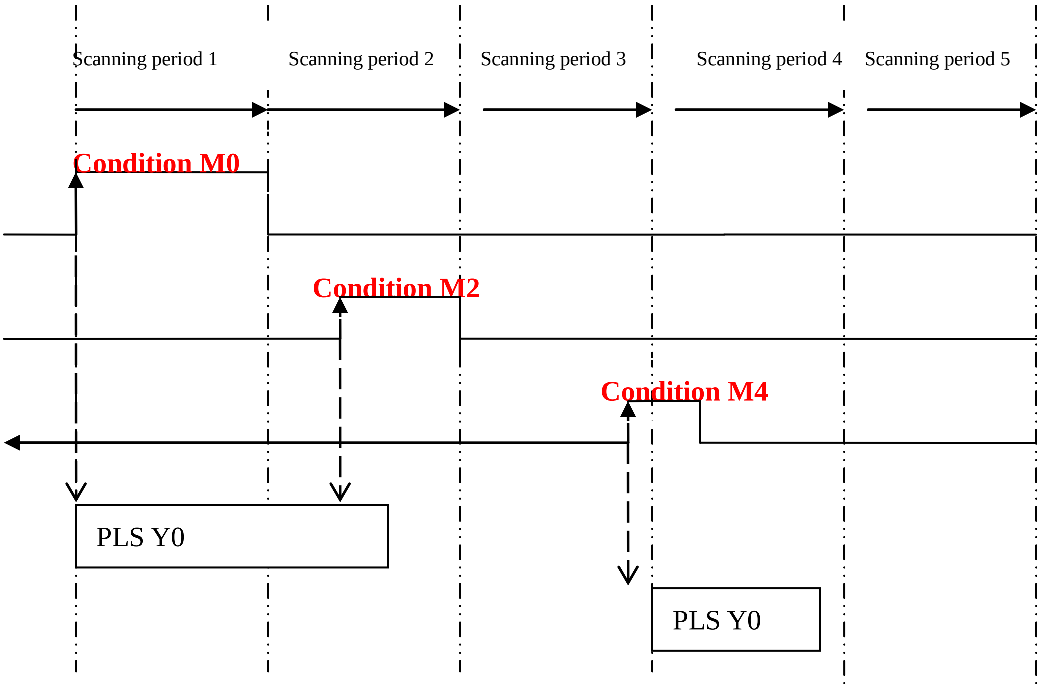




When M0 is from OFF→ON, run “DSPLSR D0 D2 D4 Y0” in the BLOCK to output the pulse; when M2 is from OFF→ON, stop the BLOCK, the pulse will stop slowly with slope, when M3 is from OFF→ON, discards the rest pulses.

4: BSTOP（K1 K1）+BGOON（K1 K1）





When M0 is from OFF→ON, run “DSPLSR D0 D2 D4 Y0” in the BLOCK to output the pulse; when M2 is from OFF→ON, stop running the BLOCK, the pulse will stop slowly with slope; when M4 is from OFF→ON, output the rest pulses.

Please note that though the BSTOP stops the pulse with slope, there maybe still some pulses; in this case, if run BGOON K1 K1 again, it will output the rest of the pulses.

10-7 BLOCK Flag Bit and Register

1:BLOCK flag bit:

Address	Function	Explanation
M8630		1: running 0: not running
M8631	BLOCK1 running flag	
M8632	BLOCK2 running flag	
…….	…….	
……..	…….	
M8730	BLOCK100 running flag	

2: BLOCK flag register

Address	Function	Explanation
D8630		BLOCK use this value when monitoring
D 8631	BLOCK1 current running instruction	
D8632	BLOCK2 current running instruction	
…….	…….	
……..	…….	
D8730	BLOCK10 current running instruction	

10-8 Program Example

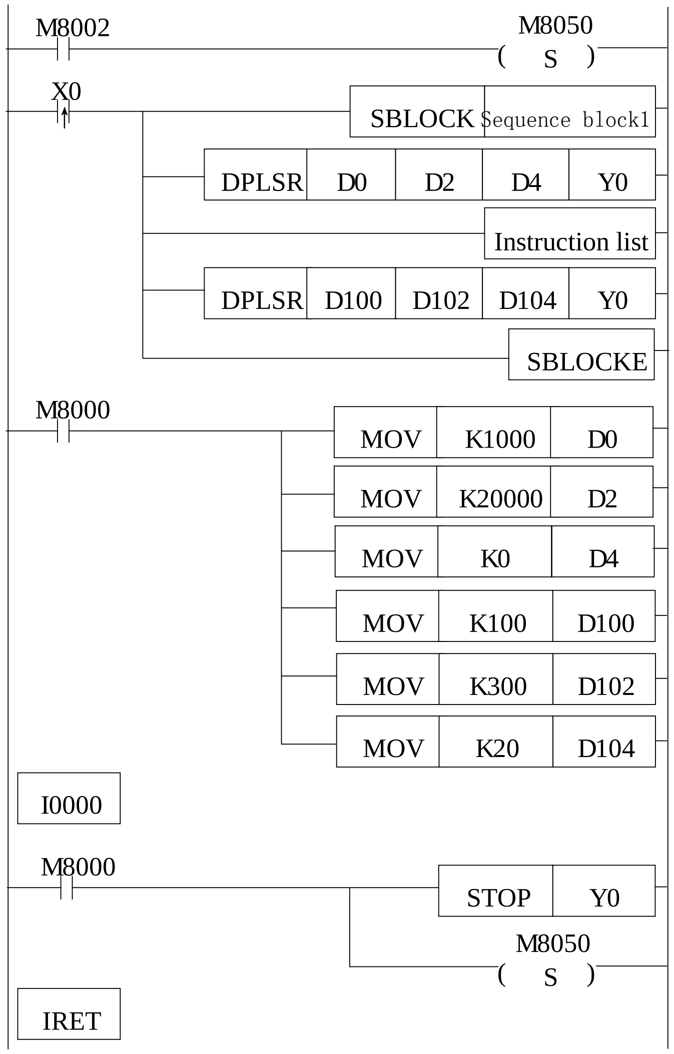
Example:

This example is used in the tracking system. The process as follows:

Output some pulses and prohibit exterior interruption.

Continue outputting the pulse but at low speed, and allow exterior interruption. When checked the exterior cursor signal, stop the pulse outputting and machine running.

Ladder chart:



PLC power on, prohibit exterior interruption

BLOCK starts

Output the pulses and move some distance

Reset M8050, open exterior interruption

Output the pulses at low speed

BLOCK ends

The first pulse frequency

The first pulse numbers

Accelerate/decelerate time for the first pulse

The second pulse frequency

The second pulse numbers

Accelerate/decelerate time for the second pulse

The interruption starts

Stop outputting the pulse

Close the interruption

The interruption ends

The instruction list content:

RST M8050

Notes:

M8050: prohibit the exterior interruption

11 Special Function Instructions

In this chapter, we introduce PWM pulse width modulation, frequency detect, precise time, interruption etc;

Mnemonic	Function	Circuit and soft components	Chapter
Pulse Width Modulation, Frequency Detection			
PWM	Output pulse with the specified occupied ratio and frequency	PWM S1 S2 D	11-1
FRQM	Frequency Detection	FRQM S1 D S2 S3	11-2
Time			
STR	Precise Time	STR D1 D2	11-3
STRR	Read Precise Time Register	STRR S	11-3
STRS	Stop Precise Time	STRS S	11-3
Interruption			
EI	Enable Interruption	EI	11-4-1
DI	Disable Interruption	DI	11-4-1
IRET	Interruption Return	IRET	11-4-1

11-1 PWM Pulse with Modulation

1: Instruction's Summary

Instruction to realize PWM pulse width modulation

PWM pulse width modulation [PWM]			
16 bits instruction	PWM	32 bits instruction	-
execution condition	normally ON/OFF coil	suitable models	XC1、XC2、XC3、XC5、XCM
hardware requirement	-	software requirement	-

2: Operands

Operands	Function	Type
S1	specify the occupy ratio value or soft component's ID number	16 bits, BIN
S2	specify the output frequency or soft component's ID number	16 bits, BIN
D	specify the pulse output port	bit

3: Suitable Soft Components

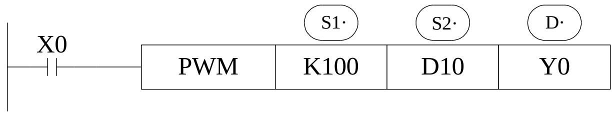
Word

Operands	System									Constant	Module	
	D	FD	ED	TD	CD	DX	DY	DM	DS	K/H	ID	QD
S1	●	●		●	●					●		
S2	●	●		●	●					●		

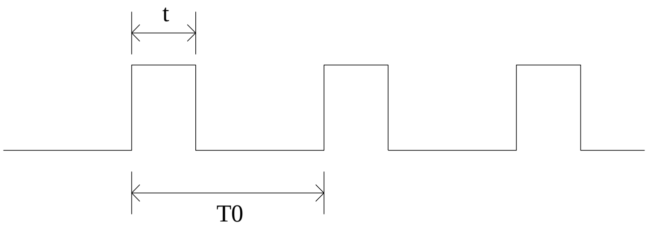
Bit

Operands	System						
	X	Y	M	S	T	C	Dn.m
D		●					

Function and Action



- The occupy ratio **n**: 1~255
- Output pulse **f**: 0~72KHz
- Pulse is output at Y000 or Y001 (Please use transistor output)
- The output occupy/empty ratio of PMW =n /256×100%
- PWM output use the unit of 0.1Hz, so when set (S2) frequency, the set value is 10 times of the actual frequency (i.e. 10f). E.g.：to set the frequency as 72KHz, then set value in (S2) is 720000.
- When X000 is ON, output PWM wave；when X000 is OFF, stop output. PMW output doesn’t have pulse accumulation.



In the left graph: T0=1/f
T/T0=n/256

11-2 Frequency Testing

1: Instruction's Summary

Instruction to realize frequency testing

frequency testing [FRQM]			
16 bits instruction	FRQM	32 bits instruction	-
execution condition	normally ON/OFF coil	suitable models	XC1、XC2、XC3、XC5、XCM
hardware requirement	-	software requirement	-

2: Operands

Operands	Function	Type
S1	Specify the sampling pulse number or soft component's ID number	16 bits, BIN
S2	Specify the frequency division choice's number	16 bits, BIN
S3	Specify the pulse input port	bit
D	specify the tested result's soft component's number	16 bits, BIN

3: Suitable Soft Components

Word

Operands	System									Constant	Module	
	D	FD	ED	TD	CD	DX	DY	DM	DS	K/H	ID	QD
S1	●			●	●					●		
S2										●		
D	●			●	●							

Bit

Operands	System						
	X	Y	M	S	T	C	Dn.m
S3	●						

Function and Action

X000

	S1·	D·	S2·	S3·
FRQM	K20	D100	K1	X003

- S1: sampling pulse number: the number to calculate the pulse frequency
- D: tested result, the unit is Hz.
- S2: Frequency division choice. It can be K1 or K2;
  When the frequency division is K1, the range is: no less than 9Hz, precision range: 9~18KHz.
  When the frequency division is K2, the range: no less than 300Hz, precision range: 300~400KHz.
- In frequency testing, if choose frequency division as K2, the frequency testing precision is higher than frequency division K1.
- When X000 is ON, FRQM will test 20 pulse cycles from X003 every scan cycle. Calculate the frequency’s value and save into D100. Test repeatedly. If the tested frequency’s value is smaller than the test bound, then return the test value as 0.

The pulse output to X number:

Model		X Number
XC2 series	14/16/24/32/48/60 I/O	X1、X6、X7
XC3 series	14 I/O	X2、X3
	24/32 I/O	X1、X11、X12
	48/60 I/O、XC3-19AR-E	X4、X5
XC5 series	24/32 I/O	X3
	48/60 I/O	X1、X11、X12
XCM series	24/32 I/O	X3

11-3 Precise Time

1: Instruction List

Read and stop precise time when execute precise time;

precise time [STR]			
16 bits instruction	-	32 bits instruction	STR
execution condition	edge activation	suitable models	XC1、XC2、XC3、XC5、XCM
hardware requirement	-	software requirements	-
read precise time [STRR]			
16 bits instruction	-	32 bits instruction	STRR
execution condition	edge activation	suitable models	XC1、XC2、XC3、XC5、XCM
hardware requirement	V3.0e and above	software requirements	-
stop precise time [STRS]			
16 bits instruction	-	32 bits instruction	STRS
execution condition	edge activation	suitable models	XC1、XC2、XC3、XC5、XCM
hardware requirement	V3.0e and above	software requirements	-

2: Operands

Operands	Function	Type
D	Timer’s Number	bit
D1	Timer’s Number	bit
D2	specify timer’s value or soft component’s ID number	16 bits, BIN

3: Suitable Soft Components

Word

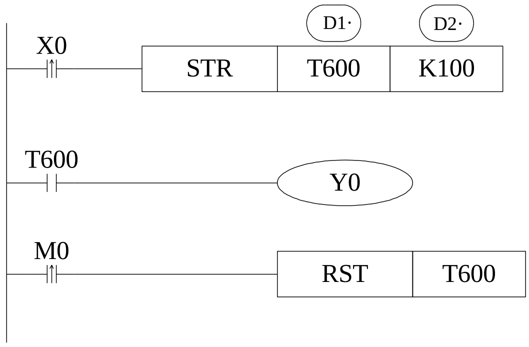
operands	system									constant	module	
	D	FD	ED	TD	CD	DX	DY	DM	DS	K/H	ID	QD
D2	●	●		●	●					●		

Bit

operands	system						
	X	Y	M	S	T	C	Dn.m
D					●		
D1					●		

Function and Actions

《Precise Time》

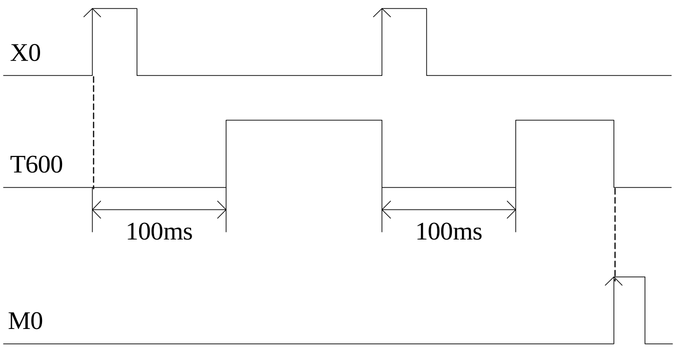


D1: Timer’s number. Range: T600~T618 (T600、T602、T604…T618, the number should be even)

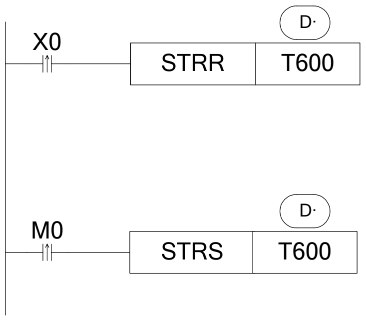
D2: Time Value

- The precise timer works in form of 1ms
- The precise timer is 32 bits, the count range is 0~+2,147,483,647.
- When X000 turns from OFF to ON, timer T600 starts to time, when time accumulation reaches 100ms, set T600; if X000 again turns from OFF to ON, timer T600 turns from ON to OFF，restart to time, when time accumulation reaches 100ms, T600 again reset. See graph below:
- When run STR instruction, reset the timer, then start to time;

See time graph below:



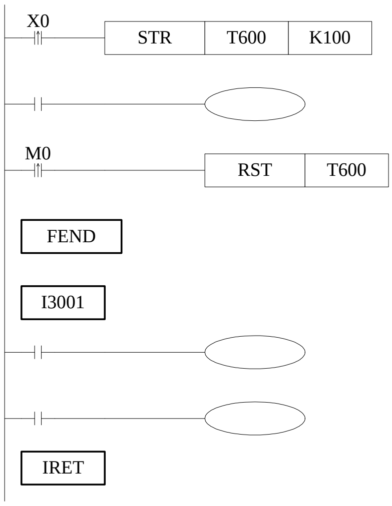
《read the precise time》、《stop precise time》



- When X000 changes from OFF to ON, move the current precise time value into TD600 immediately, regardless of the scan cycle;
- When M000 changes from OFF to ON, execute STRS instruction immediately, stop precise time and refresh the count value in TD600. Regardless of the scan cycle;

- When the precise time reaches the count value, generate a corresponding interruption tag, execute some interruption subroutines.
- Start the precise time in precise time interruption;
- Every precise timer has its own interruption tag, see table below:

Precious Time Interruption



When X000 changes from OFF to be ON, timer T600 starts to time. When time accumulates to 100ms, set T600; meantime, generate an interruption, the program jumps to interruption tag I3001 and execute the subroutine.

Interruption Tag correspond with the Timer

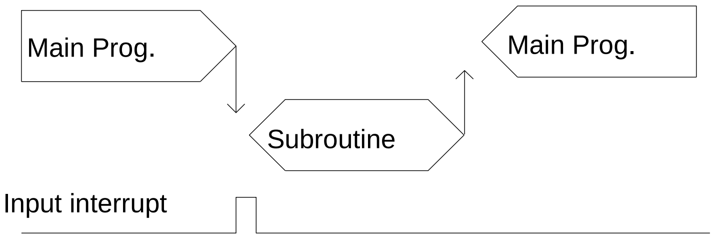
Timer’s Nr.	Interruption Tag
T600	I3001
T602	I3002
T604	I3003
T606	I3004
T608	I3005
T610	I3006
T612	I3007
T614	I3008
T616	I3009
T618	I3010

11-4 Interruption

XC Series PLCs are equipped with an interruption function. The interruption function includes external interruption and time interruption. With the interruption function we can utilize some special programs. This function is not effected by the scan cycle.

11-4-1 External Interruption

The input terminals X can be used to input external interruption. Each input terminal corresponds with one external interruption. The input's rising/falling edge can activate the interruption. The interruption subroutine is written behind the main program (behind FEND). After interruption generates, the main program stops running immediately, turn to run the correspond subroutine. After subroutine running ends, continue to execute the main program.



External Interruption's Port Definition

XC3-14

Input Terminal	Pointer Nr.		Disable the interruption instruction
	Rising Interruption	Falling Interruption	
X7	I0000	I0001	M8050

XC2 series、XC3-24/32、XC5-48/60

Input Terminal	Pointer Nr.		Disable the interruption instruction
	Rising Interruption	Falling Interruption	
X2	I0000	I0001	M8050
X5	I0100	I0101	M8051
X10	I0200	I0201	M8052

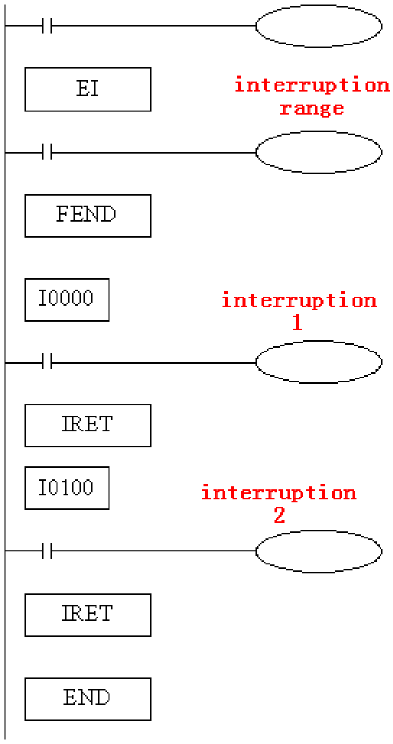
XC3-48/60、XC3-19AR-E

Input Terminal	Pointer Nr.		Disable the interruption instruction
	Rising Interruption	Falling Interruption	
X10	I0000	I0001	M8050
X7	I0100	I0101	M8051
X6	I0200	I0201	M8052

XC5-24/32、XCM-24/32-

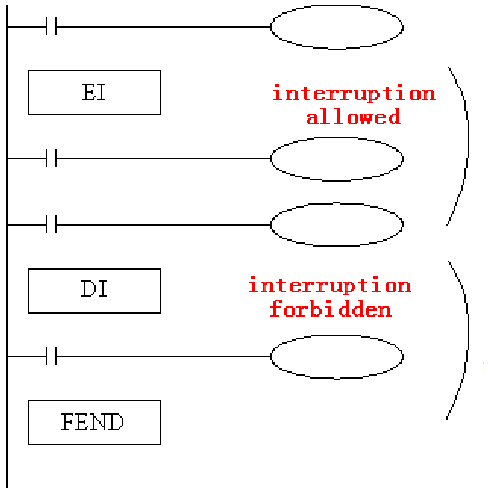
Input Terminal	Pointer Nr.		Disable the interruption instruction
	Rising Interruption	Falling Interruption	
X2	I0000	I0001	M8050
X5	I0100	I0101	M8051
X10	I0200	I0201	M8052
X11	I0300	I0301	M8053
X12	I0400	I0401	M8054

Enable Interruption [EI]、Disable Interruption [DI]、Interruption Return [IRET]



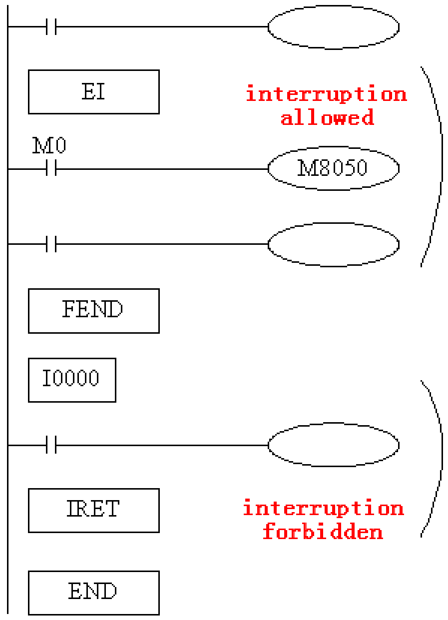
- If use EI instruction to allow interruption, then when scanning the program, if interruption input changes from OFF to be ON, then execute subroutine①、②, return to the original main program;
- Interruption pointer (I****) should be behind FEND instruction;
- PLC is default to allow interruption

Interruption's Range Limitation



- Via program with DI instruction, set interruption forbidden area;
- Allow interruption input between EI~DI
- If interruption forbidden is not required, please program only with EI, program with DI is not required.

Disable the Interruption

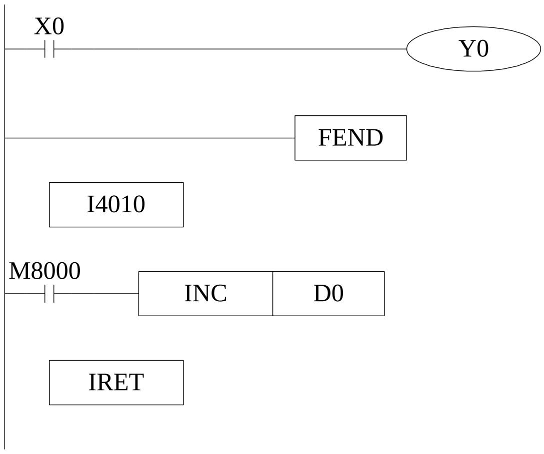


- Every input interruption is equipped with special relay (M8050~M8052) to disable interruption;
- In the left program, if use M0 to set M8050 “ON”, then disable the interruption input at channel 0.

11-4-2 Time Interruption

Functions and Actions

Within the main program's execution cycle, if you need to handle a special program; or during the sequential scanning, a special program needs to be executed at a certain time, time interruption function is required. This function is not affected by PLC's scan cycle, every Nm, executes a time interruption subroutine.



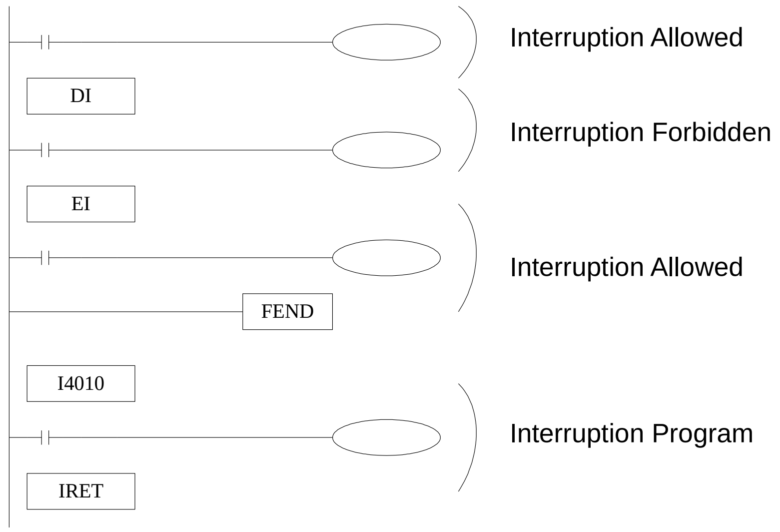
- Time interruption is defaulted in open status, time interruption subroutine is similar with other interruption subroutine, it should be written behind the main program, starts with I40xx, ends with IRET.
- There are 10CH time interruptions. The represent method is I40**~I49** ("**" means time interruption's time, unit is ms. For example, I4010 means run one channel time interruption every 10ms.

Interruption Number

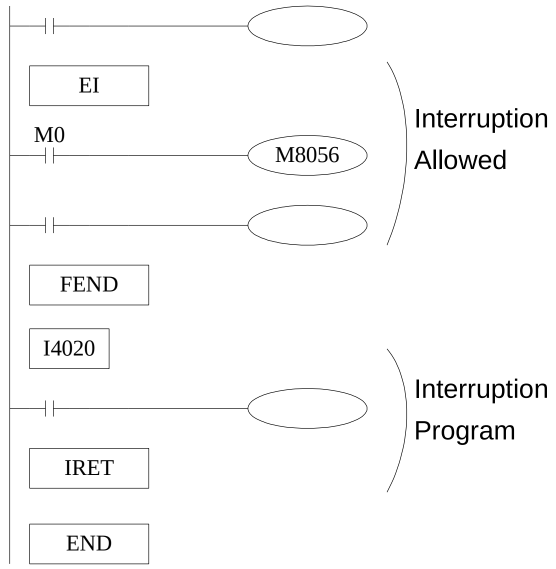
Interruption Nr.	Interruption Forbidden Instruction	Description
I40**	M8056	"**" represents time interruption's time, range from 1 to 99, unit is ms.
I41**	M8057	
I42**	M8058	
I43**	-	
I44**	-	
I45**	-	
I46**	-	
I47**	-	
I48**	-	
I49**	-	

Interruption Range's Limitation

- Normally time interruption is in "allow" status
- With EI、DI can set interruption's allow or forbidden area. As in the above graph, all time interruptions are forbidden between DI~EI, and allowed beyond DI~EI.



Interruption Forbidden



- The first 3CH interruptions are equipped with special relays (M8056~M8059) to forbid interrupt
- In the left example program, if use M0 to enable M8056 “ON”, the forbid 0CH’s time interruption.

12 Program Application Samples

In this chapter, we make some samples about pulse output instruction, Modbus communication instructions and free format communication instructions etc.

12-1 Pulse Output Application

Example: below is the example program to send high/low pulse in turn

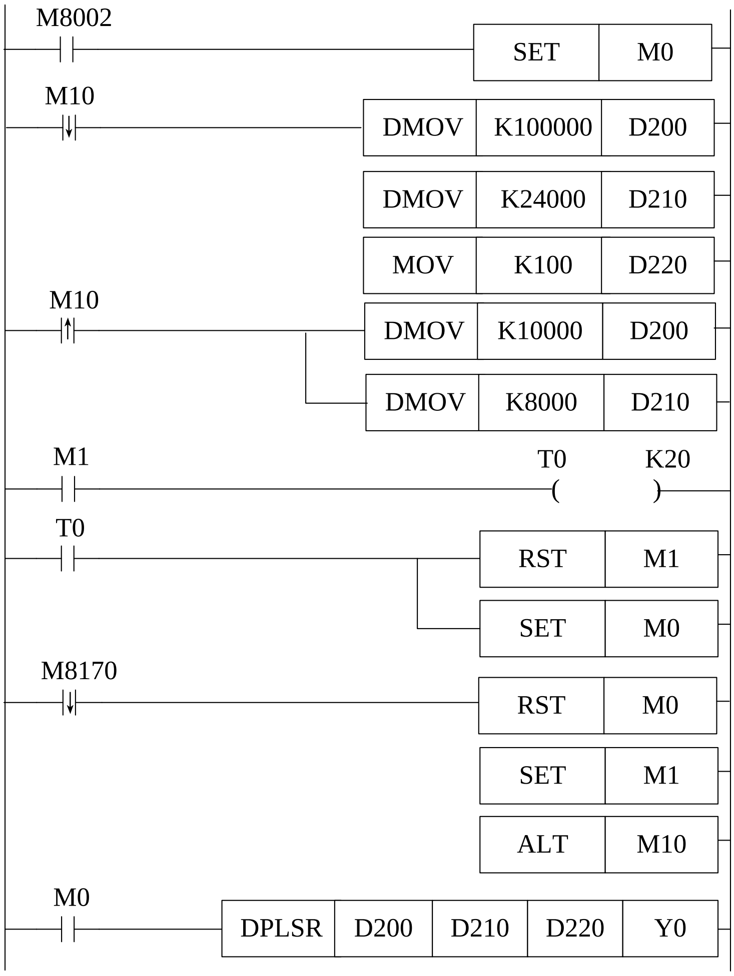
Each Parameter:

Stepping motor parameters: step angle= 1.8 degrees/step, scale=40, pulse number per rotate is 8000

High frequency pulse: maximum frequency is 100KHz, total pulse number is 24000 (3 rotates)

Low frequency pulse: maximum frequency is 10KHz, total pulse number is 8000 (1 rotates)

Ladder Program:



Instruction List:

LD	M8002		//initial positive pulse coil
SET	M0		//set M0 ON
LDF	M10		//M10 falling edge activate condition
OR	M8002		//Initial data
DMOV	K100000	D200	//move decimal data 100000 into DWORD D200
DMOV	K24000	D210	// move decimal data 24000 into DWORD D210
MOV	K100	D220	// move decimal data 100 into DWORD D220

```
LDP     M10                          //M10 rising edge activate condition
DMOV    K10000  D200                 // move decimal data 10000 into DWORD D200
DMOV    K8000   D210                 // move decimal data 8000 into DWORD D210
LD      M1                           //M1 status activate condition
OUT     T0  K20                      //100ms timer T0, time 2 seconds
LD      T0                           //T0 status activate condition
RST     M1                           //reset M1
SET     M0                           //set M0
LDF     M8170                        //M8170 falling edge activate condition
RST     M0                           //reset M0
SET     M1                           //set M1
ALT     M10                          //M10 status NOT
LD      M0                           //M0 status activate condition
DPLSR   D200  D210  D220  Y0         //value in D200 is frequency、value in D210 is
        pulse number、value is D220 is acceleration/deceleration time, send pulse via Y0;
```

Explanation:

When PLC changes from STOP to be RUN, M8002 gets a scan cycle;
set the high frequency pulse parameters into D200、D210,
set the acceleration/deceleration speed to D220,
set M0, the motor starts to run 3 rounds with high frequency.
Meantime M8170 sets; the motor runs 3 rounds and decelerate, stop, coil M8170 reset;
then reset M0, set M1, **NOT** M10;
set the low frequency pulse parameters into D200、D210;
the timer time lags 2sec, when time reaches, reset M1;
set M0, the motors starts to run 1 round with low frequency;
after this starts to run with high frequency.
Repeat this alternation time by time;

12-2 Modbus Communication Application

E.g.1: realize Modbus read/write among one master and three slaves

Operation: (1) write content in D10~D14 to D10~D14 of 2# slave;

(2) read D15~D19 of the slaves to D15~D19 of the mater; anyhow, write the first five registers' content to the slaves, the left five registers are used to store the content from the slaves;

（3）3# 、4# slaves are similar;

Soft component's comments:

D0: communication station number

D1: offset

M2: 2# communication error

M3: 3# communication error

M4: 4# communication error

M8137: COM2 communication error end signal

M8138: COM2 communication correct end signal

S0: write the target station

S1: read the target station

S2: judge the communication status

S3: offset the communication ID

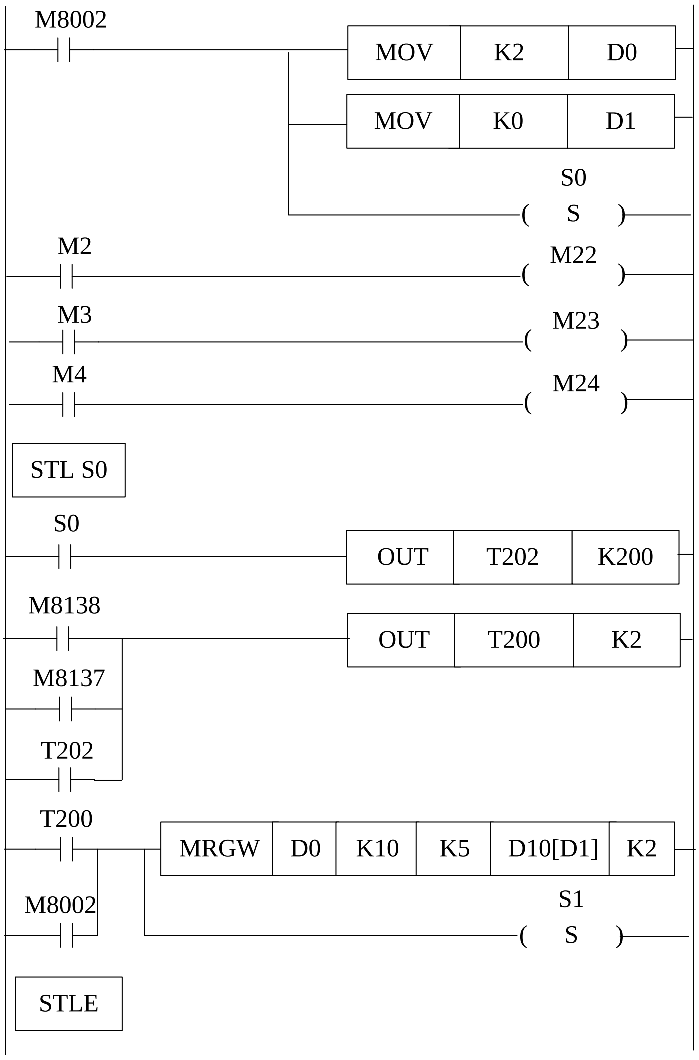
T200: communication interval 1

T201: communication interval 2

T202: self reset 1 of communication error

T203: self reset 2 of communication error

Ladder



In PLC's first scan cycle, evaluate the "communication station" to be 2;
Evaluate the "offset" to be 0

2# communication error reset

3# communication error reset

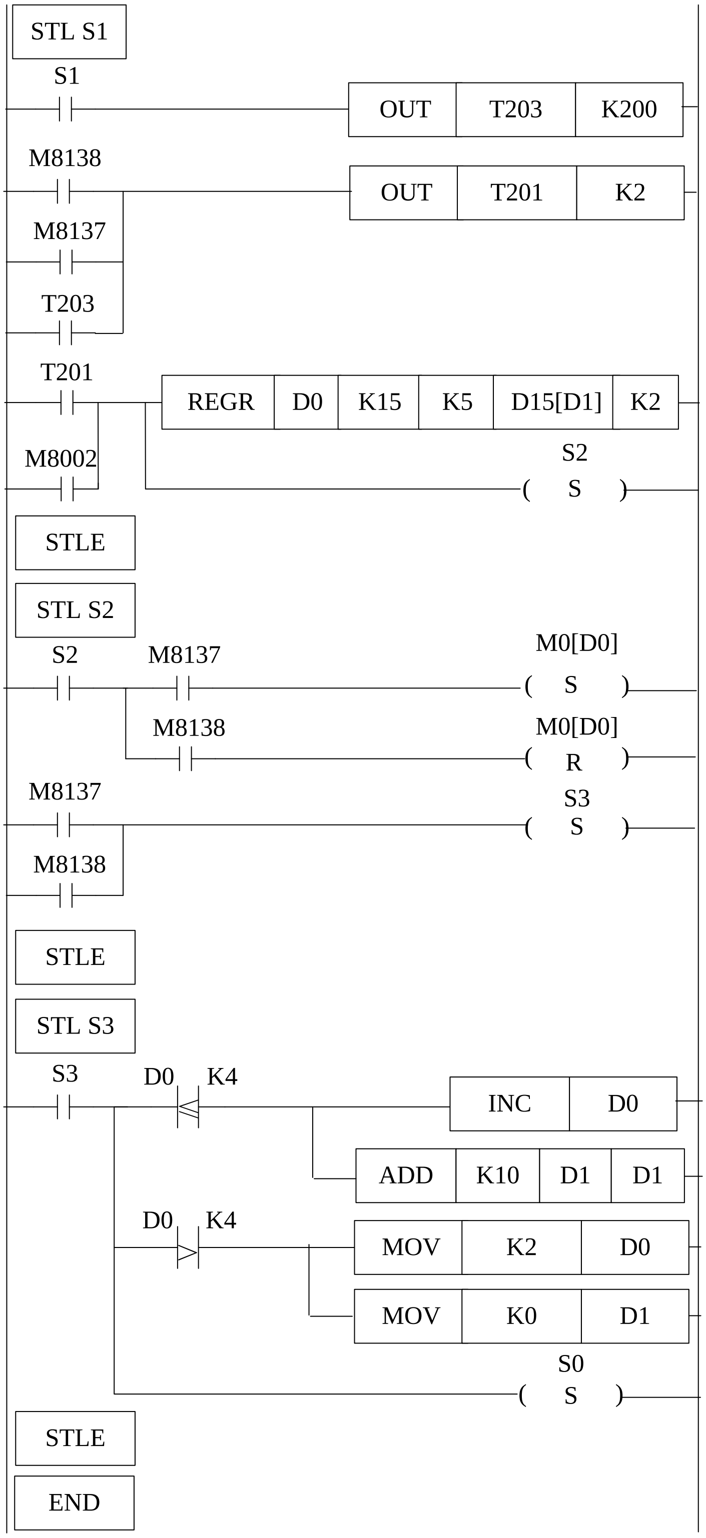
4# communication error reset

S0 starts, T202 counts 2S, which is the communication wait time

When the communication wait time reaches, no matter the communication succeeds or not, T200 time 20ms, this time is used start the next

T200 time reaches, or on the power up, execute the RUN operation to the target station

Open the flow S1



S0 starts, T203 time 2s, which is the communication waiting time

When communication waiting time reaches, no matter the communication succeeded or not, T201 counts 20ms, this time is used to start the next

T201 times reach, or on the power up, execute the read operation with the target stations

Open flow S2

Flow S2 is used to judge the communication status. Failure will set the correspond coil; success will reset the correspond coil;

If the station number is not larger than 4, the station register add 1, the offset add 10

If the station number is larger than 4, evaluate the station register 1; clear the offset register

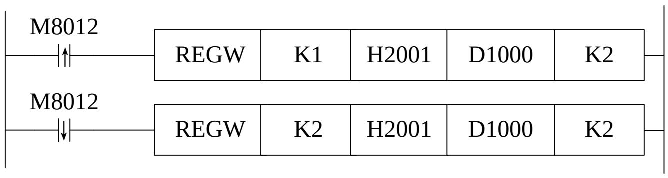
Open flow S0

Program Explanation:

When PLC turns from STOP to RUN, M8002 gets a scan cycle. S0 flow open, write the master's D10——D14 to slave 2# D10——D14. no matter the communication is success or not, turn to S1 flow; check the previous communication written condition. After certain time delay, continue to read D15~D19 data from 2#. After this reading entr S2 flow, check if the communication is success. If failed, set M23, enter alarming. After finishing the communication with 2#, enter S3, then flow S3 will judge with the station number. If the station number is less than 1, the offset add 10; or else start from 2# again.

e.g. 2: Below is a sample of XC Series PLC with two XINJE inverters, they communicate via Modbus communication, XC Series PLCs write the frequency to the two inverters;

set the first inverter's station to be 1; set the second inverter's station to be 2; store the frequency's set value in D1000 and D2000. execute the frequency setting order via COM ports;



Program Description:

On the rising edge of M8012, write frequency to the first inverter; on the falling edge of M8012, write frequency to the second inverter;

12-3 Free Format Communication Application

In this example, we use DH107/DH108 series instruments;

1、Interface Specifications

DH107/DH108 series instruments use asynchronous serial communication interface, the interface level fits RS232C or RS485 standard. The data format is: 1 start bit, 8 data bits, no parity, one/two stop bit. The baud rate can be 1200~19200bit/s.

2、Communication Instruction Format

DH107/108 instruments use Hex data form to represent each instruction code and data;

Read/write instructions:

Read: address code +52H (82) +the para.(to read) code +0+0+CRC parity code

Write: address code +43H（67）+ the para.(to write) code +low bytes of the wrote data + high bytes of the wrote data +CRC parity code

The read instruction's CRC parity code is: the para. (to read) code *256+82+ADDR

ADDR is instrument's address para., the range is 0~100 (pay attention not to add 80H). CRC is the remainder from the addition of the above data (binary 16bits integral). The reminder is 2 bytes, the high byte is behind the low byte;

The write instruction's CRC parity code is: the para. (to write) code *256+67+ the para. value (to write) +ADDR

The para. to write represents with 16 bits binary integral;

Regardless of whether it is write or read, the instrument should return data as shown below:

The test value PV+ given value SV+ output value MV and alarm status +read/write parameters value +CRC parity code

Among in, PV、SV and the read parameters are all in integral form, each occupies two bytes, MV occupies one byte, the value range is 0~220, alarm status occupies one byte, CRC parity code occupies two bytes, totally 10 byes.

CRC parity code is the reminder from the result of PV+SV+ (alarm status *256+MV)+ para. value +ADDR;

(for details, please refer to AIBUS communication description)

3、Write the program

After power on the PLC, the PLC read the current temperature every 40ms. During this period, the user can write the set temperature.

Data zone definition: buffer area of sending data D10~D19

buffer area of accepting data D20~D29

instruction's station number: D30

read command's value: D31=52 H

write command's value: D32=43 H

parameter's code: D33

temperature setting: D34

CRC parity code: D36

Temperature display: D200,D201

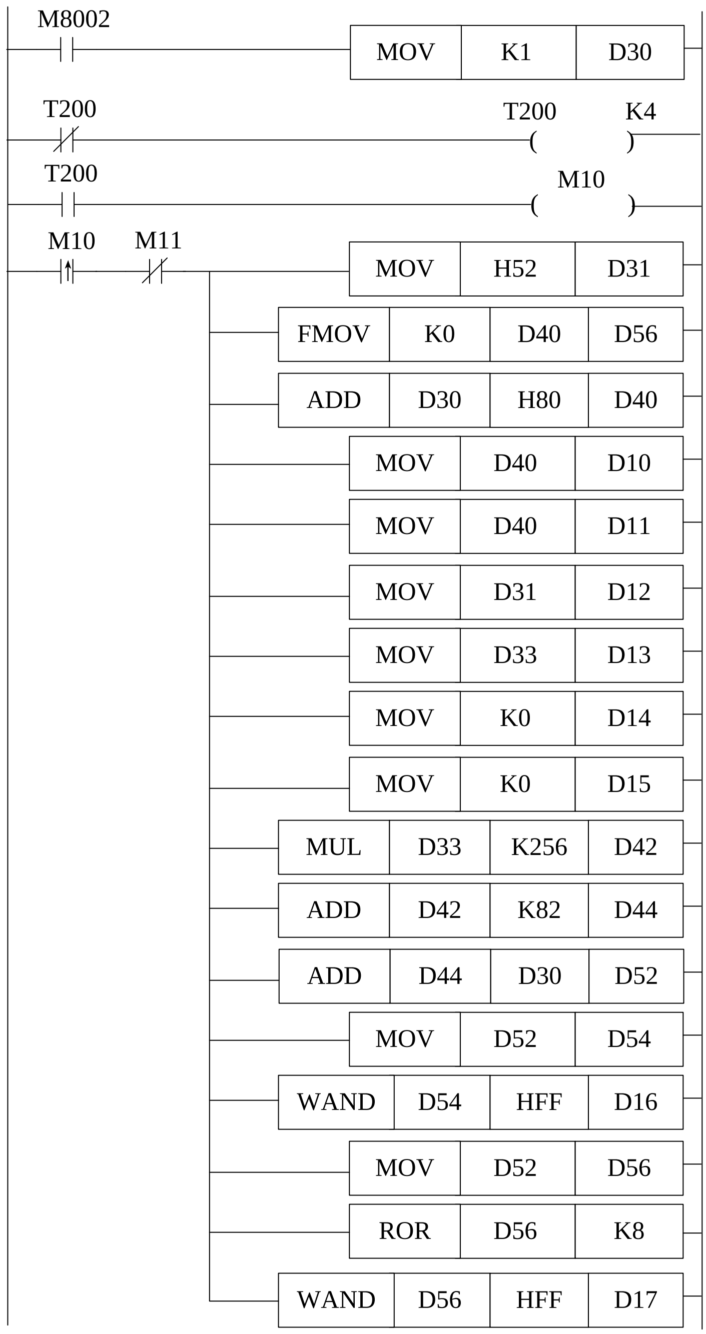
The send data form: 81H 81H 43H 00H c8H 00H 0cH 01H (current temperature display)

Communication parameters setting: baud rate: 9600, 8 data bits, 2 stop bits, no parity

Set FD8220=255; FD8221=5

(the hardware and software must be V2.4 or above)

Ladder:



Write instrument's station Nr. K1 in to D30

Time 40ms

Output M10

Write the read code 52H into D31

Clear registers D40-D56

D30 add H80 to get value 81H

move D40 (81H) to D10

move D40 (81H) to D11

move D31 (read code 52H) to D12

move D33 (para. code) to D13

write zero to D14

write zero to D15

below is to calculate CRC parity;

D33 multiply K256, the result is saved in D42

D42 add K82, the result is stored in D44

D44 add D30 (instrument's station), the result is saved in D52

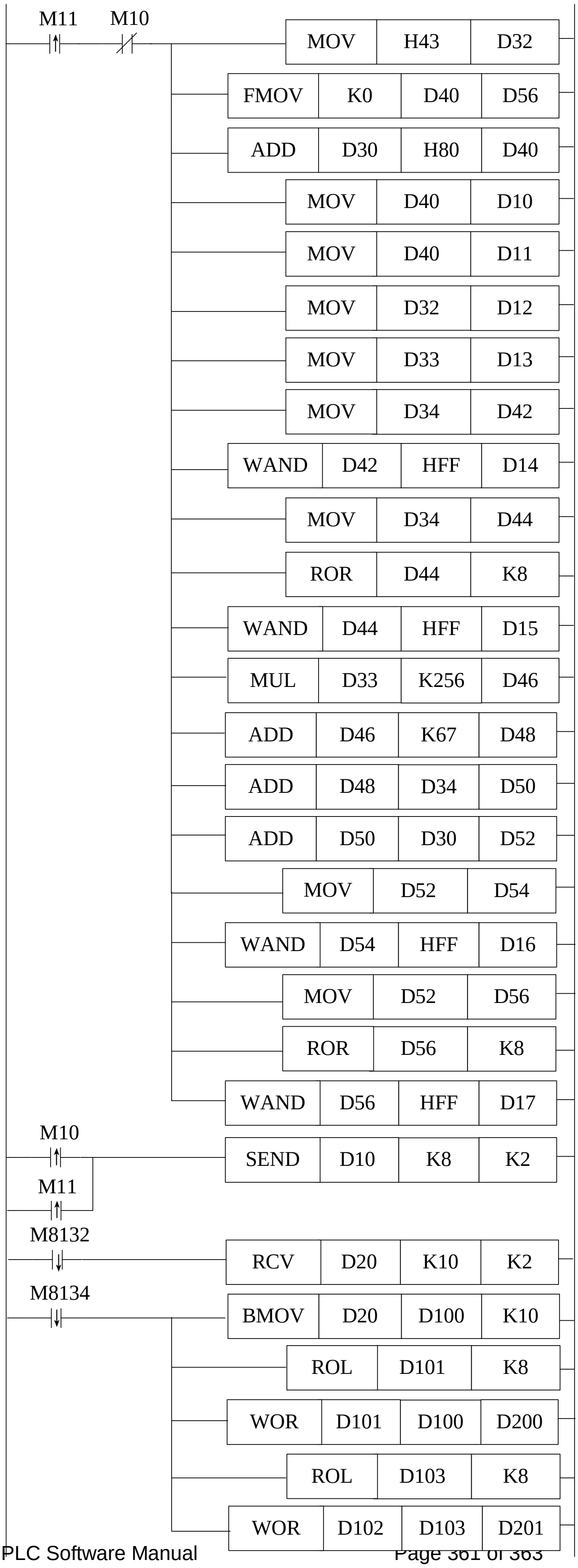
Move D52 into D54

Logic AND D54 with HFF, save the result in D16

Move D52 into D56

Right shift 8 bits with D56 (convert the high 8bits to the low 8 bits)

Logic AND D56 with HFF, save the result in D17



Write code H43 into D32

Clear registers D40-D56

D30 (station Nr.) add H80, save the result in D40

Move D40 to D10

Move D40 to D11

Move D32 (write code H43) to D12

Move D33 (para .code) to D13

Move D34 (temp. set) to D42

Logic and D42 with HFF, save data in D14

Move D34 (temp. set) to D44

D44 right shift 8 bits

Logic and D44 with HFF, save data in D15

Below is to calculate CRC parity:

D33 (para. code) multiply K256, save result in D46

D46 add K67, save data in D48

D48 add D34, save data in D50

D50 add D30, save data in D52

Move D52 to D54

Logic and D54 with HFF, save result in D16

Move D52 to D56

Right shift 8 bits with D56

Logic and D56 with HFF, save result in D17

Send data D10-D17 out

Read the returned data and save in D20-D29

Move the returned data to D100~109

Left shift 8 bits with D101

Logic OR D101 with D100, save result in D200

Left shift 8 bits with D103

Logic OR D102 with D103, save result in D201

Program Description:

The above program is written according to DH instrument's communication protocol, the soft component's functions are listed below:

Relationship of sent (SEND) data string and registers:

	D10	D11	D12	D13	D14	D15	D16	D17
Read	Address code	Address code	Read code 52H	Parameters code	0	0	CRC low bytes	CRC high bytes
Write	Address code	Address code	Write code 42H	Parameters code	low bytes of the written data	high bytes of the written data	CRC low bytes	CRC high bytes

Relationship of received (RCV) data (data returned by the instrument) and the registers:

D20	D21	D22	D23	D24	D25	D26	D27	D28	D29
PV low bytes	PV high bytes	SV low bytes	SV high bytes	Output value	Alarm status	Read/write low bytes	Read/write high bytes	CRC low bytes	CRC high bytes

When writing a data string according to the communication objects' protocol, use SEND and RCV commands from free format communication, user will get the communication with the objects.

Documentation Reference				
Document Number				Revision Date
MANU	L021	R2	V1	04/07/2011
XINJE IS A REGISTERED TRADEMARK OF XINJE ELECTRICAL CO.LTD. REPLICATION OF THE INFORMATION CONTAINED WITHIN THIS DOCUMENT WITHOUT PRIOR NOTIFICATION AND AGREEMENT IS PROHIBITED.				

Engineered and supplied by:

LISTO-LTD

International partners with:

XINJE

ECON Solutions Group
"Simple Solutions to Difficult Problems"

SUPPORT XINJE CENTRE

Web: www.listo-ltd.com
www.xinje-support-centre-listo.com

E-mail: support@listo-ltd.com

Please consider the environment before printing this document