XD/XL series PLC
User manual [Instruction]

WUXI XINJE ELECTRIC CO., LTD.

Data No. PD05 20231220EN 1.3

XINJE

XD/XL series PLC
User manual［Instruction］

- Basic explanation

Thank you for purchasing Xinje XD/XL series PLC.

This manual mainly introduces XD/XL series PLC instructions.

Please read this manual carefully before using and wire after understanding the content.

About software and programming instructions, please refer to related manuals.

Please hand this manual over to operation users.

- Notices for users

Only experienced operator can wire the plc. If any problem, please contact our technical department.

The listed examples are used to help users to understand, so it may not act.

Please conform that PLC specifications and principles are suitable when connect PLC to other products. Please conform safety of PLC and machines by yourself when use the PLC. Machines may be damaged by PLC errors.

- Responsibility declaration

The manual content has been checked carefully, however, mistakes may happen.

We often check the manual and will correct the problems in subsequent version. Welcome to offer advices to us.

Excuse us that we will not inform you if manual is changed.

- Contact information

If you have any problem about products, please contact the agent or Xinje company.

Tel: 0086 510-85134136 85123803

Fax: 0086 510-85111290

Address: Building 7 fourth floor, No.100, Dicui Rd, Wuxi, China.

Code : 214072

Catalog

1 Programming Summary

XD/XL series PLC accept the signal and execute the program in the controller, to fulfill the requirements of the users. This chapter introduces the PLC features, two kinds of programming language and etc.

1-1 PLC Features

Programming Language

XD/XL series PLC support two kinds of program language, instruction and ladder chart, the two kinds of language can convert to each other.

Security of the Program

To avoid the stolen or wrong modifying of user program, we encrypt the program. When uploading the encrypted program, it will check in the form of password. This can protect the user copyright; meanwhile, it limits the downloading, to avoid change program by mistake.
XD/XL series added new register FS. (For different XD/XL models, please check the Data monitor in XDPpro software for FS register range, common range is FS0~FS47). FS value can be modified but cannot be read through Modbus instruction. FS cannot be compared to register but only constant in XDPpro software. The value cannot be read. FS is used to protect the user's copyright. The register D, HD… can replace by FS.

Program comments

When the user program is too long, the comments of program and soft components are necessary in order to change the program easily later.

Offset Function

Add offset appendix (like X3[D100], M10[D100], D0[D100]) after coils, data registers can make indirect addressing. For example, when D100=9, X3[D100]=X[3+9]=X14; M10[D100]=M19, D0[D100]=D9

Rich Basic Functions

XD/XL series PLC has enough basic instructions including basic sequential control, data moving and comparing, arithmetic operation, logic control, data loop and shift etc.
XD/XL series PLC also support interruption, high speed pulse, frequency testing, precise time, PID control and so on.

C Language Function Block

XD/XL series PLC support C language; users can call the C program in ladder chart. This function improves the programming efficiency.

Stop PLC whenreboot

XD/XL series PLC support “Stop PLC when reboot” function. When there is a serious problem during PLC running, this method can stop all output immediately. Besides, if the COM port parameters are changed by mistake, this function can help PLC connect to the PC.

Communication Function

XD/XL series PLC has many communication modes, such as Modbus-RTU, Modbus-ASCII. When the COM port parameters are changed, the new parameters will be valid immediately without restarting the PLC.
Wait time can be added before Modbus instructions.

1-2 Programming Language

1-2-1 Type

XD/XL series PLC support two types of programming language:

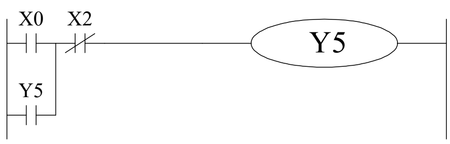
Instruction

Make the program with instructions directly, such as “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	operand
0	LD	X000
1	OR	Y005
2	ANI	X002
3	OUT	Y005

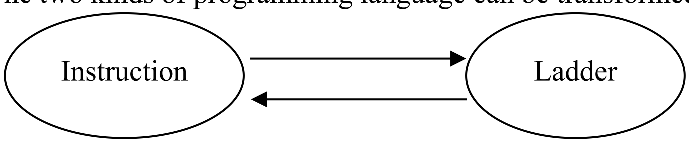
Ladder chart

Make sequential control graph with sequential control signal and soft components. This method is called “Ladder chart”. This method uses coils and contactors to represent sequential circuit. The ladder chart is easy to understand and can be used to monitor the PLCstatus online.
E.g.:



1-2-2 Alternation

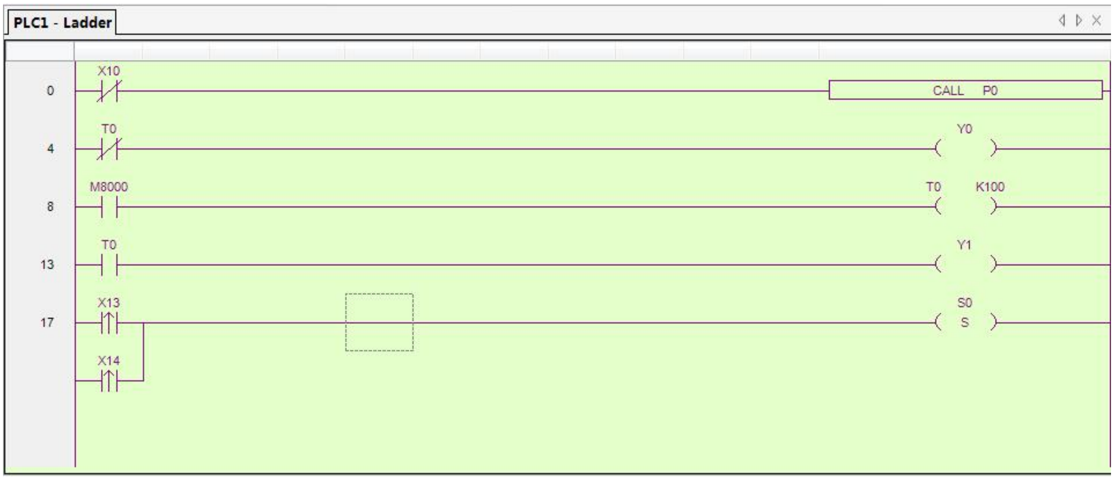
The two kinds of programming language can be transformed to each other.



1-3 Programming mode

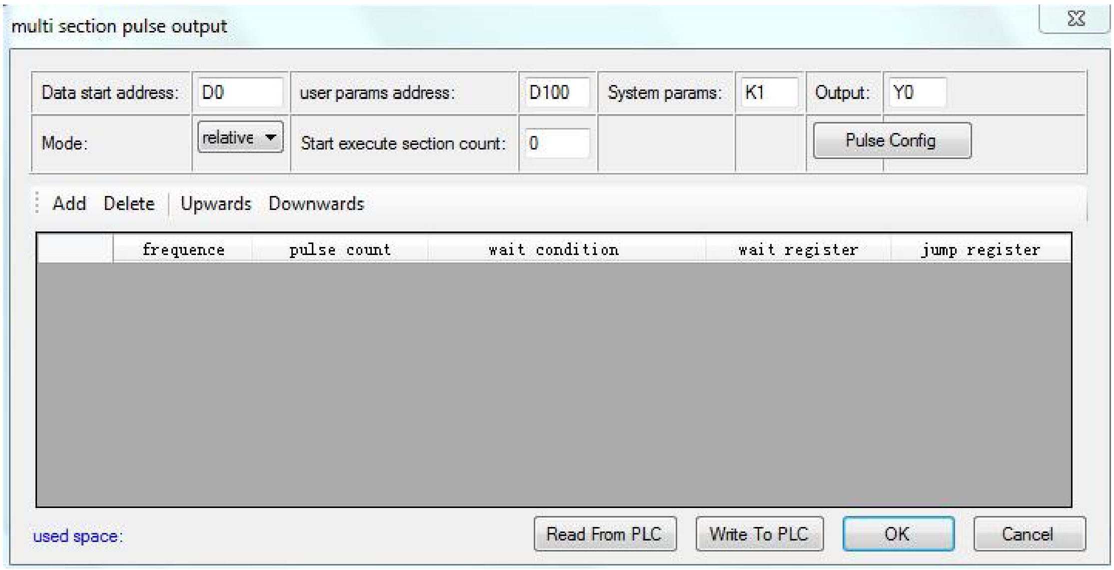
Direct Input

The two kinds of programming language can be input directly in the editing window. The ladder chart window has hint function which improves the programming efficiency greatly.



Instruction Configuration

Some instruction is complicated to use, like pulse output, PID etc. XDPPro software has the configuration window for these special instructions. User just needs to input parameters in the configuration window without remembering complicated instructions. The following window is multi section pulse output.



For the details of instruction configuration, please refer to XD/XL series PLC user manual 【software part】.

2 Soft Component Function

In chapter 1, we briefly introduce the programming language. However, the most important element in a program is the operands. These elements include the relays and registers. In this chapter, we will describe the functions and using methods of these relays and registers.

2-1 Summary of the Soft Components

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

Input Relay (X)

- The functions of input relays

The input relays are used to receive the external ON/OFF signal, the sign is **X**.

- Address AssignmentPrinciple

➢ In each basic unit, X address is in the form of octal, such as X0~X7, X10~X17 …

➢ The extension module address:module 1 starts from X10000, module 2 starts from X10100…XD1/XD2/XL1 cannot support extension module. Up to 10extension modules can be connected to the XD3/XL3main unit. XD5/XDM/XDC/XD5E/XDME/XDH/XL5/XL5E/XLME can connect 16 extension modules.

➢ Extension BD board: BD 1 starts from X20000;The 24-32 points PLC can connect one extended BD board and the 48-60 points PLC can connect two extended BD boards. (16-point PLC does not support extended BD board, XL/XDH series does not support extended BD board.)

➢ The address number of the left extended ED module, starting from X30000 according to octal system, XD/XL series PLC supports a left extended I/O ED module. (XDH cannot support ED module)

- Using notes

The digital filter is used in the input filter of the input relay. Users can change the filter parameters by setting the special register SFD0, default value is 10ms, modification range: 0 ~ 1000ms.

There are enough input relays in the PLC. The input relay whose address is more than input points can be seemed to auxiliary relay.

Output Relay (Y)

- Function of the output relays

Output relays are the interface to drive the external loads, the sign is **Y**;

- Address Assignment Principle

In each basic unit, Y address is in the form of octal, such as Y0~Y7, Y10~Y17 …

The extension module address: module 1 starts from Y10000, module 2 starts from Y10100…

XD1/XD2/XL1 does not support extension modules, XD3/XL3 can accept 10 extension modules, XD5/XDM/XDC/XD5E/XDME/XDH/XL5/XL5E/XLME can accept 16 extension modules.
Expanding the address number of BD board, starting from X20000 according to octal system, 24-32 points PLC can extend one BD board, 48-60 points PLC can extend two BD boards. (16-point PLC does not support extended BD board, XL/XDH series does not support extended BD board.)
The address number of the left extended ED module, starting from Y30000 according to octal system, XD/XL series PLC supports a left extended input and output ED module. (XDH cannot support ED module)

Using notes
There are enough output relays in the PLC. The output relay whose address is more than output points can be seemed to auxiliary relay.

Auxiliary Relays (M, HM)

- Function of Auxiliary Relays

Auxiliary relays is internal relays of PLC, the sign is M and HM;

- Address assignment principle

In basic units, assign the auxiliary address in decimal form

- Using notes

This type of relays are different from the input/output relays, they can't drive external load and receive external signal, but only be used in the program;
Retentive relays can keep its ON/OFF status when PLC power OFF;

Status Relays (S, HS)

- Function of status relays

Used as relays in Ladder, the sign is S, HS.

- Address assignment principle

In basic units, assign the address in decimalform.

- Using notes

If it is not used as operation number, they can be used as auxiliary relays, programming as normal contactors/coils. Besides, they can be used as signal alarms, for external diagnose.

Timer (T, HT)

- Function of the timers

Timers are used to accumulate the time pulse like 1ms, 10ms, 100ms etc. when reach the set value, the output contactors acts, represent sign is T and HT.

- Address assignment principle

In basic units, assign the timer address in decimal form. Please refer to chapter 2-2 for details.

- Time pulse

There are three timer pulses: 1ms, 10ms, and 100ms. For example, 10ms means accumulate 10ms pulses.

- Accumulation/not accumulation

The timer has two modes: accumulation timer means even the timer drive coil is OFF, the timer will still keep the current value; while the not accumulation timer means when the accumulation value reaches the set value, the output acts, the accumulation value reset to 0.

Counter (C, HC)

According to different application purposes, the counters contain different types:

- For internal counting (for general using/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 are for PLC internal signal. The response speed is one scan cycle or longer.

- For High Speed Counting (Power-off retentive)

32 bits counter: the count range is -2,147,483,648~ +2,147,483,647
(Single phase increment count, AB phase count). For special input terminals.
The high speed counterwill not be affected by PLC scanning period. For increment mode, it can count max 80KHz pulses; for AB phase mode, it can count max 50KHz pulses.

- Address assignment principle

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

Data Register (D, HD)

- Function of Data Registers

Data Registers are used to store data, the sign is D and HD.

- Address assignment principle

The data registers in XD/XL series PLC are16 bits (the highest bit is sign bit), combine two data registers together is for32 bits (the highest bit is sign bit) data processing.

- Using notes

Same to other soft components, data registers also have common type and power-off retentive type.

FlashROM Register (FD)

- Function of FlashROM registers

FlashROM registers are used to store data, the sign is FD.

- Address assignment principle

In basic units, FlashROM registers address is in form of decimal;

- Using notes

Even the battery powered off, this area can remember the data. So this area can store important parameters. FlashROM can be writen for about 1,000,000 times, and it takes timewhen writing. Frequently writing can cause permanent damage for FD.

Special secret Register (FS)

- The Function of Secret Register

A part of the FlashROM register is used to store data in soft components, which are represented by the symbol FS. The values in the FS register can be written but can not be read, so they can be used to protect the intellectual property rights of users.

- Address Allocation Principle

In the basic unit, FS registers are addressed in decimal numbers.

- Since the number of FS registers of different types of PLC may be different, please refer to the "PLC Initial Settings" shown in the online PLC software, generally FS0-FS47.
- Attention Points in Use

The storage area can remember data even if the battery is powered down, so it can be used to store important process parameters. FS can be written about 1,000,000 times, and it takes more time to write each time. Frequent writing will cause permanent damage to FS, so it is not recommended that users write frequently. When using MOV instruction to transmit data to FS, the rising edge is valid.

- The value of the soft element can be set arbitrarily in the FS register, but the value of the register can not be read (always returned to 0); and it can not be compared with the register in the PLC software, only with the constant, so the actual value of the register can not be read.

Constant (B) (K) (H)

B means Binary, K represents Decimal, H represents Hexadecimal. They are used to set timers and counters value, or operands of application instructions. For example hex FF will be HFF.

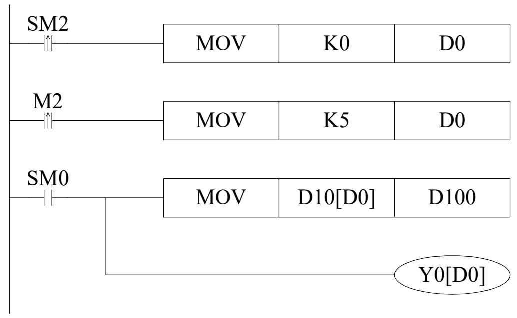
2-2 Structure of Soft Components

2-2-1 Structure of Memory

In XD/XL series PLC, there are many registers. Besides D, HD, FlashROM registers, we can also combine bit to register.

Data Register D, HD,FD

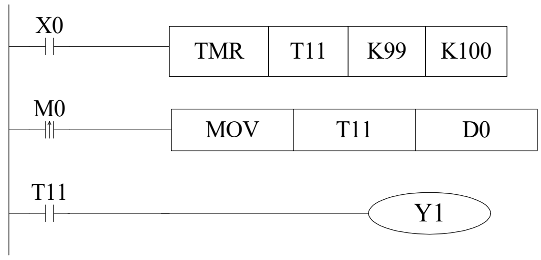
For common use,16 bits
For common use,32 bits (combine two continuous 16-bits registers)
For common use,64 bits (combine two 32-bit registers, but addresses must be consecutive).
For power off retentive use, cannot modify the retentive range
For special use, occupied by the system, can’t be used to common instruction parameters
For offset use (indirect assignment)
Form: Dn[Dm], HDn[Dm], Xn[Dm] , Yn[Dm] , Mn[Dm], etc.



When D0=0, D100=D10, Y0 is ON.
When M2 turns from OFF to 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: DXn[Dm] represents DX[n+Dm].
The soft components with offset, the offset can represent by soft component D, HD.

Timer T, HT/Counter C, HC

For common usage,16 bits, represent the current value of timer/counter;
For common usage,32 bits, (combine two continuous16 bits registers)
To represent them, just use the letter+address method, such as T10, C11, HT10, HC11.
E.g.



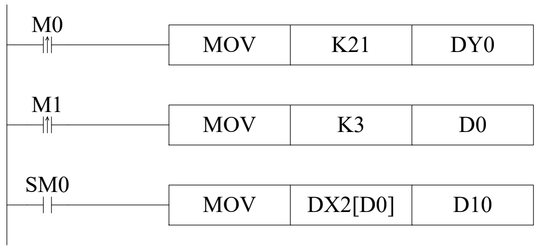
In the above example, MOV T11 D0, T11 represents word register;
LD T11, T11 represents bit register.

FlashROM Register FD

For power off retentive usage,16 bits
For power off retentive usage,32 bits, (combine two continuous16 bits registers)
For special usage, occupied by the system, can’t be used as common instruction parameters

Register combined by bits

For common usage,16 bits, (combine16 bits)
The soft components which can be combined to words are: X, Y, M, S, T, C, HM, HS, HT, HC.
Format: add “D” in front of soft components, like DM10, represents a 16-bits register from M10~M25 .
Get16 bits beginning from DXn, cannot beyond the soft components range;
The word combined by bits cannot do bit addressing;
E.g.:



When M0 changes from OFF to ON, the value in the word which is combined by Y0~Y17 equals to 21, i.e. Y0, Y2, Y4 become ON.
Before M1 activates, if D0=0, DX2[D0] represents a word combined by X2~X21.
If M1 changes from OFF to ON, D0=3, then DX2[D0] represents a word combined by X5~X24.

2-2-2 Structure of Bit Soft Components

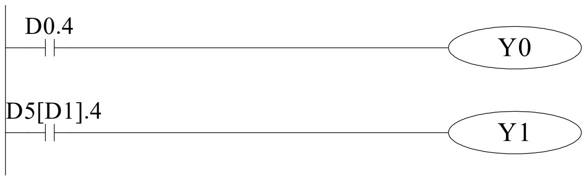
Bit soft components include X, Y, M, S, T, C, HM, HS, HT, HC. Besides, the bit of the register also can be used as bit sofst component.

Relay

Input Relay X, octal form
Output Relay Y, octal form
Auxiliary Relay M, HM, S, HS; decimal form
Auxiliary Relay T, HT, C, HC, decimal form. The represent method is same to registers, so we need to judge if it’s word register or bit register according to the instruction.

The bit of register

Composed by bit of register, support register D
Represent method: Dn.m (0≤m≤15): for example D10.2 means the second bit of D10
The represent method of bit with offset: Dn[Dm].x
Bit of register can’t compose to word soft component again;
E.g.:



D0.4 means when the fourth bit of D0 is 1, set Y0 ON.
D5[D1].4 means bit addressing with offset, if D1=5, then D5[D1] means the fourth bit of D10

2-3 Soft Components List

2-3-1 Soft Components List

XD1 series PLC soft components list:

	Name	Range				Points			
		10 I/O	16 I/O	24 I/O	32 I/O	10	16	24	32
X	Input points	X0~X4	X0~X7	X0~X13	X0~X17	5	8	12	16
Y	Output points	Y0~Y4	Y0~Y7	Y0~Y13	Y0~Y17	5	8	12	16
M	Internal relay	M0~M7999				8000			
HM		HM0~HM959※1				960			
SM		Special purpose SM0~SM2047※2				2048			
S	Flow	S0~S1023				1024			
HS		HS0~HS127※1				128			
T	Timer	T0~T575				576			
HT		HT0~HT95※1				96			
ET		Precise timer ET0~ET31				32			
C	Counter	C0~C575				576			
HC		HC0~HC95※1				96			
HSC		High speed counter HSC0~HSC31				32			
D	Data register	D0~D7999				8000			
HD		HD0~HD999※1				1000			
SD		Special purpose SD0~SD2047				2048			
HSD		Special purpose HSD0~HSD499※2				500			
FD	FlashROM register	FD0~FD5119				5120			
SFD		Special purpose SFD0~SFD1999※2				2000			
FS	Special secret register	FS0~FS47				48			
ID※6	Main body	ID0~ID99				100			
QD※7	Main body	QD0~QD99				100			
SEM	Special coil of Sequence block instruction WAIT	SEM0~SEM31				32			

XD2 series PLC soft components list:

	Name	Range						Points					
		16I/O	24 I/O	32 I/O	42I/O	48 I/O	60 I/O	16	24	32	42	48	60
X	Input points	X0~X7	X0~X15	X0~X21	X0~X27	X0~X33	X0~X43	8	14	18	24	28	36
Y	Output points	Y0~Y7	Y0~Y11	Y0~Y15	Y0~Y21	Y0~Y23	Y0~Y27	8	10	14	18	20	24
X	Input points※4	X20000~X20077(#1 expansionBD) X20100~X20177(#2 expansionBD)						128					
Y	Output points※4	Y20000~Y20077(#1 expansionBD) Y20100~Y20177(#2 expansionBD)						128					
X	Input points※5	X30000~X30077(#1 expansionED)						64					
Y	Output points※5	Y30000~Y30077(#1 expansionED)						64					
M	Internal relay	M0~M7999						8000					

HM		HM0~HM959※1	960
SM		Special purposeSM0~SM2047※2	2048
S	Flow	S0~S1023	1024
HS		HS0~HS127※1	128
T	Timer	T0~T575	576
HT		HT0~HT95※1	96
ET		Precise timer ET0~ET31	32
C	Counter	C0~C575	576
HC		HC0~HC95※1	96
HSC		High speed counter HSC0~HSC31	32
D	Data register	D0~D7999	8000
HD		HD0~HD999※1	1000
SD		Special purposeSD0~SD2047	2048
HSD		Special purposeHSD0~HSD499※2	500
FD	FlashROM register	FD0~FD5119	5120
SFD		Special purposeSFD0~SFD1999※2	2000
FS	Special secret register	FS0~FS47	48
ID※6	Main body	ID0~ID99	100
	expansion BD	ID20000~ID20099(#1 expansionBD) ID20100~ID20199(#2 expansionBD)	200
	expansion ED	ID30000~ID30099(#1 expansionED)	100
QD※7	Main body	QD0~QD99	100
	expansion BD	QD20000~QD20099(#1 expansionBD) QD20100~QD20199(#2 expansionBD)	200
	expansion ED	QD30000~QD30099(#1 expansionED)	100
SEM	Special coil of Sequence block instruction WAIT	SEM0~SEM31	32

XD3 series PLC soft components list:

	Name	Range					Points				
		16 I/O	24 I/O	32 I/O	48 I/O	60 I/O	16	24	32	48	60
X	Input points	X0~X7	X0~X15	X0~X21	X0~X33	X0~X43	8	14	18	28	36
Y	Output points	Y0~Y7	Y0~Y11	Y0~Y15	Y0~Y23	Y0~Y27	8	10	14	20	24
X	Input points※3	X10000~X10077(#1 expansion module) …… X11100~X11177(#10 expansion module)					640				
Y	Output points※3	Y10000~Y10077(#1 expansion module) …… Y11100~Y11177(#10 expansion module)					640				
X	Input points※4	X20000~X20077(#1expansionBD) X20100~X20177(#2expansionBD)					128				
Y	Output points※4	Y20000~Y20077(#1expansionBD) Y20100~Y20177(#2expansionBD)					128				
X	Input points※5	X30000~X30077(#1expansionED)					64				
Y	Output points※5	Y30000~Y30077(#1expansionED)					64				
M	Internal relay	M0~M7999					8000				

HM		HM0~HM959[※1]	960
SM		special purpose SM0~SM2047[※2]	2048
S	Flow	S0~S1023	1024
HS		HS0~HS127[※1]	128
T	Timer	T0~T575	576
HT		HT0~HT95[※1]	96
ET		precise timer ET0~ET31	32
C	Counter	C0~C575	576
HC		HC0~HC95[※1]	96
HSC		High speed counter HSC0~HSC31	32
D	Data register	D0~D7999	8000
HD		HD0~HD999[※1]	1000
SD		special purpose SD0~SD2047	2048
HSD		special purpose HSD0~HSD499[※2]	500
FD	FlashROM register	FD0~FD5119	5120
SFD		special purpose SFD0~SFD1999[※2]	2000
FS	Special secret register	FS0~FS47	48
ID[※6]	Main body	ID0~ID99	100
	Expansion module	ID10000~ID10099(#1 expansion module) …… ID10900~ID10999(#10 expansion module)	1000
	expansion BD	ID20000~ID20099(#1expansionBD) ID20100~ID20199(#2expansionBD)	200
	expansion ED	ID30000~ID30099(#1expansionED)	100
QD[※7]	Main body	QD0~QD99	100
	Expansion module	QD10000~QD10099(#1 expansion module) …… QD10900~QD10999(#10 expansion module)	1000
	expansion BD	QD20000~QD20099(#1expansionBD) QD20100~QD20199(#2expansionBD)	200
	expansion ED	QD30000~QD30099(#1expansionED)	100
SEM	Special coil of Sequence block instruction WAIT	SEM0~SEM31	32

XD5 series PLC soft components list:

	Name	Range					Points				
		16I/O	24 I/O	32 I/O	48 I/O	60 I/O	16	24	32	48	60
X	Input points	X0~X7	X0~X15	X0~X21	X0~X33	X0~X43	8	14	18	28	36
Y	Output points	Y0~Y7	Y0~Y11	Y0~Y15	Y0~Y23	Y0~Y27	8	10	14	20	24
X	Input points※3	X10000~X10077(#1 expansion module) …… X11700~X11777(#16 expansion module)					1024				
Y	Output points※3	Y10000~Y10077(#1 expansion module) …… Y11700~Y11777(#16 expansion module)					1024				
X	Input points※4	X20000~X20077(#1expansionBD) X20100~X20177(#2expansionBD)					128				
Y	Output points※4	Y20000~Y20077(#1expansionBD) Y20100~Y20177(#2expansionBD)					128				
X	Input points※5	X30000~X30077(#1expansionED)					64				
Y	Output points※5	Y30000~Y30077(#1expansionED)					64				
M	Internal relay	M0~M69999					70000				
HM		HM0~HM11999※1					12000				
SM		special purpose SM0~SM4999※2					5000				
S	Flow	S0~S7999					8000				
HS		HS0~HS999※1					1000				
T	Timer	T0~T4999					5000				
HT		HT0~HT1999※1					2000				
ET		precise timer ET0~ET39					40				
C	Counter	C0~C4999					5000				
HC		HC0~HC1999※1					2000				
HSC		high speed counter HSC0~HSC39					40				
D	Data register	D0~D69999(firmware V3.5.3 and up)					70000				
		D0~D59999(firmware V3.5.2 and down)					60000				
HD		HD0~HD24999※1					25000				
SD		special purpose SD0~SD4999					5000				
HSD		special purpose HSD0~HSD1023※2					1024				
FD	FlashROM Register	FD0~FD8191					8192				
SFD		special purpose SFD0~SFD5999※2					6000				
FS	Special secret register	FS0~FS47					48				
ID※6	Main body	ID0~ID99					100				
	Expansion module	ID10000~ID10099(#1 expansion module) …… ID11500~ID11599(#16 expansion module)					1600				
	expansion BD	ID20000~ID20099(#1expansionBD) ID20100~ID20199(#2expansionBD)					200				
	expansion ED	ID30000~ID30099(#1expansionED)					100				
QD※7	Main body	QD0~QD99					100				
	Expansion module	QD10000~QD10099(#1 expansion module) …… QD11500~QD11599(#16 expansion module)					1600				

	expansion BD	QD20000~QD20099(#1expansionBD) QD20100~QD20199(#2expansionBD)	200
	expansion ED	QD30000~QD30099(#1expansionED)	100
SEM	Special coil of Sequence block instruction WAIT	SEM0~SEM31	32

XDM series PLC soft components list:

	Name	Range			Points		
		24 I/O	32 I/O	60 I/O	24	32	60
X	Input points	X0~X15	X0~X21	X0~X43	14	18	36
Y	Output points	Y0~Y11	Y0~Y15	Y0~Y27	10	14	24
X	Input points※3	X10000~X10077(#1 expansion module) …… X11700~X11777(#16 expansion module)			1024		
Y	Output points※3	Y10000~Y10077(#1 expansion module) …… Y11700~Y11777(#16 expansion module)			1024		
X	Input points※4	X20000~X20077(#1expansionBD) X20100~X20177(#2expansionBD)			128		
Y	Output points※4	Y20000~Y20077(#1expansionBD) Y20100~Y20177(#2expansionBD)			128		
X	Input points※5	X30000~X30077(#1expansionED)			64		
Y	Output points※5	Y30000~Y30077(#1expansionED)			64		
M	Internal relay	M0~M69999			70000		
HM		HM0~HM11999※1			12000		
SM		special purpose SM0~SM4999※2			5000		
S	Flow	S0~S7999			8000		
HS		HS0~HS999※1			1000		
T	Timer	T0~T4999			5000		
HT		HT0~HT1999※1			2000		
ET		precise timer ET0~ET39			40		
C	Counter	C0~C4999			5000		
HC		HC0~HC1999※1			2000		
HSC		High speed counter HSC0~HSC39			40		
D	Data register	D0~D69999			70000		
HD		HD0~HD24999※1			25000		
SD		special purpose SD0~SD4999			5000		
HSD		special purpose HSD0~HSD1023※2			1024		
FD	FlashROM register	FD0~FD8191			8192		
SFD		special purpose SFD0~SFD5999※2			6000		
FS	Special secret register	FS0~FS47			48		
ID※6	Main body	ID0~ID99			100		

	Expansion module	ID10000~ID10099(#1 expansion module) …… ID11500~ID11599(#16 expansion module)	1600
	expansion BD	ID20000~ID20099(#1expansionBD) ID20100~ID20199(#2expansionBD)	200
	expansion ED	ID30000~ID30099(#1expansionED)	100
QD※7	Main body	QD0~QD99	100
	Expansion module	QD10000~QD10099(#1 expansion module) …… QD11500~QD11599(#16 expansion module)	1600
	expansion BD	QD20000~QD20099(#1expansionBD) QD20100~QD20199(#2expansionBD)	200
	expansion ED	QD30000~QD30099(#1expansionED)	100
SEM	Special coil of Sequence block instruction WAIT	SEM0~SEM31	32

XDC series PLC soft components list:

	Name	Range				Points			
		24 I/O	32 I/O	48 I/O	60 I/O	24	32	48	60
X	Input points	X0~X15	X0~X21	X0~X33	X0~X43	14	18	28	36
Y	Output points	Y0~Y11	Y0~Y15	Y0~Y23	Y0~Y27	10	14	20	24
X	Input points※3	X10000~X10077(#1 expansion module) …… X11700~X11777(#16 expansion module)				1024			
Y	Output points※3	Y10000~Y10077(#1 expansion module) …… Y11700~Y11777(#16 expansion module)				1024			
X	Input points※4	X20000~X20077(#1expansionBD) X20100~X20177(#2expansionBD)				128			
Y	Output points※4	Y20000~Y20077(#1expansionBD) Y20100~Y20177(#2expansionBD)				128			
X	Input points※5	X30000~X30077(#1expansionED)				64			
Y	Output points※5	Y30000~Y30077(#1expansionED)				64			
M	Internal relay	M0~M69999				70000			
HM		HM0~HM11999※1				12000			
SM		special purpose SM0~SM4999※2				5000			
S	Flow	S0~S7999				8000			
HS		HS0~HS999※1				1000			
T	Timer	T0~T4999				5000			
HT		HT0~HT1999※1				2000			
ET		Precise timer ET0~ET39				40			
C	Counter	C0~C4999				5000			
HC		HC0~HC1999※1				2000			
HSC		High speed counter HSC0~HSC39				40			
D	Data register	D0~D69999				70000			

HD		HD0~HD24999※1	25000
SD		Special purpose SD0~SD4999	5000
HSD		Special purpose HSD0~HSD1023※2	1024
FD	FlashROM register	FD0~FD8191	8192
SFD		Special purpose SFD0~SFD5999※2	6000
FS	Special secret register	FS0~FS47	48
ID※6	Main body	ID0~ID99	100
	Expansion module	ID10000~ID10099(#1 expansion module) …… ID11500~ID11599(#16 expansion module)	1600
	expansion BD	ID20000~ID20099(#1expansionBD) ID20100~ID20199(#2expansionBD)	200
	expansion ED	ID30000~ID30099(#1expansionED)	100
QD※7	Main body	QD0~QD99	100
	Expansion module	QD10000~QD10099(#1 expansion module) …… QD11500~QD11599(#16 expansion module)	1600
	expansion BD	QD20000~QD20099(#1expansionBD) QD20100~QD20199(#2expansionBD)	200
	expansion ED	QD30000~QD30099(#1expansionED)	100
SEM	Special coil of Sequence block instruction WAIT	SEM0~SEM31	32

XD3E series PLC soft components list:

	Name	Range	
		24 points	
X	Input points	14 points X0~X15	
Y	Output points	10 points Y0~Y11	
X	Input points※3	X10000~X10077(#1expansion module) …… X11700~X11777(#16 expansion module)	832
Y	Output points※3	Y10000~Y10077(#1 expansion module) …… Y11700~Y11777(#16 expansion module)	832
X	Input points※4	X20000~X20077(#1 expansion BD) X20100~X20177(#2 expansion BD)	128
Y	Output points※4	Y20000~Y20077(#1 expansion BD) Y20100~Y20177(#2 expansion BD)	128
X	Input points※5	X30000~X30077(#1 expansion ED)	64

	Name	Range	
		24 points	
Y	Output points※5	Y30000~Y30077(#1 expansion ED)	64
M	Internal relay	M0~M7999	8000
HM		HM0~HM959※1	960
SM		special purpose SM0~SM2047※2	2048
S	Flow	S0~S1023	1021
HS		HS0~HS127※1	128
T	Timer	T0~T575	576
HT		HT0~HT95※1	96
ET		precise timer ET0~ET24	25
C	Counter	C0~C575	576
HC		HC0~HC95※1	96
HSC		high speed counter HSC0~HSC39	40
D	Data register	D0~D7999	8000
HD		HD0~HD999※1	1000
SD		special purpose SD0~SD2047	2048
HSD		special purpose HSD0~HSD499※2	500
FD	FlashROM register	FD0~FD5119	5120
SFD		special purpose SFD0~SFD1999※2	2000
FS	Special secret register	FS0~FS47	48
ID※6	Main body	ID0~ID99	100
	Expansion module	ID10000~ID10099(#1 expansion module) …… ID10900~ID10999(#10 expansion module)	1000
	expansion BD	ID20000~ID20099(#1 expansion BD) ID20100~ID20199(#2 expansion BD)	200
	expansion ED	ID30000~ID30099(#1 expansion ED)	100
QD※7	Main body	QD0~QD99	100
	Expansion module	QD10000~QD10099(#1 expansion module) …… QD10900~QD10999(#10 expansion module)	1000
	Expansion BD	QD20000~QD20099(#1 expansion BD) QD20100~QD20199(#2 expansion BD)	200
	Expansion ED	QD30000~QD30099(#1 expansion ED)	100

	Name	Range	
		24 points	
SEM	Special coil of Sequence block instruction WAIT	SEM0~SEM31	32

XD5E series PLC soft components list:

	Name	Range				Points			
		24 I/O	30 I/O	48 I/O	60 I/O	24	30	48	60
X	Input points	X0~X15	X0~X17	X0~X33	X0~X43	14	16	28	36
Y	Output points	Y0~Y11	Y0~Y15	Y0~Y23	Y0~Y27	10	14	20	24
X	Input points※3	X10000~X10077(#1expansion module) …… X11700~X11777(#16 expansion module)				1024			
Y	Output points※3	Y10000~Y10077(#1 expansion module) …… Y11700~Y11777(#16 expansion module)				1024			
X	Input points※4	X20000~X20077(#1 expansion BD) X20100~X20177(#2 expansion BD)				128			
Y	Output points※4	Y20000~Y20077(#1 expansion BD) Y20100~Y20177(#2 expansion BD)				128			
X	Input points※5	X30000~X30077(#1 expansion ED)				64			
Y	Output points※5	Y30000~Y30077(#1 expansion ED)				64			
M	Internal relay	M0~M69999				70000			
HM		HM0~HM11999※1				12000			
SM		special purpose SM0~SM4999※2				5000			
S	Flow	S0~S7999				8000			
HS		HS0~HS999※1				1000			
T	Timer	T0~T4999				5000			
HT		HT0~HT1999※1				2000			
ET		precise timer ET0~ET39				40			
C	Counter	C0~C4999				5000			
HC		HC0~HC1999※1				2000			
HSC		high speed counter HSC0~HSC39				40			
D	Data register	D0~D69999				70000			
HD		HD0~HD24999※1				25000			
SD		special purpose SD0~SD4999				5000			
HSD		special purpose HSD0~HSD1023※2				1024			
FD	FlashROM	FD0~FD8191				8192			

	Name	Range				Points			
		24 I/O	30 I/O	48 I/O	60 I/O	24	30	48	60
SFD	register	special purpose SFD0~SFD5999[※2]				6000			
FS	Special secret register	FS0~FS47				48			
ID[※6]	Main body	ID0~ID99				100			
	Expansion module	ID10000~ID10099(#1 expansion module) …… ID11500~ID11599(#16 expansion module)				1600			
	expansion BD	ID20000~ID20099(#1 expansion BD) ID20100~ID20199(#2 expansion BD)				200			
	expansion ED	ID30000~ID30099(#1 expansion ED)				100			
QD[※7]	Main body	QD0~QD99				100			
	Expansion module	QD10000~QD10099(#1 expansion module) …… QD11500~QD11599(#16 expansion module)				1600			
	Expansion BD	QD20000~QD20099(#1 expansion BD) QD20100~QD20199(#2 expansion BD)				200			
	Expansion ED	QD30000~QD30099(#1 expansion ED)				100			
SEM	Special coil of Sequence block instruction WAIT	SEM0~SEM31				32			

XDME series PLC soft components list:

	Name	Range		Points	
		30 I/O	60 I/O	30	60
X	Input points	X0~X17	X0~X43	16	36
Y	Output points	Y0~Y15	Y0~Y27	14	24
X	Input points[※3]	X10000~X10077(#1 expansion module) …… X11700~X11777(#16 expansion module)		1024	
Y	Output points[※3]	Y10000~Y10077(#1 expansion module) …… Y11700~Y11777(#16 expansion module)		1024	
X	Input points[※4]	X20000~X20077(#1expansionBD) X20100~X20177(#2expansionBD)		128	
Y	Output points[※4]	Y20000~Y20077(#1expansionBD) Y20100~Y20177(#2expansionBD)		128	

X	Input points※5	X30000~X30077(#1expansionED)	64
Y	Output points※5	Y30000~Y30077(#1expansionED)	64
M	Internal relay	M0~M69999	70000
HM		HM0~HM11999※1	12000
SM		special purpose SM0~SM4999※2	5000
S	Flow	S0~S7999	8000
HS		HS0~HS999※1	1000
T	Timer	T0~T4999	5000
HT		HT0~HT1999※1	2000
ET		precise timer ET0~ET39	40
C	Counter	C0~C4999	5000
HC		HC0~HC1999※1	2000
HSC		high speed counter HSC0~HSC39	40
D	Data register	D0~D69999	70000
HD		HD0~HD24999※1	25000
SD		special purpose SD0~SD4999	5000
HSD		special purpose HSD0~HSD1023※2	1024
FD	FlashROM register	FD0~FD8191	8192
SFD		special purpose SFD0~SFD5999※2	6000
FS	Special secret register	FS0~FS47	48
ID※6	Main body	ID0~ID99	100
	Expansion module	ID10000~ID10099(#1 expansion module) …… ID11500~ID11599(#16 expansion module)	1600
	expansion BD	ID20000~ID20099(#1expansionBD) ID20100~ID20199(#2expansionBD)	200
	expansion ED	ID30000~ID30099(#1expansionED)	100
QD※7	Main body	QD0~QD99	100
	Expansion module	QD10000~QD10099(#1 expansion module) …… QD11500~QD11599(#16 expansion module)	1600
	expansion BD	QD20000~QD20099(#1expansionBD) QD20100~QD20199(#2expansionBD)	200
	expansion ED	QD30000~QD30099(#1expansionED)	100
SEM	Special coil of Sequence block instruction WAIT	SEM0~SEM31	32

XDH series PLC soft components list:

	Name	Range		Points	
		30 I/O	60 I/O	30	60
X	Input points	X0~X17	X0~X43	16	36
Y	Output points	Y0~Y15	Y0~Y27	14	24
X	Input points※3	X10000~X10077(#1 expansion module) …… X11700~X11777(#16 expansion module)		1024	
Y	Output points※3	Y10000~Y10077(#1 expansion module) …… Y11700~Y11777(#16 expansion module)		1024	
X	Input points※4	X20000~X20077(#1 expansion BD) X20100~X20177(#2 expansion BD)		128	
Y	Output points※4	Y20000~Y20077(#1 expansion BD) Y20100~Y20177(#2 expansion BD)		128	
X	Input points※5	X30000~X30077(#1 expansion ED)		64	
Y	Output points※5	Y30000~Y30077(#1 expansion ED)		64	
M	Internal relay	M0~M19999		200000	
HM		HM0~HM19999※1		20000	
SM		special purpose SM0~SM49999※2		50000	
S	Flow	S0~S19999		20000	
HS		HS0~HS1999※1		2000	
T	Timer	T0~T19999		20000	
HT		HT0~HT1999※1		2000	
ET		Precise timer ET0~ET39		40	
C	Counter	C0~C19999		20000	
HC		HC0~HC1999※1		2000	
HSC		High speed counter HSC0~HSC39		40	
D	Data register	D0~D499999		500000	
HD		HD0~HD49999※1		50000	
SD		special purpose SD0~SD49999		50000	
HSD		special purpose HSD0~HSD49999※2		50000	
FD	FlashROM register	FD0~FD65535		65536	
SFD		special purpose SFD0~SFD49999※2		50000	
FS	Special secret register	FS0~FS47		48	
ID※6	Main body	ID0~ID99		100	
	Expansion module	ID10000~ID10099(#1 expansion module) …… ID11500~ID11599(#16 expansion module)		1600	
	expansion BD	ID20000~ID20099(#1 expansion BD) ID20100~ID20199(#2 expansion BD)		200	
	expansion ED	ID30000~ID30099(#1 expansion ED)		100	
QD※7	Main body	QD0~QD99		100	
	Expansion module	QD10000~QD10099(#1 expansion module) …… QD11500~QD11599(#16 expansion module)		1600	
	expansion BD	QD20000~QD20099(#1 expansion BD) QD20100~QD20199(#2 expansion BD)		200	
	expansion ED	QD30000~QD30099(#1 expansion ED)		100	

SEM	Special coil of Sequence block instruction WAIT	SEM0~SEM31	32

XL1 series PLC soft components list:

	Name	Range	Points
		16 I/O	16
X	Input points	X0~X7	8
Y	Output points	Y0~Y7	8
X	Input points※3	X10000~X10077(#1 expansion module) …… X11100~X11177(#10 expansion module)	640
Y	Output points※3	Y10000~Y10077(#1 expansion module) …… Y11100~Y11177(#10 expansion module)	640
X	Input points※4	X20000~X20077(#1expansionBD) X20100~X20177(#2expansionBD)	128
Y	Output points※4	Y20000~Y20077(#1expansionBD) Y20100~Y20177(#2expansionBD)	128
X	Input points※5	X30000~X30077(#1expansionED)	64
Y	Output points※5	Y30000~Y30077(#1expansionED)	64
M	Internal relay	M0~M7999	8000
HM		HM0~HM959※1	960
SM		special purpose SM0~SM2047※2	2048
S	Flow	S0~S1023	1024
HS		HS0~HS127※1	128
T	Timer	T0~T575	576
HT		HT0~HT95※1	96
ET		precise timer ET0~ET31	32
C	Counter	C0~C575	576
HC		HC0~HC95※1	96
HSC		high speed counter HSC0~HSC31	32
D	Data register	D0~D7999	8000
HD		HD0~HD999※1	1000
SD		special purpose SD0~SD2047	2048
HSD		special purpose HSD0~HSD499※2	500
FD	FlashROM register	FD0~FD5119	5120
SFD		special purpose SFD0~SFD1999※2	2000
FS	Special secret register	FS0~FS47	48
ID※6	Main body	ID0~ID99	100
	Expansion module	ID10000~ID10099(#1 expansion module) …… ID10900~ID10999(#10 expansion module)	1000

	expansion BD	ID20000~ID20099(#1expansionBD) ID20100~ID20199(#2expansionBD)	200
	expansion ED	ID30000~ID30099(#1expansionED)	100
QD[※7]	Main body	QD0~QD99	100
	Expansion module	QD10000~QD10099(#1 expansion module) …… QD10900~QD10999(#10 expansion module)	1000
	expansion BD	QD20000~QD20099(#1expansionBD) QD20100~QD20199(#2expansionBD)	200
	expansion ED	QD30000~QD30099(#1expansionED)	100
SEM	Special coil of Sequence block instruction WAIT	SEM0~SEM31	32

XL3 series PLC soft components list:

	Name	Range		Points	
		16 I/O	32 I/O	16	32
X	Input points	X0~X7	X0~X17	8	16
Y	Output points	Y0~Y7	Y0~Y17	8	16
X	Input points[※3]	X10000~X10077(#1 expansion module) …… X11100~X11177(#10 expansion module)		640	
Y	Output points[※3]	Y10000~Y10077(#1 expansion module) …… Y11100~Y11177(#10 expansion module)		640	
X	Input points[※4]	X20000~X20077(#1 expansion BD) X20100~X20177(#2 expansion BD)		128	
Y	Output points[※4]	Y20000~Y20077(#1 expansion BD) Y20100~Y20177(#2 expansion BD)		128	
X	Input points[※5]	X30000~X30077(#1 expansion ED)		64	
Y	Output points[※5]	Y30000~Y30077(#1 expansion ED)		64	
M	Internal relay	M0~M7999		8000	
HM		HM0~HM959[※1]		960	
SM		special purpose SM0~SM2047[※2]		2048	
S	Flow	S0~S1023		1024	
HS		HS0~HS127[※1]		128	
T	Timer	T0~T575		576	
HT		HT0~HT95[※1]		96	
ET		precise timer ET0~ET31		32	
C	Counter	C0~C575		576	
HC		HC0~HC95[※1]		96	
HSC		high speed counter HSC0~HSC31		32	
D	Data register	D0~D7999		8000	

	Name	Range		Points	
		16 I/O	32 I/O	16	32
HD		HD0~HD999※1		1000	
SD		special purpose SD0~SD2047		2048	
HSD		special purpose HSD0~HSD499※2		500	
FD	FlashROM register	FD0~FD5119		5120	
SFD	FlashROM register	special purpose SFD0~SFD1999※2		2000	
FS	Special secret register	FS0~FS47		48	
ID※6	Main body	ID0~ID99		100	
ID※6	Expansion module	ID10000~ID10099(#1 expansion module) …… ID10900~ID10999(#10 expansion module)		1000	
ID※6	expansion BD	ID20000~ID20099(#1 expansion BD) ID20100~ID20199(#2 expansion BD)		200	
ID※6	expansion ED	ID30000~ID30099(#1 expansion ED)		100	
QD※7	Main body	QD0~QD99		100	
QD※7	Expansion module	QD10000~QD10099(#1 expansion module) …… QD10900~QD10999(#10 expansion module)		1000	
QD※7	expansion BD	QD20000~QD20099(#1 expansion BD) QD20100~QD20199(#2 expansion BD)		200	
QD※7	expansion ED	QD30000~QD30099(#1 expansion ED)		100	
SEM	Special coil of Sequence block instruction WAIT	SEM0~SEM31		32	

XL5, XL5E, XLME series PLC soft components list:

	Name	Range			Points		
		16 I/O	32 I/O	64 I/O	16	32	64
X	Input points	X0~X7	X0~X17	X0~X37	8	16	32
Y	Output points	Y0~Y7	Y0~Y17	Y0~Y37	8	16	32
X	Input points※3	X10000~X10077(#1 expansion module) …… X11700~X11777(#16 expansion module)			1024		
Y	Output points※3	Y10000~Y10077(#1 expansion module) …… Y11700~Y11777(#16 expansion module)			1024		
X	Input points※4	X20000~X20077(#1expansionBD) X20100~X20177(#2expansionBD)			128		

Y	Output points※4	Y20000~Y20077(#1expansionBD) Y20100~Y20177(#2expansionBD)	128
X	Input points※5	X30000~X30077(#1expansionED)	64
Y	Output points※5	Y30000~Y30077(#1expansionED)	64
M	Internal relay	M0~M69999	70000
HM		HM0~HM11999※1	12000
SM		special purpose SM0~SM4999※2	5000
S	Flow	S0~S7999	8000
HS		HS0~HS999※1	1000
T	Timer	T0~T4999	5000
HT		HT0~HT1999※1	2000
ET		precise timer ET0~ET39	40
C	Counter	C0~C4999	5000
HC		HC0~HC1999※1	2000
HSC		high speed counter HSC0~HSC39	40
D	Data register	D0~D69999	70000
HD		HD0~HD24999※1	25000
SD		special purpose SD0~SD4999	5000
HSD		special purpose HSD0~HSD1023※2	1024
FD	FlashROM register	FD0~FD8191	8192
SFD		special purpose SFD0~SFD5999※2	6000
FS	Special secret register	FS0~FS47	48
ID※6	Main body	ID0~ID99	100
	Expansion module	ID10000~ID10099(#1 expansion module) …… ID11500~ID11599(#16 expansion module)	1600
	expansion BD	ID20000~ID20099(#1expansion BD) ID20100~ID20199(#2expansion BD)	200
	expansion ED	ID30000~ID30099(#1expansion ED)	100
QD※7	Main body	QD0~QD99	100
	Expansion module	QD10000~QD10099(#1 expansion module) …… QD11500~QD11599(#16 expansion module)	1600
	expansion BD	QD20000~QD20099(#1expansion BD) QD20100~QD20199(#2expansion BD)	200
	expansion ED	QD30000~QD30099(#1expansion ED)	100
SEM	Special coil of Sequence block instruction WAIT	SEM0~SEM31	32

XL5H series PLC soft components list:

	Name	Range	
		24 points	
X	Input points※5	12 points X0-X13	
Y	Output points※5	12 points Y0-Y13	
X	Module input points※3	X10000~X10077(#1 expansion module) …… X11100~X11177(#16 expansion module)	1024
Y	Module output points※3	Y10000~Y10077(#1 expansion module) …… Y11100~Y11177(#16 expansion module)	1024
X	BD input points※4	X20000~X20077(#1 expansion BD) X20100~X20177(#2 expansion BD)	128
Y	BD output points※4	Y20000~Y20077(#1 expansion BD) Y20100~Y20177(#2 expansion BD)	128
X	ED input points※5	X30000~X30077(#1 expansion ED)	64
Y	ED output points※5	Y30000~Y30077(#1 expansion ED)	64
M	Internal relay	M0~M69999	70000
HM		HM0~HM11999※1	12000
SM		special purpose SM0~SM4999※2	5000
S	Flow	S0~S7999	8000
HS		HS0~HS999※1	1000
T	Timer	T0~T4999	5000
HT		HT0~HT1999※1	2000
ET		precise timer ET0~ET39	40
C	Counter	C0~C4999	5000
HC		HC0~HC1999※1	2000
HSC		high speed counter HSC0~HSC39	40
D	Data register	D0~D69999	70000
HD		HD0~HD24999※1	5000
SD		special purpose SD0~SD4999	5000
HSD		special purpose HSD0~HSD1023※2	1024
FD	FlashROM register	FD0~FD8191	8192
SFD		special purpose SFD0~SFD5999※2	6000
FS	Special secret register	FS0~FS47	48
ID※6	Main body	ID0~ID99	100

	Name	Range	
		24 points	
	Expansion module	ID10000~ID10099(#1 expansion module) …… ID11500~ID11599(#16 expansion module)	1600
	expansion BD	ID20000~ID20099(#1 expansion BD) ID20100~ID20199(#2 expansion BD)	200
	expansion ED	ID30000~ID30099(#1 expansion ED)	100
QD[※7]	Main body	QD0~QD99	100
	Expansion module	QD10000~QD10099(#1 expansion module) …… QD11500~QD11599(#16 expansion module)	1600
	expansion BD	QD20000~QD20099(#1 expansion BD) QD20100~QD20199(# expansion BD)	200
	expansion ED	QD30000~QD30099(#1 expansion ED)	100
SEM	Special coil of Sequence block instruction WAIT	SEM0~SEM31	32

XL5N series PLC soft components list:

	Name	Range	
		32 points	
X	Input points[※5]	16 points X0-X17	
Y	Output points[※5]	16 points Y0-Y17	
X	Module input points[※3]	X10000~X10077(#1 expansion module) …… X11100~X11177(#16 expansion module)	1024
Y	Module output points[※3]	Y10000~Y10077(#1 expansion module) …… Y11100~Y11177(#16 expansion module)	1024
X	BD input points[※4]	X20000~X20077(#1 expansion BD) X20100~X20177(#2 expansion BD)	128

	Name	Range	
		32 points	
Y	BD output points※4	Y20000~Y20077(#1 expansion BD) Y20100~Y20177(#2 expansion BD)	128
X	ED input points※5	X30000~X30077(#1 expansion ED)	64
Y	ED output points※5	Y30000~Y30077(#1 expansion ED)	64
M	Internal relay	M0~M199999	200000
HM		HM0~HM19999※1	20000
SM		special purpose SM0~SM4999※2	5000
S	Flow	S0~S19999	20000
HS		HS0~HS1999※1	2000
T	Timer	T0~T19999	20000
HT		HT0~HT1999※1	2000
ET		precise timer ET0~ET39	40
C	Counter	C0~C19999	20000
HC		HC0~HC1999※1	2000
HSC		high speed counter HSC0~HSC39	40
D	Data register	D0~D499999	500000
HD		HD0~HD49999※1	50000
SD		special purpose SD0~SD49999	50000
HSD		special purpose HSD0~HSD1999※2	2000
FD	FlashROM register	FD0~FD65535	65536
SFD		special purpose SFD0~SFD49999※2	50000
FS	Special secret register	FS0~FS47	48
ID※6	Main body	ID0~ID99	100
	Expansion module	ID10000~ID10099(#1 expansion module) …… ID11500~ID11599(#16 expansion module)	1600
	expansion BD	ID20000~ID20099(#1 expansion BD) ID20100~ID20199(#2 expansion BD)	200
	expansion ED	ID30000~ID30099(#1 expansion ED)	100
QD※7	Main body	QD0~QD99	100
	Expansion module	QD10000~QD10099(#1 expansion module) …… QD11500~QD11599(#16 expansion module)	1600

	Name	Range	
		32 points	
	expansion BD	QD20000~QD20099(#1 expansion BD) QD20100~QD20199(# expansion BD)	200
	expansion ED	QD30000~QD30099(#1 expansion ED)	100
SEM	Special coil of Sequence block instruction WAIT	SEM0~SEM31	32

XLH series PLC soft components list:

	Name	Range	Points
		24 I/O	24
X	Input points※5	X0-X13	12
Y	Output points※5	Y0-Y13	12
X	Internal input points	X0-X77	64
Y	Internal output points	Y0-Y77	64
X	Module input points※3	X10000~X10077(#1 expansion module) …… X11100~X11177(#16 expansion module)	1024
Y	Module output points※3	Y10000~Y10077(#1 expansion module) …… Y11100~Y11177(#16 expansion module)	1024
X	BD input points※4	X20000~X20077(#1 expansion BD) X20100~X20177(#2 expansion BD)	128
Y	BD output points※4	Y20000~Y20077(#1 expansion BD) Y20100~Y20177(#2 expansion BD)	128
X	ED input points※5	X30000~X30077(#1 expansion ED)	64
Y	ED output points※5	Y30000~Y30077(#1 expansion ED)	64
M	Internal relay	M0~M199999	200000
HM		HM0~HM19999※1	20000
SM		special purpose SM0~SM49999※2	50000
S	Flow	S0~S19999	20000
HS		HS0~HS1999※1	2000
T	Timer	T0~T19999	20000

	Name	Range	Points
		24 I/O	24
HT		HT0~HT1999※1	2000
ET		precise timer ET0~ET39	40
C	Counter	C0~C19999	20000
HC		HC0~HC1999※1	2000
HSC		high speed counter HSC0~HSC39	40
D	Data register	D0~D499999	500000
HD		HD0~HD49999※1	50000
SD		special purpose SD0~SD49999	50000
HSD		special purpose HSD0~HSD49999※2	50000
FD	FlashROM register	FD0~FD65535	65536
SFD		special purpose SFD0~SFD65487※2	65488
FS	Special secret register	FS0~FS47	48
ID※6	Main body	ID0~ID99	100
	Expansion module	ID10000~ID10099(#1 expansion module) …… ID11500~ID11599(#16 expansion module)	1600
	expansion BD	ID20000~ID20099(#1 expansion BD) ID20100~ID20199(#2 expansion BD)	200
	expansion ED	ID30000~ID30099(#1 expansion ED)	100
QD※7	Main body	QD0~QD99	100
	Expansion module	QD10000~QD10099(#1 expansion module) …… QD11500~QD11599(#16 expansion module)	1600
	expansion BD	QD20000~QD20099(#1 expansion BD) QD20100~QD20199(# expansion BD)	200
	expansion ED	QD30000~QD30099(#1 expansion ED)	100
SEM	Special coil of Sequence block instruction WAIT	SEM0~SEM31	32

※1: 【】Memory area is the default power outage holding area (Note: XD/XL series PLC power outage holding area can not be modified).
※2: Special use (non-power-down maintenance) refers to registers for special use occupied by the system, which can not be used for other purposes. For details, refer to the relevant sections of the List of Special Soft Components in the appendix of this manual.
※3: I/O address assignment (octal) of the extended module, which can be used as intermediate relay when the extension module is not connected. (XL1/XD1/XD2 does not support extension modules, XD3/XD3E/XL3 can expand up to 10 at the same time, XD5/XDM/XDC/XD5E/XDME/XDH/XL5/XL5E/XLME/XL5N/XL5H/XLH can expand up to 16 at the same time)
※4: Extended BD I/O address allocation (octal), can be used as intermediate relay when not connected to BD. (24/32/30 points can be extended up to 1, 48/60 points can be extended up to 2, 16 points do not support extended BD, XL/XDH series does not support extended BD)
※5: Extended ED I/O address allocation (octal), can be used as intermediate relay when not connected to ED. (XD/XL series can extend up to one ED module, XDH cannot support ED module)
※6: Analog input soft component address, can be used as auxiliary register when not connected to extended equipment.
※7: Analog output soft component address, can be used as auxiliary registers when not connected to extended devices.
※8: The range of soft components mentioned above is the valid range of PLC in X-NET communication mode. In MODBUS communication mode, some relays can not read and write. The specific usable range is shown in chapter 6-2-3.

2-4 Input/output relays (X, Y)

Number List

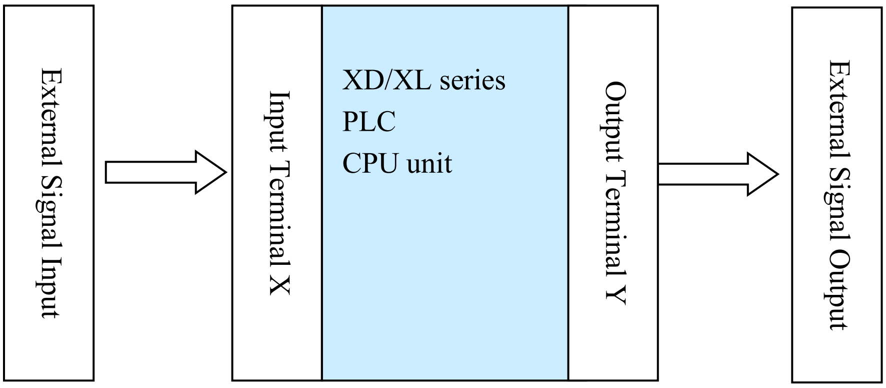
XD series PLC input/output are all in octal form, each series numbers are listed below:

Series	Name	Range								Points							
		10 I/O	16 I/O	24 I/O	30 I/O	32 I/O	42 I/O	48 I/O	60 I/O	10	16	24	30	32	42	48	60
XD1	X	X0~X4	X0~X7	X0~X13	-	X0~X17		-	-	5	8	12	-	16	-	-	-
	Y	Y0~Y4	Y0~Y7	X0~X13	-	Y0~Y17		-	-	5	8	12	-	16	-	-	-
XD2 XD3 XD5	X	-	X0~X7	X0~X15	-	X0~X21	X0~X27	X0~X33	X0~X43	-	8	14	-	18	24	28	36
	Y	-	Y0~Y7	Y0~Y11	-	Y0~Y15	Y0~Y21	Y0~Y23	Y0~Y27	-	8	10	-	14	18	20	24
XDM	X	-	-	X0~X15	-	X0~X21		-	X0~X43	-	-	14	-	18	-	-	36
	Y	-	-	Y0~Y11	-	Y0~Y15		-	Y0~Y27	-	-	10	-	14	-	-	24
XDC	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
XD3E	X	-	-	X0~X15	-	-		-	-	-	-	14	-	-	-	-	-
	Y	-	-	Y0~Y11	-	-		-	-	-	-	10	-	-	-	-	-
XD5E	X	-	-	X0~X15	X0~X17	-		X0~X33	X0~X43	-	-	14	16	-	-	28	36
	Y	-	-	Y0~Y11	Y0~Y15	-		Y0~Y23	Y0~Y27	-	-	10	14	-	-	20	24
XDME	X	-	-	-	X0-X17	-		-	X0~X43	-	-	-	16	-	-	-	36
	Y	-	-	-	Y0-Y15	-		-	Y0~Y27	-	-	-	14	-	-	-	24
XDH	X	-	-	-	X0-X17	-		-	X0~X43	-	-	-	16	-	-	-	36
	Y	-	-	-	Y0-Y15	-		-	Y0~Y27	-	-	-	14	-	-	-	24

XL series PLC input/output are all in octal form, each series numbers are listed below:

Series	Name	Range					Points				
		16 I/O	24 I/O	30 I/O	32 I/O	64 I/O	16	24	30	32	64
XL1	X	X0~X7	-	-	-	-	8	-	-	-	-
	Y	Y0~Y7	-	-	-	-	8	-	-	-	-
XL3 XL5 XL5E	X	X0~X7	-	-	X0~X17	X0~X37	8	-	-	16	32
	Y	Y0~Y7	-	-	Y0~Y17	Y0~Y37	8	-	-	16	32
XL5H	X	-	X0~X13	-	-	-	-	12	-	-	-
	Y	-	Y0~Y13	-	-	-	-	12	-	-	-
XL5N	X	-	-	-	X0~X17	-	-	-	-	16	-
	Y	-	-	-	Y0~Y17	-	-	-	-	16	-
XLME	X	-	-	-	X0~X17	X0~X37	-	-	-	16	32
	Y	-	-	-	Y0~Y17	Y0~Y37	-	-	-	16	32
XLH	X	-	X0~X13	X0-X15	-	-	-	12	14	-	-
	Y	-	Y0~Y13	Y0~Y17	-	-	-	12	16	-	-

Function



Input Relay X

PLC input terminals are used to recive the external signal. the input relays are optocoupler to connect PLC and input terminals

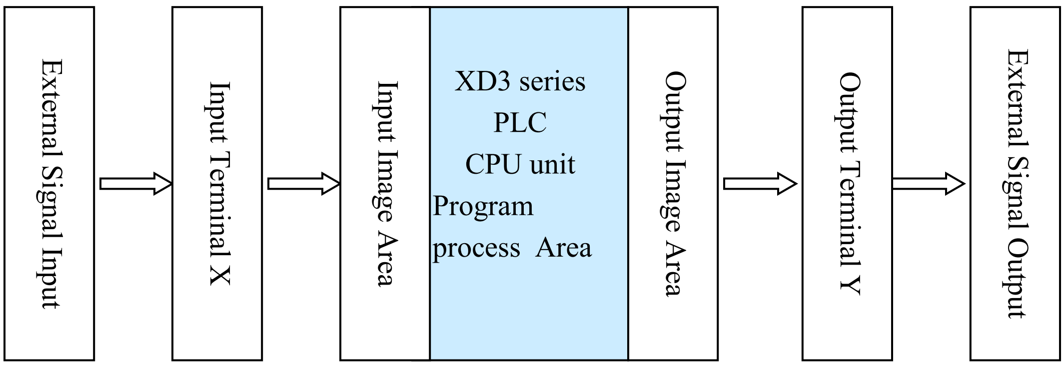
The input relays which are not connected with external devices can be seemed to fast internal relays

Output Relay Y

PLC 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 which are not connected with external devices can be seemed to fast internal relays

Execution Order



Input processing

Before PLC executing the program, read every input terminal's ON/OFF status to the image area.
When the program is running, even the input changed, the content in the input image area will not change until the next scanning period coming.
Output processing
After running 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 output contactors will delay the action according to the output soft components reponse.

2-5 Auxiliary Relay (M, HM, SM)

Number List

The auxiliary relays in XD/XL series PLC are all in decimal form, please see the following table:

Series	Name	Range		
		Normal	Power-off holding	Special
XD1/XD2/XD3/XD3E	M HM SM	M0~M7999	HM0-HM959	SM0~SM2047
XD5/XDM/XDC/XD5E/XDME		M0~M69999	HM0-HM11999	SM0~SM4999
XDH		M0~M199999	HM0~HM19999	SM0~SM49999
XL1/XL3		M0~M7999	HM0-HM959	SM0~SM2047
XL5/XL5E/XLME/XL5H		M0~M69999	HM0-HM11999	SM0~SM4999
XLH/XL5N		M0~M199999	HM0~HM19999	SM0~SM49999

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

- For common use

This type of auxiliary relays can be used only as normal auxiliary relays. I.e. if power supply suddenly shut down during the running, the relays will be off.
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, even the PLC is OFF, they can keep the ON/OFF status.
Power off retentive zone cannot be modified;
Power off retentive relays are usually used to memory the status before stop the power, then when power the PLC on again, the status can run again;

- For Special Usage

Special relays are some relays which are defined with special meanings or functions, start from SM0.

There are two functions for special relays, first is used to drive the coil, the other type is forspecial running.
E.g.: SM2 is the initial pulse, activates only at the moment of start
SM34 is "all output disabled"
Special auxiliary relays can't be used as normal relay M;

Note: The range of soft components mentioned above is the valid range of PLC in the X-NET communication mode. In the MODBUS communication mode, some relays can not read and write. The specific usable range is shown in chapter 6-2-3.

2-6 Status Relay (S, HS)

Address List

Status relays addresses of XD/XL series PLC are in form of decimal, the address are shown below:

Series	Name	Range	
		Normal	Power-off holding
XD1/XD2/XD3/XD3E	S HS	S0~S1023	HS0~HS127
XD5/XDM/XDC/XD5E/XDME		S0~S7999	HS0~HS999
XDH		S0~S19999	HS0~HS1999
XL1/XL3		S0~S1023	HS0~HS127
XL5/XL5E/XLME/XL5H		S0~S7999	HS0~HS999
XLH/XL5N		S0~S19999	HS0~HS1999

Function

Status relays S and HS are very import in ladder program; they are used together with instruction "STL" in the flow. The flow can make the program clear and easy to modify.

- For common use

After shut off the PLC power, S relays will be OFF

- For Power Off Retentive Use

HS relays can keep the ON/OFF status even PLC power is off

- The status relays also have countless "normally ON/OFF" contactors. So users can use them freely in the program.

Note: The range of soft components mentioned above is the valid range of PLC in the X-NET communication mode. In the MODBUS communication mode, some relays can not read and write. The specific usable range is shown in chapter 6-2-3.

2-7 Timer (T, HT)

Address List

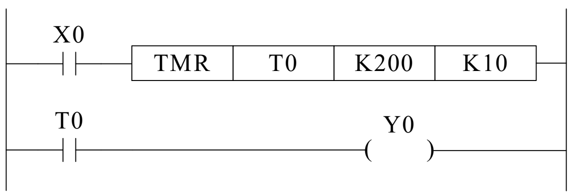
The timer addresses of XD/XL series PLC are in the form of decimal; please see the following table:

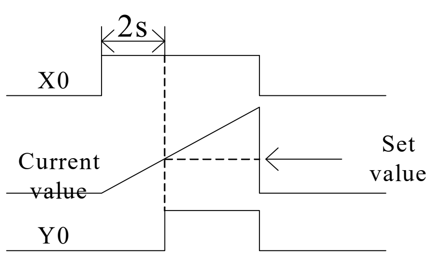
Series	Name	Range		
		Normal	Power-off holding	Precise timer
XD1/XD2/XD3	T HT ET	T0~T575	HT0~HT95	ET0~ET24
XD5/XDM/XDC/XD5E/XDME		T0~T4999	HT0~HT1999	ET0~ET24
XDH		T0~T19999	HT0~HT1999	-
XL1/XL3		T0~T575	HT0~HT95	ET0~ET24
XL5/XL5E/XLME/XL5H		T0~T4999	HT0~HT1999	ET0~ET24
XLH/XL5N		T0~T19999	HT0~HT1999	-

Function

The timers accumulate the 1ms, 10ms, 100ms pulse, the output contactor activates when the accumulation reaches the set value;
TMR instruction is for common timers. The set value can be constant (K) or data register (D).

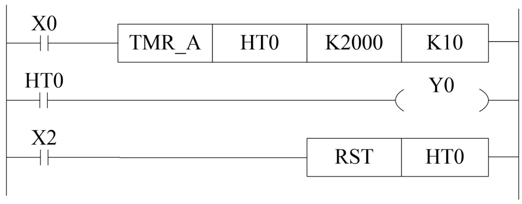
Normal type

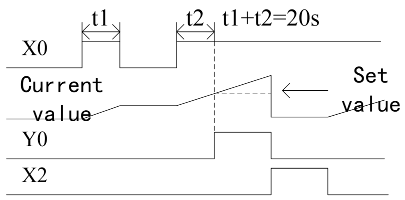




If X0 is ON, then T0 accumulates 10ms pulse based on the current value; when the accumulation value reaches the set value K200, the timer outputactivates. I.e. the output activates 2s later. If X0 is OFF, the timer resets, the output resets;

Accumulation type





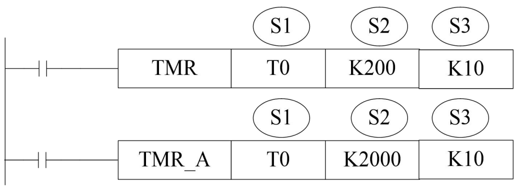
If X0 is ON, HT0 accumulates the 10ms pulse based on the current value. When the accumulation value reaches the set value K2000, the timer outputactivates.

If X0 is suddenly OFF during timer working, the timer value will be retentive. Then X0 is ON again, the timer will continue working.

When X2 is ON, the timer and output will be reset.

Appoint the set value

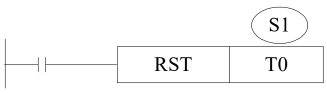
1. Instruction format



(Not accumulation)

(Accumulation)

Reset the timer and output:



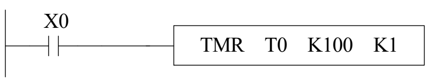
S1: timer (T0, HT10)

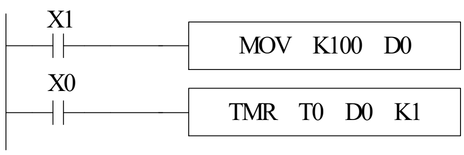
S2: set time (such as K100)

S3: time unit (K1—1ms, K10—10ms, K100—100ms)

Power-off not retentive, not accumulation

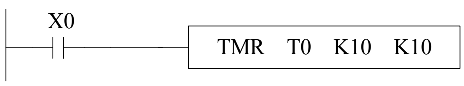
(1) Time unit is 1ms, set time is K100, the real time is 1ms *100=0.1s

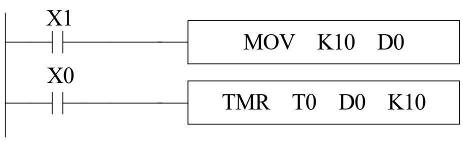




Set value is constant K　　　　　　set value is register D

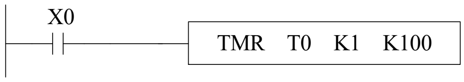
(2) Time unit is 10ms, set time is K10, the real time is 10ms*10=0.1s

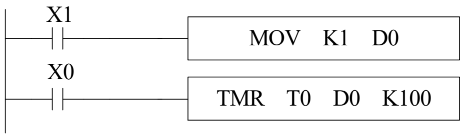




Set value is constant K　　　　　　set value is register D

(3) Time unit is 100ms, set time is K1, the real time is 100ms*1=0.1s

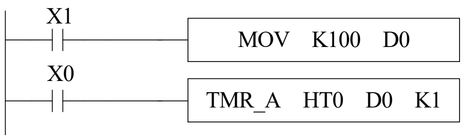




Set value is constant K set value is register D

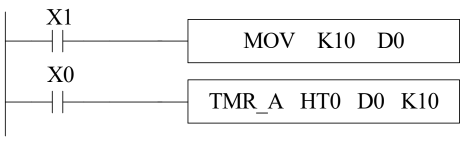
Power-off retentive, accumulation

(1) Time unit is 1ms, set time is K100, the real time is 1ms *100=0.1s



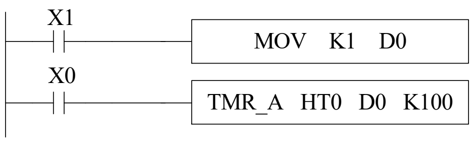
Set value is constant K set value is register D

(2) Time unit is 10ms, set time is K10, the real time is 10ms*10=0.1s



Set value is constant K set value is register D

(3) Time unit is 100ms, set time is K1, the real time is 100ms*1=0.1s



Set value is constant K set value is register D

Notes

(1) The timer has cumulative, non-cumulative, 1ms, 10ms and 100ms, so it can be distinguished by instructions; that is to say, the same timer can be used as either cumulative or non-cumulative, and its time base unit is also specified by instructions as 1ms, 10ms or 100ms.

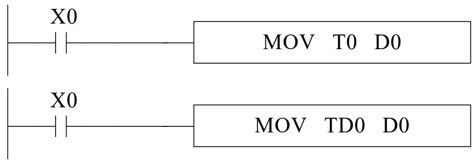
(2) The third parameter of instruction can only be based on K1, K10 and K100. Please do not write other values or registers besides these three parameters. Otherwise, although the program can be written into the programming software and downloaded to the PLC, the timing instruction will not be executed.

(3) The setting range of constant K and the actual setting value of timer are shown in the following table:

Timer	K range	Actual value
1ms timer	1~32,767	0.001~32.767s
10ms timer		0.01~327.67s
100ms timer		0.1~3276.7s

Time value

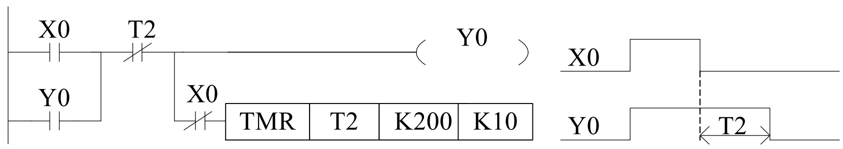
The time value is stored in register TD. The working mode of timer T0~T575 and HT0~HT95 are 16-bits linear increasing. The time range is from 0 to 32767. When the time value in TD reaches 32767, the timer will stop timing and keep the status.



The two instructions are the same. In the first instruction, T0 is seemed to TD0.

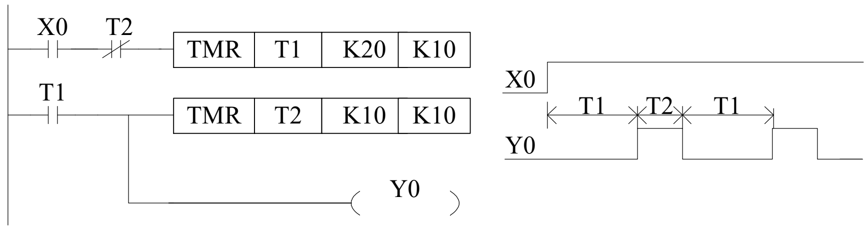
Application

Output delay



X0 is ON, output Y0. X0 changes from ON to OFF, delay 2s then cut off Y0.

Twinkle



X0 is ON, Y0 begin to twinkle. T1 is Y0-OFF time; T2 is Y0-ON time.

Note: The range of soft components mentioned above is the valid range of PLC in the X-NET communication mode. In the MODBUS communication mode, some relays can not read and write. The specific usable range is shown in chapter 6-2-3.

2-8 Counter (C, HC, HSC)

Number list

The counter addresses of XD/XL series PLC are in decimal; please see the following table for details:

Series	Name	Range		
		Normal	Power-off holding	High speed counter
XD1/XD2/XD3/XD3E	C HC HSC	C0~C575	HC0~HC95	HSC0~HSC31
XD5/XDM/XDC/XD5E/XDME		C0~C4999	HC0~HC1999	HSC0~HSC39
XDH		C0~C19999	HC0~HC1999	HSC0~HSC39
XL1/XL3		C0~C575	HC0~HC95	HSC0~HSC31
XL5/XL5E/XLME/XL5H		C0~C4999	HC0~HC1999	HSC0~HSC39
XLH/XL5N		C0~C19999	HC0~HC1999	HSC0~HSC39

The counter range:

Counter type	Explanation
16/32 bits up/down counter	C0~C575 HC0~HC95 (32-bits counter occupies two registers, the counter address must be even number)
High speed counter	HSC0~HSC30 (HSC0,HSC2...HSC30) (each counter occupies two registers, the counter address must be even number)

1: Please refer to chapter 5 for details of high speed counter.
2: XD/XL series counters can be 16 or32 bits count up/down mode. The mode is appointed by the instruction. Which means the same counter can be used as 16-bit or 32-bit. The increment/subtraction counting mode is also specified by the instruction mode.

Counter features

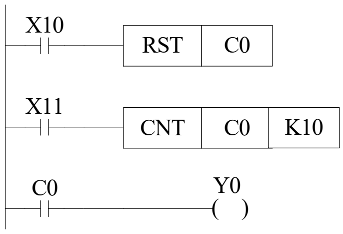
Item	16-bit counter	32-bit counter
Count direction	Count down/up	Count up/down
Set value	-32,768~32,767	-2,147,483,648~+2,147,483,647
Set value type	Constant K or register	Constant K or a couple of registers
Count value	The value will not change when reaching the max or min value	The value will not change when reaching the max or min value
Output	Keep the state for count up	Reset for count down
Reset	Run RST instruction, the counter and output will be reset	
Present count value register	16-bit	32-bit

Function

The soft component will appoint the type of counter: common counter or power-off retentive counter.

16-bit common counter and power-off retentive counter

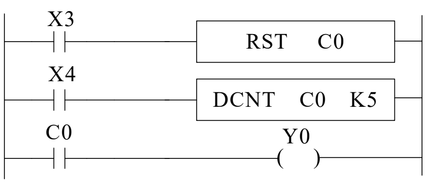
The set value range of 16-bit count-up counter is K1~K32,767(decimal). K0 and K1 have the same function. They mean the counter output will act at the first counting.
If the PLC power supply is cut off, common counter value will be reset. The power-off retentive counter value will be kept.

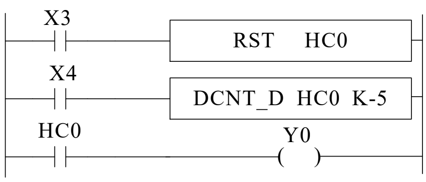


The counter C0 increases one when the X11 drives once. When C0 value reaches 10, the output acts. Then X11 drives again, C0 will continue increase one.
If X10 is ON, the C0 and output will be reset.
The counter set value can be constant K or register. For example, if D10 is 123, the set value is equal to K123.

32-bit common counter and power-off retentive counter

The set value range of 32-bit count-up/down counter is K+2,147,483,648~K-2,147,483,647 (decimal). The count direction is set through instruction.





Common count up counter power-off retentive count down counter

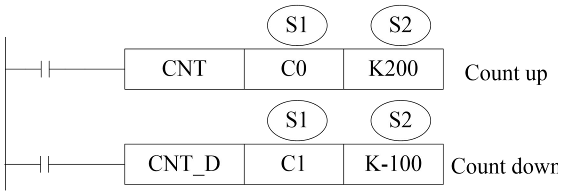
If X3 is ON, the counter and output will be reset.
For power-off retentive counter, the present counter value, output state will be kept after power supply is off.
32-bit counter can be seemed to 32-bit register.

Counter set value

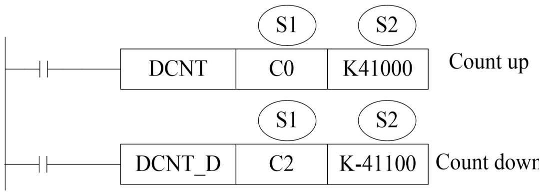
The set value contains two conditions: 16-bit and 32-bit. The counter types include common counter (C) and power-off retentive counter (HC).

Count instruction:

16-bit counter:

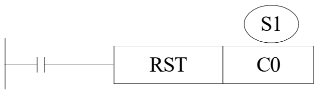


32-bit counter:

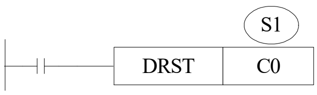


Reset instruction:

16-bit counter:



32-bit counter:



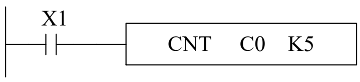
S1: counter (such as C0, HC10)

S2: counter set value (such as K100)

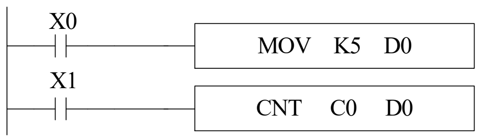
The counter is different from XC series. They don’t have 16-bit and 32-bit type. The type is set through instruction.

16-bit counter (common, count up)

《set value is constant K》

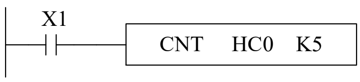


《set value is register 》

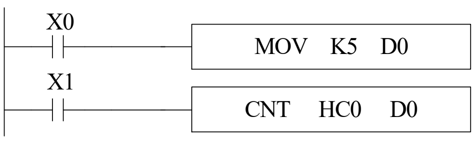


16-bit counter (power-off retentive, count up)

《set value is constant K》

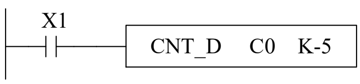


《set value is register 》

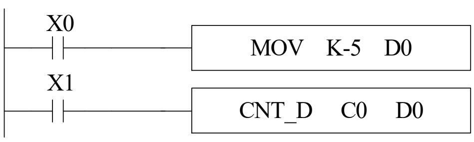


16-bit counter (common, count down)

《set value is constant K》

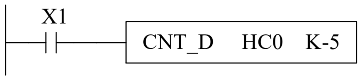


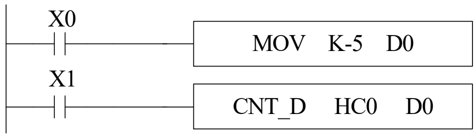
《set value is register 》



16-bit counter (power-off retentive, count down)

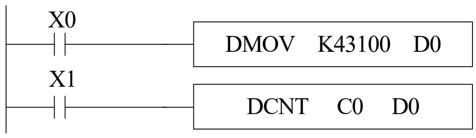
《set value is constant K》 《set value is register 》





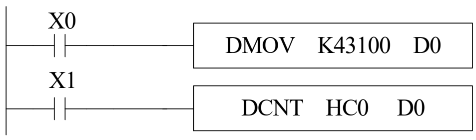
32-bit counter (common, count up)

《set value is constant K》 《set value is register 》



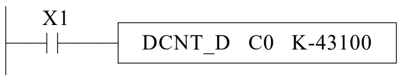
32-bit counter (power-off retentive, count up)

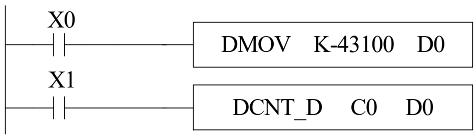
《set value is constant K》 《set value is register 》



32-bit counter (common, count down)

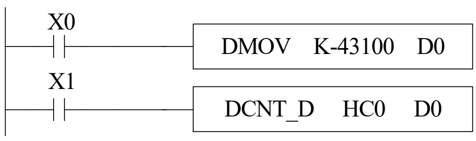
《set value is constant K》 《set value is register 》





32-bit counter (power-off retentive, count down)

《set value is constant K》 《set value is register 》



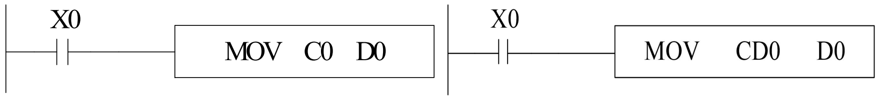
Note: The setting range and actual setting value of constant K are shown in the following table:

Counter	K setting range	Actual setting range
16-bit counter	1~32,767	1~32,767
32-bit counter	1~2,147,483,647	1~2,147,483,647

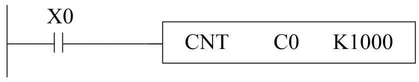
Count value

The counter counting mode is 16-bit linear incremental mode (0~K32,767). When the counter's count value CD reaches the maximum value K32,767, the counter will stop counting and the state of the counter will remain unchanged.

The counter counting mode is a 16-bit linear decreasing mode (-32768-0). When the counter counting value CD decreases to the minimum value K-32, 768 will stop counting and the state of the counter remains unchanged.
The counter counting mode is 32-bit linear increase/decrease mode (-2,147,483,648~+2,147,483,647). When the counter counting value increases to the maximum value K2,147,483,647, it will become K-2,147,483,648. When the counter counting value decreases to the minimum value K-2,147,483,648 will become K2,147,483,647, the ON/OFF state of the counter will also change with the change of the count value.



The above two instructions are equivalent. In the left instruction, C0 is processed as a register, while in the right instruction, CD0 is a data register corresponding to the timer C0. CD and C are one-to-one correspondences.



The highest frequency that this instruction can count is related to the selection of filter parameters and the scanning period of PLC. A high-speed counter is recommended when the input frequency exceeds 25Hz. High-number counter must use HSC0-HSC30 and corresponding hardware wiring.

High-speed counter, when SM0 is on, HSC0 counts the pulse signal of input terminal X0. High-speed counter is not affected by the response lag time of input filter and cycle scan time. Therefore, higher frequency input pulses can be processed. Refer to the details in chapter 5.

Note: The range of soft components mentioned above is the valid range of PLC in the X-NET communication mode. In the MODBUS communication mode, some relays can not read and write. The specific usable range is shown in chapter 6-2-3.

2-9 Data register (D, HD, SD, HSD)

Address list

The data register of XD/XL series PLC is in decimal format. Please see the following table:

Series	Name	Range			
		Normal	Power-off holding	Special	Special power-off holding
XD1/XD2/XD3/XD3E	D HD SD HSD	D0~D7999	HD0~HD999	SD0~SD2047	HSD0~HSD499
XD5/XDM/XDC/XD5E/ XDME		D0~D69999	HD0~HD24999	SD0~SD4999	HSD0~HSD1023
XDH(except XDH-60A32)		D0~D499999	HD0~HD49999	SD0~SD49999	HSD0~HSD49999
XDH-60A32		D0~D999999	HD0~HD99999	SD0~SD49999	HSD0~HSD49999
XL1/XL3		D0~D7999	HD0~HD999	SD0~SD2047	HSD0~HSD499
XL5/XL5E/XLME/XL5H		D0~D69999	HD0~HD24999	SD0~SD4999	HSD0~HSD1023
XL5N		D0~D69999	HD0~HD24999	SD0~SD4999	HSD0~HSD1999
XLH-24A16		D0~D499999	HD0~HD49999	SD0~SD49999	HSD0~HSD49999
XLH-30A32		D0~D999999	HD0~HD99999	SD0~SD49999	HSD0~HSD49999

Note: For XD5 firmware version V3.4.6 and above, data register D ranges from D0 to D69999;
XD5 firmware version below V3.4.6, and data register D ranges from D0 to D59999.

Structure

Data register is used to store data; it includes16 bits(the higheset bit is sign bit) and32 bits. (32 bits contains two registers, the highest bit is sign bit)

16 bits

16-bits register range is -32,768 ~ +32,767

Read and write the register data through instruction or other device such as HMI.

D0 16-bits

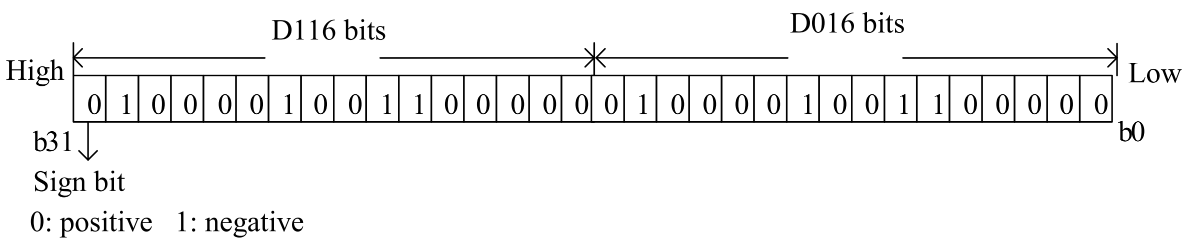
b15															b0
0	1	0	0	0	0	1	0	0	1	1	0	0	0	0	0

Sign bit

0: positive 1: negative

32 bits

32 bits value is consisted of two continuous registers. The range is -2147483648 ~ 2147483647. For example: (D1 D0) D1 is high16 bits, D0 is low16 bits.
For32 bits register, if the low 16-bits are appointed, such as D0, then D1 will be the high16 bits automatically. The address of low 16-bits register must be even number.



Function

- Normal type

When write a new value in the register, the former value will be covered.
When PLC changes from RUN to STOP or STOP to RUN, the value in the register will be cleared.

- Retentive type

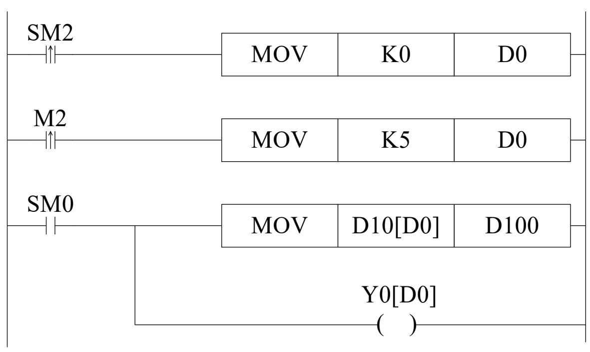
When PLC changes from RUN to STOP or power off, the value in the register will be retained.
The retentive register range cannot be changed.

- Special type

Special register is used to set special data, or occupied by the system.
Some special registers are initialized when PLC is power on.
Please refer to the appendix for the special register address and function.

- Used as offset (indirect appoint)

Data register can be used as offset of soft element.
Format : Dn[Dm], Xn[Dm], Yn[Dm], Mn[Dm].
Word offset: DXn[Dm] means DX[n+Dm].
The offset value only can be set as D register.

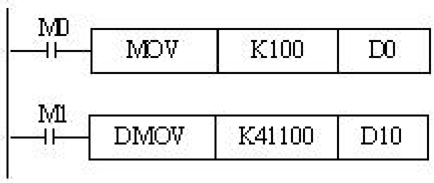


When D0=0, D100=D10, Y0 is ON;

When M2 is from OFF→ON, D0=5, D100=D15, Y5 is ON.
D10[D0]=D[10+D0], Y0[D0]=Y[0+D0].

Example

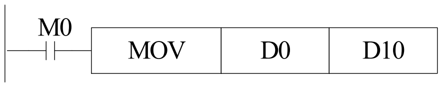
Data register D can deal with many kinds of data.
Data storage



When M0 is ON, write 100 into D0.(16 bits value)

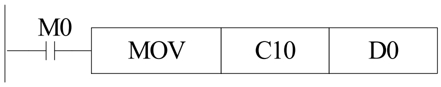
When M1 is ON, write 41100 into D11,D10 (32bits value)

Data transfer



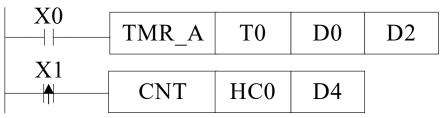
When M0 is ON, transfer the value of D10 to D0

Read the timer and counter



When M0 is ON, move the value of C10 to D0.

As the set value of timer and counter



When X0 is ON, T10 starts to work, T0 will set ON when D0 value is equal to timer value, time unit is D2.

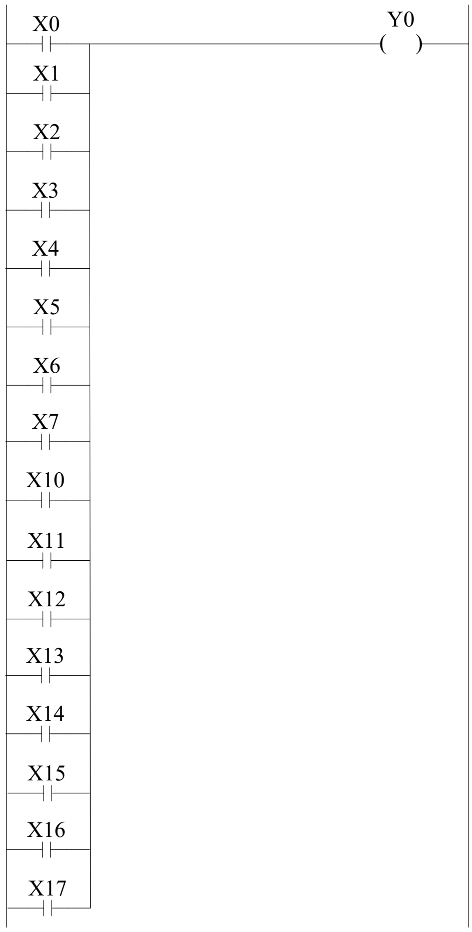
X1 is ON, HC0 starts to work, HC0 will set ON when D4 value is equal to counter value.

Note: The range of soft components m
communication mode. In the MODBUS communication mode, some relays can not read and write. The specific usable range is shown in chapter 6-2-3.

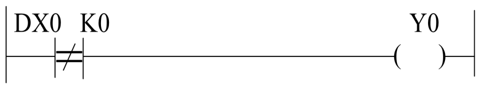
2-9-1 Word consist of bits

One of the coils from X0 to X17 is ON, Y0 will be ON.

Programming method one:



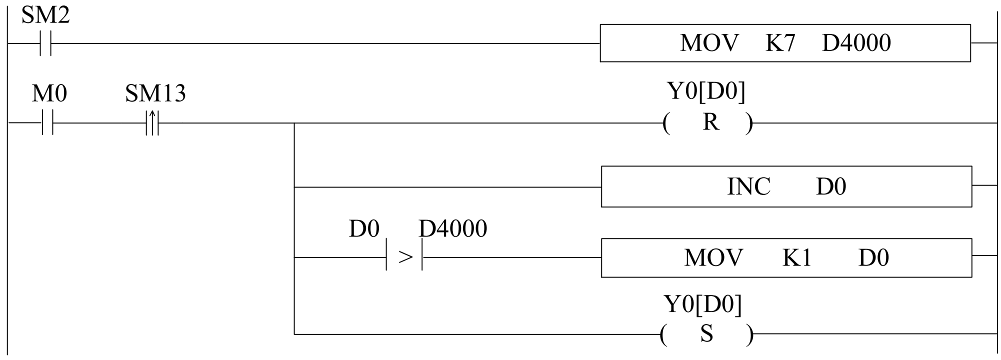
Programming method two: (application of word consists of bits)



2-9-2 Offset application

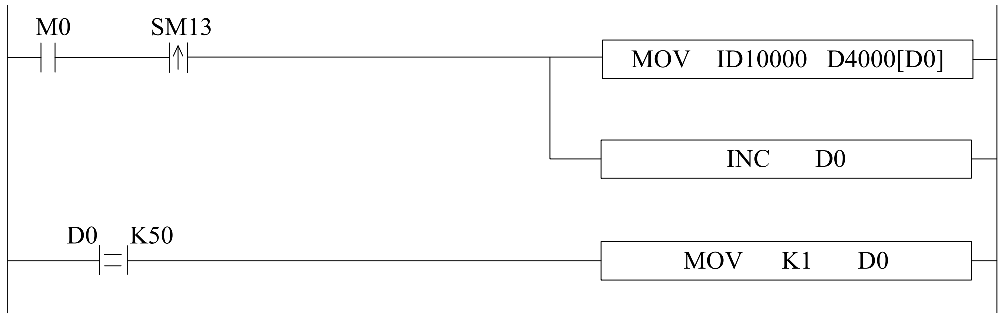
Application 1:

When M0 is ON, the output from Y1 to Y7 will be ON one by one. D0 is offset address. If there are many output points, M can replace Y.



Application 2:

When M0 is ON, read the ID10000 value every second and store in the register starting from D4000 (amounts is 50 registers). D0 is offset address.



2-10 Flash register (FD, SFD, FS)

The FLASH registers of XD/XL series PLC are all addressed in decimal system. The serial numbers are shown in the corresponding table.

Series	Name	Range		
		FLASH user data register	FLASH system data register	Password read protection FLASH register
XD1/XD2/XD3/XD3E	FD SFD FS	FD0~FD5119	SFD0~SFD1999	FS0~FS47
XD5/XDM/XDC/XD5E/XDME		FD0~FD8191	SFD0~SFD5999	FS0~FS47
XDH(except XDH-60T4/30A16)		FD0~FD65535	SFD0~SFD49999	FS0~FS47
XDH-60T4/30A16		FD0~FD65535	SFD0~SFD65487	FS0~FS47
XL1/XL3		FD0~FD5119	SFD0~SFD1999	FS0~FS47

XL5/XL5E/XLME/XL5H		FD0~FD8191	SFD0~SFD5999	FS0~FS47
XLH/XL5N		FD0~FD65535	SFD0~SFD65487	FS0~FS47

Function

- FLASH User Data Register (FD)

Used to store important data of users, can be maintained when the power is off.
This storage area can remember data even if the battery is powered down, so it can be used to store important process parameters.

- FLASH System Data Register (SFD)

Used to store system parameters and be able to maintain the data when power off.
The storage area is a system parameter block, and users can not modify it at will.

- Password Read Protection FLASH Register (FS)

A part of the FlashROM register is used to store data soft components, which are represented by the symbol FS. The values in the FS register can be written but can not be read, so they can be used to protect the intellectual property rights of users.
The value of the soft element can be set arbitrarily in the FS register, but the value of the register can not be read (always returned to 0); and it can not be compared with the register in the host computer software, only with the constant, so the actual value of the register can not be read.
This storage area can remember data even if the battery is powered down, so it can be used to store important process parameters.

Note:
(1) When using MOV instruction to transmit data to FD, SFD and FS, only the rising edge is valid, even if the driving condition is normally open/closed coil, the instruction is executed only once.
(2) Flash registers can be written about 1,000,000 times, and each write is erased for the whole Flash registers, which is time-consuming. Frequent writing will cause permanent damage to Flash registers, so it is not recommended that users write frequently. Do not use oscillating coil (e.g. SM11) as driving condition.
(3) When data is transmitted to the same Flash register several times, if the value in the source register does not change from the previous transmission, the transmission instruction will not be executed even if the driving condition is established again. For example, if the value in D0 is transmitted to FD100, the value in D0 is 300 when the transmission instruction is executed for the first time; if the driving condition is established for the second time, the transmission instruction is not executed if the value in D0 is still 300.
(4) In order to prevent the interference of burr signal when transmitting data to Flash registers, it is not recommended to use coils such as SM0 and SM2 as direct driving conditions. It is suggested that the transmission instructions be executed after the PLC power-on for a period of time.
(5) The FS register can only be modified by setting the initial value of the software component.

2-11 Variables

XDPPro version 3.7.16 and above supports custom variables, and users can directly make program using variable names by defining global variables. When defining global variable names, the following rules need to be followed:

- Only uppercase and lowercase letters are allowed to start;
- Only uppercase and lowercase letters, numbers, and "_" (underline) are allowed;
- Repeated naming is not allowed;
- It is not allowed to duplicate the name of a classic PLC instruction;
- It is not allowed to duplicate data type names;
- It is not allowed to have duplicate names with existing software components in the PLC.

(1) Data type

Supported basic types:

Data type	Type description	Length/bit	Data type	Type description	Length/bit
BIT	Bit	1	LINT	Long integer	64
BOOL	Boolean	8	ULINT	Unsigned long integer	64
SINT	Short integer	8	REAL	real number	32
USINT	Unsigned short integer	8	LREAL	Long real number	64
INT	Integer	16	BYTE	Bit string	8
UINT	Unsigned integer	16	WORD	Bit string	16
DINT	Double integer	32	DWORD	Bit string	32
UDINT	Unsigned double integer	32	LWORD	Bit string	64

Supported copying data type:

- Support user-defined structure types;
- Support function block types;
- Support pointer types;
- Supports array types (currently only supports one-dimensional arrays).

Other POU functions please refer to POU function manual.

2-12 Constant

Data process

XD/XL series PLC has the following 5 number systems.

- DEC: DECIMAL NUMBER

The preset number of counter and timer (constant K)
The number of Auxiliary relay M, HM; timer T, HT; counter C, HC; state S, HS; register D, HD.
Set as the operand value and action of applied instruction (constant K)

- HEX: HEXADECIMAL NUMBER

Set as the operand value and action of applied instruction (constant H)

- BIN: BINARY NUMBER

Inside the PLC, all the numbers will be processed in binary. But when monitoring on the device, all the binary will be transformed into HEX or DEC.

- OCT: OCTAL NUMBER

XD/XL series PLC I/O relays are in octal. Such as [X0-7, X10-17,….X70-77].

- BCD: BINARY CODE DECIMAL

BCD uses 4 bits binary number to represent decimal number 0-9. BCD can be used in 7 segments LED and BCD output digital switch

- Other numbers (float number)

XD/XL series PLC can calculate high precision float numbers. It is calculated in binary numbers, and display in decimal numbers.

Display

PLC program should use K, H to process values. K means decimal numbers, H means hex numbers. Please note the PLC input/output relay use octal address.

- Constant K

K is used to display decimal numbers. K10 means decimal number 10. It is used to set timer and counter value, operand value of applied instruction.

- Constant H

H is used to display hex numbers. HA means decimal number 10. It is used to set operand value of applied instruction.

- Constant B

B is used to display binary numbers. B10 means decimal number 2. It is used to set operand value of applied instruction.

2-13 Programming principle

Sign P and I

P is the program sign for condition and subprogram jump.

I is the program sign for interruption (external interruption, timer interruption, high speed counter interruption, precise time interruption…).

P and I addresses are in decimal. Please refer to the following table:

Series	Sign	Address
XD, XL	P	P0~P9999

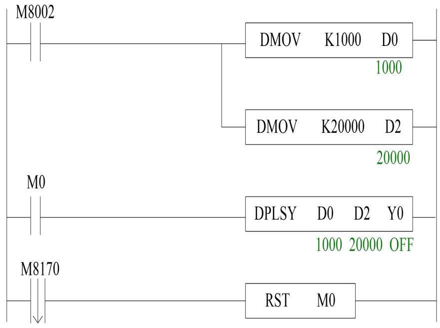
Model	Name	Range			
		External interruption			Timer interruption
		Input terminal	Rising interruption	Falling interruption	
XD/XL series 16 points	I	X2	I0000	I0001	There are 20 timer interruptions. From I40** to I59**. "**" means the timeof timer interruption, the unit is ms.
		X3	I0100	I0101	
		X4	I0200	I0201	
		X5	I0300	I0301	
		X6	I0400	I0401	
		X7	I0500	I0501	

Model	Name	Range			
		External interruption			Timer interruption
		Input terminal	Rising interruption	Falling interruption	
XD/XL series 24-64 points	I	X2	I0000	I0001	There are 20 timer interruptions. From I40** to I59**. "**" means the timeof timer interruption, the unit is ms.
		X3	I0100	I0101	
		X4	I0200	I0201	
		X5	I0300	I0301	
		X6	I0400	I0401	
		X7	I0500	I0501	
		X10	I0600	I0601	
		X11	I0700	I0701	
		X12	I0800	I0801	
		X13	I0900	I0901	

Sign P

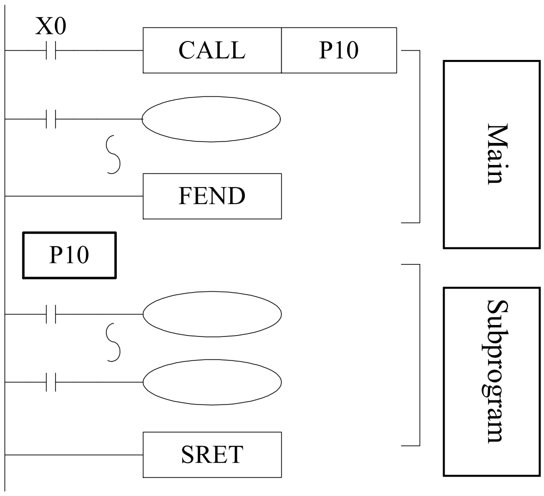
P is usually used in flow; it is used together with CJ (condition jump), CALL (call subprogram), etc.

Condition Jump CJ



If coil X0 is ON, jump to the programafter P1;
If the coil X0 is not ON, do not execute jump action, but run the original program;

Call the subprogram (CALL)



If X0 is ON, jump to the subprogram
If the coil is not ON, run the original program;
After executing the subprogram, return to the main program;

The subprogram will start from Pn and finish with SRET. CALL Pn is used to call the subprogram. n is a integer in the range of 0 to 9999.

Sign I

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

- External interruption

Accept the input signal from the special input terminals, not affected by the scan cycle. Activate the input signal, execute 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 isdifferent from PLC's operation cycle;

- Action sequence of input/output relays and response delay

Input

Before PLC executing the program, read all the input terminal's ON/OFF status 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 next scan cycle, the changes will be read.

Output

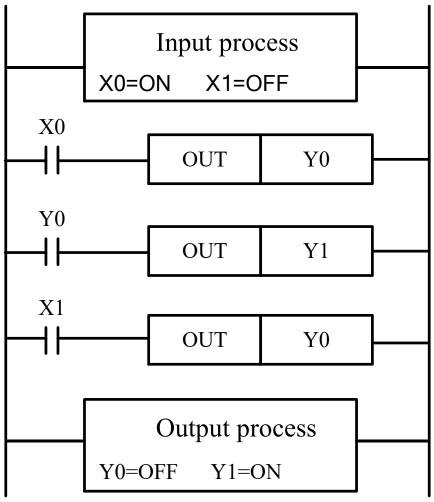
Once all the instructions end, transfers the ON/OFF status of output Y image area to the output lock memory area. This will be the actual output of the PLC. The output contactors will act according to the device's response delay time.

When use batch input/output mode, the drive time and operation cycle of input filter and output device will also show response delay.

- Not accept narrow input pulse signal

PLC's input ON/OFF time should be longer than its loop time. If consider 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 processed. But, this condition could be improved when use PLC's special function and applied instructions (such as high speed count, input interruption, input filter adjustment).

- Dual output(Dual coils)action



As shown in the left map, please consider the case of using the same coil Y0 at many positions:
E.g.X0=ON, X1=OFF
The first Y0: X0 is ON, its image area is ON, output Y1 is also ON.
The second Y0: as input X1 is OFF, the image area is OFF.
So, the actual output is: Y0=OFF, Y1= ON.

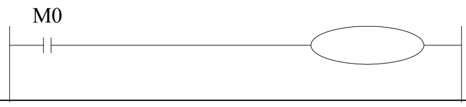
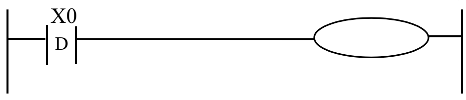
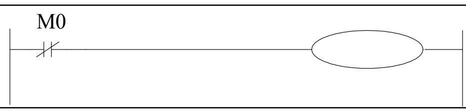
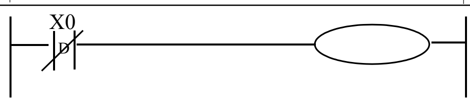
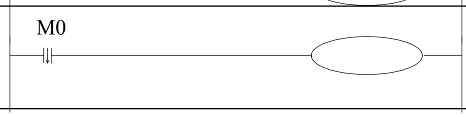
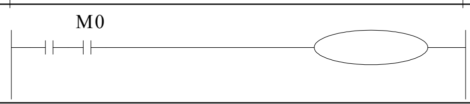
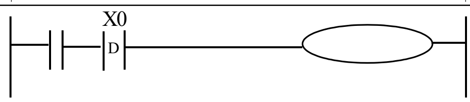
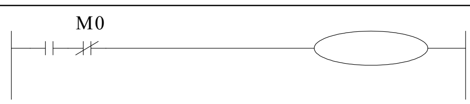
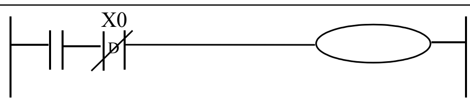
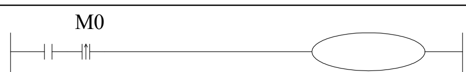
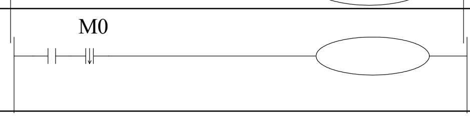
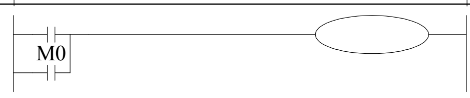
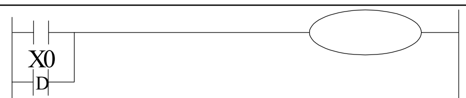
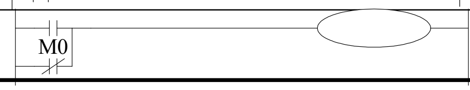
When executing dual output (use dual coil), the after one is act in priority.

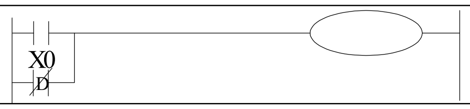
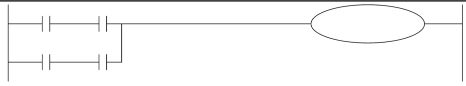
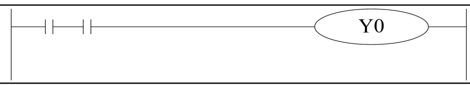
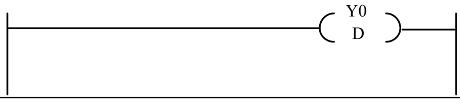
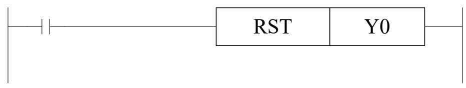
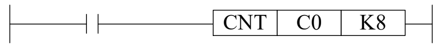
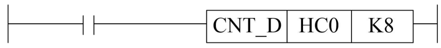
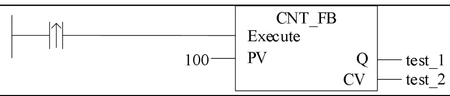
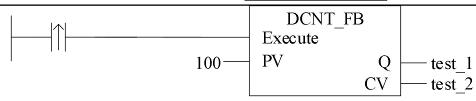
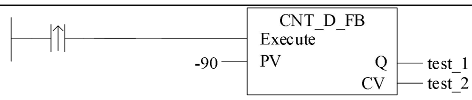
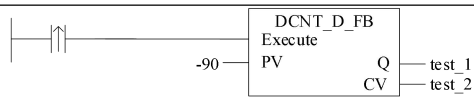
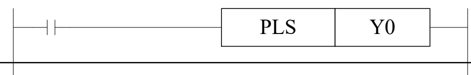
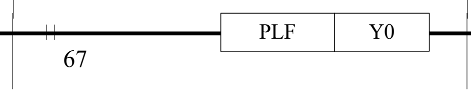
3 Basic Program Instructions

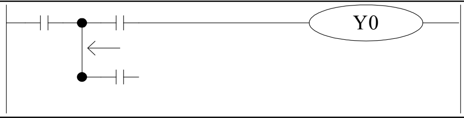
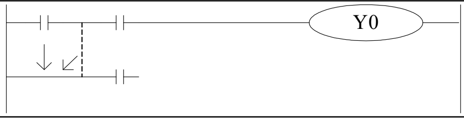
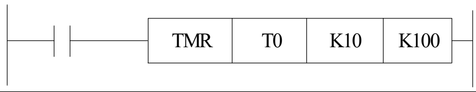
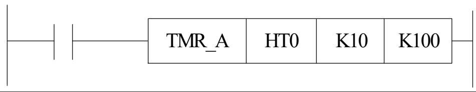
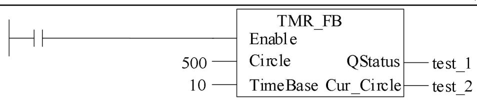
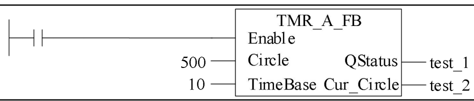
This chapter introduces the basic instructions and their functions.

3-1 Basic Instructions List

XD, XL series support all the basic instructions:

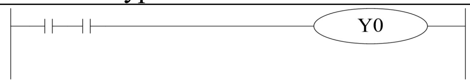
Mnemonic	Function	Format and Device	**Chapter**
LD	Initial logical operation contact type NO (normally open)		3-2
LDD	Read the status from the contact directly		3-6
LDI	Initial logical operation contact type NC (normally closed)		3-2
LDDI	Read the normally closed contact directly		3-6
LDP	Initial logical operation-Rising edge pulse		3-5
LDF	Initial logical operation-Falling /trailing edge pulse		3-5
AND	Serial connection of NO (normally open) contacts		3-3
ANDD	Read the status from the contact directly		3-6
ANI	Serial connection of NC (normally closed) contacts		3-3
ANDDI	Read the normally closed contact directly		3-6
ANDP	Serial connection of rising edge pulse		3-5
ANDF	Serial connection of falling/trailing edge pulse		3-5
OR	Parallel connection of NO (normally open) contacts		3-4
ORD	Read the status from the contact directly		3-6
ORI	Parallel connection of NC (normally closed)		3-4

	contacts		
ORDI	Read the normally closed contact directly		3-6
ORP	Parallel connection of rising edge pulse		3-5
ORF	Parallel connection of falling/trailing edge pulse		3-5
ANB	Serial connection of multiply parallel circuits		3-8
ORB	Parallel connection of multiply parallel circuits		3-7
OUT	Final logic operation type coil drive		3-2
OUTD	Output to the contact directly		3-6
SET	Set a bit device permanently ON		3-12
RST	Reset a bit device permanently OFF		3-12
CNT	16-bit non-power-off retentive incremental count		3-13
CNT_D	16-bit power-off retentive decremented count		3-13
DCNT	32-bit non-power-off retentive incremental count		3-13
DCNT_D	32-bit power-off retentive decremented count		3-13
CNT_FB			3-13
DCNT_FB			3-13
CNT_D_FB			3-13
DCNT_D_FB			3-13
PLS	Turn on a scan cycle when rising edge		3-11
PLF	Turn on a scan cycle		3-11

	when falling edge		
MCS	Connect the public serial contacts		3-9
MCR	Clear the public serial contacts		3-9
ALT	The status of the assigned device is inverted on every operation of the instruction		3-10
TMR	Non-power-off holding timer		3-14
TMR_A	Power-off holding timer		3-14
TMR_FB			3-14
TMR_A_FB			3-14
END	Force the current program scan to end		3-15
GROUP	Group		3-15
GROUPE	Group End		3-16

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

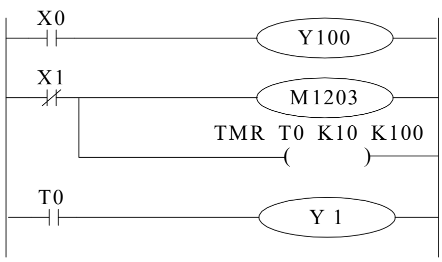
Mnemonic and Function

Mnemonic	Function	Format and Operands
LD (positive)	Initial logic operation contact type NO (Normally Open)	 Operands: X,Y,M,HM,SM,S,HS,T,HT,C,HC,Dn.m Variable type: BIT/BOOL
LDI (negative)	Initial logic operation contact type NC (Normally Closed)	 Devices: X,Y,M,HM,SM,S,HS,T,HT,C,HC,Dn.m Variable type: BIT/BOOL
OUT (OUT)	Final logic operation type drive coil	 Operands: X,Y,M,HM,SM,S,HS,T,HT,C,HC,Dn.m Variable type: BIT/BOOL

Statement

- Connect the LD and LDI instructions directly to the left bus bar. It can work with ANB and be used at the branch start.
- OUT instruction can drive the output relays, auxiliary relays, status, timers, and counters. But this instruction can’t be used for the input relays.
- The LD, LDI, and OUT commands do not support operating on the actual output points on the PLC, and can only control virtual input points as intermediate relays.

Program




```
LD    X0
OUT   Y100
LDI   X1
OUT   M1203
TMR   T0        K10  K100
LD    T0
OUT   Y1
```

3-3 [AND] , [ANI]

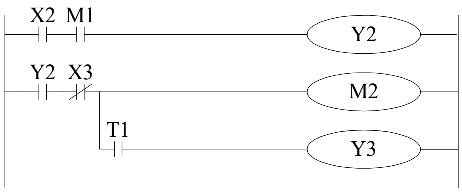
Mnemonic and Function

Mnemonic	Function	Format and Operands
AND (and)	Normal open contactor in series	 Operand: X,Y,M,HM,SM,S,HS,T,HT,C,HC,Dn.m Variable type: BIT/BOOL
ANI (and reverse)	Normal close contactor in series	 Operand: X,Y,M,HM,SM,S,HS,T,HT,C,HC,Dn.m Variable type: BIT/BOOL

Statements

- Use AND and ANI to connect the contactors in series. There is no limit for contactors in series. They can be used for many times.
- Use OUT instruction through other coil is called “follow-on” output (For an example see the program below: OUT M2 and OUT Y3). Follow-on outputcanrepeat as long as the output order is correct. There’s no limit for the serial connected contactors and follow-on output times.

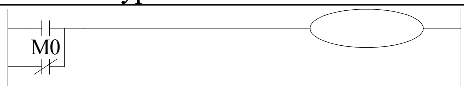
Program



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

3-4 [OR] , [ORI]

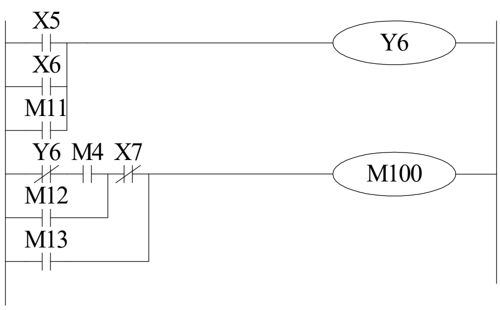
Mnemonic and Function

Mnemonic	Function	Format and Operands
OR (OR)	Parallel connection of NO (Normally Open) contactors	 Operand: X,Y,M,HM,SM,S,HS,T,HT,C,HC,Dn.m Variable type: BIT/BOOL
ORI (OR reverse)	Parallel connection of NC (Normally Closed) contactors	 Operand: X,Y,M,HM,SM,S,HS,T,HT,C,HC,Dn.m Variable type: BIT/BOOL

Statements

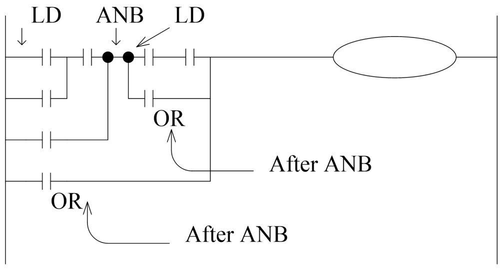
- Use the OR and ORI instructions for parallel connection of contactors. To connect a block that contains more than one contactor connected in series to another circuit block in parallel, use ORB instruction, which will be described later;
- OR and ORI start from the instruction step, parallel connect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

Program




```
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]

3-5-1. Single operand

Mnemonic and Function

Mnemonic	Function	Format and Operands
LDP (LoaD Pulse)	Initial logical operation-Rising edge pulse	 X,Y,M,HM,SM,S,HS,T,HT,C,HC,Dn.m
LDF (LoaD Falling pulse)	Initial logical operation Falling/trailing edge pulse	 X,Y,M,HM,SM,S,HS,T,HT,C,HC,Dn.m
ANDP (AND Pulse)	Serial connection of Rising edge pulse	 X,Y,M,HM,SM,S,HS,T,HT,C,HC,Dn.m
ANDF (AND Falling pulse)	Serial connection of Falling/trailing edge pulse	 X,Y,M,HM,SM,S,HS,T,HT,C,HC,Dn.m
ORP (OR Pulse)	Parallel connection of Rising edge pulse	 X,Y,M,HM,SM,S,HS,T,HT,C,HC,Dn.m
ORF (OR Falling pulse)	Parallel connection of Falling/trailing edge pulse	 X,Y,M,HM,SM,S,HS,T,HT,C,HC,Dn.m

- Edge+BIT type

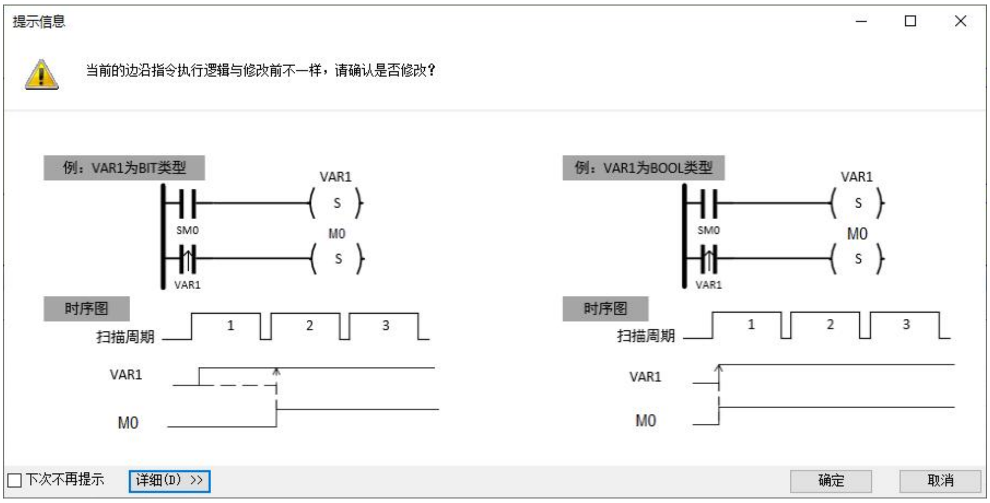
Using range	Supported variable categories	Supported variable type
Main ladder chart	VAR_GLOBAL	BIT type variables, members of BIT arrays
Function - ladder diagram language	VAR_GLOBAL	BIT type variables, members of BIT arrays
Function Block - Ladder Diagram Language	VAR_GLOBAL	BIT type variables, members of BIT arrays

- Edge+BOOL type

Using range	Supported variable categories	Supported variable type
Main ladder chart	VAR_GLOBAL	BOOL type variables, BOOL pointer type variables, members of BOOL arrays
Function - ladder diagram language	VAR_GLOBAL	BOOL type variables, BOOL pointer type variables, members of BOOL arrays
Function Block - Ladder Diagram Language	VAR_GLOBAL, VAR, VAR_TEMP, VAR_INPUT, VAR_OUTPUT, VAR_INOUT	BOOL type variables, BOOL pointer type variables, members of BOOL arrays

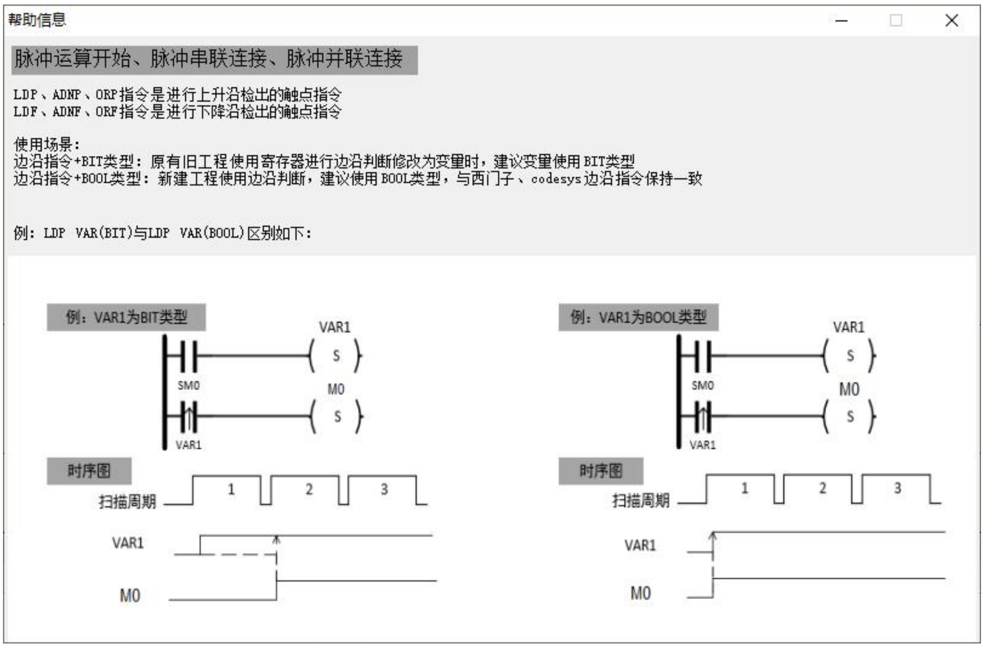
Statements

- LDP, ANDP, and ORP commands are contact commands for detecting rising edges. At the rising edge of specified soft components and BIT types (OFF → ON change), a scanning cycle is connected, and the edge signal lags behind by one scanning cycle; When using BOOL type, the current scanning cycle triggers the edge signal.
- LDF, ANDF, and ORF commands are contact commands for detecting the falling edge. At the soft components and BIT type falling edges (when changing from ON to OFF), a scanning cycle is connected, and the edge signal lags behind by one scanning cycle; When using BOOL type, the current scanning cycle triggers the edge signal.
- Edge+BOOL type: indicated by a red arrow; Edge+soft components/BIT type: indicated by a black arrow.
- When the instruction operand changes from BIT to BOOL or changes in the opposite direction, a pop-up will prompt "The current edge instruction execution logic is different from before modification. Please confirm whether to modify it?"
  Click “details” to show below interface:

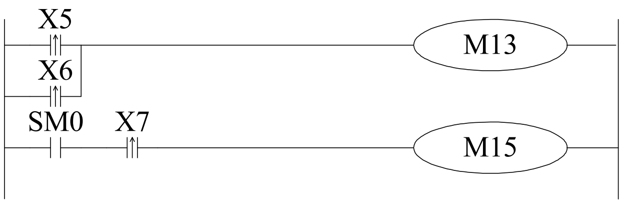


Click the "OK" button to complete the modification, click the "Cancel" button to keep the original instruction display unchanged; If you check the option to no longer prompt next time and click OK, the next time you modify the edge command to a different type, this dialog box will no longer pop up.

- Select the edge command in the ladder diagram, press F1, and a help message box will pop up as shown below:



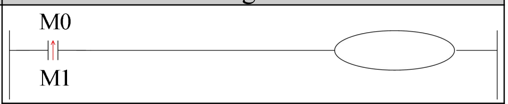
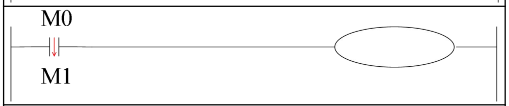
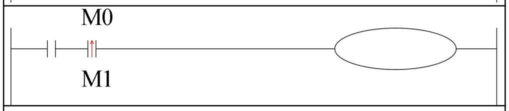
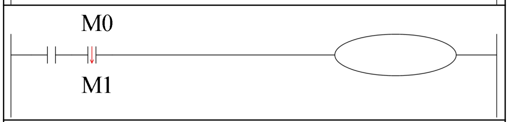
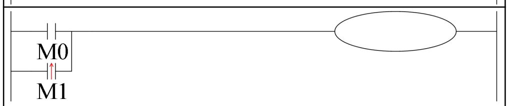
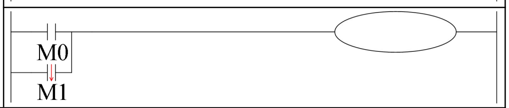
Program



LDP	X5
ORP	X6
OUT	M13
LD	M8000
ANDP	X7
OUT	M15

3-5-2. Double operand

Mnemonic and Function

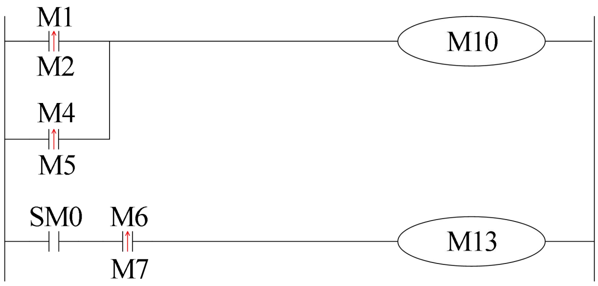
Mnemons, names	Operand S1	Operand S2	Object type	Circuit representation and variable categories
LDP Take the rising edge of the pulse	Indicates the current signal status of the operand	Indicates the signal status of the last scan of operand S1	Register type: X, Y, M, S, T, C, SM, ET, HM, HS, HT, HC, HSC Variable type: BIT/BOOL	
LDF Take the falling edge of the pulse				
ANDP And the pulse rising edge				
ANDF And the pulse falling edge				
ORP Or the pulse rising edge				
ORF Or the pulse falling edge				

Statement

- The instruction will compare operands S1 and S2. If operand 1 is ON and operand 2 is OFF, an rising edge will be generated immediately; If operand 1 is OFF and operand 2 is ON, a falling edge is immediately generated.
- The operand S2 of the edge signal can only be used once at most in the program, otherwise it

will overwrite the memory of that bit. This step will affect edge detection, resulting in the result being no longer unique.

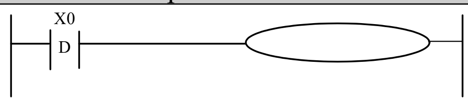
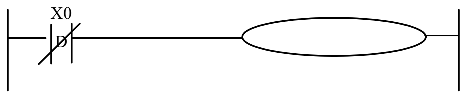
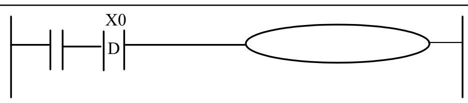
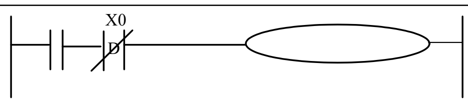
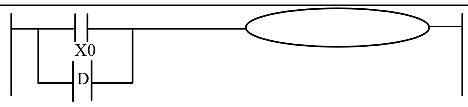
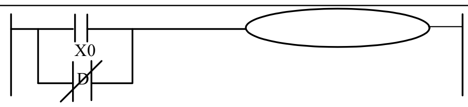
Program

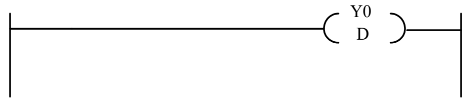


LDP	M1	M2
ORP	M4	M5
OUT	M10	
LD	SM0	
ANDP	M6	M7
OUT	M13	

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

Mnemonic and Function

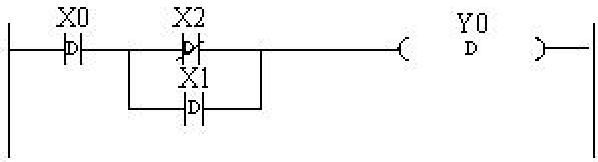
Mnemonic	Function	Format and Operands
LDD	Read the status from the contact directly	 Devices: X
LDDI	Read the normally closed contact directly	 Devices: X
ANDD	Read the status from the contact directly	 Devices: X
ANDDI	Read the normally closed contact directly	 Devices: X
ORD	Read the status from the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

OUTD	Output to the contact directly	 Devices: Y

Statement

The function of LDD, ANDD, ORD instructions are similar to LD, AND, OR; LDDI, ANDDI, ORDI instructions are similar to LDI, ANDI, ORI; but if the operand is X, the LDD, ANDD, ORD commands read the signal from the terminals directly.
OUTD and OUT are output instructions. OUTD will output immediately when the condition is satisfied, needn't wait for the next scan cycle.

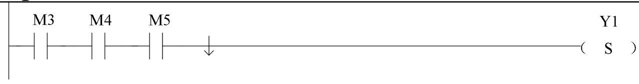
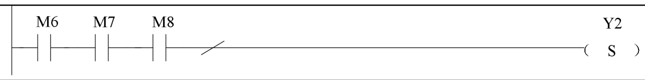
Program




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

3-7 [MEP],[MEF],[INV]

Mnemonic and Function

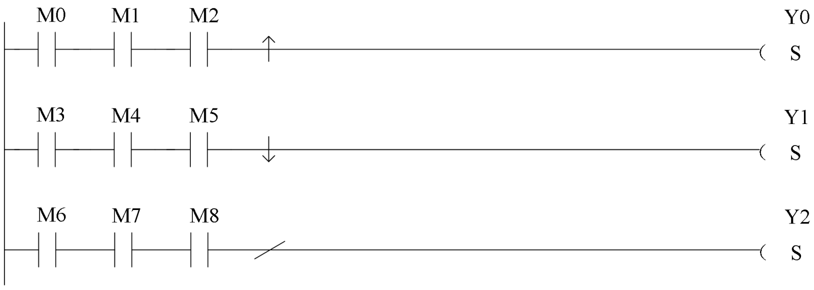
Mnemonic	Function	Format and Devices
MEP	Arithmetic result rising edge pulse	 Operation source: none
MEF	Arithmetic result falling edge pulse	
INV	Negate the operation result	

Statements

- When the operation result before the MEP instruction changes from OFF to ON, an edge signal is output, and the output state is OFF at other times.
- When the operation result before the MEF instruction changes from ON to OFF, an edge signal is output, and the output state is OFF at other times.

- The INV instruction performs a reverse operation on the operation block before this instruction.
- Support pulsing of switch contacts LD, AND, OR (BIT, BOOL, register bits), comparison contacts LD=, DLD=, ELD=, DELD=(=,<>,>,<,<=,>=).
- FB supports MEP, MEF, and INV instructions.
- FC supports INV instructions.
- MEP, MEF, and INV commands cannot be directly connected to the left busbar.
- MEP and MEF instructions cannot appear separately on parallel branches.

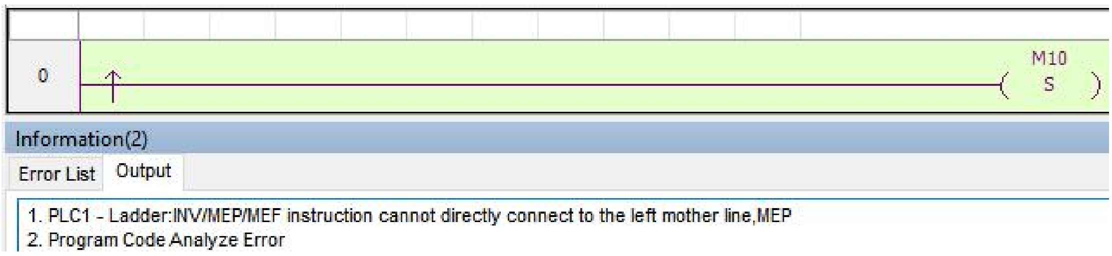
Program



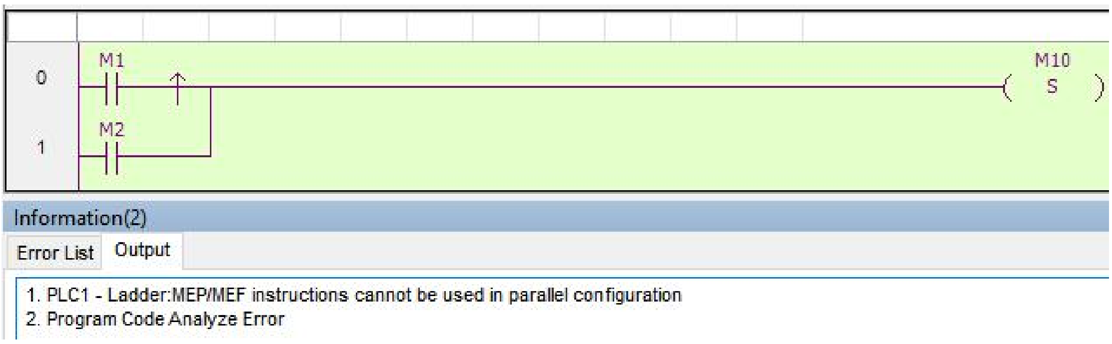
LD	M0
AND	M1
AND	M2
ANDMEP	
SET	Y0
LD	M3
AND	M4
AND	M5
ANDMEF	
SET	Y1
LD	M6
AND	M7
AND	M8
ANDINV	
SET	Y2

Example

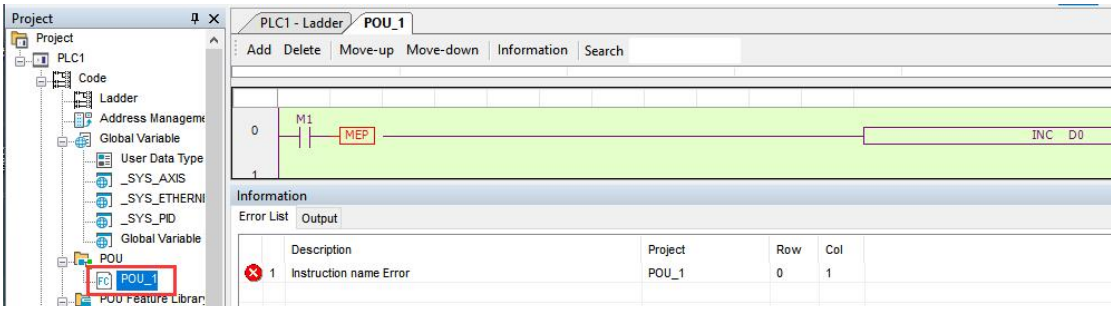
- Error example
  (1) MEP and MEF commands cannot be directly connected to the left busbar. No errors will be reported during instruction editing, and error prompts will appear during compilation:



  (2) MEP and MEF instructions cannot appear on parallel branches. No errors will be reported during instruction editing, and error prompts will appear during compilation:

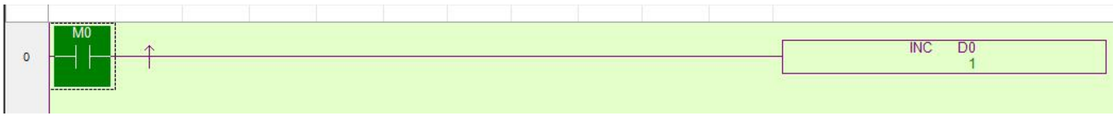


(3) The MEP and MEF instructions cannot be used within FC, and there is no error message when editing instructions. An error message will appear during compilation:



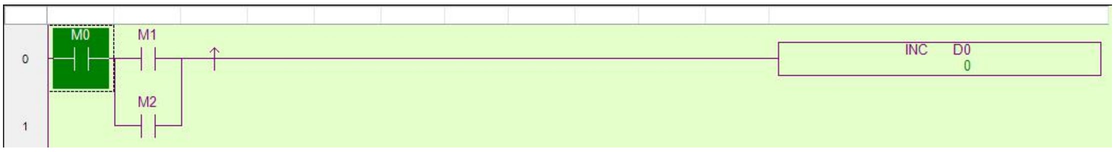
- Using example

(1) Single node pulse: When the M0 state is true, pulse the instruction state with D0=1;

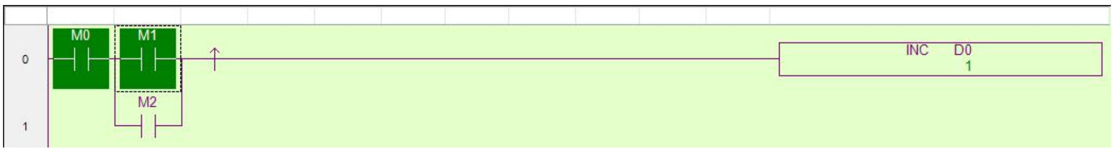


(2) Series parallel hybrid pulse conversion:

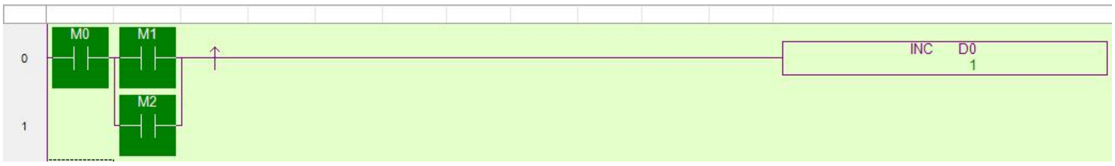
① When M0 is true and M1 and M2 are both false, D0 is 0 and self addition is not performed. M0 status is false, M1 or M2 status is not self added.



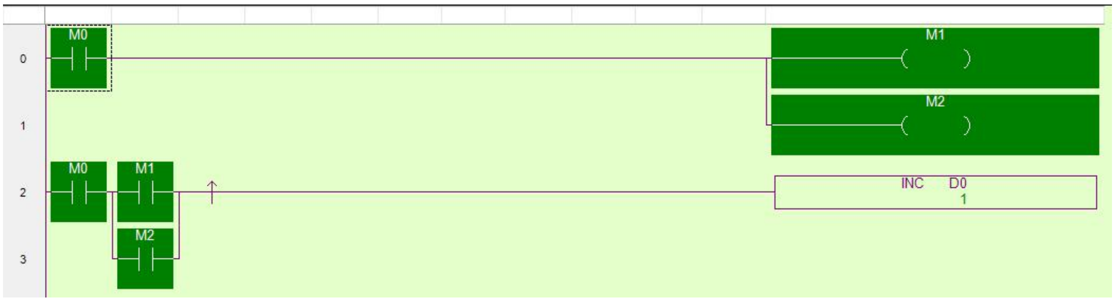
② M0 status is true, set M1 or M2 separately, and D0=1 for self addition.



③ The M0 state is true. If M1 is true and M2 is set to true, the D0 state will not self add and maintain the previous state.

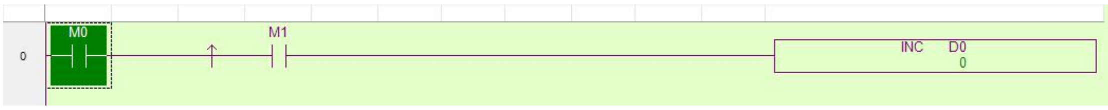


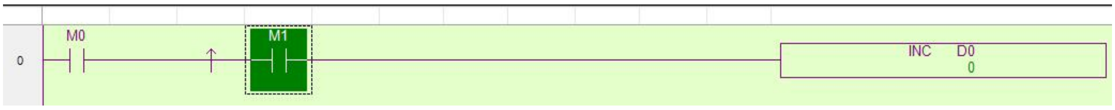
④ When M0 is set to true and M1 and M2 are set to true, D0 only adds itself once, with D0=1. Setting true to M0, M1, and M2 yields the same result; First, set M1 and M2 to true simultaneously, and then set M0 to true. D0 self adds once, D0=1.



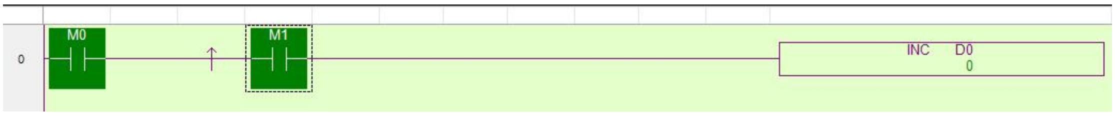
(3) After pulsation, there are nodes:

① Set on node before or after pulse, D0 will not add by itself.

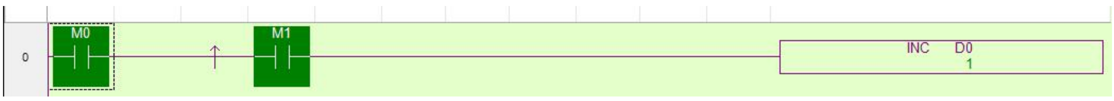




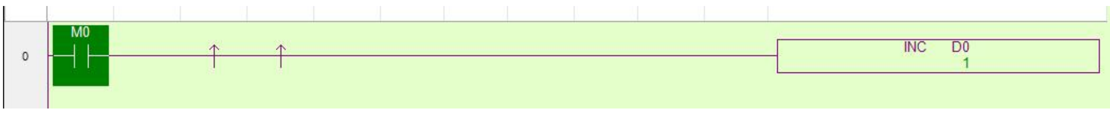
② First, set the node before the pulse instruction (M0) to true, and then set the node after the pulse instruction (M1) to true. D0 does not self add, and D0=0.



③ First, set the node after the pulse instruction (M1) to true, then set the node before the pulse instruction (M0) to true, D0 self adds, and D0=1.

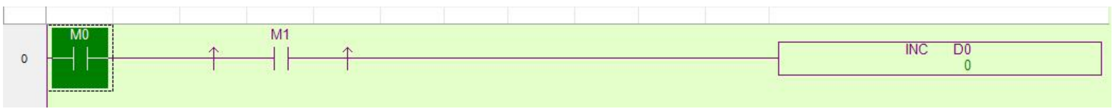


(4) Multiple pulsation: M0 is set to true, D0 self adds, and D0=1. (From the execution result, it appears to be the same as node pulsation for a single pulse)

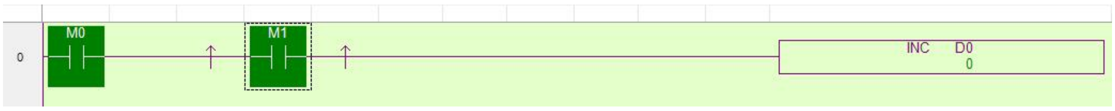


(5) There is a node between two pulsations

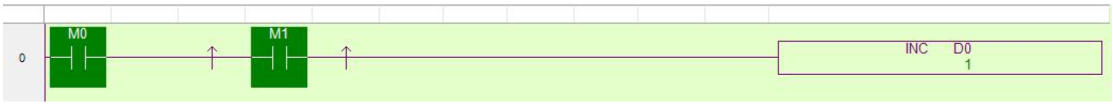
① Set M0 or M1 separately, D0 does not self add, D0=0.



② Set M0 first, then M1. D0 does not self add, and D0=0.

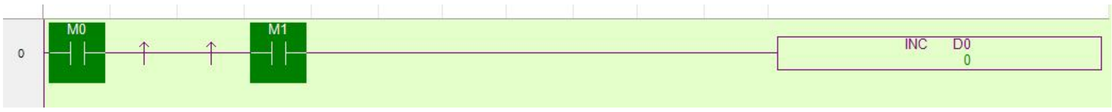


③ Set M1 first and then set M0, D0 self adds, D0=1.

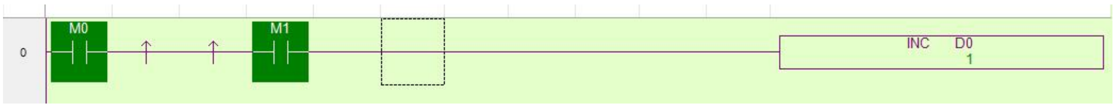


(6) There are multiple pulse instructions in the middle of the node

① First set M0 to true, then set M1 to true. D0 does not add itself, and D0=0.

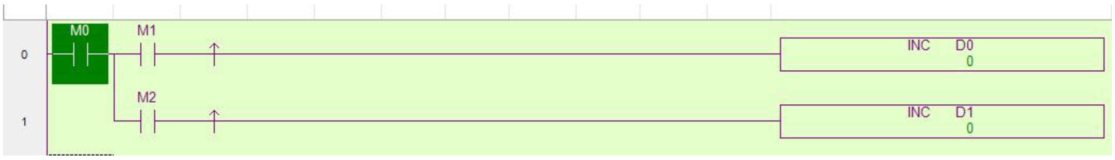


② First set M1 to true, then set M0 to true, D0 adds itself, and D0=1.

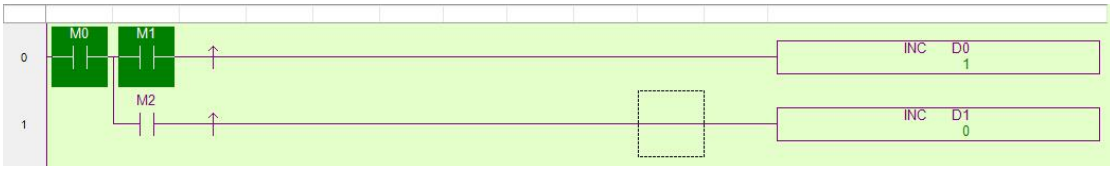


(7) Conditional pulse in output with conditions

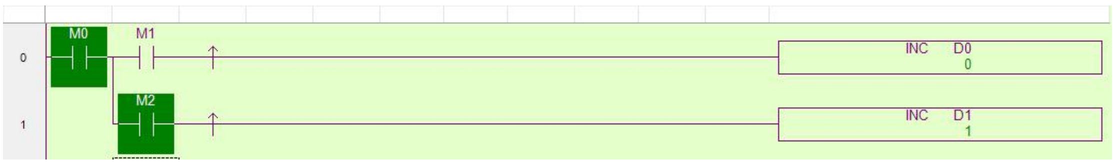
① When M0 is set to true and M1 and M2 are both false, D0 and D1 are not self added.



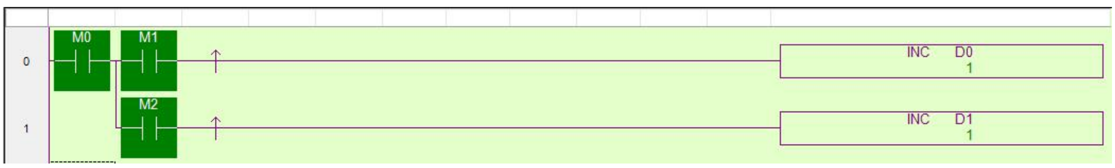
② When M0 is true and M1 is set to true (the same applies when M0 and M1 are set to true at the same time), D0=1 and D1=0. If the order in which M1 and M0 are set is changed, the result remains unchanged.



③ When M0 is true and M2 is set to true (the same applies when M0 and M2 are set to true at the same time), D0=0, D1=1. If the order in which M2 and M0 are set is changed, the result remains unchanged.

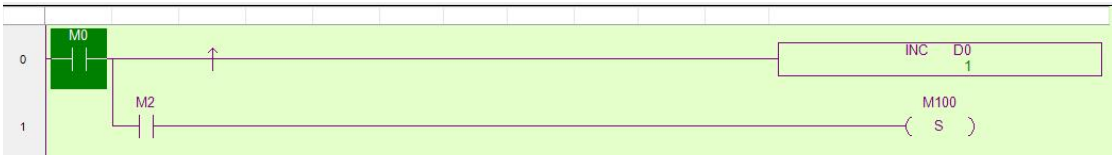


④ When M0 is true and M1 and M2 are set to true (the same applies when M0, M1, and M2 are set to true at the same time), D0=1 and D1=1; if the order in which M1/2 and M0 are set is changed, the result remains unchanged.

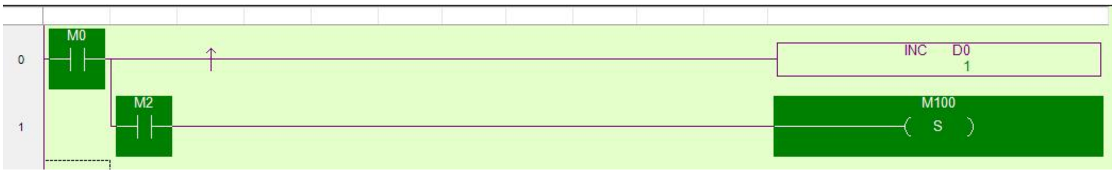


(8) Output with and without pulse coexisting

① Set M0, unset M2, D0 self adding, D0=1; M100 unset, M100=false.

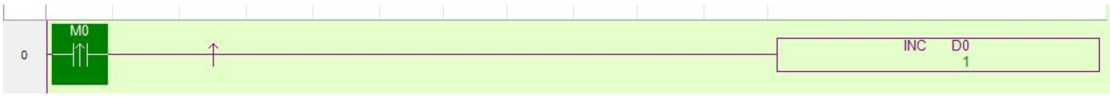


② Set M0, set M2, D0 self add, D0=1; set M100, M100=true.

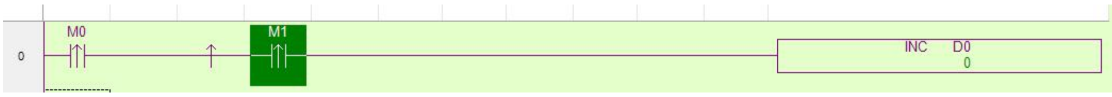


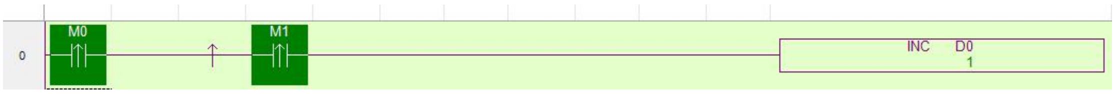
(9) Connect pulse command after edge LDP instruction

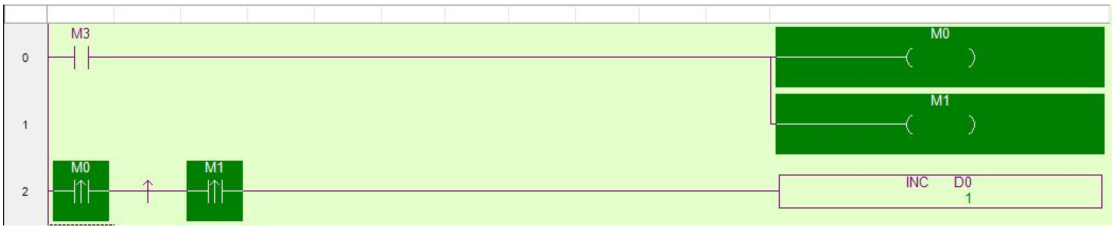
① One edge instruction and one pulse instruction: set M0 to true, D10 self adds, D10=1.



② Multiple edge instructions and multiple pulse instructions: M0 and M1 not set true at the same time, D0 does not self add, D0=0; M0 and M1 set true at the same time, D0 self adds, D0=1.

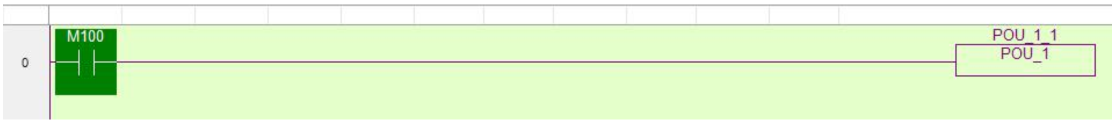




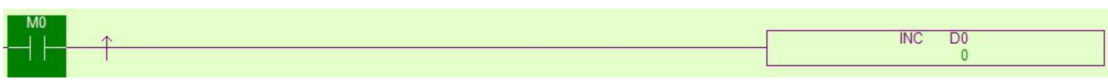


(10)FB has pulse inside: If the main ladder diagram M100=false, set M0 in FB to true, D0 is not self added, and D0=0; in this case, set M100 to true, D0 is self added, and D0=1 (only for the first time); If the main ladder diagram M100 is set to true, M0 in FB is not set to true, D0 is not self added, D0=0; if the main ladder diagram M100 is set to true, M0 in FB is set to true, D0 is self added, D0=1.

Main ladder diagram

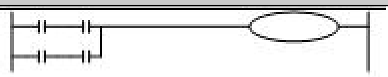


FB program



3-8 [ORB]

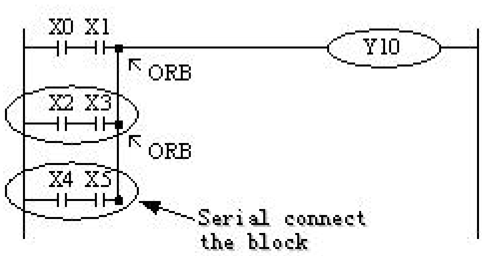
Mnemonic and Function

Mnemonic	Function	Format and Devices
ORB (OR Block)	Parallel connect the serial circuits	 Devices: none

Statements

Two or more contactors are called "serial block". If parallel connect the serial block, use LD, LDI at the branch start point, use ORB at the branch end point;
As the ANB instruction, an ORB instruction is an independent instruction which is not associated with any soft component.
There are no limits for parallel circuits’quantity when using ORB for every circuit.

Program



Recommended good programming method:

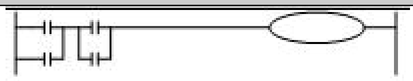
```
LD      X0
AND     X1
LD      X2
AND     X3
ORB
LD      X4
AND     X5
ORB
OUT     Y10
```

Non-preferred programming method:

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

3-9 [ANB]

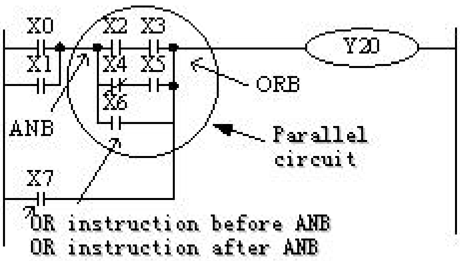
Mnemonic and Function

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

Statements

Use ANB to serial connects two parallel circuits. Use LD, LDI at the brach start point; use ANB at the branch end point.
There are no limits for ANB instruction using times.

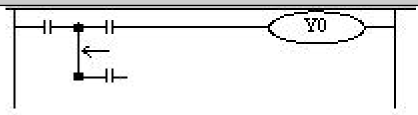
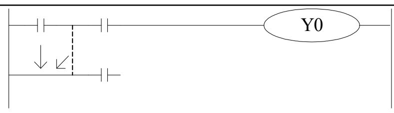
Program



LD	X0
OR	X1
LD	X2
AND	X3
LDI	X4
AND	X5
ORB	
OR	X6
ANB	
OR	X7
OUT	Y20

3-10 [MCS] , [MCR]

Mnemonic and Function

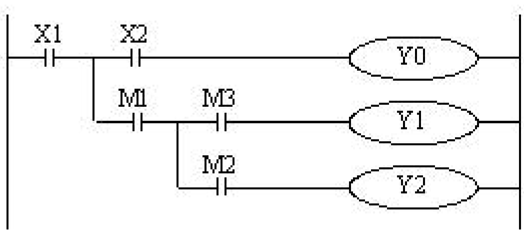
Mnemonic	Function	Format and Devices
MCS (Master control)	The start of new bus line	 Devices：None
MCR (Master control Reset)	Reset the bus line	 Devices：None

Statements

- After the execution of an MCS instruction, the bus line (LD, LDI) moves to a point after the MCS instruction. An MCR instruction resets this to the original bus line.
- MCS, MCR instructions should use in pair.
- The bus line can be nesting. Use MCS, MCR instructions between MCS, MCR instructions. The nesting level increase with the using of MCS instruction. The max nesting level is ten. When executing MCR instruction, go back to the last level of bus line.
- When use flow program, bus line management could only be used in the same flow. When the flow ends, it must go back to the main bus line.

Note: The MCS and MCR instructions can not be written directly in the ladder diagram of XD/XL series PLC programming software. They can be constructed by horizontal and vertical lines.

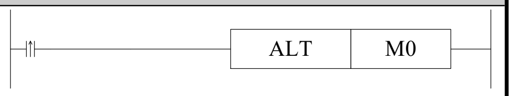
Program



LD	X1
MCS	
LD	X2
OUT	Y0
LD	M1
MCS	
LD	M3
OUT	Y1
LD	M2
OUT	Y2
MCR	
MCR	

3-11 [ALT]

Mnemonic and Function

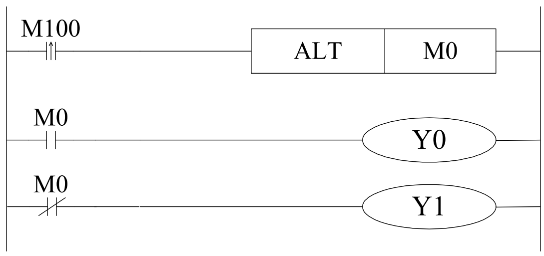
Mnemonic	Function	Format and Devices
ALT (Alternate)	Alternate the coil	 Operand soft components: X,Y,M,HM,SM,S,HS,T,HT,C,HC,Dn.m Variable type: BIT/BOOL

Statements

After executing ALT, the state of the coil can be reversed. Change from the original ON state to OFF state, or from the original OFF state to ON state.

The ALT instruction is an edge triggered instruction. If normally open and normally closed is used as the triggering condition, it will be executed once per scanning cycle.

Program

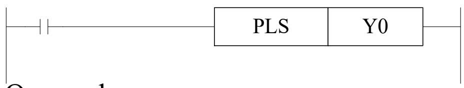
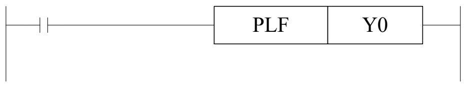


LDP	M100
ALT	M0
LD	M0
OUT	Y0
LDI	M0
OUT	Y1

3-12 [PLS] , [PLF]

This instruction has two kinds of using method: PLS/PLF+BOOL and PLS/PLF+BIT.

Mnemonic and Function

Mnemonic	Function	Format and Devices
PLS (Rising Pulse)	Turn on a scan cycle when Rising edge	 Operand: X,Y,M,HM,SM,S,HS,T,HT,C,HC,Dn.m
PLF (Falling Pulse)	Turn on a scan cycle when Falling edge	 Operand: X,Y,M,HM,SM,S,HS,T,HT,C,HC,Dn.m

- PLS/PLF+BIT type

Using range	Supported variable categories	Supported variable types
Main ladder chart	VAR_GLOBAL	BIT type variables, members of BIT arrays
Function-ladder chart language	VAR_GLOBAL	BIT type variables, members of BIT arrays
Function block-ladder diagram language	VAR_GLOBAL	BIT type variables, members of BIT arrays

- PLS/PLF+BOOL type

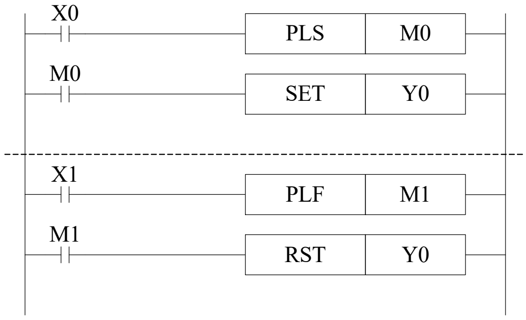
Using range	Supported variable categories	Supported variable types
Main ladder chart	VAR_GLOBAL	BOOL type variables, BOOL pointer type variables, members of BOOL arrays
Function-ladder chart language	VAR_GLOBAL	BOOL type variables, BOOL pointer type variables, members of BOOL arrays
Function block-ladder diagram language	VAR_GLOBAL, VAR, VAR_TEMP, VAR_INPUT, VAR_OUTPUT, VAR_INOUT	BOOL type variables, BOOL pointer type variables, members of BOOL arrays

Statements

For using PLS instruction: soft component Y and M will act during one scanning period after the drive is ON.

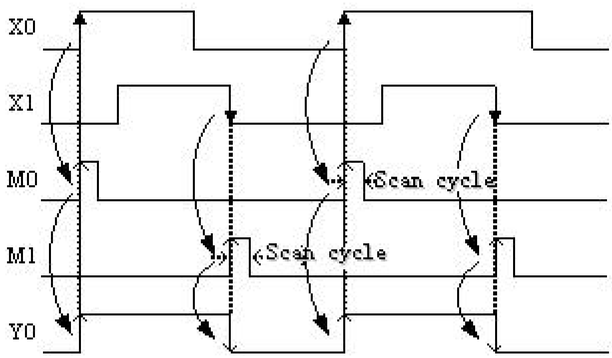
For using PLF instruction: soft component Y and M will act during one scanning period after the drive is OFF.

Program



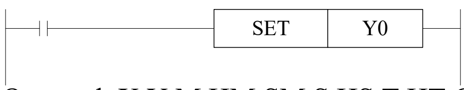
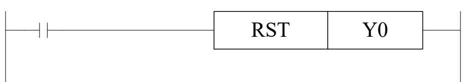
LD X0
PLS M0
LD M0
SET Y0

LD X1
PLF M1
LD M1
RST Y0



3-13 [SET], [RST]

Mnemonic and Function

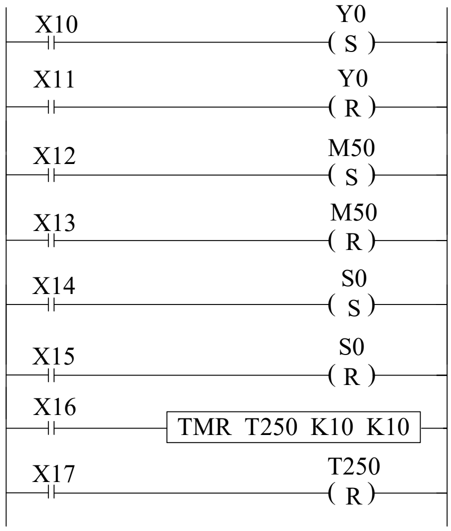
Mnemonic	Function	Format and Devices
SET (Set)	Set a bit device permanently ON	 Operand: X,Y,M,HM,SM,S,HS,T,HT,C,HC,Dn.m Variable type: BIT/BOOL
RST (Reset)	Reset a bit device permanently OFF	 Operand: X,Y,M,HM,SM,S,HS,T,HT,C,HC,Dn.m Variable type: BIT/BOOL

Statements

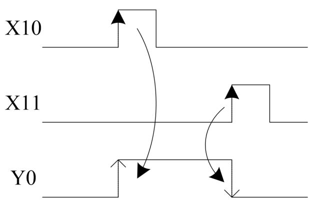
In the following program, Y0 will keep ON even X10 turns OFF after turning ON. Y0 will not ON even X11 turns OFF after turning ON. This is the same to S and M.

SET and RST can be used for many times for the same soft component. Any order is allowed, but the last one is effective.
RST can be used to reset the counter, timer and contactor.
When using SET or RST, it cannot use the same soft component with OUT.

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      X16
TMR     T250     K10     K10
LD      X17
RST     T250
```



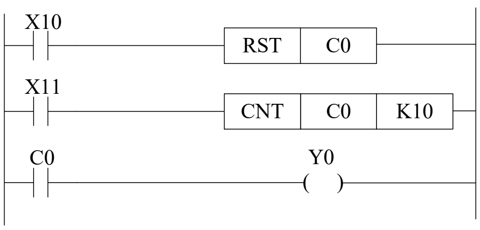
3-14 [CNT],[CNT_D],[DCNT],[DCNT_D],[RST],[CNT_FB], [DCNT_FB], [CNT_D_FB], [DCNT_D_FB] for the counters

3-14-1 Ladder chart counters [CNT],[CNT_D],[DCNT],[DCNT_D],[RST]

Mnemonic and Function

Mnemonic	Function	Format and devices
CNT Output	16 bits non power-off retentive increase count, the drive of count coil	CNT C0 K8 Operand: K, D
CNT_D Output	16 bits power-off retentive decrease count, the drive of count coil	CNT_D HC0 K8 Operand: K, D
DCNT Output	32 bits non power-off retentive increase count, the drive of count coil	DCNT C0 K8 Operand: K, D
DCNT_D Output	32 bits power-off retentive decrease count, the drive of count coil	DCNT_D HC0 K8 Operand: K, D
RST Reset	Reset the output coil, clear the current count value	RST HSC0 Operand: C, HC, HSC

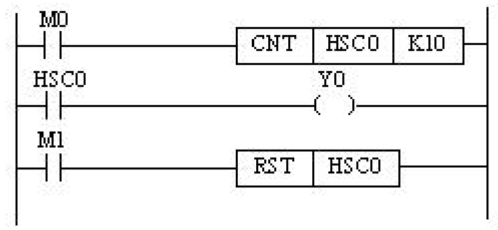
Internal counter programming



C0 increase counts the X11 OFF to ON times. When C0 reaches K10, C0 will become OFF to ON. When X11 becomes OFF to ON, the C0 current value will keep increasing, and the C0 coil will still be ON. When X10 is ON, reset the C0 coil.

Power-off retentive counter will keep the current value and counter coil status when the power is off.

High speed counter programming



Increase count the OFF to ON times of M0.
When the count value reaches set value (value of K or D), the count coil will be ON.
When M1 is ON, the count coil of HSC0 reset, the current value becomes 0.

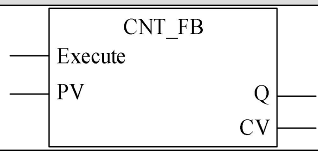
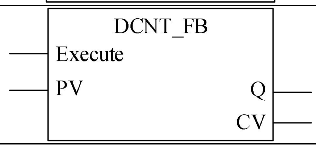
3-14-2 Function block counters

[CNT_FB],[DCNT_FB],[CNT_D_FB],[DCNT_D_FB]

3-14-2-1 Incremental counter function block [CNT_FB],[DCNT_FB]

(1) Instruction overview

CNT_FB is incremental counter function block.

Incremental counter function block [CNT_FB]			
16-bit instruction	CNT_FB	Block display	
32-bit instruction	DCNT_FB	Block display	
Execution condition	Normally open/close, edge triggering	Suitable model	XD, XL series
Firmware	-	Software	V3.7.16 and above

Note: Among the applicable models mentioned above, models other than XDH and XLH require software version V3.7.17 and above.

(2) Operand

Operand	Name	Function	Data type
-	Execute	Instruction triggering	BOOL
S1	PV	Set counting value	INT/DINT
D1	Q	Counting started/completed	BOOL
D2	CV	Current counting value	INT/DINT

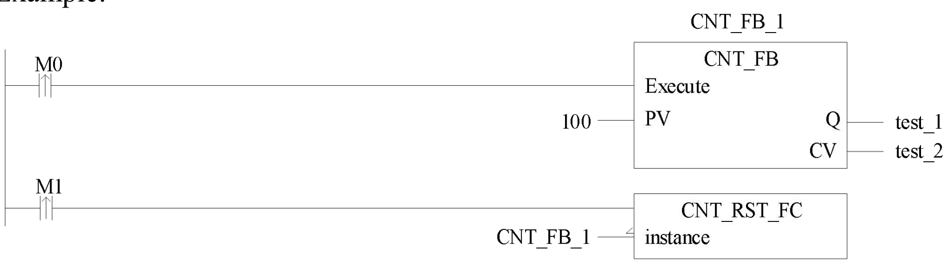
(3) Suitable soft component

Operand	Word soft component												Bit soft component							
	System								Constant	Module		Custom variables	System							Custom variables
	D	FD	TD	CD	DX	DY	DM	DS	K/H	ID	QD	.	X	Y	M	S	T	C	Dn.m	.
S1												●								
S2												●								
D1												●								
D2												●								

Note: D represents D, HD; TD represents TD, HTD; CD represents CD, HCD, HSCD, HSD; DM represents DM, DHM; DS represents DS, DHS.
M represents M, HM, SM; S represents S, HS; T represents T, HT; C represents C, HC.

(4) Function and action

- When Execute changes from OFF to ON, the count is incremented. When the value of CV reaches the set value PV, the Q state becomes ON. Afterwards, when Execute continues to change from OFF to ON, the value of CV will continue to self increase, but the Q state remains unchanged;
- It can be reset using CNT _RST_FC/DCNT _RST_FC.
  Example:

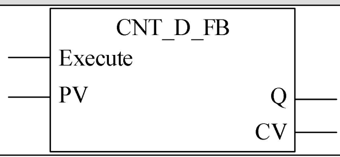
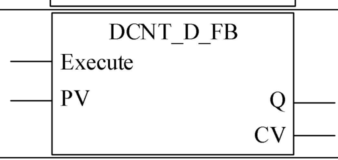


- When M0 becomes ON each time, the counter accumulates the number of self additions, and the value of test_2 is the number of counts at that time; When the timer reaches the set 100, test_1 becomes ON; When M0 continues to turn ON afterwards, the counter will continue to self increase the number of times;
- During the counting process, when M1 changes from OFF to ON, the state of test_1 becomes OFF, and the value of test_2 is reset to zero.

3-14-2-2. Decreasing counter function block [CNT_D_FB],[DCNT_D_FB]

(1) Instruction overview

CNT_D_FB is decreasing counter function block.

Decreasing counter function block [CNT_D_FB]			
16-bit instruction	CNT_D_FB	Block display	
32-bit instruction	DCNT_D_FB	Block display	

Execution condition	Normally open/close, edge triggering	Suitable model	XD, XL series
Firmware	-	Software	V3.7.16 and above

Note: Among the applicable models mentioned above, models other than XDH and XLH require software version V3.7.17 and above.

(2) Operand

Operand	Name	Function	Data type
-	Execute	Instruction triggering	BOOL
S1	PV	Set counting value	INT/DINT
D1	Q	Counting started/completed	BOOL
D2	CV	Current counting value	INT/DINT

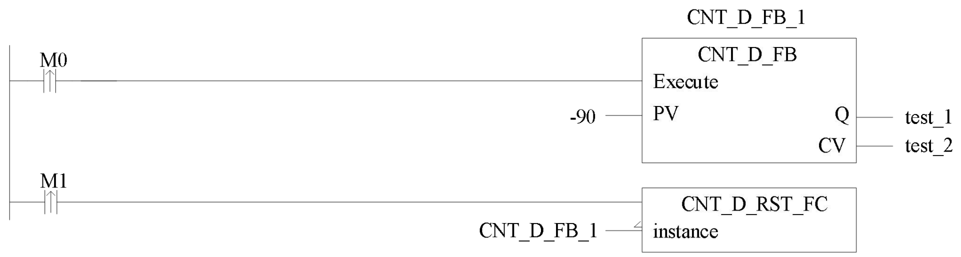
(3) Suitable soft component

Operand	Word soft component												Bit soft component							
	System								Constant	Module		Custom variables	System							Custom variables
	D	FD	TD	CD	DX	DY	DM	DS	K/H	ID	QD	.	X	Y	M	S	T	C	Dn.m	.
S1												●								
S2												●								
D1												●								
D2												●								

Note: D represents D, HD; TD represents TD, HTD; CD represents CD, HCD, HSCD, HSD; DM represents DM, DHM; DS represents DS, DHS.
M represents M, HM, SM; S represents S, HS; T represents T, HT; C represents C, HC.

(4) Function and action

- When Execute changes from OFF to ON, subtract the count. When the CV value reaches the set value PV, the Q state becomes ON; Afterwards, when Execute continues to change from OFF to ON, the value of CV will continue to self decrease, but Q will remain in the same state;
- Can be reset using CNT _D_RST_FC/DCNT _D_RST_FC.
  Example:

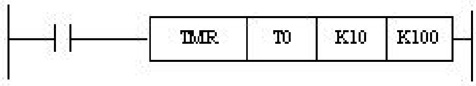
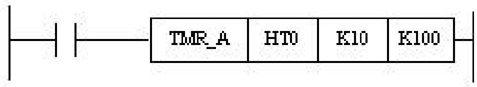
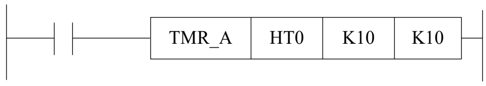
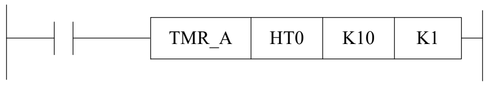


- When M0 becomes ON each time, the counter self decreases the number of times, and the value of test_2 is the counting times at that time. When the counter reaches the set value of -90, test_1 becomes ON. When M0 continues to turn ON afterwards, the counter will continue to self reduce the number of times;
- During the counting process, when M1 changes from OFF to ON, the state of test_1 becomes OFF, and the value of test_2 is reset to zero.

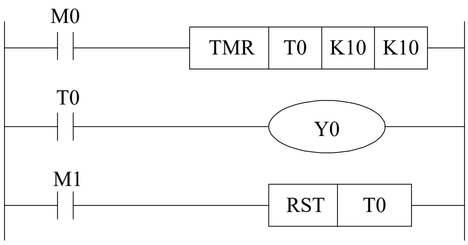
3-15 [TMR], [TMR_A],[TMR_FB],[TMR_A_FB] for timers

3-15-1 Ladder chart timer [TMR],[TMR_A]

Mnemonic and Function

Mnemonic	Function	Format and devices
TMR output	Non power-off retentive 100ms timer, the drive of coil	 operand: K, D
TMR output	Non power-off retentive 10ms timer, the drive of coil	 operand: K, D
TMR output	Non power-off retentive 1ms timer, the drive of coil	 operand: K, D
TMR_A output	Power-off retentive 100ms timer, the drive of coil	 operand: K, D
TMR_A output	Power-off retentive 10ms timer, the drive of coil	 operand: K, D
TMR_A output	Power-off retentive 1ms timer, the drive of coil	 operand: C, HC, HSC

Internal timer programming



When M0 is ON, T0 starts to timing. When T0 reaches K10, T0 coil is ON. Then T0 continues timing. When M1 is ON, reset the T0.

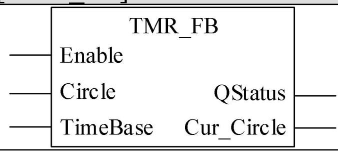
Power-off retentive timer will keep the current value and counter coil status when the power is off.

3-15-2 Function block timer [TMR_FB],[TMR_A_FB]

3-15-2-1 Not holding timer function block [TMR_FB]

(1) Instruction overview

TMR_FB is condition not conducting, not holding timer function block.

Condition not conducting, not holding timer function block [TMR_FB]			
16-bit instruction	TMR_FB	Block display	
Execution condition	Normally open/close	Suitable model	XD, XL series
Firmware	-	Software	V3.7.16 and above

Note: Among the applicable models mentioned above, models other than XDH and XLH require software version V3.7.17 and above.

(2) Operand

Operand	Name	Function	Data type
-	Enable	Instruction triggering	BOOL
S1	Circle	Setting time= Circle*TimeBase(ms)	INT
S2	TimeBase	Time base (ms), only can set to 1/10/100	INT
D1	QStatus	Timing status output	BOOL
D2	Cur_Circle	Current time	INT

(3) Suitable soft component

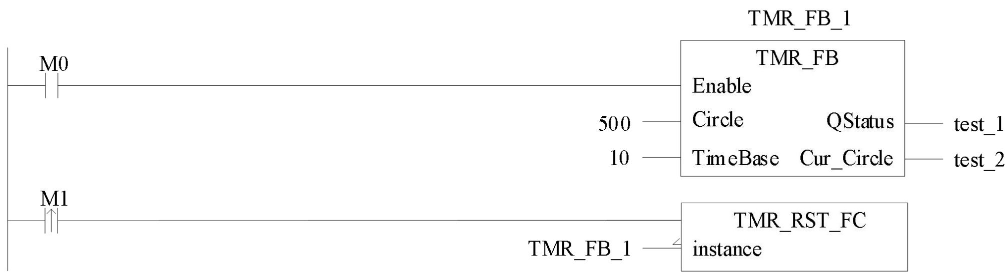
Operand	Word soft component												Bit soft component							
	System								Constant	Module		Custom variables	System							Custom variables
	D	FD	TD	CD	DX	DY	DM	DS	K/H	ID	QD	.	X	Y	M	S	T	C	Dn.m	.
S1												●								
S2												●								
D1												●								
D2												●								

Note: D represents D, HD; TD represents TD, HTD; CD represents CD, HCD, HSCD, HSD; DM represents DM, DHM; DS represents DS, DHS.
M represents M, HM, SM; S represents S, HS; T represents T, HT; C represents C, HC.

(4) Function and action

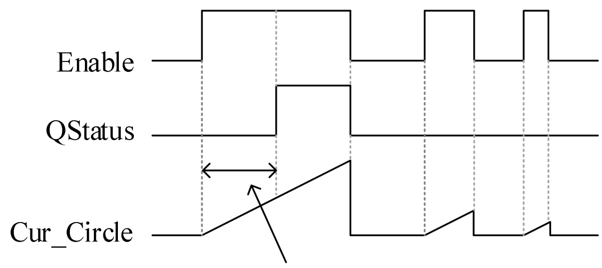
- When the Enable of the timer changes from OFF to ON, the timer starts timing. When the current timing time Cur_Circle reaches the set time Circle*TimeBase, QStatus changes from OFF to ON;
- When Enable changes from ON to OFF during the timing process, the timer Cur_Circle resets and QStatus resets;
- TMR_RST_FC can be used for resetting the timer.

Example:



- When M0 becomes ON, the timer starts timing, and the value of test_2 is the timing time at that time. When the timer reaches 5000ms, test_1 becomes ON; When M0 changes from OFF to ON, the timing stops, the state of test_1 becomes OFF, and the value of test_2 is reset to zero; During the timing process, when M1 changes from OFF to ON, the state of test_1 becomes OFF, and the value of test_2 is reset to zero.

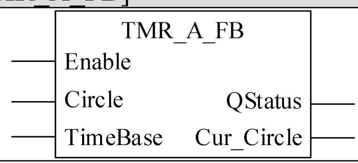
(5) Sequence digram



3-15-2-2. Holding timer function block [TMR_A_FB]

(1) Instruction overview

TMR_A_FB is ondition not conducting, holding timer function block.

Condition not conducting, holding timer function block [TMR_A_FB]			
16-bit instruction	TMR_A_FB	Block display	
Execution condition	Normally open/close	Suitable model	XD, XL series
Firmware	-	Software	V3.7.16 and above

Note: Among the applicable models mentioned above, models other than XDH and XLH require software version V3.7.17 and above.

(2) Operand

Operand	Name	Function	Data type
-	Enable	Instruction triggering	BOOL
S1	Circle	Setting time= Circle*TimeBase(ms)	INT
S2	TimeBase	Time base (ms), only can set to 1/10/100	INT
D1	QStatus	Timing status output	BOOL
D2	Cur_Circle	Current time	INT

(3) Suitable soft component

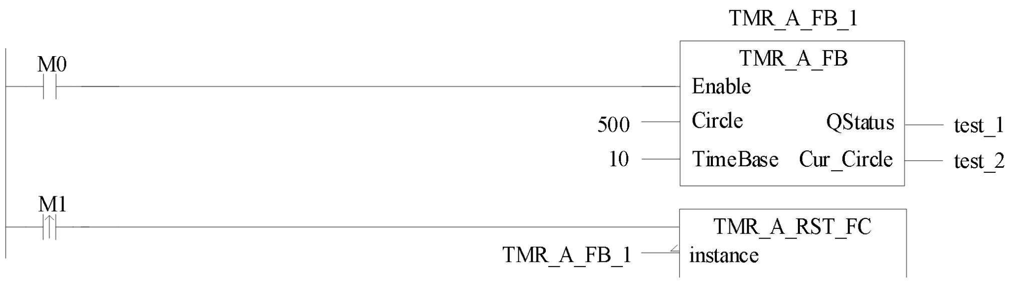
Operand	Word soft component												Bit soft component							
	System								Constant	Module		Custom variables	System							Custom variables
	D	FD	TD	CD	DX	DY	DM	DS	K/H	ID	QD	.	X	Y	M	S	T	C	Dn.m	.
S1												●								
S2												●								
D1												●								
D2												●								

Note: D represents D, HD; TD represents TD, HTD; CD represents CD, HCD, HSCD, HSD; DM represents DM, DHM; DS represents DS, DHS.
M represents M, HM, SM; S represents S, HS; T represents T, HT; C represents C, HC.

(4) Function and action

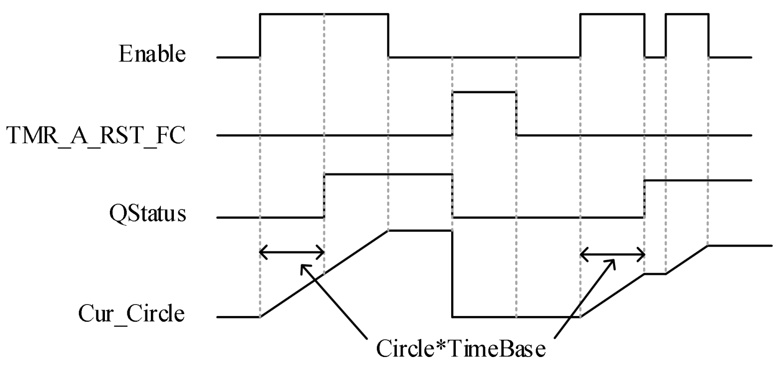
- When the Enable of the timer changes from OFF to ON, the timer starts timing. When the current timing time Cur_Circle reaches the set time Circle*TimeBase, QStatus changes from OFF to ON;
- When Enable changes from ON to OFF during the timing process, the timers Cur_Circle and QStatus remain in their current state. When Enable changes to ON, the timing continues;
- TMR_A_RST_FC can be used for resetting the timer.

Example:



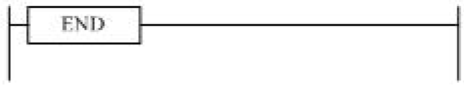
- When M0 becomes ON, the timer starts timing, and the value of test_2 is the timing time at that time. When the timer reaches 5000ms, test_1 becomes ON; When M0 changes from OFF to ON, the timing pauses, and the status of test_1 and the value of test_2 remain. Set M0 again to continue timing, and test_2 continues to accumulate; During the timing process, when M1 changes from OFF to ON, the state of test_1 becomes OFF, and the value of test_2 is reset to zero.

(5) Sequence diagram

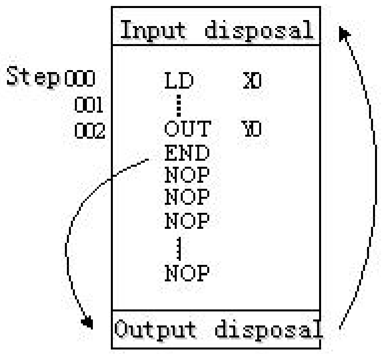


3-16 [END]

Mnemonic and Function

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

Statements



PLC repeatedly carries on 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 then repeats executing the program from step 0.
When debug, insert END in each program segment to check out each program's action. Then, after confirm the correction of preceding block's action, delete END instruction. Besides, the first execution of RUN begins with END instruction.
When executing END instruction, refresh monitor timer. (Check if scan cycle is a long timer.)

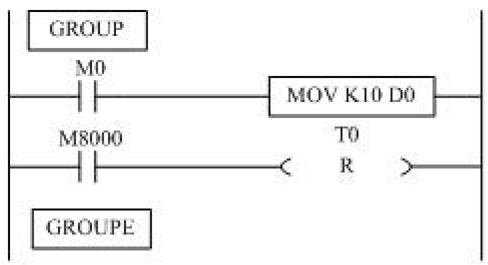
3-17 [GROUP] , [GROUPE]

Mnemonic and Function

Mnemonic	Function	Format and Device
GROUP	GROUP	GROUP Devices: None
GROUPE	GROUP END	GROUPE Devices: None

Statements

GROUP and GROUPE should use 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 affect 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. Meantime, the programmed instructions can be FOLDED or UNFOLDED. To a redundant project, these two instructions are quite useful.

3-17 Programming notes

Contactor structure and steps

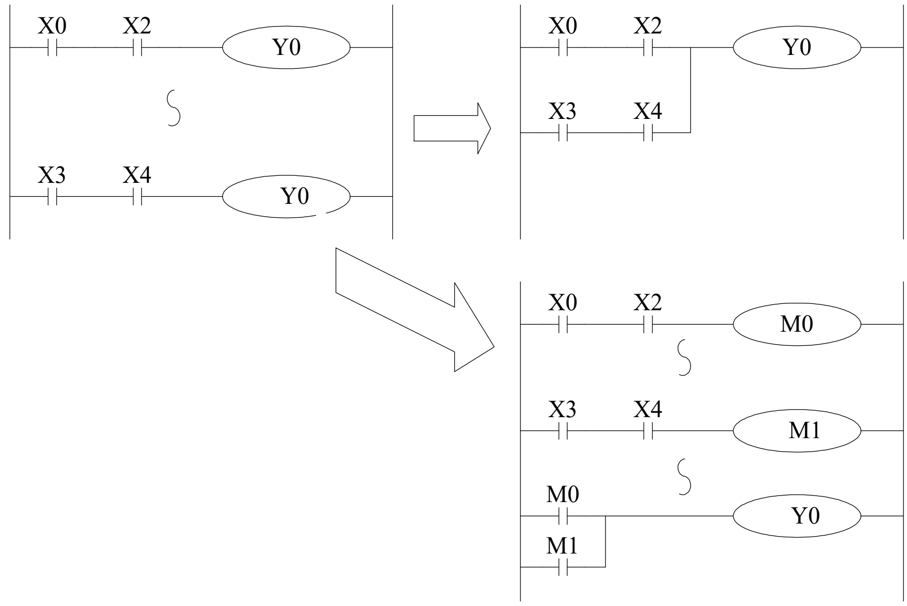
Even in the sequencial control circuit with the same function, it’s also available to simplify the program and shorten the program steps according to the contactors’ structure. General programming principle is: (a) write the circuit with many serial contacts on the top; (b) write the circuit with many parallel contactors in the left.

Program’s executing sequence

Handle the sequencial control program by【From top to bottom】and【From left to right】Sequencial control instructions also encode following this procedure.

Dual output dual coil’s activation and the solution

If carry on coil's dual output (dual coil) in the sequencial control program, then the last action is prior.
Dual output (dual coil) doesn't go against the input rule. But as the preceding action is very complicate, please modify the program as in the following example.



There are other methods. E.g. jump instructions or flow instructions.

4 Applied Instructions

In this chapter, we describe applied instruction’s function of XD, XL series PLC.

4-1 Applied Instructions List

Mnemonic	Function	Ladder chart	Chapter
Program Flow			
CJ	Condition jump	CJ Pn	4-3-1
CALL	Call subroutine	CALL Pn	4-3-2
SRET	Subroutine return	SRET	4-3-2
STL	Flow start	STL Sn	4-3-3
STLE	Flow end	STLE	4-3-3
SET	Open the assigned flow, close the current flow	SET Sn	4-3-3
ST	Open the assigned flow, not close the current flow	ST Sn	4-3-3
FOR	Start a FOR-NEXT loop	FOR S	4-3-4
NEXT	End of a FOR-NEXT loop	NEXT	4-3-4
FEND	Main program END	FEND	4-3-5
END	Program END	END	4-3-5
Data Compare			
LD＝	LD activates if (S1) = (S2)	LD= S1 S2	4-4-1
LD＞	LD activates if (S1) > (S2)	LD＞ S1 S2	4-4-1
LD＜	LD activates if (S1) =< (S2)	LD＜ S1 S2	4-4-1
LD＜＞	LD activates if (S1) ≠(S2)	LD＜＞ S1 S2	4-4-1
LD＜＝	LD activates if(S1) ≤ (S2)	LD<= S1 S2	4-4-1
LD＞＝	LD activates if(S1) ≥ (S2)	LD>= S1 S2	4-4-1
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)	AND<= S1 S2	4-4-2
AND＞＝	AND activates if(S1) ≥ (S2)	AND= S1 S2	4-4-2
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)	OR＜ S1 S2	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)	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	ZCP S1 S2 S D	4-5-2
MOV	Move	MOV S D	4-5-3
BMOV	Block move	BMOV S D n	4-5-4
PMOV	Transfer the Data block	PMOV S D n	4-5-5
FMOV	Multi-points repeat move	FMOV S D n	4-5-6
EMOV	Float number move	EMOV S D	4-5-7
FWRT	Flash ROM written	FWRT S D	4-5-8
MSET	Zone set	MSET S1 S2	4-5-9
ZRST	Zone reset	ZRST S1 S2	4-5-10
SWAP	Swap the high and low byte	SWAP S	4-5-11
XCH	Exchange two values	XCH D1 D2	4-5-12
Data Operation			
ADD	Addition	ADD S1 S2 D	4-6-1
SUB	Subtraction	SUB S1 S2 D	4-6-2
MUL	Multiplication	MUL S1 S2 D	4-6-3
DIV	Division	DIV S1 S2 D	4-6-4

INC	Increment	INC D	4-6-5
DEC	Decrement	DEC D	4-6-5
MEAN	Mean	MEAN S D n	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 eXD3lusive 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
DWTD	32 bits integer to64 bits integer	DWTD S D	4-8-1
BDWTD	32 bits integer to64 bits integer batch conversion	BDWTD S D n	4-8-2
FLT	16 bits integer converts to float point	FLT S D	4-8-3
DFLT	32 bits integer converts to float point	DFLT S D	4-8-3
FLTD	64 bits integer converts to float point	FLTD S D	4-8-3
DFLTD	32 bits integer to double precision floating point	DFLTD S D	4-8-4

QFLTD	64 bits integer to double precision floating point	QFLTD S D	4-8-4
INT	Float point converts to integer	INT S D	4-8-5
DINTD	Double - precision floating point to32 bits integer	DINTD S D	4-8-6
QINTD	Double - precision floating point to64 bits integer	QINTD S D	4-8-6
ECON	Single precision floating point to double precision floating point	ECON S D	4-8-7
BECON	Single precision floating point to double precision floating point batch conversion	BECON S D n	4-8-8
BIN	BCD converts to binary	BIN S D	4-8-9
BCD	Binary converts to BCD	BCD S D	4-8-10
ASCI	Hex. converts to ASCII	ASCI S D n	4-8-11
HEX	ASCII converts to Hex.	HEX S D n	4-8-12
DECO	Coding	DECO S D n	4-8-13
ENCO	High bit coding	ENCO S D n	4-8-14
ENCOL	Low bit coding	ENCOL S D n	4-8-15
GRY	Binary to Gray code	GRY S D	4-8-16
GBIN	Gray code to binary	GBIN S D	4-8-17
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	Float Sine	ASIN S D	4-9-11
ACOS	Float Cosine	ACOS S D	4-9-12
ATAN	Float 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
MOV	Accurate clock BD board data read	MOV S D	4-10-3
TO	Accurate clock BD board data write	TO S1 S2 S3 D	4-10-4
TADD	Clock data add	TADD S1 S2 D	4-10-5
TSUB	Clock data sub	TSUB S1 S2 D	4-10-6
HTOS	Convert hour, minute, and second data to seconds	HTOS S D	4-10-7
STOH	Convert second data to hours, minutes, and seconds	STOH S D	4-10-8
TCMP	Time (hours, minutes, seconds) compare	TCMP S1 S2 S3 S D	4-10-9
DACMP	Date (year, month, day) compare	DACMP S1 S2 S3 S D	4-10-10

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	XD, XL
Hardware requirement	-	Software requirement	-

2)Operands

Operands	Function	Data Type
S1	Specify the data or register address	16 bits/32 bits, BIN
S2	Specify the data or register address	16 bits/32 bits, BIN
D	Specify the register to store the sum result	16 bits/32 bits, BIN

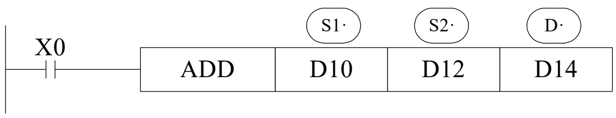
3)Suitable Soft Components

Operands	Word soft elements											Bit soft elements						
	System								Constant	Module		System						
	D	FD	TD	CD	DX	DY	DM	DS	K/H	ID	QD	X	Y	M	S	T	C	Dn.m
S1	●	●	●	●	●	●	●	●	●									
S2	●	●	●	●	●	●	●	●	●									
D	●	●	●	●		●	●	●										

*Note: D includes D, HD. TD includes TD, HTD. CD includes CD, HCD, HSCD, HSD. DM includes DM, DHM. DS includes DS, DHS. M includes M, HM, SM. S includes S and HS. T includes T and HT. C includes C and HC.

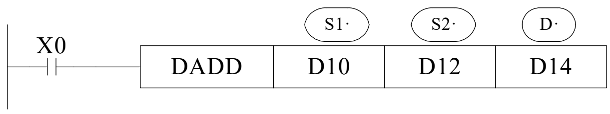
Description

<16 bits instruction>



(D10) + (D12) → (D14)

<32 bits instruction>



(D11D10) + (D13D12) → (D15D14)

- Two source data make binary addition and the result data store in object address.
- The highest bit of each data is positive (0) and negative (1) sign bit. These data will make addition operation through algebra. Such as 5 + (-8) = -3.
- If the result of a calculations is “0”, the “0’ flag acts. If the result exceeds 323,767(16 bits

operation) or 2,147,483,648 (32 bits operation), the carry flag acts. (refer to the next page). If the result exceeds -323,768 (16 bits operation) or -2,147,483,648 (32 bits operation), the borrow flag acts (Refer to the next page).

- When carry on32 bits operation, low16 bits of 32-bit register are assigned, the register address close to the low16 bits register will be assigned to high16 bits of 32-bit register. Even number is recommended for the low16 bits register address.
- The source and object can be same register address.
- **In the above example, when X0 is ON, the addition operation will be excuted in each scanning period. Please be sure to pay attention. Alternatively, you can choose to trigger using edge commands.**

Related flag

Flag	Name	Function
SM20	Zero	ON: the calculate result is zero OFF: the calculate result is not zero
SM21	Borrow	ON: the calculate result is over 32767(16bits) or 2147483647(32bits) OFF: the calculate result is not over 32767(16bits) or 2147483647(32bits)
SM22	Carry	ON: the calculate result is over 32767(16bits) or 2147483647(32bits) OFF: the calculate result is not over 32767(16bits) or 2147483647(32bits)

Notes

- The assignment of the data

The data register of XD, XL series PLC is a single word (16 bits) data register, single word data only occupy one register which is used to single word instruction. The process range is decimal –327,68~327,67, or hex 0000~FFFF.

Single word object instruction D(NUM)

Instruction	D(NUM)	→	Object

Double words (32 bits) occupy two data registers; the two registers' address is continuous. The process range is: decimal -214,748,364,8~214,748,364,7 or hex 00000000~FFFFFFFF.

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

Instruction	D(NUM)	→	Object	Object

- The way to represent32 bits instruction

Add letter "D" before16 bits instruction to represent32 bits instruction.

For example:

ADD D0 D2 D4 16 bits instruction

DADD D10 D12 D14 32 bits instruction

※1: It shows the flag bit following the instruction action.
※2: (S·) Source operand which won't change with instruction working
※3: (D·) Destinate operand which will change with instruction working
※4: It introduces the instruction's basic action, using way, applied example, extend function, note items and so on.

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 the instruction to execute part of the program, CJ shortens the operation cycle and avoids using the dual coil

Condition Jump [CJ]			
16 bits	CJ	32 bits	-
Execution condition	Normally ON/OFF coil	Suitable Models	XD, XL
Hardware requirement	-	Software requirement	-

2)Operands

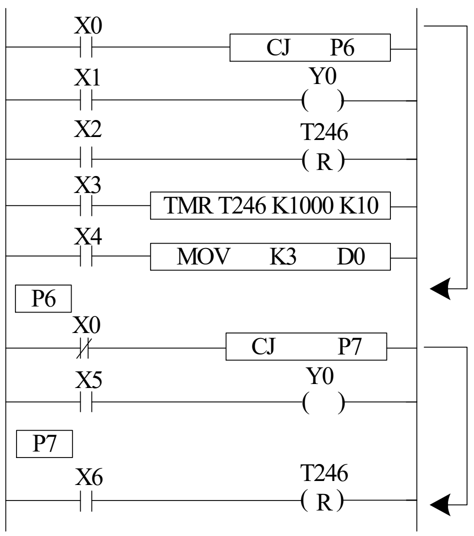
Operands	Function	Data Type
Pn	Jump to the target (with pointer Nr.) P (P0~P9999)	Pointer's Nr.

3)Suitable Soft Components

Others	Pointer	
	P	I
	●	

Description

In the below graph, if X0 is ON, jump from the first step to the next step behind P6 tag. If X0 is OFF, do not execute the jump instruction;



- In the left graph, Y0 becomes to be dual coil output, but when X0=OFF, X1 activates; when X0=ON, X5 activates
- CJ can't jump from one STL to another STL;
- After driving timer T0~T575, HT0~HT795 and HSC0~HSC30, if executes CJ, continue working, the output activates.
- The Tag must be match when using CJ instruction.

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	XD, XL
Hardware requirement	-	Software requirement	-
Subroutine Return [SRET]			
16 bits	SRET	32 bits	-
Execution condition	-	Suitable Models	XD, XL
Hardware requirement	-	Software requirement	-

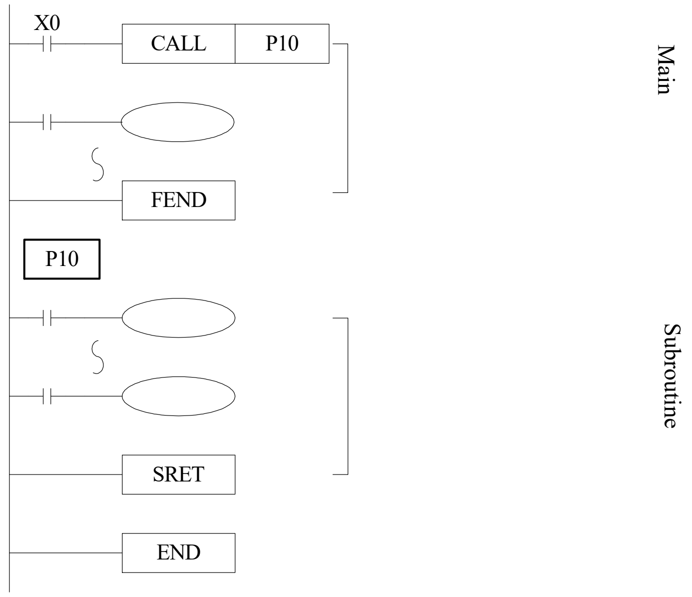
2)Operands

Operands	Function	Data Type
Pn	Jump to the target (with pointer No.) P (P0~P9999)	Pointer's No.

3) Suitable Soft Components

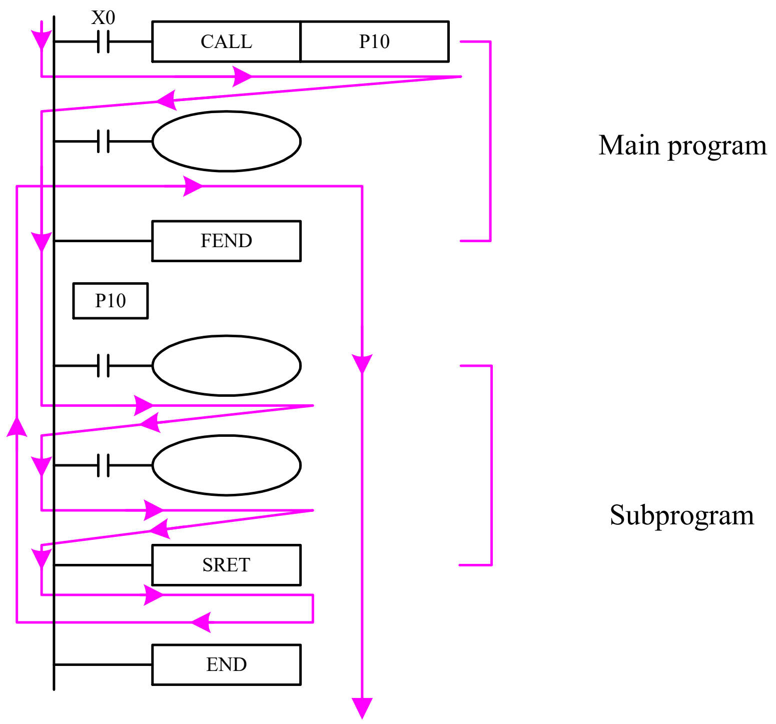
Others	Pointer	
	P	I
	●	

Description



- If X0= ON, execute the call instruction and jump to 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.
- When calling the subprogram, all the timer, OUT, PLS, PLF of the main program will keep the status.
- All the OUT, PLS, PLF, timer of subprogram will keep the status when subprogram returning.
- Do not write pulse, counter or timer inside the subprogram which cannot be completed in one scan period.
- The call instruction and subroutine pointer P must be used together, otherwise an error message "non-existent label type" will be reported.

Subprogram executing diagram:



If X0=ON, the program executes as the arrow.
If X0=OFF, the CALL instruction will not work; only the main program works.
The notes to write the subprogram:
Please programming the tag after FEND. Pn is the start of subprogram; SRET is the end of subprogram. CALL Pn is used to call the subprogram. The range of n is 0 to 9999.
The subprogram calling can simplify the programming. If the program will be used in many places, make the program in subprogram and call it.

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	XD, XL
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	XD, XL
Hardware requirement	-	Software requirement	-
Flow starts [STL]			
16 bits	STL	32 bits	-
Execution condition	-	Suitable Models	XD, XL

Hardware requirement	-	Software requirement	-
Flow ends [STLE]			
16 bits	STLE	32 bits	-
Execution condition	-	Suitable Models	XD, XL
Hardware requirement	-	Software requirement	-

2)Operands

Operands	Function	Data Type
Sn	Jump to the target flow S	Flow No.

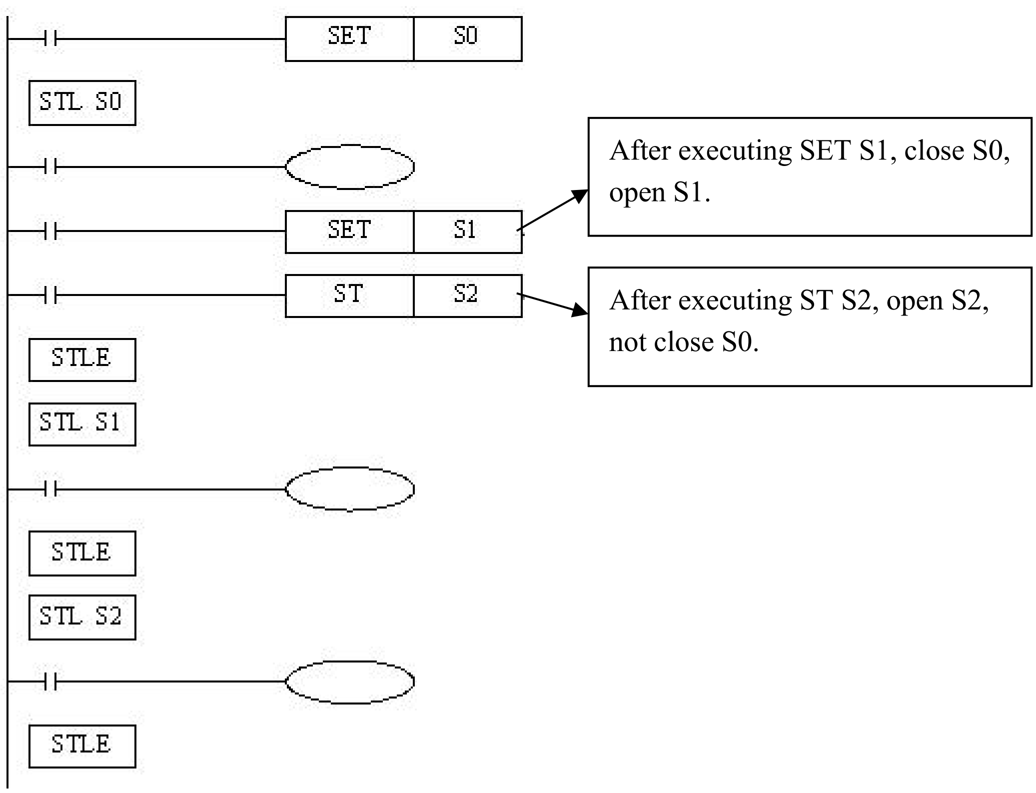
3)Suitable Soft Components

Operands	Word soft elements											Bit soft elements						
	System								Constant	Module		System						
	D	FD	TD	CD	DX	DY	DM	DS	K/H	ID	QD	X	Y	M	S	T	C	Dn.m
Sn															●			

*Notes: D includes D, HD; TD includes TD, HTD; CD includes CD, HCD, HSCD, HSD; DM includes DM, DHM; DS includes DS, DHS. M includes M,HM,SM； S includes S,HS； T includes T,HT； C includes C, HC.

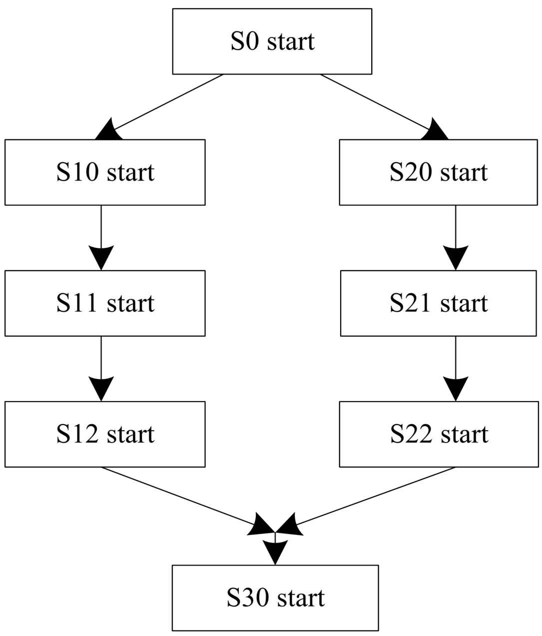
Description

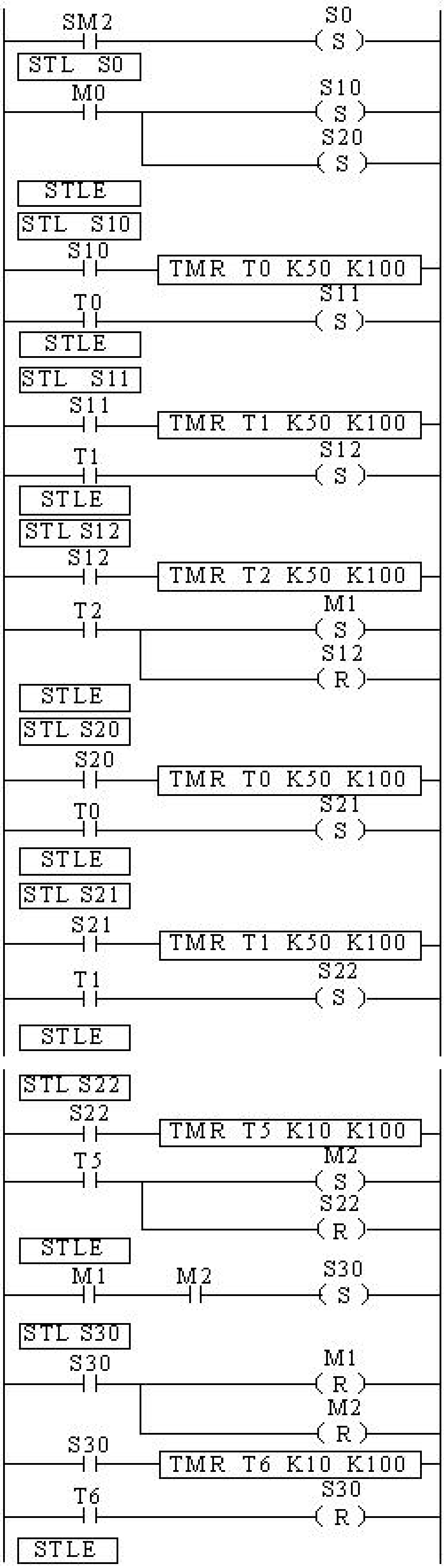
- STL and STLE should be used in pairs. STL represents the start of a flow; STLE represents the end of a flow.
- Every flow is independent. They cannot be nesting. There is no need to write the flow as the order S0, S1, S2… you can make the order. For example, executing S10, then S5, S0.
- After executing 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. in the flow.
- ST instruction is usually used when a program needs to run many flows at the same time.
- After executing**SET Sxxx** instruction and jump to the next flow, the pulse instructions in the former flow will be closed. (including one-segment, multi-segment, relative or absolute, return to the origin)



Example

Example 1: the flows run in branch then merge in one flow.
Program diagram:





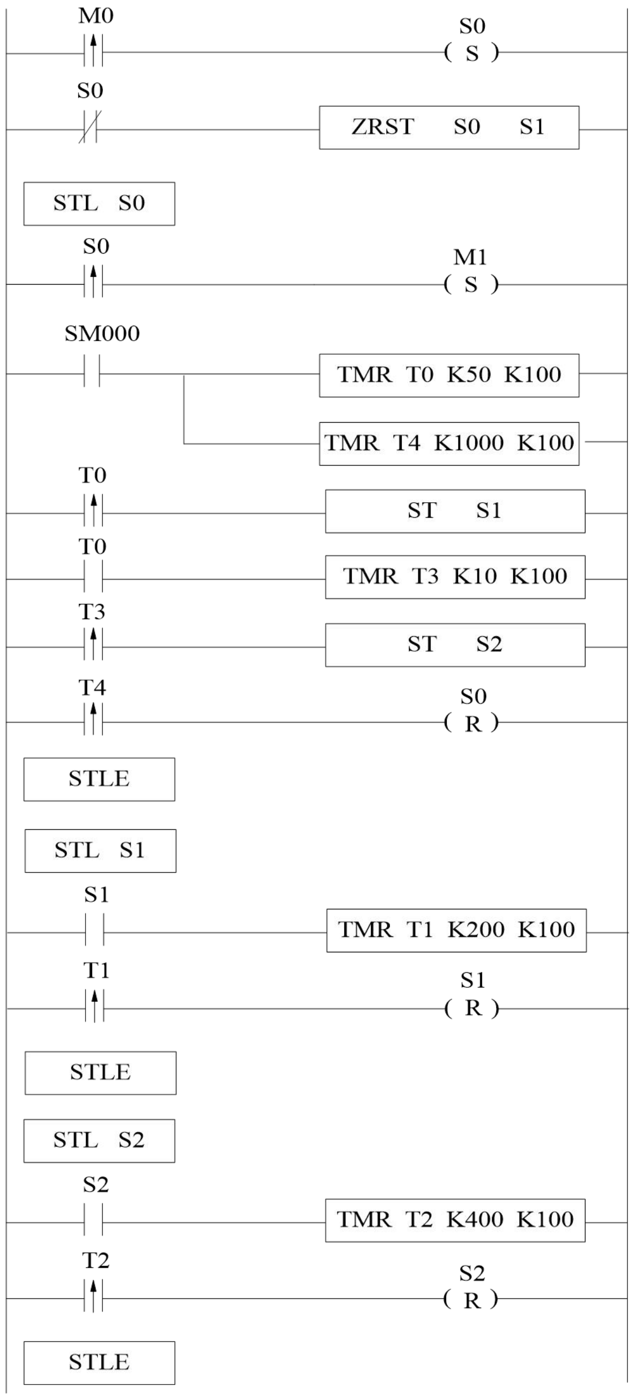
The program explanation:
When SM2 is ON, set ON flow S0. When M0 is ON, set ON flow S10 and S20.

In S10 branch, it runs S10, S11 and S12. Set on M1 means the S10 branch is finished.

In S20 branch, it runs S20, S21 and S22. Set on M2 means the S20 branch is finished.

When both branch S10 and S20 end, set on S30. When S30 end, reset S30.

Example 2: flow nesting. When S0 is running for a while, S1 and S2 start to run; the running status of S1 is kept. When S0 is running for certain time, closes S0 and force close S1 and S2.



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	XD, XL
Hardware requirement	-	Software requirement	-
Loop ends [NEXT]			
16 bits	NEXT	32 bits	-
Execution condition	Normally ON/OFF, Rising/Falling edge	Suitable Models	XD, XL
Hardware requirement	-	Software requirement	-

2)Operands

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

3)Suitable Soft Components

Operands	Word soft elements											Bit soft elements						
	System								Constant	Module		System						
	D	FD	TD	CD	DX	DY	DM	DS	K/H	ID	QD	X	Y	M	S	T	C	Dn.m
Sn	●								●									

*Notes: D includes D, HD; TD includes TD, HTD; CD includes CD, HCD, HSCD, HSD; DM includes DM, DHM; DS includes DS, DHS. M includes M,HM,SM； S includes S,HS； T includes T,HT； C includes C, HC.

Description

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

The program after NEXT will not be executed unless the program between FOR and NEXT is executed for specified times.

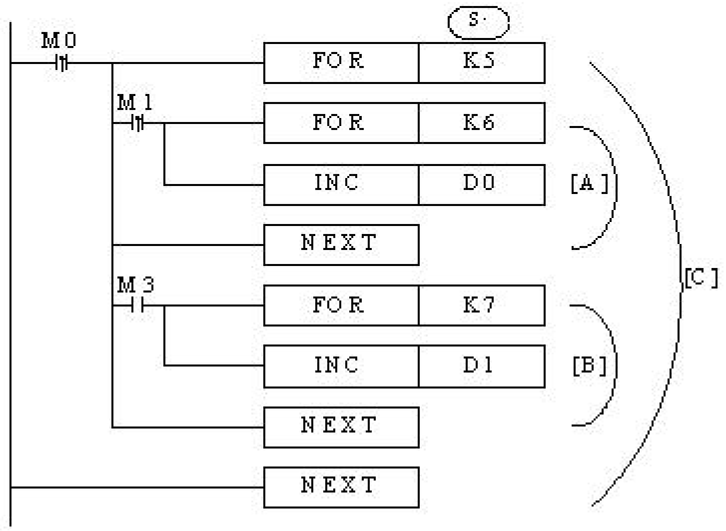
Between FOR and NEXT, LDP, LDF instructions are effective for one time. Every time when M0 turns from OFF to ON, and M1 turns from OFF to ON, [A] loop is executed 5×6=30 times.

Every time if M0 turns from OFF to ON and M3 is ON, [B] loop is executed 5×7=35 times.

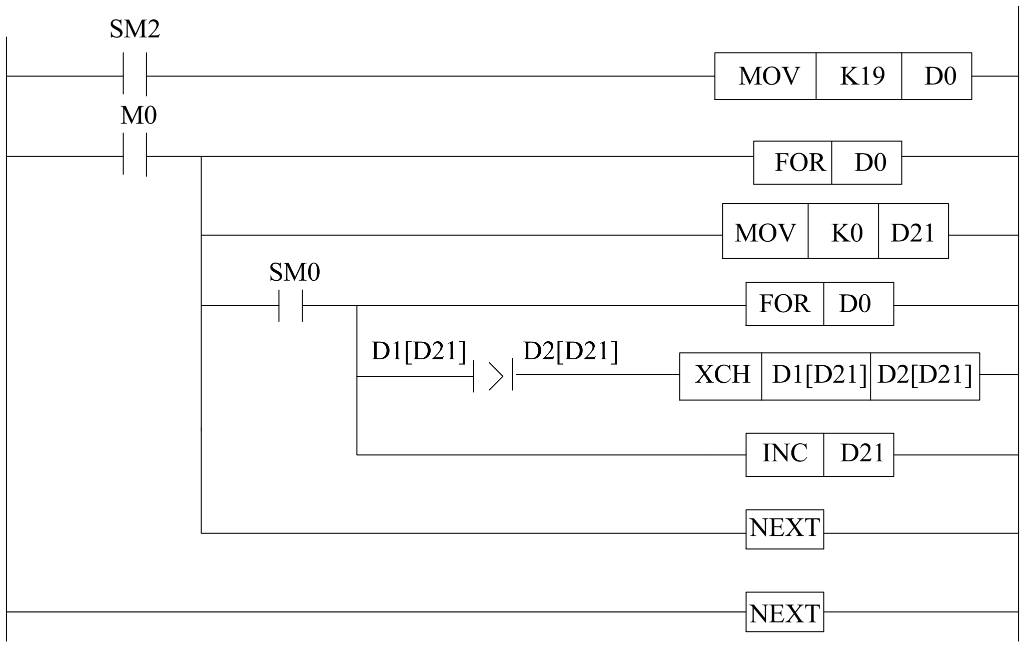
If there are many loop times, the scan cycle will be prolonged. Monitor timer error may occur, please note this.

If NEXT is before FOR, or no NEXT, or NEXT is behind FEND,END, or FOR and NEXT number is not equal, an error will occur.

Between FOR~NEXT, CJ nesting is not allowed. FOR~NEXT must be in pairs in one STL.



Example 1: when M0 is ON, the FOR NEXT starts to sort the numbers in the range of D1 to D20 from small to large. D21 is offset value. If there are many sortings in the program, please use C language to save the programming time and scanning time.




```
LD      SM2               //SM2 is initial ON coil
MOV     K19     D0        //the times of FOR loop
LD      M0                //M0 to trigger the FOR loop
MCS               //
FOR     D0                //Nesting FOR loop, the loop times is D0
MOV     K0      D21       //the offset starts from 0
LD      SM0               //SM0 is always ON coil
MCS               //
FOR             D0        //nesting FOR loop, the loop times is D0
LD>     D1[D21]     D2[D21]     //if the current data is larger than the next, it will be ON
```

```
XCH    D1[D21]          D2[D21]   //exchange the two neighbouring data
LD      SM0                 //M8000 is always ON coil
INC      D21               //increase one for D21
MCR                  //
NEXT                  //match the second FOR
MCR                  //
NEXT                  //match the first FOR
```

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	XD, XL
Hardware requirement	-	Software requirement	-
program ends [END]			
Execution condition	-	Suitable Models	XD, XL
Hardware requirement	-	Software requirement	-

2)Operands

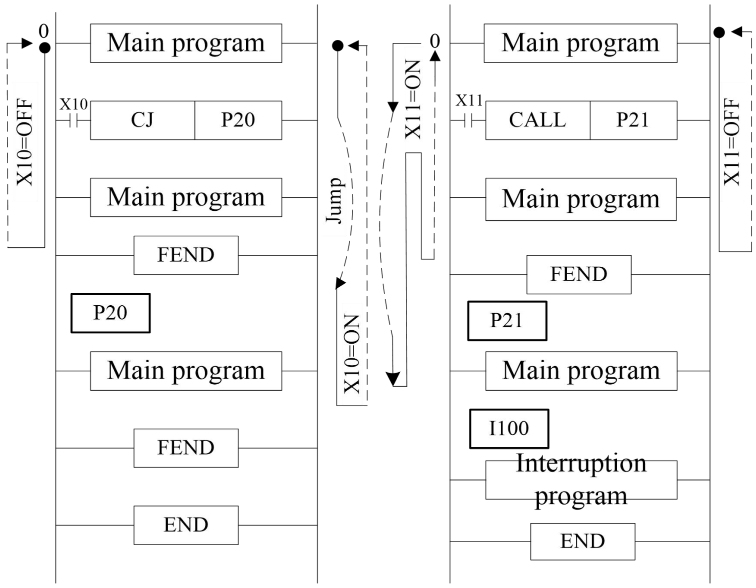
Operands	Function	Data Type
None	-	-

3)Suitable Soft Components

None

Description

Even though [FEND] instruction represents the end of the main program, the function is same to END toprocess the output/input, monitor the refresh of the timer, return to program step0.



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, an error will occur.

In the condition of using many FEND instructions, please make program or subprogram 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]

4-4-1-1 Integer compare [LD],[DLD]

(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	XD, XL
Hardware requirement	-	Software requirement	-

2) Operands

Operands	Function	Data Type
S1	Being compared number address	16/32bits, BIN; INT/DINT
S2	Comparand address	16/32 bits, BIN; INT/DINT

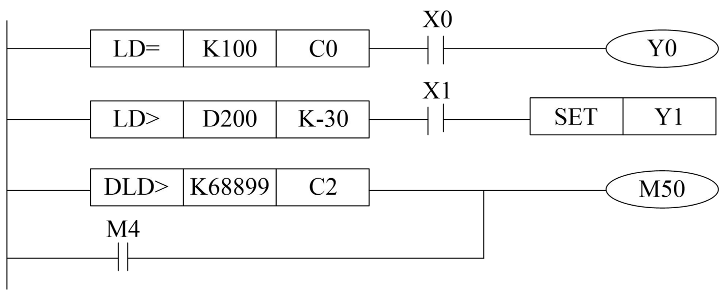
3) Suitable soft components

Operands	Word soft elements											Bit soft elements						
	System								Constant	Module		System						
	D	FD	TD	CD	DX	DY	DM	DS	K/H	ID	QD	X	Y	M	S	T	C	Dn.m
S1	●	●	●	●	●	●	●	●	●									
S2	●	●	●	●	●	●	●	●	●									

*Notes: D includes D, HD; TD includes TD, HTD; CD includes CD, HCD, HSCD, HSD; DM includes DM, DHM; DS includes DS, DHS. M includes M,HM,SM； S includes S,HS； T includes T,HT； C includes C, HC.

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)



Note Items

- When the source data's highest bit (16 bits: b15,32 bits: b31) is 1, the data is seemed to a negative number.
- The comparison of 32 bits counter should use 32 bits instruction. If using16 bits instruction, the program or operation will be error.

4-4-1-2. Floating number compare [ELD],[EDLD]

(1) Instruction overview

ELD is a contact comparison command which connecting busbars.

Start comparing [ELD□]			
32-bit instruction	Described below	64-bit instruction	Described below
Execution condition	-	Suitable model	XDH, XLH, XG2
Firmware	V3.7.3 and above	Software	V3.7.16 and above

(2) Operand

Operand	Function	Type
Two operands		
S1	Specify the numerical value or software component address of the number to be compared	32/64 bits BIN; REAL/LREAL
S2	Specify a numerical value or software component address for comparison	32/64 bits, BIN; REAL/LREAL
Three operands		
S1	Specify the numerical value or software component address of the number to be compared	32/64 bits; REAL/LREAL
S2	Specify a numerical value or software component address for comparison	32/64 bits; REAL/LREAL
S3	Specify the numerical value of absolute error during comparison $1*10^{-S3}$	16 bits, BIN

(3) Suitable soft components

Operand	Word soft component											Bit soft component						
	System								Constant	Module		System						
	D	FD	TD	CD	DX	DY	DM	DS	K/H	ID	QD	X	Y	M	S	T	C	Dn.m
Two operands																		
S1	●	●							●									
S2	●	●							●									
Three operands																		
S1	●	●							●									
S2	●	●							●									
S3									●									

Note: D represents D, HD; TD represents TD, HTD; CD represents CD, HCD, HSCD, HSD; DM represents DM, DHM; DS represents DS, DHS.

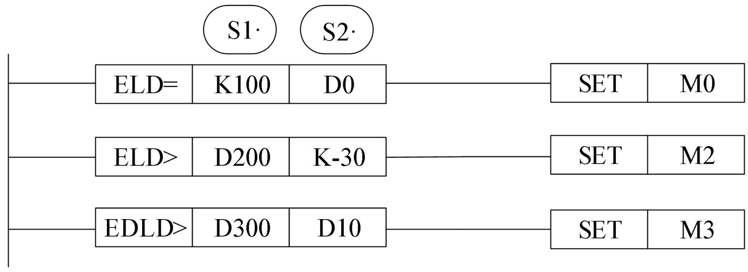
M represents M, HM, SM; S represents S, HS; T represents T, HT; C represents C, HC.

(4) Function and action

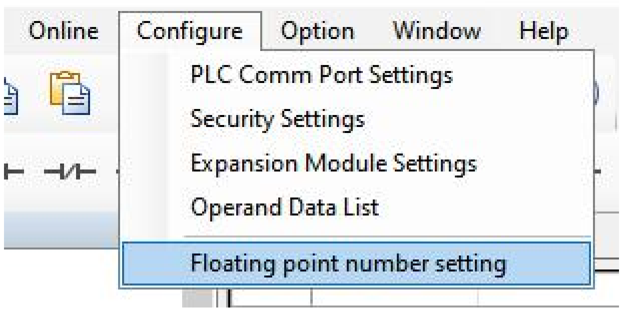
Two operands:

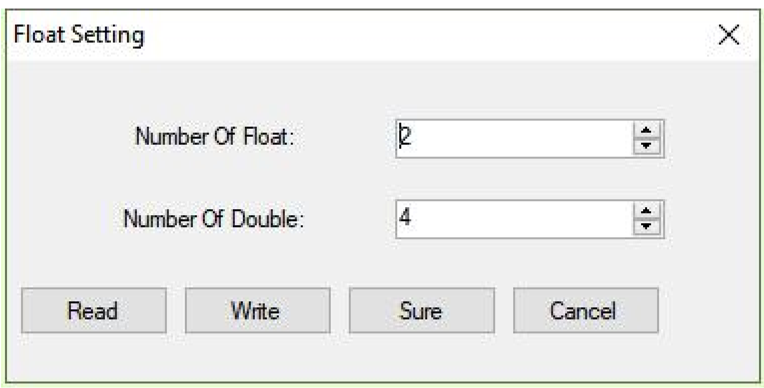
32-bit instruction	64-bit instruction	Conduction conditions	Non conducting condition
ELD＝	EDLD＝	(|S1-S2|)<= absolute error	(|S1-S2|)> absolute error
ELD＞	EDLD＞	(S1-S2)> absolute error	(S1-S2)<= absolute error

ELD<	EDLD<	(S1-S2)<- absolute error	(S1-S2)>=- absolute error
ELD<>	EDLD<>	(|S1-S2|)> absolute error	(|S1-S2|)<= absolute error
ELD<=	EDLD<=	(S1-S2)<= absolute error	(S1-S2)> absolute error
ELD>=	EDLD>=	(S1-S2)>=- absolute error	(S1-S2)<- absolute error



- The "absolute error" in the above two operand tables refers to the precision of the comparison bits. For example, if the floating-point comparison set for 32-bit floating-point comparison is 2, its absolute error is 0.01.
- 32-bit instructions use "floating-point comparison bits" as absolute errors, stored in SFD8; The 64 bits instruction "double precision comparison bit" is stored as an absolute error in SFD9.
- The number of comparison bits can be set in the software "Configure" - "Floating point number settings"; The range of floating-point comparison bits for 32-bit instructions is 1-7, and the range of double precision comparison bits for 64 bits instructions is 1-15. Effective immediately after writing.



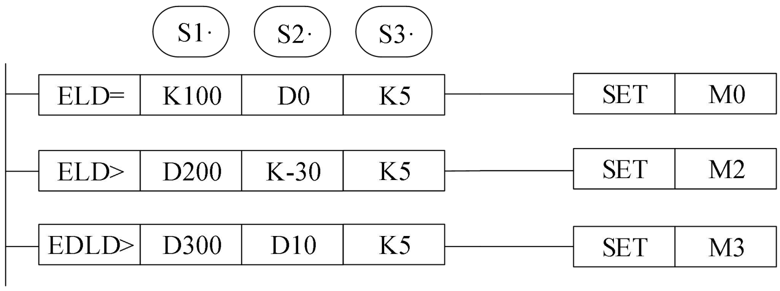


- When the highest bit of the source data (32-bit: b31, 64-bit: b63) is 1, compare the value as a negative number.

- The comparison of 64-bit floating-point numbers must be performed using 64-bit instructions. If you choose the 32-bit instruction, it will cause program errors or arithmetic errors.

Three operands:

32-bit instruction	64-bit instruction	Conduction conditions	Non conducting condition
ELD＝	EDLD＝	(|S1-S2|)<=S3	(|S1-S2|)>S3
ELD＞	EDLD＞	(S1-S2)>S3	(S1-S2)<=S3
ELD＜	EDLD＜	(S1-S2)<-S3	(S1-S2)>=-S3
ELD＜＞	EDLD＜＞	(|S1-S2|)>S3	(|S1-S2|)<=S3
ELD＜＝	EDLD＜＝	(S1-S2)<=S3	(S1-S2)>S3
ELD＞＝	EDLD＞＝	(S1-S2)>=-S3	(S1-S2)<-S3



- Range of operand S3: 32-bit instructions 1-7, 64-bit instructions 1-15.
- When the highest bit of the source data (32-bit: b31, 64-bit: b63) is 1, compare the value as a negative number.
- When the 32-bit instruction S3 is less than 1 or greater than 7, an error will be reported in the error list during operation, which does not affect the PLC operation. It will run with default accuracy (default accuracy 1E-2); The error message is: the software component instruction range exceeds K1-K7.
- When the 64-bit instruction S3 is less than 1 or greater than 15, an error will be reported in the error list during operation, which does not affect the PLC operation. It will run with default accuracy (default accuracy 1E-4); The error message is: the software component instruction range exceeds K1-K15.
- The comparison of 64-bit floating-point numbers must be performed using 64-bit instructions. If a 32-bit instruction is specified, it will cause program errors or arithmetic errors.

4-4-2 Serial Compare

4-4-2-1 Integer serial compare [AND],[DAND]

(1) Summary

AND: serial connection comparison instruction.

AND Compare [AND]			
16 bits	As Below	32 bits	As Below
Execution condition	Normally ON/OFF coil	Suitable Models	XD, XL
Hardware requirement	-	Software requirement	-

(2) Operands

Operands	Function	Data Type
S1	Being compared number address	16/32bit, BIN; INT/DINT
S2	Comparand address	16/32bit, BIN; INT/DINT

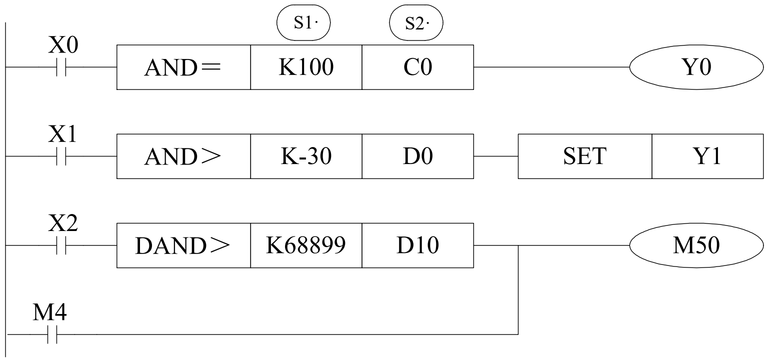
(3) Suitable soft components

Operand	Word soft elements											Bit soft elements						
	System								Constant	Module		System						
	D	FD	TD	CD	DX	DY	DM	DS	K/H	ID	QD	X	Y	M	S	T	C	Dn.m
S1	●	●	●	●	●	●	●	●	●									
S2	●	●	●	●	●	●	●	●	●									

*Notes: D includes D, HD; TD includes TD, HTD; CD includes CD, HCD, HSCD, HSD; DM includes DM, DHM; DS includes DS, DHS. M includes M,HM,SM； S includes S, HS； T includes T,HT； C includes C, HC.

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)



Note Items

When the source data's highest bit (16 bits: b15,32 bits: b31) is 1, it is seemed to negative number.
The comparison of 32 bits counter should use 32 bits instruction. If using16 bits instruction, the program or operation will be error.

4-4-2-2 Floating number serial compare [EAND],[EDAND]

(1) Instruction overview

EAND is a comparison instruction that is connected in series with other contacts.

Serial compare [EAND]			
32 bits instruction	Described below	64 bits instruction	Described below
Execution condition	-	Suitable model	XDH, XLH, XG2
Firmware	V3.7.3 and above	Software	V3.7.16 and above

(2) Operand

Operand	Function	Type
Two operands		
S1	Specify the value or software component address of the number to be compared	32/64 bits, BIN; REAL/LREAL
S2	Specify the value or software component address of the number to be compared	32/64 bits, BIN; REAL/LREAL
Three operands		
S1	Specify the value or software component address of the number to be compared	32/64 bits, BIN; REAL/LREAL
S2	Specify the value or software component address of the number to be compared	32/64 bits, BIN; REAL/LREAL
S3	Specify the numerical value of absolute error during comparison $1*10^{-S3}$	16 bits, BIN

(3) Suitable soft component

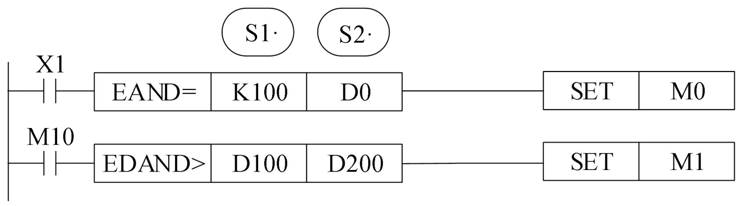
Operand	Word soft elements											Bit soft elements						
	System								Constant	Module		System						
	D	FD	TD	CD	DX	DY	DM	DS	K/H	ID	QD	X	Y	M	S	T	C	Dn.m
Two operands																		
S1	●	●							●									
S2	●	●							●									
Three operands																		
S1	●	●							●									
S2	●	●							●									
S3									●									

Note: D represents D, HD; TD represents TD, HTD; CD represents CD, HCD, HSCD, HSD; DM represents DM, DHM; DS represents DS, DHS.
M represents M, HM, SM; S represents S, HS; T represents T, HT; C represents C, HC.

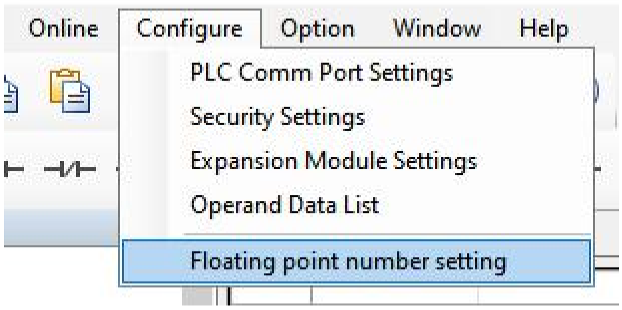
(4) Function and action

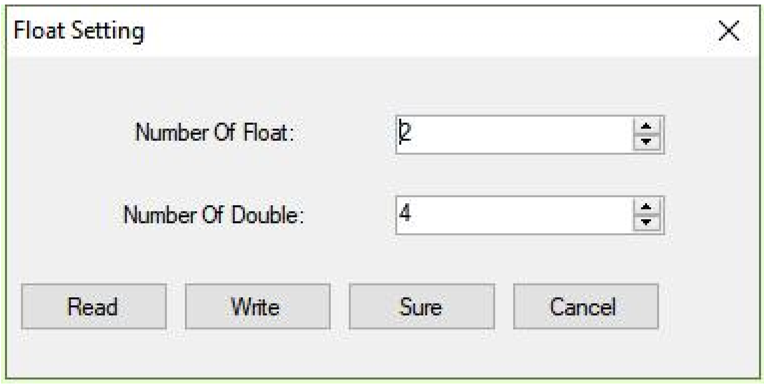
Two operands

32-bit instruction	64-bit instruction	Conduction conditions	Non conducting condition
EAND＝	EDAND＝	(|S1-S2|)<= absolute error	(|S1-S2|)> absolute error
EAND＞	EDAND＞	(S1-S2)> absolute error	(S1-S2)<= absolute error
EAND＜	EDAND＜	(S1-S2)<- absolute error	(S1-S2)>=- absolute error
EAND＜＞	EDAND＜＞	(|S1-S2|)> absolute error	(|S1-S2|)<= absolute error
EAND＜＝	EDAND＜＝	(S1-S2)<= absolute error	(S1-S2)> absolute error
EAND＞＝	EDAND＞＝	(S1-S2)>=- absolute error	(S1-S2)<- absolute error



- The "absolute error" in the above two operand tables refers to the precision of the comparison bits. For example, if the floating-point comparison bit set for 32-bit floating-point comparison is 2, its absolute error is 0.01.
- 32-bit instructions use "floating-point comparison bits" as absolute errors, stored in SFD8; The 64 bits instruction "double precision comparison bit" is stored as an absolute error in SFD9.
- The number of comparison bits can be set in the software "Configure" - "Floating point number settings"; The range of floating-point comparison bits for 32-bit instructions is 1-7, and the range of double precision comparison bits for 64 bits instructions is 1-15. Effective immediately after writing.

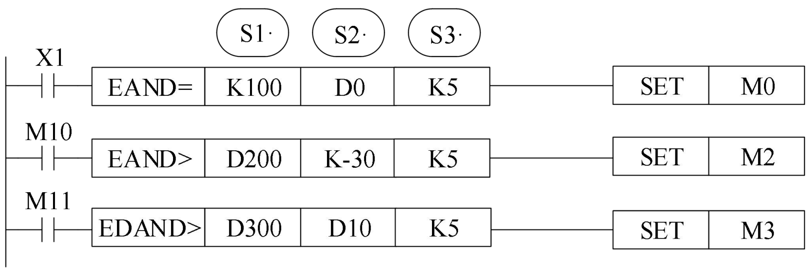




- When the highest bit of the source data (16 bits instruction: b15, 32 bits: b31) is 1, compare the value as a negative number.
- The comparison of 64 bits counters must be performed using 64 bits instructions. If 32-bit instructions are specified, it will cause program errors or arithmetic errors.

Three operands:

32-bit instruction	64-bit instruction	Conduction conditions	Non conducting condition
EAND＝	EDAND＝	(|S1-S2|)<=S3	(|S1-S2|)>S3
EAND＞	EDAND＞	(S1-S2)>S3	(S1-S2)<=S3
EAND＜	EDAND＜	(S1-S2)<-S3	(S1-S2)>=-S3
EAND＜＞	EDAND＜＞	(|S1-S2|)>S3	(|S1-S2|)<=S3
EAND＜＝	EDAND＜＝	(S1-S2)<=S3	(S1-S2)>S3
EAND＞＝	EDAND＞＝	(S1-S2)>=-S3	(S1-S2)<-S3



- Range of operands S3: 32-bit instructions 1-7, 64-bit instructions 1-15.
- When the highest bit of the source data (32-bit: b31, 64-bit: b63) is 1, compare the value as a negative number.
- When the 32-bit instruction S3 is less than 1 or greater than 7, an error will be reported in

the error list during operation, which does not affect the PLC operation. It will run with default accuracy (default accuracy 1E-2); The error message is: The software component instruction range exceeds K1-K7.

- When the 64-bit instruction S3 is less than 1 or greater than 15, an error will be reported in the error list during operation, which does not affect the PLC operation. It will run with default accuracy (default accuracy 1E-4); The error message is: The software component instruction range exceeds K1-K15.
- The comparison of 64-bit floating-point numbers must be performed using 64-bit instructions. If a 32-bit instruction is specified, it will cause program errors or arithmetic errors.

4-4-3 Parallel Compare

4-4-3-1 Integer parallel compare [OR]

(1) Summary

OR: parallel connection comparison instruction.

Parallel Compare [OR]			
16 bits	As below	32 bits	As below
Execution condition	-	Suitable Models	XD, XL
Hardware requirement	-	Software requirement	-

(2) Operands

Operands	Function	Data Type
S1	Being compared number address	16/32 bits,BIN; INT/DINT
S2	Comparand address	16/32 bits,BIN; INT/DINT

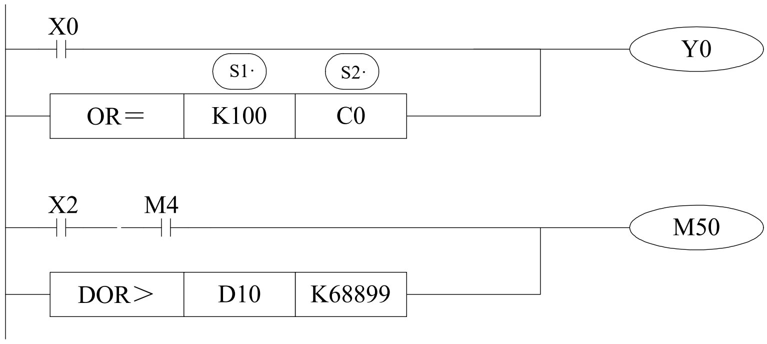
(3) Suitable soft components

Operands	Word soft elements											Bit soft elements						
	System								Constant	Module		System						
	D	FD	TD	CD	DX	DY	DM	DS	K/H	ID	QD	X	Y	M	S	T	C	Dn.m
S1	●	●	●	●	●	●	●	●	●									
S2	●	●	●	●	●	●	●	●	●									

*Notes: D includes D, HD; TD includes TD, HTD; CD includes CD, HCD, HSCD, HSD; DM includes DM, DHM; DS includes DS, DHS. M includes M,HM,SM； S includes S,HS； T includes T,HT； C includes C, HC.

Description

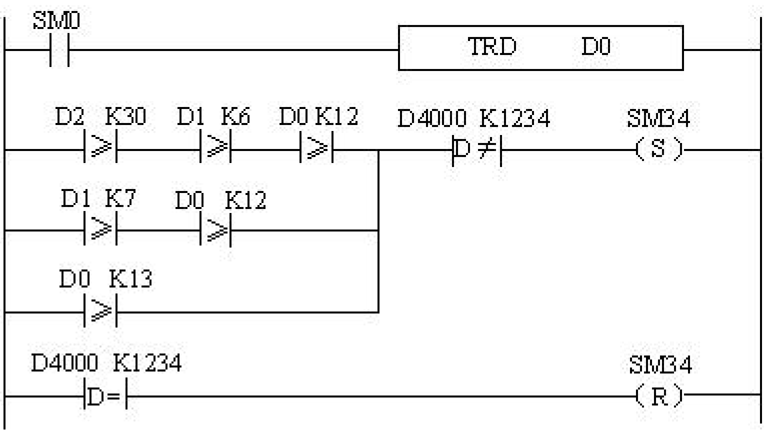
16 bits instruction	32 bits instruction	Activate Condition	Not Activate Condition
OR＝	DOR＝	(S1)＝ (S2)	(S1)≠ (S2)
OR＞	DOR＞	(S1)＞ (S2)	(S1)≤ (S2)
OR＜	DOR＜	(S1)＜ (S2)	(S1)≥ (S2)
OR＜＞	DOR＜＞	(S1)≠ (S2)	(S1)＝ (S2)
OR＜＝	DOR＜＝	(S1)≤ (S2)	(S1)＞ (S2)
OR＞＝	DOR＞＝	(S1)≥ (S2)	(S1)＜ (S2)



Note Items

- When the source data's highest bit (16 bits: b15,32 bits: b31) is 1, it is seemed to negative number.
- The comparison of 32 bits counter should use 32 bits instruction. If using16 bits instruction, the program or operation will be error.

Example: forbid the outputs when it reaches the certain time. In the below program, when the date is June 30th, 2012, all the outputs will be disabled. The password 1234 is stored in (D4000, D4001). When the password is correct, all the outputs are enabled.




```
LD      SM0              //SM0 is always ON coil
TRD     D0               //read the RTC (real time clock) value and store in D0~D6
LD>=    D2      K30      //RTC date ≥30
```

```
AND>=       D1      K6              //RTC month ≥6
AND>=       D0      K12             //RTC year ≥12
LD>=  D1    K7              //or RTC month ≥ 7
AND>=       D0      K12             //RTC year ≥ 12
ORB                 //or
OR>=  D0    K13             //RTC year ≥ 13
DAND<>      D4000 K1234       //and password ≠1234
SET     SM34                //set ON M34, all the outputs are disabled
DLD=  D4000  K1234          //password=1234, correct password
RST    SM34                 //reset M34, all the outputs are enabled
```

4-4-3-2. Floating number parallel compare [EOR]

(1) Instruction overview

EOR is a command to compare contacts in parallel with other contacts.

Parallel compare [EAND]			
32-bit instruction	Described below	64-bit instruction	Described below
Execution condition	-	Suitable model	XDH, XLH, XG2
Firmware	V3.7.3 and above	Software	V3.7.16 and above

(2) Operand

Operand	Function	Type
Two operands		
S1	Specify the numerical value or software component address of the number to be compared	32/64 bits, BIN; REAL/LREAL
S2	Specify a numerical value or software component address for comparison	32/64 bits, BIN; REAL/LREAL
Three operands		
S1	Specify the numerical value or software component address of the number to be compared	32 bits/64 bits; REAL/LREAL
S2	Specify a numerical value or software component address for comparison	32 bits/64 bits; REAL/LREAL
S3	Specify the numerical value of absolute error during comparison $1*10^{-S3}$	16 bits, BIN

(3) Suitable soft component

Operand	Word soft component											Bit soft component						
	System								Constant	Module		System						
	D	FD	TD	CD	DX	DY	DM	DS	K/H	ID	QD	X	Y	M	S	T	C	Dn.m
Two operands																		
S1	●	●							●									
S2	●	●							●									
Three operands																		
S1	●	●							●									

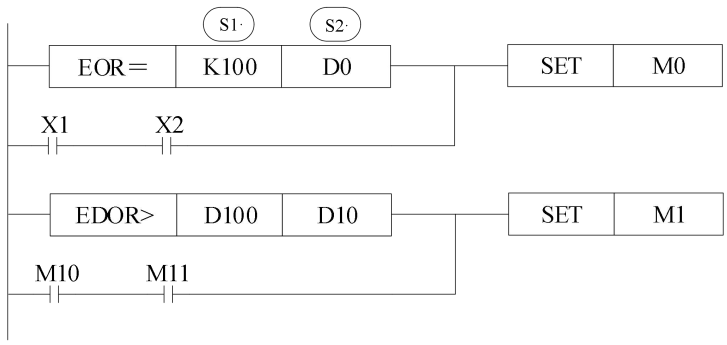
S2	●	●							●									
S3									●									

Note: D represents D, HD; TD represents TD, HTD; CD represents CD, HCD, HSCD, HSD; DM represents DM, DHM; DS represents DS, DHS.
M represents M, HM, SM; S represents S, HS; T represents T, HT; C represents C, HC.

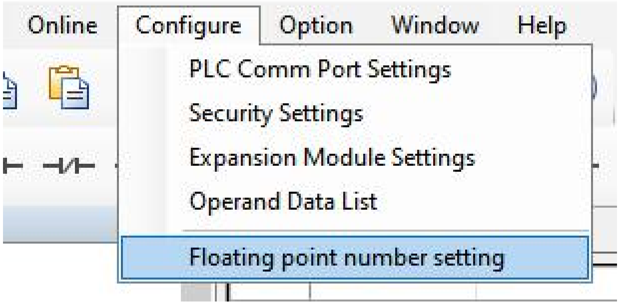
(4) Function and action

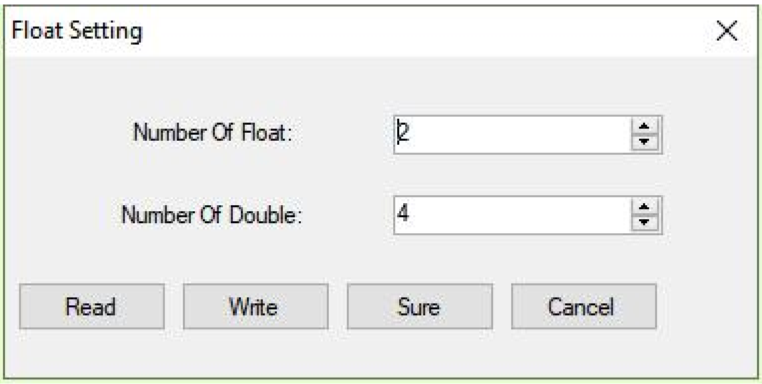
Two operands:

16 bits instruction	32 bits instruction	Activate Condition	Not Activate Condition
EOR＝	EDOR＝	(|S1-S2|)<= absolute error	(|S1-S2|)> absolute error
EOR＞	EDOR＞	(S1-S2)> absolute error	(S1-S2)<= absolute error
EOR＜	EDOR＜	(S1-S2)<- absolute error	(S1-S2)>=- absolute error
EOR＜＞	EDOR＜＞	(|S1-S2|)> absolute error	(|S1-S2|)<= absolute error
EOR＜＝	EDOR＜＝	(S1-S2)<= absolute error	(S1-S2)> absolute error
EOR＞＝	EDOR＞＝	(S1-S2)>=- absolute error	(S1-S2)<- absolute error



- The "absolute error" in the above two operand tables refers to the precision of the comparison bits. For example, if the floating-point comparison set for 32-bit floating-point comparison is 2, its absolute error is 0.01.
- 32-bit instructions use "floating-point comparison bits" as absolute errors, stored in SFD8; The 64-bit instruction "double precision comparison bit" is stored as an absolute error in SFD9.
- The number of comparison bits can be set in the software "Configure" - "Floating point number settings"; The range of floating-point comparison bits for 32-bit instructions is 1-7, and the range of double precision comparison bits for 64-bit instructions is 1-15. Effective immediately after writing.

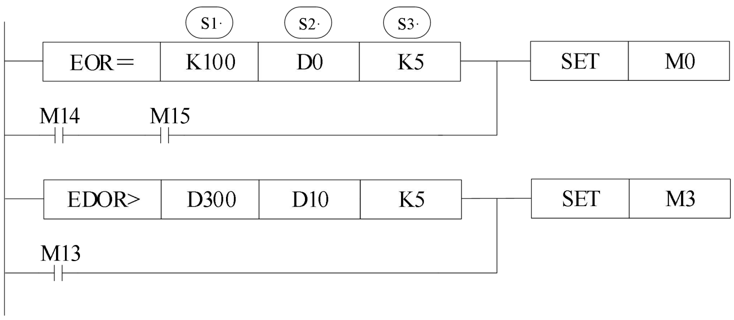




- When the highest bit of the source data (16 bits instruction: b15, 32-bit: b31) is 1, compare the value as a negative number.
- The comparison of 64 bits counters must be performed using 64 bits instructions. If a 32-bit instruction is specified, it will cause program errors or arithmetic errors.

Three operands:

32-bit instruction	64-bit instruction	Conduction condition	Non-conductive condition
EOR＝	EDOR＝	(|S1-S2|)<=S3	(|S1-S2|)>S3
EOR＞	EDOR＞	(S1-S2)>S3	(S1-S2)<=S3
EOR＜	EDOR＜	(S1-S2)<-S3	(S1-S2)>=-S3
EOR＜＞	EDOR＜＞	(|S1-S2|)>S3	(|S1-S2|)<=S3
EOR＜＝	EDOR＜＝	(S1-S2)<=S3	(S1-S2)>S3
EOR＞＝	EDOR＞＝	(S1-S2)>=-S3	(S1-S2)<-S3



- Operand S3 range: 32-bit instructions 1-7, 64 bits instructions 1-15.
- When the highest bit of the source data (32-bit: b31, 64-bit: b63) is 1, compare the value as a negative number.
- When the 32-bit instruction S3 is less than 1 or greater than 7, an error will be reported in

the error list during operation, which does not affect the operation of the PLC and will run at the default precision (default precision 1E-2); The error message is: The software component instruction range exceeds K1-K7.

- When the 64-bit instruction S3 is less than 1 or greater than 15, an error will be reported in the error list during operation, which does not affect the operation of the PLC and will run at the default precision (default precision 1E-4); The error message is: The software component instruction range exceeds K1-K15.
- The comparison of 64-bit floating-point numbers must be performed using 64-bit instructions. If 32-bit instructions are specified, it will cause program errors or arithmetic errors.

4-5 Data Move Instructions

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
EMOV	Float number move	4-5-7
FWRT	FlashROM written	4-5-8
MSET	Zone set	4-5-9
ZRST	Zone reset	4-5-10
SWAP	The high and low byte of the destinated devices are exchanged	4-5-11
XCH	Exchange two data	4-5-12

4-5-1 Data Compare [CMP, DCMP, QCMP]

1) Summary

Compare the two data, output the result.

Data Compare [CMP,DCMP,QCMP]			
16 bits	CMP	32 bits	DCMP
Execution condition	Normally ON/OFF, rising/falling edge	Suitable Models	XD, XL
Hardware requirement	-	Software requirement	-
64 bits	QCMP		
Execution condition	Normal ON/OFF/falling or rising pulse edge	Suitable Models	XDH, XLH
Hardware requirement	Version V3.7.1 or later	Software requirement	Version V3.7.4a or later

2) Operands

Operands	Function	Data Type
S1	Specify the data (to be compared) or soft component's address code	16/32/64 bits,BIN
S	Specify the comparand's value or soft component's address code	16/32/64 bits,BIN
D	Specify the compare result's address code	bit

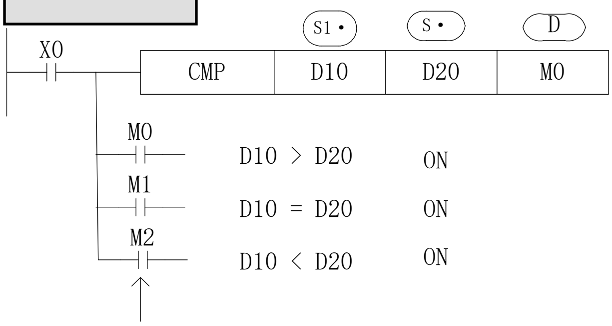
3) Suitable soft component

Operands	Word soft elements											Bit soft elements						
	System								Constant	Module		System						
	D	FD	TD	CD	DX	DY	DM	DS	K/H	ID	QD	X	Y	M	S	T	C	Dn.m
S1	●	●	●	●	●	●	●	●	●									
S2	●	●	●	●	●	●	●	●	●									

D													●	●	●			

*Notes: D includes D, HD; TD includes TD, HTD; CD includes CD, HCD, HSCD, HSD; DM includes DM, DHM; DS includes DS, DHS. M includes M,HM,SM； S includes S,HS； T includes T,HT； C includes C, HC.

Description



Even X0=OFF to stop CMP instruction,
M0~M2 will keep the original status

- Compare data (S1·) and (S·), show the result in three soft components starting from (D·)
- (D·), (D·)+1, (D·)+2: the three soft components will show the compare result.
- Note:The addresses of operands in QCMP instructions must be even.

4-5-2 Data zone compare [ZCP, DZCP]

1) Summary

Compare the current data with the data in the zone, output the result.

Data Zone compare [ZCP, DZCP]			
16 bits	ZCP	32 bits	DZCP
Execution condition	Normally ON/OFF, rising/falling edge	Suitable Models	XD, XL
Hardware requirement	-	Software requirement	-

2) Operands

Operands	Function	Data Type
S1	The low limit of zone	16/32 bits, BIN
S2	The high limit of zone	16/32 bits, BIN
S	The current data address	16/32 bits, BIN
D	The compare result	bit

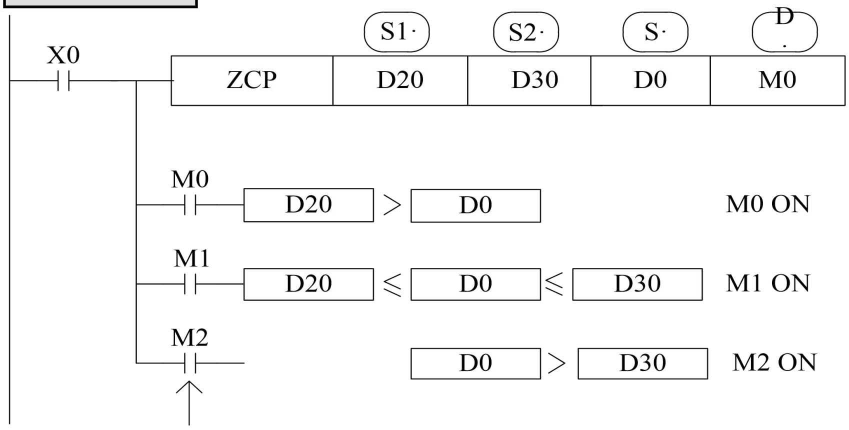
3) Suitable soft components

Operands	Word soft elements											Bit soft elements						
	System								Constant	Module		System						
	D	FD	TD	CD	DX	DY	DM	DS	K/H	ID	QD	X	Y	M	S	T	C	Dn.m
S1	●	●	●	●	●	●	●	●	●									

S2	●	●	●	●	●	●	●	●	●									
S	●	●	●	●	●	●	●	●	●									
D													●	●	●			

*Notes: D includes D, HD; TD includes TD, HTD; CD includes CD, HCD, HSCD, HSD; DM includes DM, DHM; DS includes DS, DHS. M includes M,HM,SM； S includes S,HS； T includes T,HT； C includes C, HC.

Description



Even X0=OFF stop ZCP instruction, M0~M2 will keep the original status

- Compare S· with S1 and S2 , output the three results starting from D·
- D· , D· +1, D· +2 : store the three results.

4-5-3 MOV [MOV, DMOV, QMOV]

1) Summary

Move the specified data to the other soft components

MOV [MOV,DMOV,QMOV]			
16 bits	MOV	32 bits	DMOV
Execution condition	Normally ON/OFF, rising/falling edge	Suitable Models	XD, XL
Hardware requirement	-	Software requirement	-
64 bits	QMOV		
Execution condition	Normal ON/OFF/falling or rising pulse edge	Suitable Models	XDH, XLH
Hardware requirement	Version V3.7.1 or later	Software requirement	Version V3.7.4a or later

2) Operands

Operands	Function	Data Type
S	Specify the source data or register's address code	16 bits/32 bits/64 bits, BIN; INT/DINT/LINT
D	Specify the target soft component's address code	16 bits/32 bits/64 bits, BIN; INT/DINT/LINT

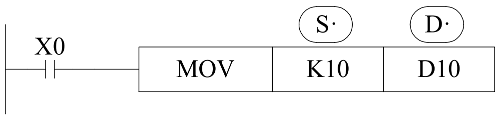
3) Suitable soft component

Operands	Word soft elements											Bit soft elements						
	System								Constant	Module		System						
	D	FD	TD	CD	DX	DY	DM	DS	K/H	ID	QD	X	Y	M	S	T	C	Dn.m
S	●	●	●	●	●	●	●	●	●	●								
D	●		●	●		●	●	●			●							

*Notes: D includes D, HD; TD includes TD, HTD; CD includes CD, HCD, HSCD, HSD; DM includes DM, DHM; DS includes DS, DHS. M includes M,HM,SM； S includes S,HS； T includes T,HT； C includes C, HC.

Description

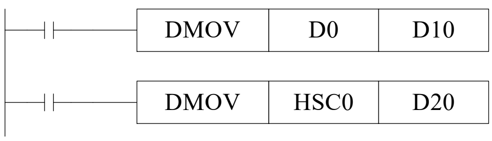
< Move 16 bits data >



- Move the source data to the target
- When X0 is off, the data will not change
- Move K10 to D10

< Move 32 bits data >

Please use DMOV when the value is32 bits, such as MUL instruction, high speed counter…

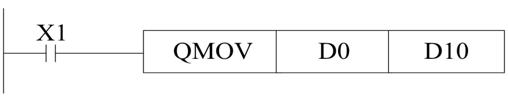


(D1, D0) → (D11, D10)

(the current value of HSC0)→(D21, D20)

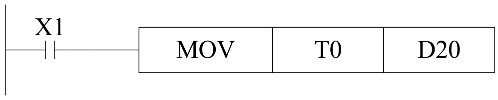
< Move 64 bits data >

Please use QMOV when the value is64 bits, such as DMUL instruction



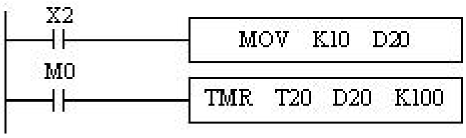
(D3,D2,D1,D0)→(D13,D12,D11,D10)

<read the counter or timer current value>



(The current value of T0)→(D20)
The same as counter

<indirect set the timer value>



(K10) (D20)
D20=K10

4-5-4 Data block Move [BMOV]

1) Summary

Move the data block to other soft component

Data block move [BMOV]			
16 bits	BMOV	32 bits	-
Execution condition	Normally ON/OFF coil, rising/falling edge	Suitable Models	XD, XL
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

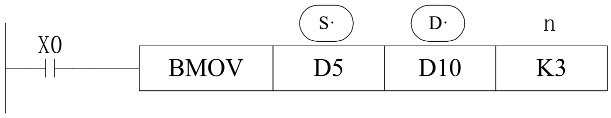
Operands	Word soft elements											Bit soft elements						
	System								Constant	Module		System						
	D	FD	TD	CD	DX	DY	DM	DS	K/H	ID	QD	X	Y	M	S	T	C	Dn.m
S	●	●	●	●	●	●	●	●					●	●	●			
D	●		●	●		●	●	●					●	●	●			
n	●		●	●	●		●	●	●									

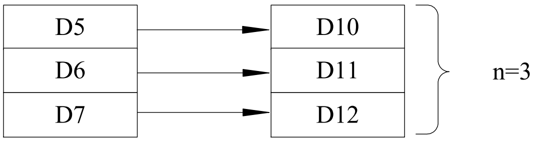
*Notes: D includes D, HD; TD includes TD, HTD; CD includes CD, HCD, HSCD, HSD; DM includes DM, DHM; DS includes DS, DHS. M includes M,HM,SM； S includes S,HS； T includes T,HT； C includes C, HC.

Description

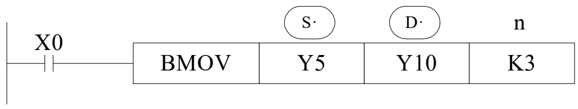
Move the source data block to the target data block. The data quantity is n.

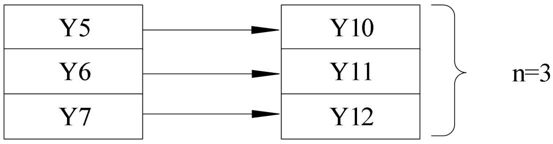
<word move>



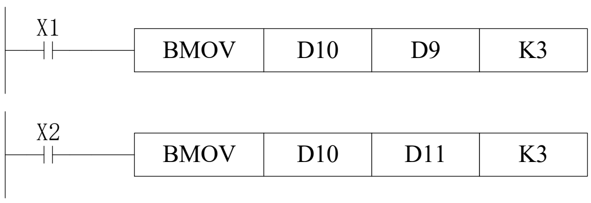


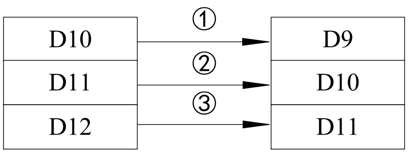
<bit move>

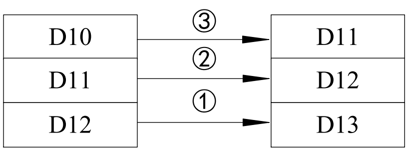




As shown in the figure below, when the transmission number range overlaps, in order to prevent the transmission source data from being overwritten without transmission, according to the method of number overlap, this instruction will be carried out in the order of ① ~ ③.







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, rising/falling edge	Suitable Models	XD, XL
Hardware requirement	-	Software requirement	-

2) Operands

Operands	Function	Data Type
S	Specify the source data block or soft component address	16 bits, BIN;
D	Specify the target soft components address	16 bits, BIN;
n	Specify the data quantity	16 bits, BIN;

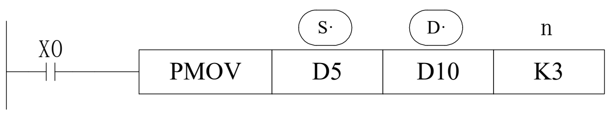
3) Suitable soft components

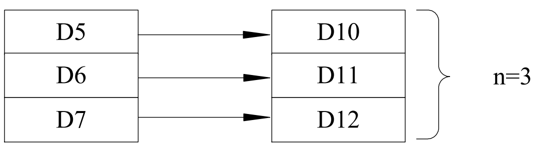
Operands	Word soft elements											Bit soft elements						
	System								Constant	Module		System						
	D	FD	TD	CD	DX	DY	DM	DS	K/H	ID	QD	X	Y	M	S	T	C	Dn.m
S	●																	
D	●																	
n	●		●	●	●		●	●	●									

*Notes: D includes D, HD; TD includes TD, HTD; CD includes CD, HCD, HSCD, HSD; DM includes DM, DHM; DS includes DS, DHS. M includes M,HM,SM； S includes S,HS； T includes T,HT； C includes C, HC.

Description

Move the source data block to target data block, the data quantity is n





The function of PMOV and BMOV is mostly the same, but the PMOV execution speed is faster.

PMOV finish in one scan cycle, when executing PMOV, close all the interruptions.

Mistake may happen if the source address and target address are overlapped.

4-5-6 Fill Move [FMOV, DFMOV]

1) Summary

Move the specified data to the other soft components

Fill Move [FMOV, DFMOV]			
16 bits	FMOV	32 bits	DFMOV
Execution condition	Normally ON/OFF, rising/falling edge	Suitable Models	XD, XL
Hardware requirement	-	Software requirement	-

2) Operands

Operands	Function	Data Type
S	Specify the source data or soft component address	16/32 bits, BIN;
D	Specify the target soft components address	16/32 bits, BIN;
n	Specify the move data's number	16 bits, BIN;

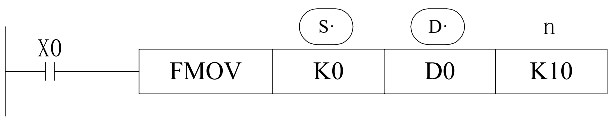
3) Suitable soft component

Operands	Word soft elements											Bit soft elements						
	System								Constant	Module		System						
	D	FD	TD	CD	DX	DY	DM	DS	K/H	ID	QD	X	Y	M	S	T	C	Dn.m
S	●	●	●	●	●	●	●	●	●									
D	●		●	●		●	●	●										
n	●		●	●		●	●	●	●									

*Notes: D includes D, HD; TD includes TD, HTD; CD includes CD, HCD, HSCD, HSD; DM includes DM, DHM; DS includes DS, DHS. M includes M,HM,SM； S includes S,HS； T includes T,HT； C includes C, HC.

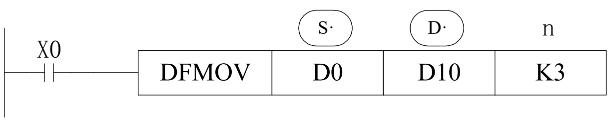
Description

<16 bits instruction>



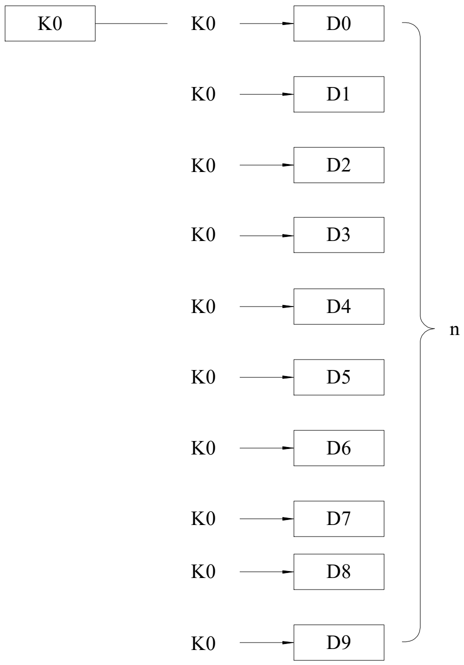
- Move K0 to D0~D9, copy a single data device to a range of destination device
- Move the source data to target data, the target data quantity is n
- If the set range exceeds the target range, move to the possible range

<32 bits instruction >

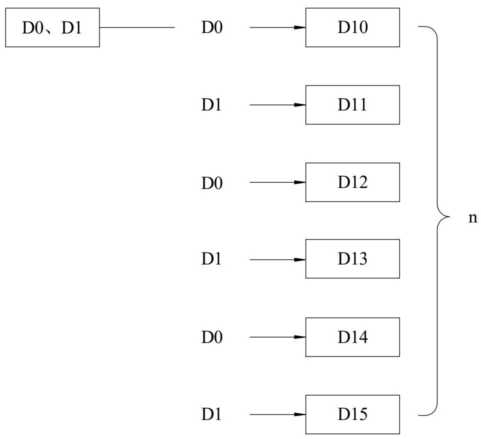


- Move D0.D1 to D10.D11:D12.D13:D14.D15.

<16 bits data transfer >



<32 bits data transfer >



4-5-7 Floating move [EMOV, EDMOV]

1)Summary

Move the float number to target address

Floating move [EMOV, EDMOV]			
16 bits	-	32 bits	EMOV
Execution condition	Normally ON/OFF, rising/falling edge	Suitable Models	XD, XL
Hardware requirement	-	Software requirement	-
64 bits	EDMOV		
Execution condition	Normal ON/OFF/falling or rising pulse edge	Suitable Models	XDH, XLH
Hardware requirement	Version V3.7.1 or later	Software requirement	Version V3.7.4a or later

2)Operands

Operand	Function	Type
S	Source soft element address	32 /64bits, BIN; REAL/LREAL
D	Destination soft element address	32 /64bits, BIN; REAL/LREAL

3)Suitable soft element

Operands	Word soft elements											Bit soft elements						
	System								Constant	Module		System						
	D	FD	TD	CD	DX	DY	DM	DS	K/H	ID	QD	X	Y	M	S	T	C	Dn.m
S	●	●			●	●	●	●	●									
D	●					●	●	●										

*Notes: D includes D, HD; TD includes TD, HTD; CD includes CD, HCD, HSCD, HSD; DM includes DM, DHM; DS includes DS, DHS. M includes M,HM,SM； S includes S,HS； T includes T,HT； C includes C, HC.

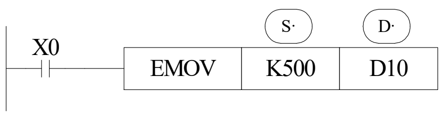
Description

<32 bits instruction>

Binary floating → binary floating

(D1,D0) → (D11,D10)

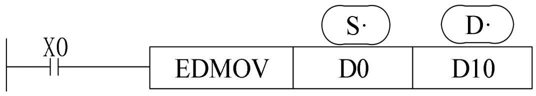
- X0 is ON, send the floating number from (D1, D0) to (D11, D10).
- X0 is OFF, the instruction doesn't work.



(K500) → (D11,D10)

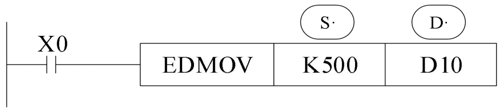
- If constant value K, H is source soft element, they will be converted to floating number.
- K500 will be converted to floating value.

<64 bits instruction>



(D3,D2,D1,D0)→(D13,D12,D11,D10)

- X0 is ON, send the floating number from (D3,D2,D1,D0)to(D13,D12,D11,D10).
- X0 is OFF, the instruction doesn't work.



(K500)→(D13,D12,D11,D10)

- If constant value K, H is source soft element, they will be converted to floating number.
- K500 will be converted to floating value.
- The addresses of operands in EDMOV instructions must be even.

4-5-8 FlashROM Write [FWRT, DFWRT, QFWRT]

1) Summary

Write the specified data to FlashROM register.

FlashROM Write [FWRT,DFWRT,QFWRT]			
16 bits	FWRT	32 bits	DFWRT
Execution condition	Normally ON/OFF, rising/falling edge	Suitable Models	XD, XL
Hardware requirement	-	Software requirement	-
64 bits	QFWRT		
Execution condition	Normal ON/OFF/falling or rising pulse edge	Suitable Models	XDH, XLH
Hardware requirement	Version V3.7.1 or later	Software requirement	Version V3.7.4a or later

2) Operands

Operands	Function	Data Type
S	The data write in the source or save in the soft element	16 /32/64 bits, BIN
D	target soft element	16 /32/64 bits
D1	target soft element start address	16 /32/64 bits
D2	Write in data quantity	16 /32/64 bits, BIN

3) Suitable soft components

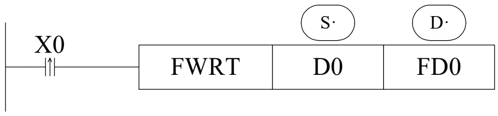
Operands	Word soft elements											Bit soft elements						
	System								Constant	Module		System						
	D	FD	TD	CD	DX	DY	DM	DS	K/H	ID	QD	X	Y	M	S	T	C	Dn.m
S1	●	●	●	●	●	●	●	●	●									
S2		●																
S		●																

D	●		●	●	●	●	●	●	●									

*Notes: D includes D, HD; TD includes TD, HTD; CD includes CD, HCD, HSCD, HSD; DM includes DM, DHM; DS includes DS, DHS. M includes M,HM,SM； S includes S,HS； T includes T,HT； C includes C, HC.

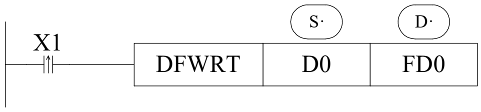
Description

< Written of single word >



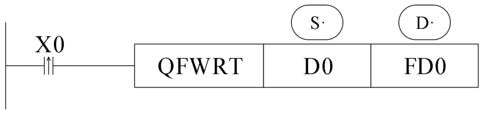
Write value from D0 to FD0

<Written of double words>



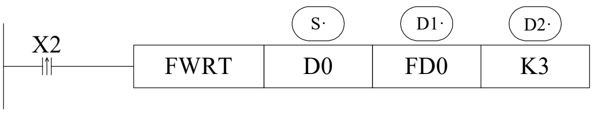
Write value from D0,D1 to FD0,FD1

<Written of four words>



Write value from D0,D1,D2,D3to FD0,FD1,FD2,FD3.

<Written of multi-word>



Write value from D0, D1, D2 to FD0, FD1,FD2

NOTE:

※1: FWRT instruction only can write data into FlashROM register. FlashROM can keep the data even the power supply is off. It can store the important technical parameters.

※2: Written of FWRT needs a long time, about 500ms, so frequently write-in is not recommended

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

※4: Frequently write-in will damage the FlashROM.

4-5-9 Zone set [MSET]

1)Summary

Set the soft element in certain range

Multi-set [MSET]			
16 bits	MSET	32 bits	-
Execution condition	Normally ON/OFF; falling or rising pulse edge signal	Suitable Models	XD, XL
Hardware requirement	-	Software requirement	-

2)Operands

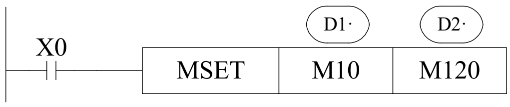
Operands	Function	Data Type
D1	Start soft element address	bit
D2	End soft element address	bit

3) Suitable soft components

Operands	Word soft elements											Bit soft elements						
	System								Constant	Module		System						
	D	FD	TD	CD	DX	DY	DM	DS	K/H	ID	QD	X	Y	M	S	T	C	Dn.m
D1												●	●	●	●	●	●	
D2												●	●	●	●	●	●	

*Notes: D includes D, HD; TD includes TD, HTD; CD includes CD, HCD, HSCD, HSD; DM includes DM, DHM; DS includes DS, DHS. M includes M,HM,SM； S includes S,HS； T includes T,HT； C includes C, HC.

Description



Set ON M10~M120

- Set the coil from M10 to M120.
- D1·、D2· are specified as the same type of soft component, and D1· < D2·
- When D1· >, D2· will not run Zone set, but set SM409 SD409 = 2

4-5-10 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, falling or rising pulse edge	Suitable Models	XD, XL
Hardware requirement	-	Software requirement	-

2) Operands

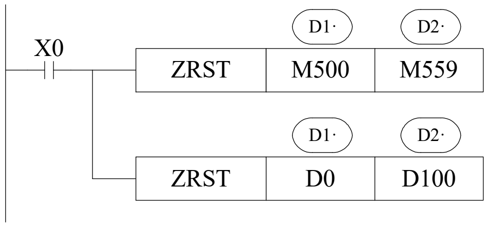
Operands	Function	Data Type
D1	Start address of soft element	Bit,16 bits,BIN
D2	End address of soft element	Bit,16 bits,BIN

3) Suitable soft components

Operands	Word soft elements											Bit soft elements						
	System								Constant	Module		System						
	D	FD	TD	CD	DX	DY	DM	DS	K/H	ID	QD	X	Y	M	S	T	C	Dn.m
D1	●				●	●	●					●	●	●	●	●	●	
D2	●			●	●	●	●					●	●	●	●	●	●	

*Notes: D includes D, HD; TD includes TD, HTD; CD includes CD, HCD, HSCD, HSD; DM includes DM, DHM; DS includes DS, DHS. M includes M,HM,SM;S includes S,HS;T includes T,HT;C includes C, HC.

Description



Reset M500~M559

Reset D0~D100

- D1·、D2· are specified as the same type of soft units, and < D1· D2·
- When D1· > D2· , only reset the specified soft unit, and set SM409, SD409 = 2.

Other Reset Instruction

RST can reset one soft component. The operand can be Y, M, HM, S, HS, T, HT, C, HC, TD, HTD, CD, HCD, D, HD

FMOV can move 0 to these soft components: DX, DY, DM, DS, T(TD), HT(HTD), C(CD), HC(HCD), D, HD.

4-5-11 Swap the high and low byte [SWAP]

1) Summary

Swap the high 8-bit and low 8-bit of specified register

High and low byte swap [SWAP]			
16 bits	SWAP	32 bits	-
Execution condition	Falling or rising pulse edge	Suitable Models	XD, XL
Hardware requirement	-	Software requirement	-

2) Operands

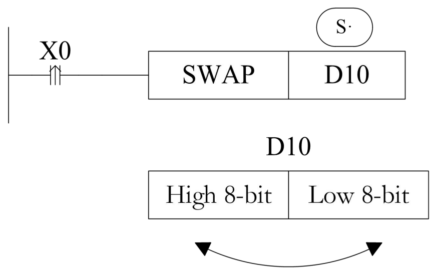
Operands	Function	Data Type
S	The address of the soft element	16 bits; BIN; INT

3) Suitable soft components

Operands	Word soft elements											Bit soft elements						
	System								Constant	Module		System						
	D	FD	TD	CD	DX	DY	DM	DS	K/H	ID	QD	X	Y	M	S	T	C	Dn.m
S	●		●	●														

*Notes: D includes D, HD; TD includes TD, HTD; CD includes CD, HCD, HSCD, HSD; DM includes DM, DHM; DS includes DS, DHS. M includes M,HM,SM;S includes S,HS;T includes T,HT;C includes C, HC.

Description



- Exchange the high 8-bit and low 8-bit of 16-bit register.
- If this instruction is activated by normal ON/OFF coil, the instruction will be executed in every scanning period when X0 is ON. Falling or rising pulse is recommended to activate the instruction.
- For example, the original binary value in register D10 was 0100 1000 0111 0111 (decimal 18551), but after executing the SWAP instruction, the binary value becomes 0111 0111 0100 1000 (decimal 30536).

4-5-12 Exchange [XCH, DXCH]

1) Summary

Exchange the data in two soft element

Exchange [XCH, DXCH]			
16 bits	XCH	32 bits	DXCH
Execution condition	Rising or falling pulse edge	Suitable Models	XD, XL
Hardware requirement	-	Software requirement	-

2) Operands

Operands	Function	Data Type
D1	The soft element address	16 bits/32 bits, BIN; INT/DINT
D2	The soft element address	16 bits/32 bits, BIN; INT/DINT

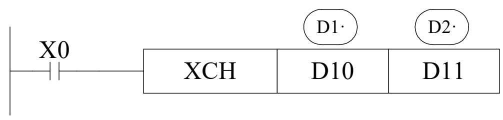
3) Suitable soft component

Operands	Word soft elements											Bit soft elements						
	System								Constant	Module		System						
	D	FD	TD	CD	DX	DY	DM	DS	K/H	ID	QD	X	Y	M	S	T	C	Dn.m
D1	●		●	●		●	●	●										
D2	●		●	●		●	●	●										

*Notes: D includes D, HD; TD includes TD, HTD; CD includes CD, HCD, HSCD, HSD; DM includes DM, DHM; DS includes DS, DHS. M includes M,HM,SM;S includes S,HS;T includes T,HT;C includes C, HC.

Description

<16 bits instruction>

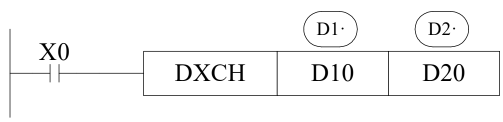


Before (D10) =100 →After (D10) =101

(D11) =101 (D11) =100

- The contents of the two destination devices D1 and D2 are swapped,
- When X0 is ON, the instruction will be executed in every scanning period. Falling or rising pulse is recommended to activate the instruction.

<32 bits instruction >



32 bits instruction [DXCH] swaps the dword value D10,D11 and D20, D21.

Before (D10) =100		→	after (D10) =200	
(D11) =1	(D11D10) =65636		(D11) =10	(D11D10) =655460
(D20) =200			(D20) =100	
(D21) =10	(D21D20) =655460		(D21) =1	(D21D20) =65636

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, DADD, QADD]

1) Summary

Add two numbers and store the result

Add [ADD,DADD, QADD]			
16 bits	ADD	32 bits	DADD
Execution condition	Normal ON/OFF/falling or rising pulse edge	Suitable Models	XD, XL
Hardware requirement	-	Software requirement	-
64 bits	QADD		
Execution condition	Normal ON/OFF/falling or rising pulse edge	Suitable Models	XDH, XLH
Hardware requirement	Version V3.7.1 or later	Software requirement	Version V3.7.4a or later

2) Operands

Operands	Function	Data Type
Three operands		
S1	The add operation data address	16 bits/32 bits/64 bits, BIN; INT/DINT/LINT
S2	The add operation data address	16 bits/32bit/64 bits, BIN; INT/DINT/LINT
D	The result address	16 bits/32bit/64 bits, BIN; INT/DINT/LINT
Two operands		
D	Be Added data and result data address	16 bits/32 bits/64 bits, BIN; INT/DINT/LINT
S1	Add data address	16 bits/32 bits/64 bits, BIN; INT/DINT/LINT

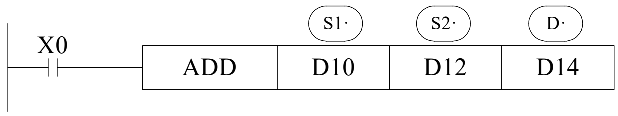
3) Suitable soft components

Operands	Word soft elements											Bit soft elements						
	System								Constant	Module		System						
	D	FD	TD	CD	DX	DY	DM	DS	K/H	ID	QD	X	Y	M	S	T	C	Dn.m
Three operands																		
S1	●	●	●	●	●	●	●	●	●									
S2	●	●	●	●	●	●	●	●	●									
D	●		●	●		●	●	●										
Two operands																		
D	●																	
S1	●	●							●									

*Notes: D includes D, HD; TD includes TD, HTD; CD includes CD, HCD, HSCD, HSD; DM includes DM, DHM; DS includes DS, DHS. M includes M,HM,SM； S includes S,HS； T includes T,HT； C includes C, HC.

Description

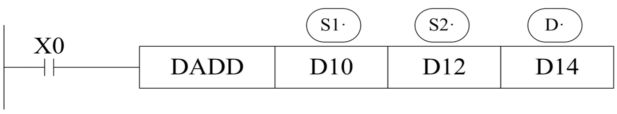
<Three operands>



(D10) ＋ (D12) → (D14)

- Two source data do binary addition and send the result to target address. 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)
- If the result of a calculation is "0", the "0" flag acts. If the result exceeds 323767 (16 bits operation) or 2147483647 (32 bits operation) or 9223372036854775807(64 bits operation), the carry flag acts (refer to the Related flag). If the result exceeds –323768(16 bits operation)or –2147483648 (32 bits operation) or -9223372036854775808(64 bits operation), the borrow flag acts (refer to the Related flag).
- When doing 32/64 bits operation, the lower 16-bit side of the word soft component is specified, and the next numbered soft component will be used as the high position. To avoid ID repetition, it is recommended that the soft component be specified with an even number.

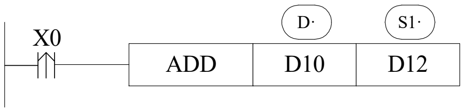
  For example, the 32-bit notation of the preceding example is shown in the following figure. In 32-bit operation, the address of the second addend must start from D12 because the first addend occupies registers D10 and D11. To avoid registers being occupied repeatedly, it is recommended that the soft components be numbered as even numbers.



(D11，D10)＋(D13，D12)→(D15，D14)

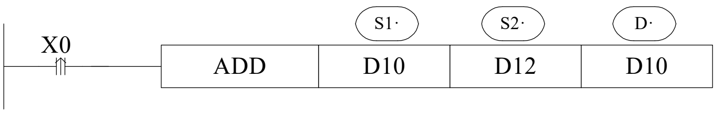
- The source and target address can be the same. In the above example, when X0 is ON, the instruction will be executed in every scanning period.

<Two operands>



$(D10) + (D12) \rightarrow (D10)$

- Two source data do binary addition and send the result to addend data address. 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)
- If the result of a calculation is "0", the "0" flag acts. If the result exceeds 323767 (16 bits operation) or 2147483647 (32 bits operation) or 9223372036854775807(64 bits operation), the carry flag acts (refer to the Related flag). If the result exceeds –323768(16 bits operation)or -2147483648 (32 bits operation) or -9223372036854775808(64 bits operation),the borrow flag acts (refer to the Related flag).
- When doing 32/64 bits operation, the lower 16-bit side of the word soft component is specified, and the next numbered soft component will be used as the high position. To avoid ID repetition, we recommend you assign device's ID to be even number.
- Note:The addresses of operands in QADD instructions must be even.
- In the above example, when X0 is ON, the instruction will be executed in every scanning period. The rising or falling pulse edge is recommended to activate the instruction.



The two instructions are the same.

Related flag

Flag meaning

Flag	Name	Function
SM20	Zero	ON: the calculate result is zero OFF: the calculate result is not zero
SM21	Borrow	ON: the calculate result is over -32768(16 bits) or -2147483648(32 bits) or -9,223,372,036,854,775,808(64 bits), borrowing flag bit action. OFF: the calculate result is less than -32768(16 bits) or -2147483648(32 bits) or -9,223,372,036,854,775,808 (64 bits)
SM22	Carry	ON: the calculate result is over 32768(16 bits) or 2147483648(32 bits) or 9,223,372,036,854,775,807(64 bits), carrying flag bit action. OFF: the calculate result is less than 32768(16 bits) or 2147483648(32 bits) or 9,223,372,036,854,775,807(64 bits)

4-6-2 Subtraction [SUB]

1) Summary

Two numbers do subtraction, store the result

Subtraction [SUB, DSUB, QSUB]			
16 bits	SUB	32 bits	DSUB
Execution condition	Normally ON/OFF/rising or falling pulse edge	Suitable Models	XD, XL
Hardware requirement	-	Software requirement	-
64 bits	QSUB		
Execution condition	Normal ON/OFF/falling or rising pulse edge	Suitable Models	XDH, XLH
Hardware requirement	Version V3.7.1 or later	Software requirement	Version V3.7.4a or later

2)Operands

Operands	Function	Data Type
Three operands		
S1	The sub operation data address	16 bits /32 bits/64 bits, BIN; INT/DINT/LINT
S2	The sub operation data address	16 bits /32 bits/64 bits, BIN; INT/DINT/LINT
D	The result address	16 bits /32 bits/64 bits, BIN; INT/DINT/LINT
Two operands		
D	Be subtracted data and result address	16 bits /32 bits/64 bits,BIN; INT/DINT/LINT
S1	Subtract data address	16 bits /32 bits/64 bits,BIN; INT/DINT/LINT

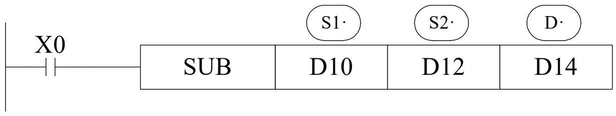
3)Suitable soft component

Operands	Word soft elements											Bit soft elements						
	System								Constant	Module		System						
	D	FD	TD	CD	DX	DY	DM	DS	K/H	ID	QD	X	Y	M	S	T	C	Dn.m
Three operands																		
S1	●	●	●	●	●	●	●	●	●									
S2	●	●	●	●	●	●	●	●	●									
D	●		●	●		●	●	●										
Two operands																		
D	●																	
S1	●	●							●									

*Notes: D includes D, HD; TD includes TD, HTD; CD includes CD, HCD, HSCD, HSD; DM includes DM, DHM; DS includes DS, DHS. M includes M,HM,SM； S includes S,HS； T includes T,HT； C includes C, HC.

Description

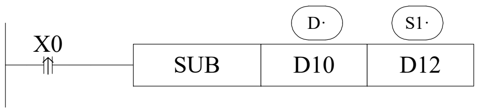
<Three operands>



(D10) — (D12) → (D14)

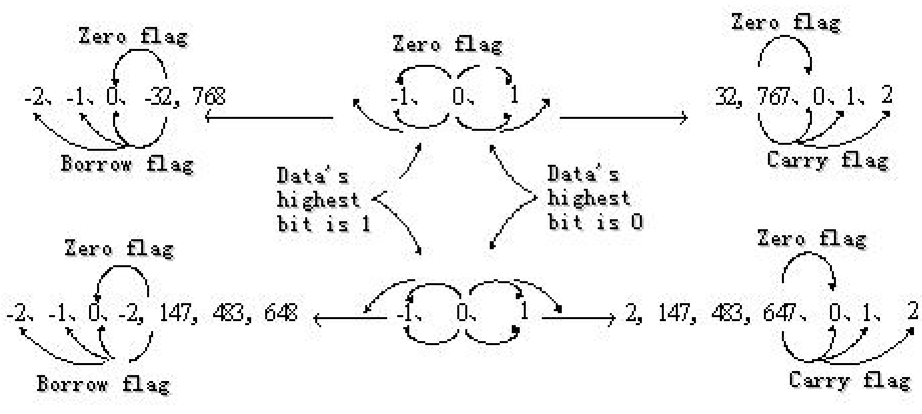
- S1 appoint the soft unit's content, subtract the soft unit's content appointed by S2 algebraically. The result will be stored in the soft unit appointed by D.
- The action of each flag, the setting method of 32/64 bits operation's soft units are both the same with the preceding ADD instruction.
- The importance is: in the preceding program, if X0 is ON, SUB operation will be executed every scan cycle.
- Refer to chapter 4-6-1 for flag action and functions.

<Two operands>



(D10) — (D12) → (D10)

- D appoint the soft unit's content, subtract the soft unit's content appointed by S1 algebraically. The result will be stored in the soft unit appointed by D.
- The action of each flag, the setting method of 32/64 bits operation's soft units are both the same with the preceding ADD instruction.
- The importance is: in the preceding program, if X0 is ON, SUB operation will be executed every scan cycle. Rising or falling pulse edge is recommended to activate the instruction.
- Refer to chapter 4-6-1 for flag action and functions.The relationship of the flag's action and vale's positive/negative is shown below:



Note: The addresses of the operands in the QSUB instruction must be even.

4-6-3 Multiplication [MUL, DMUL, QMUL]

1)Summary

Multiply two numbers, store the result

Multiplication [MUL, DMUL, QMUL]			
16 bits	MUL	32 bits	DMUL
Execution condition	Normally ON/OFF / pulse edge	Suitable Models	XD, XL
Hardware requirement	-	Software requirement	-
64 bits	QMUL		
Execution condition	Normal ON/OFF/falling or rising pulse edge	Suitable Models	XDH, XLH
Hardware requirement	Version V3.7.1 or later	Software requirement	Version V3.7.4a or later

2) Operands

Operands	Function	Data Type
S1	The multiplication operation data address	16 bits /32 bits/64 bits, BIN; INT/DINT/LINT
S2	The multiplication operation data address	16 bits /32 bits/64 bits, BIN; INT/DINT/LINT
D	The result address	32 bits /64 bits, BIN; INT/DINT/LINT

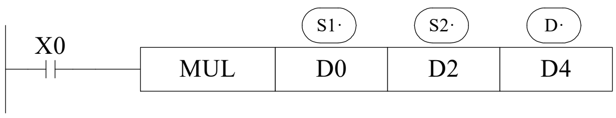
3) Suitable soft component

Operands	Word soft elements											Bit soft elements						
	System								Constant	Module		System						
	D	FD	TD	CD	DX	DY	DM	DS	K/H	ID	QD	X	Y	M	S	T	C	Dn.m
S1	●	●	●	●	●	●	●	●	●									
S2	●	●	●	●	●	●	●	●	●									
D	●		●	●		●	●	●										

*Notes: D includes D, HD; TD includes TD, HTD; CD includes CD, HCD, HSCD, HSD; DM includes DM, DHM; DS includes DS, DHS. M includes M,HM,SM； S includes S,HS； T includes T,HT； C includes C, HC.

Description

<16 bits Operation>



BIN BIN BIN

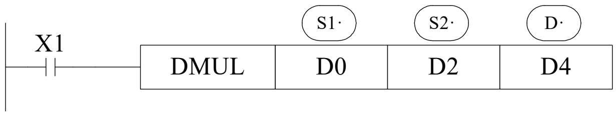
(D0) × (D2) → (D5, D4)

16 bits 16 bits → 32 bits

- The contents of the two source devices are multiplied together and the result is stored at the destination device in the format of32 bits. As the above chart: when (D0)=8, (D2)=9, (D5, D4) =72.

- The result's highest bit is the symbol bit: positive (0), negative (1).
- In the above example, when X0 is ON, the instruction will be executed in every scanning period.

<32 bits Operation >



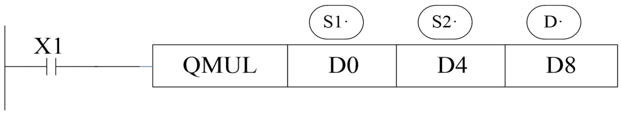
BIN BIN BIN

(D1, D0) × (D3, D2) → (D7, D6, D5, D4)

32 bits 32 bits → 64 bits

- When use32 bits operation, the result is stored at bits.
- Even use word device,64 bits results can't be monitored. Please change to floating value operation for this case.

<64 bits Operation >



BIN BIN BIN

(D3,D2,D1, D0) × (D7, D6,D5,D4) → (D11, D10, D9, D8)

64 bits 64 bits → 64 bits

- In 64-bit operation, a target address uses a bit soft element to get 64-bit results (occupying four consecutive registers, so don't reuse them). When using the word element, the result of 64-bit data operation cannot be directly monitored. Floating point arithmetic is recommended in this case.
- Note: The addresses of the operands in the QMUL instruction must be even.

4-6-4 Division [DIV, DDIV, QDIV]

1)Summary

Divide two numbers and store the result

Division [DIV, DDIV, QDIV]			
16 bits	DIV	32 bits	DDIV
Execution condition	Normally ON/OFF, rising/falling edge	Suitable Models	XD, XL
Hardware requirement	-	Software requirement	-
64 bits	QDIV		
Execution condition	Normal ON/OFF/falling or rising pulse edge	Suitable Models	XDH, XLH
Hardware requirement	Version V3.7.1 or later	Software requirement	Version V3.7.4a or later

2) Operands

Operands	Function	Data Type
S1	The divide operation data address	16 bits /32 bits/64 bits, BIN
S2	The divide operation data address	16 bits /32 bits/64 bits, BIN
D	The result address	16 bits /32 bits/64 bits, BIN

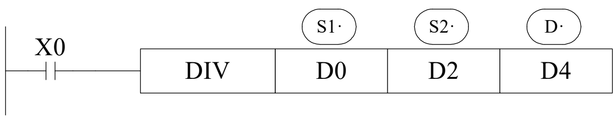
3)Suitable soft components

Operands	Word soft elements											Bit soft elements						
	System								Constant	Module		System						
	D	FD	TD	CD	DX	DY	DM	DS	K/H	ID	QD	X	Y	M	S	T	C	Dn.m
S1	●	●	●	●	●	●	●	●	●									
S2	●	●	●	●	●	●	●	●	●									
D	●		●	●		●	●	●										

*Notes: D includes D, HD; TD includes TD, HTD; CD includes CD, HCD, HSCD, HSD; DM includes DM, DHM; DS includes DS, DHS. M includes M,HM,SM； S includes S,HS； T includes T,HT； C includes C, HC.

Description

<16 bits operation>



Dividend Divisor Result Remainder

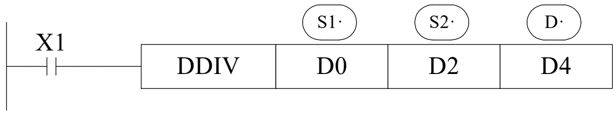
BIN BIN BIN BIN

(D0) ÷ (D2) → (D4) --- (D5)

16 bits 16 bits 16 bits 16 bits

- S1 appoints the dividend soft component,S2 appoints the divisor soft component, and D specifies the software component and the next number of the software component to be deposited and the remainder .
- In the above example, if input X0 is ON, devision operation is executed every scan cycle.

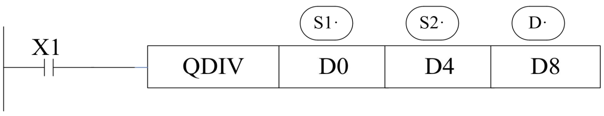
<32 bits operation>



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

- 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.
- If the value of the divisor is 0, the instruction will be error.
- 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.

<64 bits operation>



Dividend	Divisor	Result	Remainder
BIN	BIN	BIN	BIN
(D3,D2,D1,D0) ÷	(D7,D6,D5,D4) →	(D11,D10,D9,D8) ---	(D15,D14,D13,D12)
64 bit	64 bits	64 bits	64 bits

- 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.
- If the value of the divisor is 0, the instruction will be error.
- 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.
- Note: The addresses of the operands in the QDIV instruction must be even.

4-6-5 Increment [INC, DINC, QINC] & Decrement [DEC, DDEC, QDEC]

1) Summary

Increase or decrease the number

Increase one [INC,DINC,QINC]			
16 bits	INC	32 bits	DINC
Execution condition	Normally ON/OFF, rising/falling edge	Suitable Models	XD, XL
Hardware requirement	-	Software requirement	-
64 bits	QINC		
Execution condition	Normal ON/OFF/falling or rising pulse edge	Suitable Models	XDH, XLH
Hardware requirement	Version V3.7.1 or later	Software requirement	Version V3.7.4a or later
Decrease one [DEC,DDEC,QDEC]			
16 bits	DEC	32 bits	DDEC
Execution condition	Normally ON/OFF, rising/falling edge	Suitable Models	XD, XL
Hardware requirement	-	Software requirement	-
64 bits	QDEC		
Execution condition	Normal ON/OFF/falling or rising pulse edge	Suitable Models	XDH, XLH
Hardware requirement	Version V3.7.1 or later	Software requirement	Version V3.7.4a or later

2) Operands

Operands	Function	Data Type
D	The increase or decrease data address	16 bits / 32 bits/64 bits,BIN; INT/DINT/LINT

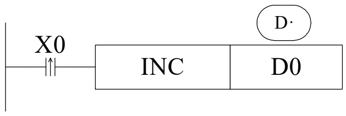
3) Suitable soft components

Operands	Word soft elements											Bit soft elements						
	System								Constant	Module		System						
	D	FD	TD	CD	DX	DY	DM	DS	K/H	ID	QD	X	Y	M	S	T	C	Dn.m
D	●		●	●		●	●	●										

*Notes: D includes D, HD; TD includes TD, HTD; CD includes CD, HCD, HSCD, HSD; DM includes DM, DHM; DS includes DS, DHS. M includes M,HM,SM； S includes S,HS； T includes T,HT； C includes C, HC.

Description

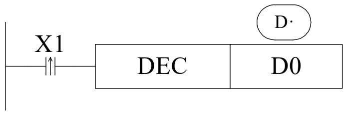
< Increment [INC]>



(D0) ＋1→(D0)

- D will increase one when X0 is ON.
- For 16 bits operation, when +32767 increase one, it will become -32768; The flag bit will act. for 32 bits operation, +2147483647 increases one is -2147483647. The flag bit will act.
  for 64 bits operation, +9223372036854775807 increases one is -9223372036854775808. The flag bit will act.

<Decrement [DEC]>



(D0) －1 → (D0)

- D will decrease one when X1 is ON.
- -32767 or -2147483647 decrease one, the result will be +32767 or +2147483647. The flag bit will act. For 64 bits operation, -9223372036854775808 decrease one is +9223372036854775807. The flag bit will act.
- The addresses of operands in QINC and QDEC instruction must be even.

Note: When the edge instruction is triggered, the automatic addition and subtraction operation is performed for each trigger. If it is triggered by normally open/normally closed, the operation of auto-addition and auto-subtraction will be performed in each scanning period after the conduction.

4-6-6 Mean [MEAN, DMEAN]

1)Summary

Get the mean value of data

Mean [MEAN,DMEAN]			
16 bits	MEAN	32 bits	DMEAN
Execution condition	Normally ON/OFF, rising/falling edge	Suitable Models	XD, XL
Hardware requirement	-	Software requirement	-

2)Operands

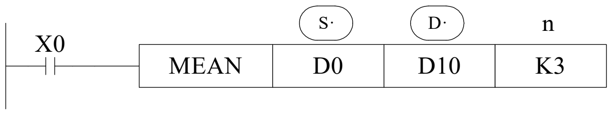
Operands	Function	Data Type
S	The source datastart address	16 bits/32 bits, BIN
D	The mean result address	16 bits/32 bits, BIN
n	The data quantity	16 bits/32 bits, BIN

3)Suitable soft components

Operands	Word soft elements											Bit soft elements						
	System								Constant	Module		System						
	D	FD	TD	CD	DX	DY	DM	DS	K/H	ID	QD	X	Y	M	S	T	C	Dn.m
S	●	●	●	●		●	●	●										
D	●	●	●	●		●	●	●										
n									●									

*Notes: D includes D, HD; TD includes TD, HTD; CD includes CD, HCD, HSCD, HSD; DM includes DM, DHM; DS includes DS, DHS. M includes M,HM,SM; S includes S,HS; T includes T,HT; C includes C, HC.

Description



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

- Store the mean value of source data (source sum divide by source quantity n). give the remainder .
- The n cannot larger than soft component quantity, otherwise there will be error.

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

1)Summary

Do logic AND, OR, XOR for data

Logic AND [WAND, DWAND]			
16 bits	WAND	32 bits	DWAND
Execution condition	Normally ON/OFF, rising/falling edge	Suitable Models	XD, XL
Hardware requirement	-	Software requirement	-
Logic OR[WOR,DWOR]			
16 bits	WOR	32 bits	DWOR
Execution condition	Normally ON/OFF, rising/falling edge	Suitable Models	XD, XL
Hardware requirement	-	Software requirement	-
Logic Exclusive OR [WXOR,DWXOR]			
16 bits	WXOR	32 bits	DWXOR
Execution condition	Normally ON/OFF, rising/falling edge	Suitable Models	XD, XL
Hardware requirement	-	Software requirement	-

2) Operands

Operands	Function	Data Type
S1	The operation data address	16bits/32bits,BIN; INT/DINT
S2	The operation data address	16bits/32bits,BIN; INT/DINT
D	The result address	16bits/32bits,BIN; INT/DINT

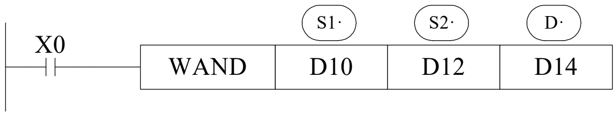
3)Suitable soft components

Operands	Word soft elements											Bit soft elements						
	System								Constant	Module		System						
	D	FD	TD	CD	DX	DY	DM	DS	K/H	ID	QD	X	Y	M	S	T	C	Dn.m
S1	●	●	●	●	●	●	●	●	●									
S2	●	●	●	●	●	●	●	●	●									
D	●		●	●		●	●	●										

*Notes: D includes D, HD; TD includes TD, HTD; CD includes CD, HCD, HSCD, HSD; DM includes DM, DHM; DS includes DS, DHS. M includes M,HM,SM； S includes S,HS； T includes T,HT； C includes C, HC.

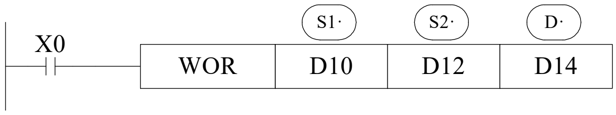
Description

<Logic AND >



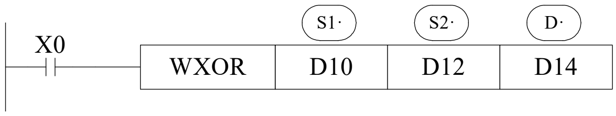
0&0=0 0&1=0
1&0=0 1&1=1

<Logic OR >



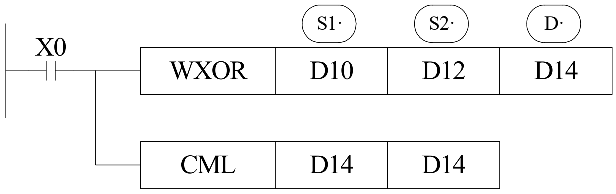
0 or 0=0 0 or 1=1
1 or 0=1 1 or 1=1

< Logic WXOR >



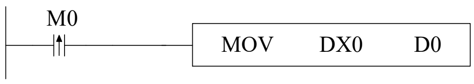
0 xor 0=0 0 xor 1=1
1 xor 0=1 1 xor 1=0

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



Example 1:

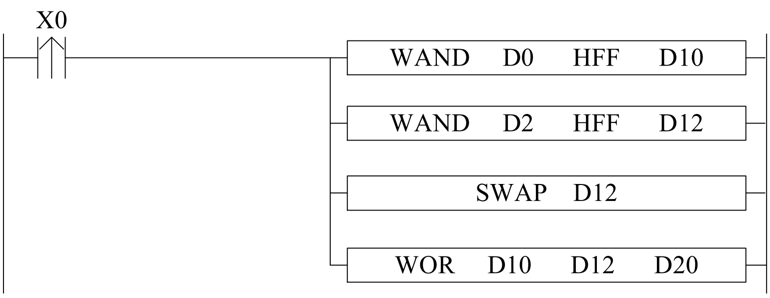
The16 bits data is composed by X0~X7, and store in D0.



Transform the state of X0, X1, X2, X3 to 8421 code and store in D0.

Example 2:

Combine the low 8 bits of D0 and D2 to a word.



```
LDP    X0                 //X0 rising edge
WAND   D0    HFF    D10   //Logic and, take the low 8 bits of D0 and save in D10
WAND   D2    HFF    D12   // Logic and, take the low 8 bits of D2 and save in D12
SWAP   D12                //swap the low 8 bits and high 8 bits of D12
WOR    D10   D12    D20   //combine the low 8 bits of D10 and high 8 bits of D12,
                          and save in D20
```

4-6-8 Logic converse [CML, DCML]

1) Summary

Logic converse the data

Converse [CML,DCML]			
16 bits	CML	32 bits	DCML
Execution condition	Normally ON/OFF, rising/falling edge	Suitable Models	XD, XL
Hardware requirement	-	Software requirement	-

2)Operands

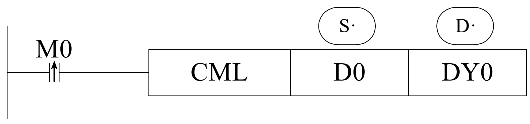
Operands	Function	Data Type
S	Source data address	16 bits/32 bits, BIN; INT/DINT
D	Result address	16 bits/32 bits, BIN; INT/DINT

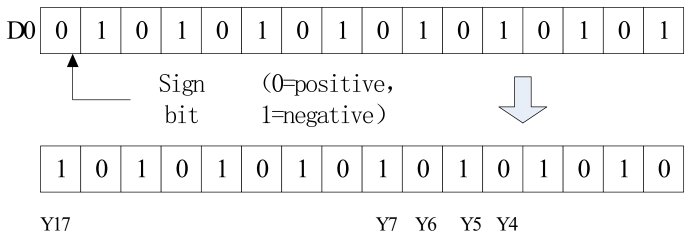
3)Suitable soft components

Operands	Word soft elements											Bit soft elements						
	System								Constant	Module		System						
	D	FD	TD	CD	DX	DY	DM	DS	K/H	ID	QD	X	Y	M	S	T	C	Dn.m
S	●	●	●	●	●	●	●	●	●									
D	●		●	●		●	●	●										

*Notes: D includes D, HD; TD includes TD, HTD; CD includes CD, HCD, HSCD, HSD; DM includes DM, DHM; DS includes DS, DHS. M includes M,HM,SM; S includes S,HS; T includes T,HT; C includes C, HC.

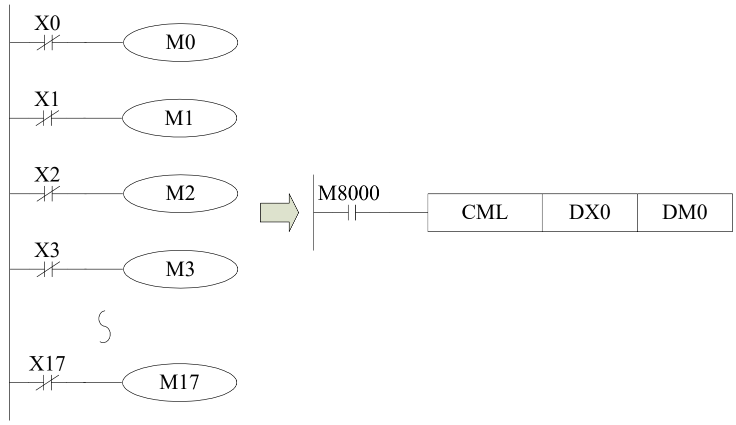
Description





- Each data bit in the source device is reversed (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.
- This instruction is fit for PLC logical converse output.

<Read the converse input>



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

4-6-9 Negative [NEG, DNEG]

1) Summary

Get the negative data

Negative [NEG,DNEG]			
16 bits	NEG	32 bits	DNEG
Execution condition	Normally ON/OFF, rising/falling edge	Suitable Models	XD, XL
Hardware requirement	-	Software requirement	-

2) Operands

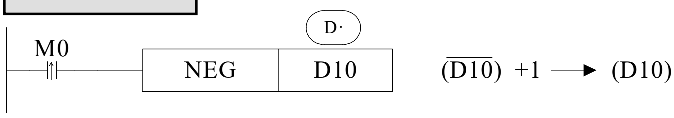
Operands	Function	Data Type
D	The source data address	16 bits/ 32 bits, BIN; INT/DINT

3) Suitable soft components

Operands	Word soft elements											Bit soft elements						
	System								Constant	Module		System						
	D	FD	TD	CD	DX	DY	DM	DS	K/H	ID	QD	X	Y	M	S	T	C	Dn.m
D	●		●	●		●	●	●										

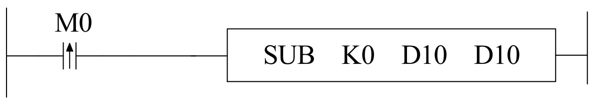
*Notes: D includes D, HD; TD includes TD, HTD; CD includes CD, HCD, HSCD, HSD; DM includes DM, DHM; DS includes DS, DHS. M includes M,HM,SM;S includes S,HS;T includes T,HT;C includes C, HC.

Description



- Converse each bit of source data (1→0, 0→1), then plus one and store the result in the source data address.
- For example, the source data D10 is 20, when M0 rising edge is coming, D10 become -20.

The following two instructions are the same.



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,DSHL], Arithmetic shift right [SHR,DSHR]

1) Summary

Do arithmetic shift left/right for the numbers

Arithmetic shift left [SHL,DSHL]			
16 bits	SHL	32 bits	DSHL
Execution condition	Normally ON/OFF, rising/falling edge	Suitable Models	XD, XL
Hardware requirement	-	Software requirement	-
Arithmetic shift right [SHR,DSHR]			
16 bits	SHR	32 bits	DSHR
Execution condition	Normally ON/OFF, rising/falling edge	Suitable Models	XD, XL
Hardware requirement	-	Software requirement	-

2) Operands

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

3) Suitable soft components

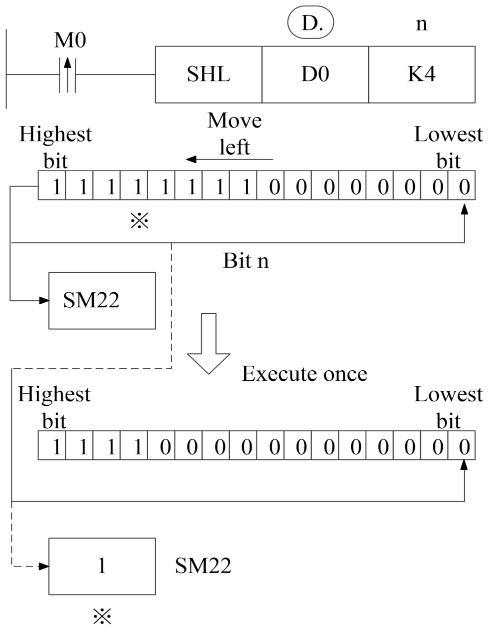
Operands	Word soft elements											Bit soft elements						
	System								Constant	Module		System						
	D	FD	TD	CD	DX	DY	DM	DS	K/H	ID	QD	X	Y	M	S	T	C	Dn.m
D	●	●	●	●		●	●	●										
n									●									

*Notes: D includes D, HD; TD includes TD, HTD; CD includes CD, HCD, HSCD, HSD; DM includes DM, DHM; DS includes DS, DHS. M includes M,HM,SM;S includes S,HS;T includes T,HT;C includes C, HC.

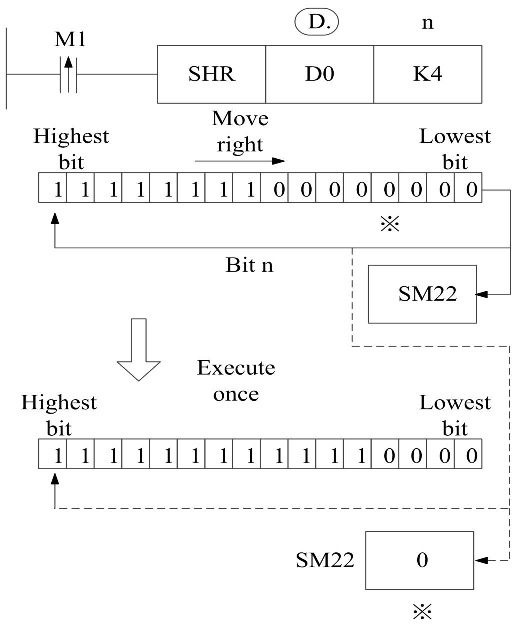
Description

After executing SHL once, the lowest bit is filled with 0, the last bit is stored in carry flag.
After executing SHR once, the highest bit is the same; the last bit is stored in carry flag.

< Arithmetic shift left >



<Arithmetic shift right>



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

1) Summary

Do logic shift right/left for the data.

Logic shift left [LSL, DLSL]			
16 bits	LSL	32 bits	DLSL
Execution condition	Normally ON/OFF, rising/falling edge	Suitable Models	XD, XL
Hardware requirement	-	Software requirement	-
Logic shift right [LSR,DLSR]			
16 bits	LSR	32 bits	DLSR
Execution condition	Normally ON/OFF, rising/falling edge	Suitable Models	XD, XL
Hardware requirement	-	Software requirement	-

2) Operands

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

3) Suitable soft components

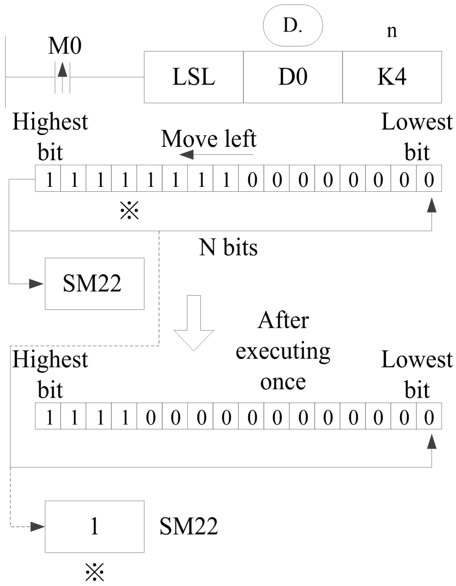
Operands	Word soft elements											Bit soft elements						
	System								Constant	Module		System						
	D	FD	TD	CD	DX	DY	DM	DS	K/H	ID	QD	X	Y	M	S	T	C	Dn.m
D	●	●	●	●		●	●	●										
n									●									

*Notes: D includes D, HD; TD includes TD, HTD; CD includes CD, HCD, HSCD, HSD; DM includes DM, DHM; DS includes DS, DHS. M includes M,HM,SM;S includes S,HS;T includes T,HT;C includes C, HC.

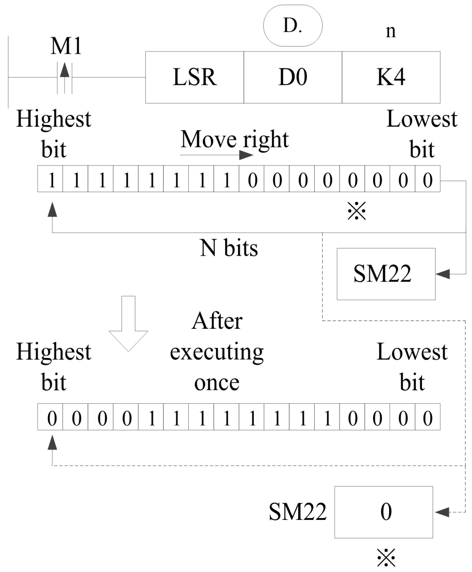
Description

- After executing LSL once, the lowest bit is filled with 0; the last bit is stored in carry flag.
- LSL meaning and operation are the same to SHL.
- After executing LSR once, the highest bit is filled with 0; the last bit is stored in carry flag.
- LSR and SHR are different, LSR add 0 inthe highest bit when moving, SHR all bits are moved.

< Logic shift left >



< Logic shift right >



4-7-3 Rotation shift left [ROL, DROL], Rotation shift right [ROR, DROR]

1)Summary

Cycle shift left or right

Rotation shift left [ROL, DROL]			
16 bits	ROL	32 bits	DROL
Execution condition	rising/falling edge	Suitable Models	XD, XL
Hardware requirement	-	Software requirement	-
Rotation shift right [ROR, DROR]			
16 bits	ROR	32 bits	DROR
Execution condition	rising/falling edge	Suitable Models	XD, XL
Hardware requirement	-	Software requirement	-

2) Operands

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

3)Suitable soft components

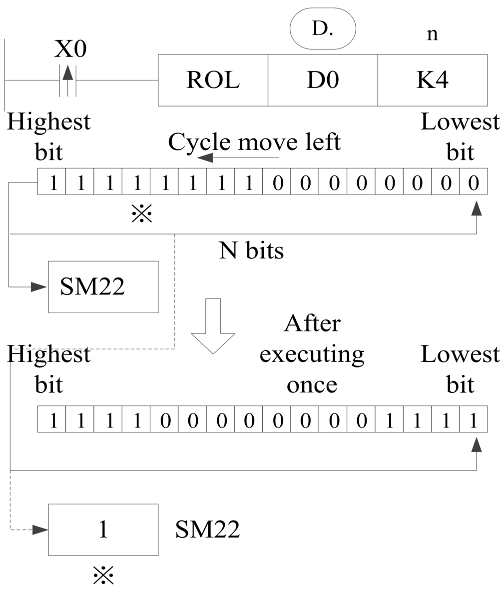
Operands	Word soft elements											Bit soft elements						
	System								Constant	Module		System						
	D	FD	TD	CD	DX	DY	DM	DS	K/H	ID	QD	X	Y	M	S	T	C	Dn.m
D	●	●	●	●		●	●	●										
n									●									

*Notes: D includes D, HD; TD includes TD, HTD; CD includes CD, HCD, HSCD, HSD; DM includes DM, DHM; DS includes DS, DHS. M includes M,HM,SM;S includes S,HS;T includes T,HT;C includes C, HC.

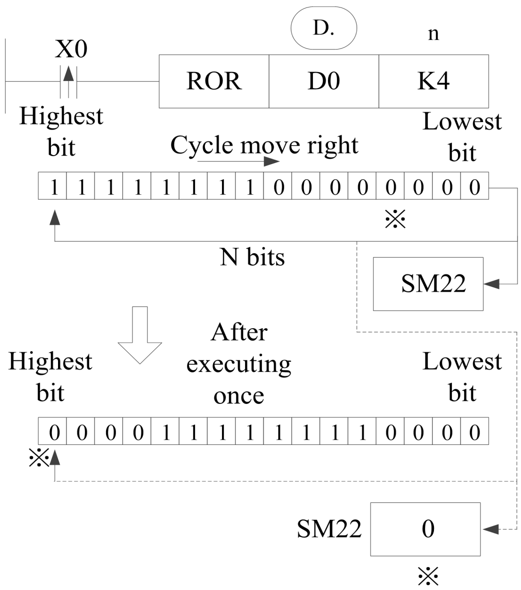
Description

- When X0 changes from OFF to ON, the value will be cycle moved left or right, the last bit is stored in carry flag.

< Cycle shift left>



< Cycle shift right>



4-7-4 Bit shift left [SFTL]

1) Summary

Bit shift left

Bit shift left [SFTL]			
16 bits	SFTL	32 bits	-
Execution condition	rising/falling edge	Suitable Models	XD, XL
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(no more than 1024)	16 bits, BIN
n2	Shift left times(no more than 1024)	16 bits, BIN

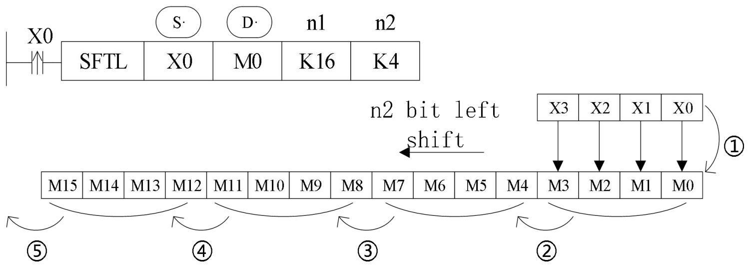
3) Suitable soft components

Operands	Word soft elements											Bit soft elements						
	System								Constant	Module		System						
	D	FD	TD	CD	DX	DY	DM	DS	K/H	ID	QD	X	Y	M	S	T	C	Dn.m
S												●	●	●	●	●	●	
D													●	●	●	●	●	
n1	●	●	●	●	●	●	●	●	●									
n2	●	●	●	●	●	●	●	●	●									

*Notes: D includes D, HD; TD includes TD, HTD; CD includes CD, HCD, HSCD, HSD;
DM includes DM, DHM; DS includes DS, DHS.M includes M, HM, SM; S includes S, HS;
T includes T, HT; C includes C, HC.

Description

- Move n2 bits left for the object which contains n1 bits.
- When X0 changes from OFF to ON, the instruction will move n2 bits for the object.
- For example, if n2 is K1, the object will move 1 bit left when the instruction executes once.



① X 3~X 0→M3~M0

② M 3~M 0→M7~M4

③ M 7~M 4→M11~M8

④ M11~M 8→M15~M 12

⑤ M15~M12→Overflow

4-7-5 Bit shift right [SFTR]

1) Summary

Bit shift right

Bit shift right [SFTR]			
16 bits	SFTR	32 bits	-
Execution condition	rising/falling edge	Suitable Models	XD, XL
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(no more than 1024)	16 bits, BIN
n2	Shift right times(no more than 1024)	16 bits, BIN

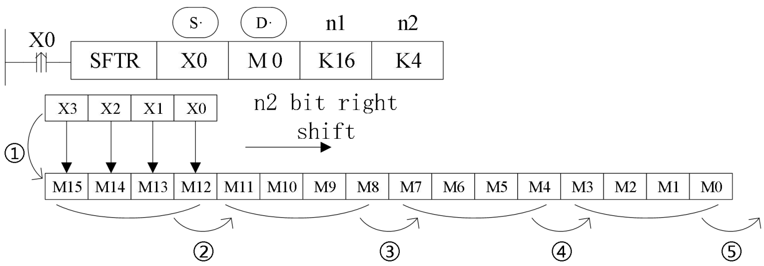
3) Suitable soft components

Operands	Word soft elements											Bit soft elements						
	System								Constant	Module		System						
	D	FD	TD	CD	DX	DY	DM	DS	K/H	ID	QD	X	Y	M	S	T	C	Dn.m
S												●	●	●	●	●	●	
D													●	●	●	●	●	
n1	●		●	●	●	●	●	●	●									
n2	●		●	●	●	●	●	●	●									

*Notes: D includes D, HD; TD includes TD, HTD; CD includes CD, HCD, HSCD, HSD; DM includes DM, DHM; DS includes DS, DHS.M includes M, HM, SM; S includes S, HS; T includes T, HT; C includes C, HC.

Description

- Move n2 bits right for the object which contains n1 bits.
- When X0 changes from OFF to ON, the instruction will move n2 bits for the object.
- For example, if n2 is 1, the object will move 1 bit right when the instruction executes once.



① X3~X 0→M15~M12

② M15~M12→M11~M8

③ M11~M 8→M7~M4

④ M7~M4→M3~M0

⑤ M3~M0→overflow

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	XD, XL
Hardware requirement	-	Software requirement	-

2) Operands

Operands	Function	Data Type
S	Source soft element head address	16 bits, BIN
D	Target soft element head address	16 bits, BIN
n1	Source data quantity(no more than 512)	16 bits, BIN
n2	Word shift left times (no more than 512)	16 bits, BIN

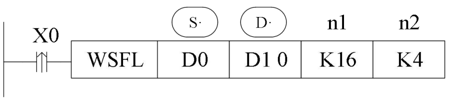
3) Suitable soft components

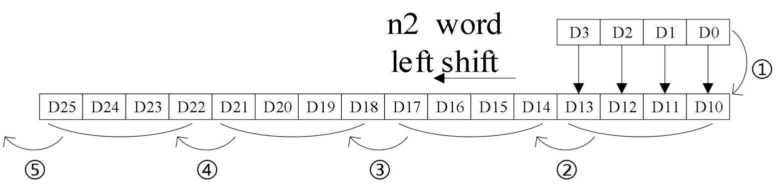
Operands	Word soft elements											Bit soft elements						
	System								Constant	Module		System						
	D	FD	TD	CD	DX	DY	DM	DS	K/H	ID	QD	X	Y	M	S	T	C	Dn.m
S	●	●	●	●														
D	●	●	●	●														
n1	●	●	●	●	●	●	●	●	●									
n2	●	●	●	●	●	●	●	●	●									

*Notes: D includes D, HD; TD includes TD, HTD; CD includes CD, HCD, HSCD, HSD; DM includes DM, DHM; DS includes DS, DHS. M includes M, HM, SM; S includes S, HS; T includes T, HT; C includes C, HC.

Description

- Move n2 words left for the object which contains n1 words.
- When X0 changes from OFF to ON, the instruction will move n2 words for the object.





① D 3~D 0→D13~D10
② D13~D10→D17~D14
③ D17~D14→D21~D18
④ D21~D18→D25~D22
⑤ D25~D22→overflow

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	XD, XL
Hardware requirement	-	Software requirement	-

2)Operands

Operands	Function	Data Type
S	Source soft element head address	16 bits, BIN
D	Target soft element head address	16 bits, BIN
n1	Source data quantity(no more than 512)	16 bits, BIN
n2	Shift right times(no more than 512)	16 bits, BIN

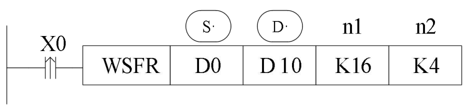
3)Suitable soft components

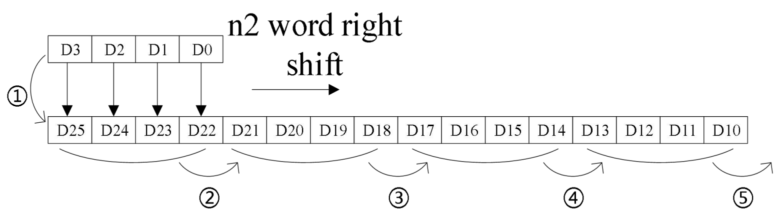
Operands	Word soft elements											Bit soft elements						
	System								Constant	Module		System						
	D	FD	TD	CD	DX	DY	DM	DS	K/H	ID	QD	X	Y	M	S	T	C	Dn.m
S	●	●	●	●														
D	●	●	●	●														
n1	●	●	●	●	●	●	●	●	●									
n2	●	●	●	●	●	●	●	●	●									

*Notes: D includes D, HD; TD includes TD, HTD; CD includes CD, HCD, HSCD, HSD; DM includes DM, DHM; DS includes DS, DHS. M includes M, HM, SM; S includes S, HS; T includes T, HT; C includes C, HC.

Description

- Move n2 words right for the object which contains n1 words.
- When X0 changes from OFF to ON, the instruction will move n2 words for the object.





① D 3~D 0→D25~D22

② D25~D22→D21~D18

③ D21~D18→D17~D14

④ D17~D14→D13~D10

⑤ D13~D10→overflow

4-8 Data Convert

Mnemonic	Function	Chapter
WTD	Single word integer converts to double word integer	4-8-1
DWTD	double word integer to four word integer	4-8-1
BDWTD	32 bits integer to64 bits integer batch conversion	4-8-2
FLT	16 bits integer converts to float point	4-8-3
DFLT	32 bits integer converts to float point	4-8-3
FLTD	64 bits integer converts to float point	4-8-3
DFLTD	32 bits integer to double precision floating point	4-8-4
QFLTD	64 bits integer to double precision floating point	4-8-4
INT	Float point converts to integer	4-8-5
DINTD	Double - precision floating point to32 bits integer	4-8-6
QINTD	Double - precision floating point to64 bits integer	4-8-6
ECON	Single precision floating point to double precision floating point	4-8-7
BECON	Single precision floating point to double precision floating point batch conversion	4-8-8
BIN	BCD convert to binary	4-8-9
BCD	Binary converts to BCD	4-8-10
ASCI	Hex. converts to ASCII	4-8-11
HEX	ASCII converts to Hex	4-8-12
DECO	Coding	4-8-13
ENCO	High bit coding	4-8-14
ENCOL	Low bit coding	4-8-15
GRY	Binary converts to gray code	4-8-16
GBIN	Gray code converts to binary	4-8-17

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

1) Summary

Single word integer converts to double word integer [WTD.DWTD]			
16 bits	WTD		
Execution condition	Normally ON/OFF, rising/falling edge	Suitable Models	XD, XL
Hardware requirement	-	Software requirement	-
32 bits	DWTD		
Execution condition	Normal ON/OFF/falling or rising pulse edge	Suitable Models	XDH、XLH
Hardware requirement	Version V3.7.1 or later	Software requirement	Version V3.7.4a or later

2) Operands

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

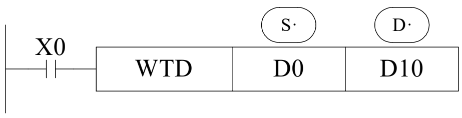
3) Suitable soft components

Operands	Word soft elements											Bit soft elements						
	System								Constant	Module		System						
	D	FD	TD	CD	DX	DY	DM	DS	K/H	ID	QD	X	Y	M	S	T	C	Dn.m
S	●	●	●	●	●	●	●	●										
D	●		●	●		●	●	●										

*Notes: D includes D, HD; TD includes TD, HTD; CD includes CD, HCD, HSCD, HSD; DM includes DM, DHM; DS includes DS, DHS. M includes M, HM, SM; S includes S, HS; T includes T, HT; C includes C, HC.

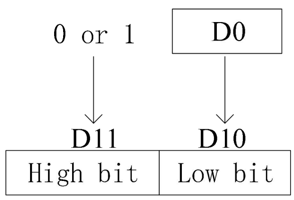
Description

<16 bits instruction>



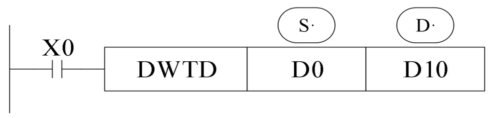
(D0) → (D11, D10)

Single Word Double Word

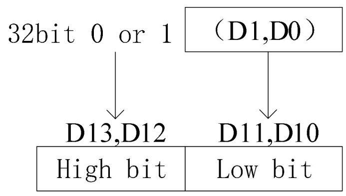


- When single word D0 is positive integer, after executing this instruction, the high bit of double word D10 is 0.
- When single word D0 is negative integer, after executing this instruction, the high bit of double word D10 is 1.
- the high bit 0 and 1 is binary value.

<32 bits instruction>



(D1,D0)→(D13,D12,D11,D10)
Double word four word



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

4-8-2 32 bits integer to 64 bits integer batch conversion [BDWTD]

1) Summary

32 bits integer to 64 bits integer batch conversion [BDWTD]			
32 bits	BDWTD		
Execution condition	Normal ON/OFF/falling or rising pulse edge	Suitable Models	XDH, XLH
Hardware requirement	Version V3.7.1 or later	Software requirement	Version V3.7.4a or later

2) Operands

Operands	Function	Data Type
S	Specify the source data or register's address code	32 bits ,BIN
D	Specify the target soft component's address code	64 bits ,BIN
N	Specify the value of the transfer point	16 bits ,BIN

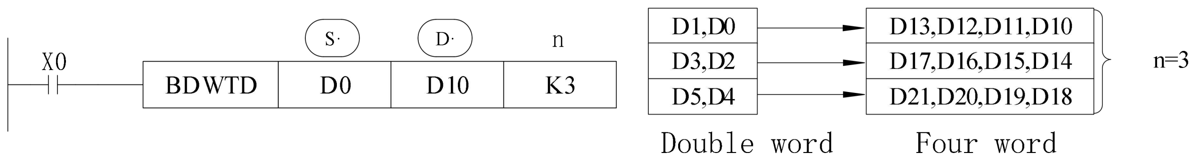
3) Suitable soft components

Operands	Word soft elements											Bit soft elements						
	System								Constant	Module		System						
	D	FD	TD	CD	DX	DY	DM	DS	K/H	ID	QD	X	Y	M	S	T	C	Dn.m
S	●	●																
D	●																	
n	●		●	●	●		●	●	●									

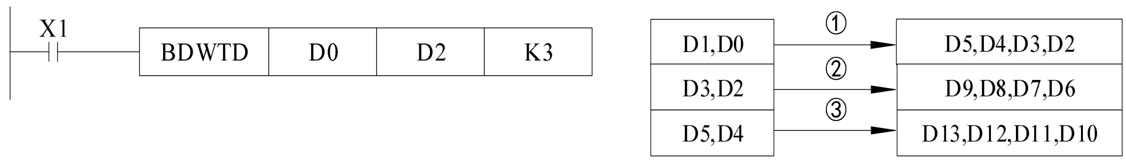
*Notes: D includes D, HD; TD includes TD, HTD; CD includes CD, HCD, HSCD, HSD; DM includes DM, DHM; DS includes DS, DHS. M includes M, HM, SM; S includes S, HS; T includes T, HT; C includes C, HC.

Description

- Converts n-point data starting with the source-specified soft element to n-point soft element starting with the target-specified soft element as a data block. (When the soft component number range is exceeded, convert to the extent possible).



- According to the method of overlapping numbering, the commands are automatically transmitted in the order of ① to ③ in order to prevent the transmission source data from being overwritten when the transmission number ranges overlap as shown in the figure below.



Note: The address of the four-word integer register in the BDWTD instruction must be even.

4-8-3 Integer converts to float point [FLT, DFLT,FLTD]

1)Summary

bit integer converts to float point [FLT, DFLT,FLTD]					
16 bits	FLT	32 bits	DFLT	64 bits	FLTD
Execution condition	Normally ON/OFF, rising/falling edge		Suitable Models	XD, XL	
Hardware requirement	-		Software requirement	-	

2) Operands

Operands	Function	Data Type
S	Source soft element address	16 bits/32 bits/64 bits, BIN; INT/DINT/LINT
D	Target soft element address	32 bits,BIN; REAL

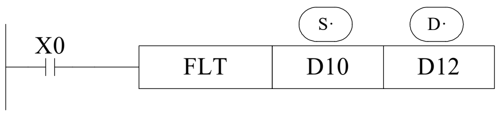
3) Suitable soft components

Operands	Word soft elements											Bit soft elements						
	System								Constant	Module		System						
	D	FD	TD	CD	DX	DY	DM	DS	K/H	ID	QD	X	Y	M	S	T	C	Dn.m
S	●	●																
D	●																	

*Notes: D includes D, HD; TD includes TD, HTD; CD includes CD, HCD, HSCD, HSD; DM includes DM, DHM; DS includes DS, DHS. M includes M, HM, SM; S includes S, HS; T includes T, HT; C includes C, HC.

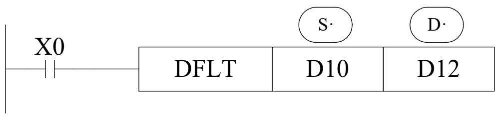
Description

<16 bits instruction>



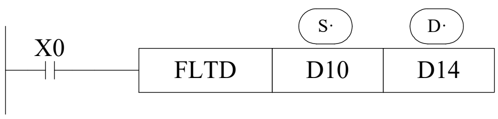
(D10) → (D13,D12)
BIN integer Binary float point

<32 bits instruction >



(D11,D10) → (D13,D12)
BIN integer Binary float point

<64 bits instruction>



(D13,D12,D11,D10) → (D15,D14)
BIN integer Binary float point

- Convert BIN integer to binary floating point. As the constant K, H will auto convert by the floating operation instruction, so this FLT instruction can't be used.

- The inverse transformation instruction is INT.
- FLTD can change the64 bits integer to32 bits floating value.
- The S operand of the FLTD instruction does not support constant K/H.

D0 is integer 20, after executing the instruction, D10 is floating value 20.

Note: Before using floating number operation instructions such as EADD, ESUB, EMUL, EDIV, EMOV and ECMP, make sure that all operation parameters are floating number.

4-8-4 Integer to double precision floating point[DFLTD,QFLTD]

1) Summary

integer to double precision floating point[DFLTD,QFLTD]			
32 bits	DFLTD	64 bits	QFLTD
Execution condition	Normal ON/OFF/falling or rising pulse edge	Suitable Models	XDH, XLH
Hardware requirement	Version V3.7.1 or later	Software requirement	Version V3.7.4a or later

2) Operands

Operands	Function	Data Type
S	Source soft element address	32 bits/64 bits, BIN; DINT/LINT
D	Target soft element address	64 bits, BIN; LREAL

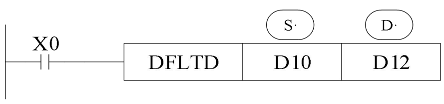
3) Suitable soft components

Operands	Word soft elements											Bit soft elements						
	System								Constant	Module		System						
	D	FD	TD	CD	DX	DY	DM	DS	K/H	ID	QD	X	Y	M	S	T	C	Dn.m
S	●	●																
D	●																	

*Notes: D includes D, HD; TD includes TD, HTD; CD includes CD, HCD, HSCD, HSD; DM includes DM, DHM; DS includes DS, DHS. M includes M, HM, SM; S includes S, HS; T includes T, HT; C includes C, HC.

Description

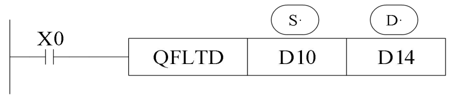
<32 bits instruction >



(D11,D10,)→(D15,D14,D13,D12)

BIN integer   Binary float point

<64 bits instruction>



(D13,D12,D11,D10)→(D17,D16,D15,D14)
BIN integer Binary float point

- An instruction to convert binary integer values to binary floating-point values. Constants K and H are automatically converted in each floating-point operation instruction, and FLT instruction can not be used.
- The inverse transformation of this instruction is DINTD/QINTD.
- QFLTD instruction converts 64-bit integer to 64-bit floating-point number. (Note: the address of the operand in the QFLTD instruction must be even.)
- The S operand of the QFLTD instruction does not support constant K/H.

4-8-5 Float point converts to integer [INT, DINT]

1)Summary

Floating point converts to integer [INT, DINT]			
16 bits	INT	32 bits	DINT
Execution condition	Normally ON/OFF, rising/falling edge	Suitable Models	XD, XL
Hardware requirement	-	Software requirement	-

2) Operands

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

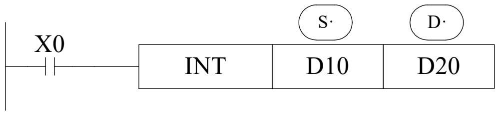
3) Suitable soft components

Operands	Word soft elements											Bit soft elements						
	System								Constant	Module		System						
	D	FD	TD	CD	DX	DY	DM	DS	K/H	ID	QD	X	Y	M	S	T	C	Dn.m
S	●	●																
D	●																	

*Notes: D includes D, HD; TD includes TD, HTD; CD includes CD, HCD, HSCD, HSD; DM includes DM, DHM; DS includes DS, DHS. M includes M, HM, SM; S includes S, HS; T includes T, HT; C includes C, HC.

Description

<16 bits instruction>

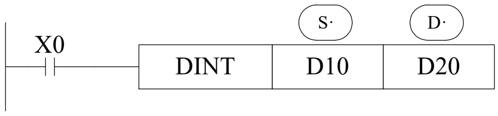


(D11,D10) → (D20)

Binary Float BIN integer

Give up the data after the decimal dot

<32 bits instruction>

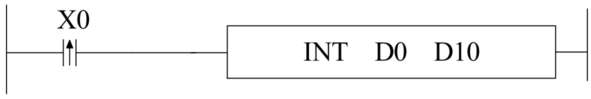


(D11,D10) → (D20,D21)

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.
- The inverse instruction is FLT.
- When the result is 0, the flag bit 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



For example, if D0 is floating value 130.2, after executing INT, D10 value is integer 130.

4-8-6 Double - precision floating point to integer[DINTD,QINTD]

1) Summary

floating point to integer [DINTD,QINTD]			
32 bits	DINTD	64 bits	QINTD
Execution condition	Normal ON/OFF/falling or rising pulse edge	Suitable Models	XDH, XLH
Hardware requirement	Version V3.7.1 or later	Software requirement	Version V3.7.4a or later

2) Operands

Operands	Function	Data Type
S	Source soft element address	64 bits, BIN; LREAL
D	Target soft element address	32 bits/64 bits,BIN; DINT/LINT

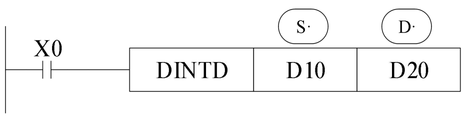
3) Suitable soft components

Operands	Word soft elements											Bit soft elements						
	System								Constant	Module		System						
	D	FD	TD	CD	DX	DY	DM	DS	K/H	ID	QD	X	Y	M	S	T	C	Dn.m
S	●	●																
D	●																	

*Notes: D includes D, HD; TD includes TD, HTD; CD includes CD, HCD, HSCD, HSD; DM includes DM, DHM; DS includes DS, DHS. M includes M, HM, SM; S includes S, HS; T includes T, HT; C includes C, HC.

Description

<32 bits instruction>

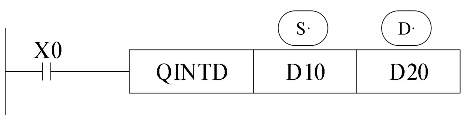


(D13,D12,D11,D10) → (D21,D20)

Binary Float BIN integer

Give up the data after the decimal dot

<64 bits instruction>



(D13,D12,D11,D10)→ (D23,D22,D21,D20)

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.
- The inverse instruction is DFLTD/QFLTD.
- For 64-bit instructions, the register address number must be even.
- When the result is 0, the flag bit is ON.

- The result is over below data, the carry flag is ON.
  64 bits operation: -9223372036854775808~9223372036854775807.

4-8-7 Single precision floating point to double precision floating point[ECON]

1) Summary

Single precision floating point to double precision floating point [ECON]			
32 bits	ECON		
Execution condition	Normal ON/OFF/falling or rising pulse edge	Suitable Models	XDH, XLH
Hardware requirement	Version V3.7.1 or later	Software requirement	Version V3.7.4a or later

2) Operands

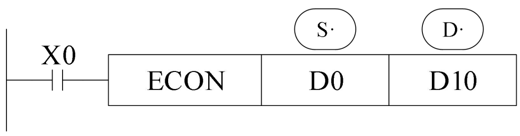
Operands	Function	Data Type
S	Source soft element address	32 bits, BIN; REAL
D	Target soft element address	64 bits, BIN; LREAL

3) Suitable soft components

Operands	Word soft elements											Bit soft elements						
	System								Constant	Module		System						
	D	FD	TD	CD	DX	DY	DM	DS	K/H	ID	QD	X	Y	M	S	T	C	Dn.m
S	●	●																
D	●																	

*Notes: D includes D, HD; TD includes TD, HTD; CD includes CD, HCD, HSCD, HSD; DM includes DM, DHM; DS includes DS, DHS. M includes M, HM, SM; S includes S, HS; T includes T, HT; C includes C, HC.

Description



(D1,D0) → (D13,D12,D11,D10)Single precision floating point double precision floating point

- When X0 turns on, the single-precision floating-point value in the source data address is converted into a double-precision floating-point value and stored in the target address.
- Register addresses for double - precision floating - point numbers must start with an even number.

4-8-8 Single precision floating point to double precision floating point batch conversion [BECON]

1) Summary

Single precision floating point to double precision floating point batch conversion[BECON]			
32 bits	BECON		
Execution condition	Normal ON/OFF/falling or rising pulse edge	Suitable Models	XDH, XLH
Hardware requirement	Version V3.7.1 or later	Software requirement	Version V3.7.4a or later

2) Operands

Operands	Function	Data Type
S	Specify the source data or register’s address code	32 bits, BIN
D	Specify the target soft component’s address code	64 bits, BIN
N	Specify the value of the transfer point	16 bits, BIN

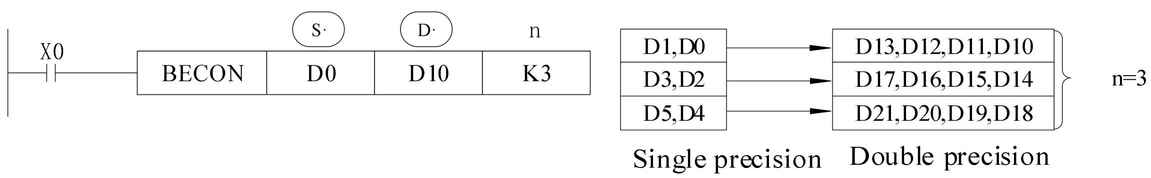
3) Suitable soft components

Operands	Word soft elements											Bit soft elements						
	System								Constant	Module		System						
	D	FD	TD	CD	DX	DY	DM	DS	K/H	ID	QD	X	Y	M	S	T	C	Dn.m
S	●	●																
D	●																	
n	●		●	●	●		●	●	●									

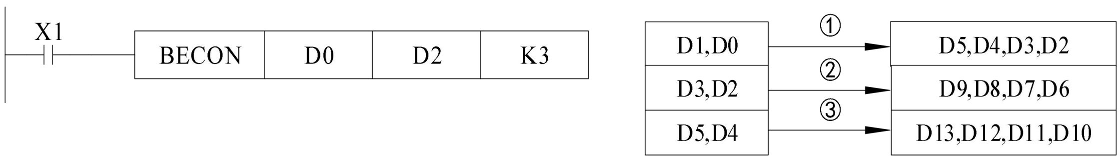
*Notes: D includes D, HD; TD includes TD, HTD; CD includes CD, HCD, HSCD, HSD; DM includes DM, DHM; DS includes DS, DHS. M includes M, HM, SM; S includes S, HS; T includes T, HT; C includes C, HC.

Description

- According to a scheme, n-point data starting with a source-specified software component is transferred as a data block to an N-point software component starting with a target-specified software component. (When the soft component number range is exceeded, convert to the extent possible).



- When the transmission number range overlapped, in order to prevent the transmission source data rewriting without conversion, the command will be automatically transmitted in the order of ①~③.



Note: The register header address of a double - precision floating-point must be even.

4-8-9 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	XD, XL
Hardware requirement	-	Software requirement	-

2) Operands

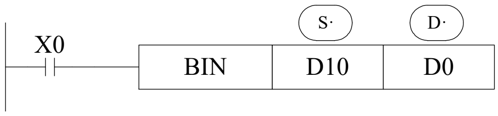
Operands	Function	Data Type
S	Source soft element address	BCD; INT
D	Target soft element address	16 bits, BIN; INT

3) Suitable soft components

Operands	Word soft elements											Bit soft elements						
	System								Constant	Module		System						
	D	FD	TD	CD	DX	DY	DM	DS	K/H	ID	QD	X	Y	M	S	T	C	Dn.m
S	●	●	●	●	●	●	●	●										
D	●		●	●		●	●	●										

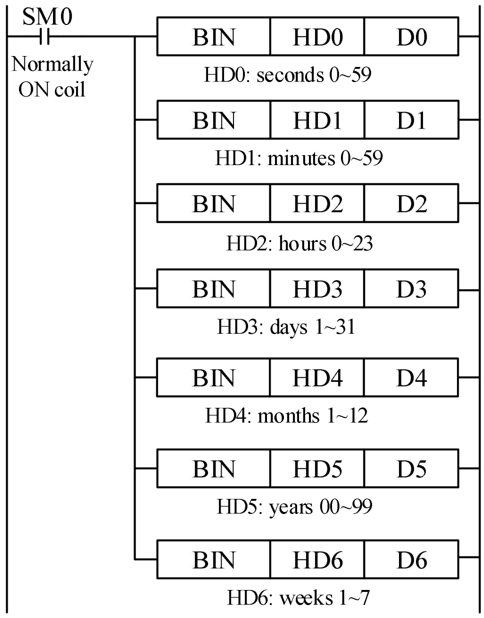
*Notes: D includes D, HD; TD includes TD, HTD; CD includes CD, HCD, HSCD, HSD; DM includes DM, DHM; DS includes DS, DHS. M includes M, HM, SM; S includes S, HS; T includes T, HT; C includes C, HC.

Description



Source (BCD) → destination (BIN)

- If source data is not BCD code, SM409 will be ON (Operation error), SD409=4 (error occurs).
- As constant K automatically converts to binary, so it’s not suitable for this instruction.
- For example, PLC reads external clock information and stores it in HD0~HD6 in the form of BCD code information. However, we are accustomed to using decimal values, so we can use BIN instructions to convert the required time information from BCD code information to binary:



4-8-10 Binary convert to BCD [BCD]

1) Summary

Convert binary data to BCD code

Binary convert to BCD [BCD]			
16 bits	BCD	32 bits	-
Execution condition	Normally ON/OFF, rising/falling edge	Suitable Models	XD, XL
Hardware requirement	-	Software requirement	-

2) Operands

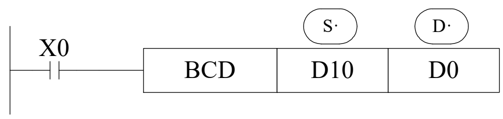
Operands	Function	Data Type
S	Source soft element address	16 bits, BIN; INT
D	Target soft element address	BCD code; INT

3) Suitable soft components

Operands	Word soft elements											Bit soft elements						
	System								Constant	Module		System						
	D	FD	TD	CD	DX	DY	DM	DS	K/H	ID	QD	X	Y	M	S	T	C	Dn.m
S	●	●	●	●	●	●	●	●										
D	●		●	●		●	●	●										

*Notes: D includes D, HD; TD includes TD, HTD; CD includes CD, HCD, HSCD, HSD; DM includes DM, DHM; DS includes DS, DHS. M includes M, HM, SM; S includes S, HS; T includes T, HT; C includes C, HC.

Description



source (BIN)→destination (BCD)

- This instruction can change the binary value to BCD code.
- BCD is a method of representing one bit decimal numbers by 4 bits binary numbers, ranging from 0 to 9.

4-8-11 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	XD, XL
Hardware requirement	-	Software requirement	-

2) Operands

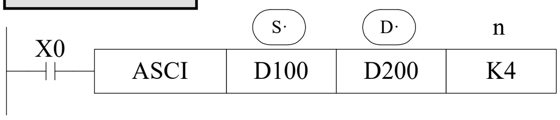
Operands	Function	Data Type
S	Source soft element address	2 bits, HEX
D	Target soft element address	ASCII code
n	Transform character quantity	16 bits, BIN

3)Suitable soft components

Operands	Word soft elements											Bit soft elements						
	System								Constant	Module		System						
	D	FD	TD	CD	DX	DY	DM	DS	K/H	ID	QD	X	Y	M	S	T	C	Dn.m
S	●	●	●	●	●	●	●	●										
D	●		●	●		●	●	●										
n	●		●	●		●	●	●	●									

*Notes: D includes D, HD; TD includes TD, HTD; CD includes CD, HCD, HSCD, HSD; DM includes DM, DHM; DS includes DS, DHS. M includes M, HM, SM; S includes S, HS; T includes T, HT; C includes C, HC.

Description



- Transform the source Hex data to ASCII code, and store in low 8-bit and high 8-bit of D· . The number of characters to be converted is specified by n.
- Store one ASCII data in the lower 8 bits and one ASCII data in the upper 8 bits.

D \ n	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]

The conversion result is shown in the figure:

Note: The function of the instruction is to convert hexadecimal values into ASCII characters, such as converting hexadecimal "A" to ASCII character "A".

4-8-12 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	XD, XL
Hardware requirement	-	Software requirement	-

2) Operands

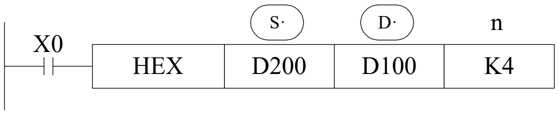
Operands	Function	Date type
S	Source soft element address	ASCII
D	Target soft element address	2 bits, HEX
n	ASCII Character quantity	16 bits, BIN

3) Suitable soft components

Operands	Word soft elements											Bit soft elements						
	System								Constant	Module		System						
	D	FD	TD	CD	DX	DY	DM	DS	K/H	ID	QD	X	Y	M	S	T	C	Dn.m
S	●	●	●	●	●	●	●	●										
D	●		●	●		●	●	●										
n									●									

*Notes: D includes D, HD; TD includes TD, HTD; CD includes CD, HCD, HSCD, HSD; DM includes DM, DHM; DS includes DS, DHS. M includes M, HM, SM; S includes S, HS; T includes T, HT; C includes C, HC.

Description



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

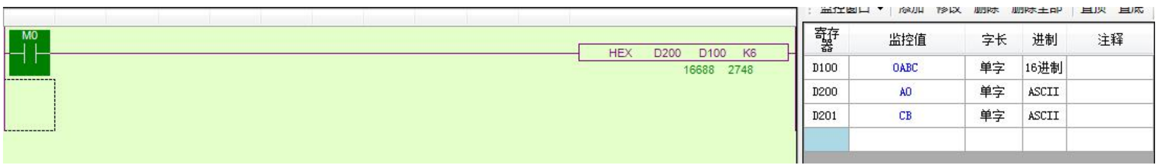
The convert process is the following:

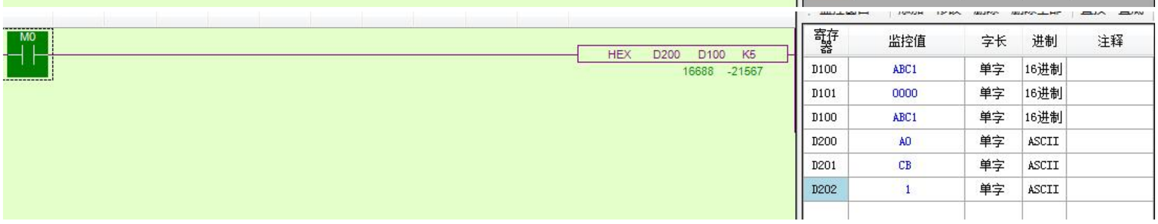
(S·)	ASCII code	HEX convert
D200 low	30H	0
D200 high	41H	A
D201 low	42H	B
D201 high	43H	C
D202 low	31H	1
D202 high	32H	2
D203 low	33H	3
D203 high	34H	4
D204 low	35H	5

n=k4

n \ (D·)	D102	D101	D100
1	Not change to be 0		···0H
2			··0AH
3			·0ABH
4			0ABCH
5		···0H	ABC1H
6		··0AH	BC12H
7		·0ABH	C123H
8		0ABCH	1234H
9	···0H	ABC1H	2345H

D200	0	1	0	0	0	0	0	1	0	0	1	1	0	0	0	0
	41H→[A]								30H→[0]							
D201	0	1	0	0	0	0	1	1	0	1	0	1	0	0	1	0
	43H→[C]								42H→[B]							
D100	0	0	0	0	1	0	1	0	1	0	1	1	1	1	0	0
	0				A				B				C			





4-8-13 Coding [DECO]

1)Summary

Change any data or bit to 1.

Coding [DECO]			
16 bits	DECO	32 bits	-
Execution condition	Normally ON/OFF, rising/falling edge	Suitable Models	XD, XL
Hardware requirement	-	Software requirement	-

2) Operands

Operands	Function	Data Type
S	The source data address	16 bits, BIN
D	The decode result head address	16 bits, BIN
n	The decoding soft element bit quantity	16 bits, BIN

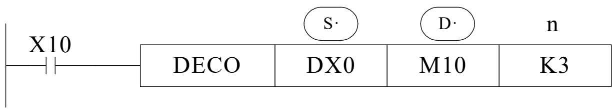
3) Suitable soft components

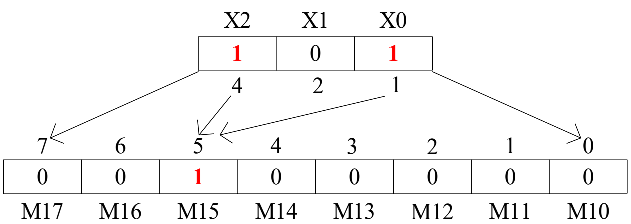
Operands	Word soft elements											Bit soft elements						
	System								Constant	Module		System						
	D	FD	TD	CD	DX	DY	DM	DS	K/H	ID	QD	X	Y	M	S	T	C	Dn.m
S	●	●	●	●	●	●	●	●										
D												●	●	●	●	●		
n									●									

*Notes: D includes D, HD; TD includes TD, HTD; CD includes CD, HCD, HSCD, HSD; DM includes DM, DHM; DS includes DS, DHS. M includes M, HM, SM; S includes S, HS; T includes T, HT; C includes C, HC.

Description

< When D· is bit unit > n≤16



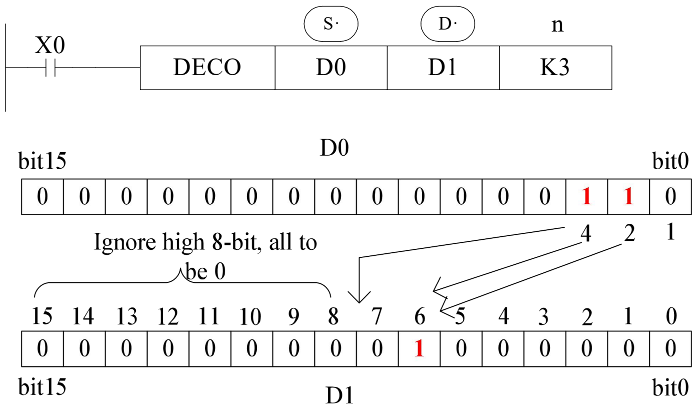


- n= 3, so the decoding object is the lower three bits in DX0, which are X2 ~ X0.
- n = 3, so the decoding results need to be expressed by 2^3 = 8 bits, which are M17 ~ M10.
- When X2 = 1, X1 = 0, X0 = 1, the value it represents is 4 + 1 = 5, so M15 in the fifth place from M10 changes to 1; when X2 ~ X0 is all zero, the value is 0, so M10 is 1 (M10 is the 0th

place).

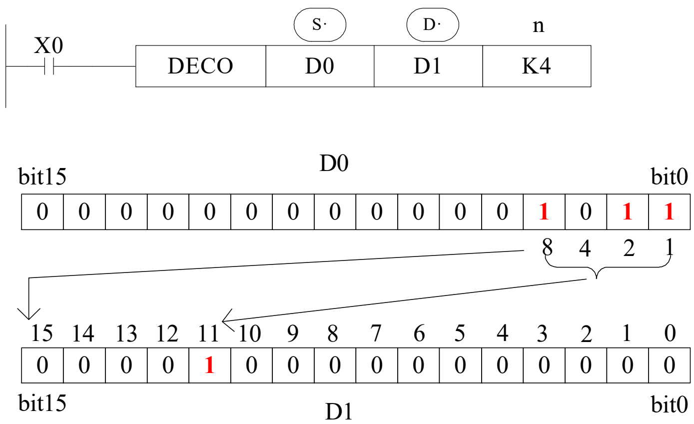
- If n = 0, the instruction will not be executed. If n is the value out of 0 ~ 16, the instruction will not be executed.
- When n = 16, if the decoding command D· is a bit soft component, the number of points is 2 ^ 16 = 65536.
- When the driver input is OFF, the instruction is not executed, and the decoding output of the action is maintained.

< When D· is word device > n≤4



- The low n-bit (n ≤ 4) of the source address is decoded to the target address. When n ≤ 3, the high 8-bit of the target turns to 0.
- If n = 0, the instruction will not be executed. If n is out of 0 ~ 4, the instruction will not be executed.
- N = 3, so the decoding object in D0 is bit2-bit0, and the maximum value it represents is 4 + 2 + 1 = 7.
- N = 3, so in D1, 2^3 = 8 bits are needed to represent the decoding result, that is, bit7 ~ bit0.
- When bit2 and bit1 are both 1 and bit0 are 0, the value is 4+2=6, so bit6 in D1 is ON.

< D· is word soft component > n≤4



- The low n-bit (n ≤4) of the source address is decoded to the target address. When n ≤ 3, the high 8-bit of the target turns to 0.
- If n = 0, the instruction will not be executed. If n is out of 0 ~ 4, the instruction will not be executed.

- N = 4, so the object of decoding in D0 is bit3 ~ bit0, which represents the maximum value of 8 + 4 + 2 + 1 = 15.
- N = 4, so in D1, 2^4 =16 bits are needed to represent the decoding result, that is, bit15 ~ bit0.
- When bit3, bit1 and bit0 are all 1 and bit2 is 0, the numerical value is 8+2+1=11, so bit11 in D1 is ON.

4-8-14 High bit coding [ENCO]

1) Summary

Find the highest bit which is 1.

High bit coding [ENCO]			
16 bits	ENCO	32 bits	-
Execution condition	Normally ON/OFF, rising/falling edge	Suitable Models	XD, XL
Hardware requirement	-	Software requirement	-

2) Operands

Operands	Function	Data Type
S	Coding data address	16 bits, BIN
D	Coding result address	16 bits, BIN
n	The bit quantity of coding result	16 bits, BIN

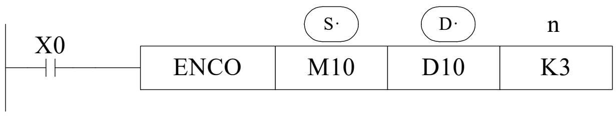
3) Suitable soft components

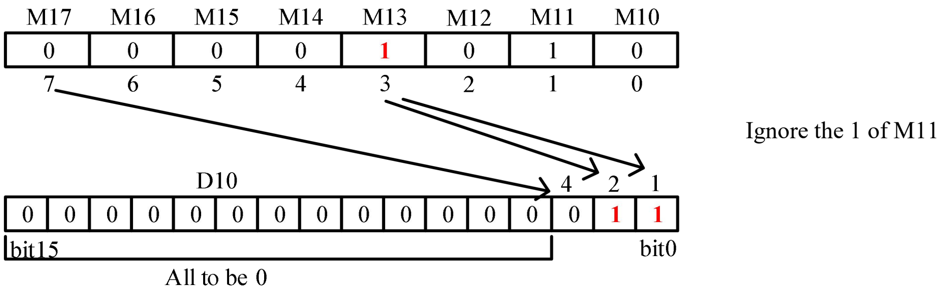
Operands	Word soft elements											Bit soft elements						
	System								Constant	Module		System						
	D	FD	TD	CD	DX	DY	DM	DS	K/H	ID	QD	X	Y	M	S	T	C	Dn.m
S	●	●	●	●	●	●	●	●				●	●	●	●	●	●	
D	●		●	●		●	●	●										
n									●									

*Notes: D includes D, HD; TD includes TD, HTD; CD includes CD, HCD, HSCD, HSD; DM includes DM, DHM; DS includes DS, DHS. M includes M, HM, SM; S includes S, HS; T includes T, HT; C includes C, HC.

Description

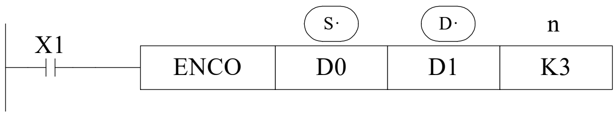
< When S· is bit device > n≤16

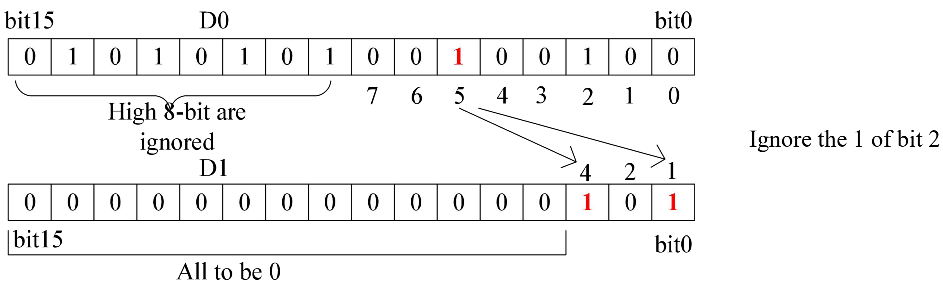




- If the number of bits in the source address is 1, the low side is ignored, and if the source address is 0, the instruction will not be executed.
- When the driving condition is OFF, the instruction is not executed and the coding output is unchanged.
- When n = 16, if the encoding instruction is a bit element, its point number is 2 ^ 16 = 65536.
- N = 3, the encoded object has $2^3 = 8$ bits, which are M17 ~ M10, and the encoding results are stored in the lower three bits of D10, which are bit2 ~ bit0.
- M13 and M11 are both 1. Ignoring M11, M13 is coded, bit2-bit0 represent 3, while bit0 and bit1 are 1.

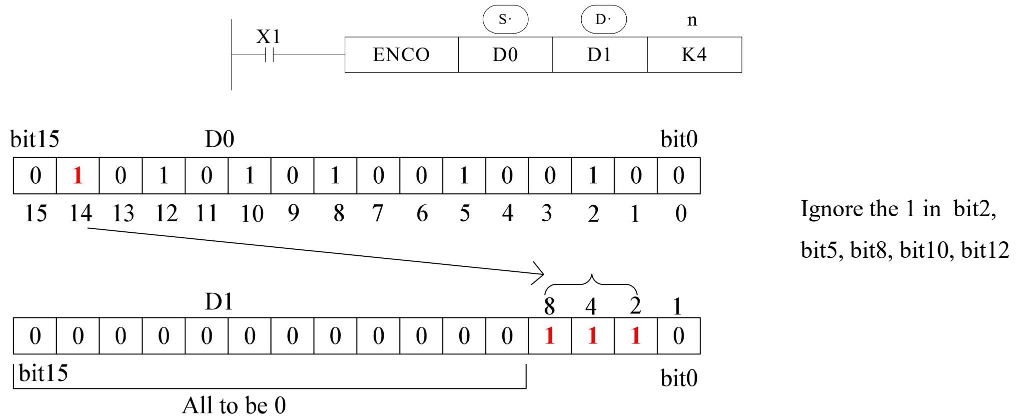
< When S· is word device n≤4





- If multiple bits in the source address is 1, the low side is ignored, and if the source address is 0, the instruction will not be executed.
- When the driver input is OFF, the instruction is not executed and the coding output is unchanged.
- When n ≤ 3, the high 8 bits in D0 are neglected.
- When n=3, the encoding object has $2^3 = 8$ bits, that is, bit7 ~ bit0 in D0. The encoding result is stored in the lower 3 bits in D1, that is, bit2 ~ bit0.
- When bit5 and bit2 in D0 are both 1, bit2 is ignored, and bit5 is coded, bit2-bit0 represent 5, bit2 and bit0 are 1.

< S· is word soft component > n≤4



- If the number of bits in the source address is 1, the low side is ignored, and if the source address is 0, the instruction will not be executed.
- When the driver input is OFF, the instruction is not executed and the coding output is unchanged.
- n = 4, the encoded object has 2^4 =16 bits, that is, bit15 ~ bit0 in D0. The encoding result is stored in the lower 4 bits in D1, that is, bit3 ~ bit0.
- The highest bit of 1 in D0 is bit14, ignoring all low bits 1, and encoding bit14, bit3-bit0 represent 14, bit3, bit2 and bit1 are 1.

4-8-15 Low bit coding [ENCOL]

1) Summary

Find the position where the low bit is ON.

Low bit coding [ENCOL]			
16 bits	ENCOL	32 bits	-
Execution condition	Normally ON/OFF, rising/falling edge	Suitable Models	XD, XL
Hardware requirement	-	Software requirement	-

2) Operands

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

3) Suitable soft components

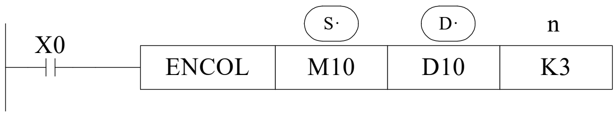
Operands	Word soft elements											Bit soft elements						
	System								Constant	Module		System						
	D	FD	TD	CD	DX	DY	DM	DS	K/H	ID	QD	X	Y	M	S	T	C	Dn.m
S	●	●	●	●	●	●	●	●				●	●	●	●	●	●	

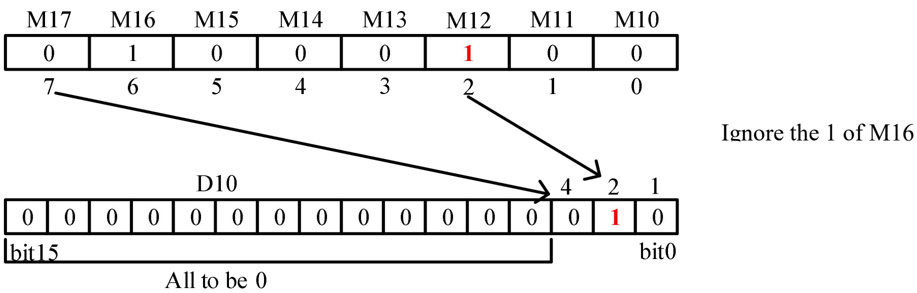
D	●		●	●		●	●	●										
n									●									

*Notes: D includes D, HD; TD includes TD, HTD; CD includes CD, HCD, HSCD, HSD; DM includes DM, DHM; DS includes DS, DHS. M includes M, HM, SM; S includes S, HS; T includes T, HT; C includes C, HC.

Description

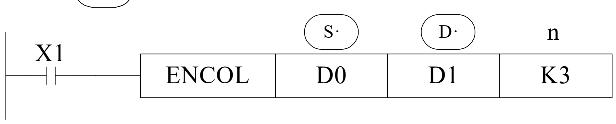
<if S· is bit device > n≤16

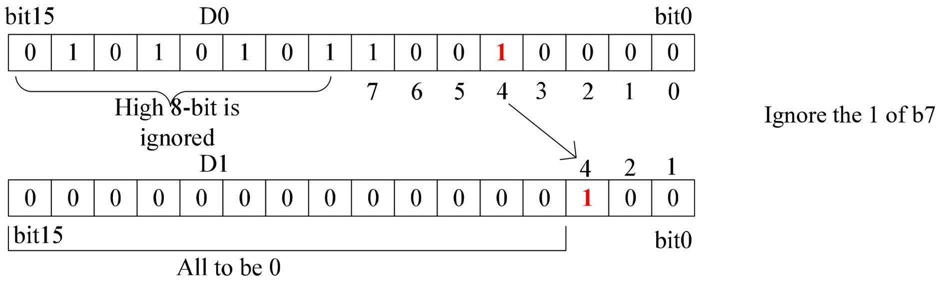




- If the number of bits in the source address is 1, the high bit side is ignored, and if the source address is 0, the instruction will not be executed.
- When the driving condition is OFF, the instruction is not executed and the coding output is unchanged.
- When n = 16, if the S· of encoding instruction is a bit element, its point is 2 ^ 16 = 65536.
- n = 3, the encoded object has 2^3 = 8 bits, which are M17 ~ M10, and the encoding results are stored in the lower three bits of D10, which are bit2 ~ bit0.
- M12 and M16 are both 1. Ignoring M16, M12 is coded, bit2-bit0 represent 2, while bit1 is 1.

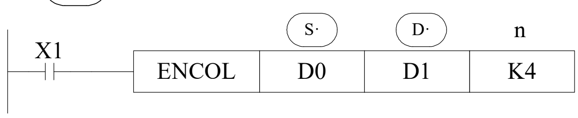
< if S· is word device> n≤4

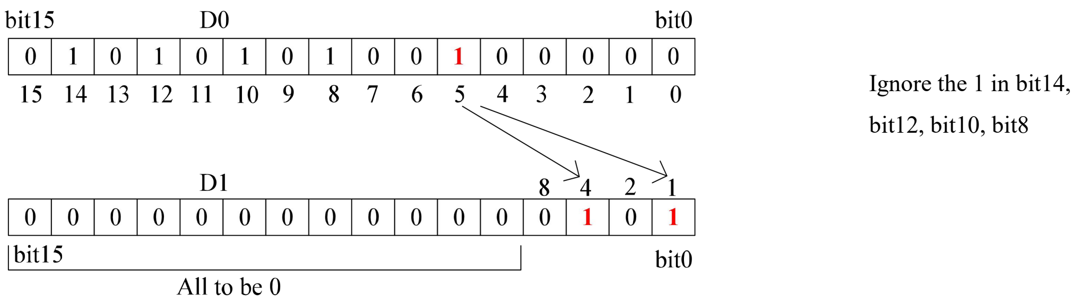




- If multiple bits in the source address is 1, the high bit side is ignored, and if the source address is 0, the instruction will not be executed.
- When the driver input is OFF, the instruction is not executed and the coding output is unchanged.
- When n ≤3, the high 8 bits in D0 are neglected.
- The encoding object has $2^3 = 8$ bits, that is, bit7 ~ bit0 in D0. The encoding result is stored in the lower 3 bits in D1, that is, bit2 ~ bit0.
- When bit7 and bit4 in D0 are both 1, bit7 is ignored and bit4 is coded. Bit 2 is 1 when bit2-bit0 is expressed as 4.

< S· is word soft component > n≤4





- If multiple bits in the source address is 1, the high bit side is ignored, and if the source address is 0, the instruction will not be executed.
- When the driver input is OFF, the instruction is not executed and the coding output is unchanged.
- n = 4, the encoded object has 2^4 =16 bits, that is, bit15 ~ bit0 in D0. The encoding result is stored in the lower 4 bits in D1, that is, bit3 ~ bit0.
- The lowest bit of 1 in D0 is bit5, ignoring all high bits 1, and encoding bit5 with bit3-bit0 as 5, bit2 and bit0 as 1.

4-8-16 Binary to Gray code [GRY, DGRY]

1) Summary

Transform the binary data to gray code.

Binary to gray [GRY,DGRY]			
16 bits	GRY	32 bits	DGRY
Execution condition	Normally ON/OFF, rising/falling edge	Suitable Models	XD, XL
Hardware requirement	-	Software requirement	-

2) Operands

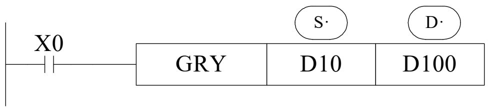
Operands	Function	Data Type
S	Soft element address need coding	16bits/32bits,BIN; INT
D	Soft element address to save coding result	16bits/32bits,BIN; INT

3) Suitable soft components

Operands	Word soft elements											Bit soft elements						
	System								Constant	Module		System						
	D	FD	TD	CD	DX	DY	DM	DS	K/H	ID	QD	X	Y	M	S	T	C	Dn.m
S	●	●	●	●	●	●	●	●	●									
D	●		●	●		●	●	●										

*Notes: D includes D, HD; TD includes TD, HTD; CD includes CD, HCD, HSCD, HSD; DM includes DM, DHM; DS includes DS, DHS. M includes M, HM, SM; S includes S, HS; T includes T, HT; C includes C, HC.

Description



Source (BIN) → target (GRY)

D10 (b15 … b0)

0	0	0	0	0	1	0	0	1	0	1	0	0	0	1	1

⇓

0	0	0	0	0	1	1	0	1	1	1	1	0	0	1	0

D100 (b15 … b0)

Each bit of D10 will XOR with the bit on its left side. As the related gray code, the left bit will not change (the left bit is 0); the transformation result is stored in D100.

- Transform the binary value to gray code.
- GRY has32 bits mode DGRY, which can transform32 bits gray code.
- S· Range is 0~32,767 (16 bits instruction); 0~2,147,483,647 (32 bits instruction).

4-8-17 Gray code to binary [GBIN,DGBIN]

1) Summary

Transform the gray code to binary data.

Gray code to binary [GBIN,DGBIN]			
16 bits	GBIN	32 bits	DGBIN
Execution condition	Normally ON/OFF, rising/falling edge	Suitable Models	XD, XL
Hardware requirement	-	Software requirement	-

2) Operands

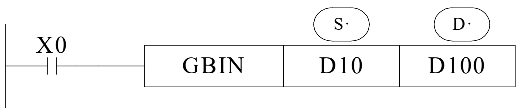
Operands	Function	Data Type
S	Soft element address need coding	16bits/32bits, BIN; INT
D	Soft element address to save coding result	16bits/32bits, BIN; INT

3) Suitable soft components

Operands	Word soft elements											Bit soft elements						
	System								Constant	Module		System						
	D	FD	TD	CD	DX	DY	DM	DS	K/H	ID	QD	X	Y	M	S	T	C	Dn.m
S	●	●	●	●	●	●	●	●	●									
D	●		●	●		●	●	●										

*Notes: D includes D, HD; TD includes TD, HTD; CD includes CD, HCD, HSCD, HSD; DM includes DM, DHM; DS includes DS, DHS. M includes M, HM, SM; S includes S, HS; T includes T, HT; C includes C, HC.

Description



Source (GRY) → target (BIN)

b15 D10 b0

0	0	0	0	0	1	1	0	1	1	1	1	0	0	1	0

⇓

0	0	0	0	0	1	0	0	1	0	1	0	0	0	1	1

b15 D100 b0

From the left second bit of D10, XOR each bit with the value after decoding, as the bit value after decoding (the left bit will not change). The transformation value will be stored in D100.

- Transform the gray code to binary value.
- GBIN has32 bits mode DBIN, which can transform32 bits binary value.
- (S·) Range is 0~32,767 (16 bits instruction); 0~2,147,483,647 (32 bits instruction).

4-9. Floating number Operation

Mnemonic	Function	Chapter
ECMP	Floating Compare	4-9-1
EZCP	Floating Zone Compare	4-9-2
EADD	Floating Add	4-9-3
ESUB	Floating Subtract	4-9-4
EMUL	Floating Multiplication	4-9-5
EDIV	Floating Division	4-9-6
ESQR	Floating 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 Floating Compare [ECMP,EDCMP]

1) Summary

Floating Compare [ECMP,EDCMP]			
16 bits	-	32 bits	ECMP
Execution condition	Normally ON/OFF, rising/falling edge	Suitable Models	XD, XL
Hardware requirement	-	Software requirement	-
64 bits	EDCMP		
Execution condition	Normal ON/OFF/falling or rising pulse edge	Suitable Models	XDH, XLH
Hardware requirement	Version V3.7.1 or later	Software requirement	Version V3.7.4a or later

2) Operands

Operands	Function	Data Type
S1	Soft element address need compare	32/64 bits, BIN
S2	Soft element address need compare	32/64 bits, BIN
D	Compare result	bit

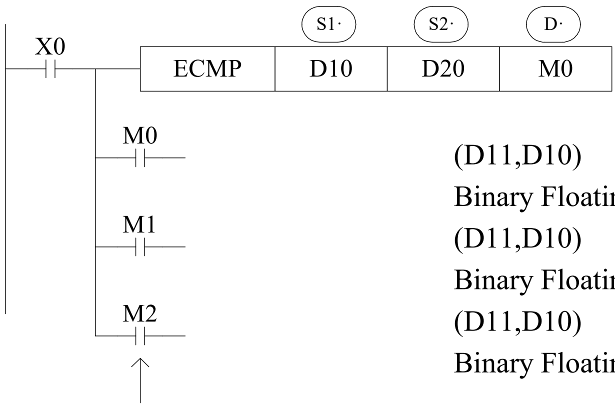
3) Suitable soft components

Operands	Word soft elements											Bit soft elements						
	System								Constant	Module		System						
	D	FD	TD	CD	DX	DY	DM	DS	K/H	ID	QD	X	Y	M	S	T	C	Dn.m
S1	●	●			●	●	●	●	●									
S2	●	●			●	●	●	●	●									
D												●	●	●				

*Notes: D includes D, HD; TD includes TD, HTD; CD includes CD, HCD, HSCD, HSD; DM includes DM, DHM; DS includes DS, DHS. M includes M, HM, SM; S includes S, HS; T includes T, HT; C includes C, HC.

Description

<32 bits operation>

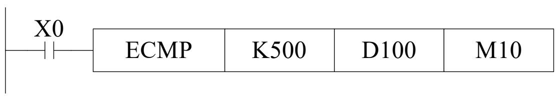


(D11,D10):(D21,D20) → M0,M1,M2
Binary FloatingBinary Floating

(D11,D10) > (D21<D20)
Binary Floating Binary Floating
(D11,D10) = (D21<D20)
Binary Floating Binary Floating
(D11,D10) < (D21<D20)
Binary Floating Binary Floating

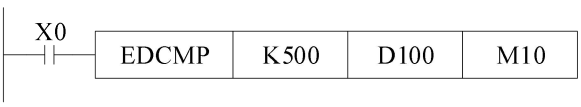
When X0 is OFF, even ECMP doesn't run, M0~M2 will keep the status before X0 is OFF.

- The instruction will compare the two source data S1 and S2. The result is stored in three bits from D.
- Before the instruction is executed, the comparison data must be all floating numbers (if it is an integer, it can be converted by FLT/DFLTD instructions); otherwise, the execution result will be wrong.
- The first address of the register in the EDCMP instruction must be an even number.
- If a constant K or H used as source data, the value is converted to floating value.



(K500) :(D101, D100) →M10,M11,M12
Binary converts Binary floating
to floating

<64 bits operation>



(K500)：(D103,D102,D101,D100)→ M10,M11,M12
Binary converts Binary floating
to floating

4-9-2 Floating Zone Compare [EZCP]

1) Summary

Floating Zone Compare [EZCP]			
16 bits	-	32 bits	EZCP
Execution condition	Normally ON/OFF, rising/falling edge	Suitable Models	XD, XL
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

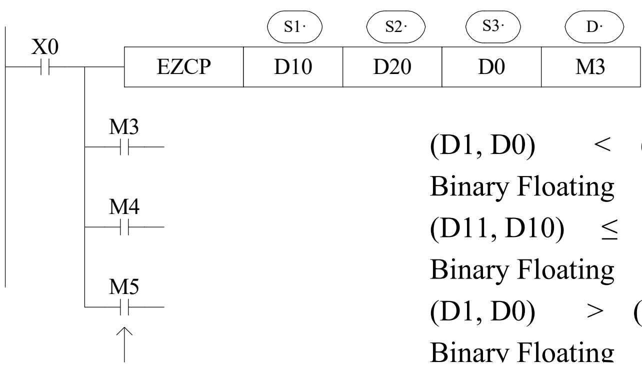
3) Suitable soft components

Operands	Word soft elements											Bit soft elements						
	System								Constant	System		Constant						
	D	FD	TD	CD	DX	DY	DM	DS	K/H	ID	QD	X	Y	M	S	T	C	Dn.m
S1	●	●			●	●	●	●	●									
S2	●	●			●	●	●	●	●									
S3	●	●			●	●	●	●	●									
D													●	●	●			

*Notes: D includes D, HD; TD includes TD, HTD; CD includes CD, HCD, HSCD, HSD; DM includes DM, DHM; DS includes DS, DHS. M includes M, HM, SM; S includes S, HS; T includes T, HT; C includes C, HC.

Description

Compare the source data with the range



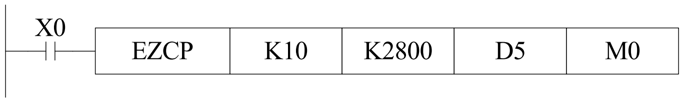
(D1, D0) < (D11, D10) ON
Binary Floating Binary Floating

(D11, D10) ≤ (D1, D0) ≤ (D21, D20) ON
Binary Floating Binary Floating Binary Floating

(D1, D0) > (D21, D20) ON
Binary Floating Binary Floating

When X0 is OFF, even EZCP doesn't run, M3~M5 will keep the status before X0 is OFF.

- Compare the source data S3 to the upper and lower limit value of the range S1~S2.
- The result will store in three coils starting from D.
- Before executing the instruction, all comparison data must be floating-point numbers (if it is an integer, it can be converted through FLT instruction); Otherwise, the execution result will be incorrect.
- Constant K and H will transform to binary floating value when they are source data.



(K10) : [D6,D5] : (K2800) → M0, M1, M2
Binary converts to Floating | Binary Floating | Binary converts to Floating

Please set S1≤S2, when S2< S1, make S2 as the same value to S1.

Note: the compare value must be floating numbers, otherwise the result will be error.

4-9-3 Floating Addition[EADD,EDADD]

1) Summary

Floating Add [EADD, EDADD]			
16 bits	-	32 bits	EADD
Execution condition	Normally ON/OFF, rising/falling edge	Suitable Models	XD, XL
Hardware requirement	-	Software requirement	-
64 bits	EDADD		
Execution condition	Normal ON/OFF/falling or rising pulse edge	Suitable Models	XDH, XLH
Hardware requirement	Version V3.7.1 or later	Software requirement	Version V3.7.4a or later

2) Operands

Operands	Function	Data Type
S1	Addition operation data address	32/64 bits, BIN; REAL/LREAL
S2	Addition operation data address	32/64 bits, BIN; REAL/LREAL
D	Result address	32/64 bits, BIN; REAL/LREAL

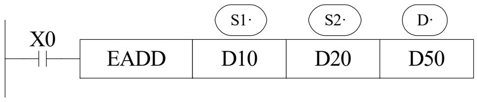
3) Suitable soft components

Operands	Word soft elements											Bit soft elements						
	System								Constant	Module		System						
	D	FD	TD	CD	DX	DY	DM	DS	K/H	ID	QD	X	Y	M	S	T	C	Dn.m
S1	●	●			●	●	●	●	●									
S2	●	●			●	●	●	●	●									
D	●					●	●	●										

*Notes: D includes D, HD; TD includes TD, HTD; CD includes CD, HCD, HSCD, HSD; DM includes DM, DHM; DS includes DS, DHS. M includes M, HM, SM; S includes S, HS; T includes T, HT; C includes C, HC.

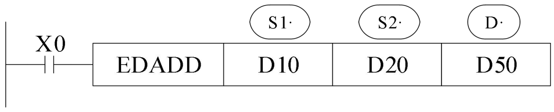
Description

<32 bits operation>



(D11,D10) + (D21,D20) → (D51,D50)

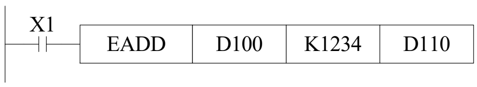
<64 bits operation>



(D13, D12,D11, D10)+ (D23, D22,D21, D20)→(D53, D52,D51, D50)
Binary Floating Binary Floating Binary Floating

- The two binary floating source data do addition operation, the result will be stored in target address.
- If a constant K or H used as source data, the value is converted to floating point before the addition operation.
- Before executing the instruction, both data must be floating-point numbers (if they are integers, they can be converted using FLT/DFLD instructions); Otherwise, the execution result will be incorrect.
- The registers in EDADD must start with an even address.

<32 bits operation>



(K1234) + (D101,D100) → (D111,D110)
Binary converts to Floating Binary Floating Binary Floating

<64 bits operation>

X1	EDADD	D100	K1234	D110

(K1234) + (D103, D102, D101, D100) →(D113, D112, D111, D110)

Binary converts to Floating Binary Floating Binary Floating

- The source data and result address can be the same. Please note that when X0 is ON, the instruction will be executed in every scanning period.

4-9-4 Floating Subtraction[ESUB,EDSUB]

1)Summary

Floating Sub [ESUB,EDSUB]			
16 bits	-	32 bits	ESUB
Execution condition	Normally ON/OFF, rising/falling edge	Suitable Models	XD, XL
Hardware requirement	-	Software requirement	-
64 bits	EDSUB		
Execution condition	Normal ON/OFF/falling or rising pulse edge	Suitable Models	XDH, XLH
Hardware requirement	Version V3.7.1 or later	Software requirement	Version V3.7.4a or later

2) Operands

Operands	Function	Data Type
S1	Subtraction operation data address	32/64 bits, BIN; REAL/LREAL
S2	Subtraction operation data address	32/64 bits, BIN; REAL/LREAL
D	Result address	32/64 bits, BIN; REAL/LREAL

3) Suitable soft components

Operands	Word soft elements											Bit soft elements						
	System								Constant	Module		System						
	D	FD	TD	CD	DX	DY	DM	DS	K/H	ID	QD	X	Y	M	S	T	C	Dn.m
S1	●	●			●	●	●	●	●									
S2	●	●			●	●	●	●	●									

D	●					●	●	●										

*Notes: D includes D, HD; TD includes TD, HTD; CD includes CD, HCD, HSCD, HSD; DM includes DM, DHM; DS includes DS, DHS. M includes M, HM, SM; S includes S, HS; T includes T, HT; C includes C, HC.

Description

<32 bits operation>

X0		S1·	S2·	D·
─┤├─	ESUB	D10	D20	D50

(D11,D10) － (D21,D20) → (D51,D50)
Binary Floating Binary Floating Binary Floating

<64 bits operation>

X0		S1·	S2·	D·
─┤├─	EDSUB	D10	D20	D50

(D13, D12,D11, D10)－(D23,D22,D21,D20) →(D53, D52,D51, D50)
Binary Floating Binary Floating Binary Floating

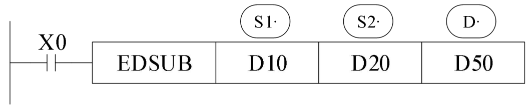
- The binary floating value S1 subtract S2, the result is stored in the target address.
- Before executing the instruction, both the decremental and the decremental data must be floating-point numbers (if they are integers, they can be converted using FLT/DFLD instructions); Otherwise, the execution result will be incorrect.
- The first address of the register in the EDSUB instruction must be an even number.
- If a constant K or H used as source data, the value is converted to floating point before the subtraction operation.

<32 bits operation>

X1				
─┤├─	ESUB	K1234	D100	D110

(K1234)－ (D101, D100) → (D111, D110)
Binary converts to Floating Binary Floating Binary Floating

<64 bits operation>



(D13, D12,D11, D10)－(D23,D22,D21,D20) →(D53, D52,D51, D50)
Binary converts to Floating Binary Floating Binary

- The source data and result address can be the same. Please note that when X0 is ON, the instruction will be executed in every scanning period.

4-9-5 Floating Multiplication[EMUL,EDMUL]

1)Summary

Floating Multiply [EMUL, EDMUL]			
16 bits	-	32 bits	EMUL
Execution condition	Normally ON/OFF, rising/falling edge	Suitable Models	XD, XL
Hardware requirement	-	Software requirement	-
64 bits	EDMUL		
Execution condition	Normal ON/OFF/falling or rising pulse edge	Suitable Models	XDH, XLH
Hardware requirement	Version V3.7.1 or later	Software requirement	Version V3.7.4a or later

2) Operands

Operands	Function	Data Type
S1	Multiplication operation data address	32 /64bits, BIN; REAL/LREAL
S2	Multiplication operation data address	32 /64bits, BIN; REAL/LREAL
D	Result address	32 /64bits, BIN; REAL/LREAL

3) Suitable soft components

Operands	Word soft elements											Bit soft elements						
	System								Constant	Module		System						
	D	FD	TD	CD	DX	DY	DM	DS	K/H	ID	QD	X	Y	M	S	T	C	Dn.m
S1	●	●			●	●	●	●	●									
S2	●	●			●	●	●	●	●									
D	●					●	●	●										

*Notes: D includes D, HD; TD includes TD, HTD; CD includes CD, HCD, HSCD, HSD; DM includes DM, DHM; DS includes DS, DHS. M includes M, HM, SM; S includes S, HS; T includes T, HT; C includes C, HC.

Description

<32 bits operation>

X0		S1·	S2·	D·
─┤├─	EMUL	D10	D20	D50

(D11, D10) × (D21,D20) → (D51,D50)
Binary Floating Binary Floating Binary Floating

<64 bits operation>

X0		S1·	S2·	D·
─┤├─	EDMUL	D10	D20	D50

(D13, D12,D11, D10)×(D23, D22,D21, D20)→(D53, D52,D51, D50)
Binary Floating Binary Floating Binary Floating

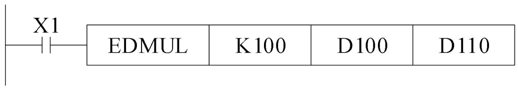
- The floating value of S1 is multiplied with the floating value point value of S2. The result of the multiplication is stored at D as a floating value.
- Before executing the instruction, both multiplied data must be floating-point numbers (if they are integers, they can be converted using FLT/DFLD instructions); Otherwise, the execution result will be incorrect.
- If a constant K or H used as source data, the value is converted to floating point before the multiplication operation.
- The registers in EDMUL must start with an even address.

<32 bits operation>

X1				
─┤├─	EMUL	K100	D100	D110

(K100)× (D101,D100)→ (D111,D110)
Binary converts to Floating Binary Floating Binary Floating

<64 bits operation>



(K00)×(D103, D102,D101, D100)→(D113, D112,D111, D110)

Binary converts to Floating   Binary Floating   Binary Floating

4-9-6 Floating Division[EDIV,EDDIV]

1) Summary

Floating Divide [EDIV, EDDIV]			
16 bits	-	32 bits	EDIV
Execution condition	Normally ON/OFF, rising/falling edge	Suitable Models	XD, XL
Hardware requirement	-	Software requirement	-
64 bits	EDDIV		
Execution condition	Normal ON/OFF/falling or rising pulse edge	Suitable Models	XDH, XLH
Hardware requirement	Version V3.7.1 or later	Software requirement	Version V3.7.4a or later

2) Operands

Operands	Function	Data Type
S1	Division operation data address	32/64 bits, BIN; REAL/LREAL
S2	Division operation data address	32/64 bits, BIN; REAL/LREAL
D	Result address	32/64 bits, BIN; REAL/LREAL

3) Suitable soft components

Operands	Word soft elements											Bit soft elements						
	System								Constant	Module		System						
	D	FD	TD	CD	DX	DY	DM	DS	K/H	ID	QD	X	Y	M	S	T	C	Dn.m
S1	●	●			●	●	●	●	●									
S2	●	●			●	●	●	●	●									
D	●					●	●	●										

*Notes: D includes D, HD; TD includes TD, HTD; CD includes CD, HCD, HSCD, HSD; DM includes DM, DHM; DS includes DS, DHS. M includes M, HM, SM; S includes S, HS; T includes T, HT; C includes C, HC.

Description

<32 bits operation>

		S1·	S2·	D·
X0	EDIV	D10	D20	D50

(D11,D10) ÷ (D21,D20) → (D51,D50)
Binary Floating Binary Floating Binary Floating

<64 bits operation>

		S1·	S2·	D·
X0	EDDIV	D10	D20	D50

(D13, D12,D11, D10)÷(D23, D22,D21, D20)→(D53, D52,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.
- If a constant K or H used as source data, the value is converted to floating point before the division operation.
- Before executing the instruction, both the divisor and dividend data must be floating-point numbers (if they are integers, they can be converted using FLT instructions); Otherwise, the execution result will be incorrect.
- The source data S2 is 0, the calculation will be error. The instruction will not work.
- The operand value must be floating numbers, otherwise the result will be error.

<32 bits operation>

X1	EDIV	D100	K100	D110

(D101, D100)÷ (K100) → (D111, D110)
Binary converts to Floating Binary Floating Binary Floating

<64 bits operation>

X1	EDDIV	D100	K100	D110

(D103, D102,D101, D100)÷(K100)→(D113, D112,D111, D110)
Binary converts to Floating Binary Floating Binary Floating

4-9-7 Float Square Root [ESQR]

1) Summary

Floating Square Root [ESQR]			
16 bits	-	32 bits	ESQR
Execution condition	Normally ON/OFF, rising/falling edge	Suitable Models	XD, XL
Hardware requirement	-	Software requirement	-

2) Operands

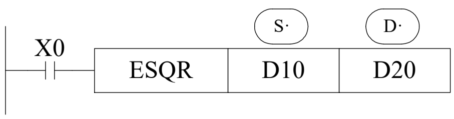
Operands	Function	Data Type
S	The soft element address need to do square root	32 bits, BIN; REAL
D	The result address	32 bits, BIN; REAL

3)Suitable soft components

Operands	Word soft elements											Bit soft elements						
	System								Constant	Module		System						
	D	FD	TD	CD	DX	DY	DM	DS	K/H	ID	QD	X	Y	M	S	T	C	Dn.m
S	●	●			●	●	●	●	●									
D	●					●	●	●										

*Notes: D includes D, HD; TD includes TD, HTD; CD includes CD, HCD, HSCD, HSD; DM includes DM, DHM; DS includes DS, DHS. M includes M, HM, SM; S includes S, HS; T includes T, HT; C includes C, HC.

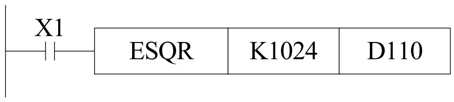
Description



(D11,D10) → (D21,D20)

Binary FloatingBinary Floating

- Perform the square root operation on the binary floating-point value specified by the source data within the component (must be a floating-point number, if it is an integer, it can be converted to a floating-point number using the FLT instruction), and store it as a binary floating-point number in the destination address.
- Before executing the instruction, the data to be square root must be a floating-point number (if it is an integer, it can be converted through FLT instruction); Otherwise, the execution result will be incorrect.
- If a constant K or H used as source data, the value is converted to floating point before the operation.



(K1024) → (D111, D110)

Binary converts to FloatingBinary 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 SM409 is set ON, SD409=7, the instruction can't be executed.

4-9-8 Sine[SIN]

1) Summary

Floating Sine[SIN]			
16 bits	-	32 bits	SIN
Execution condition	Normally ON/OFF, rising/falling edge	Suitable Models	XD, XL
Hardware requirement	-	Software requirement	-

2) Operands

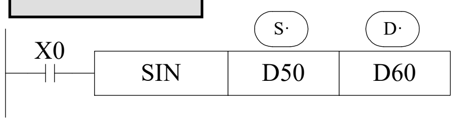
Operands	Function	Data Type
S	The soft element address need to do sine	32 bits, BIN; REAL
D	The result address	32 bits, BIN; REAL

3) Suitable soft components

Operands	Word soft elements											Bit soft elements						
	System								Constant	Module		System						
	D	FD	TD	CD	DX	DY	DM	DS	K/H	ID	QD	X	Y	M	S	T	C	Dn.m
S	●	●			●	●	●	●	●									
D	●					●	●	●										

*Notes: D includes D, HD; TD includes TD, HTD; CD includes CD, HCD, HSCD, HSD; DM includes DM, DHM; DS includes DS, DHS. M includes M, HM, SM; S includes S, HS; T includes T, HT; C includes C, HC.

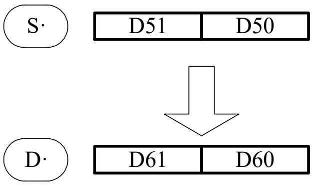
Description



(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

Note: the operand value must be floating numbers, otherwise the result will be error.

4-9-9 Cosine[COS]

1) Summary

Floating Cosine[COS]			
16 bits	-	32 bits	COS
Execution condition	Normally ON/OFF, rising/falling edge	Suitable Models	XD, XL
Hardware requirement	-	Software requirement	-

2) Operands

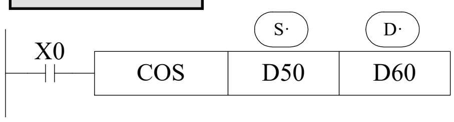
Operands	Function	Data Type
S	Soft element address need to do cos	32 bits, BIN; REAL
D	Result address	32 bits, BIN; REAL

3) Suitable soft components

Operands	Word soft elements											Bit soft elements						
	System								Constant	Module		System						
	D	FD	TD	CD	DX	DY	DM	DS	K/H	ID	QD	X	Y	M	S	T	C	Dn.m
S	●	●			●	●	●	●	●									
D	●					●	●	●										

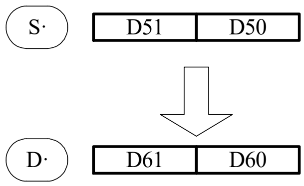
*Notes: D includes D, HD; TD includes TD, HTD; CD includes CD, HCD, HSCD, HSD; DM includes DM, DHM; DS includes DS, DHS. M includes M, HM, SM; S includes S, HS; T includes T, HT; C includes C, HC.

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

Note: Before the instruction is executed, the data in parameter S must be floating number; otherwise, the execution result will be wrong.

4-9-10 TAN [TAN]

1) Summary

TAN [TAN]			
16 bits	-	32 bits	TAN
Execution condition	Normally ON/OFF, rising/falling edge	Suitable Models	XD, XL
Hardware requirement	-	Software requirement	-

2) Operands

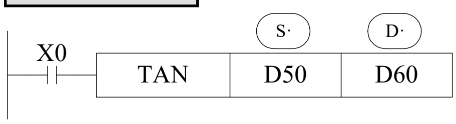
Operands	Function	Data Type
S	Soft element address need to do tan	32bit,BIN; REAL
D	Result address	32bit,BIN; REAL

3) Suitable soft components

Operands	Word soft elements											Bit soft elements						
	System								Constant	Module		System						
	D	FD	TD	CD	DX	DY	DM	DS	K/H	ID	QD	X	Y	M	S	T	C	Dn.m
S	●	●			●	●	●	●	●									
D	●					●	●	●										

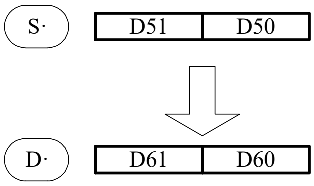
*Notes: D includes D, HD; TD includes TD, HTD; CD includes CD, HCD, HSCD, HSD; DM includes DM, DHM; DS includes DS, DHS. M includes M, HM, SM; S includes S, HS; T includes T, HT; C includes C, HC.

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

Note: Before the instruction is executed, the data in parameter S must be floating number; otherwise, the execution result will be wrong.

4-9-11 ASIN [ASIN]

1) Summary

ASIN [ASIN]			
16 bits	-	32 bits	ASIN
Execution condition	Normally ON/OFF, rising/falling edge	Suitable Models	XD, XL
Hardware requirement		Software requirement	-

2) Operands

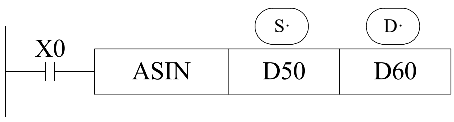
Operands	Function	Data Type
S	Soft element address need to do arcsin	32 bits, BIN; REAL
D	Result address	32 bits, BIN; REAL

3) Suitable soft components

Operands	Word soft elements											Bit soft elements						
	System								Constant	Module		System						
	D	FD	TD	CD	DX	DY	DM	DS	K/H	ID	QD	X	Y	M	S	T	C	Dn.m
S	●	●			●	●	●	●	●									
D	●					●	●	●										

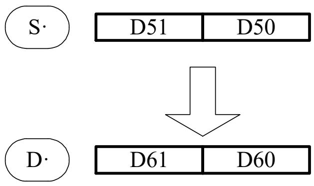
*Notes: D includes D, HD; TD includes TD, HTD; CD includes CD, HCD, HSCD, HSD; DM includes DM, DHM; DS includes DS, DHS. M includes M, HM, SM; S includes S, HS; T includes T, HT; C includes C, HC.

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

Note: Before the instruction is executed, the data in parameter S must be floating number; otherwise, the execution result will be wrong.

4-9-12 ACOS [ACOS]

1) Summary

ACOS [ACOS]			
16 bits	-	32 bits	ACOS
Execution condition	Normally ON/OFF, rising/falling edge	Suitable Models	XD, XL
Hardware requirement		Software requirement	-

2) Operands

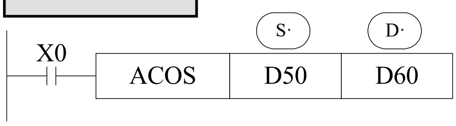
Operands	Function	Data Type
S	Soft element address need to do arccos	32 bits, BIN; REAL
D	Result address	32 bits, BIN; REAL

3)Suitable soft components

Operands	Word soft elements											Bit soft elements						
	System								Constant	Module		System						
	D	FD	TD	CD	DX	DY	DM	DS	K/H	ID	QD	X	Y	M	S	T	C	Dnm
S	●	●			●	●	●	●	●									
D	●					●	●	●										

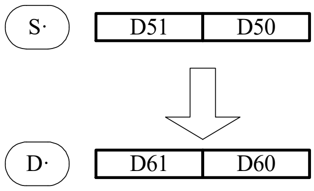
*Notes: D includes D, HD; TD includes TD, HTD; CD includes CD, HCD, HSCD, HSD; DM includes DM, DHM; DS includes DS, DHS. M includes M, HM, SM; S includes S, HS; T includes T, HT; C includes C, HC.

Description



(D51,D50)ACOS → (D61,D60)RAD
Binary Floating Binary Floating

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



ACOS value
Binary Floating
RAD value (angle×π/180)
Assign the binary floating value

Note: Before the instruction is executed, the data in parameter S must be floating number; otherwise, the execution result will be wrong.

4-9-13 ATAN [ATAN]

1) Summary

ATAN [ATAN]			
16 bits	-	32 bits	ATAN
Execution condition	Normally ON/OFF, rising/falling edge	Suitable Models	XD, XL
Hardware requirement		Software requirement	-

2) Operands

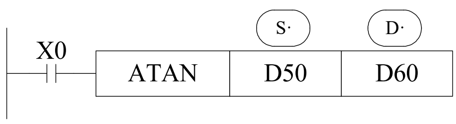
Operands	Function	Data Type
S	Soft element address need to do arctan	32 bits, BIN; REAL
D	Result address	32 bits, BIN; REAL

3) Suitable soft components

Operands	Word soft elements											Bit soft elements						
	System								Constant	Module		System						
	D	FD	TD	CD	DX	DY	DM	DS	K/H	ID	QD	X	Y	M	S	T	C	Dn.m
S	●	●			●	●	●	●	●									
D	●					●	●	●										

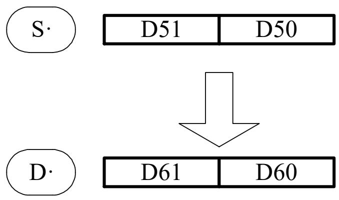
*Notes: D includes D, HD; TD includes TD, HTD; CD includes CD, HCD, HSCD, HSD; DM includes DM, DHM; DS includes DS, DHS. M includes M, HM, SM; S includes S, HS; T includes T, HT; C includes C, HC.

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

Note: Before the instruction is executed, the data in parameter S must be floating number; otherwise, the execution result will be wrong.

4-10 RTC Instructions

Mnemonic	Function	Chapter
TRD	Clock data read	4-10-1
TWR	Clock data write	4-10-2
MOV	Accurate clock BD board data read	4-10-3
TO	Accurate clock BD board data write	4-10-4
TADD	Clock data add	4-10-5
TSUB	Clock data sub	4-10-6
HTOS	Convert hour, minute, and second data to seconds	4-10-7
STOH	Convert second data to hours, minutes, and seconds	4-10-8
TCMP	Time (hours, minutes, seconds) compare	4-10-9
DACMP	Date (year, month, day) compare	4-10-10

※1: To use the instructions, The Model should be equipped with RTC function;
※2: There are some errors in the clock of XD/XL series PLC, which is about ±5 minutes per month. It can be calibrated regularly by HMI or in the PLC program.
※3: If high time accuracy is required, XD-RTC-BD can be used together ,with the error of about 13 seconds per month.

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

1) 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	XD, XL
Hardware requirement	-	Software requirement	-

2) Operands

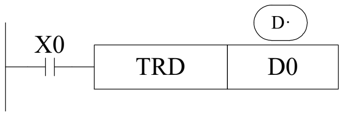
Operands	Function	Data Type
D	Register address to save clock data	16 bits, BIN

3) Suitable Soft Components

Operands	Word soft elements											Bit soft elements						
	System								Constant	Module		System						
	D	FD	TD	CD	DX	DY	DM	DS	K/H	ID	QD	X	Y	M	S	T	C	Dn.m
D	●		●	●														

*Notes: D includes D, HD; TD includes TD, HTD; CD includes CD, HCD, HSCD, HSD; DM includes DM, DHM; DS includes DS, DHS. M includes M, HM, SM; S includes S, HS; T includes T, HT; C includes C, HC.

Description



The instruction to read the clock data of the real-time clock (decimal) of the programmable controller into the data register (decimal).

- Read PLC's real time clock according to the following format.
  Read the special data register (SD013~SD019).

Special data register for real time clock t

Unit	Item	Clock data		Unit	Item
SD018	Year	0-99	→	D0	Year
SD017	Month	1-12	→	D1	Month
SD016	Date	1-31	→	D2	Date
SD015	Hour	0-23	→	D3	Hour
SD014	Minute	0-59	→	D4	Minute
SD013	Second	0-59	→	D5	Second
SD019	Week	1 (Mon.)-7(Sun.)	→	D6	Week

- Real time clock data is stored in decimal form in special data registers (SD013~SD019).
- The clock data read out through the TRD instruction is also in decimal form, and decimal can

be used for monitoring.

- After the instruction is executed once, the seven registers D0~D6 are all occupied, storing the year, month, day, hour, minute, second, week.
- Attention should be paid to avoiding special auxiliary registers for D· (see Appendix 2 for a list of special auxiliary data registers), as changes in related values may affect the use of the PLC.

4-10-2 Write Clock Data [TWR]

1) Summary

Write the clock data:

Write clock data [TWR]			
16 bits	TWR	32 bits	-
Execution condition	Edge triggering	Suitable Models	XD, XL
Hardware requirement	-	Software requirement	-

2) Operands

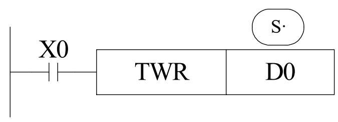
Operands	Function	Data Type
S	Write the clock data to the register	16 bits, BIN

3) Suitable Soft Components

Operands	Word soft elements											Bit soft elements						
	System								Constant	Module		System						
	D	FD	TD	CD	DX	DY	DM	DS	K/H	ID	QD	X	Y	M	S	T	C	Dn.m
D	●		●	●	●	●	●	●										

*Notes: D includes D, HD; TD includes TD, HTD; CD includes CD, HCD, HSCD, HSD; DM includes DM, DHM; DS includes DS, DHS. M includes M, HM, SM; S includes S, HS; T includes T, HT; C includes C, HC.

Description



The instruction to write clock data (decimal) into the real-time clock (decimal) of a programmable controller.

- Write the set clock data into PLC’s real time clock.
- In order to write real time clock, please set the 7 registers value from D0 to D6.

Data for clock setting

Unit	Item	Clock data		Unit	Item
D0	Year	0-99	→	SD018	Year
D1	Month	1-12	→	SD017	Month
D2	Date	1-31	→	SD016	Date
D3	Hour	0-23	→	SD015	Hour
D4	Minute	0-59	→	SD014	Minute
D5	Second	0-59	→	SD013	Second
D6	Week	1 (Mon.)-7 (Sun.)	→	SD019	Week

Special data register for real time clock t

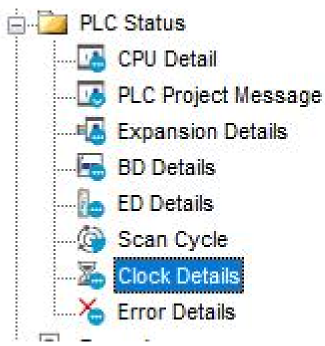
- Attention should be paid to avoiding special auxiliary registers (see Appendix 2 for a list of special auxiliary data registers), as changes in related values may affect the use of the PLC.
- When choosing secret download, the RTC only can be changed through TWR instruction.

After executing TWR instruction, the time in real time clock will immediately change to be the new time. It is a good idea to set the time few minutes late as the current time, and then drive the instruction when the real time reaches this value.

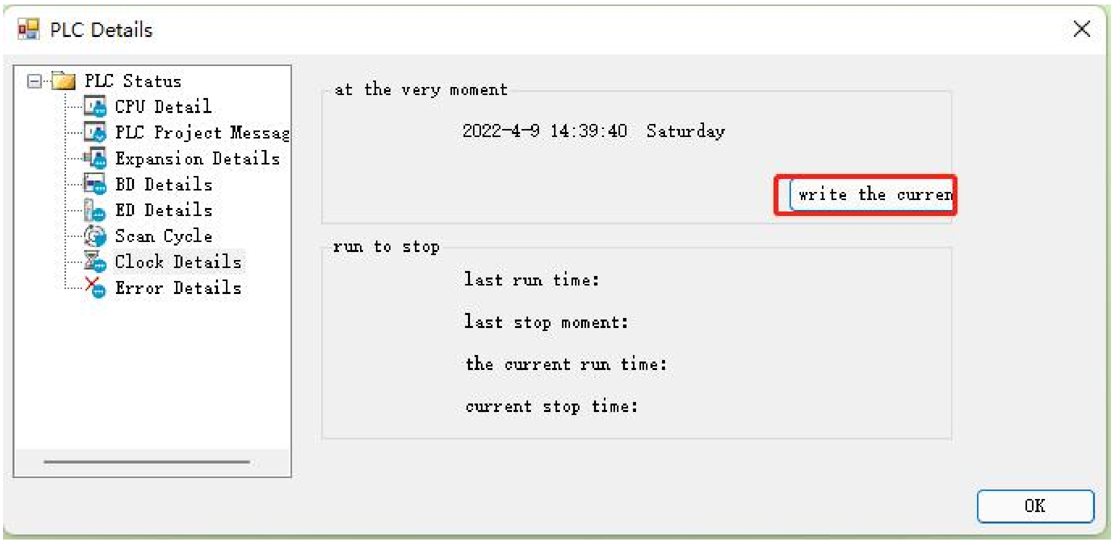
Note: When you select advanced mode when encrypting the download program, that is, when the password level is set to prohibit downloading, the PLC clock cannot be modified through communication, and can only be modified through TWR instructions.

There is another method to write the RTC.

In the XDPpro software, please click the clock details in project bar on the left.



Then click write into the current time.the PC will auto-write the current time to the PLC.

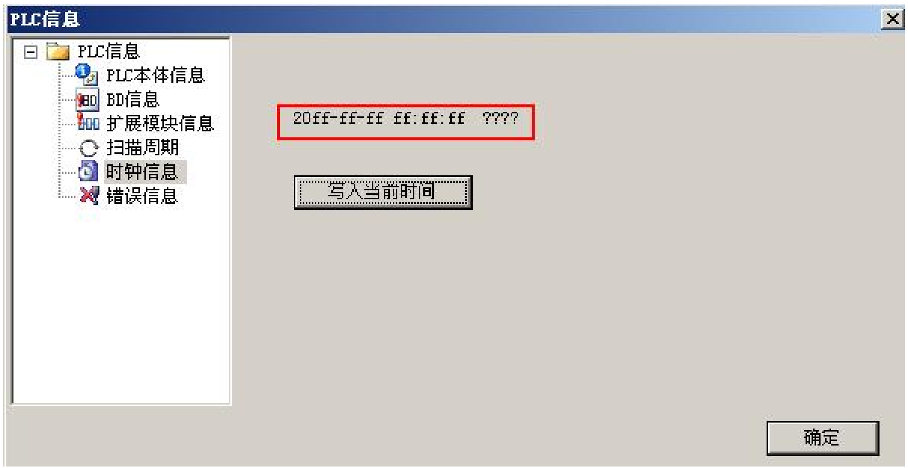


Then click write into the current time.the PC will auto-write the current time to the PLC.

How to know if the PLC has clock inside.

(1) Check the product label. There is clock logo on the label which means PLC has clock function.

(2) Check in the XDPpro software. If it shows as below, it means this PLC has no clock.



Note: If you have a clock, when you click on "Clock Information" and "Write Current Time", you can see that the seconds keep jumping in the software.

4-10-3 Accurate clock BD board data read [MOV]

1) Summary

Accurate clock BD board data read [MOV]			
16 bits	MOV	32 bits	-
Execution condition	Normally ON/OFF, rising/falling edge	Suitable Models	XD, XL
Hardware requirement	Version V3.4.6 (or V3.5.3a) or later	Software requirement	Version V3.5.3 or later

2) Operands

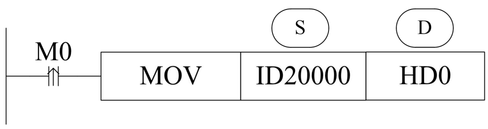
Operands	Function	Data Type
S	Soft component address of the clock data to read	16 bits, BIN
D	Register address to save clock data	16 bits, BIN

3) Suitable soft components

Operands	Word soft elements											Bit soft elements						
	System								Constant	Module		System						
	D	FD	TD	CD	DX	DY	DM	DS	K/H	ID	QD	X	Y	M	S	T	C	Dn.m
S	●																	
D	●																	

*Notes: D includes D, HD; TD includes TD, HTD; CD includes CD, HCD, HSCD, HSD; DM includes DM, DHM; DS includes DS, DHS. M includes M, HM, SM; S includes S, HS; T includes T, HT; C includes C, HC.

Description



When M0 is turned on, the "second" in #1 clock BD card XD-RTC-BD is read into the HD0 register of PLC.

- The data address of the BD board XD-RTC-BD is shown as follows.

#1 BD board address	#2 BD board address	Description	Clock Data	Remark
ID20000	ID20100	Second	0~59	Decimal
ID20001	ID20101	Minute	0~59	Decimal
ID20002	ID20102	Hour	0~23	Decimal
ID20003	ID20103	Date	1~31	Decimal
ID20004	ID20104	Month	1~12	Decimal
ID20005	ID20105	Year	00~99	Decimal
ID20006	ID20106	Week	1 (Mon.)-7 (Sun.)	Decimal

- Since the time in ID register is stored in the order of second, minute, hour, day, month, year, and week, it is not recommended to use BMOV or PMOV commands to read the clock data in batches if the read clock data is used for comparison and calculation.

4-10-4 Accurate clock BD board data write [TO]

1) Summary

Accurate clock BD board data write [TO]			
16 bits	TO	32 bits	-
Execution condition	rising/falling edge	Suitable Models	XD, XL
Hardware requirement	Version V3.4.6 (or V3.5.3a) or later	Software requirement	Version V3.5.3 or later

2) Operands

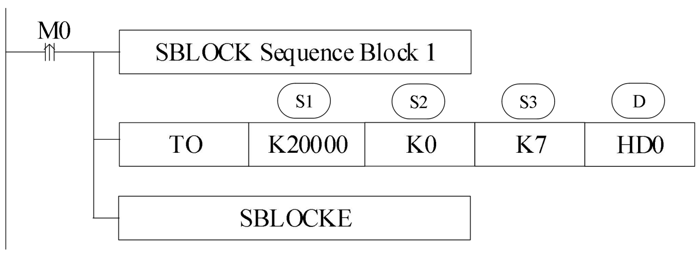
Operands	Function	Data Type
S1	Number of BD board	16 bits, BIN
S2	Soft component header address number for clock data	16 bits, BIN
S3	Number of clock data	16 bits, BIN
D	Soft component header address of local clock data	16 bits, BIN

3) Suitable soft components

Operands	Word soft elements											Bit soft elements						
	System								Constant	Module		System						
	D	FD	TD	CD	DX	DY	DM	DS	K/H	ID	QD	X	Y	M	S	T	C	Dnm
S1									●									
S2										●								
S3									●									
D	●																	

*Notes: D includes D, HD; TD includes TD, HTD; CD includes CD, HCD, HSCD, HSD; DM includes DM, DHM; DS includes DS, DHS. M includes M, HM, SM; S includes S, HS; T includes T, HT; C includes C, HC.

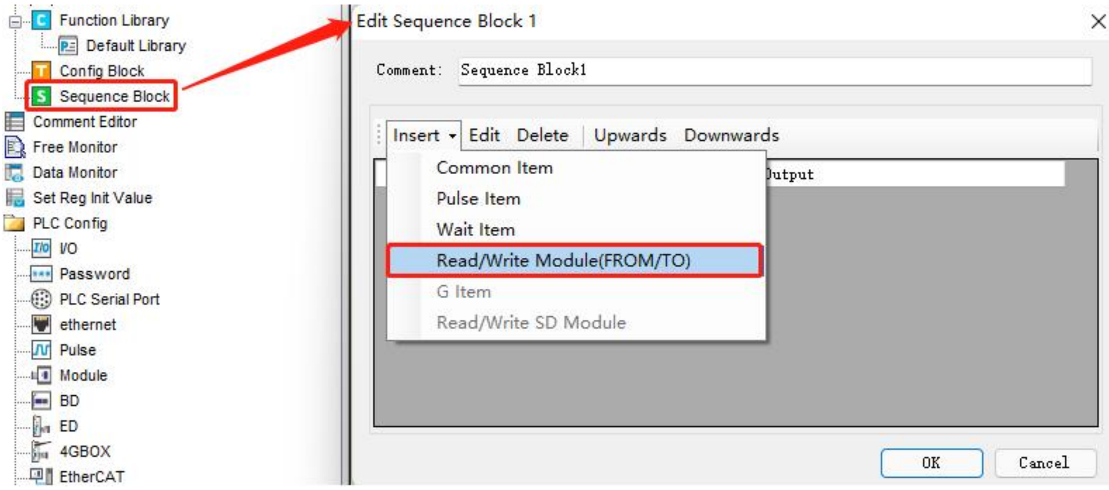
Description



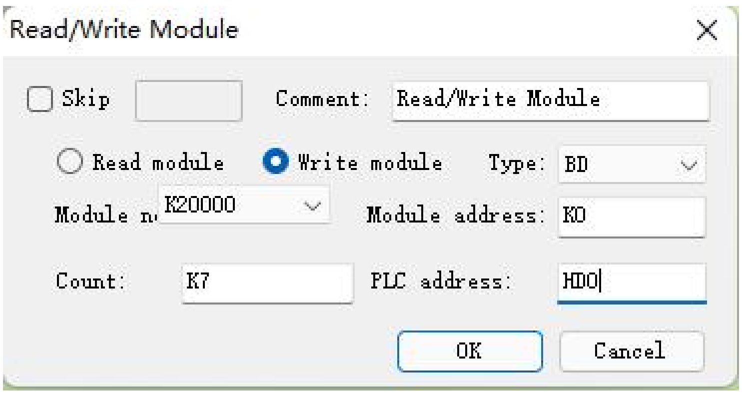
- When M0 is turned on, write the values in the 7 consecutive registers led by HD0 into the ID register in #1 clock BD board XD-RTC-BD.

Source data		#1 BD board address	Description	Clock Data	Remark
HD0	→	ID20000(K0)	Second	0~59	Decimal
HD1		ID20001(K1)	Minute	0~59	Decimal
HD2	→	ID20002(K2)	Hour	0~23	Decimal
HD3		ID20003(K3)	Day	1~31	Decimal
HD4	→	ID20004(K4)	Month	1~12	Decimal
HD5		ID20005(K5)	Year	00~99	Decimal
HD6	→	ID20006(K6)	Week	1 (Mon.)-7 (Sun.)	Decimal

● TO command needs to be entered in Sequence Block. The operation procedure is as follows. Open XDPpro software, click sequence block function, in the pop-up window, click "Insert" - "Read/Write Module (FROM/TO)":



In the configuration window that is displayed, set the parameters as follows:



Note: Module number K20000 stands for #1 BD, K20001 stands for #2 BD; Module addresses are numbered from K0, corresponding to ID20000, ID20001...... ID20006.

4-10-5 Clock data add [TADD]

1) Summary

Clock data add [TADD]			
16 bits	TADD	32 bits	-
Execution condition	Normally ON/OFF, rising/falling edge	Suitable Models	XD, XL
Hardware requirement	Version V3.4.6 (or V3.5.3a) or later	Software requirement	Version V3.5.3 or later

2) Operands

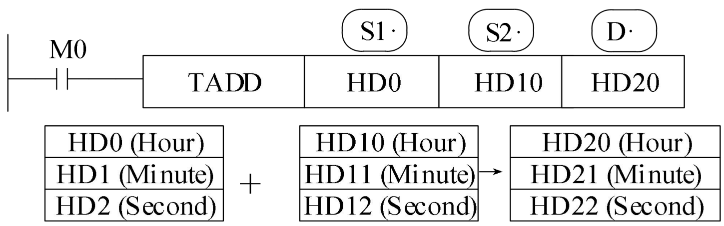
Operands	Function	Data Type
S1	Soft element header address of the clock data (hour, minute, second)	16 bits, BIN
S2	Soft element header address of the clock data (hour, minute, second)	16 bits, BIN
D	The result address	16 bits, BIN

3) Suitable soft components

Operands	Word soft elements											Bit soft elements						
	System								Constant	Module		System						
	D	FD	TD	CD	DX	DY	DM	DS	K/H	ID	QD	X	Y	M	S	T	C	Dn.m
S1	●																	
S2	●																	
D	●																	

*Notes: D includes D, HD; TD includes TD, HTD; CD includes CD, HCD, HSCD, HSD; DM includes DM, DHM; DS includes DS, DHS. M includes M, HM, SM; S includes S, HS; T includes T, HT; C includes C, HC.

Description



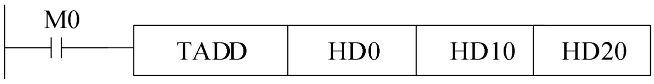
- When M0 is turned on, perform clock addition operation to add the clock data (hour, minute, second) of (HD0, HD1, HD2) to the clock data (hour, minute, second) of (HD10, HD11, HD12), and store the operation results (hour, minute, second) in (HD20, HD21, HD22). Note: The correspondence between registers is fixed, that is, they are stored in the order of hours, minutes, and seconds.
- Range of time: 0~23; Range of minutes: 0~59, range of seconds: 0~59.
- If the seconds and minutes after addition exceed 59, subtract 60 from the result and save it in the seconds and minutes register. At the same time, the values of minutes and hours are automatically incremented by 1;
- If the hour after addition exceeds 23, subtract 24 from the result and save it in the time

register, while setting the carry flag SM22 to ON.

- If the operation result is 0 hours, 0 minutes, or 0 seconds, the zero flag SM20 will be set to ON.
- The operands S1, S2, and D each occupy three consecutive registers and should not be used for other purposes.

Example 1:

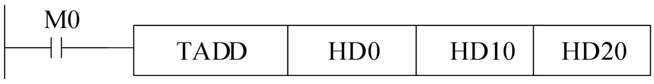
<General condition>



hour	HD0: 3		HD10: 7		HD20: 10
minute	HD1: 10	+	HD11: 30	→	HD21: 40
second	HD2: 8		HD12: 15		HD22: 23

Example 2:

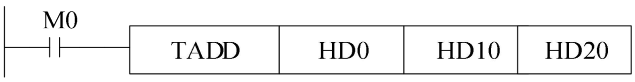
< More than 59 seconds >



hour	HD0: 3		HD10: 7		HD20: 10
minute	HD1: 10	+	HD11: 30	→	HD21: 41
second	HD2: 40		HD12: 30		HD22: 10

Example 3:

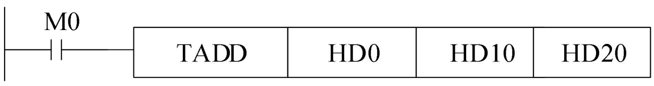
< More than 59 minutes >



hour	HD0: 3		HD10: 7		HD20: 11
minute	HD1: 35	+	HD11: 30	→	HD21: 6
second	HD2: 40		HD12: 30		HD22: 10

Example 4:

< More than 23 hours >



hour	HD0: 17		HD10: 7		HD20: 1
minute	HD1: 35	+	HD11: 30	→	HD21: 6
second	HD2: 40		HD12: 30		HD22: 10

4-10-6 Clock data sub [TSUB]

1) Summary

Clock data sub [TSUB]			
16 bits	TSUB	32 bits	-
Execution condition	Normally ON/OFF, rising/falling edge	Suitable Models	XD, XL
Hardware requirement	Version V3.4.6 (or V3.5.3a) or later	Software requirement	Version V3.5.3 or later

2) Operands

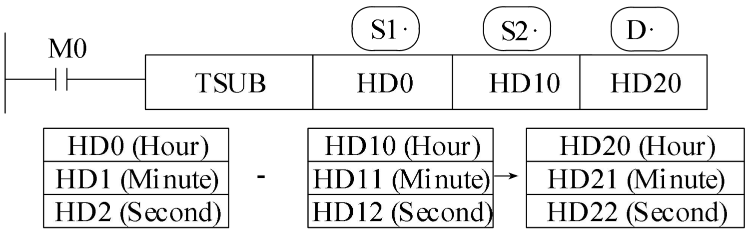
Operands	Function	Data Type
S1	Soft element header address of the clock data (hour, minute, second)	16 bits, BIN
S2	Soft element header address of the clock data (hour, minute, second)	16 bits, BIN
D	The result address	16 bits, BIN

3) Suitable soft components

Operands	Word soft elements											Bit soft elements						
	System								Constant	Module		System						
	D	FD	TD	CD	DX	DY	DM	DS	K/H	ID	QD	X	Y	M	S	T	C	Dn.m
S1	●																	
S2	●																	
D	●																	

*Notes: D includes D, HD; TD includes TD, HTD; CD includes CD, HCD, HSCD, HSD; DM includes DM, DHM; DS includes DS, DHS. M includes M, HM, SM; S includes S, HS; T includes T, HT; C includes C, HC.

Description

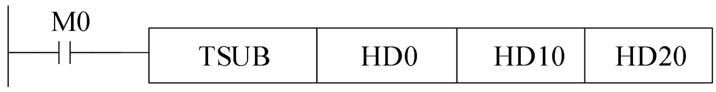


- When M0 is turned on, perform clock subtraction operation to subtract the clock data (hour, minute, second) of (HD0, HD1, HD2) from the clock data (hour, minute, second) of (HD10, HD11, HD12), and store the operation results (hour, minute, second) in (HD20, HD21, HD22).
- Note: The correspondence between registers is fixed, that is, they are stored in the order of hours, minutes, and seconds.
- Range of time: 0~23; Range of minutes: 0~59, range of seconds: 0~59.
- If the subtracted seconds and minutes are less than 0, add 60 to the result and save it in the seconds and minutes register. At the same time, the values of minutes and hours will be automatically reduced by 1;

- If the hour after subtraction is less than 0, add 24 to the result and save it in the time register, while setting the borrow flag SM21 to ON.
- If the operation result is 0 hours, 0 minutes, or 0 seconds, the zero flag SM20 will be set to ON.
- The operands S1, S2, and D each occupy three consecutive registers and should not be used for other purposes.

Example 1:

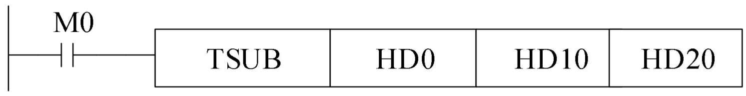
<General condition>



hour	HD0：7		HD10：3		HD20：4
minute	HD1：30	—	HD11：10	→	HD21：20
second	HD2：15		HD12：8		HD22：7

Example 2:

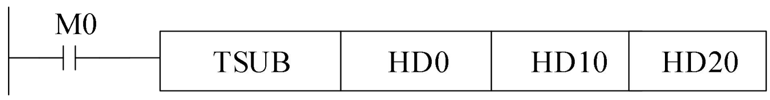
《Less than 0 seconds》



hour	HD0：7		HD10：3		HD20：4
minute	HD1：30	—	HD11：10	→	HD21：19
second	HD2：30		HD12：40		HD22：50

Example 3:

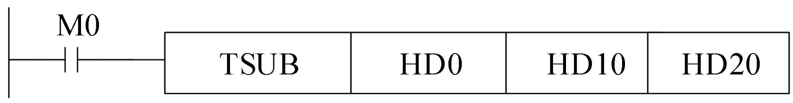
《Less than 0 minutes》



hour	HD0：7		HD10：3		HD20：3
minute	HD1：30	—	HD11：35	→	HD21：55
second	HD2：40		HD12：30		HD22：10

Example 4:

《Less than 0 hours》



hour	HD0：7		HD10：10		HD20：21
minute	HD1：35	—	HD11：30	→	HD21：5
second	HD2：40		HD12：30		HD22：10

4-10-7 Convert hour, minute, and second data to seconds [HTOS]

1) Summary

Convert hour, minute, and second data to seconds [HTOS]			
16 bits	-	32 bits	HTOS
Execution condition	Normally ON/OFF, rising/falling edge	Suitable Models	XD, XL
Hardware requirement	Version V3.4.6 (or V3.5.3a) or later	Software requirement	Version V3.5.3 or later

2) Operands

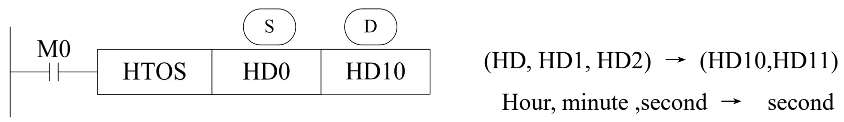
Operands	Function	Data Type
S	Clock data before conversion	16 bits, BIN
D	Clock data after conversion	16 bits, BIN

3) Suitable soft components

Operands	Word soft elements											Bit soft elements						
	System								Constant	Module		System						
	D	FD	TD	CD	DX	DY	DM	DS	K/H	ID	QD	X	Y	M	S	T	C	Dn.m
S	●																	
D	●																	

*Notes: D includes D, HD; TD includes TD, HTD; CD includes CD, HCD, HSCD, HSD; DM includes DM, DHM; DS includes DS, DHS. M includes M, HM, SM; S includes S, HS; T includes T, HT; C includes C, HC.

Description



- When the M0 switches on, it converts clock data (hours, minutes and seconds) in three consecutive registers led by HD0 into second data, which is stored in register HD10 (double word).
- Note: the correspondence of registers is fixed, that is, they are stored in order of hours, minutes and seconds.
- The operands S occupy three registers. Do not use them for other purposes.

4-10-8 Convert second data to hours, minutes, and seconds[STOH]

1) Summary

Convert hour, minute, and second data to seconds [STOH]			
16 bits	-	32 bits	STOH
Execution condition	Normally ON/OFF, rising/falling edge	Suitable Models	XD, XL
Hardware requirement	Version V3.4.6 (or V3.5.3a) or later	Software requirement	Version V3.5.3 or later

2) Operands

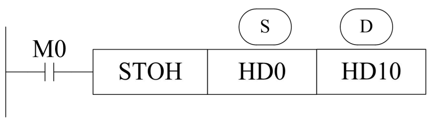
Operands	Function	Data Type
S	Clock data before conversion	16 bits, BIN
D	Clock data after conversion	16 bits, BIN

3) Suitable soft components

Operands	Word soft elements											Bit soft elements						
	System								Constant	Module		System						
	D	FD	TD	CD	DX	DY	DM	DS	K/H	ID	QD	X	Y	M	S	T	C	Dn.m
S	●																	
D	●																	

*Notes: D includes D, HD; TD includes TD, HTD; CD includes CD, HCD, HSCD, HSD; DM includes DM, DHM; DS includes DS, DHS. M includes M, HM, SM; S includes S, HS; T includes T, HT; C includes C, HC.

Description



(HD0, HD1) → (HD10, HD11, HD12)

second → hour, minute, second

- When the M0 switches on, it converts clock data (hours, minutes and seconds) in three consecutive registers led by HD0 into second data, which is stored in register HD10 (double word).
- Note: the correspondence of registers is fixed, that is, they are stored in order of hours, minutes and seconds.
- The operands S occupy three registers. Do not use them for other purposes.

4-10-9 Clock compare [TCMP]

1) Summary

Compare three continuous clocks time.

Clock compare [TCMP]			
16 bits	TCMP	32 bits	-
Condition	Normally ON/OFF, rising/falling edge	Suitable model	XD, XL
Hardware	Version V3.4.6 (or V3.5.3a) or later	Software	Version V3.5.3 or later

2) Operands

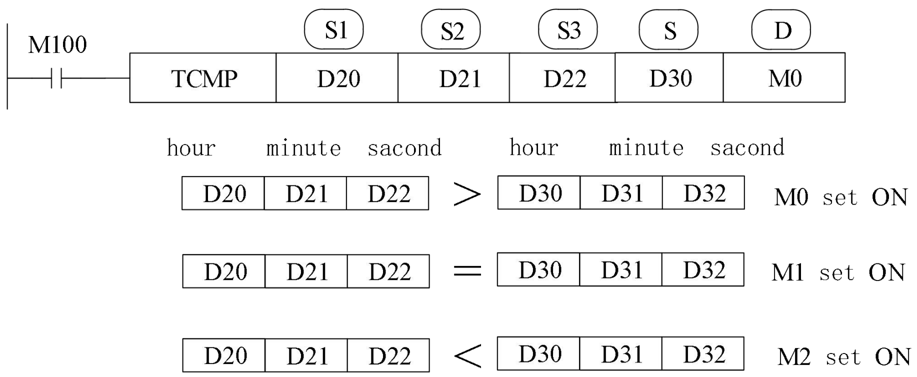
Operands	Function	Model
S1	Soft component address for hours	16 bits, BIN
S2	Soft component address for minutes	16 bits, BIN
S3	Soft component address for seconds	16 bits, BIN
S4	PLC real time clock information first address	16 bits, BIN
D2	The compare result first address	bit

3) suitable soft component

Operands	Word soft elements											Bit soft elements						
	System								Constant	Module		System						
	D	FD	TD	CD	DX	DY	DM	DS	K/H	ID	QD	X	Y	M	S	T	C	Dn.m
S1	●	●																
S2	●	●																
S3	●	●																
S	●	●																
D													●	●				

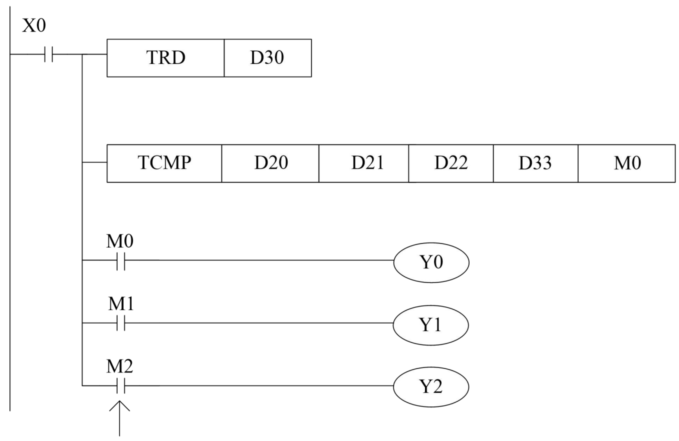
*Notes: D includes D, HD; TD includes TD, HTD; CD includes CD, HCD, HSCD, HSD; DM includes DM, DHM; DS includes DS, DHS. M includes M, HM, SM; S includes S, HS; T includes T, HT; C includes C, HC.

Description



- When M100 changes from OFF to ON, the TCMP instruction is executed, comparing the three registers starting from D30 (clock hour, minute, and second) with the hour, minute, and second composed of D20, D21, and D22, and turning on the corresponding coil according to the comparison result. When the hours, minutes, and seconds composed of D20, D21, and D22 are greater than D30, D31, and D32, M0 is set to ON; When the hours, minutes, and seconds composed of D20, D21, and D22 are equal to D30, D31, and D32, M1 is set to ON. When the hours, minutes, and seconds composed of D20, D21, and D22 are less than D30, D31, and D32, M2 is set to ON.
- When M100 is in the ON state, if the reference time (D20, D21, D22) or clock data (D30, D31, D32) changes, the comparison result will also change accordingly.
- When M100 is turned off to stop executing the TCMP command, M0~M2 still maintain the state before M100 was turned off

For example:



For example, in the above example, if the current clock is read as Wednesday, July 30, 2014 at 15:32:49, then D33=15, D34=32, D35=49. If the set clock is 16:40:21, that is, D20=16, D21=40, D22=21, then Y0=ON; If the set clock is 15:21:16, which means D20=15, D21=21, D22=16, then Y2=ON; If the set clock is 15:32:49, which means D20=15, D21=32, D22=49, then Y1=ON.

4-10-10 Date (year, month, day) compare [DACMP]

1) Summary

Convert hour, minute, and second data to seconds [STOH]			
16 bits	DACMP	32 bits	-
Execution condition	Normally ON/OFF, rising/falling edge	Suitable Models	XD, XL
Hardware requirement	Version V3.4.6 (or V3.5.3a) or later	Software requirement	Version V3.5.3 or later

2) Operands

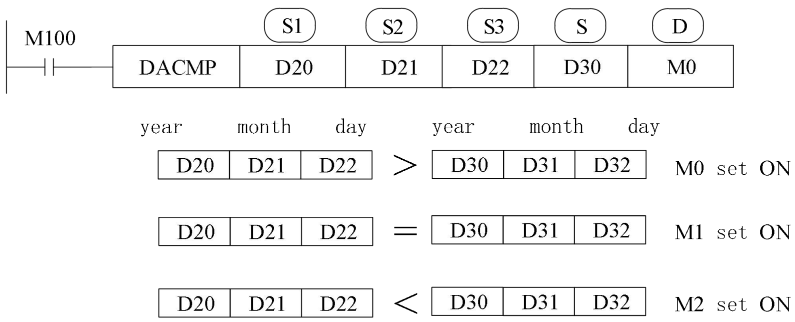
Operands	Function	Model
S1	Soft component address for years	16 bits, BIN
S2	Soft component address for months	16 bits, BIN
S3	Soft component address for days	16 bits, BIN
S4	PLC real time clock information first address	16 bits, BIN
D2	The compare result first address	bit

3) Suitable soft component

Operands	Word soft elements											Bit soft elements						
	System								Constant	Module		System						
	D	FD	TD	CD	DX	DY	DM	DS	K/H	ID	QD	X	Y	M	S	T	C	Dn.m
S1	●	●																
S2	●	●																
S3	●	●																
S	●	●																
D													●	●				

*Notes: D includes D, HD; TD includes TD, HTD; CD includes CD, HCD, HSCD, HSD; DM includes DM, DHM; DS includes DS, DHS. M includes M, HM, SM; S includes S, HS; T includes T, HT; C includes C, HC.

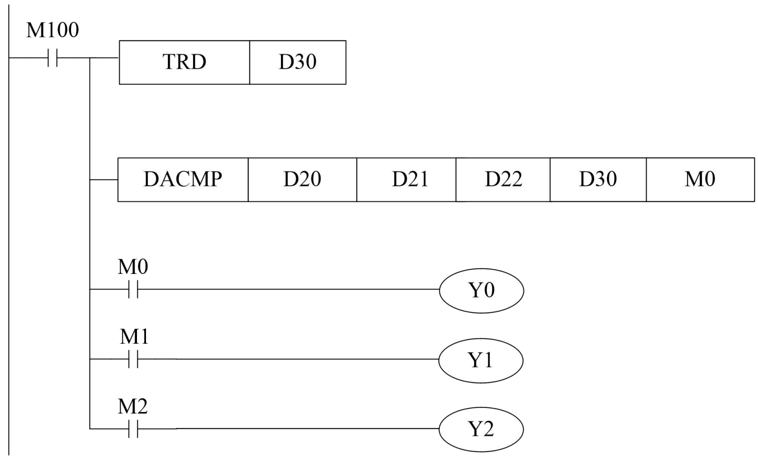
Description



- When M100 changes from OFF to ON, the DACMP instruction is executed, comparing the three registers starting from D30 (the year, month, and day of the clock) with the year, month, and day composed of D20, D21, and D22, and setting the corresponding coil ON according to the comparison result. When the year, month, and day composed of D20, D21, and D22 are greater than D30, D31, and D32, M0 is set to ON; When the year, month, and day composed of D20, D21, and D22 are equal to D30, D31, and D32, then M1 is set to ON; When the year, month, and day composed of D20, D21, and D22 are less than D30, D31, and D32, M2 is set to ON.

- When M100 is in the ON state, if the reference date (D20, D21, D22) or date data (D30, D31, D32) changes, the comparison result will also change accordingly.
- When M100 is turned off to stop executing the TCMP command, M0~M2 still maintain the state before M100 was turned off.

For example:



For example, in the above example, if the current clock is read as Wednesday, July 30, 2014 at 15:32:49, then D30=14, D31=7, D32=30. If the set date is August 17, 2014, that is, D20=14, D21=8, D22=17, then Y0=ON; If the set date is April 23, 2014, i.e. D20=14, D21=4, D22=23, then Y2=ON; If the set date is July 30th, 2014, i.e. D20=14, D21=7, D22=30, then Y1=ON.

5 HIGH SPEED COUNTER (HSC)

This chapter will introduce high speed counter's functions, including high speed count model, wiring method, read/write HSC value, reset etc.

Instructions List for HSC

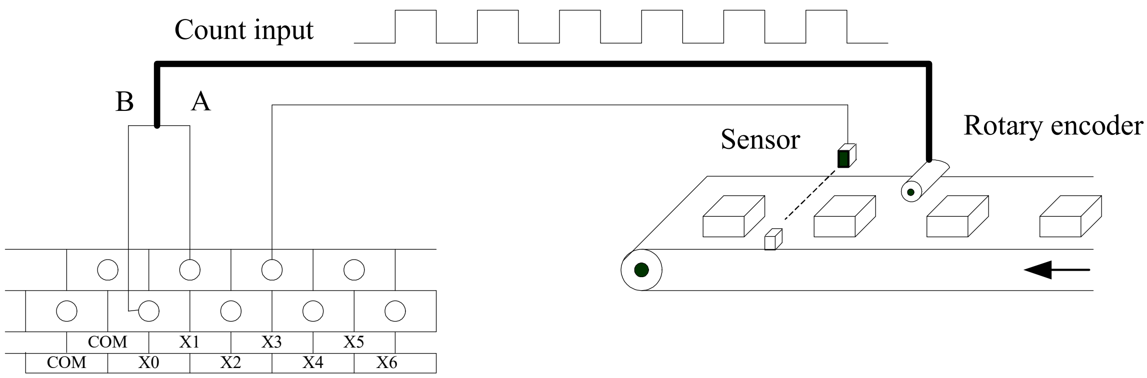
Instruction name	Function	Instruction	Chapter
HSC read/write			
CNT	No 24-segments single phase	CNT HSC0 K1000	5-7-1
CNT_AB	No 24-segments AB phase	CNT_AB HSC0 K1000	5-7-2
RST	HSC reset	RST HSC0	5-7-3
DMOV	HSC read	DMOV HSC0 D0	5-7-4
DMOV	HSC write	DMOV D4000 HSC0	5-7-5
CNT	Single-phase 100-segments high-speed counting (with interruption)	CNT HSC0 K1000 D0	5-9-2
CNT_AB	AB phase 100-segments high speed counting (with interruption)	CNT_AB HSC0 K1000 D0	5-9-3

5-1 Functions Summary

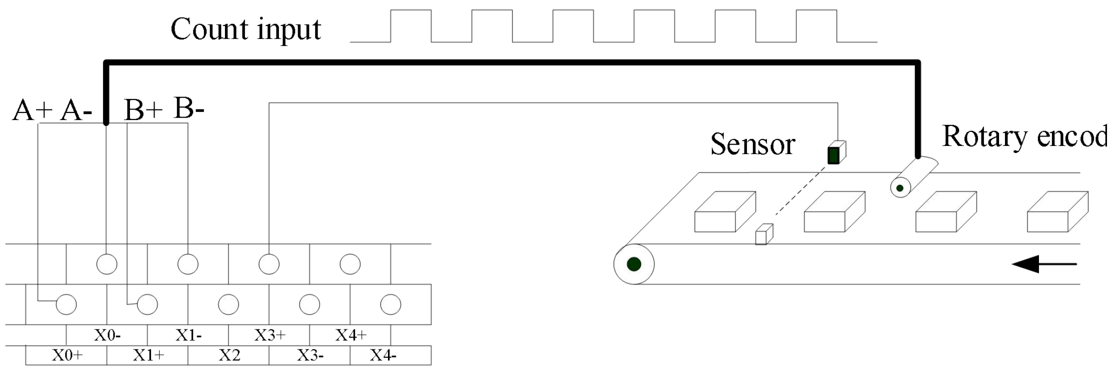
XD, XL series PLC has HSC (High Speed Counter) function which will not affect by the scanning cycle. Via choosing different counter, test the high speed input signals with detect sensors and rotary encoders. The highest testing frequency can reach 80KHz.

Note:

(1) For PLC with NPN input mode, please choose the encoder with NPN open collector output (OC) of DC24V; for PLC with PNP input mode, please choose the encoder with PNP open collector output (OC) of DC24V.



(2) The high-speed counting input of XD5-48D4T4 can receive differential signal (DIFF), please be sure to choose differential signal (DIFF) encoder.



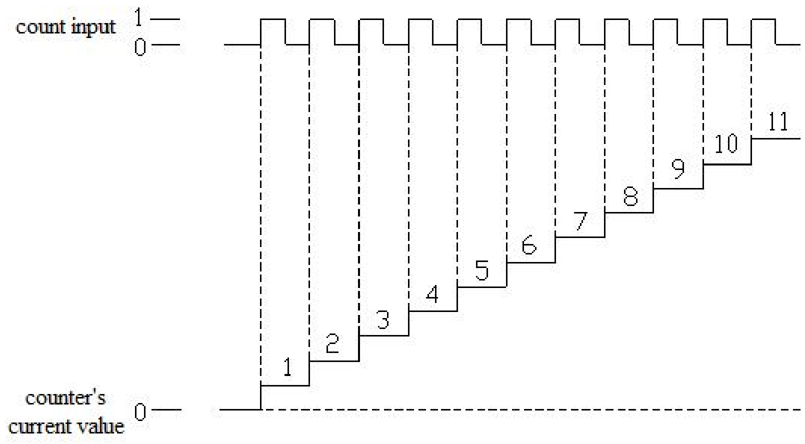
(3) When the counting frequency is higher than 25Hz, please select a high-speed counter.

5-2 HSC Mode

XD, XLseries high speed counter has two working mode: Single-phase increasing mode and AB phase mode.

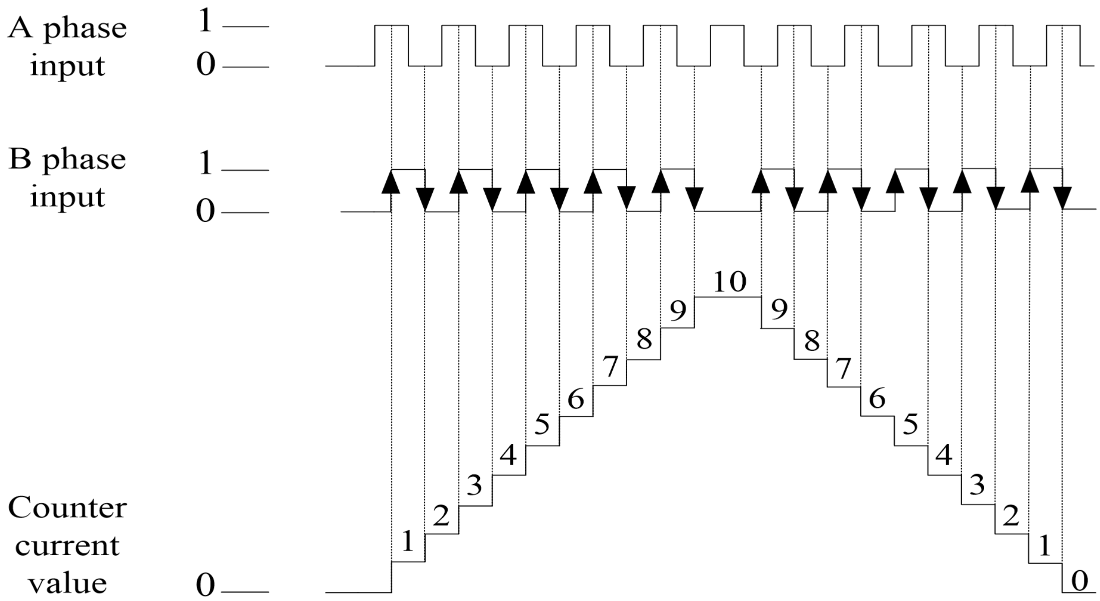
Single-phase Increasing Mode

Under this mode, the count value increase at each pulse’s rising edge;

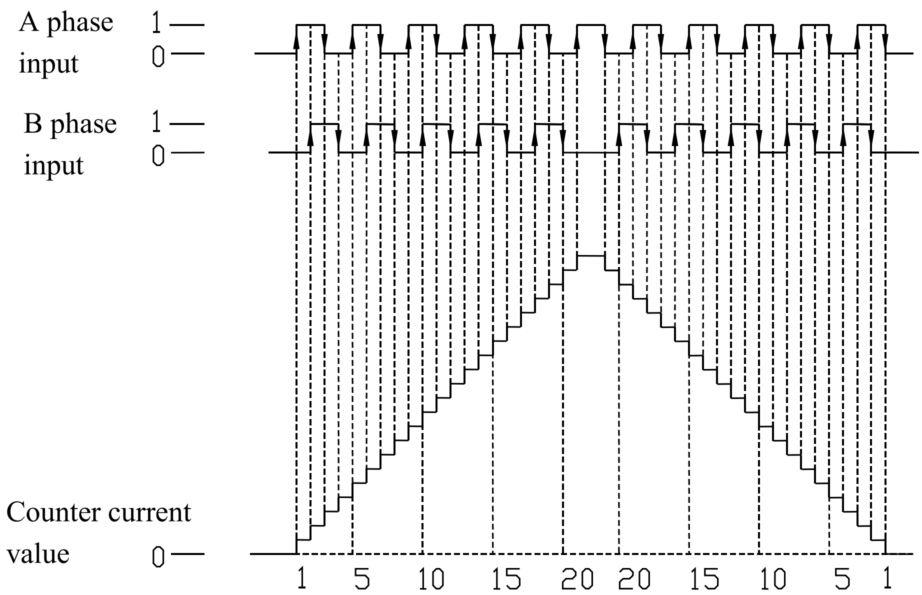


AB Phase Mode

Under this mode, the HSC value increase or decrease according to two differential signal (A phase and B phase). According to the multiplication, we have 2-time frequency and 4-time frequency, but the default count mode is 4-time frequency mode.
2-time frequency and 4-time frequency modes are shown below:
2-time Frequency



4-time Frequency

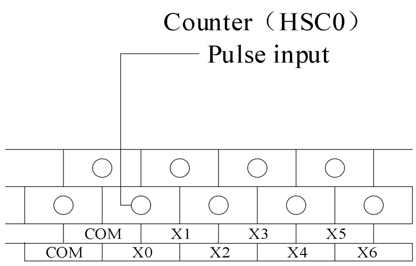


5-3 HSC Range

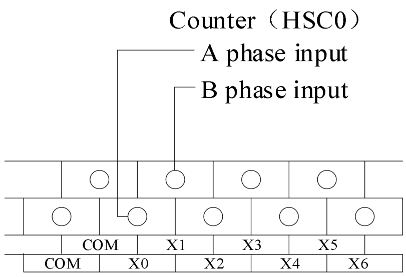
HSC's count range is: K-2,147,483,648 ~ K+2,147,483,647. If the count value overflows this range, then overflow or underflow appears;
Overflow means the count value jumps from K+2,147,483,647 to K-2,147,483,648, then continue counting; underflow means the count value jumps from -2,147,483,648 to +2,147,483,647 then continue counting.

5-4 HSC Input Wiring

For the counter's pulse input wiring, things differ with different PLC model and counter model; several typical input wiring diagrams are shown below: (take XD3-60 HSC0 as the example):



Increasing mode



AB phase mode

5-5 HSC ports assignment

1) XD series PLC HSC channels list:

PLC model		HSC channel	
		Increasing mode	AB phase mode
XD1	10/16/24/32	0	0
XD2/XD3	16/24/32/42/48/60	3	3
XD5	16/24/32/42/48/60/80	3	3
	24T4/32T4/48T4/60T4	4	4
	24D2T2	2	2
	48D4T4	8	8
	48T6/60T6	6	6
	60T10	10	10
XDM	24T4/32T4/48T4/60T4	4	4
	60T10	10	10
XDC	24/32/48/60	4	4
XD3E	24 points	3	3
XD5E	24/30/48/60	3	3
	30T4	4	4
	60T4	4	4
	60T6	6	6
	60T10	10	10
XDME	30T4/60T4	4	4
	60T10	10	10
XDH	30A16(L)/60T4/60A32/60A64	4	4
XL1	16	0	0
XL3	16/32	3	3
XL5	16/32	3	3
	32T4	4	4
	64T10	10	10
XL5E	16/32	3	3
	32T4	4	4
	64T6	6	6
	64T10	10	10
XL5N	32	3	3
XLME	32T4	4	4
	64T10	10	10
XLH	24A16(L)/30A32（L）	4	4
XL5H	24A8L	3	3

Note: The hardware versions of XL5E-64T6 starting with H4 support six-channel high-speed counting, while versions starting with H3 only support four-channel high-speed counting.

2)Each letter’s Meaning:

U	A	B	Z
Pulse input	A phase input	B phase input	Z phase pulse catching

Note: Z phase signal counting function is in developping.

Under normal conditions, input frequency of X0 and X1 can reach 80KHz and 50KHz respectively in single-phase and AB phase modes. The other terminals have maximum frequency of 10KHz and 5KHz respectively in single-phase and AB phase modes.

X can use as normal input terminals when there are no high speed pulses input. In the following table, 2 means double frequency; 4 means quadruple frequency; 2/4 means that double frequency and quadruple frequency can be adjusted.

XD2-16												
	Increasing mode							AB phase mode				
	HSC0	HSC2	HSC4	HSC6	HSC8	HSC10	HSC12	HSC0	HSC2	HSC4	HSC6	HSC8
Max frequency	10K	10K	10K					5K	5K	5K		
Quadruple frequency								2/4	2/4	2/4		
Counter interruption	√	√	√					√	√	√		
X000	U							A				
X001								B				
X002								Z				
X003		U							A			
X004									B			
X005									Z			
X006			U							A		
X007										B		
X010										Z		

XD3-16, XL3-16												
	Increasing mode							AB phase mode				
	HSC0	HSC2	HSC4	HSC6	HSC8	HSC10	HSC12	HSC0	HSC2	HSC4	HSC6	HSC8
Max frequency	80K	10K	10K					50K	5K	5K		
Quadruple frequency								2/4	2/4	2/4		
Counter interruption	√	√	√					√	√	√		
X000	U							A				
X001								B				
X002								Z				
X003		U							A			
X004									B			
X005									Z			
X006			U							A		
X007										B		
X010										Z		

XD2-24/32/42/48/60, XD3-24/32/42/48/60, XD5-16/24/32/42/48/60/80, XD3E-24, XD5E-24/30/48/60, XL3-32, XL5-16/32, XL5E-16/32, XL5H-24A8L, XL5N-32T												
	Increasing mode							AB phase mode				
	HSC0	HSC2	HSC4	HSC6	HSC8	HSC10	HSC12	HSC0	HSC2	HSC4	HSC6	HSC8
Max frequency	80K	80K	10K					50K	50K	5K		
Quadruple frequency								2/4	2/4	2/4		
Counter interruption	√	√	√					√	√	√		
X000	U							A				
X001								B				
X002								Z				

X003		U							A			
X004									B			
X005									Z			
X006			U							A		
X007										B		
X010										Z		

XD5-24T4/32T4/48T4/60T4, XDM-24T4/32T4/60T4/60T4L, XDC-24/32/48/60T XD5E-30T4/60T4, XDME-30T4/60T4, XL5-32T4, XL5E-32T4, XLME-32T4, XLH-24A16/24A16L/30A32												
	Increasing mode						AB phase mode					
	HSC0	HSC2	HSC4	HSC6	HSC8	HSC10	HSC0	HSC2	HSC4	HSC6	HSC8	HSC10
Max frequency	80K	80K	80K	80K			50K	50K	50K	50K		
Quadruple frequency							2/4	2/4	2/4	2/4		
Counter interruption	√	√	√	√			√	√	√	√		
X000	U						A					
X001							B					
X002							Z					
X003		U						A				
X004								B				
X005								Z				
X006			U						A			
X007									B			
X010									Z			
X011				U						A		
X012										B		
X013										Z		

XD5-48D4T4																
	Increasing mode								AB phase mode							
	HSC0	HSC2	HSC4	HSC6	HSC8	HSC10	HSC12	HSC14	HSC0	HSC2	HSC4	HSC6	HSC8	HSC10	HSC12	HSC14
Max frequency	1M	1M	1M	1M	80K	80K	80K	80K	1M	1M	1M	1M	50K	50K	50K	50K
Quadruple frequency									2/4	2/4	2/4	2/4	2/4	2/4	2/4	2/4
Counter interruption	√	√	√	√	√	√	√	√	√	√	√	√	√	√	√	√
X0+	U+								A+							
X0-	U-								A-							
X1+									B+							
X1-									B-							
X2																
X3+		U+								A+						
X3-		U-								A-						
X4+										B+						
X4-										B-						
X5																
X6+			U+								A+					
X6-			U-								A-					
X7+											B+					
X7-											B-					
X10																
X11+				U+								A+				
X11-				U-								A-				
X12+												B+				
X12-												B-				
X13																
X14					U								A			

XD5-48D4T4																
	Increasing mode								AB phase mode							
	HSC0	HSC2	HSC4	HSC6	HSC8	HSC10	HSC12	HSC14	HSC0	HSC2	HSC4	HSC6	HSC8	HSC10	HSC12	HSC14
X15													B			
X16																
X17						U								A		
X20														B		
X21																
X22							U								A	
X23															B	
X24																
X25								U								A
X26																B
X27																

XDH-30A16/30A16L/60T4/60A32/60A64												
	Increasing mode						AB phase mode					
	HSC0	HSC2	HSC4	HSC6	HSC8	HSC10	HSC0	HSC2	HSC4	HSC6	HSC8	HSC10
Max frequency	200K	200K	200K	200K			100K	100K	100K	100K		
Quadruple frequency							2/4	2/4	2/4	2/4		
Counter interruption	√	√	√	√			√	√	√	√		
X000	U						A					
X001							B					
X002							Z					
X003		U						A				
X004								B				
X005								Z				
X006			U						A			
X007									B			
X010									Z			
X011				U						A		
X012										B		
X013										Z		

XD5-24D2T2																
	Increasing mode								AB phase mode							
	HSC0	HSC2	HSC4	HSC6	HSC8	HSC10	HSC12	HSC14	HSC0	HSC2	HSC4	HSC6	HSC8	HSC10	HSC12	HSC14
Max frequency	1M	1M	80K	80K					1M	1M	50K	50K				
Quadruple frequency									2/4	2/4	2/4	2/4				
Counter interruption	√	√	√	√					√	√	√	√				
X0+	U+								A+							
X0-	U-								A-							
X1+									B+							
X1-									B-							
X2																
X3+		U+								A+						
X3-		U-								A-						
X4+										B+						
X4-										B-						
X5																
X6			U								A					
X7											B					
X10																
X11				U								A				
X12												B				

X13																
X14																
X15																

XD5-48T6/60T6, XD5E-60T6, XL5E-64T6												
	Increasing mode						AB phase mode					
	HSC0	HSC2	HSC4	HSC6	HSC8	HSC10	HSC0	HSC2	HSC4	HSC6	HSC8	HSC10
Max frequency	80K	80K	80K	80K	80K	80K	50K	50K	50K	50K	50K	50K
Quadruple frequency							2/4	2/4	2/4	2/4	2/4	2/4
Counter interruption	√	√	√	√	√	√	√	√	√	√	√	√
X000	U						A					
X001							B					
X002							Z					
X003		U						A				
X004								B				
X005								Z				
X006			U						A			
X007									B			
X010									Z			
X011				U						A		
X012										B		
X013										Z		
X014					U						A	
X015											B	
X016											Z	
X017						U						A
X020												B
X021												Z

Note: The hardware versions of XL5E-64T6 starting with H4 support six-channel high-speed counting, while versions starting with H3 only support four-channel high-speed counting.

XD5-60T10, XDM-60T10, XD5E-60T10, XDME-60T10, XL5E-64T10, XLME-64T10												
	Increasing mode											
	HSC0	HSC2	HSC4	HSC6	HSC8	HSC10	HSC12	HSC14	HSC16	HSC18	HSC20	HSC22
Max frequency	80K	80K	80K	80K	80K	80K	80K	80K	80K	80K		
Quadruple frequency												
Counter interruption	√	√	√	√	√	√	√	√	√	√		
X000	U											
X001												
X002												
X003		U										
X004												
X005												
X006			U									
X007												
X010												
X011				U								
X012												
X013												
X014					U							
X015												
X016												
X017						U						
X020												
X021												
X022							U					
X023												

X024												
X025								U				
X026												
X027												
X030									U			
X031												
X032												
X033										U		
X034												

XD5-60T10, XDM-60T10, XD5E-60T10, XDME-60T10, XL5E-64T10, XLME-64T10												
	AB phase mode											
	HSC0	HSC2	HSC4	HSC6	HSC8	HSC10	HSC12	HSC14	HSC16	HSC18	HSC20	HSC22
Max frequency	50K	50K	50K	50K	50K	50K	50K	50K	50K	50K		
Quadruple frequency	2/4	2/4	2/4	2/4	2/4	2/4	2/4	2/4	2/4	2/4		
Counter interruption	√	√	√	√	√	√	√	√	√	√		
X000	A											
X001	B											
X002	Z											
X003		A										
X004		B										
X005		Z										
X006			A									
X007			B									
X010			Z									
X011				A								
X012				B								
X013				Z								
X014					A							
X015					B							
X016					Z							
X017						A						
X020						B						
X021						Z						
X022							A					
X023							B					
X024							Z					
X025								A				
X026								B				
X027								Z				
X030									A			
X031									B			
X032									Z			
X033										A		
X034										B		
X035										Z		

5-6 AB phase counting frequency doubling setting

For AB phase counting, the double frequency number can be set in special FLASH data registers SFD321, SFD322, SFD323... SFD330, 2 means double frequency; 4 means quadruple frequency.

Register name	Function	Setting value	Meaning
SFD320	HSC0 frequency doubling	2	2 frequency doubling
		4	4 frequency doubling
SFD321	HSC2 frequency doubling	2	2 frequency doubling
		4	4 frequency doubling
SFD322	HSC4 frequency doubling	2	2 frequency doubling
		4	4 frequency doubling
SFD323	HSC6 frequency doubling	2	2 frequency doubling
		4	4 frequency doubling
SFD324	HSC8 frequency doubling	2	2 frequency doubling
		4	4 frequency doubling
SFD325	HSC10 frequency doubling	2	2 frequency doubling
		4	4 frequency doubling
SFD326	HSC12 frequency doubling	2	2 frequency doubling
		4	4 frequency doubling
SFD327	HSC14 frequency doubling	2	2 frequency doubling
		4	4 frequency doubling
SFD328	HSC16 frequency doubling	2	2 frequency doubling
		4	4 frequency doubling
SFD329	HSC18 frequency doubling	2	2 frequency doubling
		4	4 frequency doubling

Note: After the SFD register is modified, it is necessary to restart the high-speed counter (i.e. disconnect and reboot the drive condition) in order to make the new configuration effective!

5-7 HSC instruction

This section introduces the usage of single-phase high-speed counting instruction (CNT), AB-phase high-speed counting instruction (CNT_AB), reset of high-speed counting, reading and writing of high-speed counting.

5-7-1 Single phase HSC [CNT]

1)Instruction Summary

Single phase HSC instruction

Single phase HSC [CNT]			
16 bits Instruction	-	32 bits Instruction	CNT
Execution condition	Normally ON/OFF coil	Suitable models	XD, XL(exclude XD1 / XL1)
Hardware requirement	-	Software requirement	-

2)Operands

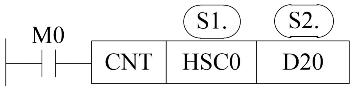
Operands	Function	Type
S	Specify HSC code (Eg. HSC0)	32 bits, BIN
D	Specify comparison value (Eg. K100, D0)	32 bits, BIN

3)Suitable Soft Components

Operands	Word soft elements											Bit soft elements						
	System								Constant	Module		System						
	D	FD	TD	CD	DX	DY	DM	DS	K/H	ID	QD	X	Y	M	S	T	C	Dn.m
S1	Only can be HSC																	
S2	●																	

*Notes: D includes D, HD; TD includes TD, HTD; CD includes CD, HCD, HSCD, HSD; DM includes DM, DHM; DS includes DS, DHS. M includes M,HM,SM； S includes S,HS； T includes T,HT； C includes C, HC.

FUNCTIONS AND ACTIONS



- When M0 is on, HSC0 counts X0 signal in single phase mode, compares the high-speed counting value with the value set in register D20. When the high-speed counting value is equal to the set value, HSC0 coil is set on immediately, and the counting value is accumulated in HSCD0 (double words).
- If countings complete and the driving condition M0 is not disconnected, HSC0 will remain ON state and continue counting, and the counting value in HSCD0 will continue to accumulate.
- If countings complete and the driving condition M0 is disconnected, HSC0 will remain on state and the counting value in HSCD0 will remain unchanged.

- During the counting process, if M0 is disconnected and connected again, the values in HSCD0 will continue to accumulate after the last counting value.
- In the counting process, if the setting value in D20 changes and the current counting value is less than the new setting value, then the new setting value is compared.
- The edge mode of single-phase high-speed counting can be set using SFD310 to SFD313 (corresponding to HSC0 to HSC6 respectively). Take HSC0 as an example, SFD310 is 0: rising edge count; 1: indicates falling edge counting; 2: indicates that both rising and falling edges count.

Note: This function is supported only by PLC firmware version V3.4.6 and later.

5-7-2 AB phase HSC [CNT_AB]

1)Instruction Summary

AB phase HSC instruction.

AB phase HSC [CNT_AB]			
16 bits Instruction	-	32 bits Instruction	CNT_AB
Execution condition	Normally ON/OFF coil	Suitable models	XD, XL(exclude XD1, XL1)
Hardware requirement	-	Software requirement	-

2)Operands

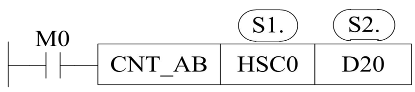
Operands	Function	Type
S	Specify HSC code (Eg. HSC0)	32 bits, BIN
D	Specify the comparison value (Eg. K100, D0)	32 bits, BIN

3)Suitable Soft Components

Operands	Word soft elements											Bit soft elements						
	System								Constant	Module		System						
	D	FD	TD	CD	DX	DY	DM	DS	K/H	ID	QD	X	Y	M	S	T	C	Dn.m
S1	Only can be HSC																	
S2	●																	

*Notes: D includes D, HD; TD includes TD, HTD; CD includes CD, HCD, HSCD, HSD; DM includes DM, DHM; DS includes DS, DHS. M includes M,HM,SM；S includes S,HS；T includes T,HT；C includes C, HC.

FUNCTIONS AND ACTIONS



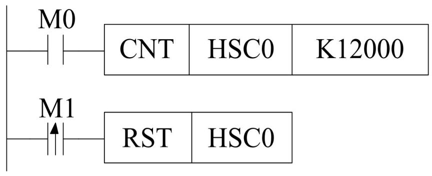
- When M0 is on, HSC0 counts X0, X1 signal in AB phase mode, compares the high-speed counting value with the value set in register D20. When the high-speed counting value is equal to the set value, HSC0 coil is set on immediately, and the counting value is accumulated in HSCD0 (double words).
- If the driving condition M0 is not disconnected, HSC0 will remain on state and continue

counting, and the counting value in HSCD0 will continue to accumulate.

- If the driving condition M0 is disconnected, HSC0 will remain on state and the counting value in HSCD0 will remain unchanged.
- During the counting process, if M0 is disconnected and connected again, the values in HSCD0 will continue to accumulate after the last counting value.
- In the counting process, if the setting value in D20 changes and the current counting value is less than the new setting value, then the new setting value is compared.

5-7-3 HSC reset [RST]

The reset mode of high-speed counter is software reset mode.



As shown above, when M0 is ON, HSC0 begins to count the pulse input of X0 port; when M1 changes fromOFF to ON, HSC0 is reset, and the count value in HSCD0 (double words) is cleared.

5-7-4 Read HSC value [DMOV]

1)Instruction Summary

Read HSC value to the specified register;

Read HSC value [DMOV]			
16 bits Instruction	-	32 bits Instruction	DMOV
Execution condition	Normally ON/OFF, rising/falling edge	Suitable models	XD, XL (exclude XD1, XL1)
Hardware requirement		Software requirement	-

2)Operands

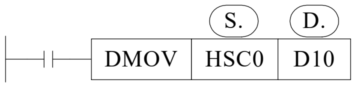
Operands	Function	Type
S	Specify HSC code	32 bits, BIN
D	Specify the read/written register	32 bits, BIN

3)Suitable Soft Components

Operands	Word soft elements											Bit soft elements						
	System								Constant	Module		System						
	D	FD	TD	CD	DX	DY	DM	DS	K/H	ID	QD	X	Y	M	S	T	C	Dn.m
S1	Only can be HSC																	
S2	●																	

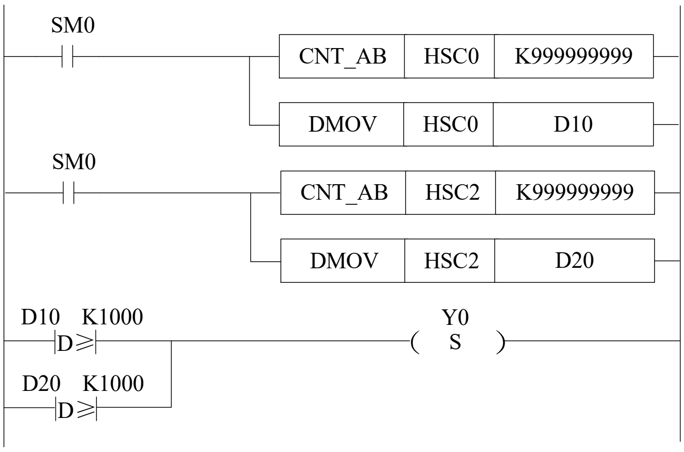
*Notes: D includes D, HD; TD includes TD, HTD; CD includes CD, HCD, HSCD, HSD; DM includes DM, DHM; DS includes DS, DHS. M includes M,HM,SM; S includes S,HS; T includes T,HT; C includes C, HC.

FUNCTIONS AND ACTIONS



- When the trigger condition is established, the high-speed count value in the accumulative register HSCD0 (double words) corresponding to HSC0 of the high-speed counter is read into the data register D10 (double words).
- High-speed counter can not directly participate in any application instructions or data comparison instructions (such as DMUL, LD > etc.) except DMOV, but can only be carried out after reading and writing into other registers.
- As high speed counter is double words counter, so it must use 32-bit instruction DMOV.
- DMOV often uses together with high speed counter.

Program example:



5-7-5 Write HSC value [DMOV]

1)Instruction Summary

Write the specified register value into HSC;

Write HSC value [DMOV]			
16 bits Instruction	-	32 bits Instruction	DMOV
Execution condition	Normally ON/OFF, rising/falling edge	Suitable models	XD, XL (exclude XD1, XL1)
Hardware requirement	-	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

Operands	Word soft elements											Bit soft elements						
	System								Constant	Module		System						
	D	FD	TD	CD	DX	DY	DM	DS	K/H	ID	QD	X	Y	M	S	T	C	Dn.m
D	Only can be HSC																	
S	●																	

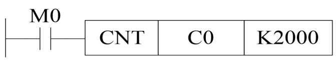
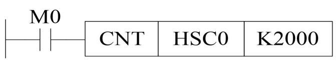
*Notes: D includes D, HD; TD includes TD, HTD; CD includes CD, HCD, HSCD, HSD; DM includes DM, DHM; DS includes DS, DHS. M includes M,HM,SM; S includes S,HS; T includes T,HT; C includes C, HC.

FUNCTIONS AND ACTIONS

- When the trigger condition is established, The value in the double-word data register D20 is written into the accumulative register HSCD0 (double-word) corresponding to the HSC0 of the high-speed counter, and the original data is replaced.
- High-speed counter can not directly participate in any application instructions or data comparison instructions (such as DMUL, LD > etc.) except DMOV, but can only be carried out after reading and writing into other registers.
- As high speed counter is double words counter, so it must use 32-bit instruction DMOV.
- DMOV often uses together with high speed counter.

5-7-6 The difference between HSC and normal counter

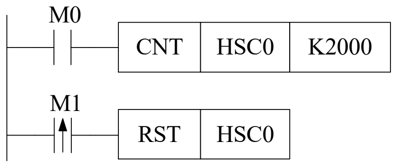
Although the instructions of high-speed counter use "CNT" in the same way as those of ordinary counter, their functions are quite different.
When M0 is changed from OFF to ON once, the value of common counter is added 1.
The high-speed counter trigger condition must be in the normally closed state when counting, which is equivalent to the high-number counter being activated, but the value of the high-number counter does not change. Only when the corresponding external signal input terminal receives the signal, the high-number counter counts. If the external signal input terminal has signal input and its trigger condition is not closed, the high-number counter will not count.
The difference is shown in the following table:

Counter type	Instruction format	Function
Normal counter		Count the OFF to ON times of M0, when the counting value reaches 2000, C0 is ON.
High-speed counter		When M0 is ON, count the X0 input signal, when the counting value reaches 2000, HSC0 is ON, M0 should be always ON when counting.

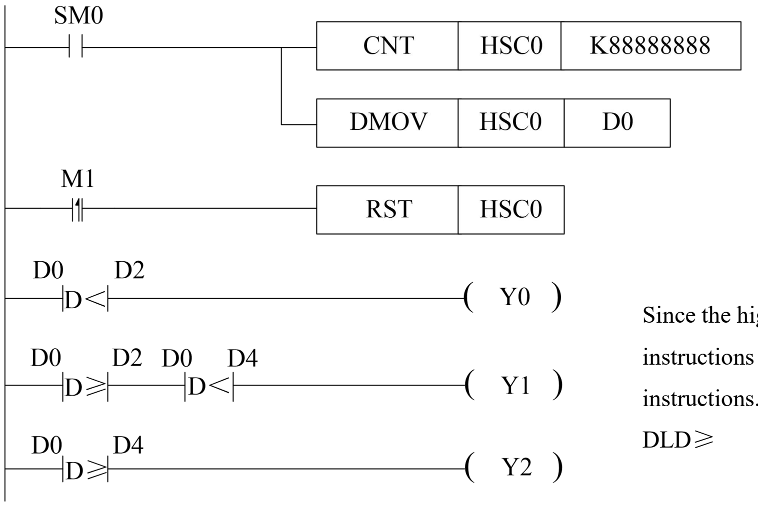
5-8 HSC Example

The following takes XD3-60 as an example to show the programming method of HSC.

Single-phase increasing mode



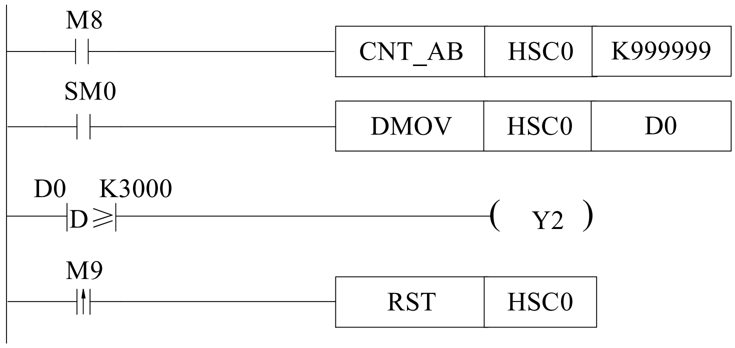
- When the M0 is ON, HSC0 counts the rising edge of the OFF to ON of the input X0 port at high speed.
- When M1 rising edge comes, reset HSC0 high-speed counter and HSCD0 (double word).



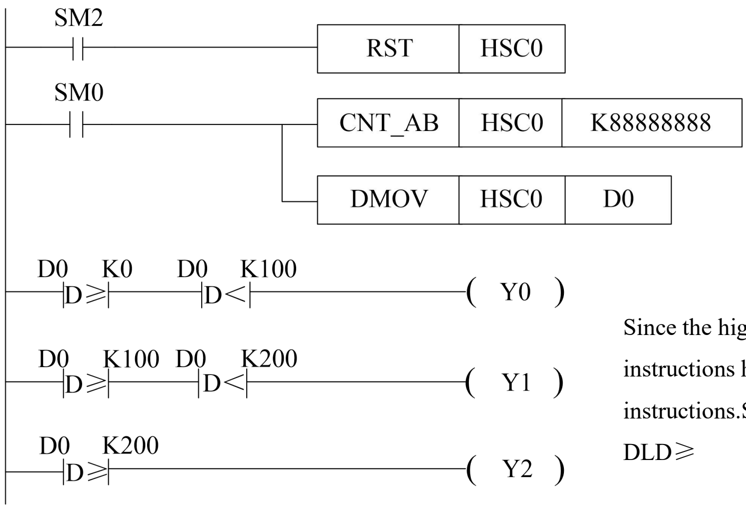
Since the high-speed count value is 32-bit, the instructions here are all 32-bit instructions.Such as DMOV，DLD<，DLD≥

- When SM0 is on, HSC0 counts X0 port in single-phase incremental mode, the setting value is K888888, and reads the high-speed counting value to D0 (double-word) in real time.
- When D0 (double words) is less than D2 (double words), Y0 is ON, when D0 (double words) is equal to or larger than D2 (double words) and less than D4 (double words), Y1 is ON. when D0 (double words) is equal to or larger than D4 (double words), Y2 is ON.
- When M1 rising edge is coming, reset HSC0 and HSCD0(double words).
- As the high speed counter is double words counter, please use double words instruction DLD < and DLD ≥.

AB phase input mode



- When M8 is ON, HSC0 starts to count. The signal inputs from X0 (A phase) and X1 (B phase).
- When SM0 is ON, the value in HSCD0 (double words) related to HSC0 is written to D0 (double words) in real-time.
- When the present counting value is over 3000, Y2 is ON.
- When the rising edge of M9 is coming, reset HSC0 and HSCD0 (double words).



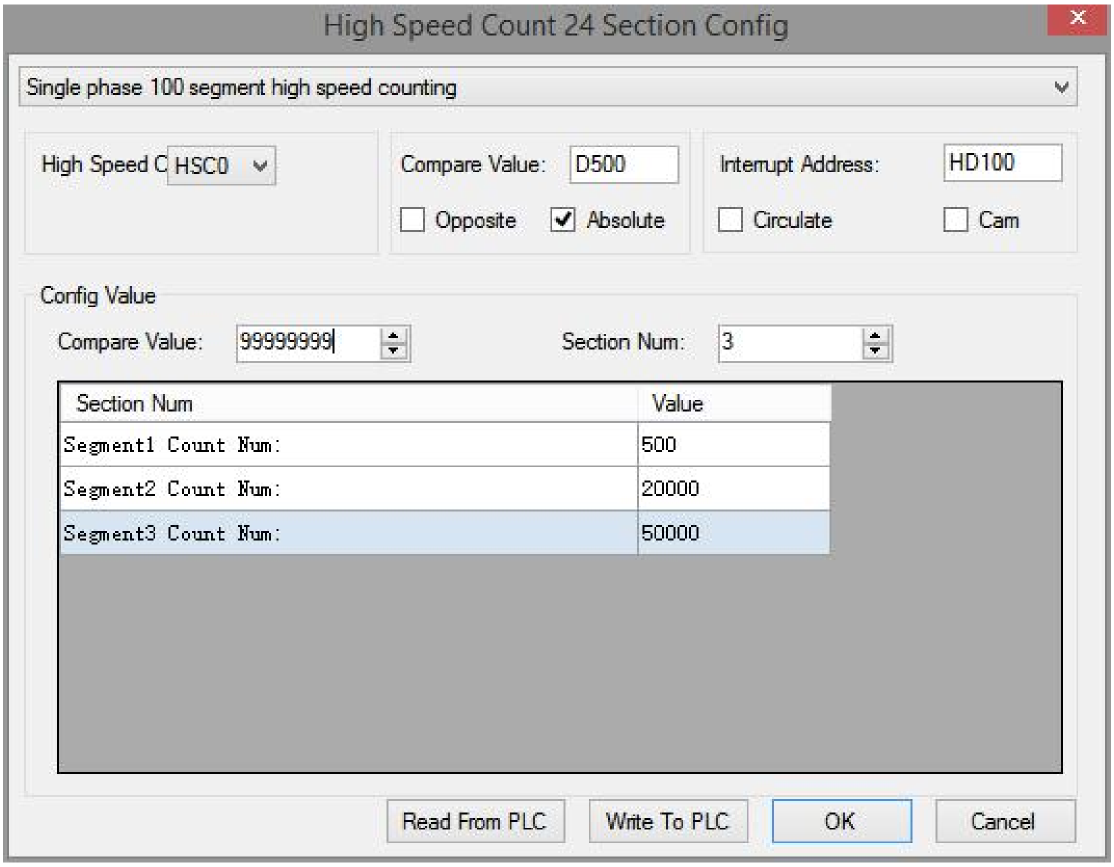
Since the high-speed count value is 32-bit, the instructions here are all 32-bit instructions.Such as DMOV，DLD<, DLD≥

- When the rising edge of the original forward pulse coil SM2 comes, that is, at the beginning of each scanning cycle, HSC0 is reset and the counting value in HSCD0 is cleared.
- When coil SM0 is on, HSC0 begins to count X0 and X1 ports in AB phase mode. The setting value of counting is K888888. At the same time, the counting value in HSCD0 (double words) is written into D0 (double words) in real time.
- When the counting value in D0 (double words) is greater than K0 and less than K100, the output coil Y0 is ON; when the counting value in D0 (double words) is greater than or equal to K100 and less than K200, the output coil Y1 is ON; and when the counting value in D0 (double words) is greater than or equal to K200, the output coil Y2 is ON.
- Since the high-speed counter is a double words counter, it is necessary to use the double words comparison instruction DLD ≥ and DLD < for comparison.

5-9 HSC interruption

5-9-1 Function overview and panel configuration

For XD/XL series PLC, some high-speed counters (referring to the high-speed counting input port allocation table of chapter 5-5 of each type of PLC) have a set value of 32 bits in 1-100 sections. When the difference of high-speed counting equals to the set value of corresponding 100 sections, the interruption will occur according to the corresponding interruption mark.
If the set value of N segment is set, there must be interrupt mark and interrupt program corresponding to N segment. The interruption marks corresponding to each high-speed counter are shown in chapter 5-9-4.
When using high-speed counting interrupt function, instructions can be written directly (see chapters 5-9-2 and 5-9-3), or can be configured by software panel. Please click  in the XDPPro software, it will show below window.

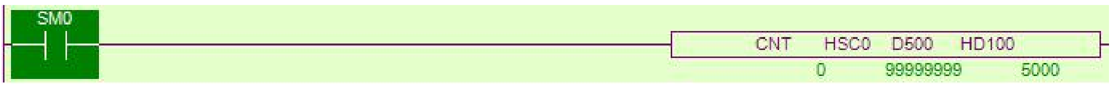


In this panel, we can configure the parameters related to high speed count interruption. Take the settings in above figure as an example to explain each parameter function.

Parameter		Function
	single phase 100 segments high speed counting	High Speed Counting in Single Phase Incremental Mode
	100 segments AB phase high speed counting	High Speed Counting in AB phase mode
	HSC0~HSC18(32-	High-speed counter number

	bit)	corresponding to high-speed input port
Compare Value: D500	Free to specify	HSC0 is ON when the count value is equal to the value in the register.
Compare Value: 99999999	Free to specify	When it counts to the comparison value, HSC0 is ON, the comparison value can be set here or put in compare reigster D500.
☐ Opposite ☑ Absolute	Relative	It will produce the interruption of segment N when the counting value = segment N-1 interruption counting value + segment N setting value.
	Absolute	It will produce the interruption when the counting value is equal to setting value.
Interrupt Address: HD100	Free to specify	The set values of 100 segments of high-speed counting interrupts are stored in the registers starting from HD100, and the set values are stored in the double-word registers HD100, HD102, HD104….
☐ Circulate ☐ Cam	Interruption cycle	It must be used in relative mode. When all interrupts are over, high-speed counting interrupts can still be generated circularly.
	CAM	It must be used in absolute mode. When the counting value equals any set value, interruption occurs.
Section Num: 3	1~100 optional	If set to 3, it means execute three high-speed counting interrupts
Value	Free to specify	Each segment corresponds to an interrupt count value, which is written to the address block starting from HD100; the interrupt time is determined by the relative/absolute count mode

For detailed usage of the above parameters, please see the following chapters.
After writing to the PLC and clicking "OK", the high-speed count interrupt instruction configuration is completed, as shown in the following figure:



5-9-2 Single phase 100-segment HSC [CNT]

1)Summarization

Single phase 100-segment HSC instruction.

Single phase 100-segment HSC [CNT]			
16-bit instruction	-	32-bit instruction	CNT
Execution condition	Normal ON/OFF	Suitable model	XD, XL (exclude XL1, XD1)
Hardware requirements	-	Software requirements	-

2)Operands

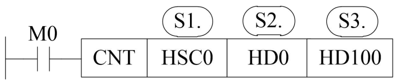
Operands	Function	Type
S1	Set the HSC (for example: HSC0)	32 bits, BIN
S2	Set the compare value (eg. K100, D0)	32 bits, BIN
S3	Set the 100-segment setting value	32 bits, BIN

3)Suitable soft components

Operands	Word soft elements											Bit soft elements						
	System								Constant	Module		System						
	D	FD	TD	CD	DX	DY	DM	DS	K/H	ID	QD	X	Y	M	S	T	C	Dn.m
S1	Only can be HSC																	
S2	●								●									
S3	●																	

*Notes: D includes D, HD; TD includes TD, HTD; CD includes CD, HCD, HSCD, HSD; DM includes DM, DHM; DS includes DS, DHS. M includes M,HM,SM; S includes S,HS; T includes T,HT; C includes C, HC.

Description



- When the high-speed counter HSC0 counts in single-phase mode, high-speed counting value is compared to data block starting from HD100 (such as HD102, HD102, HD104 and other double-word registers), it will immediately produce the corresponding high-speed counting interrupt when the condition is met, each section of the corresponding interrupt marks please refer to chapter 5-9-4.
- During the high-speed counting process, it is invalid to modify the set value of 100 segments.
- In the process of high-speed counting, the driving condition M0 can not be disconnected. If M0 is disconnected and then rebooted, no interruption will occur. The high-speed counter must be reset first, and thenset ON M0 again to produce interruption.
- When the interrupt is finished in a single execution, if it needs to start the interruption again, the high-speed counter must be reset first, and then the driving condition must be ON again.

- In interrupt loop mode, interrupts can be generated in sequence as long as M0 remains on state.

5-9-3 AB phase 100-segment HSC[CNT_AB]

1)Summarization

AB phase 100-segment HSC instruction.

AB phase 100-segment HSC [CNT_AB]			
16 bits instruction	-	32 bits instruction	CNT_AB
Execution condition	Normal ON/OFF	Suitable model	XD, XL (exclude XL1, XD1)
Hardware requirements	-	Software requirements	-

2)Operands

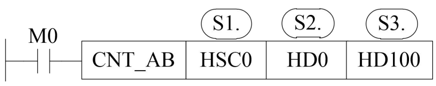
Operands	Function	Type
S1	Set the HSC (such as:HSC0)	32 bits, BIN
S2	Set the compare value (such as: K100, D0)	32 bits, BIN
S3	Set the 100-segment setting value	

3)Suitable soft components

Operands	Word soft elements											Bit soft elements						
	System								Constant	Module		System						
	D	FD	TD	CD	DX	DY	DM	DS	K/H	ID	QD	X	Y	M	S	T	C	Dn.m
S1	Only can be HSC																	
S2	●								●									
S3	●																	

*Notes: D includes D, HD; TD includes TD, HTD; CD includes CD, HCD, HSCD, HSD; DM includes DM, DHM; DS includes DS, DHS. M includes M,HM,SM; S includes S,HS; T includes T,HT; C includes C, HC.

Description



- When the high-speed counter HSC0 counts in AB phase mode, high-speed counting value is compared to data block starting from HD100 (such as HD102, HD102, HD104 and other double-word registers), it will immediately produce the corresponding high-speed counting interrupt when the condition is met, each section of the corresponding interrupt marks please refer to chapter 5-9-4.
- During the high-speed counting process, it is invalid to modify the set value of 100 segments.
- In the process of high-speed counting, the driving condition M0 can not be disconnected. If M0 is disconnected and then rebooted, no interruption will occur.

The high-speed counter must be reset first, and thenset ON M0 again to produce interruption.

- When the interrupt is finished in a single execution, if it needs to start the interruption again, the high-speed counter must be reset first, and then the driving condition must be ON again.
- In interrupt loop mode, interrupts can be generated in sequence as long as M0 remains on state.

5-9-4 Interruption flag of HSC

The 100 segments interruption flags of each HSC are in the following table. For example, the 100 segments interruption flags of HSC0 are I2000, I2001, I2002….. I2099.

HSC	Interruption flag					
	Segment 1	Segment 2	Segment 3	……	Segment N	Segment 100
HSC0	I2000	I2001	I2002	……	I(2000+N-1)	I2099
HSC2	I2100	I2101	I2102	……	I(2100+N-1)	I2199
HSC4	I2200	I2201	I2202	……	I(2200+N-1)	I2299
HSC6	I2300	I2301	I2302	……	I(2300+N-1)	I2399
HSC8	I2400	I2401	I2402	……	I(2400+N-1)	I2499
HSC10	I2500	I2501	I2502	……	I(2500+N-1)	I2599
HSC12	I2600	I2601	I2602	……	I(2600+N-1)	I2699
HSC14	I2700	I2701	I2702	……	I(2700+N-1)	I2799
HSC16	I2800	I2801	I2802	……	I(2800+N-1)	I2899
HSC18	I2900	I2901	I2902	……	I(2900+N-1)	I2999

5-9-5 Setting value meaning in absolute or relative mode

The setting value meaning is different in absolute and relative mode. Relative/absolute mode can be set in the software panel. It can also be modified by special Flash register SFD330. (**Note:** Driving conditions must be OFF and ON again to make the configuration effective.)

0: Relative mode;

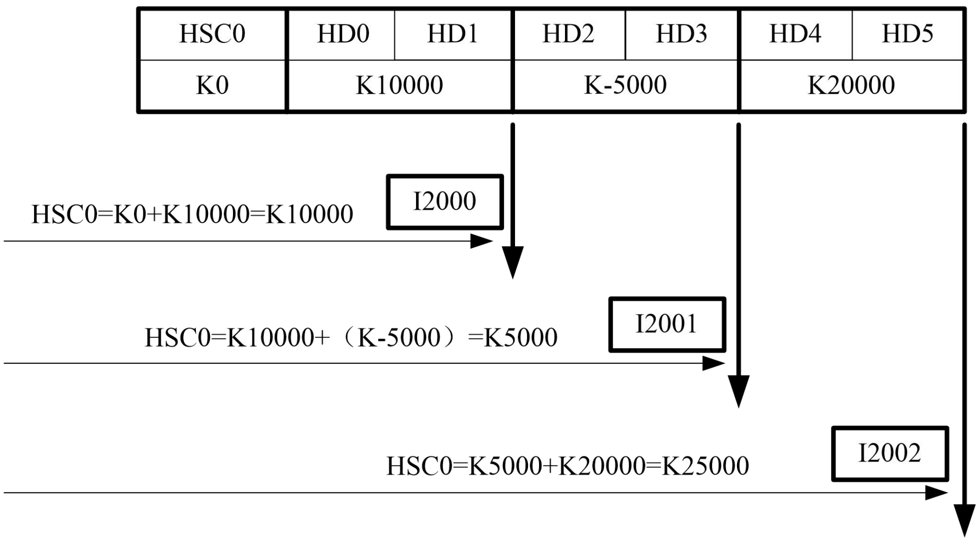
1: Absolute mode.

- **Relative mode**

In relative mode, the set value of high-speed counting 100 segments is relative cumulative value. When the set value of counting equals the sum of the interruption count value of N-1 segment and the set value of N segment, the segment N interrupt is generated.
N interrupt markers correspond to N interrupt settings. The N+1 interrupt settings register is reserved for other purposes.

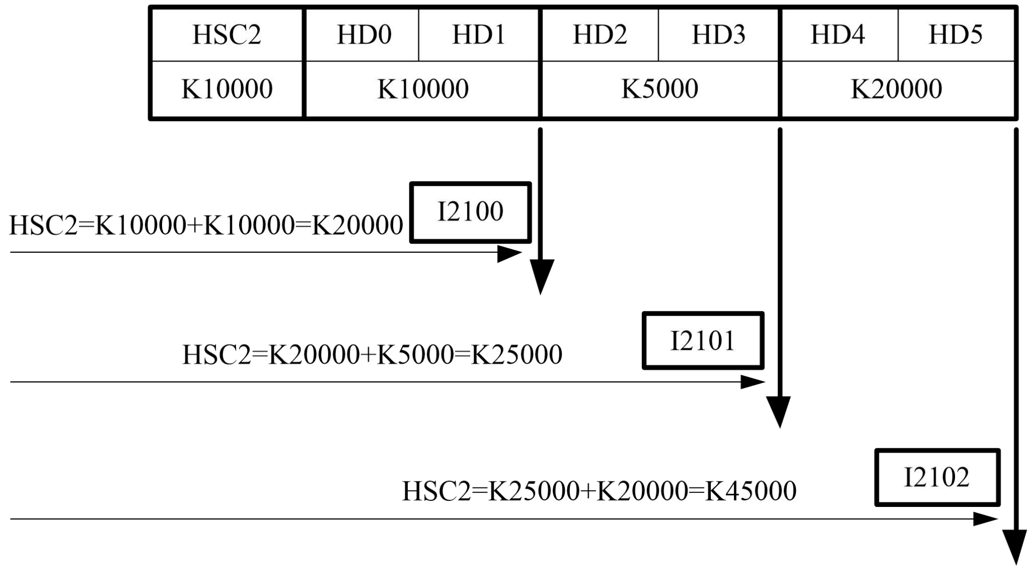
Example1:
The current value of HSC0 is 0, segment one preset value is 10000, the preset value in segment 2 is －5000, the preset value in segment 3 is 20000. When starting to count, when the counter's current value is 10000, it generates the segment 1 interruption I2000; when the counter's current value is 5000, it generates the segment 2 interruption I2001; when the counter's current value is 25000, it generates the segment 3 interruption I2002.

See graph below:



Example 2:
HSC2 current value is 10000, the segment one preset value is 10000, the preset value of segment 2 is 5000, the preset value of segment 3 is 20000. When starting to count, when the counter's current value is 20000, it generates the segment 1 interruption I2100; when the counter's current value is 25000, it generates the segment 2 interruption I2101; when the counter's current value is 45000, it generates the segment 3 interruption I2102.

See graph below:

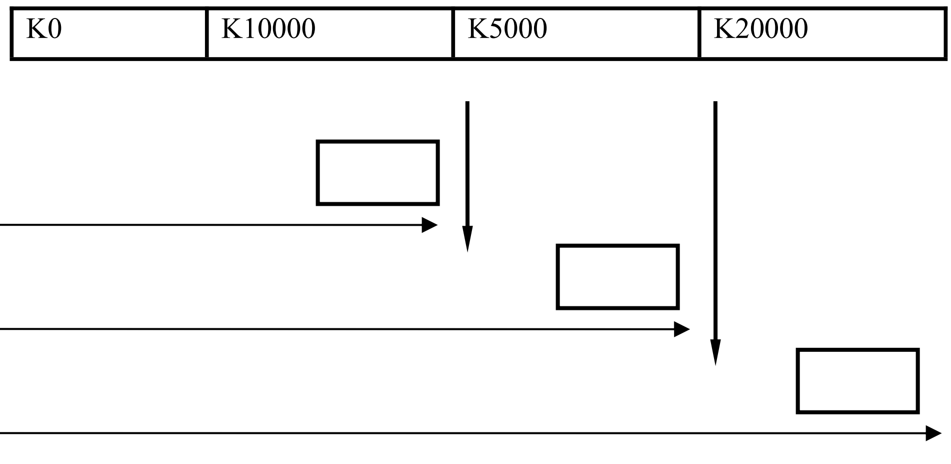


- **Absolute Mode**

In absolute mode, interruption occurs when the count value equals the set value of each section of the counter. N interrupt markers correspond to N interrupt settings. The N+1 interrupt settings register is reserved for other purposes.

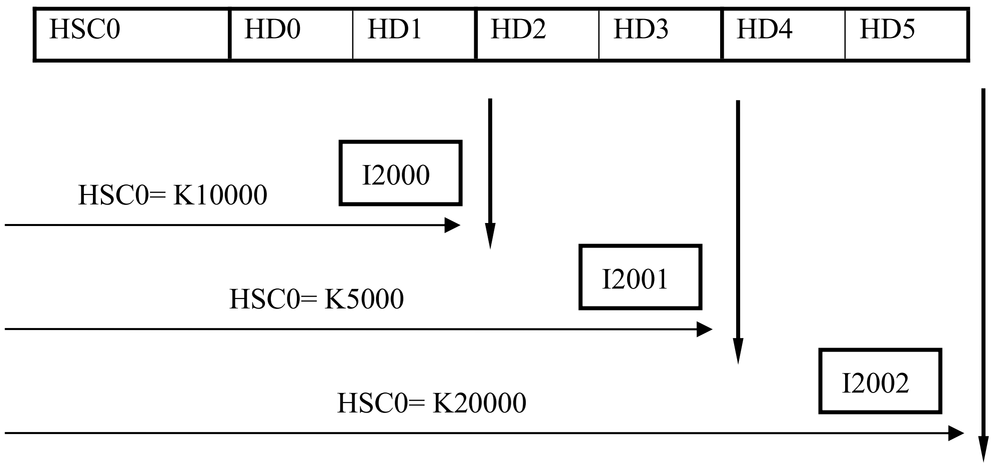
Example 1:

The current value of counter HSC0 is 0, the setting value of segment 1 is 10000, the setting value of segment 2 is 15000, and the setting value of segment 3 is 20000. When it starts counting, if the current value of the counter is 10000, the segment 1 interruption I2000 is generated; when the current value of the counter is 15000, the segment 2 interruption I2001 is generated; when the current value of the counter equals 20000, the segment 3 interruption I2002 is generated.



Example 2:

The current value of counter HSC2 is 5000, segment 1 set value is 10000, segment 2 set value is 5000, and segment 3 set value is 20000. When it starts counting, if the current value of the counter is 10000, segment 1 interrupt I2100 is generated; when the current value of the counter is 5000, segment 2 interrupt I2101 is generated; when the current value of the counter equals 20000, segment 3 interrupt I2102 is generated.



Note: When absolute counting is performed in non-cam mode, counting interrupts are generated sequentially, i.e.,segment 1 interruption, segment 2 interruption, segment 3 interruption... When a segment interrupt occurs, no interrupt occurs even if the count value reaches the set value of the segment again.

As in the example above, if the count value is increased from 4000 to 5000 and 10000 after the interruption of segment 1 and 2, the interruption of segment 1 and 2 will not occur again, and the interruption of segment 3 will occur when the count value continues to increase to 20000.

5-9-6 HSC interruption cycle mode

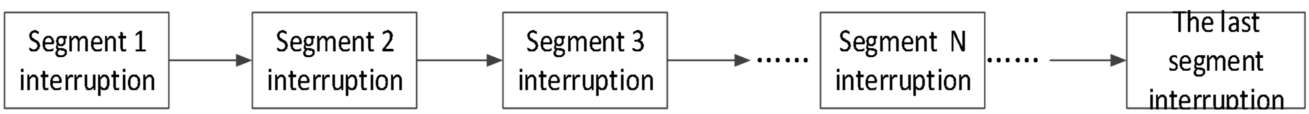
Mode 1: Single loop (normal mode)

The HSC interruption will not happen after it ends. The following conditions can start the interruption again.

(1) reset the HSC

(2) Reboot the HSC activate condition

The interruption is generated as the following sequence when single loop execution:



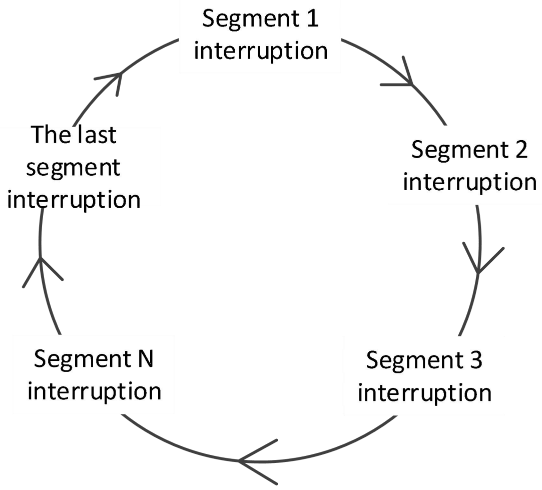
Mode 2: Continuous loop

Continous loop interruption is only suitable for relative counting mode. In continuous loop mode, the interruption will start again after it is completed. This mode is especially suitable for the following application:

(1) continuous back-forth movement.

(2) Generate cycle interruption according to the fixed pulse.

When continuous loop interruption is performed (without cam function enabled), interrupts occur in the following order:



Via setting SFD331, users can switch betweensingle loop mode or continuous loop mode. The detailed assignment is show below:
(Note: the settings will be effective after setting OFF and ON the driving condition again)

Address	HSC	Setting
Bit0	100 segments HSC interruption cycle (HSC0)	0: single loop 1: continuous loop
Bit1	100 segments HSC interruption cycle (HSC2)	
Bit2	100 segments HSC interruption cycle (HSC4)	
Bit3	100 segments HSC interruption cycle (HSC6)	
Bit4	100 segments HSC interruption cycle (HSC8)	
Bit5	100 segments HSC interruption cycle (HSC10)	
Bit6	100 segments HSC interruption cycle (HSC12)	
Bit7	100 segments HSC interruption cycle (HSC14)	
Bit8	100 segments HSC interruption cycle (HSC16)	
Bit9	100 segments HSC interruption cycle (HSC18)	

5-9-7 CAM function of high speed counter interruption

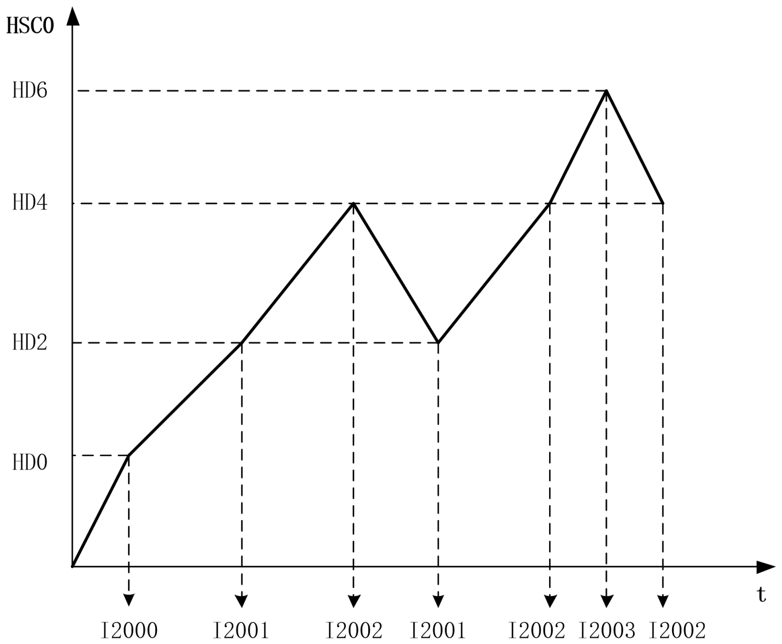
High-speed counting cam: After setting all interruption set value, the high-speed counting cam function is selected. When the high-speed counting value is equal to any of the interruption set value, the corresponding high-speed counting interruption (the same as the 100-segment high-speed counting interruption marker) is executed immediately. When the high-speed counting value changes repeatedly, the same high-speed interruption of the cam can be executed repeatedly.

High-speed counting cam not only can fully realize the cyclic sequence interruption function of ordinary electronic cam, but also can generate multiple times of positive and negative single point interruption in single cycle. It is widely used in control systems of high-speed winding machine and packaging machine.

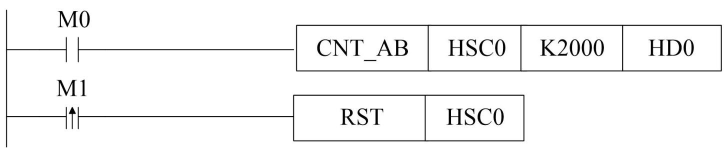
Note: CAM function is only fit for absolute counting mode.

Cam function can be set by configuration panel in XINJE PLC software, or by special Flash register SFD332: (Note: Drive condition must be set OFF and ON again to make configuration effective)
0: No cam function enabled
1: Enable Cam Function

Example:
Four values are stored in four consecutive double-word registers starting with register HD0. When HSC0 starts to count, if the HSC0 count value equals any of the four registers, the corresponding interrupt signal will be generated immediately. As shown in the following figure:



5-9-8 Interruption using notes and parameter address




```
LD  M0                        //HSC trigger condition M0 (also interruption counting condition)
CNT_AB  HSC0  K2000  HD0      //HSC and 100-segment head address setting
LDP       M1                  //HSC reset trigger condition
RST       HSC0                //HSC and 100-segment reset (also reset the interruption)
```

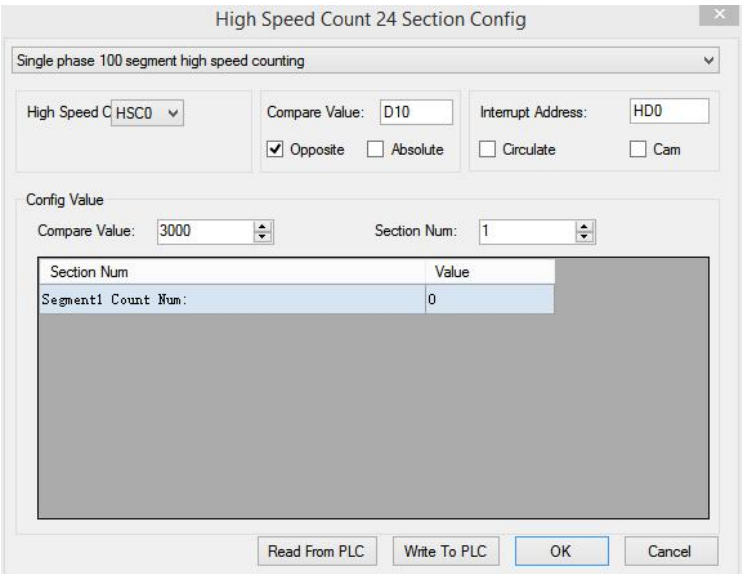
As shown in the above example (note: the interrupt sub-program is omitted, see the application example in chapter 5-9-9). The data register HD0 sets the region starting address for the set value of 100 segments, and then stores the set value of 100 segments in double-word form. Attention should be paid to using high-speed counting interrupts:

- The register after the last segment no needs to set 0, but should be reserved and cannot be used for other purpose. For example, it has 3 segments, segment 1 is HD0, segment 2 is HD2, segment 3 is HD4, then HD6 is reserved.
- It is not allowed to set the interrupt setting value without writing the interrupt program. Otherwise, errors will occur.
- 100-segment interrupt of high speed counter generate in turn, that is, if the first interrupt does not occur, the second interrupt will not occur.
- In high speed counting process, if the present counting value is changed by DMOV, ADD instruction (DMOV K1000 HSCD0), the interruption value will not change at this time. Please do not change the HSCD value when the high speed counter is running.

Some parameters can be modified in special Flash registers, as shown in the following table:

Parameter	Register address	Setting value
Counting mode	SFD330	0: relative 1: absolute
Execution mode	SFD331	0: execution once 1: interruption cycle
CAM function	SFD332	0: not enable 1: enable cam function

The above parameters can also be configured by the configuration panel in the following way: Move the mouse over the high-speed counting instruction and right-click it. Select "CNT_AB Instruction Parameter Configuration" from the drop-down menu. A configuration panel will appear to configure the parameters in this window. As shown in the following figure:

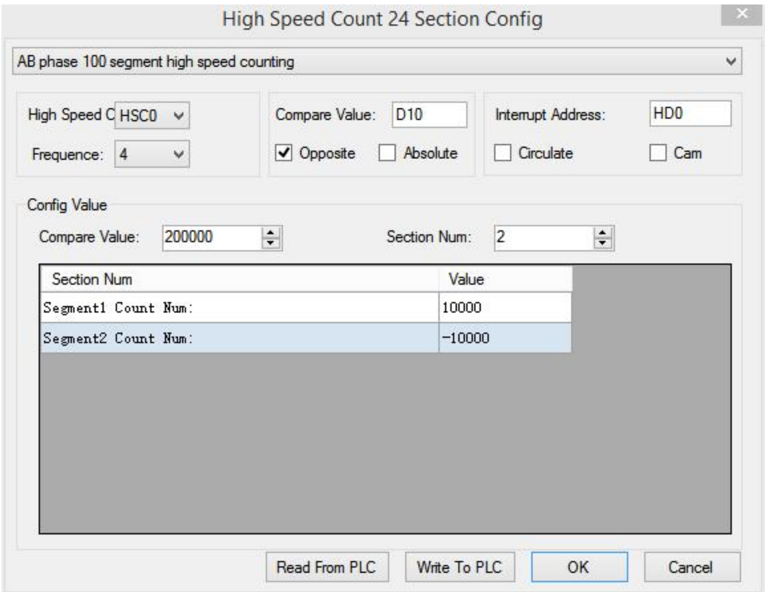


5-9-9 Application of HSC interruption

Application 1:

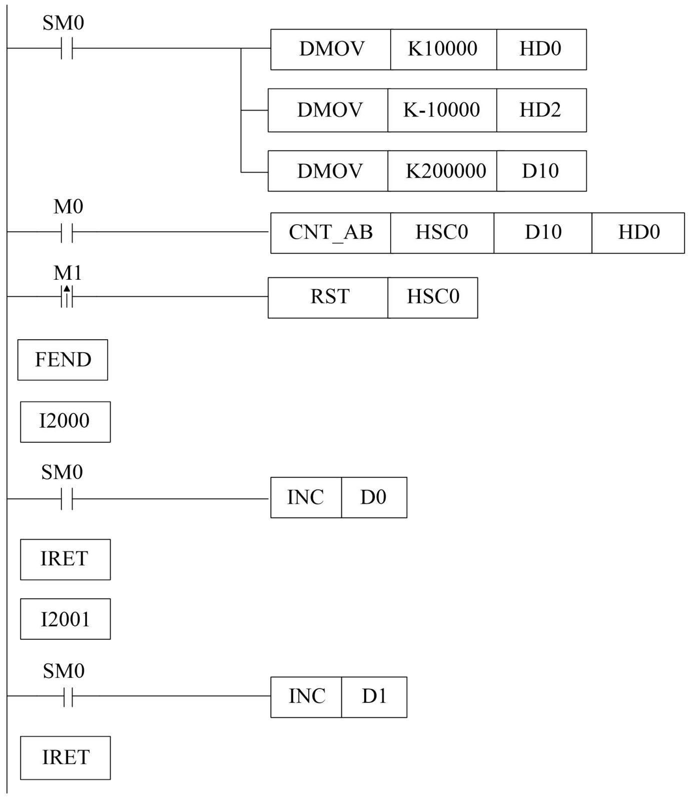
When M0 is ON, HSC0 starts counting. The counting value is stored in the address starting from HD0. When it reaches the set value, the interruption is produced. When the rising edge of M1 is coming, clear the HSC0.

Method 1: Configure the parameters through XDPpro software:



Configure item	Function
High speed counter	Choose HSC, the range is from HSC0 to HSC18
Frequency	Choose the HSC frequency doubling (2 or 4)
Compare value	The value can be register or constant, in this example, when the counting value reaches compare value, HSC0 is ON. here the compare value is 200000 which is saved in D10.
Relative and absolute	The HSC is relative mode or absolute mode
Interrupt address	The starting registers to store 100 segments interruption preset value
Circulate	100 segments interruption mode is cycle or not
Cam	The cam function is executed when any set value of 100-segment high speed counting interruption equals the counting value.

Method 2: make the program

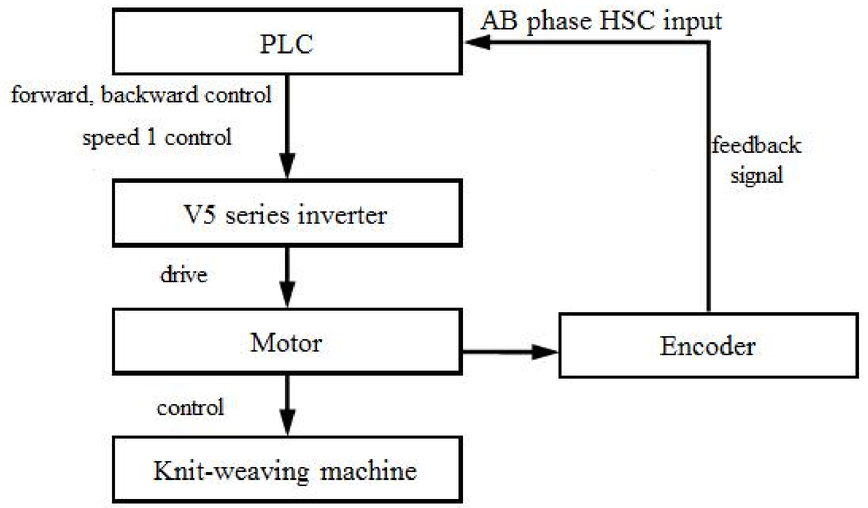


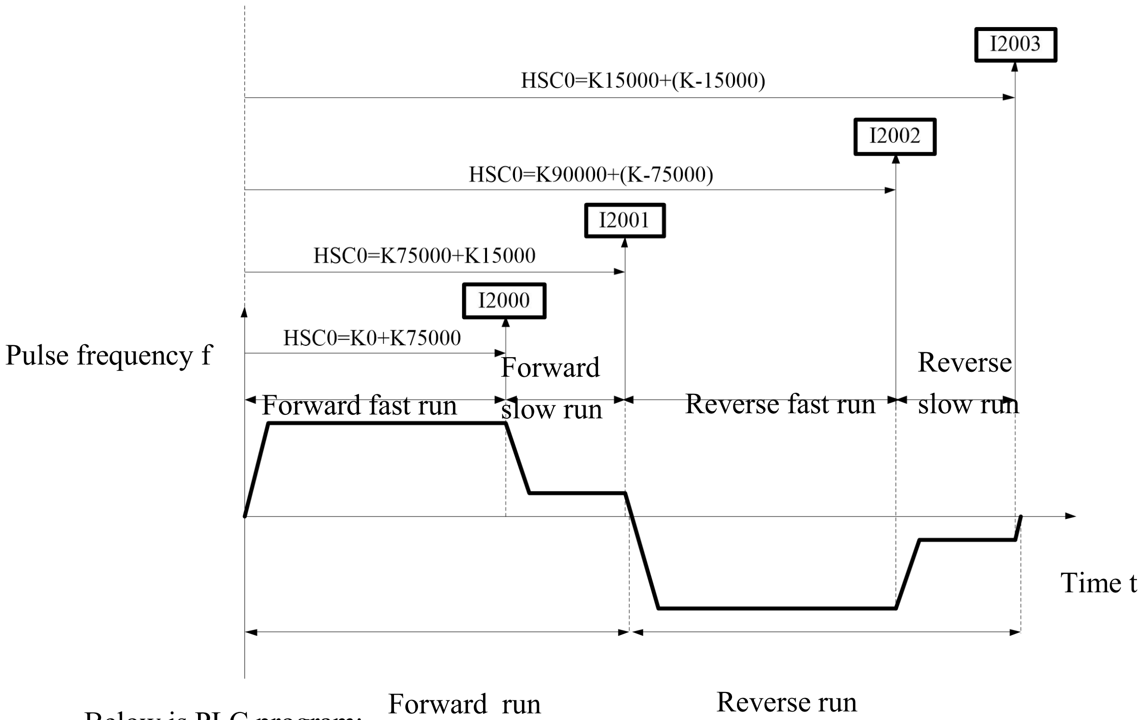
Instruction:

```
LD      SM0                 //SM0 is normally ON coil
DMOV K10000HD0              //segment one preset value HD0 is 10000
DMOV K-10000 HD2            //segment 2 preset value HD2 is -10000
DMOV K200000   D10          //set HSC compare value
LD      M0                  //HSC activate condition M0
CNT_AB  HSC0 D10 HD0        //HSC interruption instruction
LDP     M1                  //HSC reset condition M1
RST     HSC0                //reset HSC and 100 segments interruption
FEND                        //the main program end
I2000                       //segment one interruption flag
LD      SM0                 //SM0 is normally ON coil
INC     D0                  //D0= D0+1
IRET                        //interruption return flag
I2001                       //segment 2 interruption flag
LD      SM0                 //SM0 is normally ON coil
INC     D1                  //D1= D1+1
IRET                        //interruption return flag
```

Application 2: knit-weaving machine (continuous loop mode)

The machine principle: Control the inverter via PLC, thereby control the motor. Meantime, via the feedback signal from encoder, control the knit-weaving machine and the precise position.





Below is PLC program:

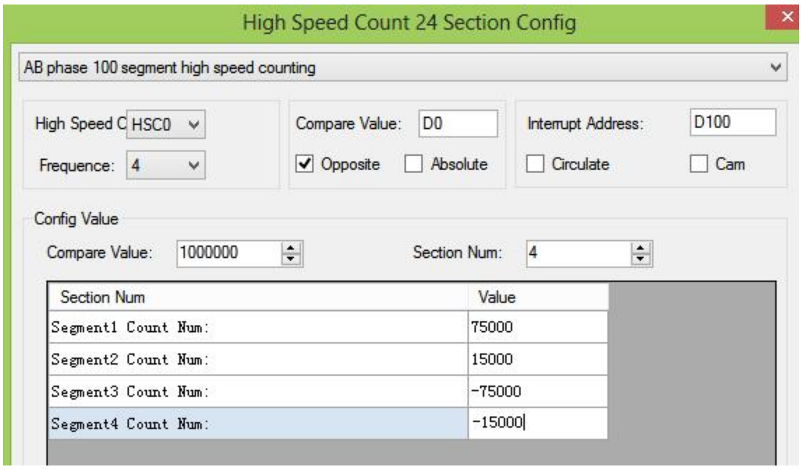
Y2 represents forward output signal;

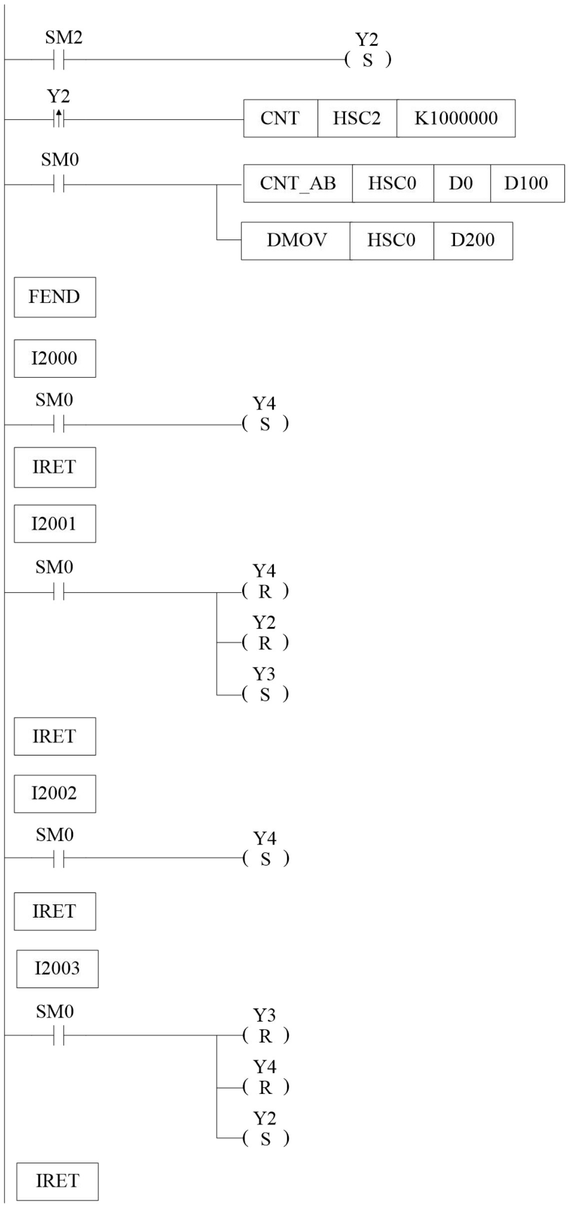
Y3 represents reverse output signal;

Y4 represents output signal of speed 1;

HSC2: Back-forth times accumulation counter;

HSC0: AB phase HSC;





Instruction List:

LD SM2 //SM2 is initial ON coil

```
SET     Y2                          //set ON Y2 (forward run)
LDP      Y2                        // Back-forth times activate condition Y2
CNT  HSC2  K1000000            //HSC2 starts counting
LD       SM0                        //SM000 is normal ON coil
CNT_AB  HSC0  D0  D100        //HSC 100 segments first address
DMOV  HSC0  D200              //read HSC0 counting value to D200
FEND                             //main program end
I2000                            //Interruption 1 flag
LD       SM0                          //SM0 is normal ON coil
SET     Y4                        //set ON Y4 (run at speed 1)
IRET                             //interruption return
I2001                            //interruption 2 flag
LD       SM0                          //SM0 is normal ON coil
RST     Y4                          //reset Y4 (stop running at speed 1)
RST     Y2                          //reset Y2 (stop forward running)
SET     Y3                          //set ON Y3 (reverse running)
IRET                             //interruption return
I2002                            //interruption 3 flag
LD       SM0                          //SM0 is normal ON coil
SET     Y4                          //set ON Y4 (run at speed 1)
IRET                             //interruption return
I2003                            //interruption 4 flag
LD       SM0                          //SM0 is normal ON coil
RST     Y3                          //reset Y3 (stop reverse running)
RST     Y4                          //reset Y4 (stop running at slow speed)
SET     Y2                          //set on Y2 (forward running)
IRET                             //interruption return
```

6 Communication Function

This chapter mainly includes: basic concept of communication, Modbus communication and free communication.

Relative Instruction

Mnemonic	Function	Circuit and soft components	Chapter
MODBUS Communication			
COLR	Coil Read	COLR S1 S2 S3 D1 D2	6-2-3
INPR	Input coil read	INPR S1 S2 S3 D1 D2	6-2-3
COLW	Single coil write	COLW D1 D2 S1 S2	6-2-3
MCLW	Multi-coil write	MCLW D1 D2 D3 S1 S2	6-2-3
REGR	Register read	REGR S1 S2 S3 D1 D2	6-2-3
INRR	Input register read	INRR S1 S2 S3 D1 D2	6-2-3
REGW	Single register write	REGW D1 D2 S1 S2	6-2-3
MRGW	Multi-register write	MRGW D1 D2 D3 S1 S2	6-2-3
Free Communication			
SEND	Send data	SEND D10 D100 K2	6-3-4
RCV	Receive data	RCV D20 D200 K2	6-3-4
Read and write serial port data			
CFGCR	Read serial port	CFGCR HD0 K7 K2	6-5-1
CFGCW	Write serial port	CFGCW HD0 K8 K2	6-5-2

6-1 Summary

The XD/XL series programmable controller body (firmware version V3.2 and above, upper computer software V3.2.2 and above) provides multiple communication methods to meet various communication and network needs of users. It not only supports Modbus RTU and Modbus ASCII, but also supports free-form communication and fieldbus X-NET. Therefore, XD/XL series PLCs can communicate with various communication protocol devices, such as printers, instruments, etc.

6-1-1 COM port

COM Port

XD, XL series PLC have multiple communication ports, such as USB port, Ethernet port, COM0~COM5, COM2-RS232, COM2-RS485.

×not support √support

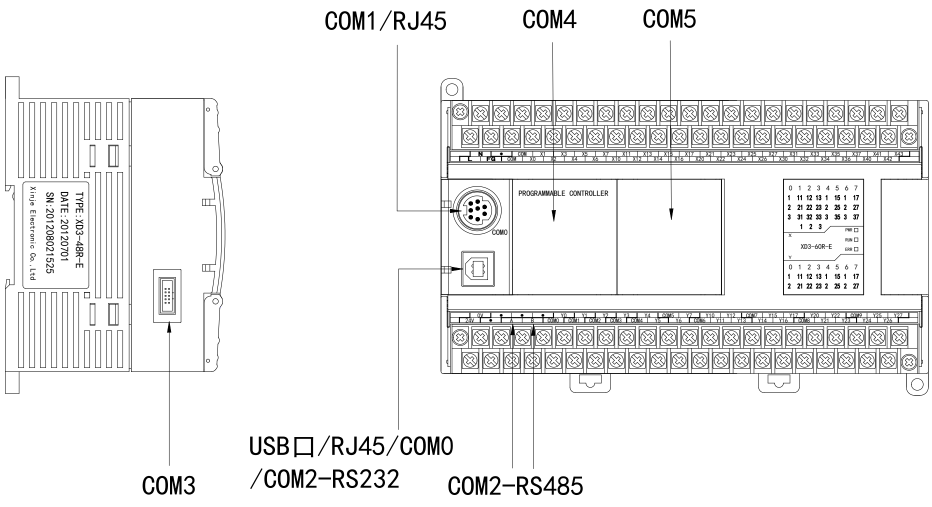
	USB	RJ45	COM0	COM1	COM2-RS232	COM2-RS485	COM3	COM4	COM5
XD1	×	×	√	√	×	√	×	×	×
XD2	×	×	√	√	×	√	√	√	√
XD3	√	×	×	√	×	√	√	√	√
XD5	√	×	×	√	×	√	√	√	√
XDM	√	×	×	√	×	√	√	√	√
XDC	×	×	×	√	√	√	√	√	√
XD3E	×	√	×	√	×	√	√	√	×
XD5E	×	√	×	√	×	√	√	√	√
XDME	×	√	×	√	×	√	√	√	√
XDH	×	√	×	√	×	√	√	×	×
XL1	×※1	×	√	√	×	√	×	×	×
XL3	√	×	×	√	×	√	√	×	×
XL5	√	×	×	√	×	√	√	×	×
XL5E	×	√	×	√	×	√	√	×	×
XL5N	×	√	×	√	×	√	√	×	×
XLME	×	√	×	√	×	√	√	×	×
XLH	×	√	×	√	×	√	√	×	×
XL5H	×	√	×	√	×	√	√	×	×

Note:

※1: XL1-16T-U has USB port.

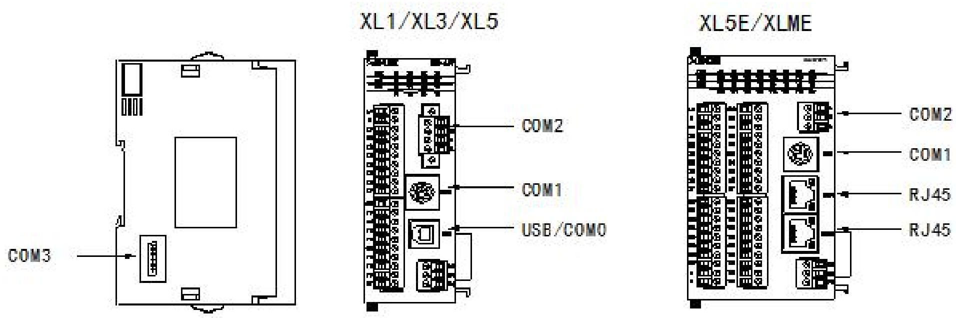
※2: In the series of "√" PLCs, there may be some models that do not support COM2-COM5. See Appendix 5 for details.

The distribution of XD series communication ports is as follows:

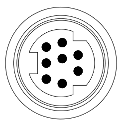


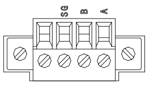
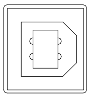
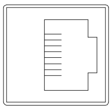
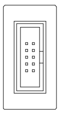
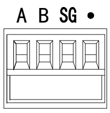
Note: The left side of output terminal block of XD5E/XDME/XDH is RS232 port.

The distribution of XL series communication ports is as follows:



The definitions and functions of each communication port are as follows:

Port	Appearance	Definition	protocol	Function
COM0		RS232 port	X-NET Modbus	Download program, set the port parameters through software or xinje config tool
COM1		RS232 port	Modbus RTU Modbus ASCII Free communication X-NET	Download program and connect external devices, set the port parameters through software or xinje config tool
COM2-RS232		RS232 port	Modbus RTU Modbus ASCII Free communication X-NET	Download program and connect external devices, set the port parameters through software or xinje config tool
COM2-RS485	A, B port	RS485 port	Modbus RTU Modbus ASCII Free	Download program and connect external devices, set the port parameters
COM2	A, B port	RS485 port		

			communication X-NET	through software or xinje config tool
USB		USB port	X-NET	High speed download port, please install the USB driver first
RJ45		Ethernet port	TCP/IP communication based on Ethernet	High speed stable download/upload program and data, remote monitoring, communicate with TCP IP device in LAN, set the port parameters through software or xinje config tool. Only XDH series LAN2 port supports EtherCAT, can synchronous control of 32-axis motor.
COM3		Left extension ED port (for extending RS232/RS485 port)	Modbus RTU Modbus ASCII Free communication X-NET	connect external devices, set the port parameters through software or xinje config tool
COM4		Above extension BD port/ RS232/RS485/Optical fiber port (see below details)	Modbus RTU Modbus ASCII Free communication X-NET	connect external devices, set the port parameters through software or xinje config tool
COM5				

Note:

(1) COM0 port is X-NET communication mode by default; COM1 of XDC is X-NET communication mode by default.

(2) The 232 port near the output terminal of XD1 and XD2 series is COM0 port, which does not conflict with the 485 port on the terminal block.

(3) COM2-RS232 and COM2-RS485 of XDC series cannot be used simultaneously; when configured in programming software, the port number is COM2.

(4) If COM1 cannot communicate with PC after changing the parameters, please click [stop PLC when reboot] in the software and then power on again to solve the problem; if unnecessary, it is better not to modify COM1 communication parameters.

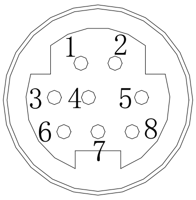
(5) LAN1 port supports Ethernet communication, LAN2 port supports EtherCAT bus function.

(6) X-NET communication function is not within the scope of this manual, please refer to the X-NET user manual.

(7) Ethernet communication content is not within the scope of this manual, please refer to the user manual of TCP IP communication based on Ethernet.

(8) The Ethernet bus is not within the scope of this manual. Please refer to the user manual of EtherCAT motion control.

1. **RS232 port (COM0, COM1, COM2-RS232)**

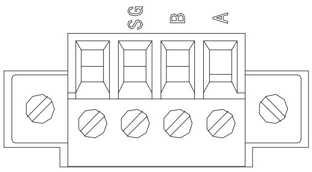


4：RxD
5：TxD
8：GND

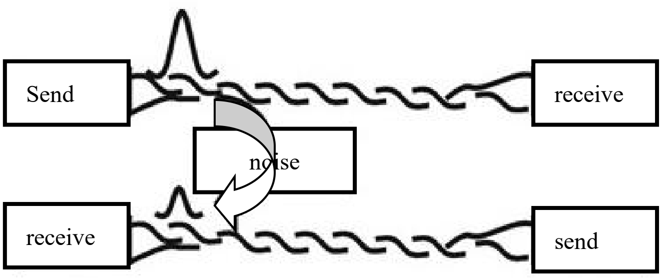
Mini Din 8-pin plug (holes)

2. **RS485 port (COM2, COM2-RS485)**

About RS485 port, A is "+" signal、B is "-" signal. XL series PLC RS485 port is put outside. SG terminal is signal ground. The terminal diagram is shown as below:



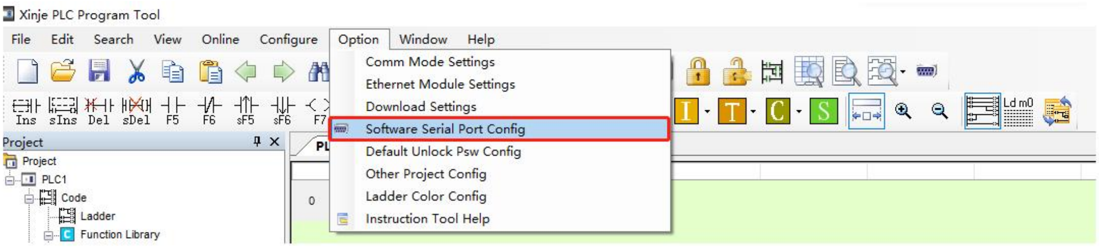
Please use twisted pair cable for RS485. (See below diagram). But shielded twisted pair cable is better and the single-ended connects to the ground.



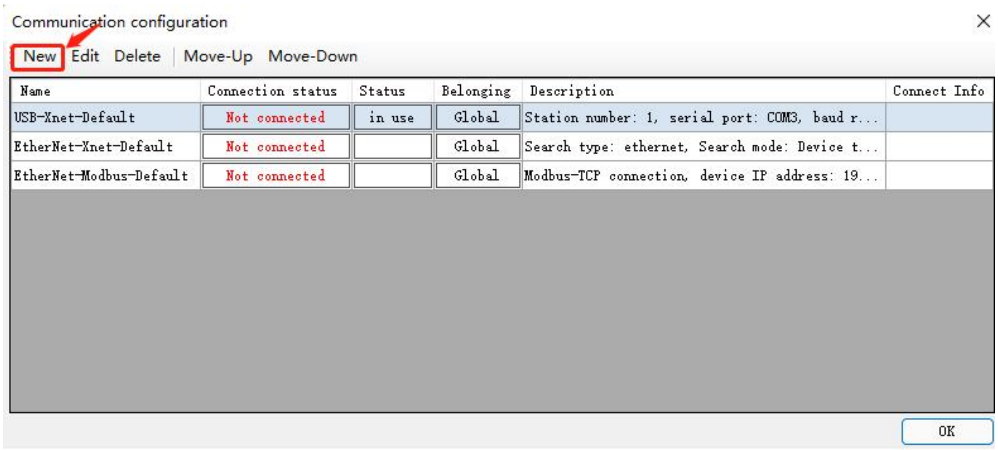
3. **USB port**

When downloading programs and data through the USB port, the USB driver and XINJEConfig tool must be installed first. Because the current USB driver has been built in the XINJEConfig software, the USB driver will be installed automatically after the XINJEConfig software is installed.
After installing the xinje config tool and usb driver, please switch to Xnet mode in the PLC software:

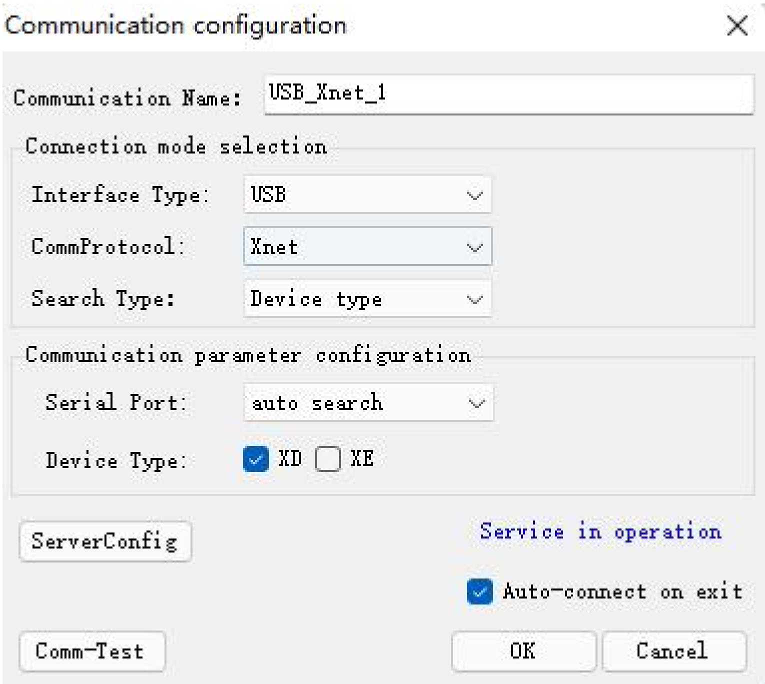
(1) Open XDPPro software, click option/software serial port config

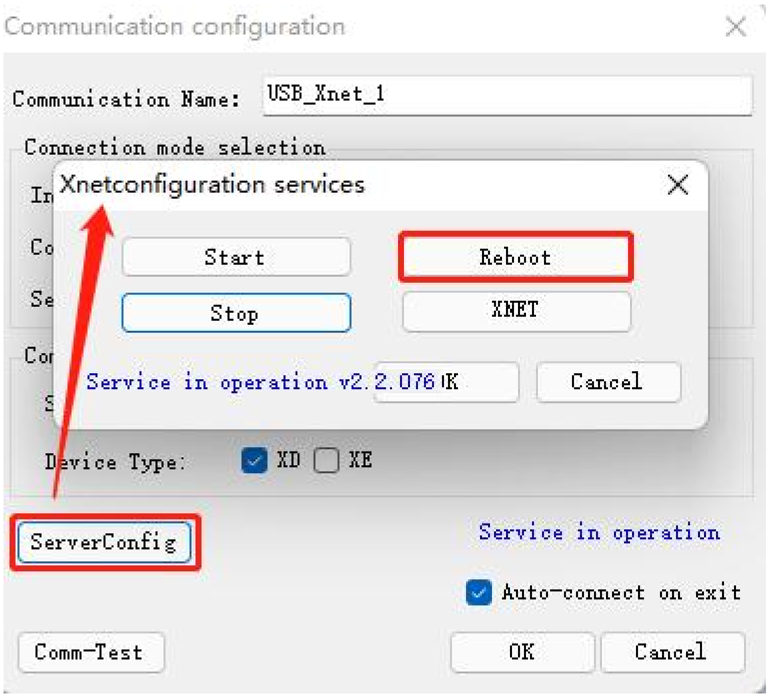


(2) The window of "Communication Configuration" as shown in the picture below pops up, click ‘New’, and the configuration interface is as follows:



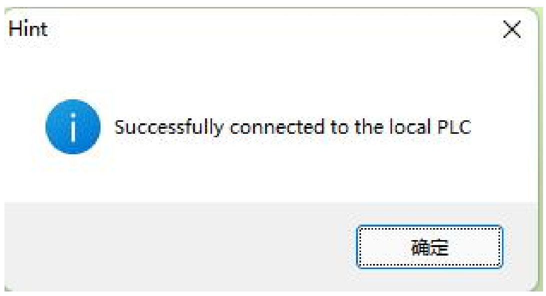
(3) Select USB as the square port communication interface, XNET as the communication protocol, and device type as the search method. After restarting the service, click OK.





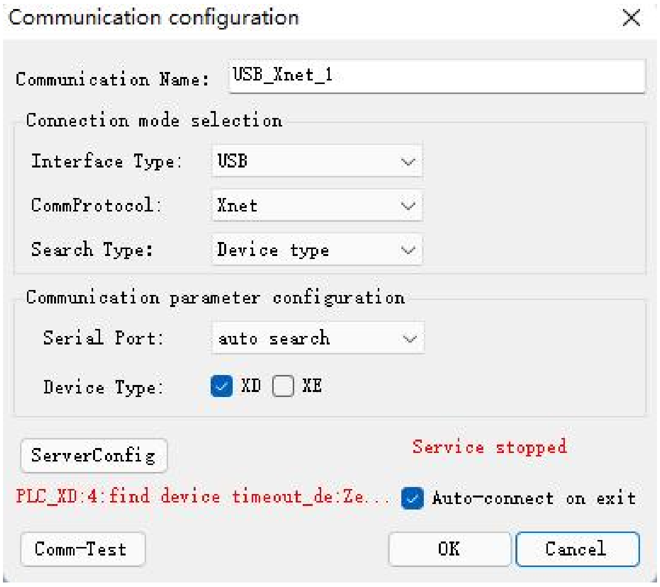
(4) After the connection status is changed to 'in use', click OK:

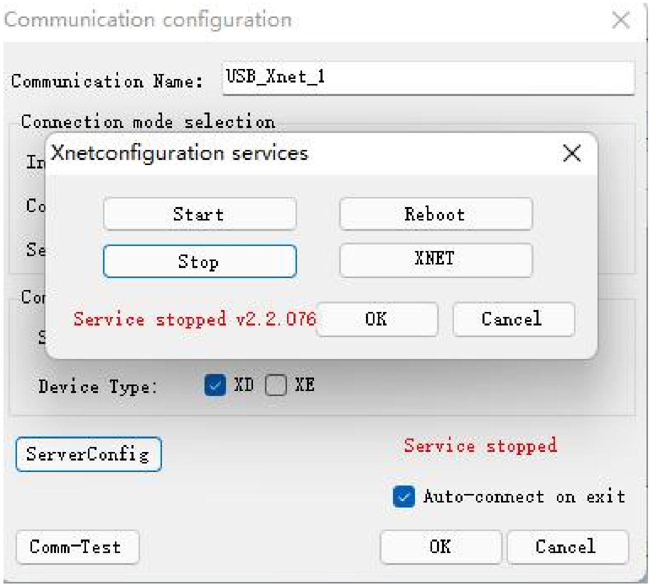
(5) If "Successfully connected to the local PLC" is displayed, the connection is successful.



Note:

(1) If it shows the error "find device timeout", you can click "Restart Service" to try to reconnect, or restart the programming software and PLC to reconnect. If you still can't connect, you need to check whether the PLC is power on, whether the USB download cable is connected properly, whether the USB driver and XINJEConfig software are installed properly.

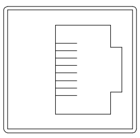




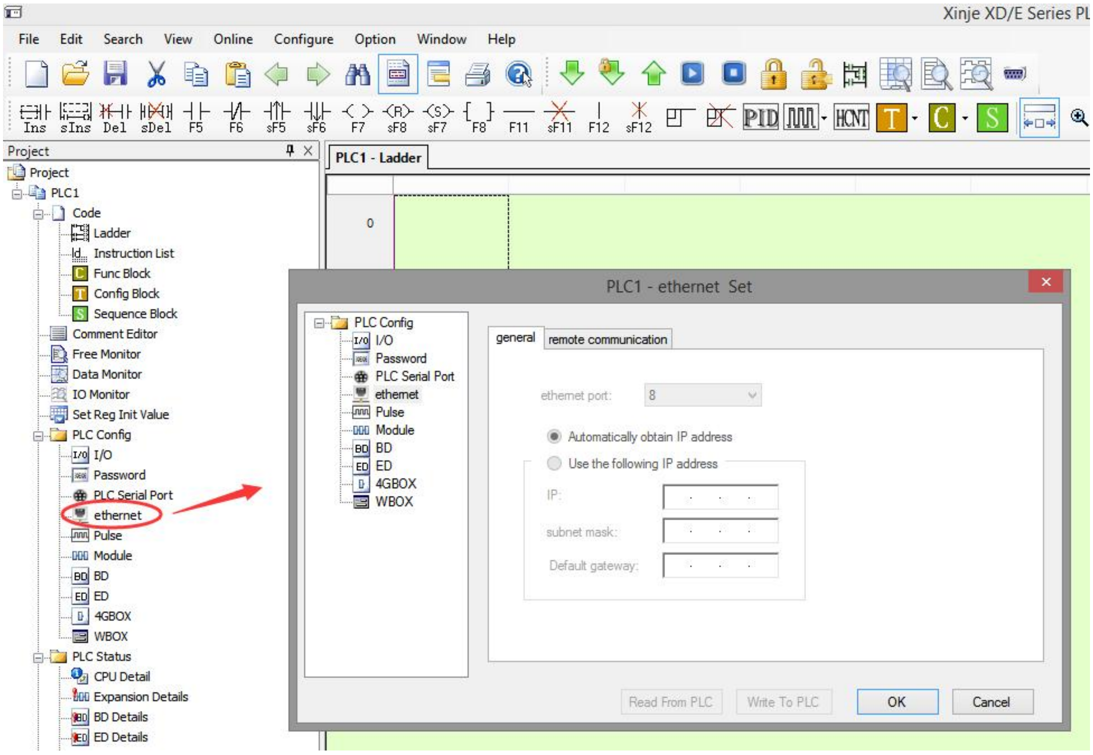
4. **Ethernet port (RJ45)**

RJ45 port is unique for Ethernet PLC, supports TCP/IP Ethernet communication, the port is faster and more stable than USB communication, the data monitoring real-time ability is better, program downloading and uploading is faster. The connection mode of Ethernet communication itself has obvious advantages over RS485 and USB. In many situations of PLC communication, users can communicate with any PLC on the spot through only one switch.

In addition to its application in LAN, Ethernet also supports the remote search, monitoring and operation of PLC, download functions, and communication with other TCP IP devices in the network through the Internet.



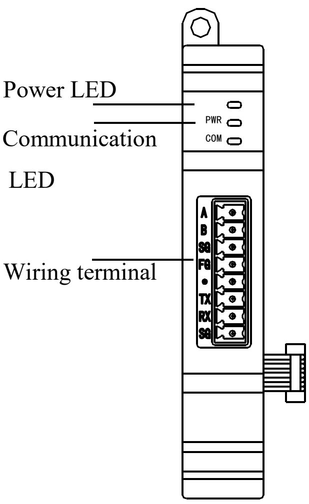
RJ45 port can be configured in "PLC Config-Ethernet" of XINJE PLC programming software, or through XINJEConfig tool. Refer to the relevant manual for details.
The LAN2 port of XDH series PLC supports EtherCAT bus control function. The number of axes is up to 64, and the control cycle is less than 1ms. Please refer to EtherCAT motion control user manual for the specific use of the function.



5. Left extension ED port (COM3)

The left extension ED port can connect ED card to extend RS232 and RS485 port. The ED models include XD-NES-ED (can extend one RS232 and one RS485 port, but the two cannot communicate at the same time).

XD-NES-ED



Each part name is shown as below:

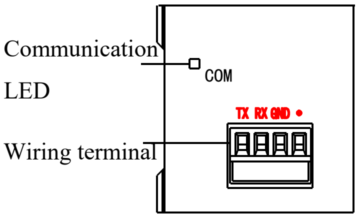
Name		Function
Power LED		The light is ON when the ED module power on
Communication LED		The light is ON when ED module communication is normal
Wiring terminal	A	RS485+
	B	RS485-
	SG	Ground
	FG	Connect to ground terminal
	-	Empty
	TX	RS232 send
	RX	RS232 receive
	SG	Ground

6. Above extension BD port (COM4, COM5)

The above extension port can connect BD card which contains RS232 mode (XD-NS-BD), RS485 mode (XD-NE-BD) and optical fiber mode (XD-NO-BD).

XD series 24/32 I/O PLC can extend one BD card, XD series 48/60 I/O PLC can extend 2 BD cards, XD series 16 I/O PLC cannot extend BD card.

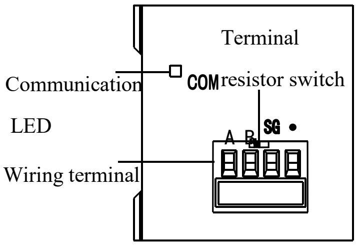
(1) XD-NS-BD



Each part name is shown as below:

Name		Function
Communication LED		Not support this function
Wiring terminal	TX	Signal send
	RX	Signal receive
	GND	Ground
	●	Empty

(2) XD-NE-BD

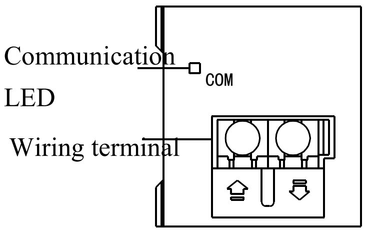


Each part name is shown as below:

Name		Function
Communication LED		The light is flashing when the BD card communication is successful
Wiring terminal	A	485+
	B	485-
	S	Signal ground
	●	Empty
Terminal resistor switch		To choose whether to use terminal resistor (120Ω)

XD-NE-BD has the switch to select whether it is terminal. The switch default setting is OFF which means not install terminal resistor. If XD-NE-BD is at the head or end of the bus, it needs to install 120Ωterminal resistorat the both side and turn on the switch (right).

(3) XD-NO-BD



Each part name is shown as below:

Name	Function
Communication LED	Not support this function
Wiring terminal	The left side is signal input terminal, the right side is signal output terminal

6-1-2 Communication parameters

Communication Parameters

Station	Modbus station number: 1~254
Baud Rate	300bps~9Mbps
Data Bit	8
Stop Bit	1, 1.5, 2
Parity	Even, Odd, even, empty, mask

The default parameters: Station number is 1, baud rate is 19200bps, 8 data bits, 1 stop bit, even parity.
There are many ways to set the parameters of PLC communication port:
There are two ways to set Modbus communication parameters: (1) setting parameters by programming software; (2) setting parameters by XINJEConfig tool, refer to chapter 6-2-6 for details.
Free format communication parameters can be set by programming software, refer to chapter 6-3-2 for details.
X-NET communication parameters can be set by Xinje Config tool. Refer to X-NET fieldbus manual for details.

Note: For the A, B terminal on the PLC body, 1Mbps and higher baud rate is only fit for X-NET communication mode.

6-2 MODBUS communication

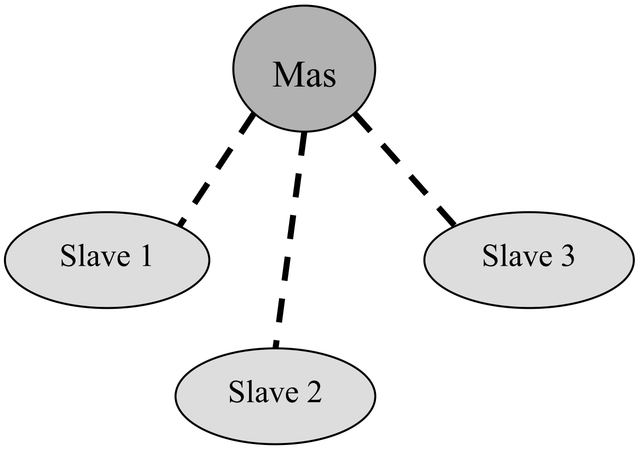
6-2-1 Function overview

XD, XL series PLC support both Modbus master and Modbus slave.

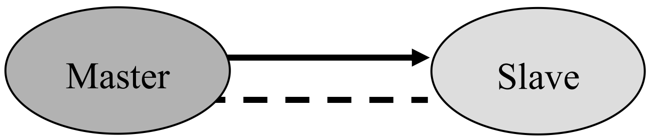
Master mode: When PLC is set to be master, it can communicate with other slave devices which have MODBUS-RTU or MODBUS-ASCII protocol via Modbus instructions; it also can change data with other devices.
For example: Xinje XD3 series PLC can control inverter by Modbus.

Slave mode: When PLC is set to be slave, it can only response with other master devices.

Master and slave: In RS485 network, there can be one master and several slaves at one time (see below diagram). The master station can read and write any slave station. Two slave stations cannot communicate with each other. Master station should write program and read or write one slave station; slave station has no program but only response the master station.(Wiring：connect all 485+, connect all 485-)



In RS232 network (see below diagram), there can only be one master and one slave at one time.



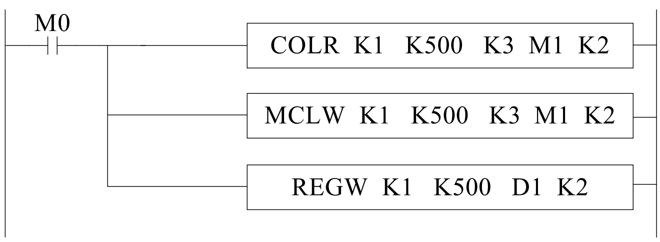
There is dotted line in the diagram. It means any PLC can be master station when all PLC in the network don’t send data. As the PLC do not have unified clock standard, communication will fail when more than one PLC send data at one time. It is not recommended to use.

Note:

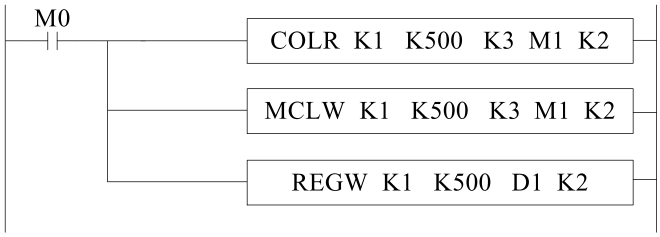
1. For XD/XL series PLC, RS232 and RS485 only support half-duplex.
2. For XC series PLC, if master PLC send one data to slave PLC, and master PLC send data again before slave PLC receiving the last one completely, slave PLC end data error may occur; For XD/XL series PLC, we solve this problem by adding waiting time before communication, which means the slave PLC will receive the next data only after some time the last data finished.

6-2-2 Changing of Modbus instruction

Modbus instruction handling mode has changed in XD/XL series PLC, users can write Modbus instructions directly in program, the protocol station will queue up Modbus requests, which is not the same task with communication; It means users can use one triggering condition to trigger multiple Modbus instructions at the same time. PLC will queue up Modbus requests according to protocol station, which will lead to communication error in XC series PLC.



XC series(×)



XD3 series(√)

Note: XD/XL series PLC sequence block has cancelled Modbus communication instructions, which is replaced by the current Modbus instruction handling mode.

6-2-3 Modbus communication address

The soft component's code in PLC corresponds with Modbus ID number, please see the following table:

1) XD1, XD2, XD3, XL1, XL3 series PLC Modbus address and internal soft component table:

type	component	Address	number	Modbus address (Hex)	Modbus address (decimal)
Coil bit	M	M0~M7999	8000	0~1F3F	0~7999
	X	X0~X77 (main unit)	64	5000~503F	20480~20543
		X10000~X10077 (#1 module)	64	5100~513F	20736~20799
		X10100~X10177 (#2 module)	64	5140~517F	20800~20863
		X10200~X10277 (#3 module)	64	5180~51BF	20864~20927
		X10300~X10377 (#4 module)	64	51C0~51FF	20928~20991
		X10400~X10477 (#5 module)	64	5200~523F	20992~21055
		X10500~X10577 (#6 module)	64	5240~527F	21056~21119
		X10600~X10677 (#7 module)	64	5280~52BF	21120~21183
		X10700~X10777 (#8 module)	64	52C0~52FF	21184~21247
		X11000~X11077 (#9 module)	64	5300~533F	21248~21311
		X11100~X11177 (#10 module)	64	5340~537F	21312~21375
		X20000~X20077(#1 BD)	64	58D0~590F	22736~22799
	Y	Y0~77(main unit)	64	6000~603F	24576~24639
		Y10000~Y10077 (#1 module)	64	6100~613F	24832~24895
		Y10100~Y10177 (#2 module)	64	6140~617F	24896~24959

		Y10200~Y10277 (#3 module)	64	6180~61BF	24960~25023
		Y10300~Y10377 (#4 module)	64	61C0~61FF	25024~25087
		Y10400~Y10477 (#5 module)	64	6200~623F	25088~25151
		Y10500~Y10577 (#6 module)	64	6240~627F	25152~25215
		Y10600~Y10677 (#7 module)	64	6280~62BF	25216~25279
		Y10700~Y10777 (#8 module)	64	62C0~62FF	25280~25343
		Y11000~Y11077 (#9 module)	64	6300~633F	25344~25407
		Y11100~Y11177 (#10 module)	64	6340~637F	25408~25471
		Y20000~Y20077(#1 BD)	64	68D0~690F	26832~26895
	S	S0~S1023	1024	7000~73FF	28672~29695
	SM	SM0~SM2047	2048	9000~97FF	36864~38911
	T	T0~T575	576	A000~A23F	40960~41535
	C	C0~C575	576	B000~B23F	45056~45631
	ET	ET0~ET31	32	C000~C01F	49152~49183
	SEM	SEM0~SEM31	32	C080~C09F	49280~49311
	HM※1	HM0~HM959	960	C100~C4BF	49408~50367
	HS※1	HS0~HS127	128	D900~D97F	55552~55679
	HT※1	HT0~HT95	96	E100~E15F	57600~57695
	HC※1	HC0~HC95	96	E500~E55F	58624~58719
	HSC※1	HSC0~HSC31	32	E900~E91F	59648~59679
Register word	D	D0~D7999	8000	0~1F3F	0~7999
	ID	ID0~ID99(main unit)	100	5000~5063	20480~20579
		ID10000~ID10099 (#1 module)	100	5100~5163	20736~20835
		ID10100~ID10199 (#2 module)	100	5164~51C7	20836~20935
		ID10200~ID10299 (#3 module)	100	51C8~522B	20936~21035
		ID10300~ID10399 (#4 module)	100	522C~528F	21036~21135
		ID10400~ID10499 (#5 module)	100	5290~52F3	21136~21235
		ID10500~ID10599 (#6 module)	100	52F4~5357	21236~21335
		ID10600~ID10699 (#7 module)	100	5358~53BB	21336~21435
		ID10700~ID10799 (#8 module)	100	53BC~541F	21436~21535
		ID10800~ID10899 (#9 module)	100	5420~5483	21536~21635
		ID10900~ID10999 (#10 module)	100	5484~54E7	21636~21735
		ID20000~ID20099	100	58D0~5933	22736~22835

		(#1 BD)			
	QD	QD0~QD99(main unit)	100	6000~6063	24576~24675
		QD10000~QD10099 (#1 module)	100	6100~6163	24832~24931
		QD10100~QD10199 (#2 module)	100	6164~61C7	24932~25031
		QD10200~QD10299 (#3 module)	100	61C8~622B	25032~25131
		QD10300~QD10399 (#4 module)	100	622C~628F	25132~25231
		QD10400~QD10499 (#5 module)	100	6290~62F3	25232~25331
		QD10500~QD10599 (#6 module)	100	62F4~6357	25332~25431
		QD10600~QD10699 (#7 module)	100	6358~63BB	25432~25531
		QD10700~QD10799 (#8 module)	100	63BC~641F	25532~25631
		QD10800~QD10899 (#9 module)	100	6420~6483	25632~25731
		QD10900~QD10999 (#10 module)	100	6484~64E7	25732~25831
		QD20000~QD20099 (#1 BD)	100	68D0~6933	26832~26931
	SD	SD0~SD2047	2048	7000~77FF	28672~30719
	TD	TD0~TD575	576	8000~823F	32768~33343
	CD	CD0~CD575	576	9000~923F	36864~37439
	ETD	ETD0~ETD31	32	A000~A01F	40960~40991
	HD[※1]	HD0~HD999	1000	A080~A467	41088~42087
	HSD[※1]	HSD0~HSD499	500	B880~BA73	47232~47731
	HTD[※1]	HTD0~HTD95	96	BC80~BCDF	48256~48351
	HCD[※1]	HCD0~HCD95	96	C080~C0DF	49280~49375
	HSCD[※1]	HSCD0~HSCD31	32	C480~C49F	50304~50335
	FD[※2]	FD0~FD5119	5120	C4C0~D8BF	50368~55487
	SFD[※2]	SFD0~SFD1999	2000	E4C0~EC8F	58560~60559
	FS[※2]	FS0~FS47	48	F4C0~F4EF	62656~62703

2) XD5, XDM, XDC, XD5E, XDME, XL5, XL5E, XLME, XL5N, XL5H series PLC Modbus address and internal soft component table:

Type	component	Address	numbers	Modbus address (hex)	Modbus address (decimal)
Coil bit	M	M0~M20479	20480	0~4FFF	0~20479
	X	X0~X77(main unit)	64	5000~503F	20480~20543
		X10000~X10077 (#1 module)	64	5100~513F	20736~20799
		X10100~X10177 (#2 module)	64	5140~517F	20800~20863
		X10200~X10277 (#3 module)	64	5180~51BF	20864~20927

		X10300~X10377 (#4 module)	64	51C0~51FF	20928~20991
		X10400~X10477 (#5 module)	64	5200~523F	20992~21055
		X10500~X10577 (#6 module)	64	5240~527F	21056~21119
		X10600~X10677 (#7 module)	64	5280~52BF	21120~21183
		X10700~X10777 (#8 module)	64	52C0~52FF	21184~21247
		X11000~X11077 (#9 module)	64	5300~533F	21248~21311
		X11100~X11177 (#10 module)	64	5340~537F	21312~21375
		X11200~X11277 (#11 module)	64	5380~53BF	21376~21439
		X11300~X11377 (#12 module)	64	53C0~53FF	21440~21503
		X11400~X11477 (#13 module)	64	5400~543F	21504~21567
		X11500~X11577 (#14 module)	64	5440~547F	21568~21631
		X11600~X11677 (#15 module)	64	5480~54BF	21632~21695
		X11700~X11777 (#16 module)	64	54C0~54FF	21696~21759
		X20000~X20077 (#1 BD)	64	58D0~590F	22736~22799
	Y	Y0~77(main unit)	64	6000~603F	24576~24639
		Y10000~Y10077 (#1 module)	640	6100~613F	24832~24895
		Y10100~Y10177 (#2 module)	64	6140~617F	24896~24959
		Y10200~Y10277 (#3 module)	64	6180~61BF	24960~25023
		Y10300~Y10377 (#4 module)	64	61C0~61FF	25024~25087
		Y10400~Y10477 (#5 module)	64	6200~623F	25088~25151
		Y10500~Y10577 (#6 module)	64	6240~627F	25152~25215
		Y10600~Y10677 (#7 module)	64	6280~62BF	25216~25279
		Y10700~Y10777 (#8 module)	64	62C0~62FF	25280~25343
		Y11000~Y11077 (#9 module)	64	6300~633F	25344~25407
		Y11100~Y11177 (#10 module)	64	6340~637F	25408~25471
		Y11200~Y11277 (#11 module)	64	6380~63BF	25472~25535
		Y11300~Y11377 (#12 module)	64	63C0~63FF	25536~25599

		Y11400~Y11477 (#13 module)	64	6400~643F	25600~25663
		Y11500~Y11577 (#14 module)	64	6440~647F	25664~25727
		Y11600~Y11677 (#15 module)	64	6480~64BF	25728~25791
		Y11700~Y11777 (#16 module)	64	64C0~64FF	25792~25855
		Y20000~Y20077(#1 BD)	64	68D0~690F	26832~26895
	S	S0~S7999	8000	7000~8F3F	28672~36671
	SM	SM0~SM4095	4096	9000~9FFF	36864~40959
	T	T0~T4095	4096	A000~AFFF	40960~45055
	C	C0~C4095	4096	B000~BFFF	45056~45151
	ET	ET0~ET39	40	C000~C027	49152~49191
	SEM	SEM0~SEM127	128	C080~C0FF	49280~49407
	HM※1	HM0~HM6143	6144	C100~D8FF	49408~55551
	HS※1	HS0~HS999	1000	D900~DCEF	55552~56551
	HT※1	HT0~HT1023	1024	E100~E4FF	57600~58623
	HC※1	HC0~HC1023	1024	E500~E8FF	58624~59647
	HSC※1	HSC0~HSC36	40	E900~E927	59648~59687
Register word	D	D0~D20479	20480	0~4FFF	0~20479
	ID	ID0~ID99(main unit)	100	5000~5063	20480~20579
		ID10000~ID10099 (#1 module)	100	5100~5163	20736~20835
		ID10100~ID10199 (#2 module)	100	5164~51C7	20836~20935
		ID10200~ID10299 (#3 module)	100	51C8~522B	20936~21035
		ID10300~ID10399 (#4 module)	100	522C~528F	21036~21135
		ID10400~ID10499 (#5 module)	100	5290~52F3	21136~21235
		ID10500~ID10599 (#6 module)	100	52F4~5357	21236~21335
		ID10600~ID10699 (#7 module)	100	5358~53BB	21336~21435
		ID10700~ID10799 (#8 module)	100	53BC~541F	21436~21535
		ID10800~ID10899 (#9 module)	100	5420~5483	21536~21635
		ID10900~ID10999 (#10 module)	100	5484~54E7	21636~21735
		ID11000~ID11099 (#11 module)	100	54E8~554B	21736~21835
		ID11100~ID11199 (#12 module)	100	554C~55AF	21836~21935
		ID11200~ID11299 (#13 module)	100	55B0~5613	21936~22035
		ID11300~ID11399 (#14 module)	100	5614~5677	22036~22135
		ID11400~ID11499	100	5678~56DB	22136~22235

		(#15 module)			
		ID11500~ID11599 (#16 module)	100	56DC~573F	22236~22335
		ID20000~ID20099(#1 BD)	100	58D0~5933	22736~22835
	QD	QD0~QD99(main unit)	100	6000~6063	24576~24675
		QD10000~QD10099 (#1 module)	100	6100~6163	24832~24931
		QD10100~QD10199 (#2 module)	100	6164~61C7	24932~25031
		QD10200~QD10299 (#3 module)	100	61C8~622B	25032~25131
		QD10300~QD10399 (#4 module)	100	622C~628F	25132~25231
		QD10400~QD10499 (#5 module)	100	6290~62F3	25232~25331
		QD10500~QD10599 (#6 module)	100	62F4~6357	25332~25431
		QD10600~QD10699 (#7 module)	100	6358~63BB	25432~25531
		QD10700~QD10799 (#8 module)	100	63BC~641F	25532~25631
		QD10800~QD10899 (#9 module)	100	6420~6483	25632~25731
		QD10900~QD10999 (#10 module)	100	6484~64E7	25732~25831
		QD11000~QD11099 (#11 module)	100	64E8~654B	25832~25931
		QD11100~QD11199 (#12 module)	100	654C~65AF	25932~26031
		QD11200~QD11299 (#13 module)	100	65B0~6613	26032~26131
		QD11300~QD11399 (#14 module)	100	6614~6677	26132~26231
		QD11400~QD11499 (#15 module)	100	6678~66DB	26232~26331
		QD11500~QD11599 (#16 module)	100	66DC~673F	26332~26431
		QD20000~QD20099(#1 BD)	100	68D0~6933	26832~26931
	SD	SD0~SD4095	4096	7000~7FFF	28672~32767
	TD	TD0~TD4095	4096	8000~8FFF	32768~36863
	CD	CD0~CD4095	4096	9000~9FFF	36864~40959
	ETD	ETD0~ETD39	40	A000~A027	40960~40999
	HD※1	HD0~HD6143	6144	A080~B87F	41088~47231
	HSD※1	HSD0~HSD1023	1024	B880~BC7F	47232~48255
	HTD※1	HTD0~HTD1023	1024	BC80~C07F	48256~49279
	HCD※1	HCD0~HCD1023	1024	C080~C47F	49280~40303
	HSCD※1	HSCD0~HSCD39	40	C480~C4A7	50304~50343
	FD※2	FD0~FD8191	8192	C4C0~E4BF	50368~58559
	SFD※2	SFD0~SFD5999	6000	E4C0~FC2F	58560~64559
	FS※2	FS0~FS47	48	F4C0~F4EF	62656~62703

3) XDH and XLH series PLC Modbus address and internal soft component table:

Type	component	Address	numbers	Modbus address (hex)	Modbus address (decimal)
Coil bit	M	M0~M20479	20480	0~4FFF	0~20479
	X	X0~X77(main unit)	64	5000~503F	20480~20543
		X10000~X10077(#1 module)	64	5100~513F	20736~20799
		X10100~X10177(#2 module)	64	5140~517F	20800~20863
		X10200~X10277(#3 module)	64	5180~51BF	20864~20927
		X10300~X10377(#4 module)	64	51C0~51FF	20928~20991
		X10400~X10477(#5 module)	64	5200~523F	20992~21055
		X10500~X10577(#6 module)	64	5240~527F	21056~21119
		X10600~X10677(#7 module)	64	5280~52BF	21120~21183
		X10700~X10777(#8 module)	64	52C0~52FF	21184~21247
		X11000~X11077(#9 module)	64	5300~533F	21248~21311
		X11100~X11177(#10 module)	64	5340~537F	21312~21375
		X11200~X11277(#11 module)	64	5380~53BF	21376~21439
		X11300~X11377(#12 module)	64	53C0~53FF	21440~21503
		X11400~X11477(#13 module)	64	5400~543F	21504~21567
		X11500~X11577(#14 module)	64	5440~547F	21568~21631
		X11600~X11677(#15 module)	64	5480~54BF	21632~21695
		X11700~X11777(#16 module)	64	54C0~54FF	21696~21759
		X20000~X20077(#1 BD)	64	58D0~590F	22736~22799
		X20100~X20177(#2 BD)	64	5910~594F	22800~22863
		X30000~X30077(#1 ED)	64	5BF0~5C2F	23536~23599
	Y	Y0~77(main unit)	64	6000~603F	24576~24639
		Y10000~Y10077(#1 module)	64	6100~613F	24832~24895
		Y10100~Y10177(#2 module)	64	6140~617F	24896~24959
		Y10200~Y10277(#3 module)	64	6180~61BF	24960~25023
		Y10300~Y10377(#4	64	61C0~61F	25024~25087

Type	component	Address	numbers	Modbus address (hex)	Modbus address (decimal)
		module)		F	
		Y10400~Y10477(#5 module)	64	6200~623F	25088~25151
		Y10500~Y10577(#6 module)	64	6240~627F	25152~25215
		Y10600~Y10677(#7 module)	64	6280~62BF	25216~25279
		Y10700~Y10777(#8 module)	64	62C0~62FF	25280~25343
	Y	Y11000~Y11077(#9 module)	64	6300~633F	25344~25407
		Y11100~Y11177(#10 module)	64	6340~637F	25408~25471
		Y11200~Y11277(#11 module)	64	6380~63BF	25472~25535
		Y11300~Y11377(#12 module)	64	63C0~63FF	25536~25599
		Y11400~Y11477(#13 module)	64	6400~643F	25600~25663
		Y11500~Y11577(#14 module)	64	6440~647F	25664~25727
		Y11600~Y11677(#15 module)	64	6480~64BF	25728~25791
		Y11700~Y11777(#16 module)	64	64C0~64FF	25792~25855
		Y20000~Y20077(#1 BD)	64	68D0~690F	26832~26895
		Y20100~Y20177(#2 BD)	64	6910~694F	26896~26956
		Y30000~Y30077(#1 ED)	64	6BF0~6C2F	27632~27695
	S	S0~S7999	8000	7000~8F3F	28672~36671
	SM	SM0~SM4095	4096	9000~9FFF	36864~40959
	T	T0~T4095	4096	A000~AFFF	40960~45055
	C	C0~C4095	4096	B000~BFFF	45056~49151
	ET	ET0~ET39	40	C000~C027	49152~49191
	SEM	SEM0~SEM127	128	C080~C0FF	49280~49407
	HM※1	HM0~HM6143	6144	C100~D8FF	49408~55551
	HS※1	HS0~HS999	1000	D900~DCEF	55552~56551
	HT※1	HT0~HT1023	1024	E100~E4FF	57600~58623
	HC※1	HC0~HC1023	1024	E500~E8FF	58624~59647
	HSC※1	HSC0~HSC39	40	E900~E927	59648~59687

Type	component	Address	numbers	Modbus address (hex)	Modbus address (decimal)
Register word	D	D0~D20479	20480	0~4FFF	0~20479
	ID	ID0~ID99(本体)	100	5000~5063	20480~20579
		ID10000~ID10099(#1 module)	100	5100~5163	20736~20835
		ID10100~ID10199(#2 module)	100	5164~51C7	20836~20935
		ID10200~ID10299(#3 module)	100	51C8~522B	20936~21035
		ID10300~ID10399(#4 module)	100	522C~528F	21036~21135
		ID10400~ID10499(#5 module)	100	5290~52F3	21136~21235
		ID10500~ID10599(#6 module)	100	52F4~5357	21236~21335
		ID10600~ID10699(#7 module)	100	5358~53BB	21336~21435
		ID10700~ID10799(#8 module)	100	53BC~541F	21436~21535
		ID10800~ID10899(#9 module)	100	5420~5483	21536~21635
		ID10900~ID10999(#10 module)	100	5484~54E7	21636~21735
		ID11000~ID11099(#11 module)	100	54E8~554B	21736~21835
		ID11100~ID11199(#12 module)	100	554C~55AF	21836~21935
		ID11200~ID11299(#13 module)	100	55B0~5613	21936~22035
		ID11300~ID11399(#14 module)	100	5614~5677	22036~22135
		ID11400~ID11499(#15 module)	100	5678~56DB	22136~22235
		ID11500~ID11599(#16 module)	100	56DC~573F	22236~22335
		ID20000~ID20099(#1 BD)	100	58D0~5933	22736~22835
		ID20100~ID20199(#2 BD)	100	5934~5997	22836~22935
		ID30000~ID30099(#1 ED)	100	5BF0~5C53	23536~23635
	QD	QD0~QD99(main unit)	100	6000~6063	24576~24675
		QD10000~QD10099(#1 module)	100	6100~6163	24832~24931
		QD10100~QD10199(#2 module)	100	6164~61C7	24932~25031
		QD10200~QD10299(#3 module)	100	61C8~622B	25032~25131
		QD10300~QD10399(#4 module)	100	622C~628F	25132~25231
		QD10400~QD10499(#5	100	6290~62F3	25232~25331

Type	component	Address	numbers	Modbus address (hex)	Modbus address (decimal)
		module)			
		QD10500~QD10599(#6 module)	100	62F4~6357	25332~25431
		QD10600~QD10699(#7 module)	100	6358~63BB	25432~25531
		QD10700~QD10799(#8 module)	100	63BC~641F	25532~25631
	QD	QD10800~QD10899(#9 module)	100	6420~6483	25632~25731
		QD10900~QD10999(#10 module)	100	6484~64E7	25732~25831
		QD11000~QD11099(#11 module)	100	64E8~654B	25832~25931
		QD11100~QD11199(#12 module)	100	654C~65AF	25932~26031
		QD11200~QD11299(#13 module)	100	65B0~6613	26032~26131
		QD11300~QD11399(#14 module)	100	6614~6677	26132~26231
		QD11400~QD11499(#15 module)	100	6678~66DB	26232~26331
		QD11500~QD11599(#16 module)	100	66DC~673F	26332~26431
		QD20000~QD20099(#1 BD)	100	68D0~6933	26832~26931
		QD20100~QD20199(#2 BD)	100	6934~6997	26932~27031
		QD30000~QD30099(#1 ED)	100	6BF0~6C53	27632~27731
	SD	SD0~SD4095	4096	7000~7FFF	28672~32767
	TD	TD0~TD4095	4096	8000~8FFF	32768~36863
	CD	CD0~CD4095	4096	9000~9FFF	36864~40959
	ETD	ETD0~ETD39	40	A000~A027	40960~40999
	HD※1	HD0~HD6143	6144	A080~B87F	41088~47231
	HSD※1	HSD0~HSD1023	1024	B880~BC7F	47232~48255
	HTD※1	HTD0~HTD1023	1024	BC80~C07F	48256~49279
	HCD※1	HCD0~HCD1023	1024	C080~C47F	49280~50303
	HSCD※1	HSCD0~HSCD39	40	C480~C4A7	50304~50343
	FD※2	FD0~FD8191	8192	C4C0~E4BF	50368~58559
	SFD※2	SFD0~SFD4095	4096	E4C0~FC2F	58560~64559
	FS※2	FS0~FS47	256	F4C0~F4EF	62656~62911

Note:
1. the power down holding area is marked with ※1, and the flash area is marked with ※2.
2: the address in the above table is used when PLC is the lower computer and Modbus RTU or MODBUS ASCII protocol is used for communication, the general upper computer is: SCADA/HMI/PLC.
3: if the upper computer is PLC, program according to Modbus RTU or MODBUS ASCII protocol.
4: if the upper computer is SCADA or HMI, there are two situations: the first one has the Xinje driver, for example: Xinje HMI / Zijinqiao SCADA.
The program can be written directly by using PLC internal soft components (Y0 / M0); for the second type, Modbus RTU or Modbus ASCII is selected if there is no Xinje driver, and then use the addresses in the table above to define the data variables.
5: input and output point is octal, please calculate corresponding input and output point MODBUS address according to octal, for example: MODBUS corresponding to Y0, the address is H6000, the Modbus address corresponding to Y10 is H6008 (not H6010), and the Modbus address corresponding to Y20 is H6010 (not H6020).
6: when the Modbus address exceeds 32767, it needs to be expressed in hexadecimal, and "0" should be added before the address. For example: MODBUS of HD0 is 41088 in decimal (beyond 32767), and 41088 cannot be written into the software, so it needs to be expressed in hexadecimal as H0A080.
7: Calculation of Modbus address of X and Y, taking X as an example, the calculation of Modbus address of Y is the same as that of X.
X0: 20480 X10: 20480+8   X20: 20480+16 X30: 16384+24…
X10000: 20736 X10010: 20736+8   X10020: 20736+16…
X10200: 20800 X10210: 20800+8   X10220: 20800+16…

6-2-4 Modbus data format

Modbus transmission mode:

There are two transmission modes: RTU and ASCII; It defines serial transmission of bit content in message domain; it decides how information to pack and decode; transmission mode (and port parameters) of all devices in Modbus serial links should be the same.

Modbus-RTU data structure

1.RTU mode:

Under Modbus RTU (remote terminal unit) mode, message has two 4-bit hexadecimal characters in every 8-bit byte. This mode has very high data density, higher throughput rate than Modbus ASCII. Every message should be sent by continuous characters.
RTU mode frame check domain: cycle redundancy check(CRC).
RTU mode frame description:

Modbus station	Function code	data	CRC	
1 byte	1 byte	0~252 byte	2 byte	
			CRC low	CRC high

Format:

START	No input signal ≧ 10ms
Address(station no.)	Communication address: 8-bit binary
Function	Function code：8-bit binary
DATA(n - 1)	Data content： N*8-bit data, N≤8, max 8 bytes
……	
DATA 0	
CRC CHK Low	CRC check code
CRC CHK High	16-bit CRC check code is consist of two 8-bit binary
END	No input signal ≧ 10ms

2. Modbus address：

00H：All the Xinje XC series PLC broadcast—— slave stations don't response.

01H：Communicate with address 01H PLC.

0FH：Communicate with address 15H PLC.

10H：Communicate with address 16H PLC and so on. Up to 254(FEH).

3. Function and DATA:

Function code	Function	Modbus instruction
01H	Read coil	COLR
02H	Read input coil	INPR(not support Xinje PLC)
03H	Read register	REGR
04H	Read input register	INRR
05H	Write coil	COLW
06H	Write register	REGW
10H	Write multi-register	MRGW
0FH	Write multi-coil	MCLW

(1) Take 06H function code as example (single register write), and introduce data format.

E.g.: upper computer write data to PLC H0002 (D2).

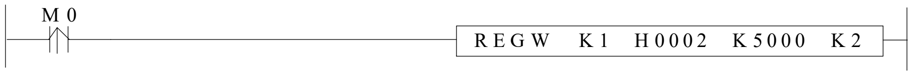
RTU mode:

Asking format		Response format	
ID	01H	ID	01H
Function code	06H	Function code	06H
Register ID	00H	Register ID	00H
	02H		02H
Data content	13H	Data contents	13H
	88H		88H
CRC CHECK High	25H	CRC CHECK High	25H
CRC CHECK Low	5CH	CRC CHECK Low	5CH

Explanation：

1. Address is PLC station no.
2. Function code is Modbus-RTU protocol read/write code.
3. Register address is the PLC modbus address, please see chapter 6-2-3.
4. Data content is the value in D2.
5. CRC CHECK High / CRC CHECK Low is high and low bit of CRC check value.

If 2 pieces of Xinje XD3 series PLC communicate with the other one, write K5000 to D2.



M0 is trigger condition (Rising edge). If communication fails, the instruction will try twice. If the third time communication fails, then communication ends.

The relationship between REGW and Modbus RTU protocol (other instructions are the same)

REGW	Function code 06H
K1	Station no.
H0002	Modbus address
K5000	Data contents 1388H
K2	PLC serial port

The complete communication datum are: 01H 06H 00H 02H 13H 88H (system take CRC checking automatically)

If monitor the serial port2 data by serial port debugging tool, the datum are: 01 06 00 02 13 88 25 5C

Note: The instruction doesn't distinguish decimal, hex, binary, octal etc. For example, B10000, K16 and H10 are the same value, so the following instructions are the same.

REGW K1 B111110100 D1 K2

REGW K1 K500 D1 K2

REGW K1 H1F4 D1 K2

(2) Function code 01H/02H: read coil/read input coil

Eg. Read coil address 6000H (Y0). At this time, Y0 and Y1 are ON.

RTU mode:

Asking format		**Response format**	
Address	01H	Address	01H
Function code	01H/02H	Function code	01H/02H
Coil address	60H	Byte number	01H
	00H		
Coil number	00H	Data contents	03H
	02H		
CRC CHECK Low	A3H	CRC CHECK Low	11H
CRC CHECK High	CBH	CRC CHECK High	89H

As the status of Y0 and Y1 is ON, the data contents are 03H (0000 0011).

(3)Function code 03H: read register

Eg. Read two register starting from03E8H (D1000, D1001).

RTU mode:

Asking format		**Response format**	
Address	01H	Address	01H
Function code	03H	Function code	03H
Register address	03H	Byte number	04H
	E8H		
Register number	00H	Data contents	12H

			2EH
	02H		04H
			E8H
CRC CHECK Low	44H	CRC CHECK Low	9DH
CRC CHECK High	7BH	CRC CHECK High	CCH

At this time, the data read fromD1000 and D1001 are 122EH (4654) and 04E8H (1256).

(4)Function code 05H: write single coil

Eg. Set on the coil address 6000H (Y0).

RTU mode:

Asking format		Response format	
Address	01H	Address	01H
Function code	05H	Function code	05H
Coil address	60H	Coil address	60H
	00H		00H
Data contents (low byte is before high byte)	FFH	Data contents	FFH
	00H		00H
CRC CHECK Low	92H	CRC CHECK Low	92H
CRC CHECK High	3AH	CRC CHECK High	3AH

Note: when writing single coil, ON is 00FFH, OFF is 0000H; the low byte is before high byte for the data contents.

(5)Function code0FH: write multiple coils

Eg. Write 16 coils start from address 6000H (Y0).

RTU mode:

Asking format		Response format	
Address	01H	Address	01H
Function code	0FH	Function code	0FH
Coil address	60H	Coil address	60H
	00H		00H
Coil number	00H	Coilnumber	00H
	10H		10H
Byte number	02H	-	-
Data contents (low byte is before high byte)	03H		
	01H		
CRC CHECK Low	43H	CRC CHECK Low	4AH
CRC CHECK High	16H	CRC CHECK High	07H

The data contents are 0103H, the binary format is 0000 0001 0000 0011, write in corresponding Y17~Y0, so Y0, Y1, Y10 are set ON.

Note: when writing the data contents, the low byte is before the high byte.

(6)Function code 10H: write multiple registers

Eg. Write 3 registers starting from address 0000H (D0).

RTU mode:

Asking format		Response format	
Address	01H	Address	01H
Function code	10H	Function code	10H
Register address	00H	Register address	00H
	00H		00H
Register number	00H	Register number	00H
	03H		03H
Byte number	06H	-	-
Data contents	00H		
	01H		
	00H		
	02H		
	00H		
	03H		
CRC CHECK Low	3AH	CRC CHECK Low	3AH
CRC CHECK High	81H	CRC CHECK High	81H

After executing, the value in D0, D1, D2 are 1, 2, 3.

Note: byte number = register number * 2.

Modbus-ASCII data structure

1. ASCII mode:

For Modbus ASCII(American Standard Code for Information Interchange)mode in serial links, every 8-bit byte is sent as two ASCII characters. When communication links and devices do not fit RTU mode timing monitor, we usually use the ASCII mode.

Note：One byte needs two characters, so ASCII mode has lower inefficiency than RTU mode.

E.g.: Byte 0X5B will be encoded as two characters: 0x35 and 0x42(ASCII code 0x35 ="5"，0x42 ="B").

ASCII mode frame check domain：Longitudinal Redundancy Checking (LRC)

ASCII mode frame description:

Start mark	Modbus no.	Function code	data	LRC	End mark	
1 character	2 characters	2 characters	0~252*2 characters	2 characters	2 characters	
0x3A					0x0D	0x0A

Format：

STX (3AH)	Start mark=3AH
Address code high bit	Communication position(no)：Consist of 2 ASCII codes
Address code low bit	
Function code high bit	Function code(command)：Consist of 2 ASCII codes
Function code low bit	
Instruction start ID	Command start bit：Consist of 4 ASCII codes
Instruction start ID	
Instruction start ID	
Instruction start ID	
Data length	Length from start to end：

Data length	Consist of 4 ASCII codes
Data length	
Data length	
LRC check high bit	LRC check code： Consist of 2 ASCII codes
LRC check low bit	
END high bit	End mark： END Hi=CR(0DH)，END Lo=CR(0AH)
END low bit	

2. Communication address：

00H：All Xinje XC series PLC broadcast—— slave stations do not response.

01H：Communicate with address 01H PLC.

0FH：Communicate with address 15H PLC.

10H：Communicate with address 16H PLC.

And so on, up to 254(FEH).

3. Function and DATA：

Function code	Function	Corresponding modbus
01H	Read coil	COLR
02H	Read input coil	INRR
03H	Read register	REGR
04H	Read input register	INRR
05H	Write single coil	COLW
06H	Write single register	REGW
10H	Write multiple registers	MRGW
0FH	Write multiple coils	MCLW

Take 06H function code(write single register)as example, and introduce data format(other functions are similar to this)：

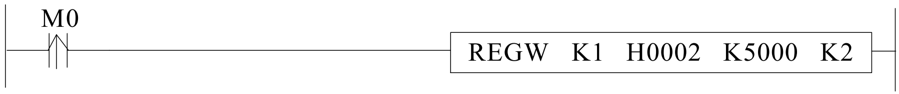
E.g.：upper computer write data K5000(H1388) to PLC H0002 (D2).

ASCII mode：

Start mark	3AH
ID	30H
	31H
Function code	30H
	36H
Register ID high byte	30H
	30H
Register ID low byte	30H
	32H
Data content high byte	31H
	33H
Data content low byte	38H
	38H
LRC	35H
	43H
End mark	0DH
	0AH

Description：
1. address is PLC station number.
2. Function code is Modbus-ASCII protocol read/write code.
3. Register ID is the PLC modbus communication ID, please see chapter 7-2-2.
4. Data content is the value in D2.
5. LRC CHECK Low / CRC CHECK High is low and high bit of CRC check value.

If two pieces of Xinje XD3 PLC communicate with each other, write K5000 to D2.



M0 is trigger condition (rising edge). When Xinje PLC communicates by Modbus, if communication fails, the instruction will try twice. If the third time communication fails, then communication ends.
The relationship between REGW and ASCII protocol (other instructions are similar to this):

REGW	Function code 06H
K1	Station number
H0002	Modbus ID
K5000	Data content is 1388H
K2	PLC communication serial port

Complete data string: 3AH 30H 31H 30H 36H 30H 30H 30H 32H 31H 33H 38H 38H 35H 43H
(system take CRC checking automatically)
If monitor the serial port2 by serial port debugging tool, the datum are: 3AH 30H 31H 30H 36H 30H 30H 30H 32H 31H 33H 38H 38H 35H 43H 0DH 0AH
Note: The data does not distinguish decimal, binary, hexadecimal etc. For example, B10000, K16 and H10 are the same value, so the following instructions are the same.
REGW K1 B111110100 D1 K2
REGW K1 K500 D1 K2
REGW K1 H1F4 D1 K2

6-2-5 Communication Instructions

Modbus instructions include coil read/write, register read/write; below will introduce the details.
Instructions in details:
The operand definition in the instruction:
1. Remote communication station and serial port number.
E.g.： one PLC connects 3 inverters. PLC needs to write and read the parameters of inverter. The inverter station number is 1.2 and 3. So the remote communication number is 1.2 and 3.

2. Remote register/coil start ID number:
Assign remote coil/register number: the start coil/register ID of PLC read and write, it is normally used with 'assigned coil/register number'.
E.g.： PLC read Xinje inverter's output frequency (H2103), output current(H2104), bus voltage(H2105), then remote register/coil start ID is H2103, assigned coil number is K3.

3. Local receipt/send coil/register address：Coil/register in PLC used to exchange data with lower computer.

E.g.： write coil M0：write M0 status to assigned address in lower computer

Write register D0：write D0 value to assigned address

Read coil M1：read content in lower computer assigned address to M1

Read register D1：read content in lower computer assigned address to D1

4. communication condition:

The preconditions of Modbus communication can be normal open/closed coil and rising/falling edge. When the open/close coil triggers, Modbus instructions will always be executed. When the communication between multiple slave stations or the traffic is large, communication delay may occur. The oscillating coil can be used as triggering condition. When the rising/falling edge triggers, Modbus instructions will only be executed once, and only when the next rising/falling edge comes, Modbus instructions will be executed again.

Coil Read [COLR]

1) Summary

Read the specified station’s coil status to the local device;

Coil read [COLR]			
16 bits instruction	COLR	32 bits instruction	-
Execution condition	Normally ON/OFF coil	Suitable models	XD, XL
Hardware requirement	-	Software Requirement	-

2) Operands

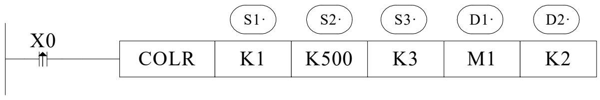
Operands	Function	Type
S1	Specify the remote communication station no.	16 bits, BIN
S2	Specify the remote coil start address	16 bits, BIN
S3	Specify the coil quantity	16 bits, BIN
D1	Specify the local coil start address	bits
D2	Specify the serial port no.	16 bits, BIN

3) Suitable soft components

Operands	Word soft elements											Bit soft elements						
	System								Constant	Module		System						
	D	FD	TD	CD	DX	DY	DM	DS	K/H	ID	QD	X	Y	M	S	T	C	Dn.m
S1	●	●		●	●				●									
S2	●	●		●	●				●									
S3	●	●		●	●				●									
D1												●	●	●	●	●	●	
D2									K									

Notes: D includes D, HD; TD includes TD, HTD; CD includes CD, HCD, HSCD, HSD; DM includes DM, DHM. M includes M, HM, SM; S includes S and HS; T includes T and HT; C includes C and HC.

Function



- Read the coil, Modbus function code 01H.
- Serial port: K0~K5. K0: COM0 (RS232), K1: COM1(RS232), K2: COM2(RS485), K3: COM3(left extension port), K4: COM4(upper extension port 1), K5: COM5(upper extension port 2).
- Operands S3: K1~K2000, the max coil quantity is 2000.
- When X0 is ON, COLR instruction is executed. When the instruction starts to execute, the Modbus read and write flag SM160 (serial port 2) is set on; when the execution is completed, SM160 (serial port 2) is set OFF. If a communication error occurs and the number of resend is set, it will be automatically resend. Users can check the relevant registers to determine the cause of the error. The execution result of Modbus read and write instructions of serial port 2 is in SD160.

Input coil read [INPR]

1)Summary

Read the specified station's input coil status to local device.

Input coil read[INPR]			
16 bits instruction	INPR	32 bits instruction	-
Execution condition	Normally ON/OFF, rising edge	Suitable models	XD, XL
Hardware requirement	-	Software requirement	-

2)Operands

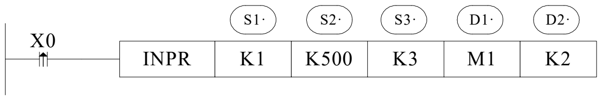
Operands	Function	Type
S1	Specify remote communication station no.	16 bits, BIN
S2	Specify remote coil start address number	16 bits, BIN
S3	Specify coil number	16 bits, BIN
D1	Specify start address number of local receipt coils	bit
D2	Specify serial port number	16 bits, BIN

3)Suitable soft components

Operands	Word soft elements											Bit soft elements						
	System								Constant	Module		System						
	D	FD	TD	CD	DX	DY	DM	DS	K/H	ID	QD	X	Y	M	S	T	C	Dn.m
S1	●	●		●	●				●									
S2	●	●		●	●				●									
S3	●	●		●	●				●									
D1												●	●	●	●	●	●	
D2									K									

Notes: D includes D, HD; TD includes TD, HTD; CD includes CD, HCD, HSCD, HSD; DM includes DM, DHM; DS includes DS, DHS. M includes M, HM, SM; S includes S and HS; T includes T and HT; C includes C and HC.

Function



- Read input coil, Modbus function code is 02H.
- Serial port: K0~K5. K0: COM0 (RS232), K1: COM1(RS232), K2: COM2(RS485), K3: COM3(left extension port), K4: COM4(upper extension port 1), K5: COM5(upper extension port 2).
- Operand S3: K1~K2000, max input coil number is 2008.
- When X0 is ON, INPR instruction is executed, Modbus read write flag SM160(serial port2) is set ON, SM160 is set OFF when the execution is completed. If a communication error occurs and the number of resend is set, it will be automatically resend. Users can check the relevant registers to determine the cause of the error. The execution result of Modbus read and write instructions of serial port 2 is in SD160.
- This instruction cannot read XINJE PLC input coil.

Single Coil Write [COLW]

1)Summary

Write local device specified coil to remote station no's coil.

Single Coil write [COLW]			
16 bits instruction	COLW	32 bits instruction	-
Execution Condition	Normally ON/OFF, edge triggering	Suitable Models	XD, XL
Hardware Requirement	-	Software Requirement	-

2)Operands

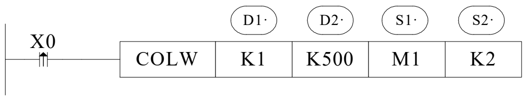
Operands	Function	Type
D1	Specify remote communication station number	16 bits, BIN
D2	Specify remote coil start address	16 bits, BIN
S1	Specify start address of local coil	bit
S2	Specify serial port number	16 bits, BIN

3)Suitable soft components

Operands	Word soft elements											Bit soft elements						
	System								Constant	Module		System						
	D	FD	TD	CD	DX	DY	DM	DS	K/H	ID	QD	X	Y	M	S	T	C	Dn.m
D1	●	●		●	●				●									
D2	●	●		●	●				●									
S1												●	●	●	●	●	●	
S2									K									

Notes: D includes D, HD; TD includes TD, HTD; CD includes CD, HCD, HSCD, HSD; DM includes DM, DHM; DS includes DS, DHS. M includes M, HM, SM; S includes S and HS; T includes T and HT; C includes C and HC.

Function



- Write single coil, Modbus function code is 05H.
- Serial port: K0~K5. K0: COM0 (RS232), K1: COM1(RS232), K2: COM2(RS485), K3: COM3(left extension port), K4: COM4(upper extension port 1), K5: COM5(upper extension port 2).
- When X0 is ON, COLW instruction is executed, Modbus read write flag SM160(serial port2) is set ON, SM160 is set OFF when the execution is completed. If a communication error occurs and the number of resend is set, it will be automatically resend. Users can check the relevant registers to determine the cause of the error. The

execution result of Modbus read and write instructions of serial port 2 is in SD160.

Multiple coils write [MCLW]

1)Summary

Write local device multiple coils to remote station no's coil.

Multiple coils write [MCLW]			
16 bits instruction	MCLW	32 bits instruction	-
Execution Condition	Normally ON/OFF, edge triggering	Suitable models	XD, XL
Hardware Requirement	-	Software Requirement	-

2)Operands

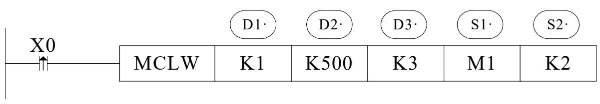
Operands	Function	Type
D1	Specify remote communication station number	16 bits，BIN
D2	Specify remote coil start address	16 bits，BIN
D3	Specify coil number	16 bits，BIN
S1	Specify start address of local coils	bit
S2	Specify serial port number	16 bits，BIN

3)Suitable soft components

Operands	Word soft elements											Bit soft elements						
	System								Constant	Module		System						
	D	FD	TD	CD	DX	DY	DM	DS	K/H	ID	QD	X	Y	M	S	T	C	Dn.m
S1	●	●	●	●					●									
S2	●	●	●	●					●									
S3	●	●	●	●					●									
D1												●	●	●	●	●	●	
D2									K									

Notes: D includes D, HD; TD includes TD, HTD; CD includes CD, HCD, HSCD, HSD; DM includes DM, DHM; DS includes DS, DHS. M includes M, HM, SM; S includes S and HS; T includes T and HT; C includes C and HC.

Function



- Write multiple coils, Modbus function code is 0FH.
- Serial port: K0~K5. K0: COM0 (RS232), K1: COM1(RS232), K2: COM2(RS485), K3: COM3(left extension port), K4: COM4(upper extension port 1), K5: COM5(upper extension port 2).

- Operand D3: max coil number is 1976.
- When X0 is ON, MCLW instruction is executed, Modbus read write flag SM160(serial port2) is set ON, SM160 is set OFF when the execution is completed. If a communication error occurs and the number of resend is set, it will be automatically resend. Users can check the relevant registers to determine the cause of the error. The execution result of Modbus read and write instructions of serial port 2 is in SD160.

Register read [REGR]

1)Summary

Read remote station no's register to local device.

Register read[REGR]			
16 bits instruction	REGR	32 bits instruction	-
Execution Condition	Normally ON/OFF, edge triggering	Suitable models	XD, XL
Hardware Requirement	-	Software Requirement	-

2)Operands

Operands	Function	Type
S1	Specify remote communication station number	16 bits，BIN
S2	Specify remote register start address	16 bits，BIN
S3	Specify register number	16 bits，BIN
D1	Specify start address of local register	16 bits，BIN
D2	Specify serial port number	16 bits，BIN

3)Suitable soft components

Operands	Word soft elements											Bit soft elements						
	System								Constant	Module		System						
	D	FD	TD	CD	DX	DY	DM	DS	K/H	ID	QD	X	Y	M	S	T	C	Dn.m
S1	●	●	●	●					●									
S2	●		●	●					●									
S3	●	●	●	●					●									
D1	●																	
D2									K									

*Notes: D includes D, HD; TD includes TD, HTD; CD includes CD, HCD, HSCD, HSD; DM includes DM, DHM; DS includes DS, DHS. M includes M,HM,SM； S includes S,HS； T includes T,HT； C includes C, HC.

Function

X0		S1·	S2·	S3·	D1·	D2·
⤒	REGR	K1	K500	K3	D1	K2

- Read register, Modbus function code is 03H.
- Serial port: K0~K5. K0: COM0 (RS232), K1: COM1(RS232), K2: COM2(RS485), K3: COM3(left extension port), K4: COM4(upper extension port 1), K5: COM5(upper extension port 2).
- Operand S3: max register number is 125.
- When X0 is ON, REGR instruction is executed, Modbus read write flag SM160(serial port2) is set ON, SM160 is set OFF when the execution is completed. If a communication error occurs and the number of resend is set, it will be automatically resend. Users can check the relevant registers to determine the cause of the error. The execution result of Modbus read and write instructions of serial port 2 is in SD160.

Input register read [INRR]

1)Summary

Read remote station no’s input register to local device.

Input register read [INRR]			
16 bits instruction	INRR	32 bits instruction	-
Execution Condition	Normally ON/OFF, edge triggering	Suitable models	XD, XL
Hardware Requirement	-	Software Requirement	-

2)Operands

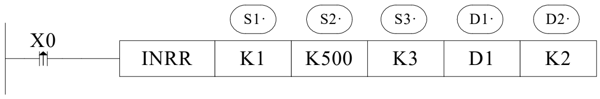
Operands	Function	Type
S1	Specify remote communication station number	16 bits，BIN
S2	Specify remote register start address	16 bits，BIN
S3	Specify register number	16 bits，BIN
D1	Specify start address of local register	16 bits，BIN
D2	Specify serial port number	16 its，BIN

3)Suitable soft components

Operands	Word soft elements											Bit soft elements						
	System								Constant	Module		System						
	D	FD	TD	CD	DX	DY	DM	DS	K/H	ID	QD	X	Y	M	S	T	C	Dn.m
S1	●	●	●	●					●									
S2	●		●	●					●									
S3	●	●	●	●					●									
D1	●																	
D2									K									

*Notes: D includes D, HD; TD includes TD, HTD; CD includes CD, HCD, HSCD, HSD; DM includes DM, DHM; DS includes DS, DHS. M includes M,HM,SM；S includes S,HS；T includes T,HT；C includes C, HC.

Function



- Read input register, Modbus function code is 04H.
- Serial port: K0~K5. K0: COM0 (RS232), K1: COM1(RS232), K2: COM2(RS485), K3: COM3(left extension port), K4: COM4(upper extension port 1), K5: COM5(upper extension port 2).
- Operand S3: max register number is 125.
- When X0 is ON, INRR instruction is executed, Modbus read write flag SM160(serial port2) is set ON, SM160 is set OFF when the execution is completed. If a communication error occurs and the number of resend is set, it will be automatically resend. Users can check the relevant registers to determine the cause of the error. The execution result of Modbus read and write instructions of serial port 2 is in SD160.

Single Register write [REGW]

1)summary

Write local device register to specified remote station no's register.

Register write[REGW]			
16 bits instruction	REGW	32 bits instruction	-
Execution Condition	Normally ON/OFF, edge triggering	Suitable models	XD, XL
Hardware Requirement	-	Software Requirement	-

2)Operands

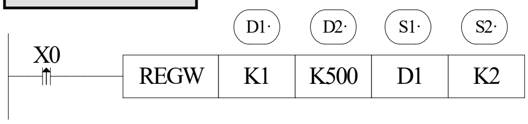
Operands	Function	Type
D1	Specify remote communication station number	16 bits，BIN
D2	Specify remote register start address	16 bits，BIN
S1	Specify start address of local register	16 bits，BIN
S2	Specify serial port number	16 bits，BIN

3)Suitable soft components

Operands	Word soft elements											Bit soft elements						
	System								Constant	Module		System						
	D	FD	TD	CD	DX	DY	DM	DS	K/H	ID	QD	X	Y	M	S	T	C	Dn.m
D1	●	●	●	●					●									
D2	●	●	●	●					●									
S1	●																	
S2									K									

*Notes: D includes D, HD; TD includes TD, HTD; CD includes CD, HCD, HSCD, HSD; DM includes DM, DHM; DS includes DS, DHS. M includes M,HM,SM； S includes S,HS； T includes T,HT； C includes C, HC.

Function



- Write register, Modbus function code is 06H.
- Serial port: K0~K5. K0: COM0 (RS232), K1: COM1(RS232), K2: COM2(RS485), K3: COM3(left extension port), K4: COM4(upper extension port 1), K5: COM5(upper extension port 2).
- When X0 is ON, REGW instruction is executed, Modbus read write flag SM160(serial port2) is set ON, SM160 is set OFF when the execution is completed. If a communication error occurs and the number of resend is set, it will be automatically resend. Users can check the relevant registers to determine the cause of the error. The execution result of Modbus read and write instructions of serial port 2 is in SD160.

Multiple registers write [MRGW]

1)Summary

Write local device multiple registers to remote station no's registers.

Multi-register write [MRGW]			
16 bits instruction	MRGW	32 bits instruction	-
Execution Condition	Normally ON/OFF, edge triggering	Suitable models	XD, XL
Hardware Requirement	-	Software Requirement	-

2)Operands

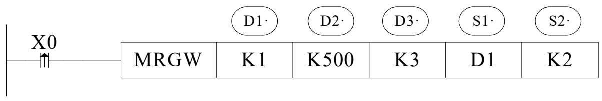
Operands	Function	Type
D1	Specify remote communication station number	16 bits，BIN
D2	Specify remote register start address	16 bits，BIN
D3	Specify register number	16 bits，BIN
S1	Specify start address of local registers	16 bits，BIN
S2	Specify serial port number	16 bits，BIN

3)Suitable soft components

Operands	Word soft elements											Bit soft elements						
	System								Constant	Module		System						
	D	FD	TD	CD	DX	DY	DM	DS	K/H	ID	QD	X	Y	M	S	T	C	Dn.m
D1	●	●	●	●					●									
D2	●	●	●	●					●									
D3	●	●	●	●					●									
S1	●																	
S2									K									

*Notes: D includes D, HD; TD includes TD, HTD; CD includes CD, HCD, HSCD, HSD; DM includes DM, DHM; DS includes DS, DHS. M includes M,HM,SM； S includes S,HS； T includes T,HT； C includes C, HC.

Function



- Write multiple registers, Modbus function code is 10H.
- Serial port: K0~K5. K0: COM0 (RS232), K1: COM1(RS232), K2: COM2(RS485), K3: COM3(left extension port), K4: COM4(upper extension port 1), K5: COM5(upper extension port 2).
- Operand D3: the max register number is 123.
- When X0 is ON, MRGW instruction is executed, Modbus read write flag SM160(serial port2) is set ON, SM160 is set OFF when the execution is completed. If a communication error occurs and the number of resend is set, it will be automatically resend. Users can check the relevant registers to determine the cause of the error. The execution result of Modbus read and write instructions of serial port 2 is in SD160.

6-2-6 Modbus serial port configuration

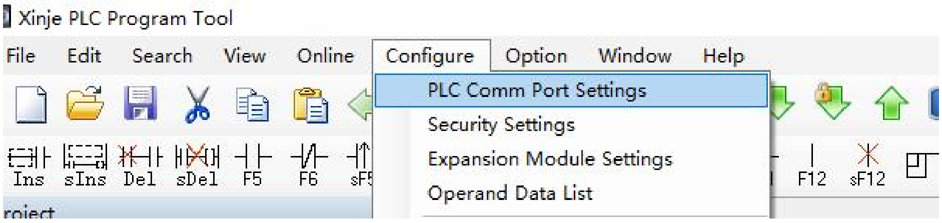
There are two ways to set Modbus communication parameters: 1. set parameters by programming software; 2. set parameters by XINJEConfig tool;

1. Set parameters by programming software

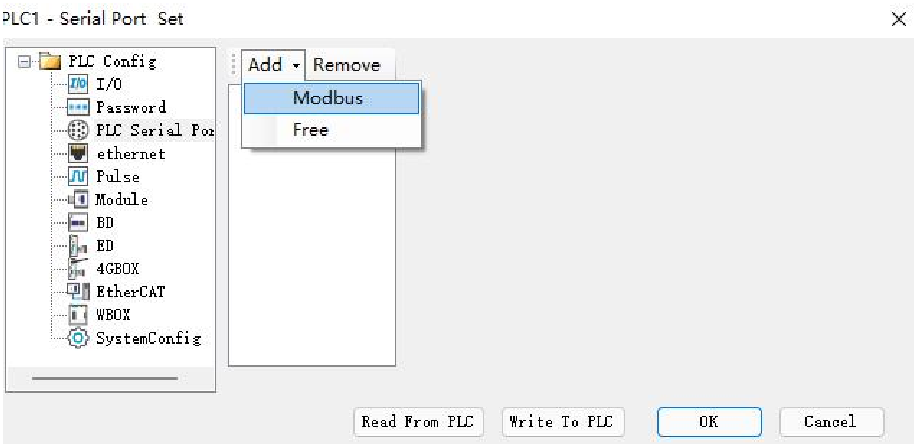
When using programming software to configure the parameters of PLC serial port, the version below V3.4 must use XNET communication mode, and the version above V3.4 can also use Modbus communication mode (RS232 port).

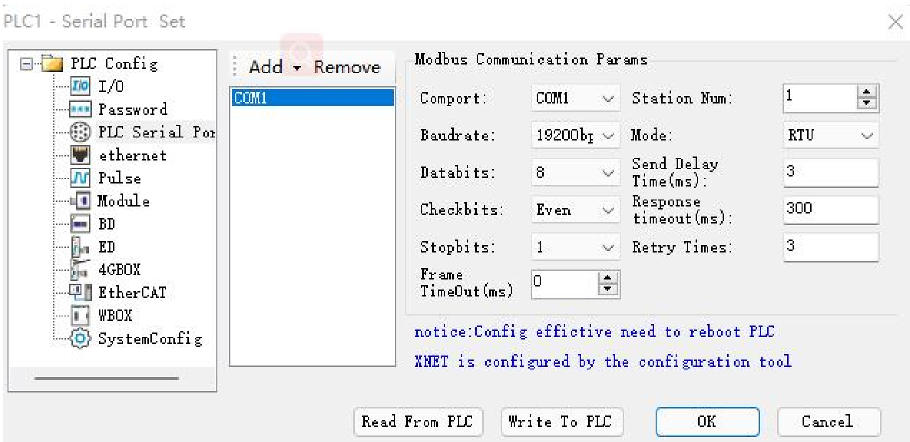
(1) Use the USB download cable to connect the PLC with the computer. Here the USB download cable is the HMI download cable, as shown below, the software must switch to XNet communication mode.

(2) Open the programming software, click configure/PLC com port settings. It will show below figure:



(3) Click add, it will show two modes, modbus mode and free mode, please select modbus mode, it will show below figure.





Port No.: It refers to Port of PLC, COM1 refers to COM1 (RS232), COM2 refers to COM2 (RS485) or COM2-RS232 (RS485) or COM2-RS485 (RS485), COM3 refers to COM3 (left extended ED port), COM4 refers to COM4 (upper extended BD port 1), COM5 refers to COM5 (upper extended BD port 2).

The baud rate, data bit, parity bit, stop bit should be same to the communication device.

Station number: if the PLC is master, the station no. is defaulted 1, if the PLC is slave, it needs to set different station no.
Two communication modes: RTU, ASCII.
Delay before sending: Waiting time before PLC sends data. In the original XC series PLC, if the master PLC communicates with the slave PLC, the master PLC sends data to the slave PLC. If the master PLC sends data to the slave PLC after the first time, and the slave PLC has not yet had time to receive the data, then the master PLC sends data to the slave PLC again, which easily leads to the error of the slave PLC; In XD series PLC, it has send delay to solve the problem. That is, after receiving data from the slave station, it must delay a certain time to receive the next communication data, so as not to cause the above problems.
Reply overtime (ms): it refers to the time when the PLC can not receive the response after sending the request andwait for sending again.
Retry times: It refers to the number of times that the PLC can not receive the reply, and each reply needs a reply timeout time.

(4) After setting, click write to PLC, then cut off the PLC power supply and power on again to make the settings effective.

Note: V3.4 version and earlier of the XD series of PLC download and upload serial configuration data must use XNET communication mode, that is, using USB port to download and upload configuration data. If the following prompt appears, you need to check whether the serial port parameters you configured are downloaded from the USB port to the PLC.

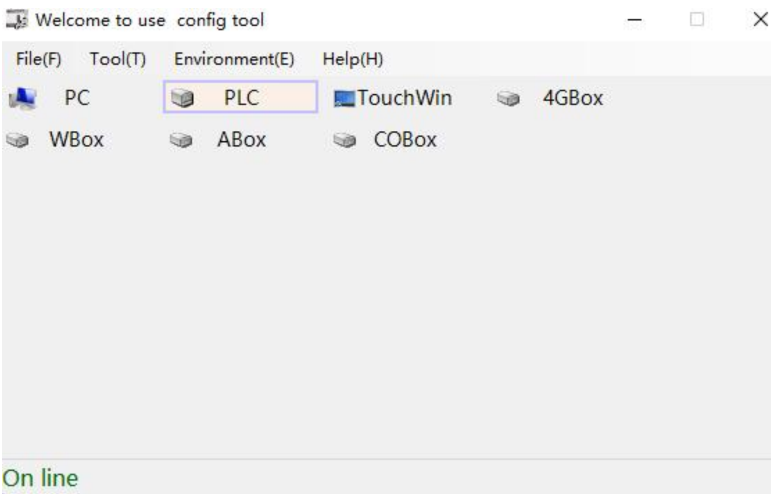
Note: Versions V3.4 and above can be configured in Modbus communication mode (RS232 port); Versions V3.4 and below XD series PLC must use X-NET communication mode when downloading and uloading serial configuration data, that is, downloading and uploading configuration data through USB port.

2. Set the parameters by using XINJEConfig tool

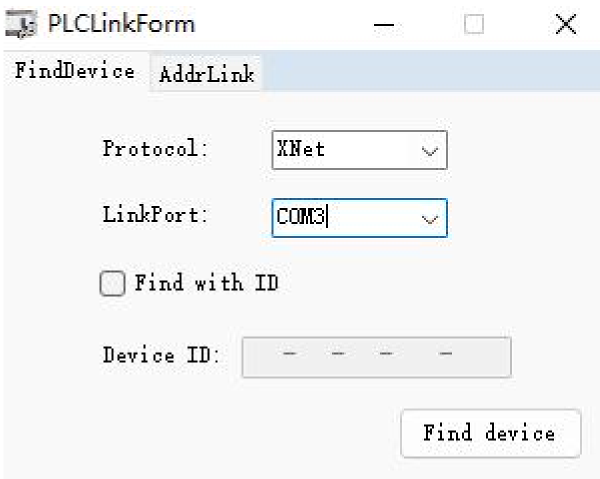
When using configuration tool XINJEConfig to configure parameters of PLC serial port, the XINJEConfig tools of V1.6.308 and below must use USB port. The XINJEConfig tool for V1.6.309 and above can also be configured using RS232 port.

(1) Use the USB download cable to connect the PLC with the computer. Here the USB download cable is the HMI download cable, as shown below.

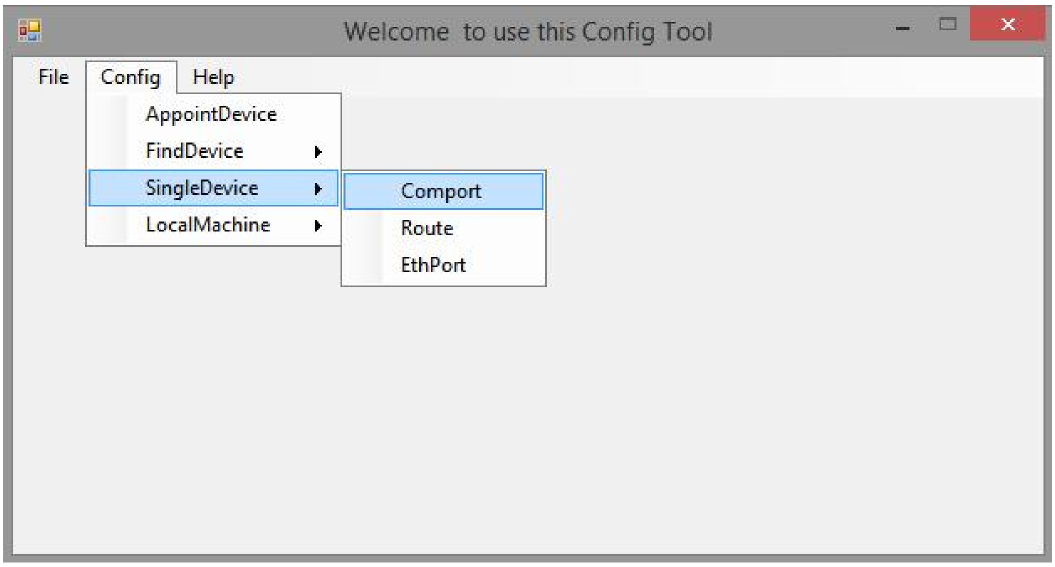
(2) Open xinjeconfig tool, click PLC



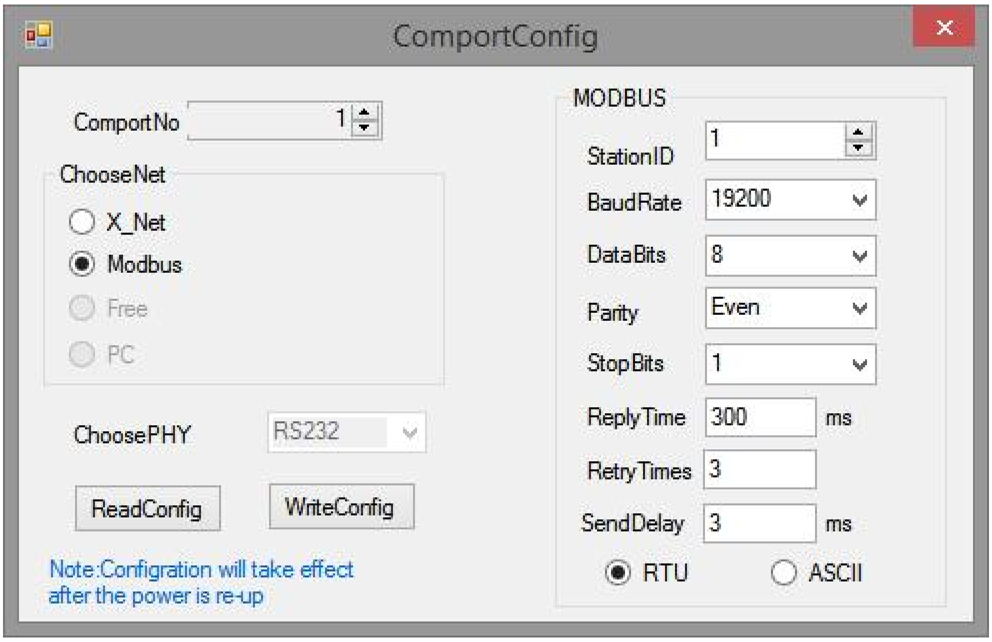
(3) Click config/find device:



(4) Choose the com port connecting PC and PLC, click ok. Click config/single device/comport.

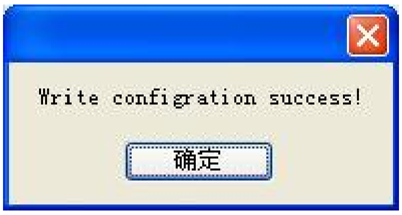


(5) It will show below window.



Serial port: K0 ~ K5. Port0 (RS232), Port1 (RS232), Port2 (RS485) or Port2-RS232 (RS232) or Port2-RS485 (RS485), Port3 (left extension port), Port4 (upper extension port 1), Port5 (upper extension port 2).
Here, we can set the communication mode and parameters of each communication port.

(6) When the com port parameters setting is completed, click writeconfig. It will show "write configuration success" message.

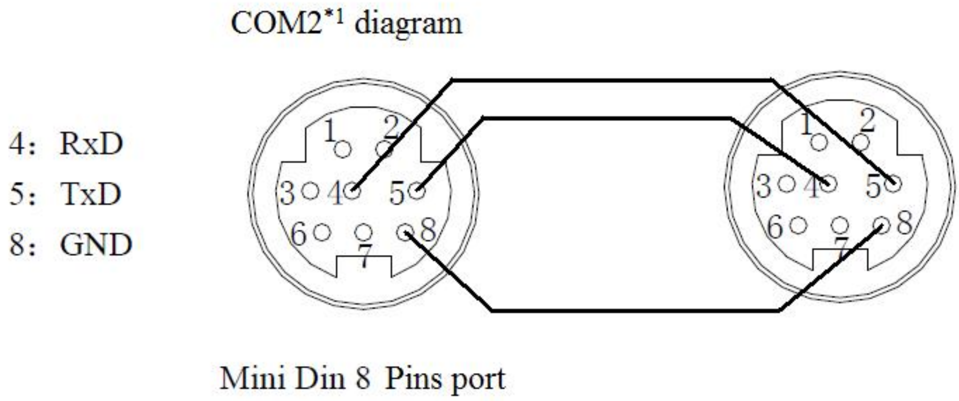


(7) Close XINJEConfig tool, cut the PLC power and power on again to make the settings effective.

6-2-7 Modbus Communication application

There are two wiring methods:

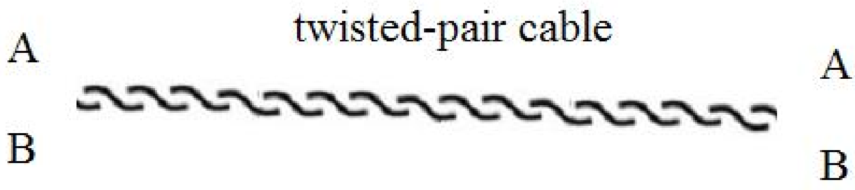
232 wiring methods



Note:

1. COM2 with *1 only show the RS232 pins.
2. XD/XL series PLC， RS232 do not support full-duplex, so it can only communicate in single direction.
3. RS232 communication distance is short (about 13m); RS485 is suitable for longer distance.

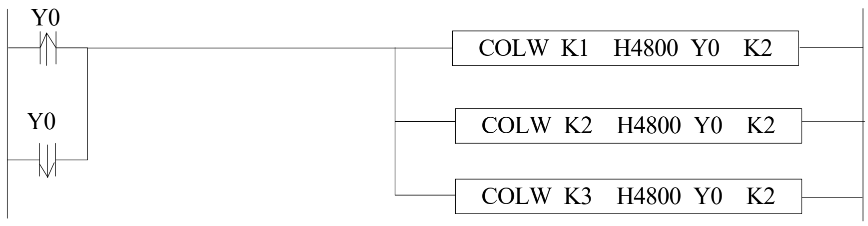
485 wiring methods



Connect all A terminals, connect all B terminals. A is RS485+, B is RS485-.

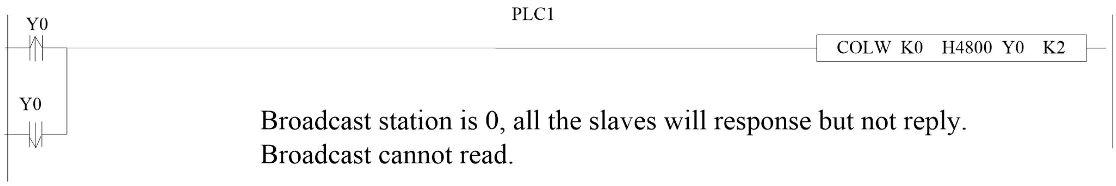
Application：One xinje XD3 series PLC controls 3 XC series PLCs, slave PLCs follow the master's action. (Master PLC Y0 ON, then slave PLC Y0 ON; Master PLC Y0 OFF, then slave PLC Y0 OFF) Precondition: on-off of Y0 makes communication have enough time to react. Also three slave PLCs can be not that synchronous (not fully synchronous).

Method 1 usual program



The program takes serial port 2 as example, so corresponding communication flag is the serial port 2's. About other serial port, please refer to appendix 1. Serial port, please refer to appendix 1.

Method 2 use broadcasting function：



When master Y0 status changes, it broadcasts the status to all the slaves. The synchronization of three PLCs is better than method 1.

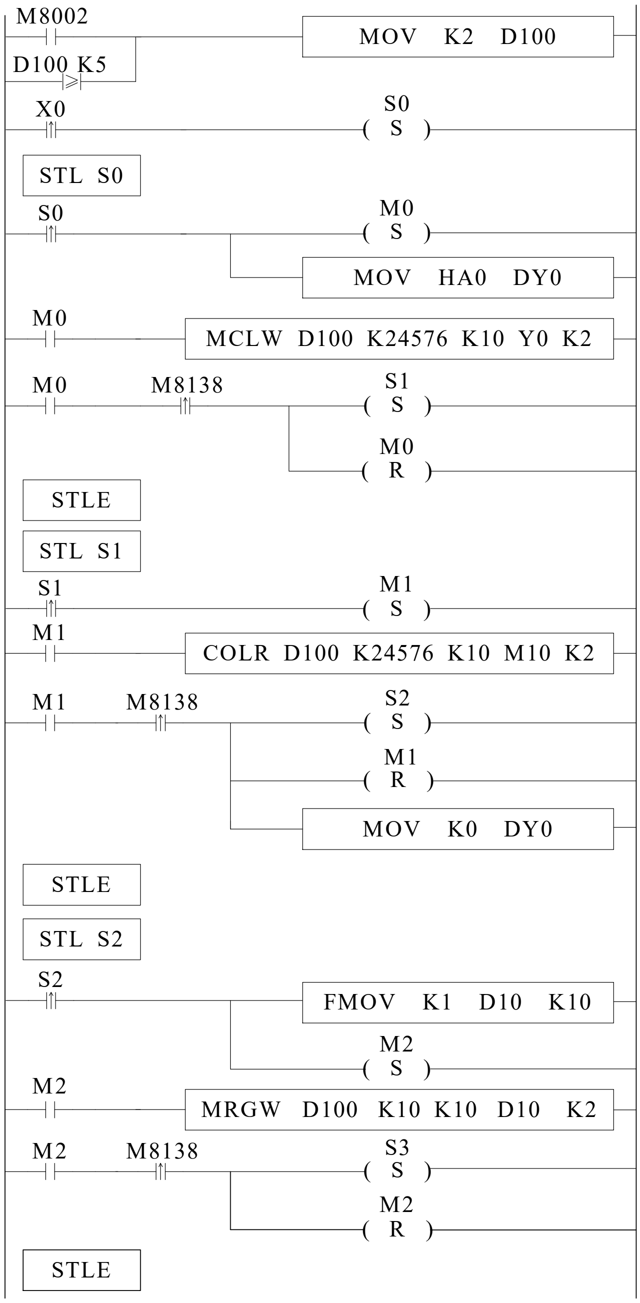
6-2-8 Application

Example 1:
Following are the programs for reading and writing Modbus communication between 1 master station and 3 slave stations.

Program operation:

(1) Write master PLC Y0~Y11 status to slave PLC 2 Y0~Y11
(2) Read slave PLC 2 Y0~Y11 to master PLC M10~M19
(3) Write master PLC D10~D19 to slave PLC 2 D10~D19
(4) Read slave PLC 2 D10~D19 to master PLC D20~D29
(5) So as slave PLC 3 and 4

The following is a comparison of XC and XD series Modbus-RTU communication programs for reference. The communication programs in XC series are as follows:

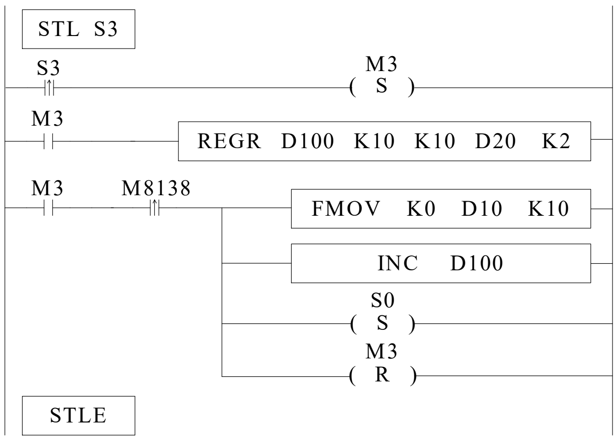


//send station no.2 to D100, execute the process S0

//set ONY0~Y11 of master station, write the master status to Y0~Y11 of slave PLC 2, 3, 4. Enter process S1 when the communication succeeded.

//read the Y0~Y11 of slave PLC 2, 3, 4 to master PLC M10~M19. Reset master PLC Y0~Y11 and enter process S2 after the communication is successful.

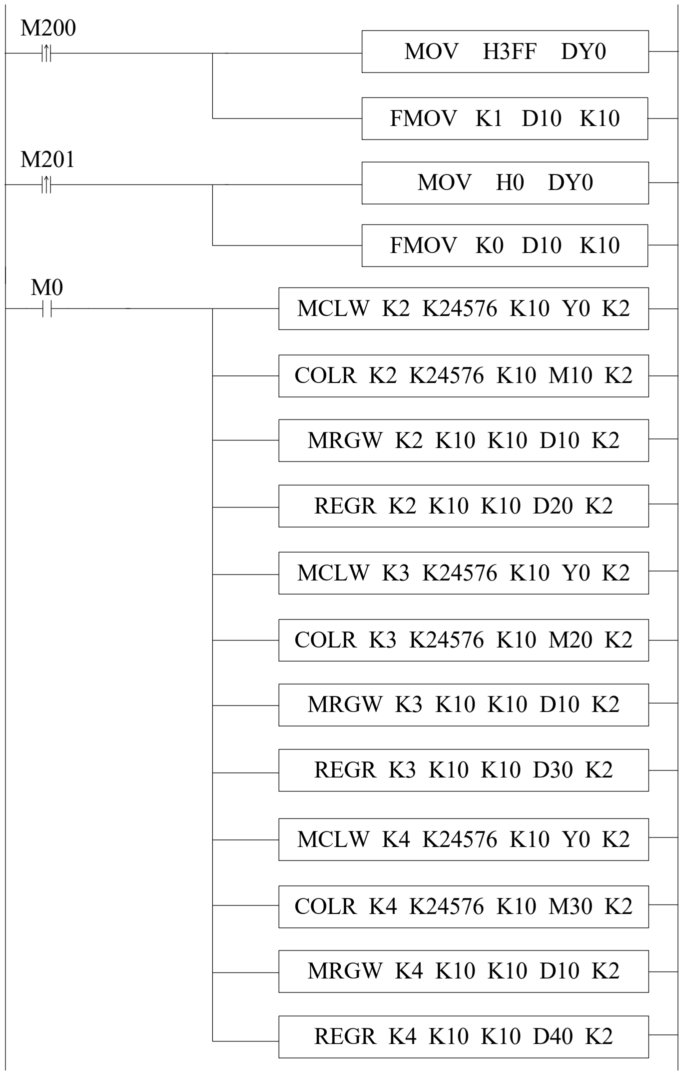
//write 1 to master PLC D10~D19, write the master PLC D10~D19 to D10~D19 of slave PLC 2, 3, 4. Enter process S3 when the communication is successful.



//read the D10~D19 of slave PLC 2, 3, 4 to master PLC D20~D29, reset D10~D19 after the communication is successful, then the station no. is added 1, process S0 is executed, cycle.

Modbus-RTU instruction processing mode has changed. Users can write Modbus-RTU instructions directly in user programs. Protocol stack will queue Modbus-RTU communication requests. Communication is another task. In the main program, users can write multiple Modbus-RTU communication instructions together and trigger them at the same time through the same triggering condition. PLC will trigger these communications. Instructions are queued according to the protocol station by Modbus-RTU, which will not cause communication errors when multiple communication instructions are executed at the same time as the original XC series PLC.

XD series program:



//at the rising edge of M200, set ON the master PLC Y0~Y11, D10~D19 are set to 1, at the rising edge of M201, set OFF Y0~Y11 of master PLC, reset D10~D19.

//write the Y0~Y11 of master PLC to Y0~Y11 of slave PLC 2, read the Y0~Y11 of slave PLC 2 to M10~M19 of master PLC. Write the D10~D19 of master PLC to D10~D19 of slave PLC 2. Read the D20~D29 of slave PLC 2 to D20~D29 of master PLC.

6-3 Free communication

6-3-1 Free communication mode

Free format communication is data transmission in the form of data blocks, limited by the PLC cache, the maximum amount of data sent each time is 256 bytes.

The so-called free communication, i.e. custom protocol communication, now many intelligent devices on the market support RS232 or RS485 communication, but the protocols used by various products are different, such as: Xinje PLC uses standard Modbus-RTU protocol, some temperature controller manufacturers use custom protocols; if using Xinje PLC to communicate with temperature controller, it is necessary to use free communication to send data in full accordance with the protocol of the instrument manufacturer, so as to communicate.

Prerequisites for free communication:

1. COM0(RS232), COM1(RS232), COM2(RS485) or COM2-RS232(RS232) or COM2-RS485(RS485), COM3(left extension port), COM4(upper extension port 1), COM5(upper extension port 2) all support free communication. As the free communication needs to change the communication parameters, COM1 is not recommended.
2. Baud rate: 300bps~3Mbps, 4.5Mbps~9Mbps (special model supported)
3. The data format must be the same as the lower device settings. There are several options as follows:
   Data bit: 5 bits (special model supported), 6 bits (special model supported), 7 bits, 8 bits, 9 bits.
   Parity bit: none, odd parity, even parity, empty, mask
   Stop bit: 1 bit, 1.5 bit, 2 bits
4. Starter: 1 byte, terminator: 1 byte
   Users can set a start/termination character. After setting the start/termination character, PLC automatically adds the start/termination character when sending data, and automatically removes the start/termination character when receiving data.
   In fact, the initiator and terminator can be regarded as the data frame head and end in the protocol. Therefore, if the lower device communication has start and termination character, it can be set in the software or written in the protocol.
5. Communication mode: 8 bits,16 bits
   When 8-bit buffer is selected for communication, the high bytes of registers are invalid. PLC only uses the low bytes of registers to send and receive data.
   When 16-bit buffer is selected for communication, the PLC will send all the data of the register, and send low-byte data first, then high-byte data.
   When it is necessary to transfer low bytes and high bytes of one 16-bit register to another 16-bit register, 16-bit buffers must be selected for communication, and the number of communication bytes is 2. When the value stored in a 16-bit register occupies only low bytes, we can choose 8-bit buffer to communicate. The number of communication bytes is 1. Usually when we communicate, the data will not exceed

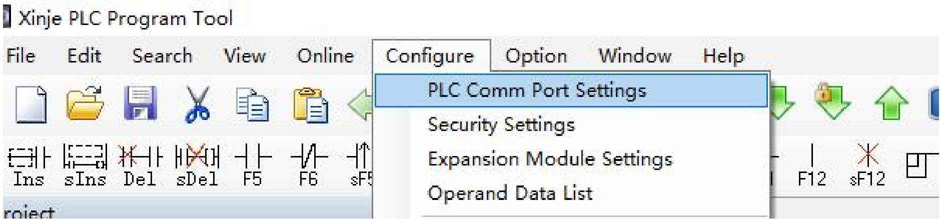
the low byte of a register (HFF), so we only need to use the default 8-bit buffer in the software to communicate.

6. Timeout: frame timeout (ms), reply timeout (ms)
   Frame: A data string.
   Frame timeout: refers to the time interval between two frames of data received by the PLC, which ensures that the PLC can distinguish the end time of receiving a frame. It is usually used to judge whether a frame of data in PLC has been received or not. When the interval between two frames of data is longer than the frame time-out, it means the end of one frame of communication data.
   Reply timeout: refers to the time when the PLC can not receive the response after sending the request, waiting for the resend. If the response time is set to exceed 300 ms, when default communicating, the PLC waits 300ms for the other party to respond. If the response time is not received, the request will be sent again.
   If you want to shorten the communication time, you can adjust the above two parameters according to the size of baud rate.

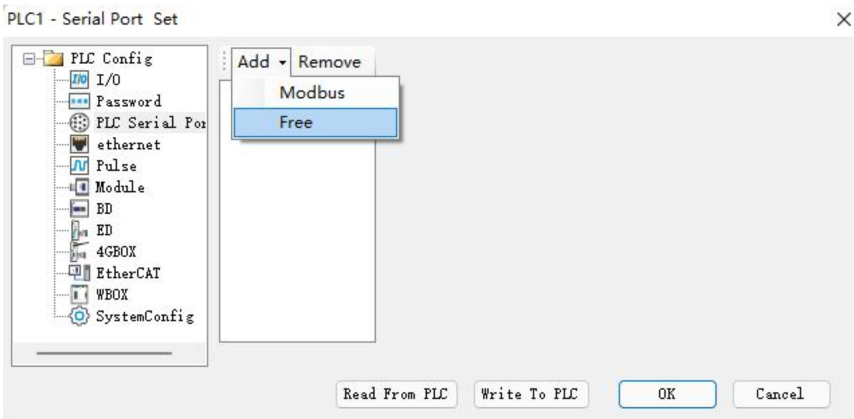
6-3-2 Serial port configuration

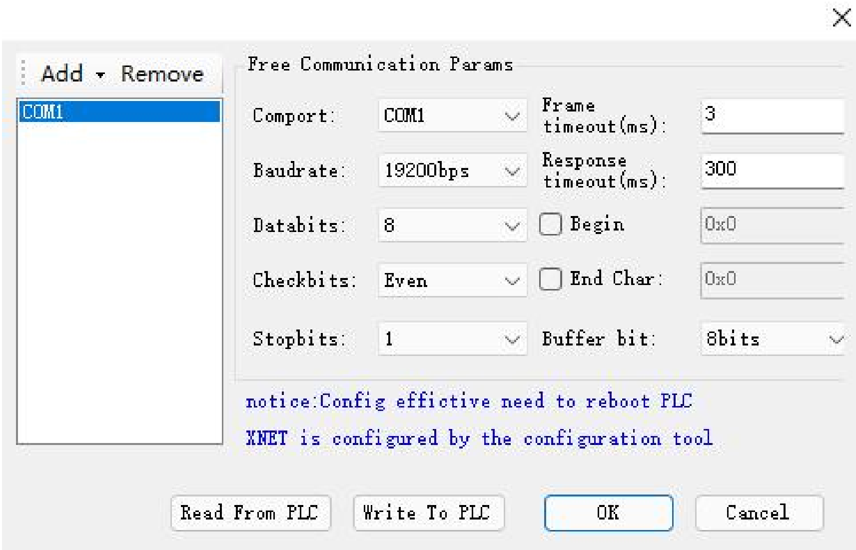
(1)Use the USB download cable to connect the PLC with the computer. Here the USB download cable is the HMI download cable, as shown below, the software must switch to XNet communication mode.

(2)Open the programming software, click configure/PLC comm port settings. It will show below figure:



(3)Click add, it will show two modes, modbus mode and free mode, please select free mode, it will show below figure.





Port No.: It refers to Port of PLC, COM1 refers to COM1 (RS232), COM2 refers to COM2 (RS485) or COM2-RS232 (RS485) or COM2-RS485 (RS485), COM3 refers to COM3 (left extended ED port), COM4 refers to COM4 (upper extended BD port 1), COM5 refers to COM5 (upper extended BD port 2).
Frame timeout (ms): It refers to the time interval between two frames of data sent by PLC, which ensures that the receiver distinguishes the end time of receiving a frame.
Response timeout (ms): refers to the time when the PLC can not receive the response after sending the request, waiting for the resend.
Other serial parameters can be set according to the parameters of the lower device.

(4)After setting, click write to PLC, then cut off the PLC power supply and power on again to make the settings effective.

Note: Versions V3.4 and above can be configured in Modbus communication mode (RS232 port); Versions V3.4 and below XD series PLC must use X-NET communication mode when downloading and uloading serial configuration data, that is, downloading and uploading configuration data through USB port.

6-3-3 Suitable occasion

When does free communication need to be used?
As an example, the situation described in the above section is that XINJE PLC communicates with the temperature control instrument, and the instrument uses its own communication protocol, which stipulates that the reading temperature should be sent four characters: "R", "T", "CR". Each character has the following meanings:

Character	Meaning
:	Data start
R	Read
T	temperature
CR	Enter, data end

PLC needs to send the ASCII code of the above characters to the instrument in order to read the current temperature value measured by the instrument. The ASCII code values (hexadecimal) of each character can be obtained by querying the ASCII code table.

Character	ASCII code value
:	3A
R	52
T	54
CR	0D

Obviously, according to the situation described above, using MODBUS instructions can not communicate, at this time you need to use free communication. Detailed usage will be used as an example to program the sample program in later chapters.

6-3-4 Free communication instruction

Send data [SEND]

1)Summary

Write the local data to specified remote station address.

Send data [SEND]			
16 bits instruction	SEND	32 bits instruction	-
Execution condition	Normally ON/OFF, rising edge triggering	Suitable model	XD, XL
Hardware	V3.2.3 and higher version	Software	V3.2.2 and higher version

2)Operand

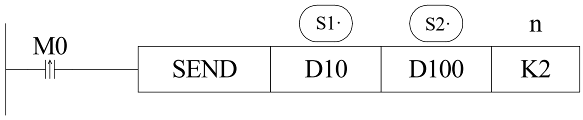
Operands	Function	Type
S1	Local data starting address	16 bits, BIN
S2	Send byte number	16 bits, BIN
n	Communication port no.	16 bits, BIN

3) Suitable soft component

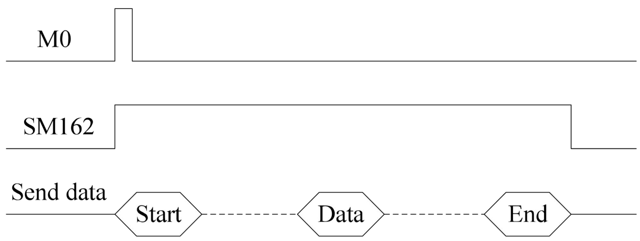
Operands	Word soft elements											Bit soft elements						
	System								Constant	Module		System						
	D	FD	TD	CD	DX	DY	DM	DS	K/H	ID	QD	X	Y	M	S	T	C	Dn.m
S1	●	●	●	●														
S2	●	●	●	●					●									
n	●								K									

*Notes: D includes D, HD; TD includes TD, HTD; CD includes CD, HCD, HSCD, HSD; DM includes DM, DHM; DS includes DS, DHS. M includes M, HM, SM; S includes S, HS; T includes T, HT; C includes C, HC.

Function and action



- Data sending instructions, M0's rising edge sends data once.
- Communication port. Range: K0 ~ K5. K0: COM0 (RS232), K1: COM1(RS232), K2: COM2(RS485), K3: COM3(left extension port), K4: COM4(upper extension port 1), K5: COM5(upper extension port 2).
- In the process of data transmission, the "sending" flag SM162 (communication port 2) is set on.



- When the buffer number is 8 bits, only low-byte data is sent, so D100 = the number of registers sent, for example, to send low-byte data in D10-D17, D100 should be set to 8.
- When the buffer number is16 bits, high and low byte data will be sent, so D100 = the number of registers sent * 2. For example, when sending high and low byte data in D10-D17, D100 should be set to 16, and when sending, low byte will be before the high byte.

Receive data [RCV]

1) Summary

Write the specified remote station no's data to local device.

Send data [RCV]			
16 bits instruction	RCV	32 bits instruction	-
Execution condition	Normally ON/OFF, rising edge triggering	Suitable model	XD, XL
Hardware	V3.2.3 and higher version	Software	V3.2.2 and higher version

2)Operands

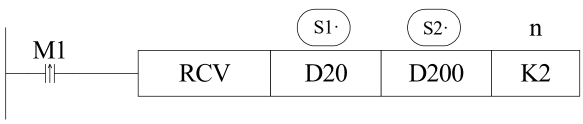
Operands	Function	Type
S1	Local data starting address	16 bits, BIN
S2	Receivebyte number or soft component address	16 bits, BIN
n	Communication port no.	16 bits, BIN

3)Suitable soft component

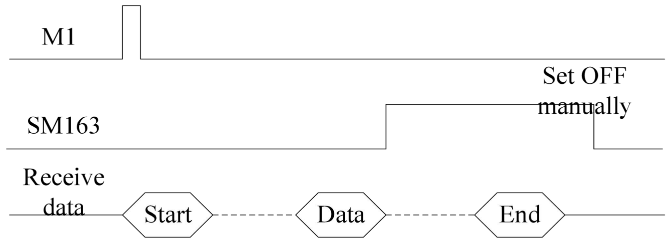
Operands	Word soft elements											Bit soft elements						
	System								Constant	Module		System						
	D	FD	TD	CD	DX	DY	DM	DS	K/H	ID	QD	X	Y	M	S	T	C	Dn.m
S1	●	●	●	●														
S2	●	●	●	●					●									
n	●								K									

*Notes: D includes D, HD; TD includes TD, HTD; CD includes CD, HCD, HSCD, HSD; DM includes DM, DHM; DS includes DS, DHS. M includes M, HM, SM; S includes S, HS; T includes T, HT; C includes C, HC.

Function and action



- Data receiving instructions, M1's rising edge receives data once.
- Communication port. Range: K0 ~ K5. K0: COM0 (RS232), K1: COM1(RS232), K2: COM2(RS485), K3: COM3(left extension port), K4: COM4(upper extension port 1), K5: COM5(upper extension port 2).
- After receiving the data, the "received" flag SM163 (communication port 2) is set on.



- When the buffer number is 8 bits, the received data is only stored in low bytes, so D200 = the number of bytes to be received * 2, for example, to receive 8 bytes of data, stored in the low bytes of the eight registers D20-D27 in turn, at this time, D200 should be set to 8.
- When the buffer number is 16 bits, the received data is stored in a complete register, so D200 = the number of bytes to be received, for example, to receive 16 bytes of data, stored in the four registers of D100-D107 in turn, at this time, D200 should be set to 16. And when receiving, low bytes are before high bytes.

Release serial port[RCVST]

1)Summary

Release the specified serial port.

Release serial port[RCVST]			
16 bits instruction	RCVST	32 bits instruction	-
Execution condition	Normally ON/OFF, rising edge triggering	Suitable model	XD, XL
Hardware	V3.2.3 and higher version	Software	V3.2.2 and higher version

2)Operand

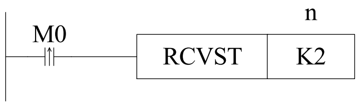
Operand	Function	Type
n	Communication port no.	16 bits, BIN

3)Suitable soft component

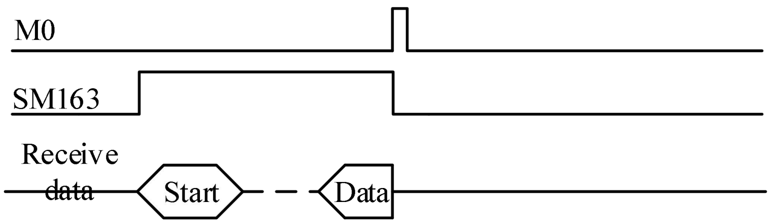
Operands	Word soft elements											Bit soft elements						
	System								Constant	Module		System						
	D	FD	TD	CD	DX	DY	DM	DS	K/H	ID	QD	X	Y	M	S	T	C	Dn.m
n	●								K									

*Notes: D includes D, HD; TD includes TD, HTD; CD includes CD, HCD, HSCD, HSD; DM includes DM, DHM; DS includes DS, DHS. M includes M, HM, SM; S includes S, HS; T includes T, HT; C includes C, HC.

Function and action



- Release serial port instructions, M0's rising edge execute once.
- Communication port. Range: K0 ~ K5. K0: COM0 (RS232), K1: COM1(RS232), K2: COM2(RS485), K3: COM3(left extension port), K4: COM4(upper extension port 1), K5: COM5(upper extension port 2).
- When releasing the serial port, the "received" flag SM163(communication port 2) is set OFF.
- For free communication, if there is no timeout or the timeout time is set too long, the occupied serial port resources can be released immediately through RCVST instructions for other communication operations.



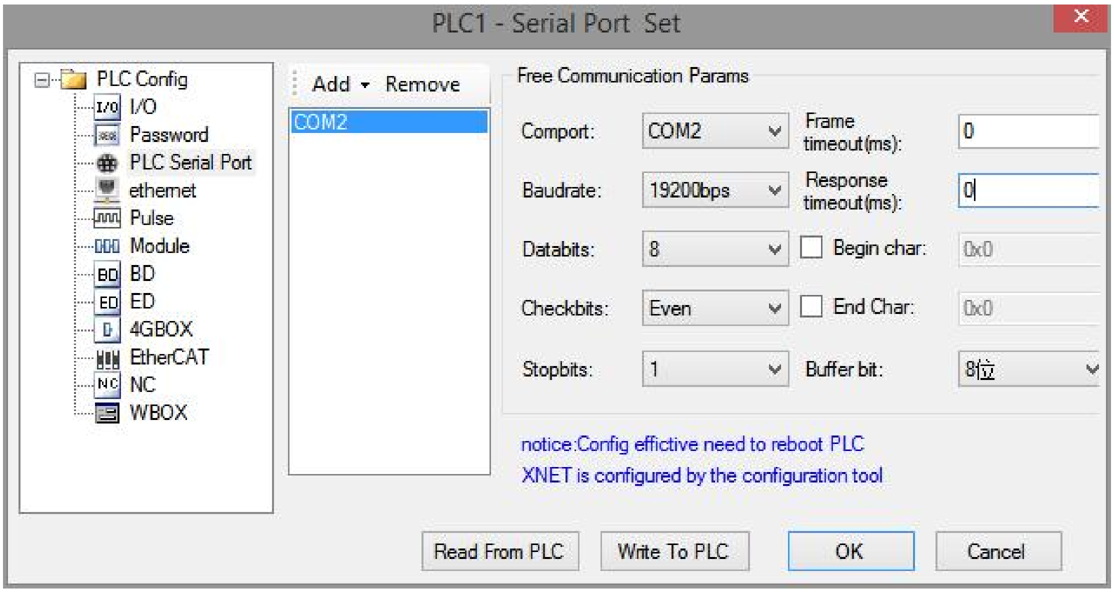
6-3-5 Free communication example

Example 1: In chapter 6-3-3, we give an example of communication between Xinje PLC and temperature control instrument when explaining why to use free communication. Here is an example.

Operation steps:

1. Connect the hardware first. Here we use the serial port 2 of the PLC to communicate, that is, 485 + on the instrument is connected to A of the output port of the PLC, and 485- on the instrument is connected to B of the output port of the PLC.

2. Set the serial port parameters of PLC according to the communication parameters of temperature control instrument. The parameters are set as follows. After setting the parameters, the power can be restarted.



3. make the program according to the descriptions in chapter 6-3-3.

Read temperature: " : ""R""T""CR"

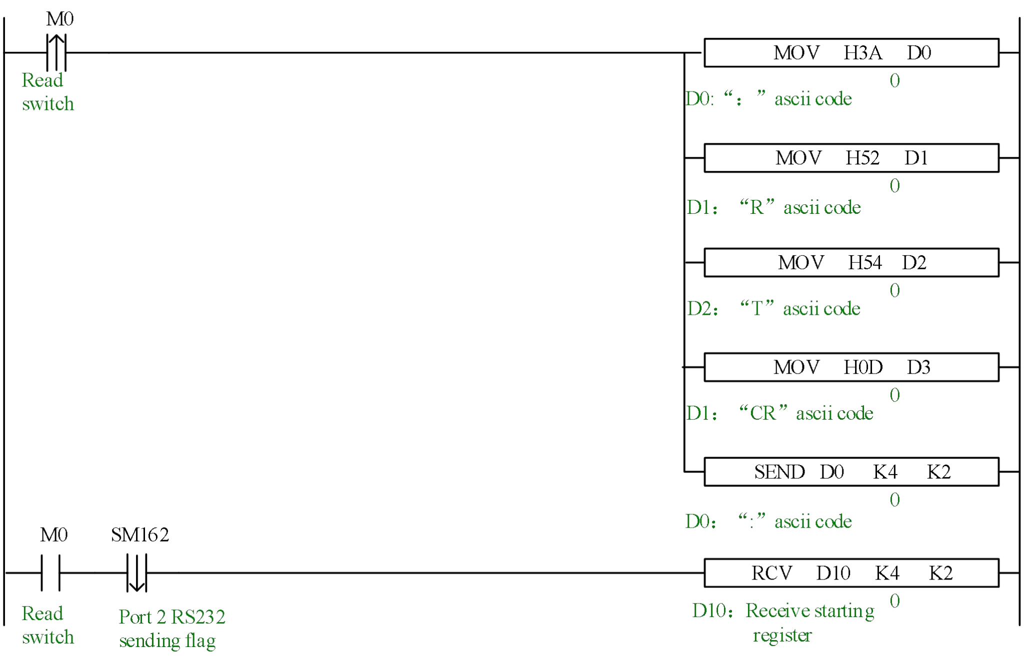
" : " -------------- data start

"R" --------------- read

"T" --------------- temperature

"CR" ------------- enter, data end

Program:



When trying to communicate between PLC and other intelligent devices, it is suggested to use serial debugging tool to determine the data format of communication, that is, protocol. The advantages of this method are: the serial debugging tool is easy to modify and flexible to use; after the serial debugging tool determines that communication can be successful, the PLC program is written according to the data format obtained, which is often twice the result with half the effort.

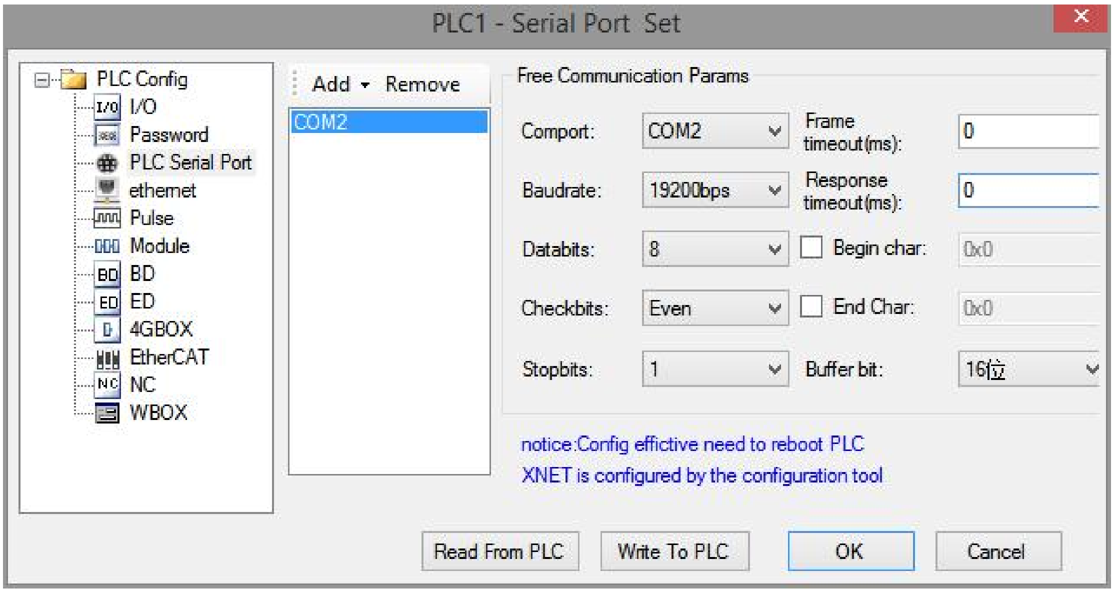
In fact, Modbus-RTU protocol can be regarded as a special kind of free protocol. The relationship between them is similar to ellipse and circle. We can try to use free format to realize the function of Modbus instruction.

Example 2: The values of the five registers of a XD3 PLC are sent to the D1-D5 of another XDM PLC.

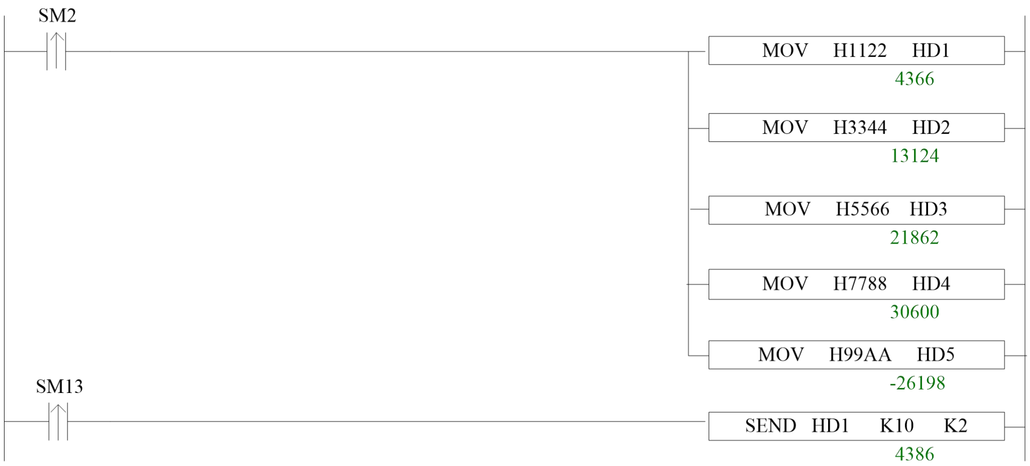
If the user understands the Modbus communication, he can use the Modbus-RTU communication mode to do so, as long as he writes a "write multiple register instructions (MRGW)" in the host. Here we do it in free communication mode.

Operation steps:

1. Connect the hardware first. Here we use the serial port 2 of the PLC to communicate, that is, connect A of the two PLC, and connect B of the two PLC.
2. Set the same serial port parameters of the two PLC. The parameters are set as follows. After setting the parameters, the power can be restarted.



3. XD3 program:



XDM program:

Sometimes the data of user communication is stored in multiple registers in the form of ASCII code. Users need to take this value out, store it in a register and display it on the HMI. Customers often consider using HEX (ASCII to hexadecimal) instructions to achieve it. But HEX instructions are difficult to use and understand. Often, we will not use this instruction to complete it. The relationship between values can be found by ASCII code comparison table.

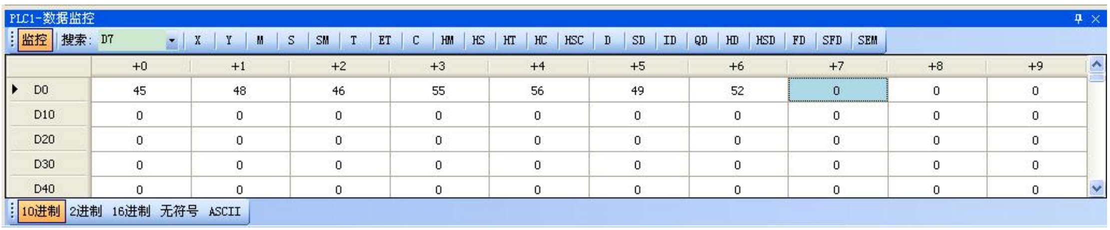
ASCII code table:

ASCII value	Control character	ASCII value	Control character	ASCII value	Control character	ASCII value	Control character
0	NUT	32	(space)	64	@	96	`
1	SOH	33	!	65	A	97	a
2	STX	34	”	66	B	98	b
3	ETX	35	#	67	C	99	c
4	EOT	36	$	68	D	100	d
5	ENQ	37	%	69	E	101	e
6	ACK	38	&	70	F	102	f
7	BEL	39	’	71	G	103	g
8	BS	40	(	72	H	104	h
9	HT	41	)	73	I	105	i
10	LF	42	*	74	J	106	j
11	VT	43	+	75	K	107	k
12	FF	44	,	76	L	108	l
13	CR	45	-	77	M	109	m
14	SO	46	。	78	N	110	n
15	SI	47	/	79	O	111	o
16	DLE	48	0	80	P	112	p
17	DC1	49	1	81	Q	113	q
18	DC2	50	2	82	R	114	r
19	DC3	51	3	83	S	115	s
20	DC4	52	4	84	T	116	t
21	NAK	53	5	85	U	117	u
22	SYN	54	6	86	V	118	v
23	TB	55	7	87	W	119	w
24	CAN	56	8	88	X	120	x
25	EM	57	9	89	Y	121	y
26	SUB	58	:	90	Z	122	z
27	ESC	59	;	91	[	123	{
28	FS	60	<	92	\	124	|
29	GS	61	=	93	]	125	}
30	RS	62	>	94	^	126	~
31	US	63	?	95	—	127	DEL

Example 3: A pressure controller communicates with PLC in free communication mode to realize data acquisition. The value displayed on the pressure controller is -0.7814 MPa. The value collected by PLC is stored from D0, and seven registers are stored in turn. However, the value of the seven registers combination needs to be taken out and stored in D46 in the form of decimal.

Through the data monitoring of PLC, ASCII codes in D0~D6 registers can be monitored as follows:

Switch to decimal format and show as below:

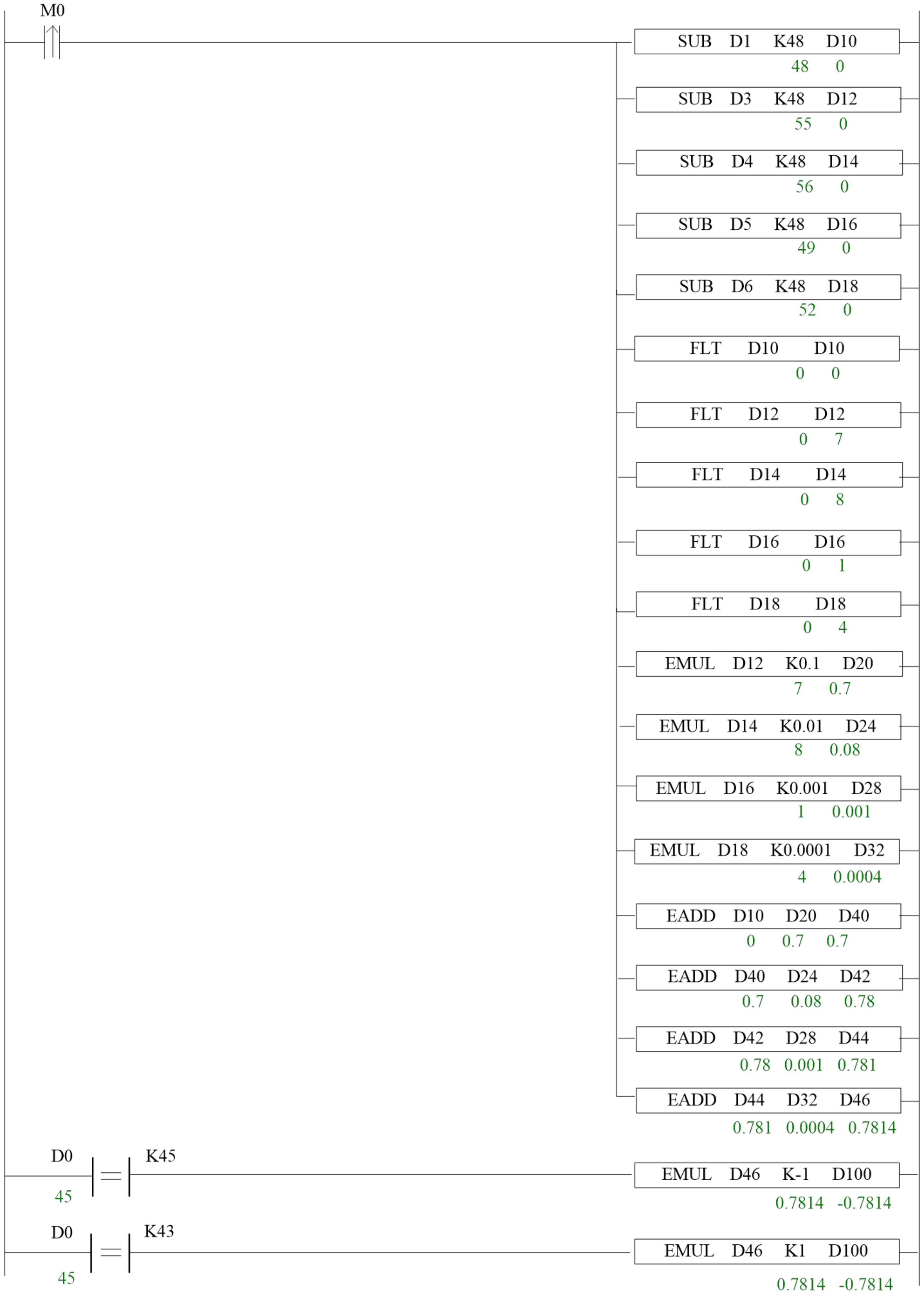


By comparing the relationship between ASCII codes and decimal values, we can find the rule that there is 48 difference between ASCII codes in D1, D3, D4, D5, D6 and decimal values. The final decimal values are obtained by subtracting the values in registers by K48 and multiplying by 10. The formula is as follows:

D46=(D1-48)*1+(D3-48)*0.1+(D4-48)*0.01+(D5-48)*0.001+(D6-48)*0.0001

D0 is a symbol bit. Looking up the table, we know that when D0 = K45, it represents a negative value; when D0 = K43, it represents a positive value.

The ladder diagram is as follows:


M0
SUB D1 K48 D10
48 0
SUB D3 K48 D12
55 0
SUB D4 K48 D14
56 0
SUB D5 K48 D16
49 0
SUB D6 K48 D18
52 0
FLT D10 D10
0 0
FLT D12 D12
0 7
FLT D14 D14
0 8
FLT D16 D16
0 1
FLT D18 D18
0 4
EMUL D12 K0.1 D20
7 0.7
EMUL D14 K0.01 D24
8 0.08
EMUL D16 K0.001 D28
1 0.001
EMUL D18 K0.0001 D32
4 0.0004
EADD D10 D20 D40
0 0.7 0.7
EADD D40 D24 D42
0.7 0.08 0.78
EADD D42 D28 D44
0.78 0.001 0.781
EADD D44 D32 D46
0.781 0.0004 0.7814
D0 = K45
45
EMUL D46 K-1 D100
0.7814 -0.7814
D0 = K43
45
EMUL D46 K1 D100
0.7814 -0.7814

6-4 Communication flag and register

Communication flag

Serial port	Register address	Function	Explanation
COM0	SM140	Modbusread-write instruction execution flag	When the instruction starts to execute, set ON When execution is completed, set OFF
	SM142	Free communication sending flag	When the instruction starts to execute, set ON When execution is completed, set OFF
	SM143	Free communication received flag	When receiving a frame of data or receiving data timeout, set ON. Require user program to set OFF
COM1	SM150	Modbusread-write instruction execution flag	When the instruction starts to execute, set ON When execution is completed, set OFF
	SM152	Free communication sending flag	When the instruction starts to execute, set ON When execution is completed, set OFF
	SM153	Free communication received flag	When receiving a frame of data or receiving data timeout, set ON. Require user program to set OFF
COM2	SM160	Modbusread-write instruction execution flag	When the instruction starts to execute, set ON When execution is completed, set OFF
	SM162	Free communication sending flag	When the instruction starts to execute, set ON When execution is completed, set OFF
	SM163	Free communication received flag	When receiving a frame of data or receiving data timeout, set ON. Require user program to set OFF
COM3	SM170	Modbusread-write instruction execution flag	When the instruction starts to execute, set ON When execution is completed, set OFF
	SM172	Free communication sending flag	When the instruction starts to execute, set ON When execution is completed, set OFF
	SM173	Free communication received flag	When receiving a frame of data or receiving data timeout, set ON. Require user program to set OFF

COM4	SM180	Modbusread-write instruction execution flag	When the instruction starts to execute, set ON When execution is completed, set OFF
	SM182	Free communication sending flag	When the instruction starts to execute, set ON When execution is completed, set OFF
	SM183	Free communication received flag	When receiving a frame of data or receiving data timeout, set ON. Require user program to set OFF
COM5	SM190	Modbusread-write instruction execution flag	When the instruction starts to execute, set ON When execution is completed, set OFF
	SM192	Free communication sending flag	When the instruction starts to execute, set ON When execution is completed, set OFF
	SM193	Free communication received flag	When receiving a frame of data or receiving data timeout, set ON. Require user program to set OFF

Communication registers

	No.	Function	Explanation
COM0	SD140	Modbusread and write instruction execution result	0: correct 100: receive error 101: receive timeout 180: CRC error 181: LRC error 182: station number error 183: send buffer overflow 400: function code error 401: address error 402: length error 403: data error 404: slave station busy 405: memory error (erase FLASH)
	SD141	X-Net communication result	0: correct 1: communication timeout 2: memory error 3: receive CRC error
	SD142	Free communication sending result	0: correct 410: free communication buffer overflow
	SD143	Free communication receiving result	0: correct 410: send data length overflow 411: receive data short 412: receive data long 413: receive error 414: receive timeout 415: no start symbol

			416: no end symbol
	SD144	free communication receiving data number	Count as byte, not include start symbol and end symbol
	……		
	SD149		
COM1	SD150	Modbusread and write instruction execution result	0: correct 100: receive error 101: receive timeout 180: CRC error 181: LRC error 182: station number error 183: send buffer overflow 400: function code error 401: address error 402: length error 403: data error 404: slave station busy 405: memory error (erase FLASH)
	SD151	X-Net communication result	0: correct 1: communication timeout 2: memory error 3: receive CRC error
	SD152	Free communication sending result	0: correct 410: free communication buffer overflow
	SD153	Free communication receiving result	0: correct 410: send data length overflow 411: receive data short 412: receive data long 413: receive error 414: receive timeout 415: no start symbol 416: no end symbol
	SD154	free communication receiving data number	Count as byte, not include start symbol and end symbol
	……		
	SD159		
COM2	SD160	Modbusread and write instruction execution result	0: correct 100: receive error 101: receive timeout 180: CRC error 181: LRC error 182: station number error 183: send buffer overflow 400: function code error 401: address error 402: length error 403: data error 404: slave station busy 405: memory error (erase FLASH)
	SD161	X-Net communication result	0: correct 1: communication timeout

			2: memory error 3: receive CRC error
	SD162	Free communication sending result	0: correct 410: free communication buffer overflow
	SD163	Free communication receiving result	0: correct 410: send data length overflow 411: receive data short 412: receive data long 413: receive error 414: receive timeout 415: no start symbol 416: no end symbol
	SD164	free communication receiving data number	Count as byte, not include start symbol and end symbol
	……		
	SD169		
COM3	SD170~SD179		
COM4	SD180~SD189		
COM5	SD190~SD199		

6-5 Read write serial port parameters

In addition to modifying communication parameters through serial configuration panel, it can also be realized by reading instruction [CFGCR] of serial parameters and writing instruction [CFGCW] of serial parameters.

6-5-1 Read serial port parameters [CFGCR]

1)Summary

Read the serial port parameters to local specified registers.

Read serial port parameters[CFGCR]			
16-bit instruction	CFGCR	32-bit instruction	-
Execution condition	Normally ON/OFF, rising edge triggering	Suitable model	XD, XL
Hardware	-	Software	V3.4 and higher version

2)Operands

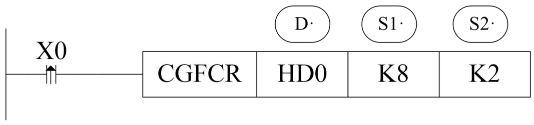
Operands	Function	Type
D	Local register starting address	16-bit, BIN
S1	Read serial port parameters number	16-bit, BIN
S2	Serial port no.	16-bit, BIN

3)Suitable soft component

Operands	Word soft elements											Bit soft elements						
	System								Constant	Module		System						
	D	FD	TD	CD	DX	DY	DM	DS	K/H	ID	QD	X	Y	M	S	T	C	Dn.m
D	●																	
S1	●	●							●									
S2	●								K									

*Notes: D includes D, HD; TD includes TD, HTD; CD includes CD, HCD, HSCD, HSD; DM includes DM, DHM; DS includes DS, DHS. M includes M, HM, SM; S includes S, HS; T includes T, HT; C includes C, HC.

Function and action



- Operator S1: The number of registers used to read serial parameters is generally 8 (XD5E/XDME series is 9).
- Operator S2: Serial port range: K0 ~ K5. K0: COM0, K1: COM1, K2: COM2 or COM2-RS232 or COM2-RS485, K3: COM3, K4: COM4, K5: COM5.
- Read 8 parameters of serial port 2 to HD0~HD7. See sections 6-5-3 for the names and definitions of specific parameters.

6-5-2 Write serial port parameters [CFGCW]

1)Summary

Write the local specified register value to specific serial port.

Write serial port parameters[CFGCW]			
16-bit instruction	CFGCW	32-bit instruction	-
Execution condition	Normally ON/OFF, rising edge triggering	Suitable model	XD, XL
Hardware	-	Software	V3.4 and higher version

2)Operand

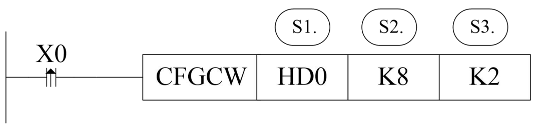
Operands	Function	Type
S1	Local register starting address	16-bit, BIN
S2	Write serial port parameters number	16-bit, BIN
S3	Serial port no.	16-bit, BIN

3)Suitable soft component

Operands	Word soft elements											Bit soft elements						
	System								Constant	Module		System						
	D	FD	TD	CD	DX	DY	DM	DS	K/H	ID	QD	X	Y	M	S	T	C	Dn.m
S1	●																	
S2	●	●							●									
S3	●								K									

*Notes: D includes D, HD; TD includes TD, HTD; CD includes CD, HCD, HSCD, HSD; DM includes DM, DHM; DS includes DS, DHS. M includes M, HM, SM; S includes S, HS; T includes T, HT; C includes C, HC.

Function and action



- Operator S2: The number of registers used to write serial parameters is generally 8 (XD5E/XDME series is 9).
- Operator S3: Serial port range: K0 ~ K5. K0: COM0, K1: COM1, K2: COM2 or COM2-RS232 or COM2-RS485, K3: COM3, K4: COM4, K5: COM5.
- Write HD0~HD7 parameters to serial port 2. See sections 6-5-3 for the names and definitions of specific parameters.

6-5-3 Serial port parameter name and setting

Assuming that HD0-HD14 corresponds to serial port parameters, the parameter names and settings represented by registers are shown in the table below.

<table>
<tr><th rowspan="3">Para meter addre ss</th><th colspan="5">Parameter name and settings</th></tr>
<tr><th rowspan="2">MODBUScom munication(HD 0=1)</th><th rowspan="2">Free communication (HD0=2)</th><th colspan="2">X-NET communication</th><th rowspan="2">Ethernet communication(HD0=3)</th></tr>
<tr><th>OMMS (HD0=3)</th><th>TBN (HD0=3)</th></tr>
<tr><td>HD0</td><td colspan="5">Network type
1: MODBUS; 2: free ; 3: X-NET; 4: MODBU-TCP</td></tr>
<tr><td>HD1</td><td>MODBUS station no.
1~254</td><td>Baud rate refer to table 1</td><td>Net ID
0~32767</td><td>Net ID
0~32767</td><td>Net ID
IP address high 2-byte</td></tr>
<tr><td>HD2</td><td>Transmission mode
0: RTU
128: ASCII</td><td>Frame format refer to table 2</td><td>Station no.
0~100</td><td>Station no.
0~100</td><td>Station no.
IP address low 2-byte</td></tr>
<tr><td>HD3</td><td>Baud rate refer to table 1</td><td>Free properties
bit7:
1: with start character
0: no start character
bit6：
1: with end character
0: no end character</td><td colspan="3">Physical layer type
1: PHY_RS485
2: PHY_SOF(Unidirectional Fiber Ring Network)
3: PHY_OFPP(Optical Fiber Point Network)
4: PHY_RS232
5: PHY_RS422
6: PHY_TTL (TTLvoltage network)</td></tr>
<tr><td>HD4</td><td>Frame format refer to table 2</td><td>Start character</td><td colspan="3">Link Layer Type
0: TBN
1: HDN
2: CCN
3: PPFD
4: PPU
5: Ethernet</td></tr>
<tr><td>HD5</td><td>retry count
0~5</td><td>End character</td><td>OMMS properties
128: Supports periodic communication, otherwise does not support</td><td>Baud rate refer to table 1</td><td>Subnet mask high 2-byte</td></tr>
<tr><td>HD6</td><td>Reply timeout
0~65535</td><td>Frame timeout
0~255</td><td>OMMS baud rate refer to table 1</td><td>Token Cycle Time
1~60000(ms)</td><td>Subnet mask low 2-byte</td></tr>
<tr><td>HD7</td><td>Delay before sending
0~255</td><td>Reply timeout
0~65535 (0 is infinite wait)</td><td>OMMS slave station list
Each bit of each byte in the array indicates whether the slave station is accessible (the master station is valid, i.e. the</td><td>Max station number
1~100</td><td>Gateway address high 2-byte</td></tr>
</table>

			station number is 1).		
HD8	-	-	-	-	Gateway address low 2-byte

Note: The table does not contain "buffer digits" in free communication mode, so "buffer digits" can not be read and written through CFGCR and CFGCW instructions, but can be read and written using MOV instructions. The address of "buffer digits" is shown in Appendix 3.

Table 1: baud rate

Value	Baud rate	Value	Baud rate	Value	Baud rate	Value	Baud rate
1	300 bps	7	19200 bps	13	256000 bps	19	1000000 bps
2	600 bps	8	28800 bps	14	288000 bps	20	1200000 bps
3	1200 bps	9	38400 bps	15	384000 bps	21	1500000 bps
4	2400 bps	10	57600 bps	16	512000 bps	22	2400000 bps
5	4800 bps	11	115200 bps	17	576000 bps	23	3000000 bps
6	9600 bps	12	192000 bps	18	768000 bps		

Table 2: frame format

Stop bit		Parity bit			Data bit length		
Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
00: 1 01: 1.5 10: 2		000: no 001: odd 010: even 011: empty 100: Mask			000: 5 001: 6 010: 7 011: 8 100: 9		

7 PID Control Function

In this chapter, we mainly introduce the applications of PID instructions for XD, XL series, including: call the instructions, set the parameters, items to notice, sample programs etc.

7-1 PID Introduction

PID instruction and auto tune function are added into XD/XL series PLC basic units. Via auto tune method, users can get the best sampling time and PID parameters and improve the control precision.
PID instruction has brought many facilities to the users.
Output can be data form D, HD, and on-off quantity Y, user can choose them freely when programming.
Via auto tune, users can get the best sampling time and PID parameters and improve the control precision.
User can choose positive or negative action via software setting. Positive action is used for heating control; negative action is used for cooling control.
PID control separates the basic units with the expansions, which improves the flexibility of this function.
XD/XL series PLC have two methods for auto tune, step response method and critical oscillation method.
For temperature control object:
Step response method: the PID auto tune will start when current temperature of object controlled is equal to ambient temperature.
Critical oscillation method: the PID auto tune can start at any temperature.

7-2 Instruction Form

1)Summary
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 trigger	Suitable models	XD/XL
Hardware requirement	-	Software requirement	V3.2 or later

2)Operands

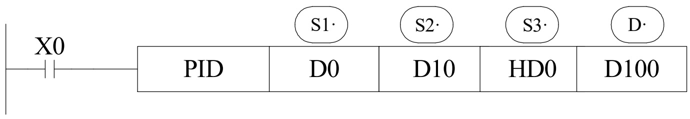
Operands	Function	Type
S1	set the address of the target value (SV)	16bits, BIN
S2	set the address of the tested value (PV)	16 bits, BIN
S3	set the start address of the control parameters	16 bits, BIN
D	the address of the operation result (MV) or output port	16 its, BIN; bit

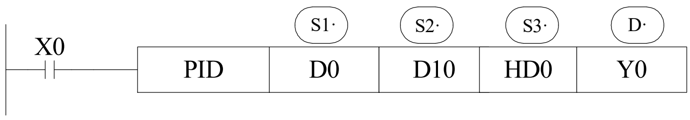
3)Suitable soft components

Operands	Word soft elements											Bit soft elements						
	System								Constant	Module		System						
	D	FD	TD	CD	DX	DY	DM	DS	K/H	ID	QD	X	Y	M	S	T	C	Dn.m
S1	●	●							●									
S2	●	●																
S3	●	●																
D	●	●											●	●	●	●	●	

*Note: D includes D, HD; TD includes TD, HTD; CD includes CD, HCD, HSCD, HSD; DM includes DM, DHM; DS includes DS, DHS. M includes M, HM, SM; S includes S and HS; T includes T and HT; C includes C and HC.

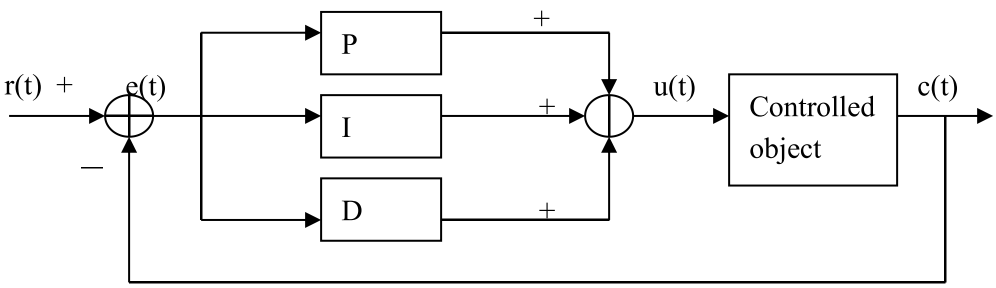
Functions and Action





- S3~ S3+ 69 will be occupied by this instruction, so please don't use them as the common data registers.
- This instruction executes when each sampling time interval comes.
- For the operation result, data registers are used to store PID output values; the output points are used to output the occupy duty ratio in the form of ON/OFF.
- PID control rules are shown as below:

P: proportion, I: integral, D: differential



Analog PID control system

$e(t) = r(t) - c(t)$ (1-1)

$u(t) = Kp\,[\,e(t) + 1/Ti \int e(t)dt + TD\, de(t)/dt\,]$ (1-2)

Here, e(t) is offset value, r (t) is the setting value, c (t) is actual output value and the 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.

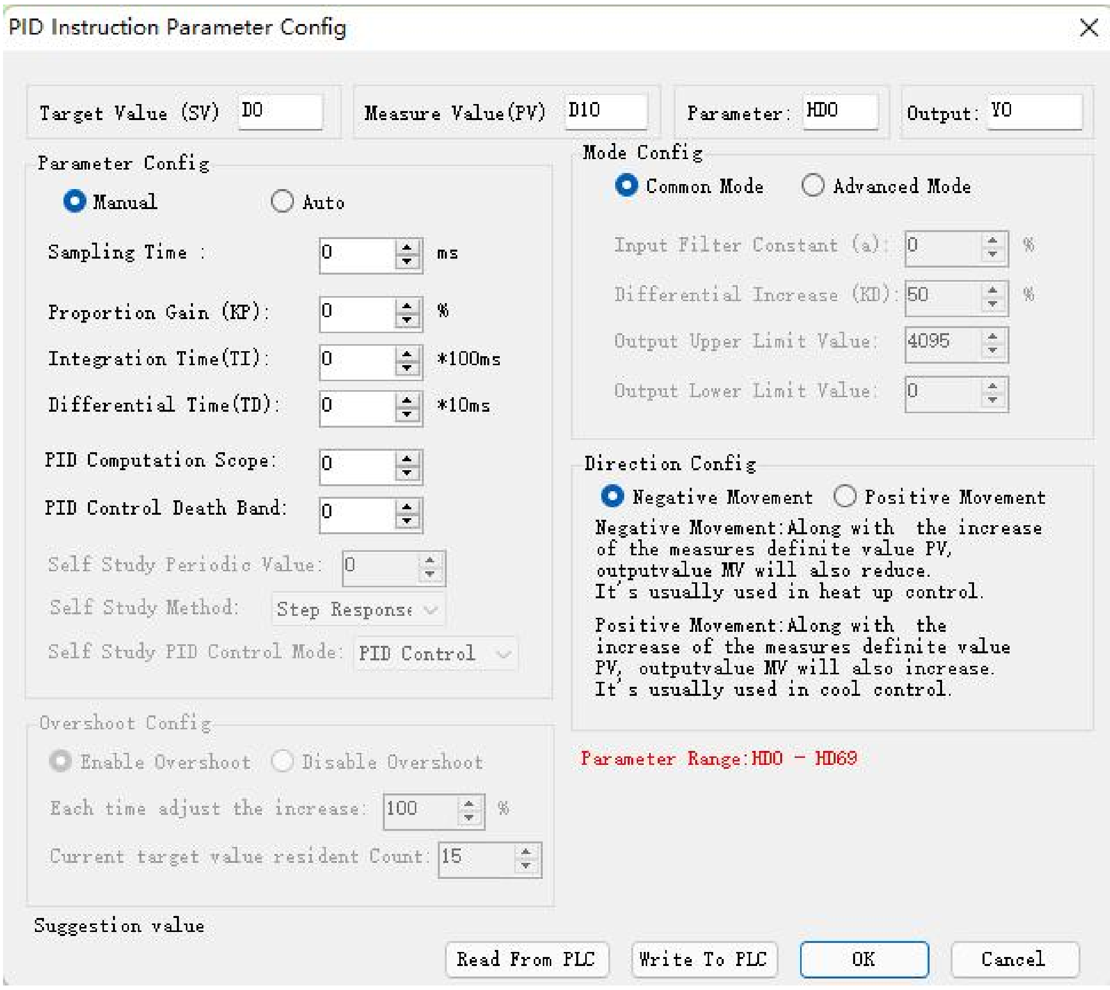
The result of the operation:

1．Analog output: digital form of MV = u(t), the default range is 0~4095.

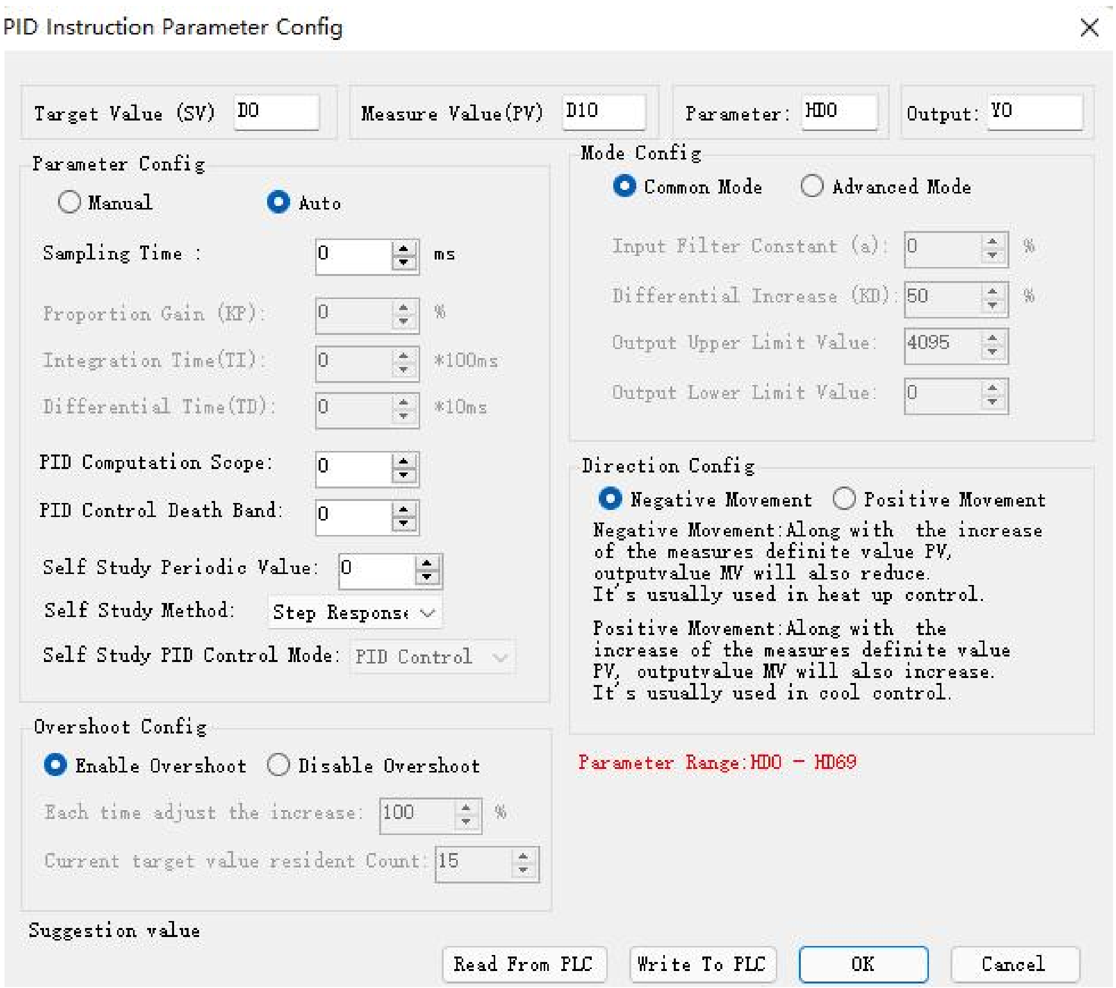
2．Digital output: Y = T * [MV / PID output upper limit]. Y is the outputs activate time within the control cycle. T is the control cycle, equals to the sampling time. PID output upper limit default value is 4095.

7-3 Parameters setting

Users can call PID in XDP Pro software directly and set the parameters in the window (see graph below), for the details please refer to XDP Pro user manual. Users can also write the parameters into the specified registers by MOV instructions before PID operation.



Auto tune mode:



V3.2 and higher version software can choose auto tune mode: step response or critical oscillation.

7-3-1 Register and their functions

PID control instruction's relative parameters ID, please refer to the below table:

ID	Function	Description	Memo
S3	Sampling time	Whatever it is manual or auto mode, all needs to set	32bits without sign, Unit: ms
S3+2	Mode setting	bit0：0：negative action; 1：positive action bit1~bit6 not usable bit7：0：manual PID; 1：auto tune PID bit8：1：auto tune successful flag bit9~bit10：auto tune method 00：step response 01：critical oscillation bit11~bit12：not useful bit13~bit14 auto tune PID mode (valid in critical oscillation	

		mode) 00：PID control 01：PI control 10：P control bit15：0：regular mode； 1：advanced mode;	
S3+3	Proportion Gain (Kp)	0~32767[%]	
S3+4	Integration time (TI)	0~32767[unit: 100ms]	0 is taken as no integral.
S3+5	Differential time (TD)	0~32767[unit: 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 output value will not change in death zone
S3+8	Sampling temperature filter coefficient	0~100[%]	Filter the input sampling temperature in advanced mode, 0 is no input filter
S3+9	Differential gain(KD)	0~100[%]	Only for advanced mode (normal mode default value is 50%), 0 is no differential gain
S3+10	Upper limit value of output	0~32767	
S3+11	Lower limit value of output	0~32767	
S3+12	Change of Unit Temperature Corresponds to Change of AD Value	full scale AD value *(0.3~1%) default value is 10	16-bit no sign, only for step PID
S3+13	PID auto tune overshoot	0：enable overshoot 1： not overshoot(try to reduce the overshoot)	only for step PID
S3+14	Current target value adjusting percentage every time in auto tune end transition stage	Cannot adjust	16-bit no sign, only for step PID
S3+15	Number of times exceeding the target value in auto tune end transition stage when limiting the overshoot		only for step PID, default value is 15
S3+16	PID type and status	Bit0~bit1： 00: manual mode 01: step mode 10: Critical oscillation mode Bit8： 0: manual control status 1: auto tune end, enter manual control status	Internal use parameters of the system for monitoring purposes only

S3+17	PID max output	0~32767	Internal use parameters of the system for monitoring purposes only
S3+18	PID min output	0~32767	Internal use parameters of the system for monitoring purposes only
S3+19	Last time sampling time	0~sampling time (unit: ms)	16-bit no sign, Internal use parameters of the system for monitoring purposes only
S3+20	Actual sampling time space	The value is around the sampling time	32-bit no sign, Internal use parameters of the system for monitoring purposes only
S3+22	Last time user set target temperature	The value before changing the target temperature	Internal use parameters of the system for monitoring purposes only
S3+23	-	-	Parameter is reserved

The following is the joint address (divided into step setting, critical oscillation setting and manual control)			
Step part (read only parameters, only for monitoring)			
S3+24	Actual sampling space	0~4294967296 (unit: ms)	Internal usage parameters of the system
S3+26	Operating segment of auto-tuning PID	0: Preparation stage 1~2: auto tune parameter collection 3: calculate PID parameters	Internal usage parameters of the system
S3+28	Duration of auto-tuning PID operating parameters	0~4294967296 (unit: ms)	Internal usage parameters of the system
S3+30	Real-time accumulation of two inflection points	Clear and recalculate the time when reaching the inflection point0~4294967296(unit: ms)	Internal usage parameters of the system
S3+32	Sampling variation of inflection point	Sampling difference between two inflection points -2147483648~2147483647	Internal usage parameters of the system
S3+34	Sampling interval time of inflection point EK	0~4294967296 (unit: ms)	Internal usage parameters of the system
S3+36	Time from auto-tuning PID to inflection point	0~4294967296 (unit: ms)	Internal usage parameters of the

			system
S3+38	Last sampling temperature	-32767~32767	Internal usage parameters of the system
S3+39	The time from auto-tuning PID operation to inflection point	-32767~32767 (unit: ms)	Internal usage parameters of the system
S3+40	Starting sampling value of auto-tuning PID operation	-32767~32767	Internal usage parameters of the system
S3+41	Number of times at inflection point during auto-tuning	0~65535	Internal usage parameters of the system
S3+42	Useless time	0~4294967296 (unit: ms)	Internal usage parameters of the system
S3+44	Stop temperature	Temperature at the end of auto-tuning Range:-32767~32767	Internal usage parameters of the system
Critical oscillation part (read only parameters, only for monitoring)			
S3+24	PID control mode	0: PID control 1: PI control 2: P control	16-bit no sign, internal usage parameters of the system
S3+25	Current auto-tuning segment	0: Preparation stage 1: start to auto tune 2~3: auto-tuning parameter collection 4: calculation of PID parameters	16-bit no sign, internal usage parameters of the system
S3+26	The auto-tuning temperature is located at the number of peaks	0: first peak 1: second peak	16-bit no sign, internal usage parameters of the system
S3+27	The lowest sampling temperature	-32767~32767	Internal usage parameters of the system
S3+28	The highest sampling temperature	-32767~32767	Internal usage parameters of the system
S3+30	sampling time of the lowest sampling temperature	0~4294967296 (unit: ms)	Internal usage parameters of the system
S3+32	sampling time of the highest sampling temperature	0~4294967296 (unit: ms)	Internal usage parameters of the system
S3+34	auto-tuning time cumulative	0~4294967296 (unit: ms)	Internal usage parameters of the system
Manual control part (read only parameters, only for monitoring)			
S3+24	current target temperature	-32767~32767	Internal usage parameters of the system
S3+25	Need to update target temperature	0: no need 1: need	16-bit no sign, internal usage

			parameters of the system
S3+26	Number of times to reach target temperature	0~65535	Internal usage parameters of the system
S3+27	PID upper limit of operational range	-32767~32767	Internal usage parameters of the system
S3+28	PID lower limit of operational range	-32767~32767	Internal usage parameters of the system
S3+30	High voltage time when PID uses Y to output	0~4294967296 (unit: ms)	Internal usage parameters of the system
S3+32	Sampling temperature after last filtering	The filtered temperature acquired in the last sampling time (the input filter constant in the advanced mode needs to be set first)	Floating point, internal usage parameters of the system
S3+34	Last temperature deviation		Floating point, internal usage parameters of the system
S3+36	Value of last integral term	digital value corresponding to Ui of the last sampling time	Floating point, internal usage parameters of the system
S3+38	Value of last differential term	digital value corresponding to Ud of the last sampling time	Floating point, internal usage parameters of the system
S3+40	Last PID output		Floating point, internal usage parameters of the system

Note: When the auto-tuning mode is changed to manual control, the value in the original address of S3+24~S3+40 will be overwritten by the value in manual control mode.

7-3-2 Parameters Description

Movement direction:

Positive movement: the output value MV will increase with the increasing of the measured value PV, usually used for cooling control.

Negative movement: the output value MV will decrease with the increasing of the measured value PV, usually used for heating control.

Mode setting

Common Mode:

Parameters register range: S3~S3+69, and S3~S3+7 need to be set by users;

S3+8~S3+69 are occupied by system, users can't use them.

Advanced Mode

Parameters register range: S3~S3+69, among them S3~S3+7 and S3+8~S3+11 need to be set by users; S3+16~S3+69 are occupied by system, users can't use them.

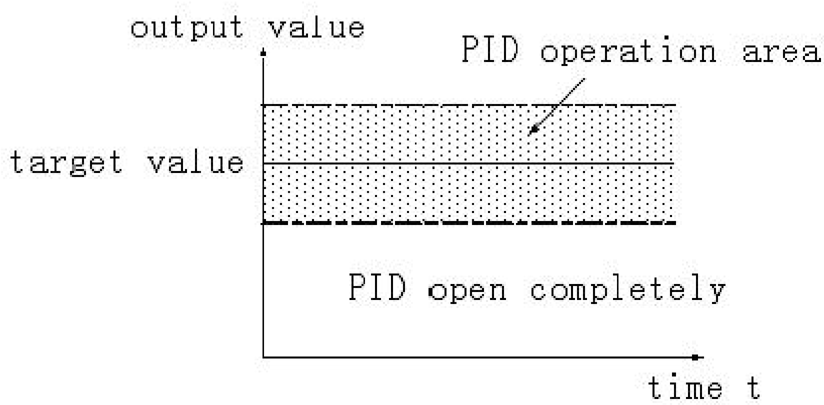
Sample time[S3]

The system samples the current values according to some certain interval and compares them with the output value. This time interval is the sample time **T**. There is no requirement for **T** during **DA** 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(default value is 4095), when it entered into the PID computation range, parameters Kp, TI, TD will be effective.

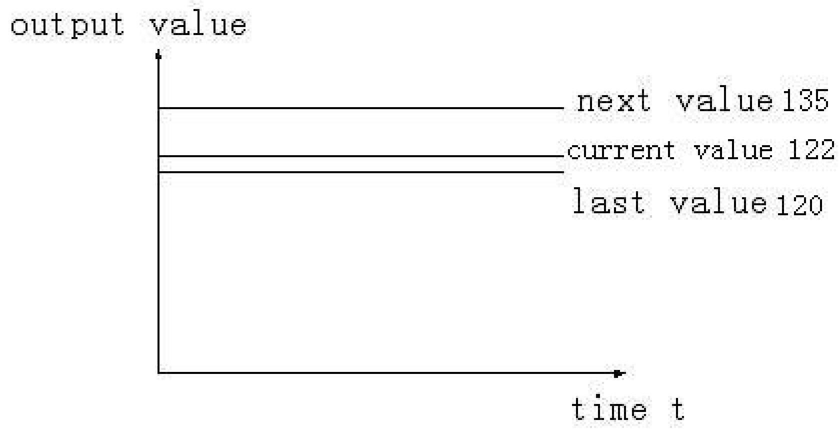
See graph below:



If the target value is 100, PID operation zone is 10, and then the real PID's operation zone is from 90~110.

Death Region [S3+7]

If the measured value changed slightly for a long time, and PID control is still in working mode, then it belongs to meaningless control. Via setting the control death region, we can overcome this situation. See graph below:



Suppose: we see 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. It will not do PID control; 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. it will do the PID control with 135.

7-4 Auto Tune Mode

If users do not know how to set the PID parameters, they can choose auto tune mode which can find the best control parameters (sampling time, proportion gain **Kp**, integral time **Ti**, differential time **TD**) automatically.

Auto tune mode is suitable for these controlled objects: temperature, pressure; not suitable for liquid level and flow.

Auto-tuning is the process of extracting PID parameters. Sometimes auto-tuning can not find the best parameters at one time. It needs auto-tuning for many times. It is normal that there is a vibration in the process. After the optimum parameters are found at the end of auto-tuning, please switch to the manual PID mode. If the control object is unstable in the process of manual PID, it can not be controlled at a constant target value, which may be caused by the unsatisfactory adjustment of parameters. It is necessary to re-adjust the parameters of PID to achieve stable control.

For step response method: 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.
For step response method: Before doing auto tune, the system should be under the non-control steady state. Take the temperature for example: the measured temperature should be the same to the environment temperature.

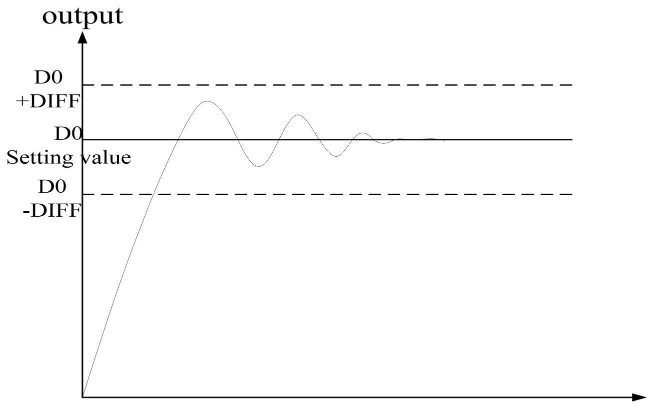
For critical oscillation method: user needs to set the sampling time at the beginning of the auto tune process. For slow response system, 1000ms. For fast response system, 10-100ms.

For critical oscillation method: the system can start the auto tune at any state. For object temperature, the current temperature doesn't need to be same to ambient temperature.

Two different methods and PID control diagram:

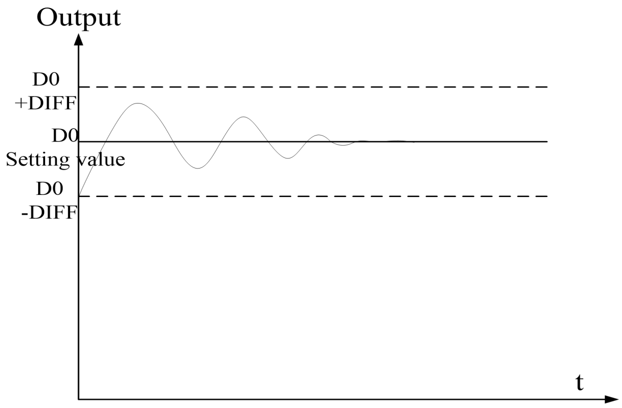
(1) Step response method

Make sure current temperature is equal to ambient temperature



(2) Critical oscillation method

The auto tune start temperature can be any value.



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)** turn to 1, it means the auto tune is successful.

PID auto tune period value [S3+12]

Set this value in S3+12 during auto tune. This value decides the auto tune performance, in a general way, set this value to be AD result corresponding to one standard tested unit. The default value is 10. The suggested setting range: fall-scale AD result×0.3~1%.

User doesn't need to change this value. However, if the system is interfered greatly by outside, this value should be increased modestly to avoid wrong judgment of positive and negative movement. If this value is too large, the PID control period (sampling time) got from the auto tune process will be too long. As the result do not set this value too large.

※1：If users have no experience, please use the default value 10, set PID sampling time (control period) to be 0msthen start the auto tune.

PID auto tune overshooting permission setting [S3+13]

If set 0, overshooting is permitted, and the system can study the optimal PID parameters all the time. But in auto tune 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+13]** to be 1 to prevent from tested value over the target value seriously.

In the process, if **[S3+2]** bit8 changes from 0 to 1, it means the auto tune is successful and the optimal parameters are got; if **[S3+2]** bit8 keeps 0, when **[S3+2]** bit7 changes from 1 to 0, it means auto tune is finished, but the parameters are not the best and they need to be modified by hand.

Every adjustment percent of current target value in auto tune end transition stage [S3+14]

This parameter is effective only when **[S3+13]** is 1.

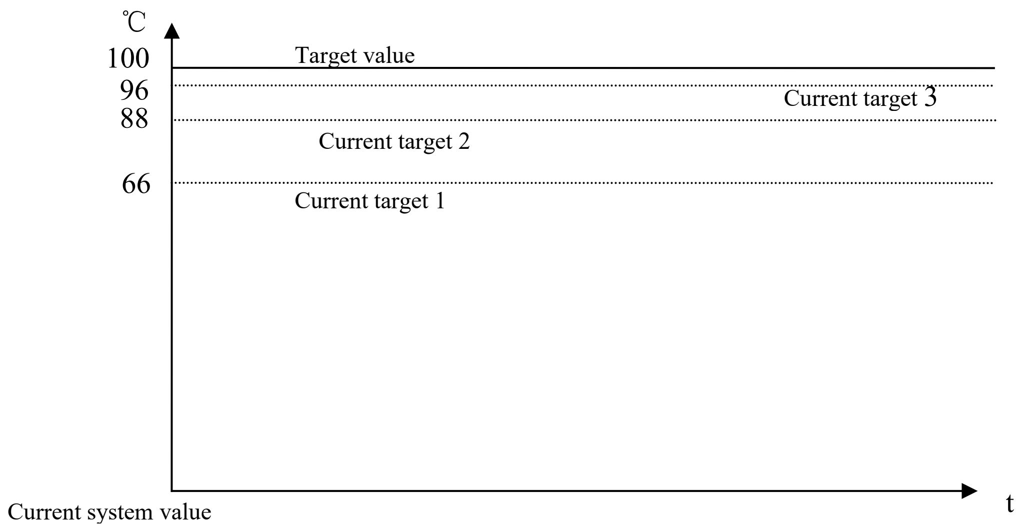
If doing PID control after auto tune, small range of overshooting may be occurred. It is better to decrease this parameter to control the overshooting. But response delay may occur if this value is too small. The defaulted value is 100% which means the parameter is not effective. The recommended range is 50~80%.

Cutline Explanation:

Current target value adjustment percent is 2/3(**S3 + 14** = 67%), the original temperature of the system is 0 ºC, target temperature is 100 ºC, and 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.



Over target value times in auto-tuning end transition stage when limiting the overshoot[S3+15]

This parameter is valid only when [S3+13] is 1;

If entering into PID control directly after auto tune, small range of overshoot may occur. It is good to prevent the overshoot if increasing this parameter properly. But it will cause responselag if this value is too large. The default value is 15 times. The recommended range is from 5 to 20.

7-5 Advanced Mode

Users can set some parameters in advanced mode in order to get better PID control effect. Enter into the advanced mode, please set **[S3+2]** bit 15 to be 1, or set it in the XDP Pro software.

Input Filter constant [S3+8]

It will smooth the sampling value. The default value is 0%, which means no filter.

Differential Gain[S3+9]

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 [S3+10], [S3+11]

Users can choose the analog output range via setting this value.

Default value: lower-limit output =0

Upper-limit =4095

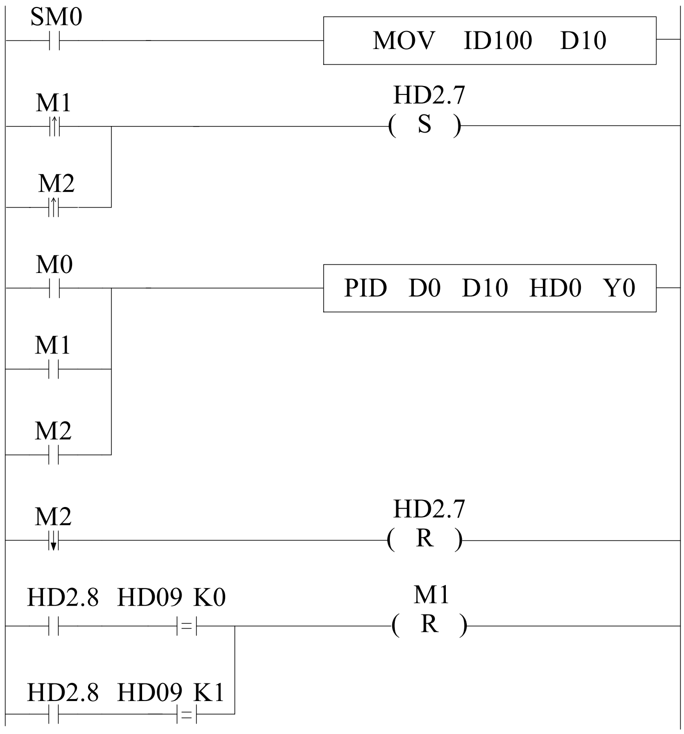
7-6 Application outlines

Under the circumstances of continuous output, the system whose effect ability will die down with the change of the feedback value can do auto tune, such as temperature or pressure. It is not suitable for flux or liquid level.
Under the condition of overshooting permission, the system will get the optimal PID parameters from auto tuning.
Under the condition that overshoot not allowed, the PID parameters got from auto tune 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 auto tune 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 (minute) 1 ~ 5

7-7 Application

Example 1:
PID control program is shown below:



// Move ID100 content into D10

// auto tune mode, or set to autotune mode after auto tune end

// start PID, D0 is target value, D10 is the measured value, from HD0 is PID parameters area; output PID result byY0

// PID control finish, close auto tune PID mode

// if auto tune is successful, and overshoot is permitted, close auto tune control bit, auto tune will finish;
If auto tune turns to be manual mode, and overshoot is not permitted, close auto tune control bit.

Soft element function comments:

HD2.7: Auto tune bit

HD2.8: Successful flag of auto tune

M0: Normal PID control

M1: Auto tune control

M2: Enter PID control after auto tune

Operation steps:

1. Send the actual temperature to PID collection register
2. Set probably value for P, I, D, sampling period
3. Set ON auto tune control bit M1 to startup PID auto tune
4. M1 will be reset after the auto tune is finished
5. Set ON M0, use the PID parameters getting from auto tune
6. If the PID effect is not good by using the auto tune PID parameters, user can adjust the PID parameters to get good effect.

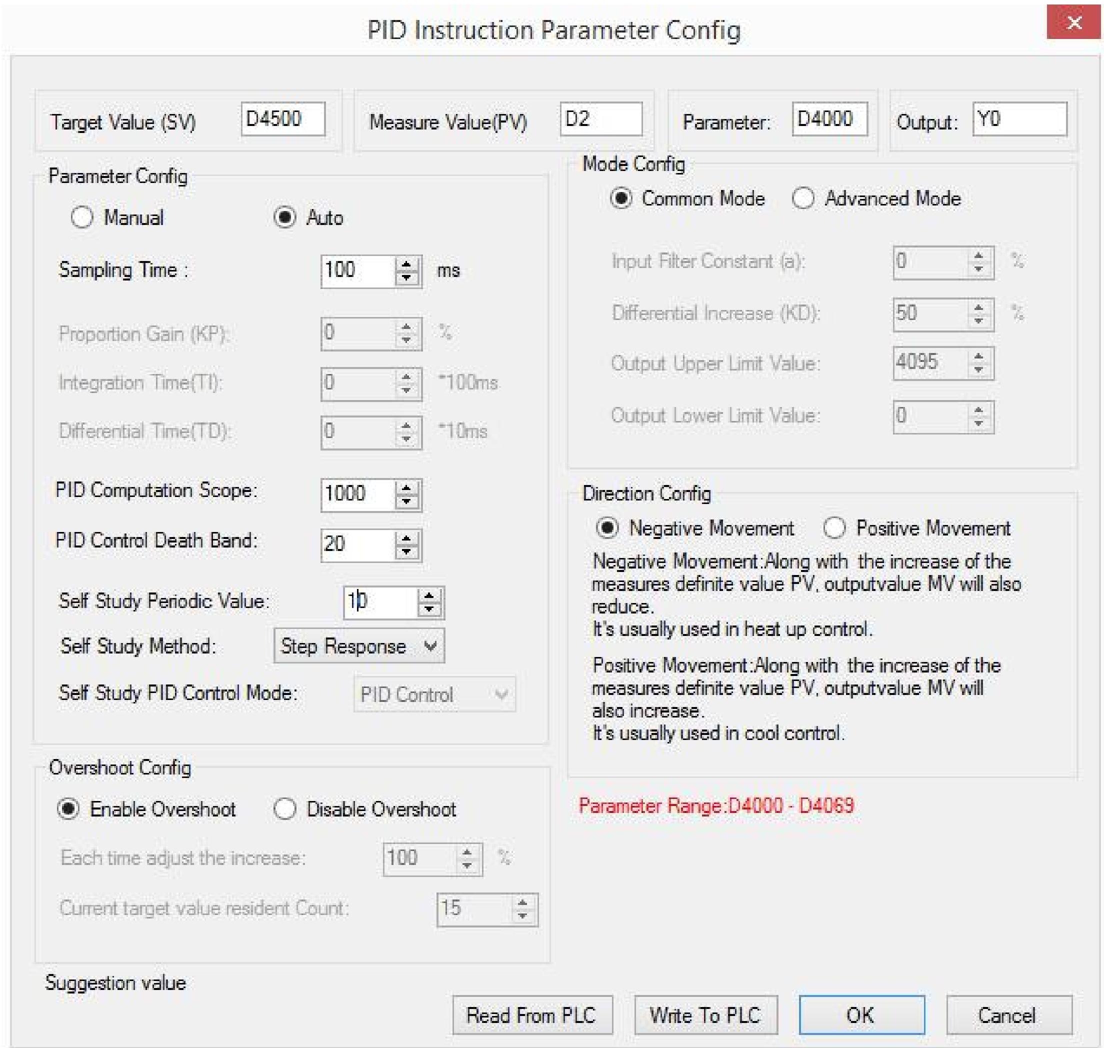
Note: This PLC temperature PID control program is applicable to almost all temperature control projects.

Example 2:

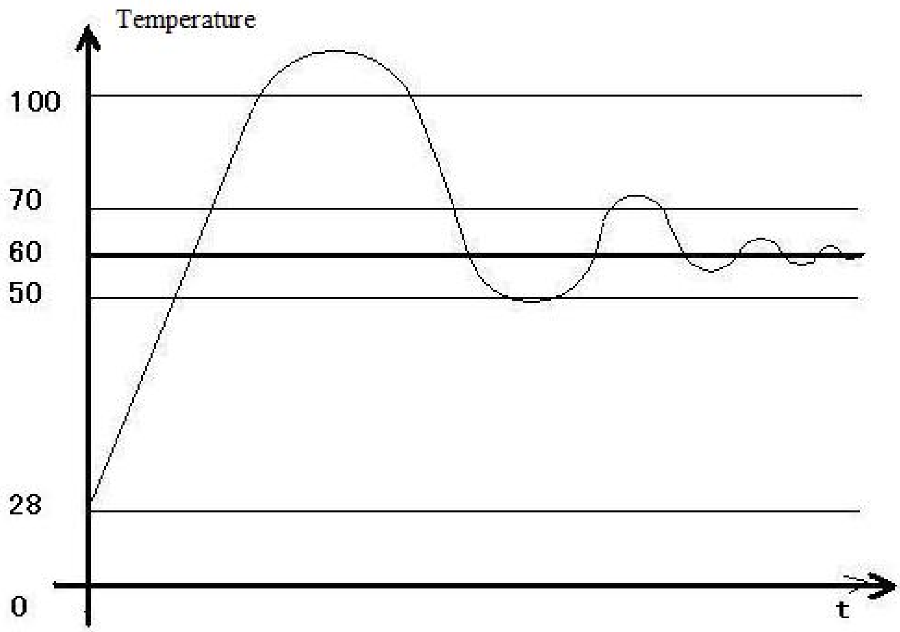
To control the target temperature 60°C in step response mode.

Overshoot is permitted:

1. The target temperature 60°C (600)
2. Parameters setting



3. The result curve

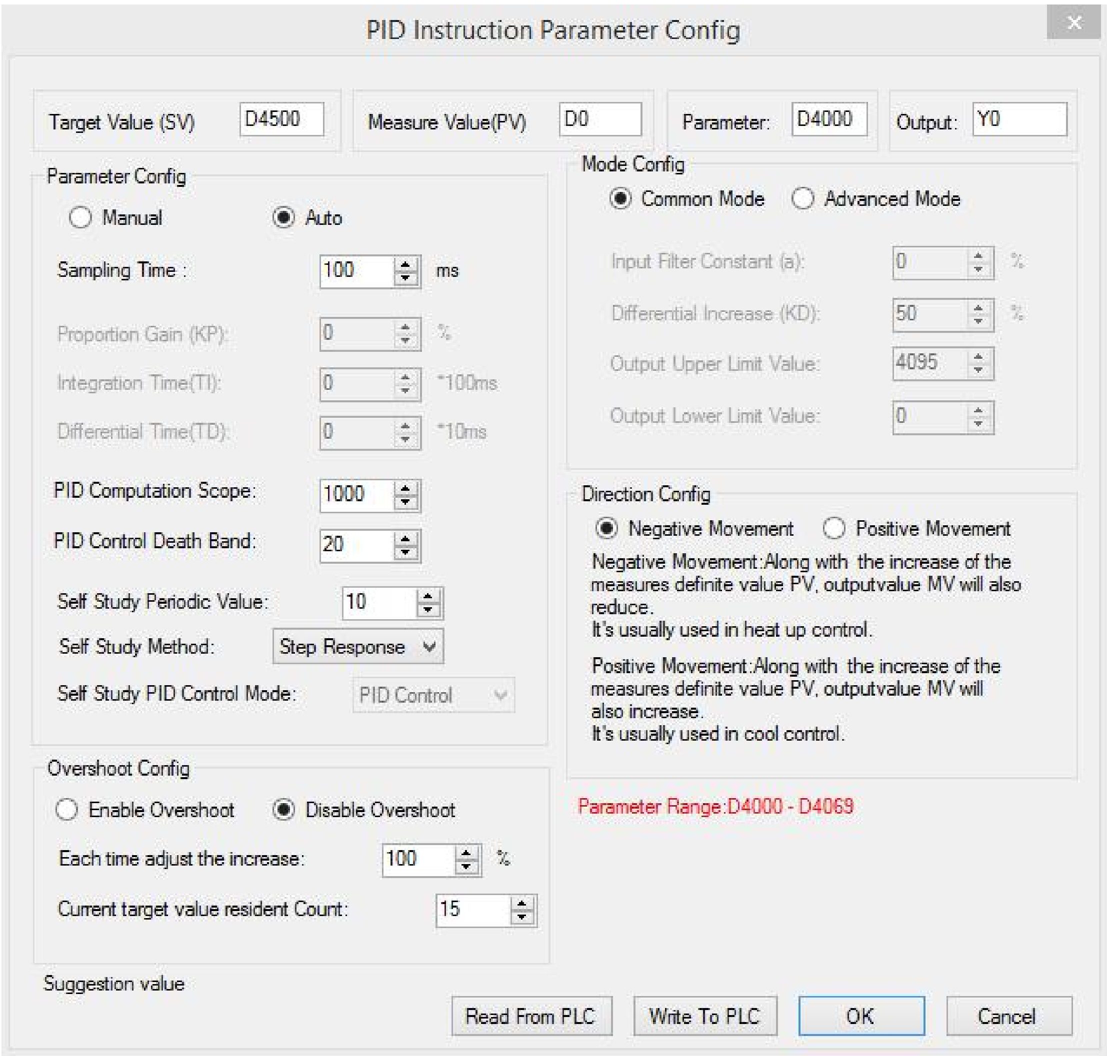


Explanation:

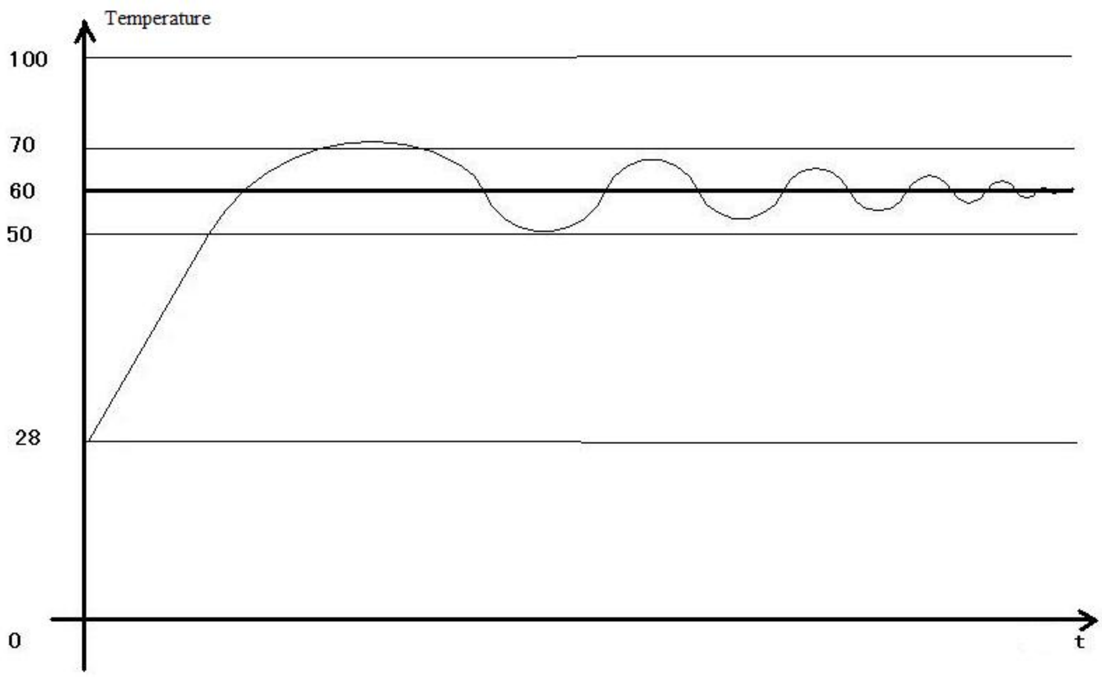
The target temperature is 60 degree, PID calculation range is 10 degree, PID control dead area is 0.2 degree, auto tune period changing value is 10. When the PID control works in normal atmospheric temperature, the PID output terminal will heat the temperature from 28 to 100 degree, then the output stops, the temperature keeps increasing to 110 degree (max temperature) as the remaining warmth. Then the temperature keeps decreasing to 60 degree, the output starts to heat again to 70 degree and stops. The temperature increases a little then decreases again. This process will repeat. Finally, the temperature will fluctuate close the target temperature.

Note:
1. When the temperature reaches 100 degree and stops heating, the PID start bit D4002.7 will not reset at once, it has delay before reset.
2. When the temperature reaches 100 degree and stops heating, the PID auto tune success bit D4002.8 will be ON at once.
3. When it starts PID calculation, the PLC will auto set a sampling time (about 2500). This parameter will be replaced by the PID best sampling time after stoping heating at 100 degree.
4. When it starts PID calculation, the PLC will auto set the PID parameters (P=4454, I=926, D=2317). These parameters will be replaced by the best PID value after stoping heating at 100 degree.
5. When the temperature reaches 100 degree and stops heating, the PID start bit D4002.7 will not reset at once, it has delay before reset. At this time, the sampling temperature is higher than target temperature. If user sets ON the PID auto tune again, PLC will get all the PID parameters as 0. Please set ON the PID after the temperature decreases under the normal atmospheric temperature.
6. If PID auto tune start bit and auto tune success bit are power-off retentive, please set or reset them propably to avoid calculation error when starting the PLC next time.
7. The final heating temperature will up to 110 degree when the overshoot is permitted. It is over the target temperature by 50 degree, the overshoot amount is too large.
8. When the PID starts to work, the output will heat the object from 28 degree to 60 degree, then the output is forced to stop heating to avoid overshoot, but this will interrupt the PID auto tune process.
9. To enlarge the PID calculation range can suppress the heating overshoot.

Overshoot is not permitted:
1. The target temperature is 60 degree (600)
2. The related parameter settings:



3. The result curve



Explanation:

The target temperature is 60 degree, PID calculation range is 10 degree, PID control dead area is 0.2 degree, auto tune period changing value is 10. When the PID control works in normal atmospheric temperature, the PID output terminal will heat the temperature from 28 to 48 degree, then the output stops, the temperature keeps increasing to 70 degree (max temperature) as the remaining warmth. Then the temperature keeps decreasing to 60 degree, the output starts to heat again to 62 degree and stops. The temperature increases a little (about 64 degree) then decreases again. This process will repeat. Finally, the temperature will fluctuate close the target temperature. The precision is $\pm$0.25 degree.

Note:

1. When the temperature reaches 48 degree and stops heating, the PID start bit D4002.7 will not reset at once, it has delay before reset.
2. When the temperature reaches 48 degree and stops heating, the PID auto tune success bit D4002.8 will not be ON at once. It hasn't set ON even when the auto tune succeeded.
3. When it starts PID calculation, the PLC will auto set a sampling time (about 2500). This parameter will be replaced by the PID best sampling time after stoping heating at 48 degree.
4. When it starts PID calculation, the PLC will auto set the PID parameters (P=4454, I=926, D=2317). These parameters will be replaced by the best PID value after stoping heating at 48 degree.
5. When the temperature reaches 48 degree and stops heating, the PID start bit D4002.7 will not reset at once, it has delay before reset. At this time, the sampling temperature is higher than target temperature. If user sets ON the PID auto tune again, PLC will get all the PID parameters as 0. Please set ON the PID after the temperature decreases under the normal atmospheric temperature.
6. If PID auto tune start bit and auto tune success bit are power-off retentive, please set or reset them propably to avoid calculation error when starting the PLC next time.
7. The final heating temperature will up to 70 degree when the overshoot is permitted. It is over the target temperature by 10 degree, the overshoot amount is small.
8. To enlarge the PID calculation range can suppress the heating overshoot.

8 C Language Function Block

In this chapter, we focus on C language function block's specifications, edition, instruction calling, application points etc. We also attach the common function list.

8-1 Summary

XD/XL series supports to write function blocks in C language in the Xinje PLC software and call them where needed. It supports almost all C language functions (compared with XC series, XD/XL series also supports global variables), which enhances the confidentiality of the program. At the same time, it can call many places and different files, greatly improves the efficiency of programmers.

8-2 Instruction Format

1) 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	XD, XL
Hardware		Software	

2) Operands

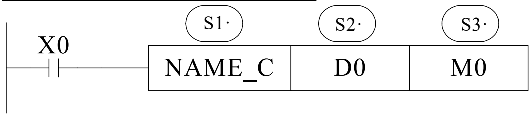
Operands	Function	Type
S1	Name of C Function Block, defined by the user	String
S2	Corresponding start ID of word W in C language function	16 bits, BIN
S3	Corresponding start ID of bit B in C language function	bit, BIN

3) Suitable Soft Components

Operands	Word soft elements											Bit soft elements						
	System								Constant	Module		System						
	D	FD	TD	CD	DX	DY	DM	DS	K/H	ID	QD	X	Y	M	S	T	C	Dn.m
S1																		
S2	●																	
S3														●				

*Note: D includes D, HD; TD includes TD, HTD; CD includes CD, HCD, HSCD, HSD; DM includes DM, DHM; DS includes DS, DHS. M includes M, HM, SM; S includes S and HS; T includes T and HT; C includes C and HC.

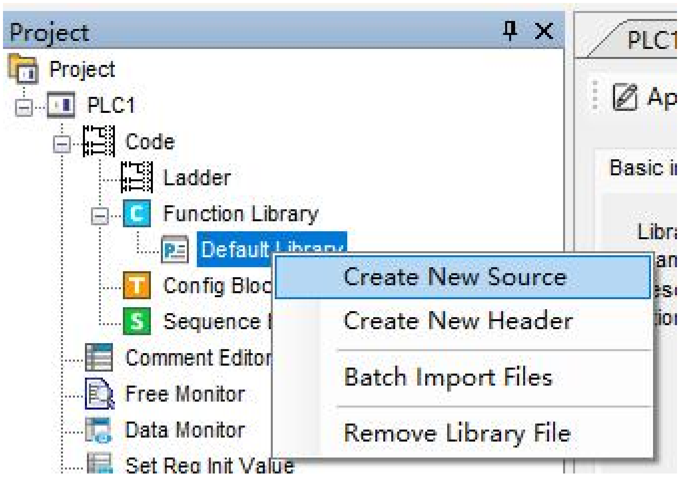
Function and Action



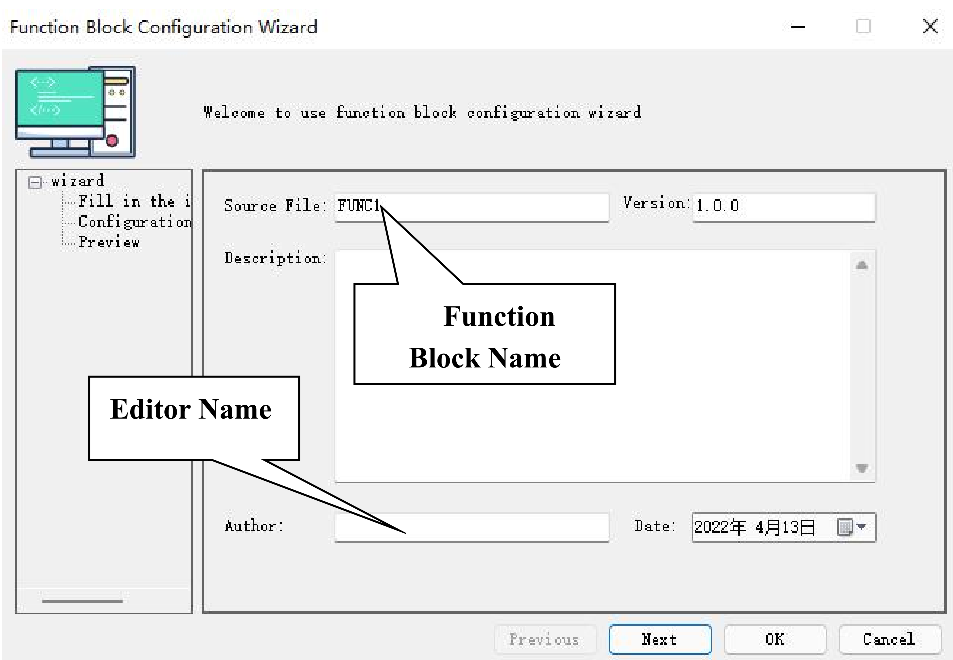
S1 is the function name. It consists of numbers, letters and underlines. The first character can't be number, and the name length should be <=9 ASCII characters.
The name can be the same with PLC's self instructions like LD, ADD, SUB, PLSR etc.
The name can't be the same with the function blocks existing in current PLC;

8-3 Operation Steps

1. Open PLC edit tool, in the left "Project" toolbar, choose "Func Block", right click it and choose "Add New Func Block".

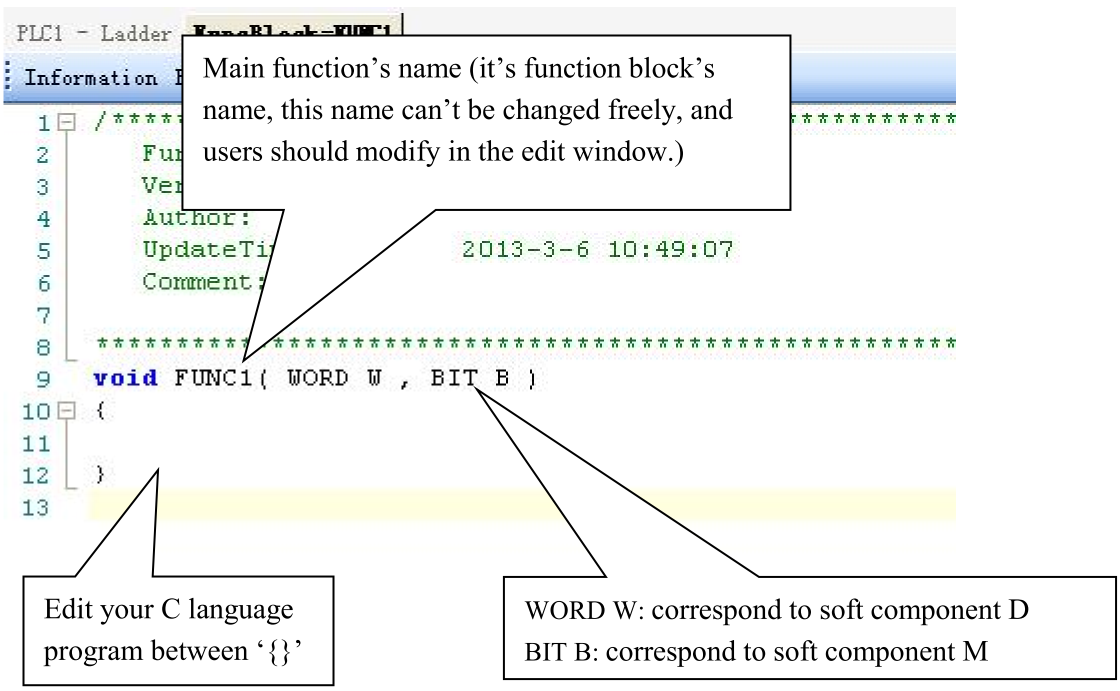


2. See graph below, fill in the information of your function;



Function Block name is the name we use to call the BLOCK. For example: the diagram of FUNC1 should be written as below:

3. After creating the new Function Block, you can see the edit interface as shown below:



- Parameters' transfer way: if call the **Function Block** in ladder, the transferred D (HD) and M (HM) 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 the parameters in the ladder are HD0, HM0, then W[0]=HD0,B[0]=HM0; if the parameters in the ladder are D100, HM100, then W[0]=D100, B[0]=HM100. So, word and bit components start address are defined in PLC program by the user.

-

Note: The local variable defined inside the C function cannot be more than 100 words.

- Parameter **W**: represent **Word** soft component, use it in the form of data group. E.g W[0]=1; W[1]=W[2]+W[3]; in the program, use soft components according to standard C language rules.
- Parameter **B**: represent **Bit** soft component, use it in the form of data group. Support **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 double-word W[10]W[11]. Double-word operation: Support the definition of floating variable in the function, and execute floating operation;(E.g: float register D0(double word) means FW[0], FW[0]=123.456)
- Other soft elements definition in C language:

When a function block is created, #define SysRegAddr_HD_D_HM_Mis default defined in the main function. If you need to use input (X) and output (Y), you need to add X, Y in the default Macro definition "#define SysRegAddrHD_D_HM_M", which will be "#define SysRegAddrHD_D_HM_M_X_Y". For example, set X0 state to coil M0, B[0]=X[0]; set Y0 state to coil M10, B[10]= Y[0]. (Note: The corresponding X and Y are expressed in decimal rather than octal in C language).
Similarly, the applications in C are same for non-power off memory process S, counter C, timer T, counter register CD, timer register TD, register D (HD) and coil M (HM), etc. Macro definition "#define SysRegAddr_S_C_T_CD_TD_D_M". If they are power off memory process HS, counter HC, timer HT, counter register HCD, timer register HTD, etc, Macro definition "#defineSysRegAddr_HS_HC_HT_HCD_HTD".
Examples: W[0]=CD[0];W[1]=TD[0];B[1]=C[0];B[2]=T[0];
Note: Software component types are supported except SEM.

- When the function block is created, default define #define SysRegAddr_HD_D_HM_M in the main function.

```
******************************************************
void FUNC1( WORD W , BIT B )
{
      #define SysRegAddr_HD_D_HM_M


}
```

  It is recommended to use it as a local macro definition, that is, inside the function body.
- Function Library: The user function block can directly use the functions and constants defined in the function library. See chapter 8-10 for the functions and constants contained in the function library.
- The other data type supported:

BOOL; //BOOL Quantity
INT8U; //8 bits unsigned integer
INT8S; //8 bits signed integer
INT16U //16 bits unsigned integer
INT16S //16 bits signed integer
INT32U //32 bits unsigned integer
INT32S //32 bits signed integer
FP32; // single precision floating
FP64; //double precision floating

Examples: #defineDHD*(INT32S*)&HD //DHD means double word HD
#define FFW*(FP64*)&D //FFW means double precision floating numbers
#define DDW*(long long*)&D //DDW means four words register

Explanation: DHD is 32-bit signed integer. DHD[0] represents a 32-bit signed integer power-off holding register composed of HD0 and HD1.

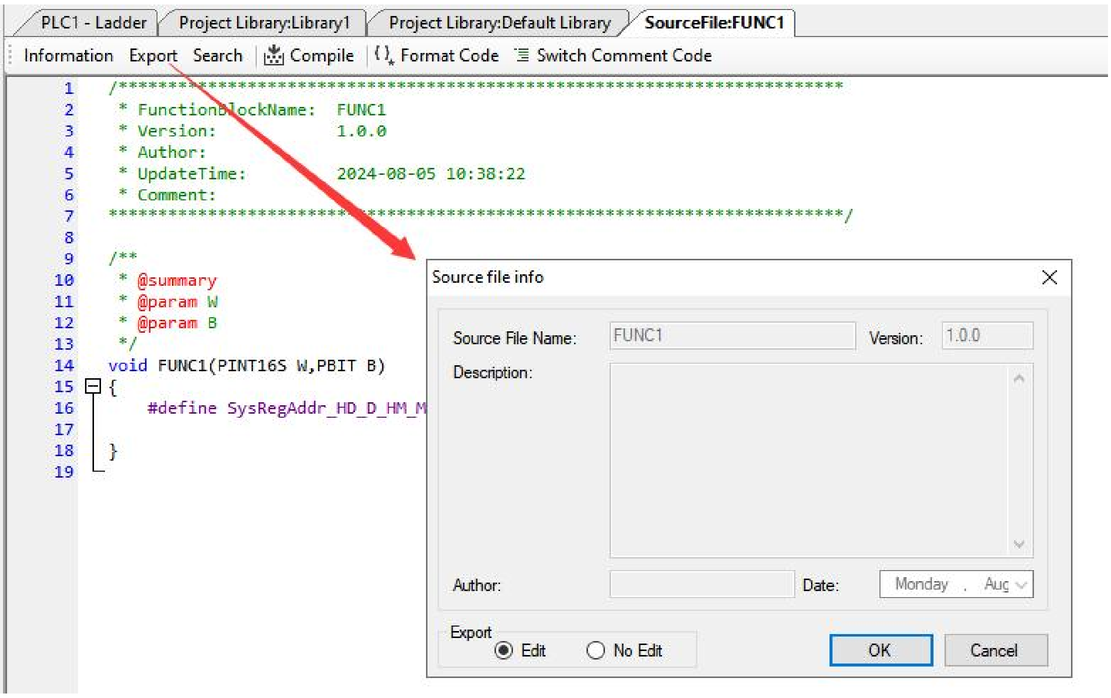
Predefined macros: #define true 1
#define false 0
#define TRUE 1
#define FALSE 0

- There is no non editable option for the export of header files, others are the same as the source files.
- In C, there are two rules for referencing header files, #include “xx.h”and #include <xxx.h>. when using the header file in the PLC project, it needs to use #include ”xxx.h in source file.
- Do not use Marco definition #define SysRegAddr in the header file, this Marco definition is ineffective in the header file, which only can be used in source file.
- Definition types that conflict with PLC standards cannot be defined in the C function block, and secondary definition of W, B, DW, FD, SFD cannot be performed, otherwise the compilation cannot pass.
- The C function block does not support the function of composing words with soft components, such as DX0 and DM0.

8-4 Import and Export the Functions

1．Export

(1) Function: Export the function as the file, then other PLC program can import to use;



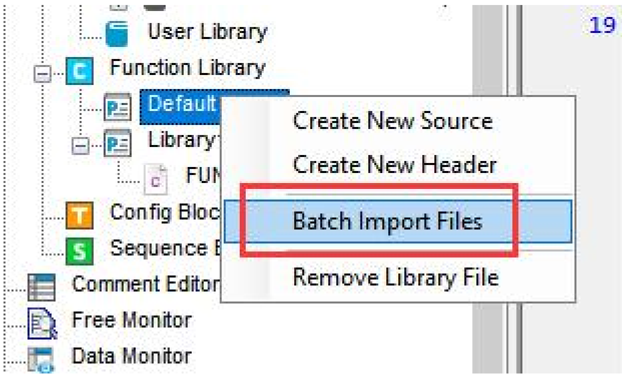
(2) Export Format

a) Edit: Export the source codes out and save as a file. If import again, the file is editable;

b) No edit: Don't export the source code, if import the file, it's not editable. Ethernet models and non Ethernet models cannot be used in common. You only need to modify the model before exporting it.

2. Import

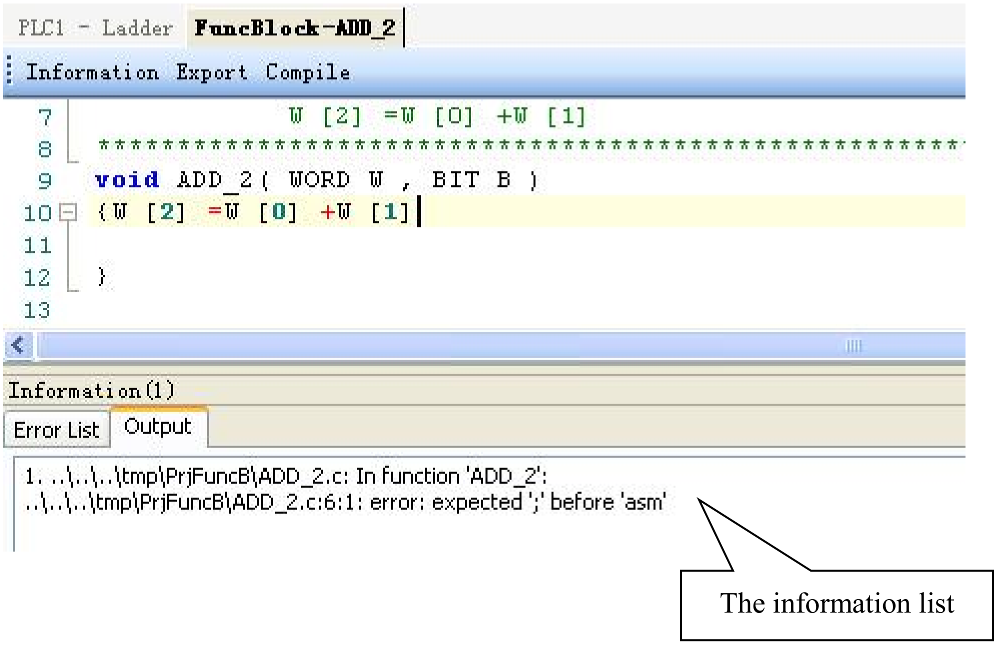
Function: Import the existing **Func Block**file, to use in the PLC program.



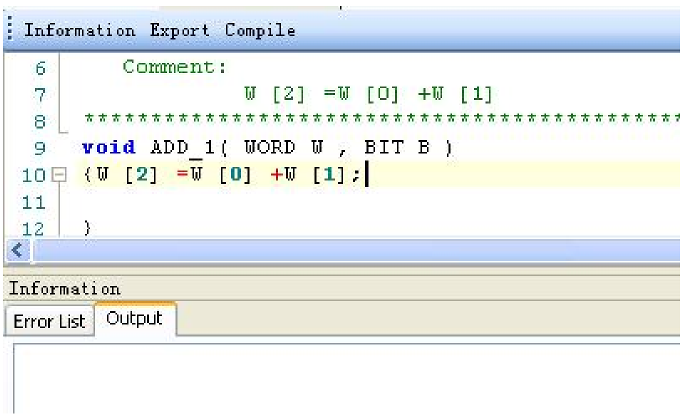
Choose the **Func Block**, right click 'Batch import files, choose the correct file, and then click OK.

8-5 Edit the Func Blocks

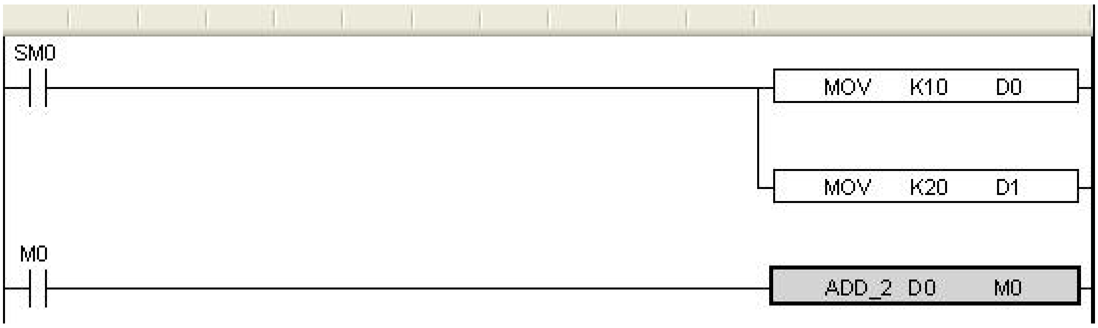
Example: Add D0 and D1 in PLC's registers, and then assign the value to D2;
(1) In 'Project' toolbar, new create a **Func Block**, here we name the **Func Block** as **ADD_2**, then edit C language program;
(2) Click 'compile' after edition.



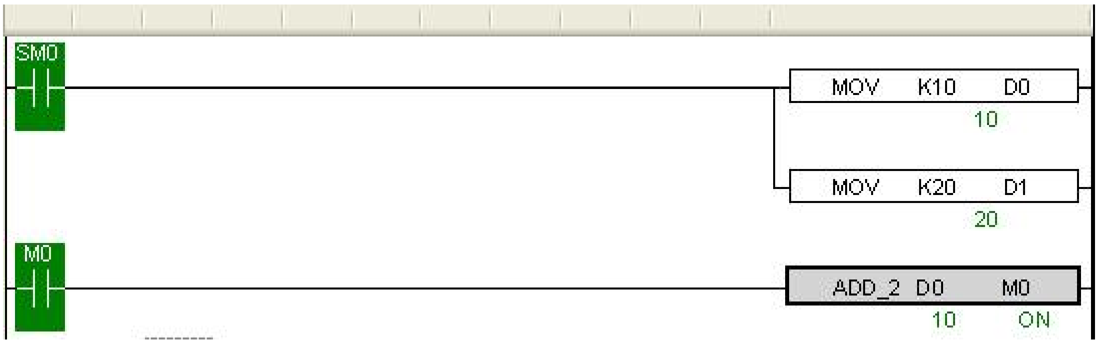
According to the information shown in the output blank, we can search and modify the grammar error in C language program. Here we can see that in the program there is no ';' sign behind W [2] =W [0] + W [1].
Compile the program again after modifying the program. In the information list, we can confirm that there is no grammar error in the program.



(3) Write PLC program, assign value 10 and 20 into registers D0, D1 separately, then call Func 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 assignment is successful;

8-6 Program Example

If PLC needs to do complicated calculation (including plus and minus calculation), the calculation will be used for many times, C language function is easy to use.

Example 1:

Calculation a= b/c + b*c+(c-3)*d

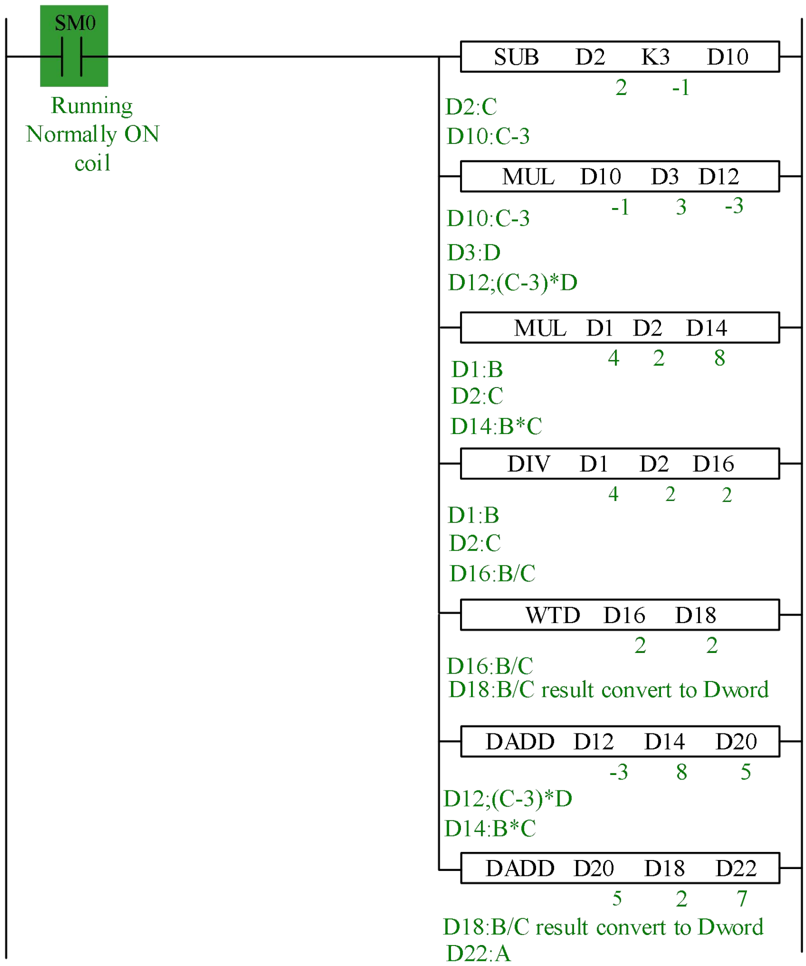
Method 1: use ladder chart:

Get the result of c-3

Get the result of three multiplication equations

Get the sum

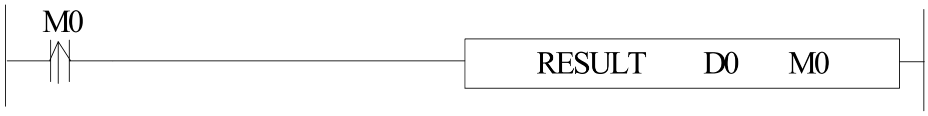
Ladder chart only support two original operands, it needs many steps to get the result.



Note:

1. The result of MUL is Dword, the result is stored in D14~D15.
2. The result of DIV has quotient D16 and remainder D17. If D17 has value, the calculation precision will decrease. Please use float format to ensure the precision.
3. D16 quotient is word value, in plus calculation all the data should be changed to Dword. The final result is stored in D22~D23.

Method 2: use C language:



RESULT	Function name
D0	In the function, W [0] =D0, W [1] =D1… If D0=D32, then W [0] =D32, W [1] =D33… If S2=HD32, then W [0] =HD32, W [1] =HD33…
M0	In the function, B [0] = M0, B [1] =M1… If S2=M32, then B [0] = M32, B [1] =M33… If S2=HM32, then B [0] = HM32, B [1] =HM33…

C program

```
void RESULT( WORD W , BIT B )
{
long int a,b,c,d;;
b=W[1];
c=W[2];
d=W[3];
a=b/c+b*c+(c-3)*d;
DW[4]=a;
}
```

Method 2 can simplify the program.
The above C language function is similar to ladder chart of method 1, whose precision is not high. If it needs to get the high precision, please use float calculation.

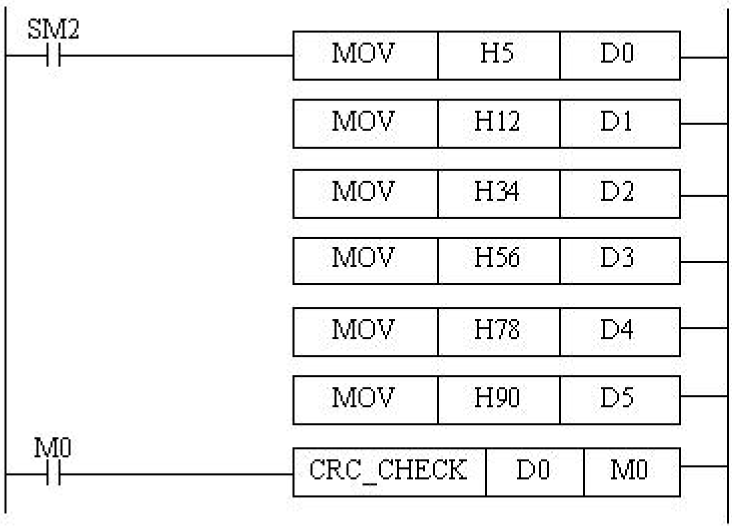
Example 2: Calculate CRC parity value via Func Block
CRC calculation rules:
(1)Set 16-bit register (CRC register) = FFFF H
(2)XOR (Exclusive OR) the first 8-bit byte message and the low 16-bit CRC register.
(3)Right shift 1 bit of CRC register, fill 0 into the highest bit.
(4)Check the right shifted 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 then save the result into the CRC register.
(5)Repeat step3&4 until all the 8-bit have been calculated.
(6) Repeat step(2)~(5), then calculate the next 8-bit message. Until all the messages have 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[n] = k>>8;
        W[m]=reg_crc&0xff;
       }
```

Edit PLC ladder program,

D0: Check byte number of data,

D1~D5: Check data content. See graph below:

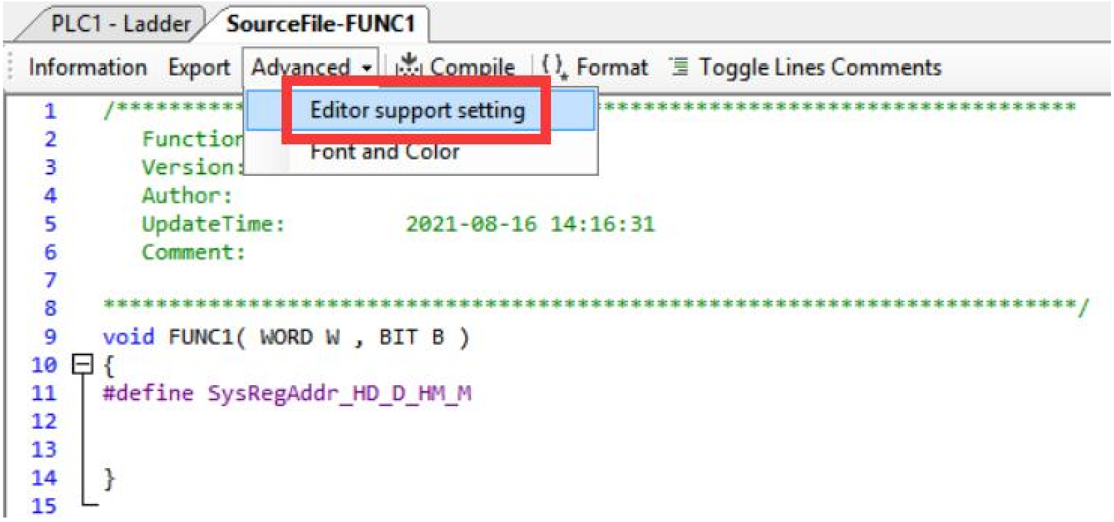


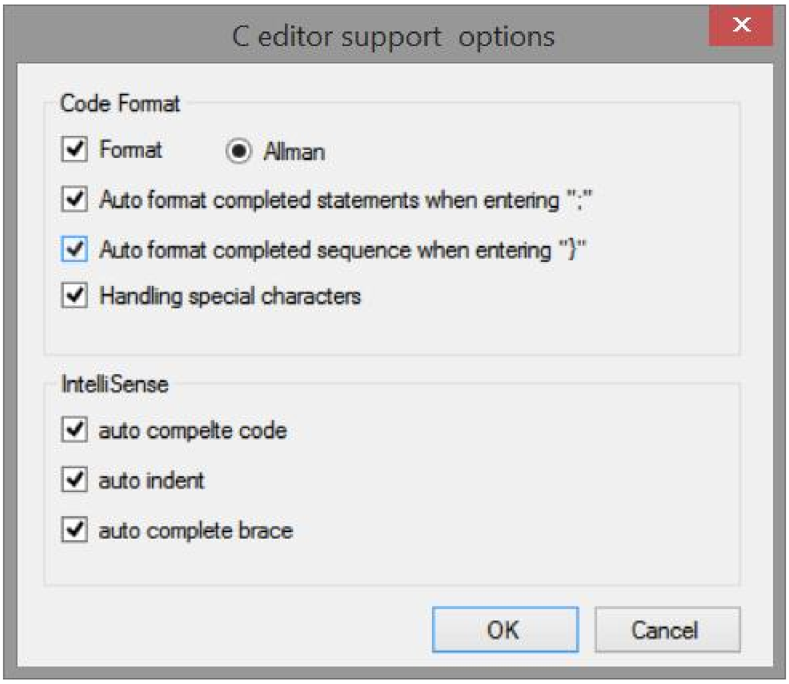
Download to PLC, then RUN PLC, set M0, via Free Monitor, we can find that values in D6 and D7 are the highest and lowest bit of CRC parity value;

8-7 New functions

(1) Format

Click the advanced/editor support setting menu to open the C editor support options window.





(2) Local code auto format

➢ Auto format completed statements when entering ';"

When the user enters the character ";" format the statement of the current row.

➢ Auto format completed sequence when entering "}"

When the user enters "}", format the contents in "{}".

(3) Handling special characters

The full width characters entered by the user into the editor need to be converted to half width characters because they are not recognized by the compiler.

(4) Auto complete code

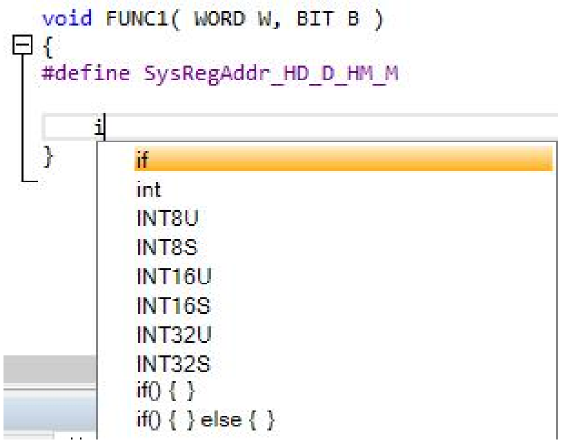
When the user inputs characters, the code prompt function will give certain prompts to help the user input and complete the code.

➢ Submit

When the user press Enter or “;”, the currently edited code will be submitted to the analyzer for analysis and a list of code tips will be generated.

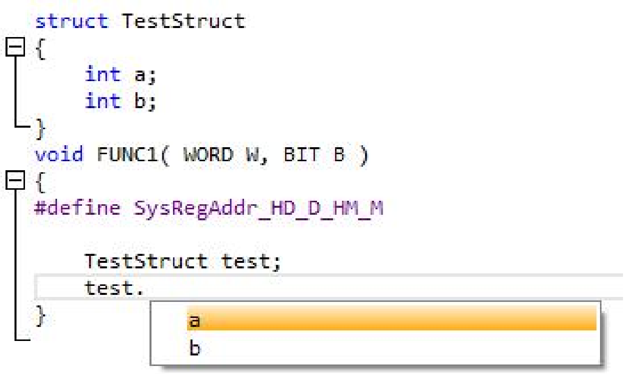
➢ Prompt

When the user inputs characters, the code prompt control will pop up automatically to match the user's input and give a prompt.



➢ Access tips for member variables

When the user enters "." "or" - > ", the code prompt function will help the user prompt the members in the structure or consortium type of the defined variable, as shown in the following figure.



➢ Auto indent

The automatic indentation function of the editor is optimized, which is more in line with user habits.

➢ Auto complete brace

When the user enters "(" ["{", it will automatically help the user generate the corresponding bracket ")"] "}".

(5) Comment / uncomment

Comment selects / deselects the comment for the row.

The shorcut key is Ctrl +/.

(6) Function library

Please refer to chapter 8-8.

8-8 Function library

It provides the functions of encryption, encapsulation, export and import of C function blocks.

8-8-1 New function

8-8-1-1 Classification of Libraries

Function library are divided into project library and global library.

Project library: the functions in the user's project library are saved under the project and can be used directly.

Global Library: the function functions in the user's global library are saved in the local directory for user's convenience.

8-8-2 Basic functions

8-8-2-1 Open and save file

Start XDPPro software, run a blank project or open any existing project to view the function library.

Notes:

The function library is divided into project library and global library. A default library (i.e. project library) is added to the blank project by default;

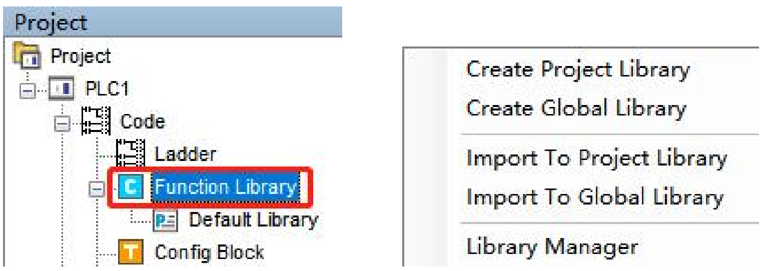
If the project under the old version is opened with the new version, its function function is added to the default library;

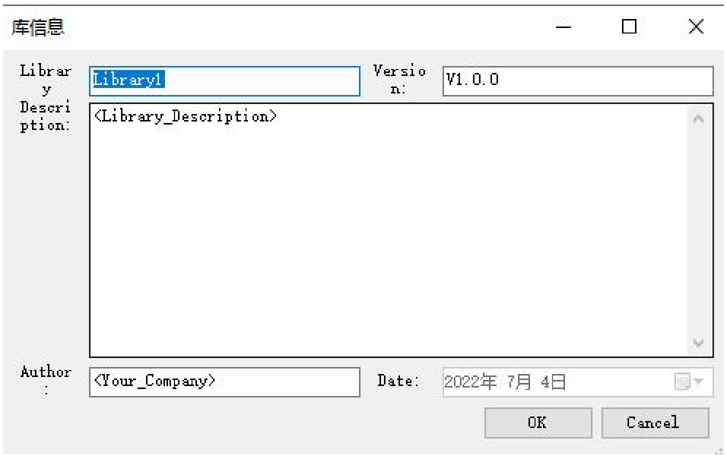
If the project under the new version is opened with the old version, the function functions in the default library are retained, and the rest cannot be parsed.

8-8-3 Newly build

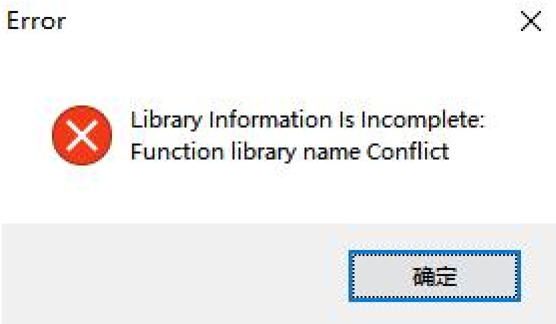
8-8-3-1 Create project library

Select "Function library" in the "Project" toolbar on the left, right-click and select "Create Project Library", and you can edit the name, version, description, author and other information of the project library in the pop-up interface, as shown in the following figure:



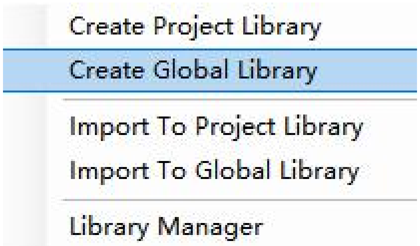


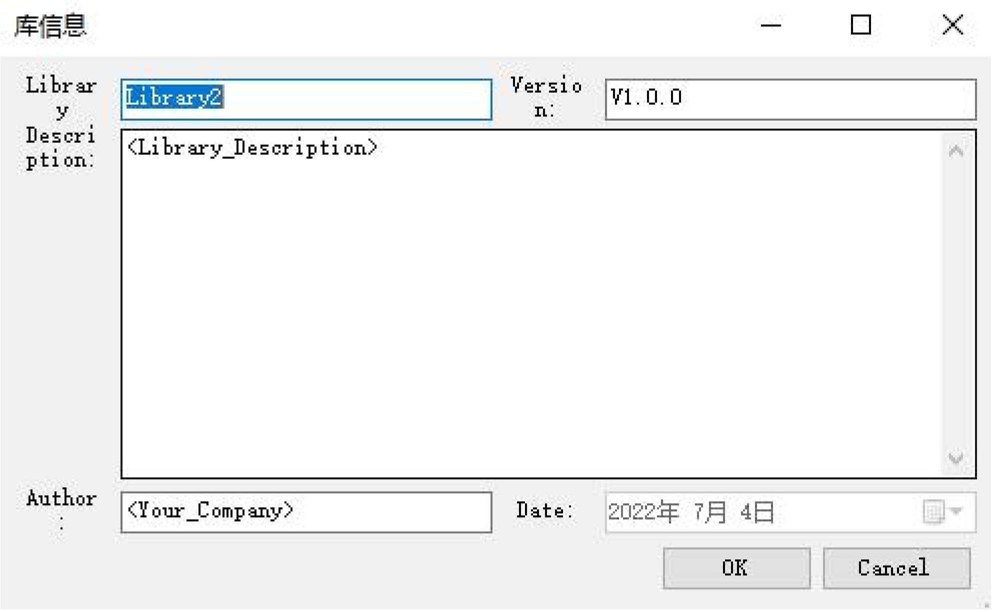
Note: if the library name is the same as any library name in the current library, the following pop-up window will appear:



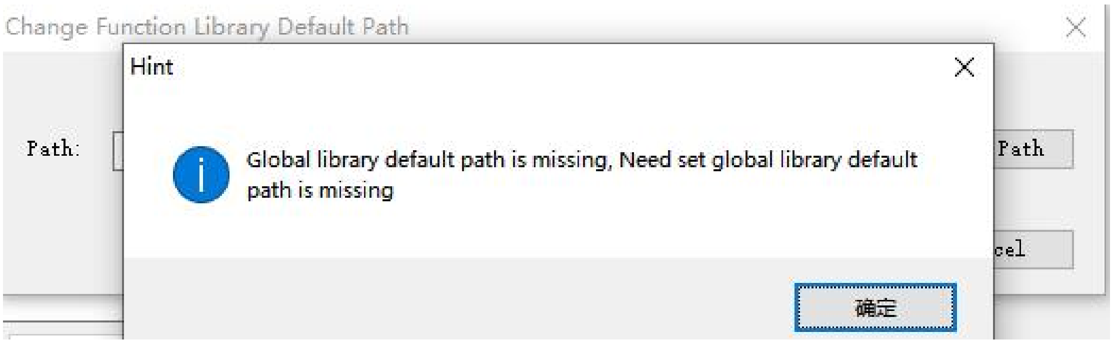
8-8-3-2 Create global library

Select "Function library" in the "Project" toolbar on the left, right-click and select "Create Global Library", and you can edit the name, version, description, author and other information of the global library in the pop-up interface, as shown in the following figure:





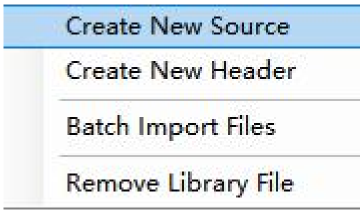
Note: if the global library directory is not set, the prompt message shown in the following figure will appear and the Global Library Directory setting interface will be displayed:

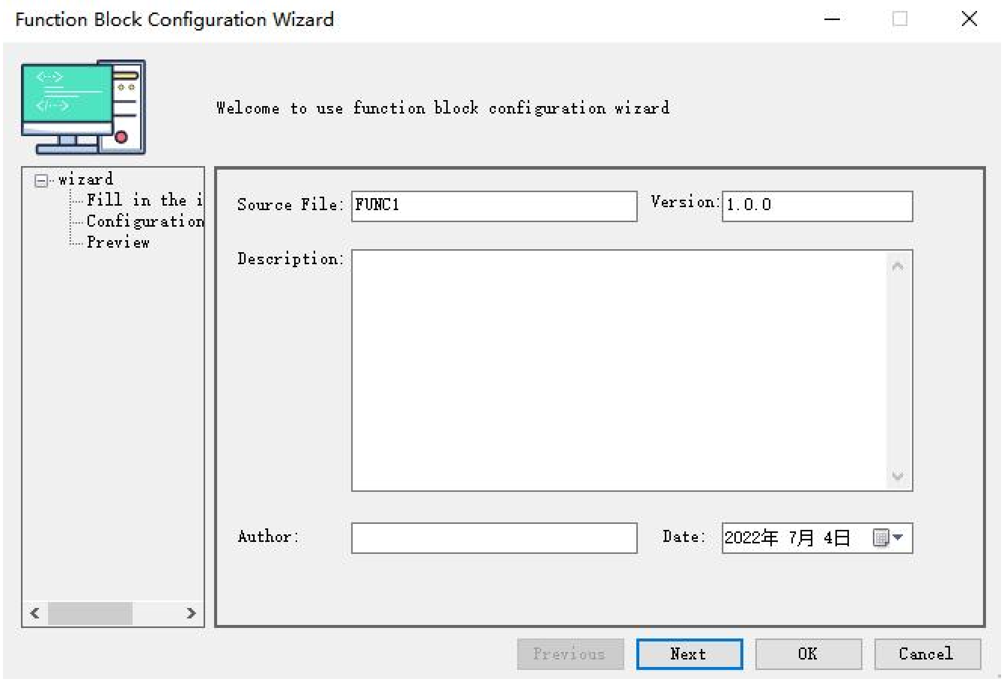


After setting the path, the new library file window is displayed, and the library information (name, version, description, author) is filled in. If the library name is the same as any library name in the current library, the following pop-up window will appear:

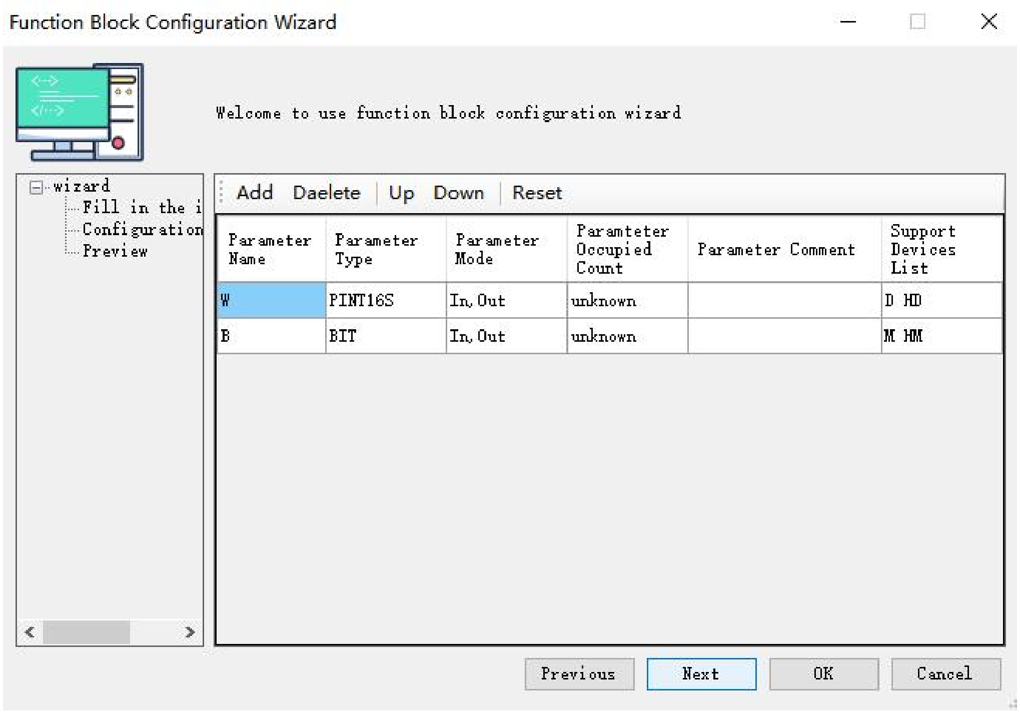
8-8-3-3 Create new source

In the "project" toolbar on the left, select the project library or global library to which the source file needs to be added in the "function library", right-click and select "new source file" to edit the name, version, description, author and other information of the source file in the pop-up interface, as shown in the following figure:

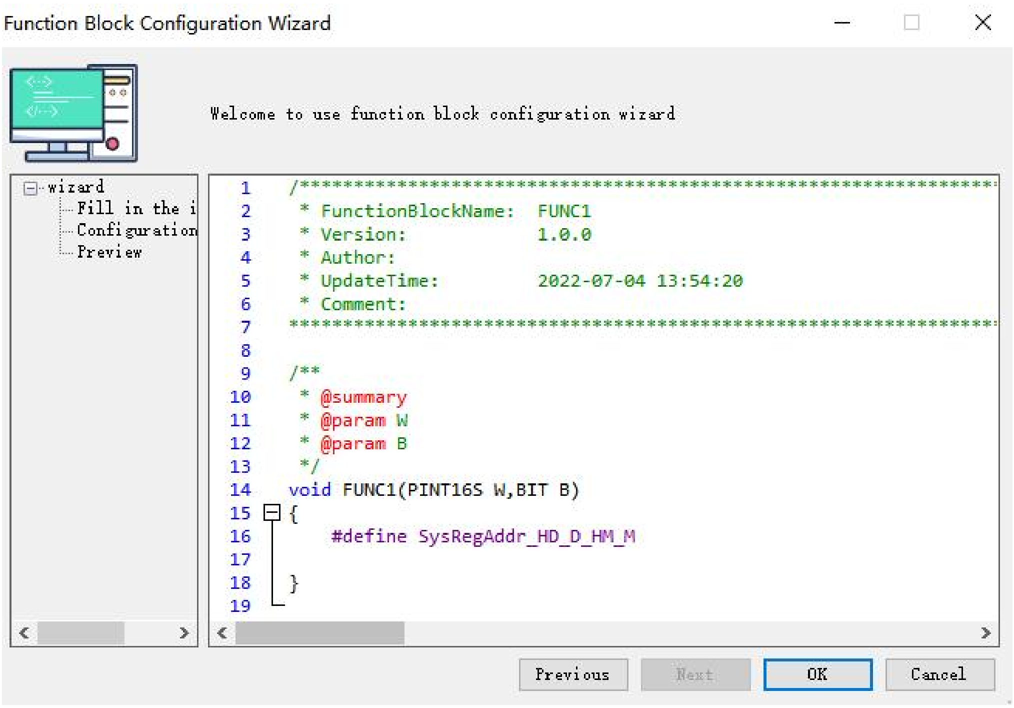




Click "next" after filling in to configure parameter information:

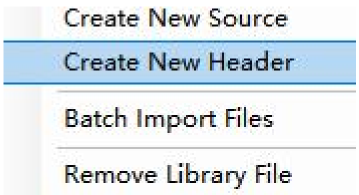


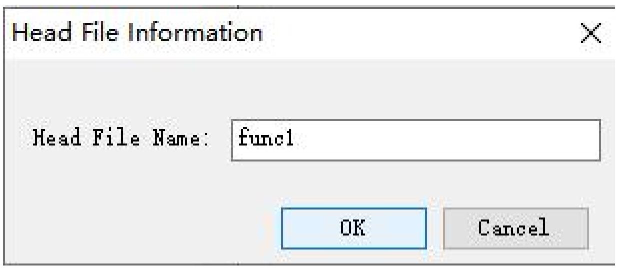
After completing the parameter configuration, click "next" to display the preview interface of the source file. If there is a problem, click "previous" to reset the parameters. If there is no problem, click "OK" to complete the addition of the source file.



8-8-3-4 Create new header

In the "Project" toolbar on the left, select the project library or global library to which the source file needs to be added in the "Function Library", right-click and select "New header file" to edit the name of the header file in the pop-up interface, as shown in the following figure:

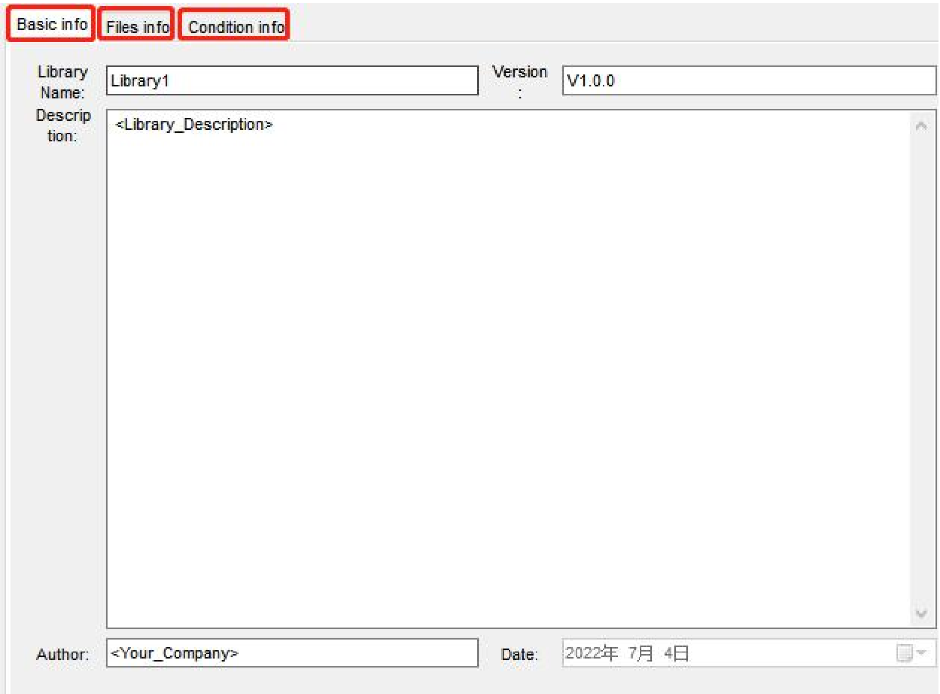




8-8-4 Edit

8-8-4-1 Edit library information

Click "project library" or "global library" in the project bar on the left to edit the information, and you can view and edit the basic information / file information / restriction information of the library in the pop-up library information interface:

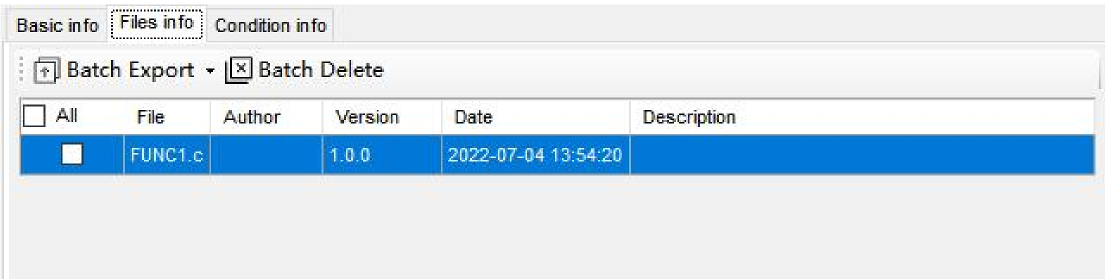


1) Basic information

Library name: only letters and numbers are allowed for the library name.

Version: the format of the library information version is "V primary. Secondary. Revision".

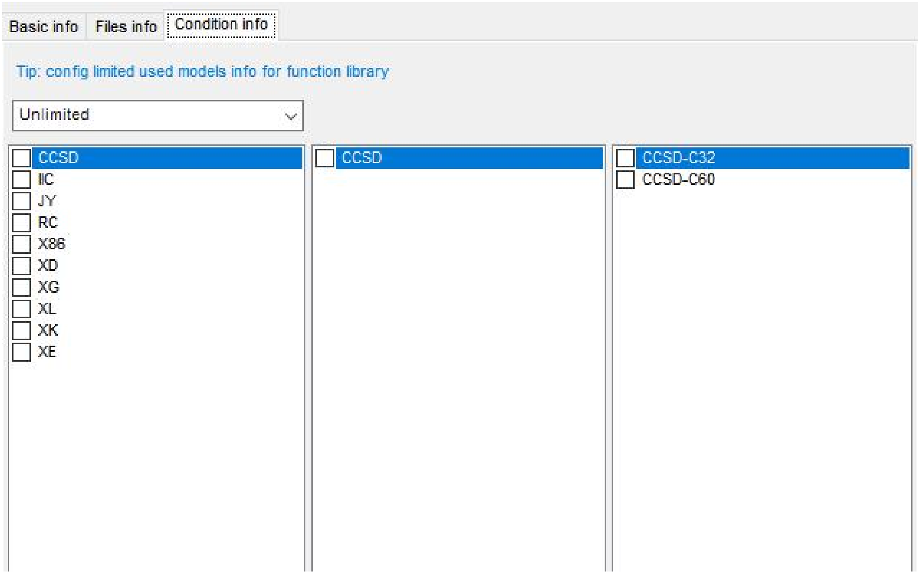
2) Files information



- Add the source file / header file under the selected function library, and the file information interface displays the basic information of the file.
- The imported file determines whether the user can edit it.
- The files exported in batch can be edited or not.
- After deleting the application in batch, remove the reference of the library file in the PLC project.

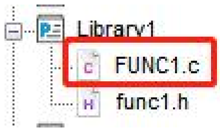
3) Condition information

Models under the blacklist cannot be used, and only those models under the whitelist can be used.

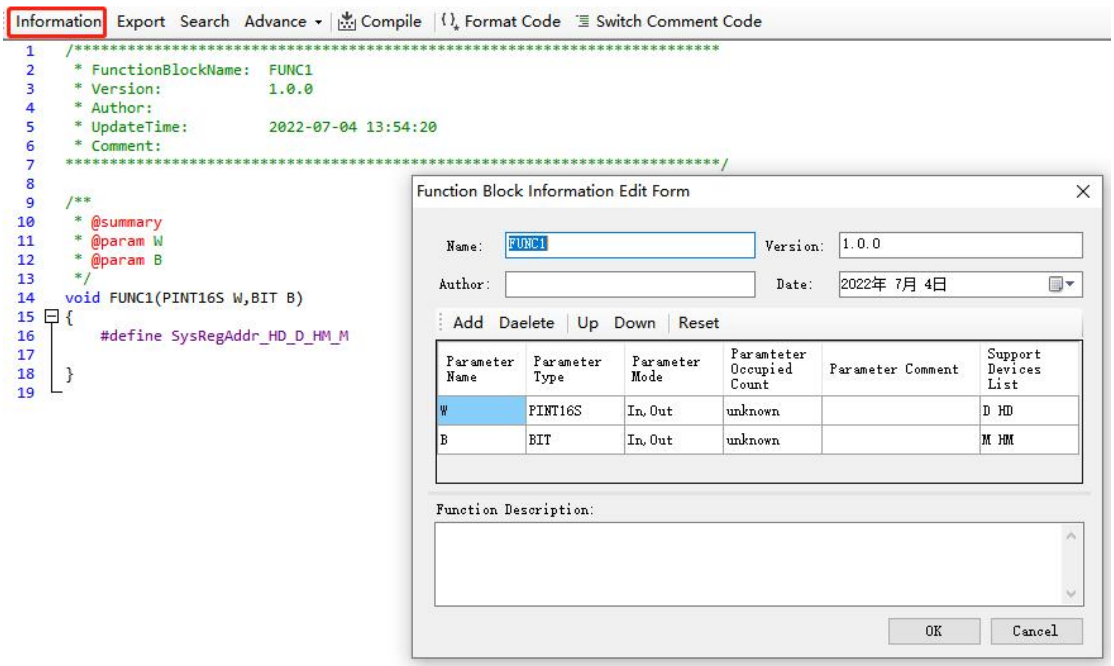


8-8-4-2 Source file information

Click the source file to edit information in the project bar:



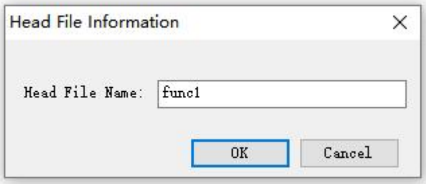
In the pop-up source file interface, click information to modify the source file information, the source file function signature is modified, and the code is modified accordingly.



8-8-4-3 Header file information

Click the header file to edit information in the project bar:

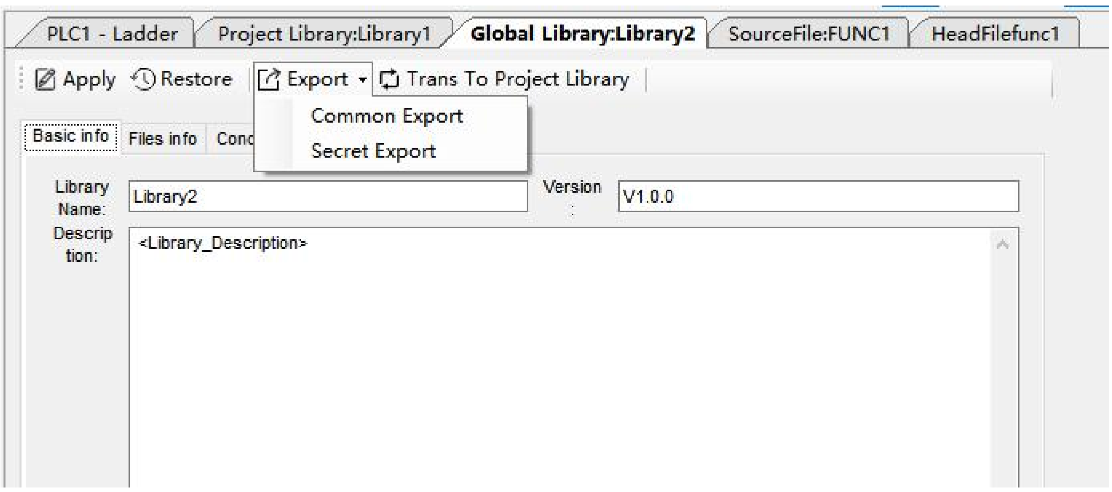
In the pop-up header interface, click information to modify the head file information, the header function signature is modified, and the code is modified accordingly.



8-8-5 Export

8-8-5-1 Export the function library

Click "Project library" or "Global library" in the project bar on the left to edit the information, and click "Export" in the pop-up library information interface:

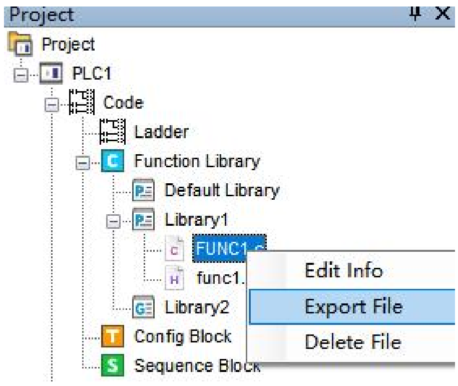


Normal export: if the library file is an editable library, export it with an editable library; If the library file is a non editable library, export it as a non editable library.

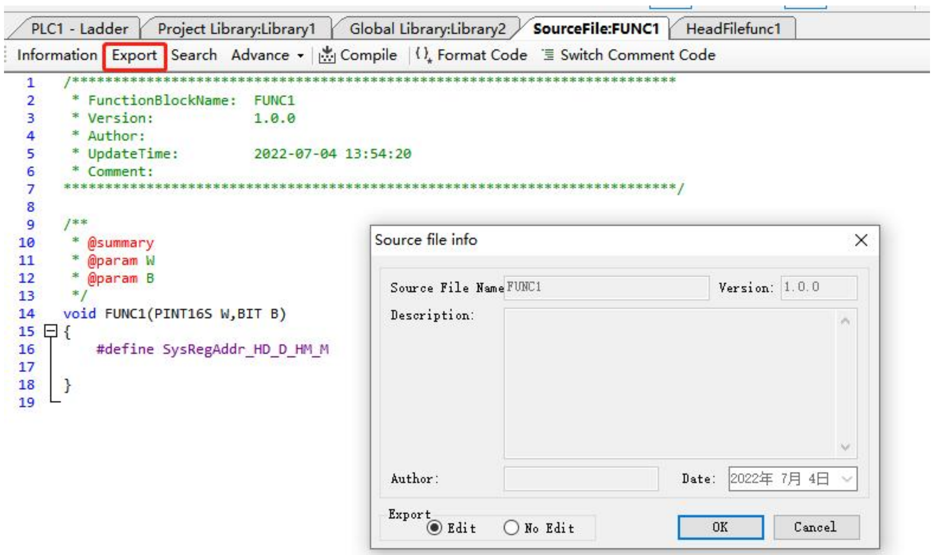
Encrypted export: if the library file is editable, the source file in the library file is compiled and exported as a non editable library; If the library file is non editable, save the library file directly.

8-8-5-2 Export source/header file

Right click the source file / header file to be exported in the project bar --> Export file:



Or click the source file / header file to be exported in the project column on the left, and click "export" in the editing interface on the right:



Select the export mode (editable or not) in the pop-up file information.

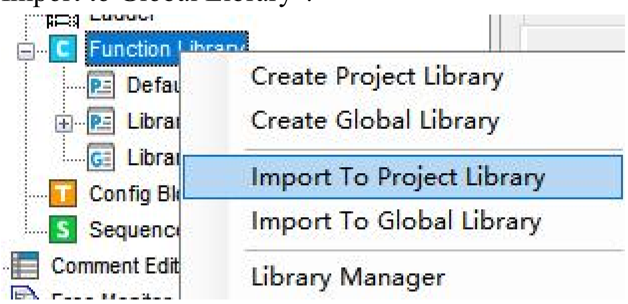
Click OK after setting and select the file saving path.

After selecting the path, click OK to complete the export.

8-8-6 Import

8-8-6-1 Import the function library

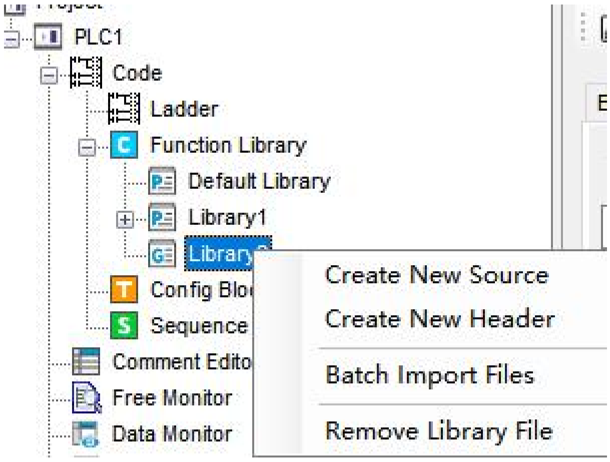
Select "Function Library" in the "Project" toolbar on the left, right-click and select "Import to Project Library" or "Import to Global Library":



In the pop-up "select function library file" interface, select a file and click "open" to complete the import.

8-8-6-2 Import function files

Right click the "Project Library" or "Global Library" in the project bar on the left to import files, and select "Batch import files":

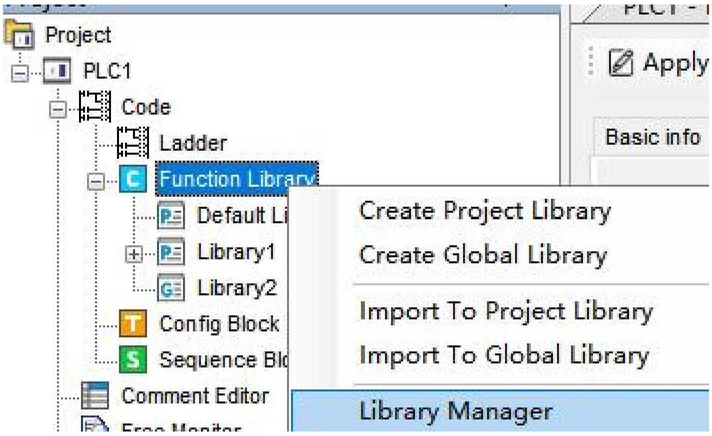


Select the function file to be imported in the "select file" interface, and click "open" to complete the file import.

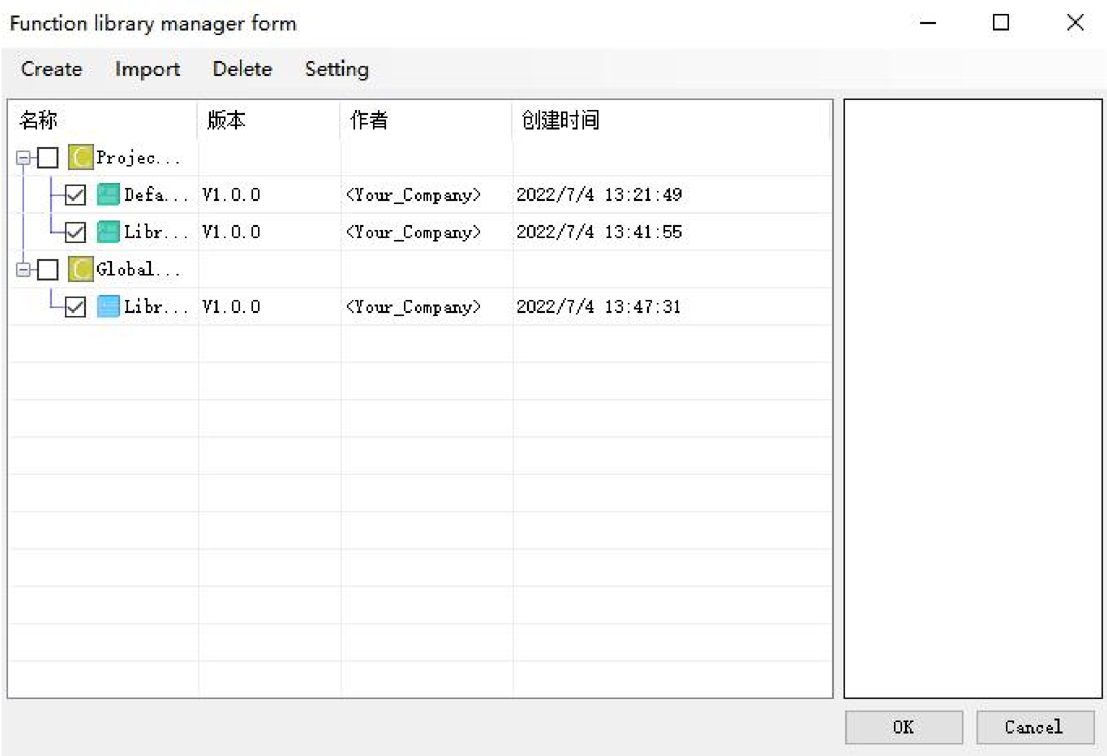
8-8-7 Other functions

8-8-7-1 Library manger

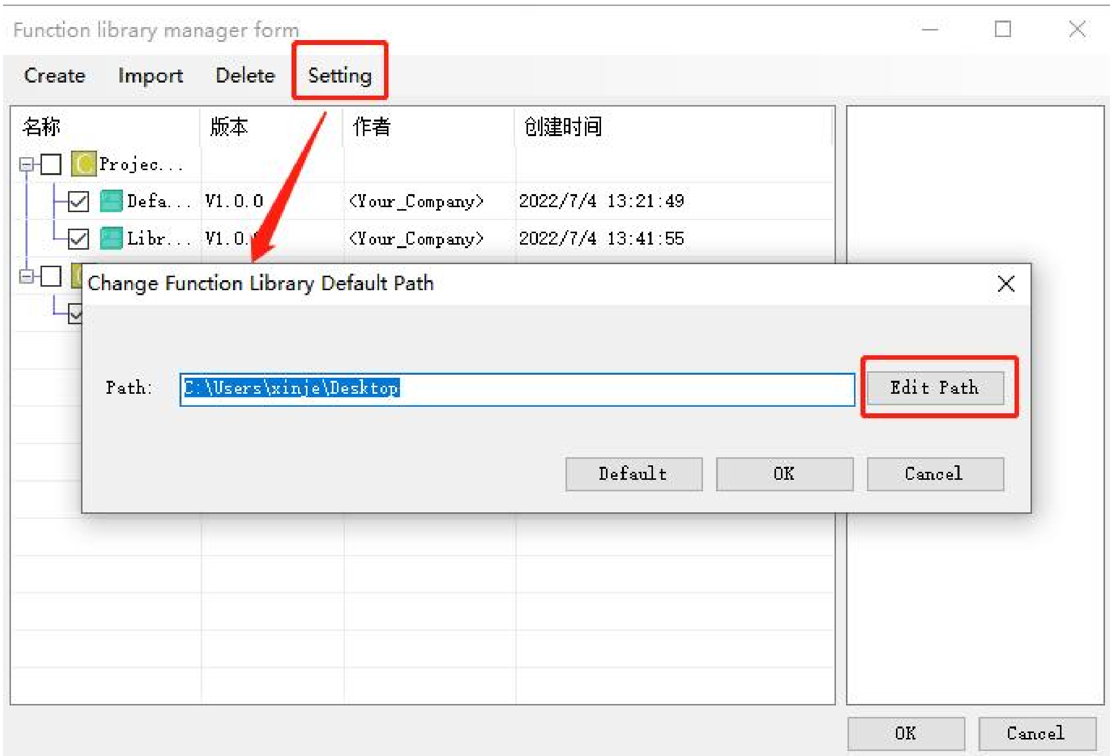
Select "Function Library" in the "Project" toolbar on the left, and right-click to select "library management":



In the pop-up "Function Library management window", you can complete the creation, import, deletion (and removal of library files referenced in the project) and setting of the function library. By checking the function library in the management, you can apply generation, and then call it in the project.

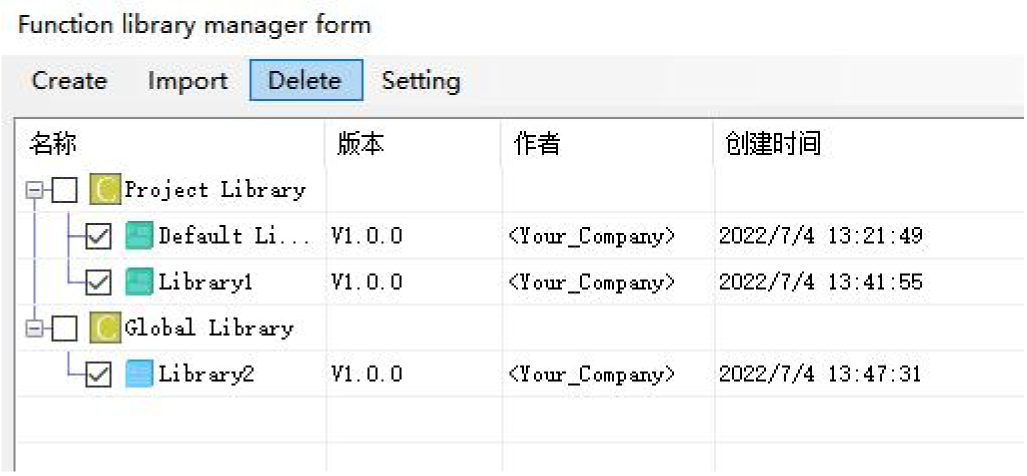


Click settings to change the Global Library Directory:



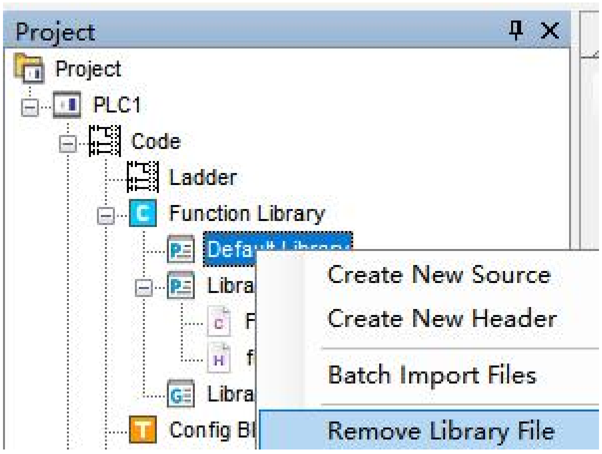
8-8-7-2 Delete library file

In the "function library management window" of the previous chapter, check the corresponding library file and click Delete to delete the library file in the current project.



8-8-7-3 Remove library file

Right click the "Project Library" or "Global Library" in the project bar on the left to import the file, and select "Remove Library file":

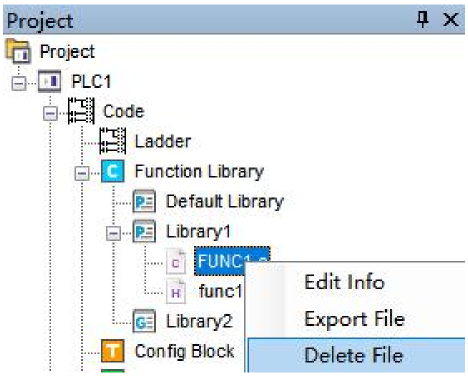


Note: Remove the library file means to cancel the application of the file from the current project without deleting it.

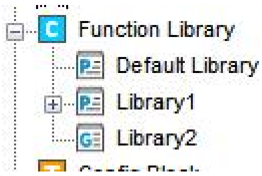
8-8-7-4 Delete source/header file

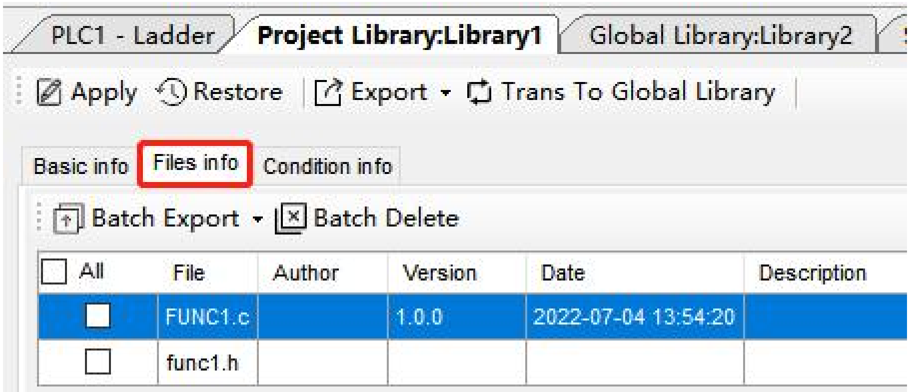
There are two ways to delete source / header files:

Method 1: right click the source file / header file to be exported in the project bar - > delete file:

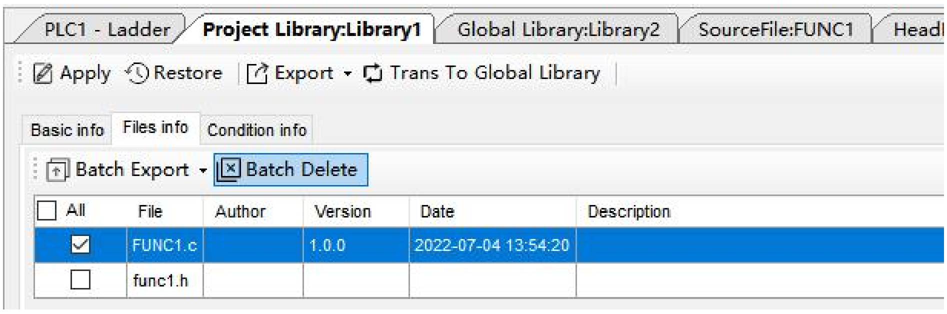


Method 2: click the function library to delete the file in the project bar on the left:

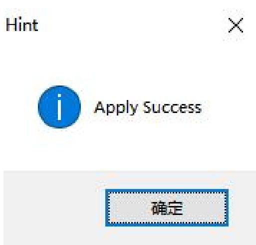




Check the files to be deleted and click "batch delete":

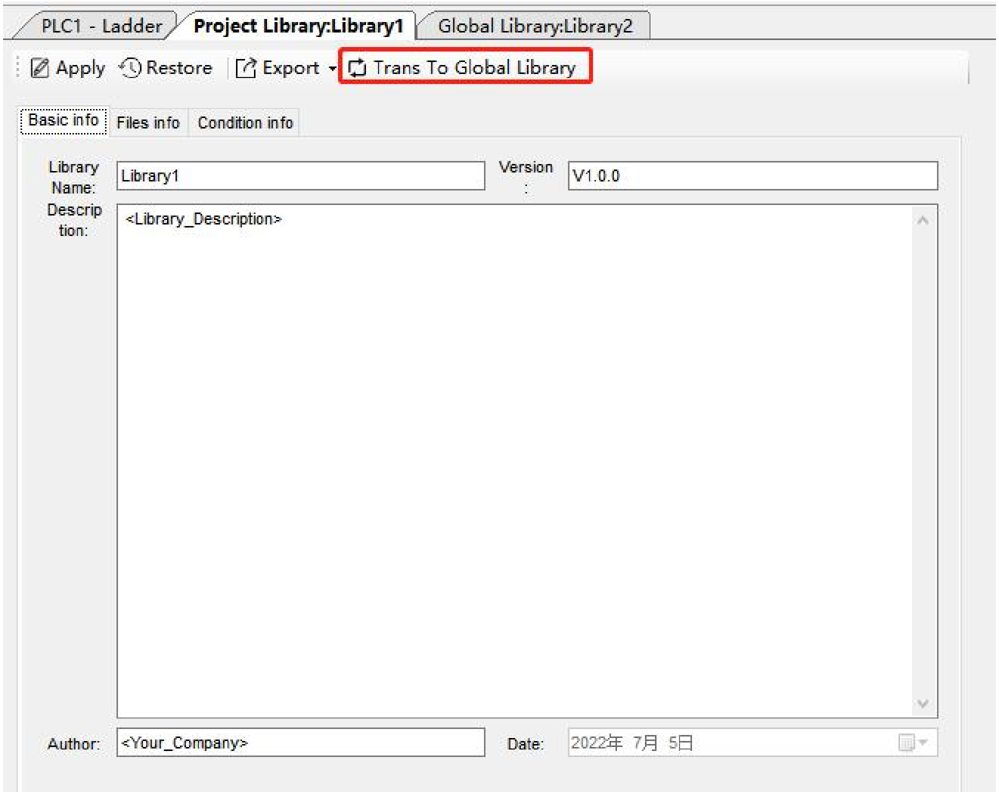


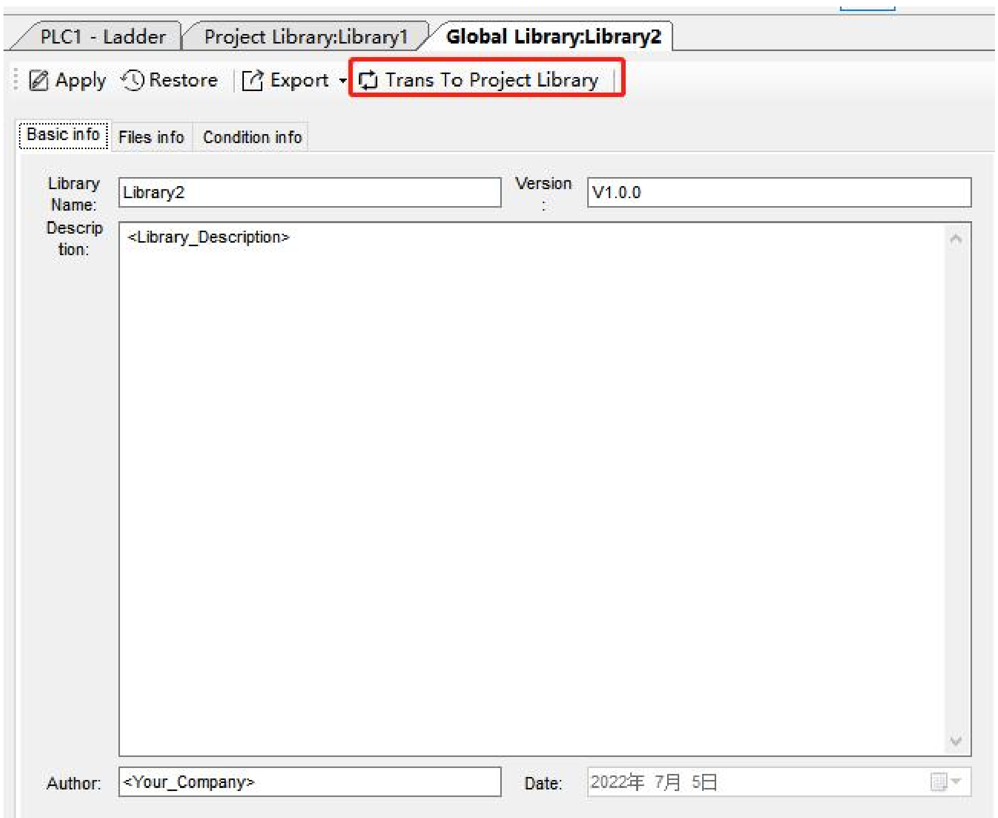
Click "Apply" and a prompt message "successfully applied" will appear. The file has been deleted.



8-8-7-5 Transfer

The "global library" and "project library" can be converted to each other, and the editing interface of the function library can be opened (for specific steps, refer to chapter 8-8-7-4, method 2).


PLC1 - Ladder
Project Library:Library1
Global Library:Library2
Apply
Restore
Export
Trans To Global Library
Basic info
Files info
Condition info
Library Name:
Library1
Version :
V1.0.0
Description:
<Library_Description>
Author:
<Your_Company>
Date:
2022年 7月 5日


PLC1 - Ladder
Project Library:Library1
Global Library:Library2
Apply
Restore
Export
Trans To Project Library
Basic info
Files info
Condition info
Library Name:
Library2
Version :
V1.0.0
Description:
<Library_Description>
Author:
<Your_Company>
Date:
2022年 7月 5日

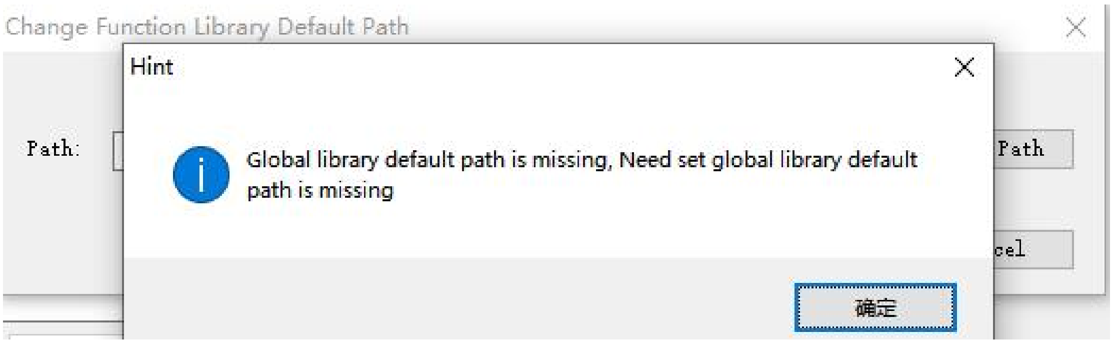
8-8-7-6 Compile

Click the source file in the project bar on the left, and click "compile" in the editing interface on the right.

8-8-7-7 Set Global Library Directory

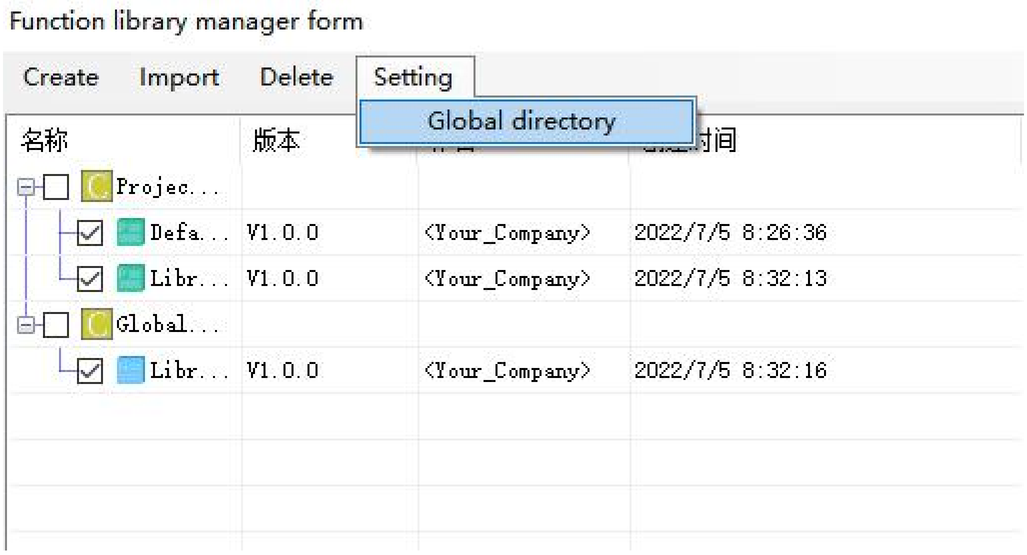
There are three methods to set the global library:

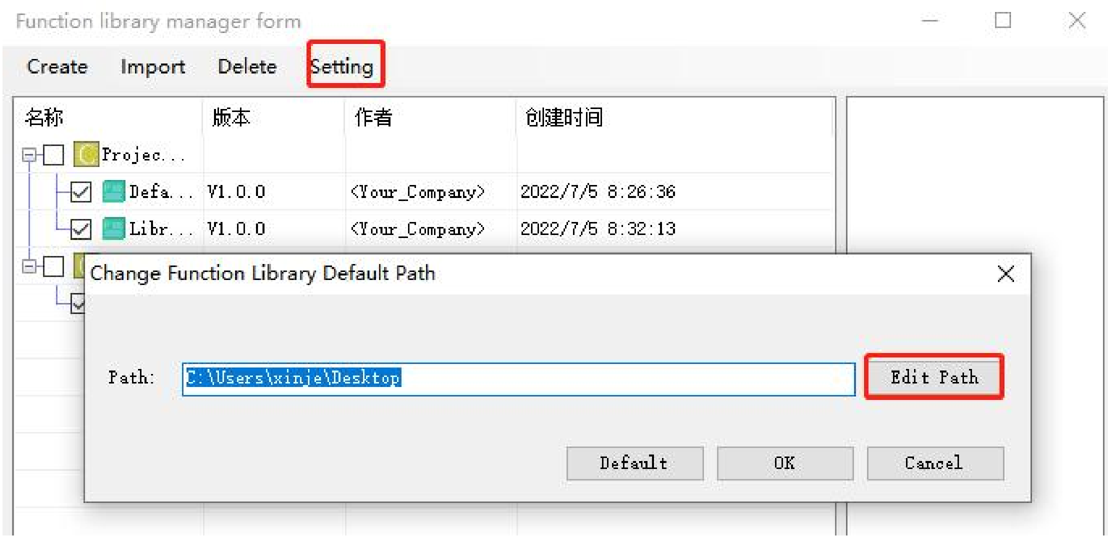
Method 1:Open the library management interface (please refer to 8-8-7-1. Library manager for specific steps). If the global library directory has not been set, the prompt to set the global library directory will appear.



Method 2:In the process of creating a new global library, if the global library directory has not been set, the same prompt as method 1 will appear. You can set the path in the "Change Function Library Default Path" pop-up window.

Method 3:Open "Library manger" interface,please refer to 8-8-7-1. Library manager and click "Settings" - > "Global Library Directory" as shown below:





8-9 Application notes

- In one Func Block file, you can write many functions, and they can be called by each other.
- Each Func Block file is independent, the function in other function block cannot be called.
- Func Block files can call C language library function in form of floating, arithmetic like sin, cos, tan.
- XC series PLC only support local variable, while XD/XL series PLC support both local and global variable. This makes C language Block more flexible and convenient.
- Recommended usage of global variables:
- ① Use the soft component area instead of ordinary memory to store the data of global variables.
- The soft component space of PLC can be used as the global variable space, and the security is guaranteed.
- ② Usage example

Take FP64 type as an example:

```
/*****************************************************************************
    FunctionBlockName:  FUNC1
    Version:            1.0.0
    Author:
    UpdateTime:         2020/1/3 10:30:47
    Comment:

*****************************************************************************/
void Test();
FP64 * GlobalV;          declaration

void FUNC1( WORD W , BIT B )
{
#define SysRegAddr_HD_D_HM_M

GlobalV = (FP64*)(&W[0]);          initialization

Test();
}
void Test()
{
    #define SysRegAddr_HD_D_HM_M
    FP64 value = GlobalV[0];          using
    *(FP64*)&HD[0] = value;
}
```

As shown in the figure above, the global pointer GlobalV is declared outside the function, and then initialized in the main function to point to the space of the software component. The first address of the space is the address where W[0] is located. Finally, the value of the variable can be obtained through pointer operation in other functions.

Take structure type as an example:

```
#ifndef _STRUCT_H
#define _STRUCT_H

typedef struct
{
    INT16U V;
    FP64 S;
}ExStruct;

#endif
```

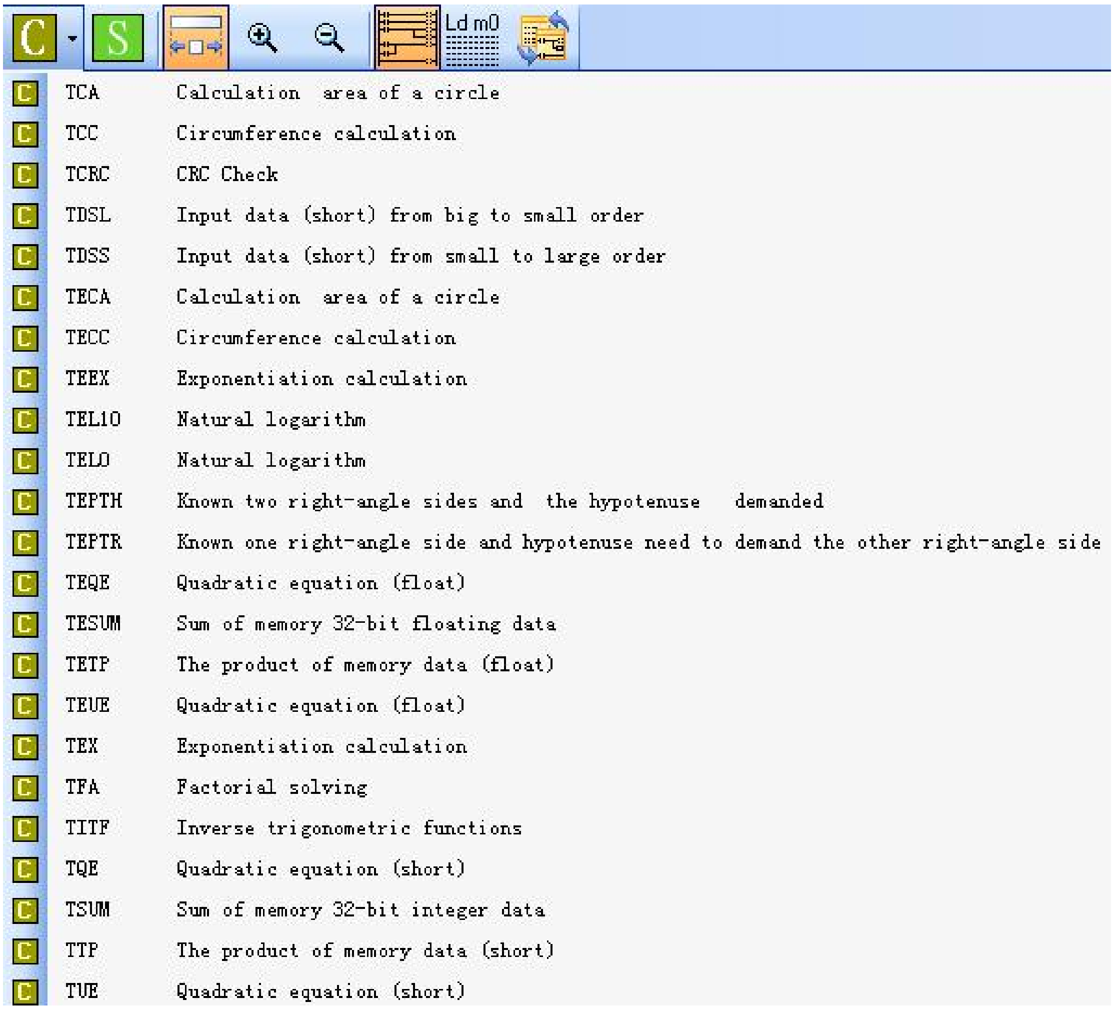
The declaration of structure

```
/*****************************************************************************
    FunctionBlockName:  STRUCT
    Version:            1.0.0
    Author:
    UpdateTime:         2020/1/3 10:58:49
    Comment:

*****************************************************************************/
#include "struct.h"          the header file contained declaration
void Test();
ExStruct* ST;
void STRUCT( WORD W, BIT B )
{

#define SysRegAddr_HD_D_HM_M

ST =((ExStruct*)&W[0]);          initialization

ST->V = 10;
ST->S = 100.001;

Test(ST);
}
void Test(ExStruct* ex)
{
#define SysRegAddr_HD_D_HM_M

*(INT16U*)&HD[0] = ex->V;
*(FP64*)&HD[2] = ex->S;          using
}
```

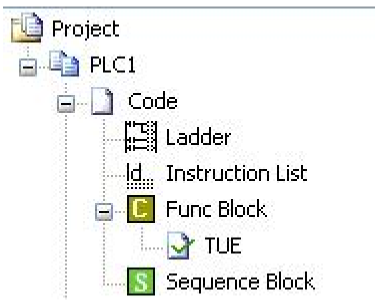
Structure type global variable usage example

- XDPPro software v3.3 and later version keep C function library:

In this function block, user can call the C function directly:



For example: click TUE, the function name will show on the project bar:



User can call it in the ladder chart editing window at any time.

8-10 Q&A of C language

(1)second macro definition for the coil

Some users have further extended the software component type after defining it, as shown in the following code:

```
#define SysRegAddr_HD_D_HM_M_X_Y
#define OUT Y[1]
OUT = 100;
```

The second macro definition of coils such as Y is not allowed because the reading and writing of coil data is not simply a pointer, but through a function. In this case, the compiler cannot handle it, resulting in an error.

(2)Use the value of the coil as the judgment condition

The user uses the value of the coil as the judgment condition of the if statement, as shown in the following code:

```
if(X[0])D[0] =10;
```

This writing method will report an error during compilation because our compiler has made an error during internal processing. It is recommended that you change the line, as follows:

```
if(X[0])
                                D[0] =10;
```

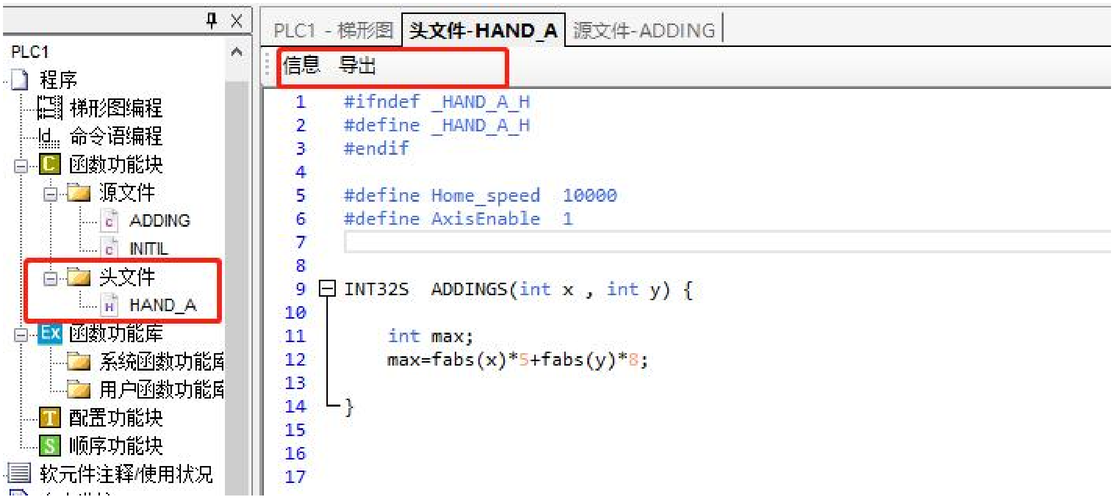
(3)Use DM

DM[0] is not supported at present. Only DW and FW double word operations are supported.

(4)An error is reported during compilation, and macro defintion color changes to black

This phenomenon is caused by full angle characters in the code. Full angle characters can be cleared by using formatting.

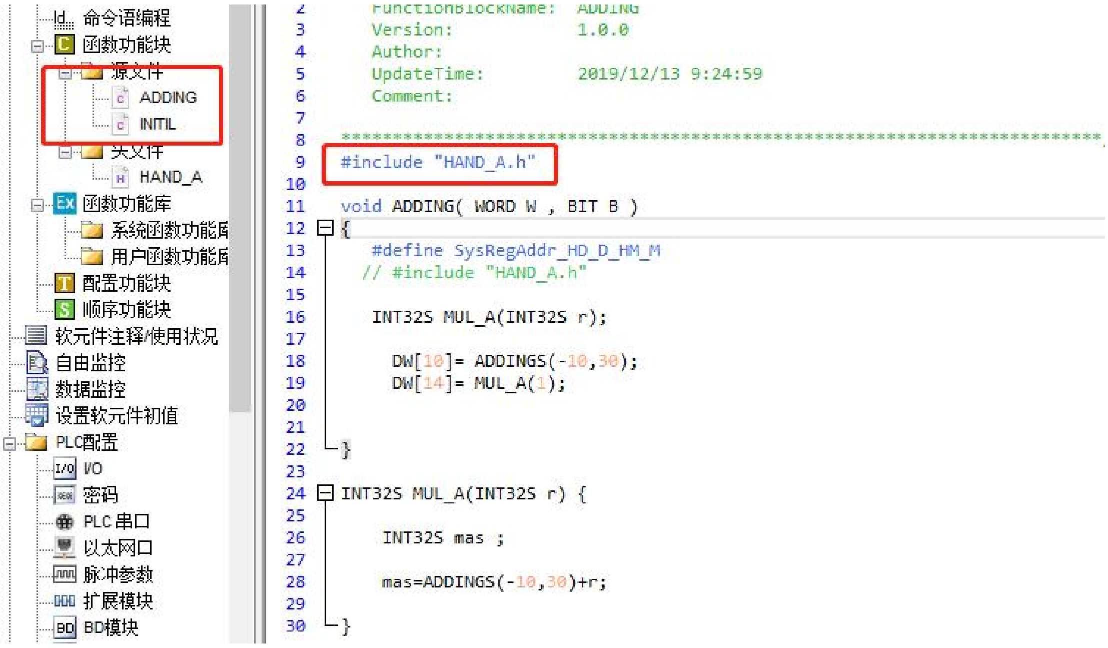
(5)The C language function in the header file has no compilation function.

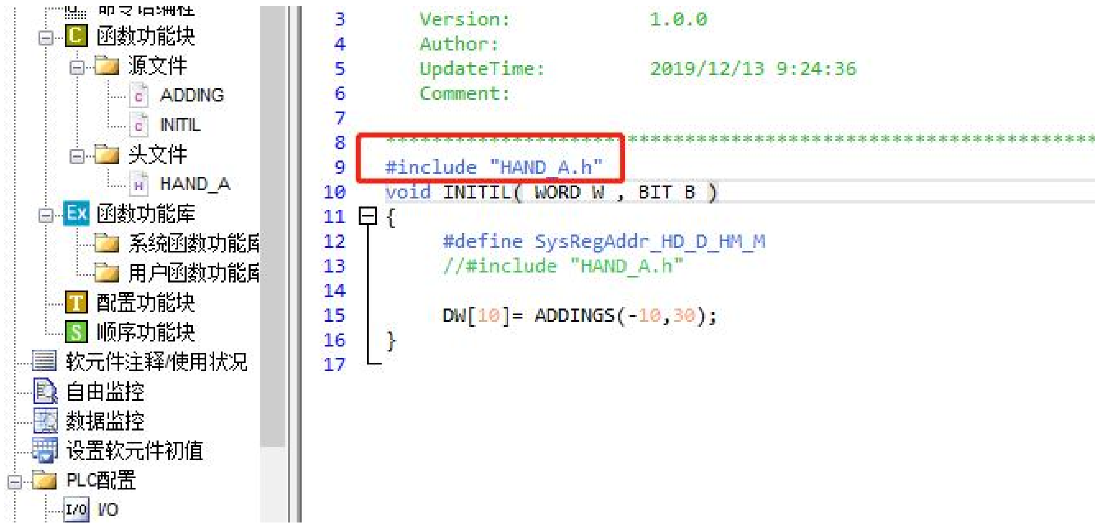


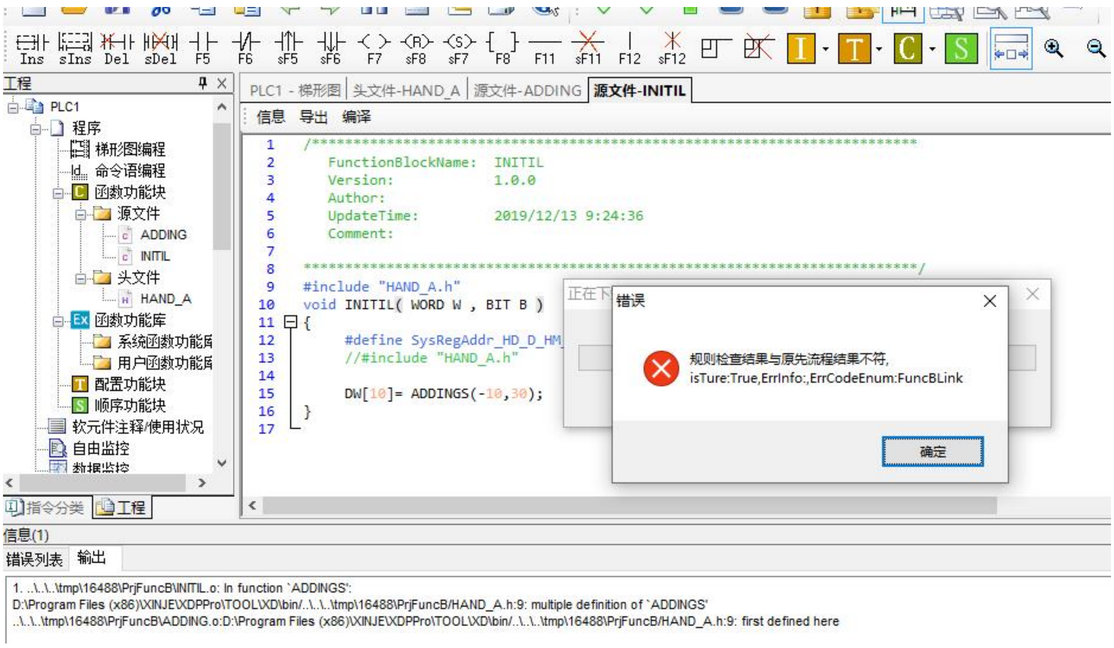
There is no compilation function in the header file. Only the source file can be compiled. The header file cannot be compiled separately.

(6)When two source files call the header file, you only need to write a declaration in one source file. Write in both source files and compile correctly, but the download program is wrong.

Using #include "xxx.h" outside the function can be understood as including this header file globally. There is no problem compiling a source file separately.
The function of the header file can be understood as: the compiler replaces #include "xxx.h" with variables and functions declared in the header file during code preprocessing.
However, during the download process, multiple source files are compiled and linked. After preprocessing, both source files have declarations of variables and functions in the header file. Repeated declaration errors will occur during linking, and XDPpro is shown as a link error.
Suggestion:
Correctly include the header file where the header file content needs to be used, rather than blindly include the header file directly outside the function.







8-11 Function Table

The default function library

Constant	Data	Description
_LOG2	(double)0.693147180559945309417232121458	Logarithm of 2
_LOG10	(double)2.302585092994045901093613792909 3	Logarithm of 10
_SQRT2	(double)1.41421356237309504880168872421	Radical of 2
_PI	(double)3.141592653589793238462643383279 5	PI
_PIP2	(double)1.570796326794896619231321691639 75	PI/2
_PIP2x3	(double)4.712388980384689857693965074919 25	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 **s1** and return **s1**
void * memset(void *s, int c, size_t n);	Replace the **n** words start from **s** position with word **c**, and return to 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);	float 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 integer 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 largest double integer 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 multiply the (two to the power of n) is x*2^n.
double log(double x);	float logf(float x);	Nature logarithm logic
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))

The using method of the functions in the table:

float asinf(float x);

float asinf: float means the return value is float format;

float x: float means the function formal parameter is float format. In actual using, it do not need to write the float. See line 14 in the following example:

```
void ZHENGXIAN( WORD W , BIT B )
{
int a;
float x,y,z;
x=FW[0];
y=asinf(x);
z=180*y/3.14159;
a=(int)z;
W[2]=a;
}
```

Flash register operation special function library

Flashregister operation special function	Explanation
flash_copy (void *dst, void *src, size_t len);	A function that copies data to a flash register. DST: the starting address of the target register copied to; SRC: source data address; Len: number of bytes copied;
flash_move (void *dst, void *src, size_t len);	the copy bytes of the flash register,if the target area and the source area overlap, flash_ Move can ensure that the bytes of the overlapping area are copied to the target area before the source string is overwritten, but the source content will be changed after copying. However, when the target area does not overlap with the source area, it is same to the function of flash_copy. DST: the starting address of the target register copied to; SRC: source data address; Len: number of bytes copied;
flash_set_int8 (void* dst, int8 data);	Make some type of assignment to the flash register. DST: the starting address of the target register; Data: different types of data;
flash_set_int16 (void* dst, int16 data);	
flash_set_int32 (void* dst, int32 data);	
flash_set_int64 (void* dst, int64 data);	
flash_set_float32(void* dst, float32 data);	
flash_set_float64(void* dst, float64 data);	

Take the copy data and assignment of flash register as an example to illustrate the use of functions in the function table:

Example 1: Copy data to Flash register FD100

flash_copy (void *dst, void *src, size_t len);

The Void in the flash_copy function represents the parameter type. In actual use, there is no need to write void. See line 13 in the following example:

```
void FUNC1( WORD W , BIT B )
{
#define SysRegAddr_HD_D_HM_M_FD_SFD
char a[8] = {'a', 'b', 'c', 'd', 'e', 'f', 'g', 'h'};
flash_copy ( &FD[100], &a, sizeof(a) );//使用sizeof(a)计算a的长度;

}
```

Example 2: set value in Flash register

flash_set_int16 (void* dst, int16 data);

The advantage offlash_set_int16compared to flash_copy:

If using flash_copyto set value in flashregister. It is very inconvenient to use.

int temp_val = 1000;

flash_copy(&FD[1000], &temp_val, sizeof(temp_val));

If using flash_set:flash_set_int32(&FD[1000], 1000);

See line 13~18 in the below example:

```
9   void FUNC1( WORD W , BIT B )
10  {
11  #define SysRegAddr_HD_D_HM_M_FD_SFD
12    //flash_set系列函数的使用示例
13      flash_set_int8   ( &FD[104], 8 );
14      flash_set_int16  ( &FD[106], 16 );
15      flash_set_int32  ( &FD[108], 32 );
16      flash_set_int64  ( &FD[112], 64 );
17      flash_set_float32 ( &FD[120], 32.32 );
18      flash_set_float64 ( &FD[122], 64.64 );
19
20  }
21
```

Note:

(1) flash_ move function requires the support of the PLC firmware version of the lower computer (firmware version: v3.7.2 firmware date: 20210528).

(2) The flash register can be written about 1000000 times, and each write is the erasure of the whole flash register, which is time-consuming. Frequent writing will cause permanent damage to the flash register. Therefore, it is not recommended that users write frequently. Carefully use the power on normally on and oscillation coil (e.g. SM0, SM11) as the driving conditions.

9 Sequence BLOCK

This chapter mainly introduces sequence block instruction and the application.

Sequence Block instruction:

Mnemonic	Function	Ladder chart	Chapter
Sequence Block			
SBSTOP	Pause BLOCK	SBSTOP S1 S2	9-6-1
SBGOON	Go to execute BLOCK	SBGOON S1 S2	9-6-1

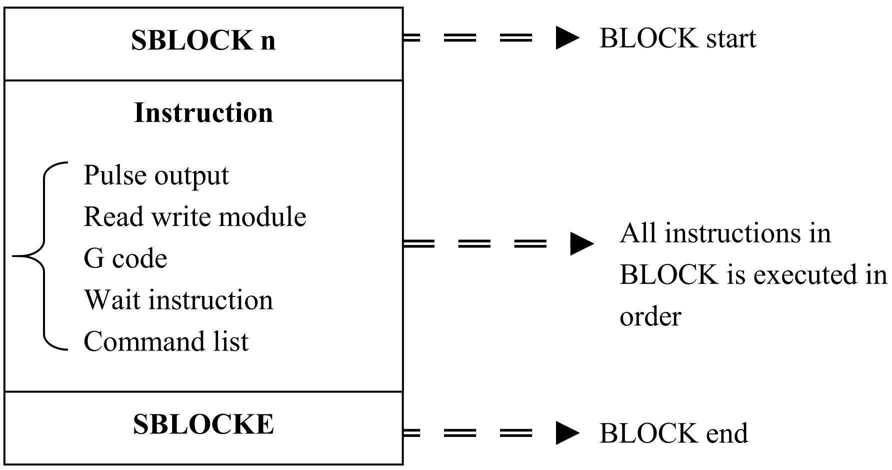
9-1 Concept of the BLOCK

Sequence block whose brief name is BLOCK is a program block to realize some functions. As a special flow, all instructions in the block are executed in order, which is the biggest difference with general processes.

BLOCK starts from SBLOCK and ends with SBLOCKE, and programmers can write instructions in the BLOCK. If one BLOCK contains multiple pulse output instructions(or other instructions), then pulse output instructions will execute in accordance with conditions meet order; And meanwhile the next pulse output instruction will not execute until the current instruction is over.

The XD3, XDM series PLC supports multiple BLOCKs※1.

A complete BLOCK structure is shown as below:



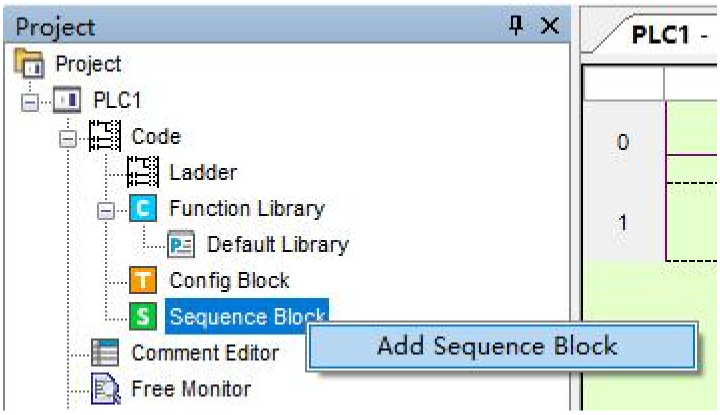
※1: Firmware version below V3.4.5: the XD series PLC allows up to eight BLOCKs. Firmware version V3.4.5 and above: XD/XL series PLC can write up to 100 BLOCKs, but at the same time can only run 8.
※2: When the trigger condition of the BLOCK is triggered by the closure of the normally open coil, it will be executed from the top of the BLOCK to the bottom in turn. When the last instruction is executed, the execution of the BLOCK will be restarted immediately from the top to the bottom. When the trigger condition is disconnected, the BLOCK will not stop immediately, but will complete the last scan and stop after the execution of the unexecuted program.
※3:When the triggering condition of BLOCK is triggered by the rising edge of the coil, the sequential function BLOCK will be executed one time from top to bottom and will not be executed circularly.

9-2 Call the BLOCK

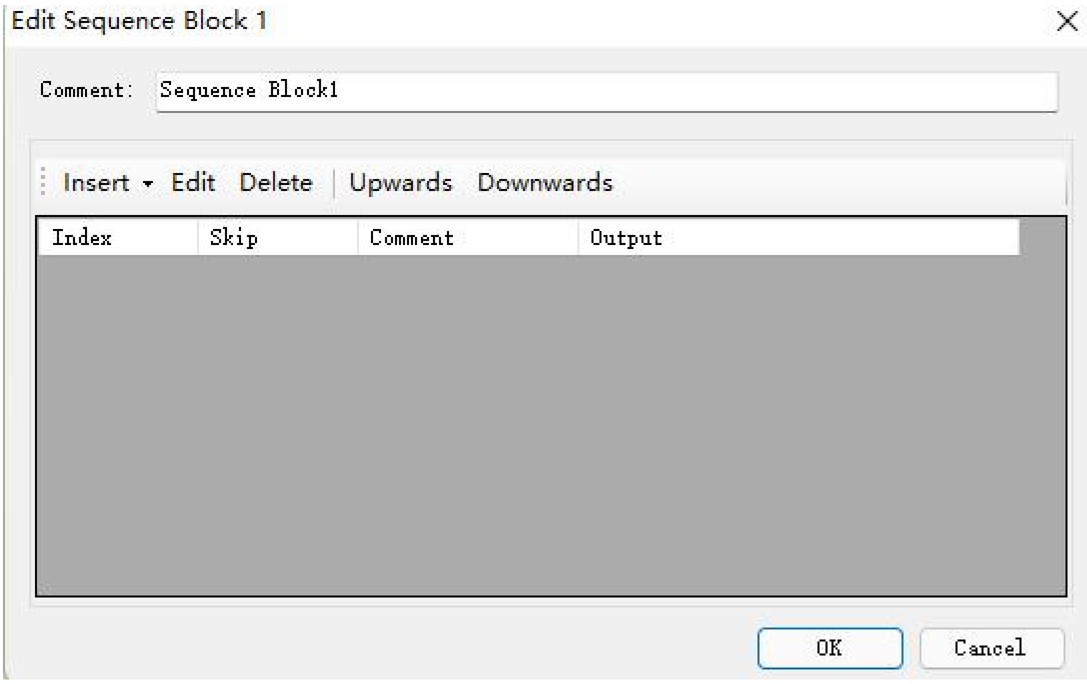
In one program file, it can call many BLOCK; the following is the method to add BLOCK in the program.

9-2-1 Add the BLOCK

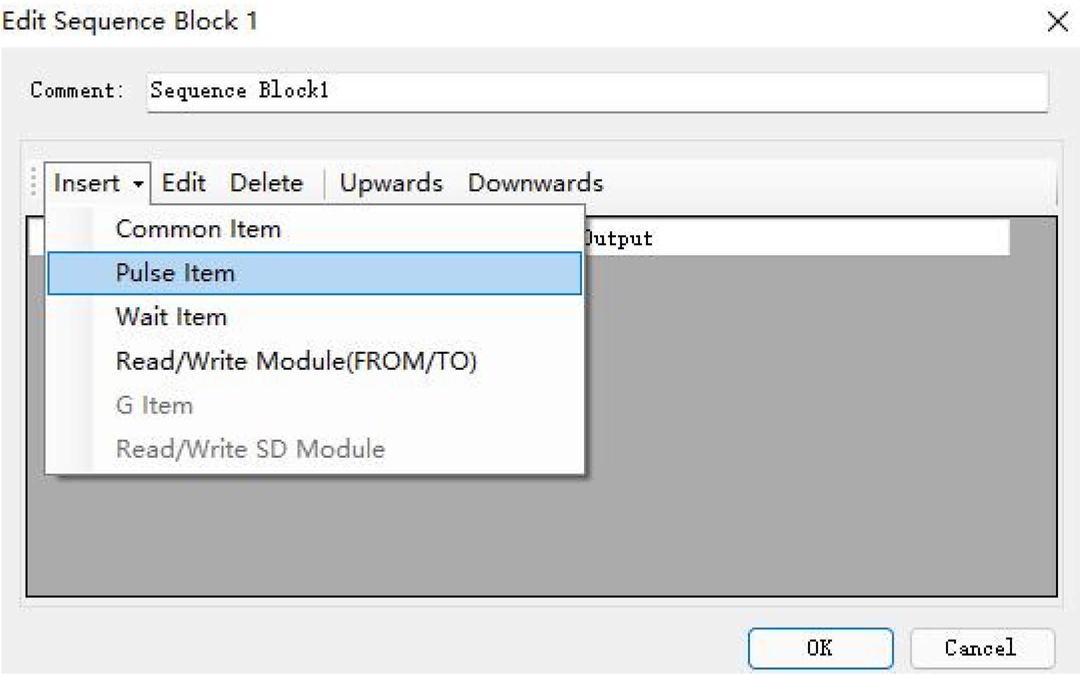
Open XDPPro software, right click the sequence block in the project bar:



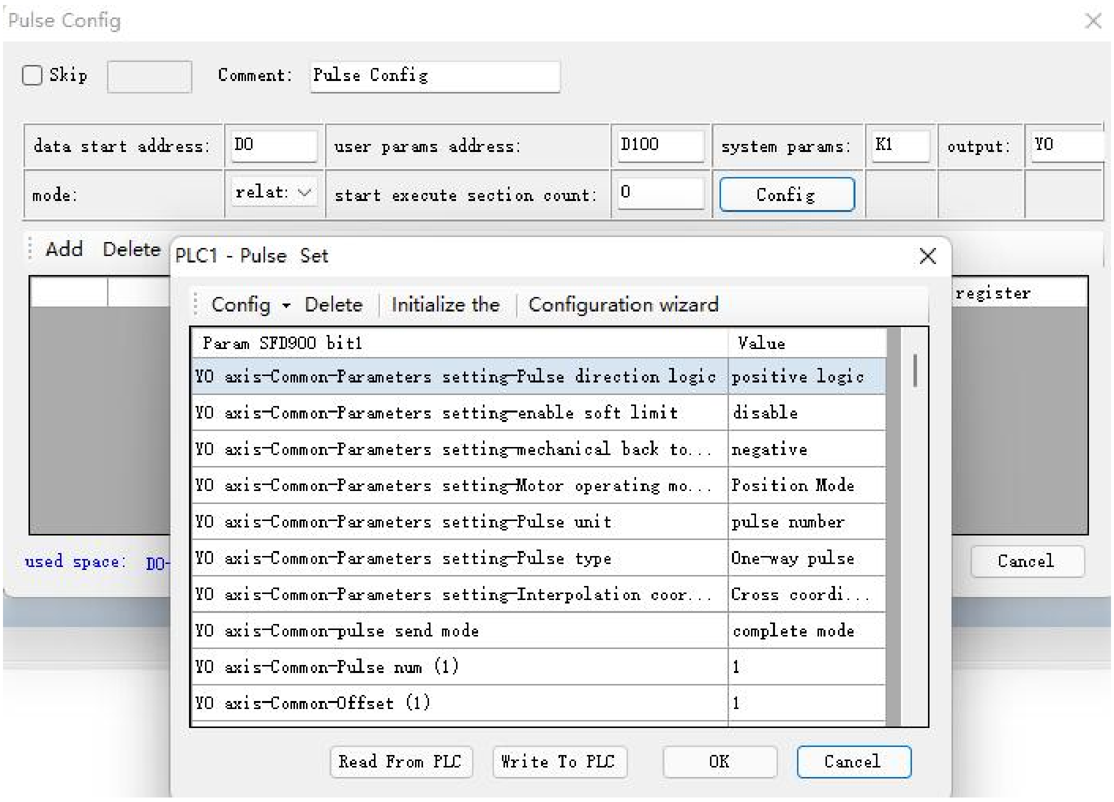
Click the command 'add sequence block', the following window will jump out:



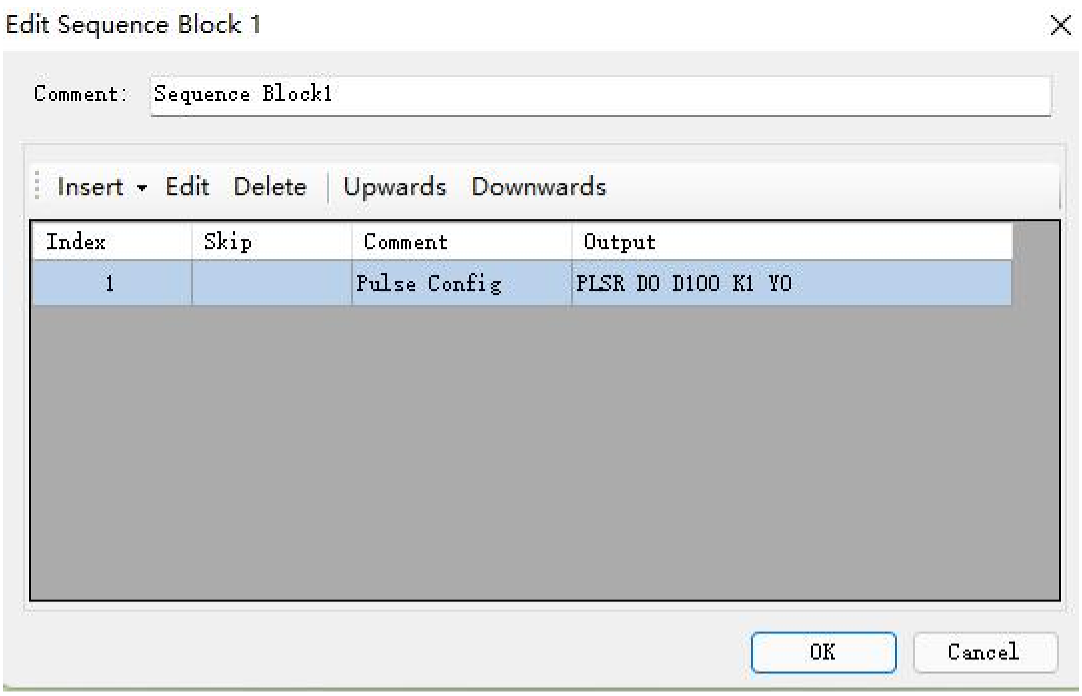
You can edit the BLOCK in the window, Upwards/Downwards are used to change the position of instructions in the block.
Click 'insert' button, some instructions list under the menu:



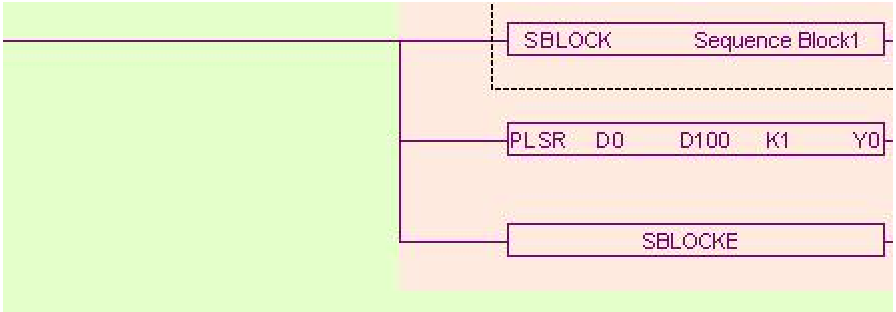
Take ‘Pulse Item’ for example:



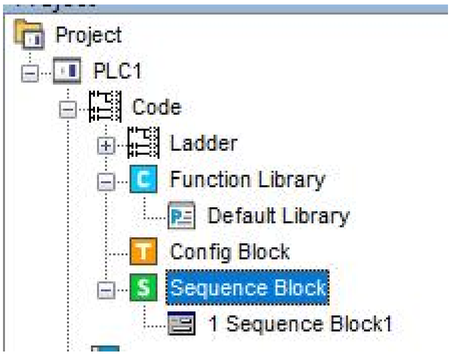
After click ‘OK’, you will find information in the configuration:



Click ‘OK’, the following instructions are added in the ladder:

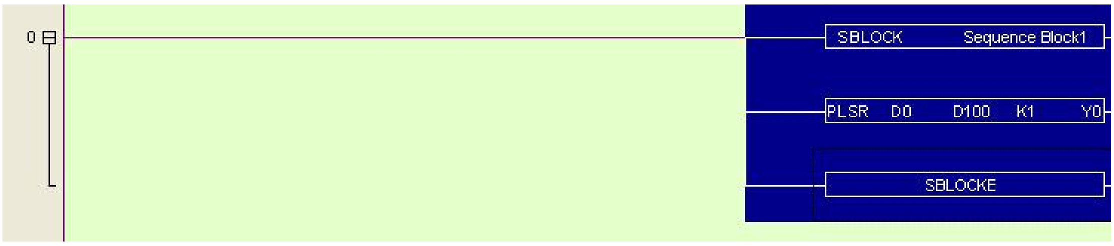


Meantime, a new sequence block is added in the left of the project bar:

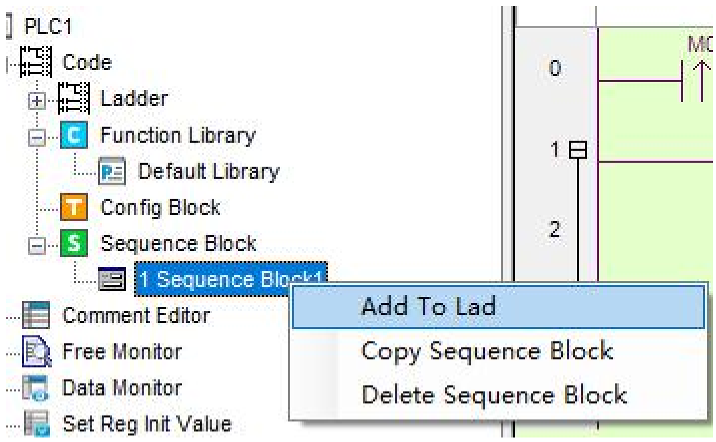


9-2-2 Move the BLOCK

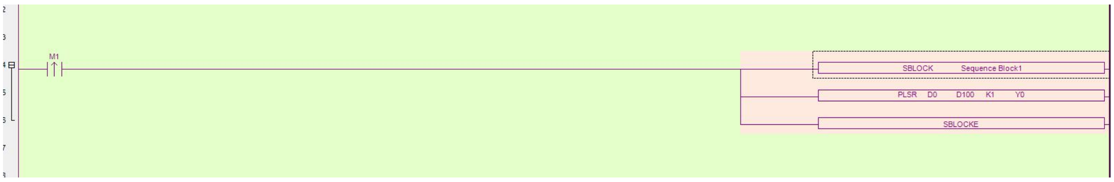
If you want to move the BLOCK to other place, you have to select the original BLOCK and delete it (select all, then delete):



Move the cursor to the new place, and then right click the BLOCK and select 'add to lad':

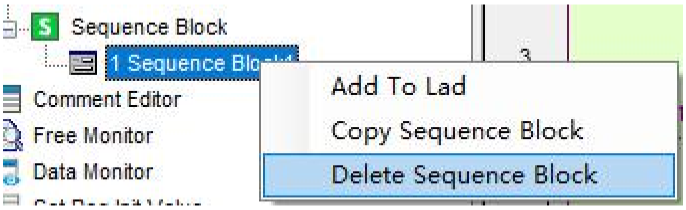


Now the BLOCK is moved to the new place:



9-2-3 Delete the BLOCK

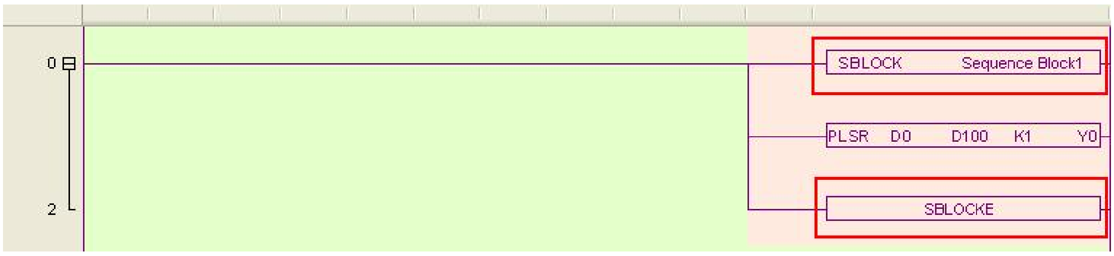
You can select the called BLOCK and delete it. If you want to completely delete the BLOCK, right click the function block and select ‘delete sequence block’. After this operation, you can’t call this BLOCK any more:

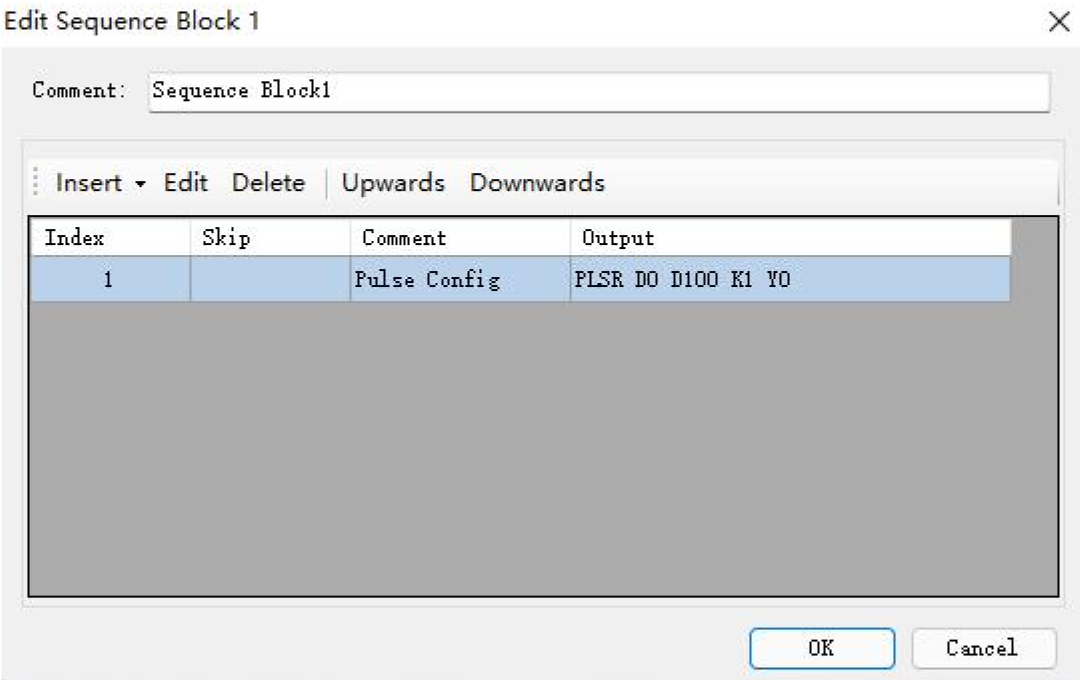


9-2-4 Modify the BLOCK

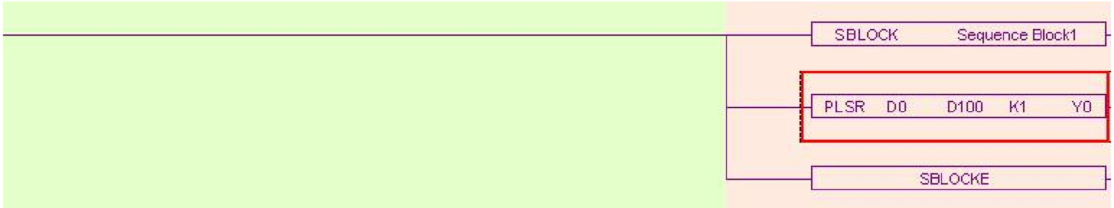
There are two methods to modify the BLOCK.

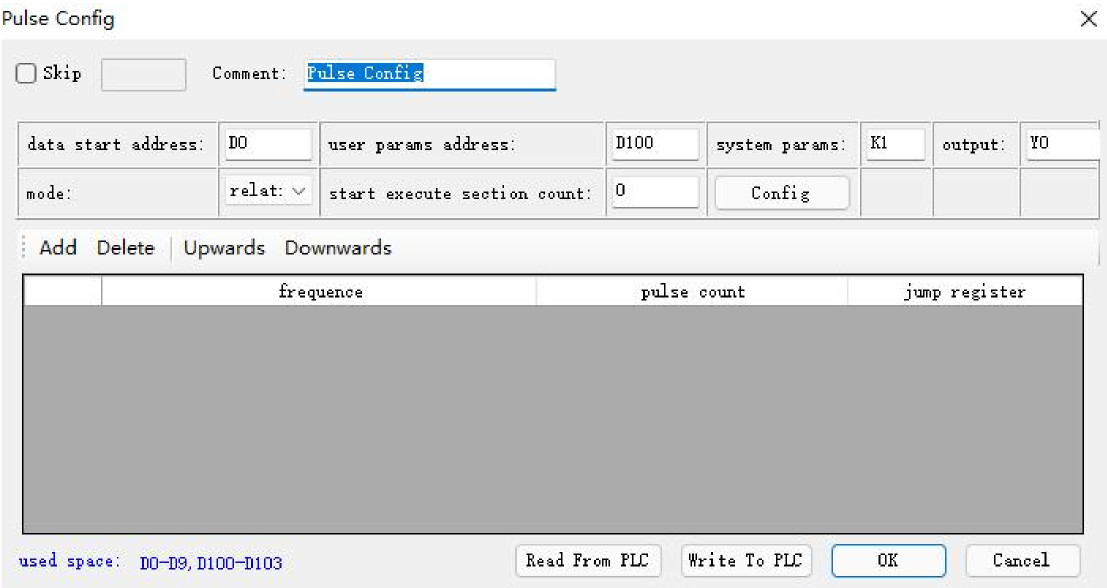
(A) Double click the start/end segment to modify the BLOCK in general:





(B) Double click the middle part to modify :

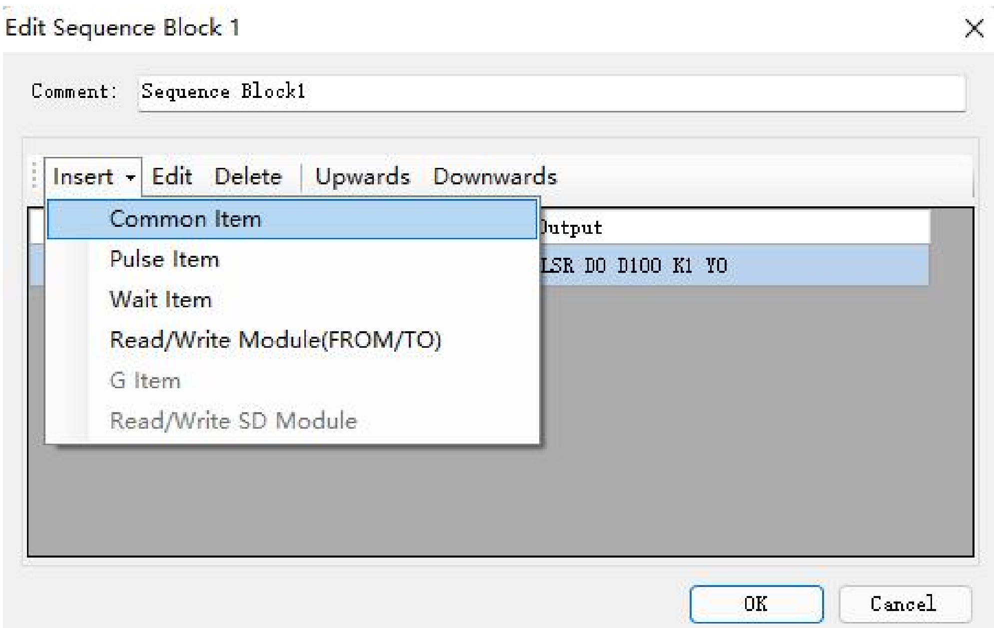




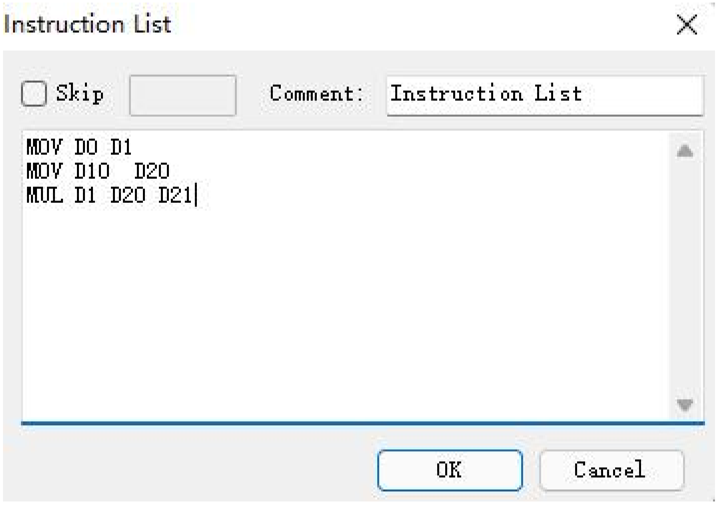
9-3 Edit the instruction of the BLOCK

9-3-1 Command item

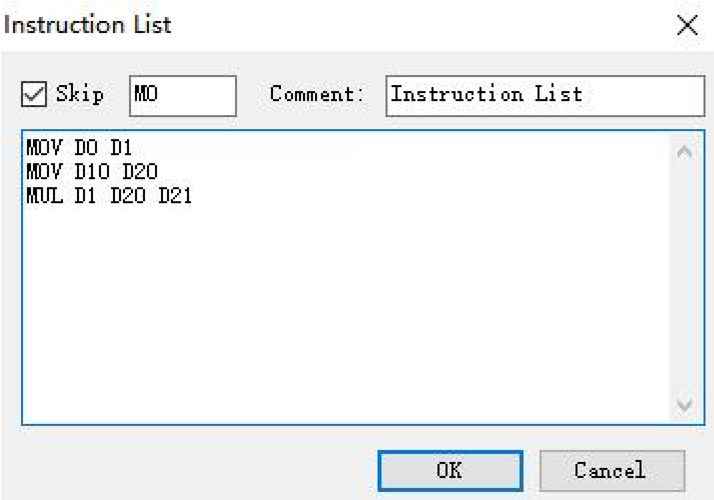
Use ‘command item’ to edit the program:



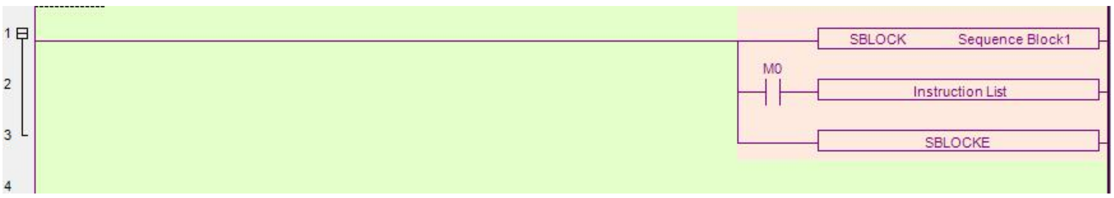
An ‘instruction list’ will jump out after click the ‘command item’:



Users can add instructions in the frame.
Skip: to control the stop and run of the instructions. If you select skip and input control coil in the frame, then when the control coil is ON, the command will not be executed. If not select, the default action is execution.
Comment: to modify the note for the instruction.

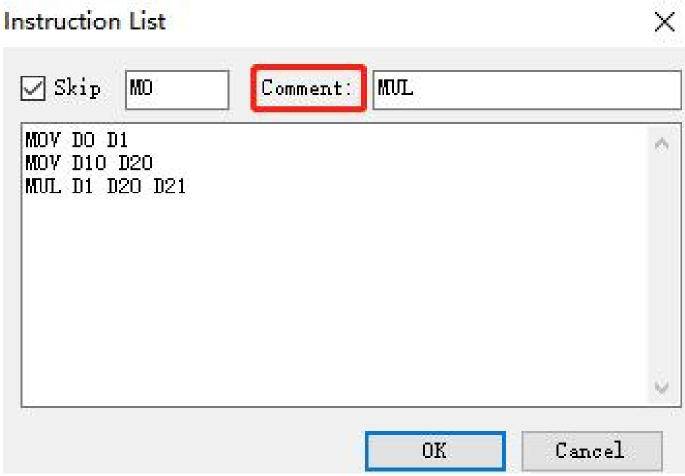


Click ‘OK’, the ladder program will change as the following:

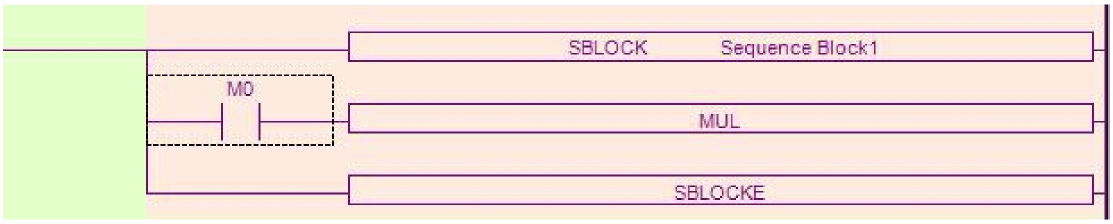


Note: We can add multiply instructions in one BLOCK and use ‘Skip’ as every instruction’s execution condition.

In the above figure, the command segment is not expanded in the ladder diagram, but its annotation can be modified according to the function of the segment, as shown in the following figure:

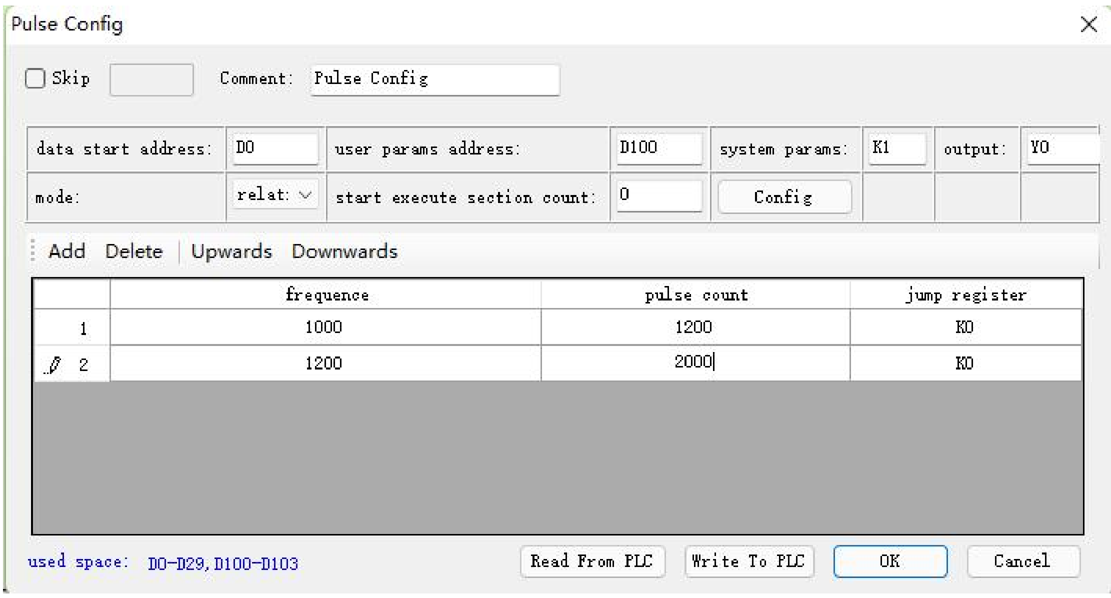


The modified block phrase has also changed accordingly, as shown in the following figure:

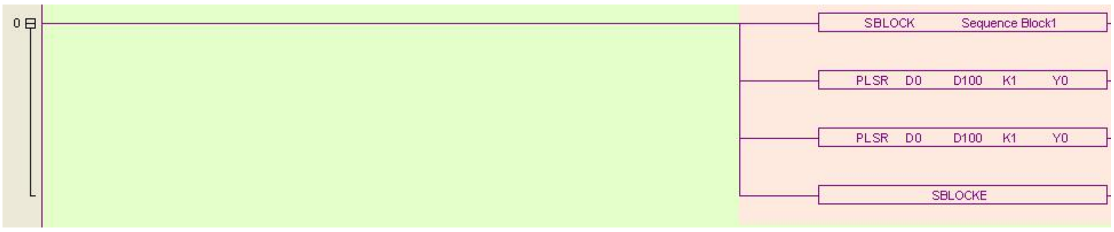


9-3-2 Pulse Item

Open the 'pulse item' in the same way:



In the following BLOCK, we add two impulse instructions:

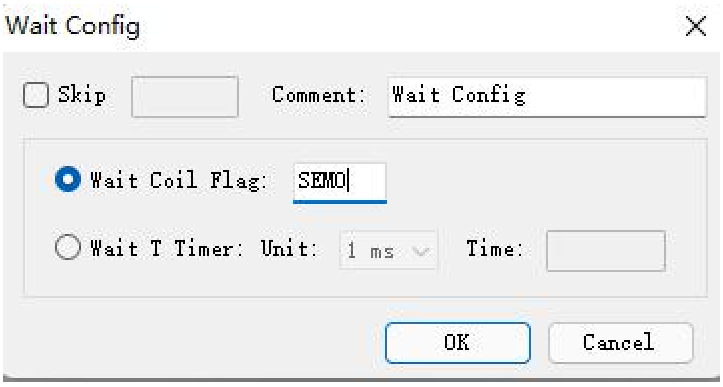


9-3-3 Wait Item

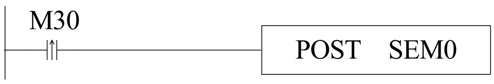
'Wait Item': to wait coil flag or timer bit.

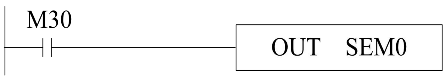
Open 'Wait Item' in the same way. There are two waiting modes: flag bit and timer wait.

(A) Flag bit

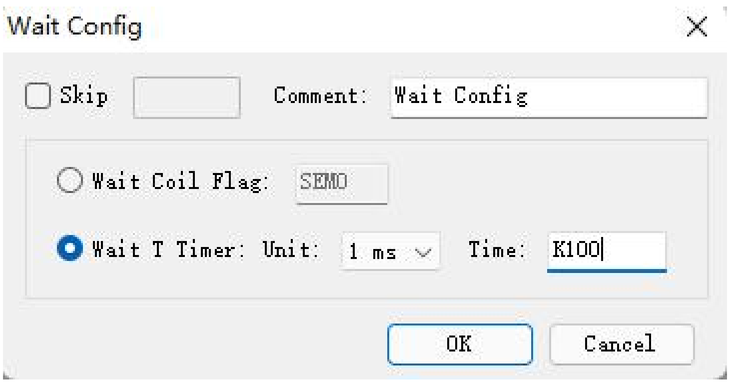


SEM corresponding ladder diagram is as below:

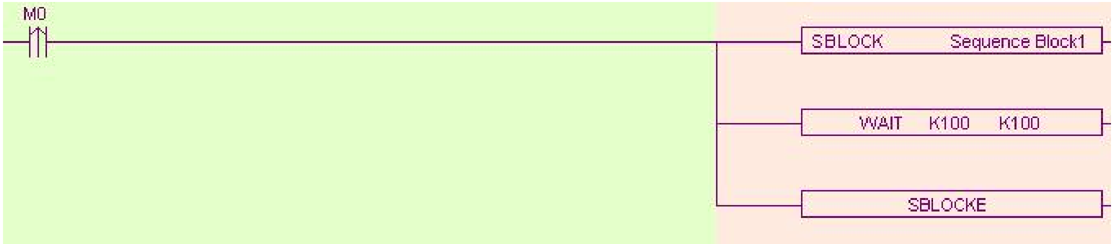




(B) Timer wait



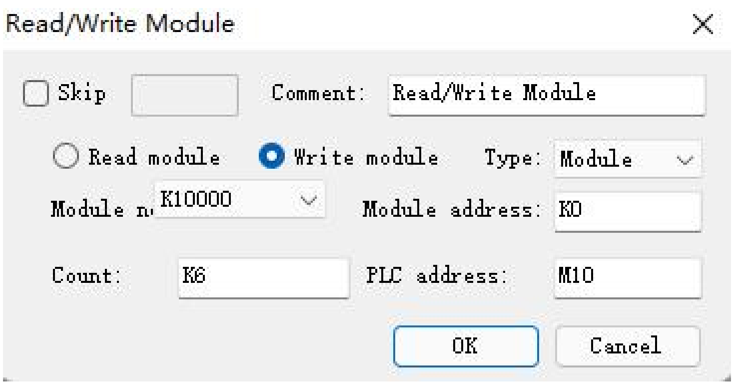
(C) Corresponding ladder diagram:



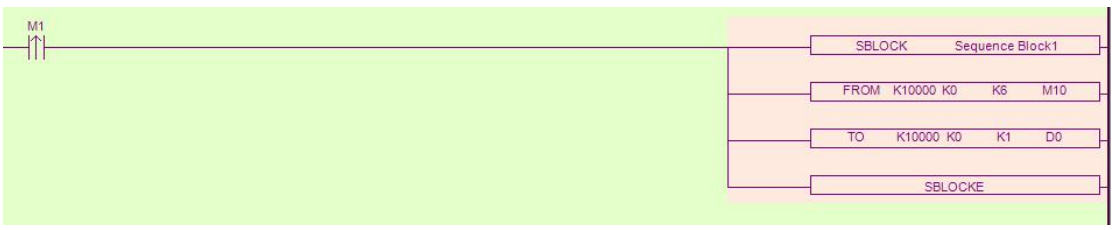
Note: Do not add normal coil after WAIT instruction in XD/XL series PLC sequence BLOCK, and add XD, XL series PLC special signal SEM bit(SEM0~SEM31); SEM cannot be controlled by set or reset. It can only be set by POST instruction and reset by WAIT SEM instruction. Or output via OUT instruction. The difference between them is that the POST command needs to be triggered by the pulse edge to keep the state of SEM; the OUT command needs to be triggered by the normally open coil, and the SEM is reset when the triggering condition is disconnected.

9-3-4 Module Read and Write(FROM/TO)instruction

This item is used to read and write data between PLC and modules, and the operate panel is as below:
1#read



FROM\TO instruction can be selected from pull-down list:

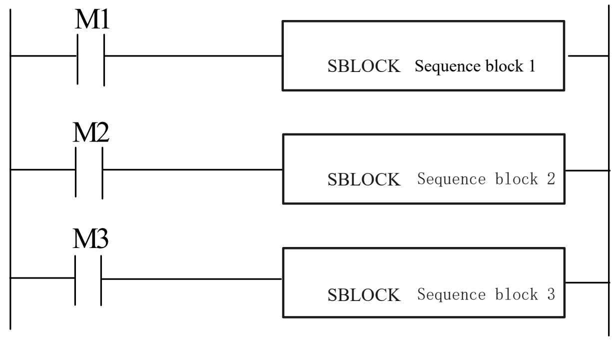


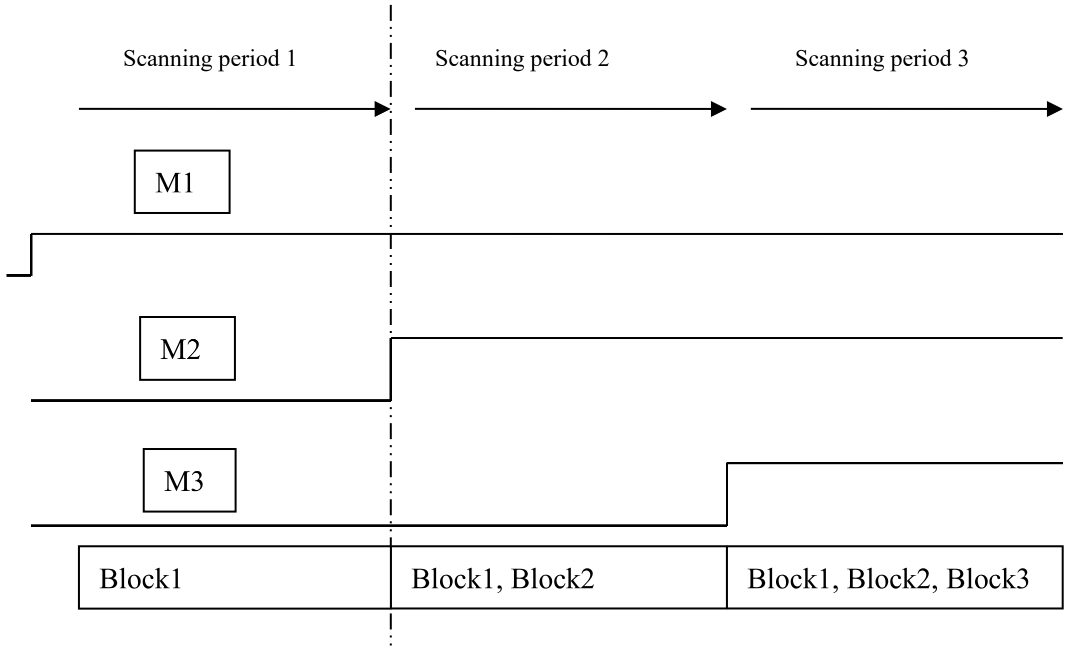
Note: As shown in the figure above, in V3.4 and above version software, when the module number is set to K0~K15, the corresponding ladder diagram will be displayed as K10000~K10015.

9-4 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.

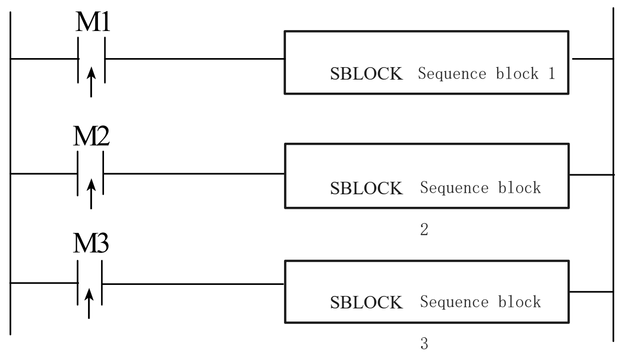
(A) The condition is normal ON, normal OFF coil





Note: When the program in the BLOCK is not executed and the triggering condition M is disconnected, the BLOCK will not stop immediately, but will complete the last scan, and will stop after the rest of the program has been executed.

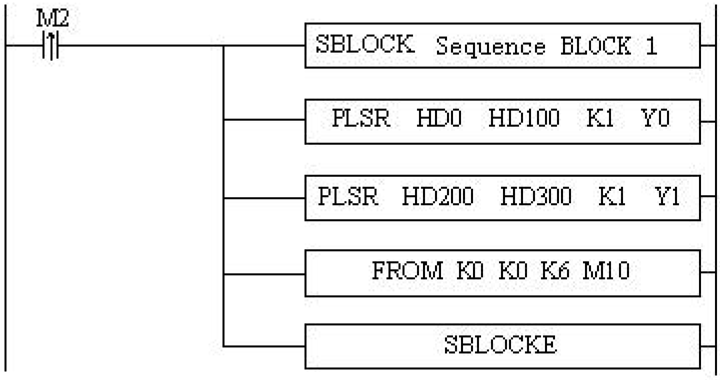
(B) The condition is rising or falling edge of pulse



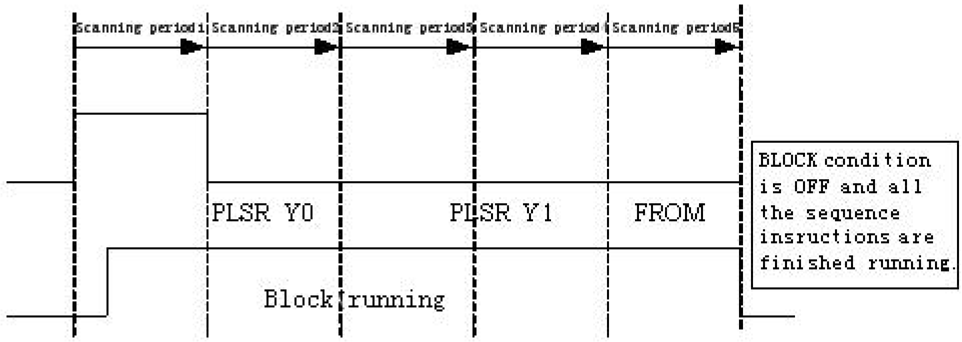
When M1, M2, M3 is from OFF to ON, all these blocks will run once.

2. The instructions in the block run in sequence according to the scanning time. They run one after another when the condition is ON.

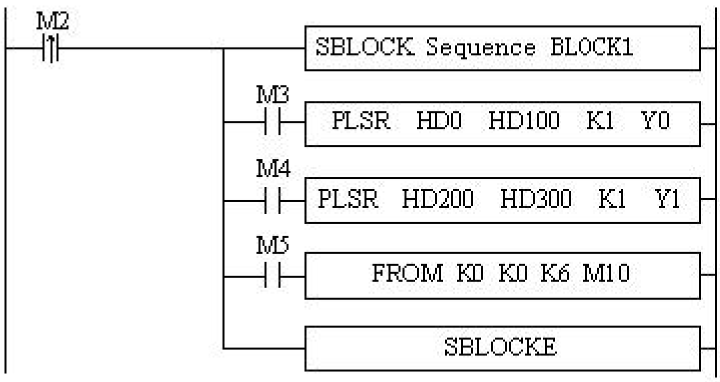
(A) Without SKIP condition



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.

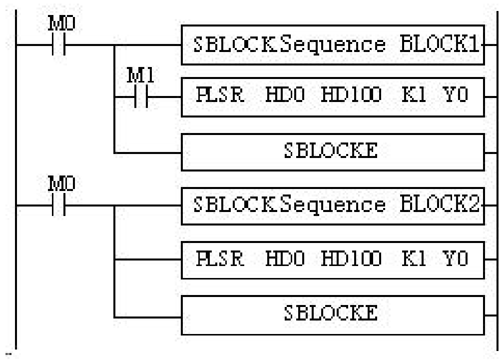
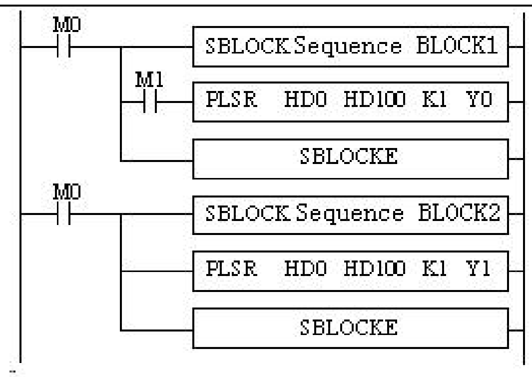
D) When M3 is OFF, if no other instructions use this Y0 pulse, PLSR HD0 HD100 K1 Y0 will run; if not, the PLSR HD0 HD100 K1 Y0 will run after it is released by other instructions.

E) After Y0 pulse sending completed, check M4. If M4 is OFF, check Y1 block, if M4 is ON, check M5. If M5 is OFF, module commmunication will run.

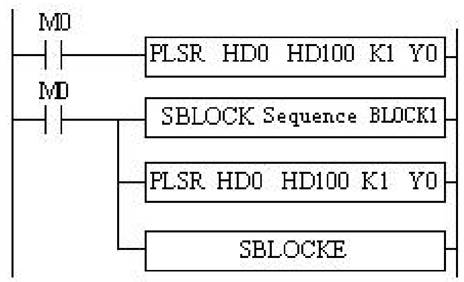
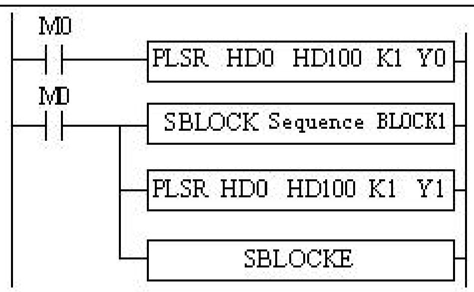
9-5 BLOCK instruction editing rules

In the BLOCK, the instruction editing should accord with some standards.

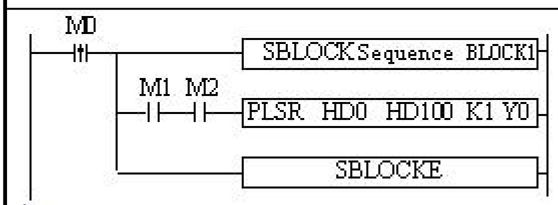
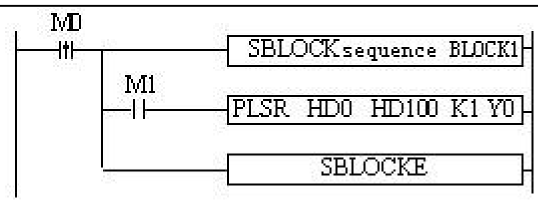
(1)Do not use the same pulse output terminal in different BLOCK.

NO(×)	**YES(√)**
	

(2)Do not use the same pulse output terminal in BLOCK and main program.

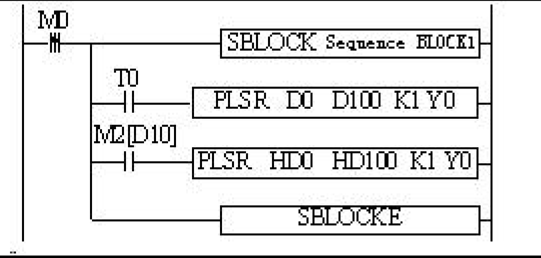
NO(×)	**YES(√)**
	

(3)There only can be one SKIP condition for one BLOCK instruction.

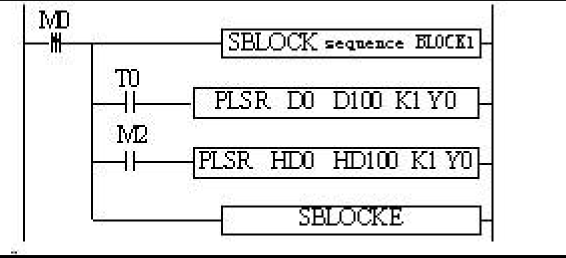
NO(×)	**YES(√)**
	

(4)The SKIP condition only can use M, X, can not use other coil or register.

NO(×)

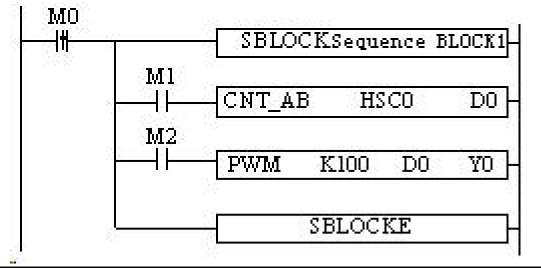


YES(√)

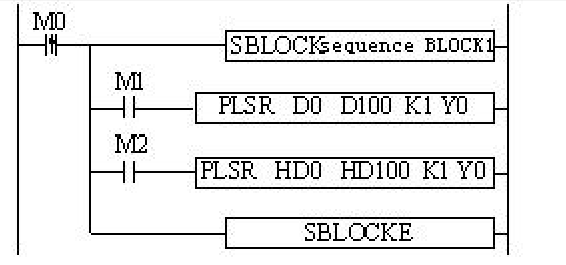


(5)The output instructions cannot be CNT_AB(CNT), PWM.

NO(×)

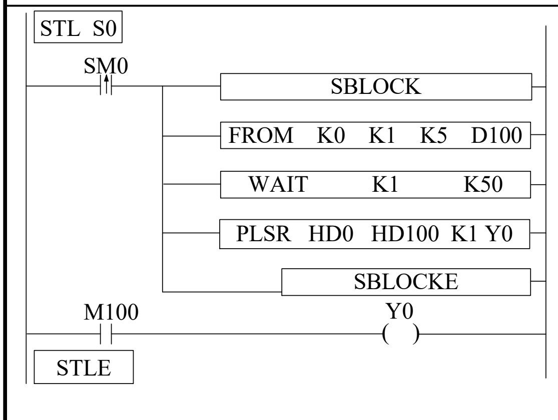


YES(√)

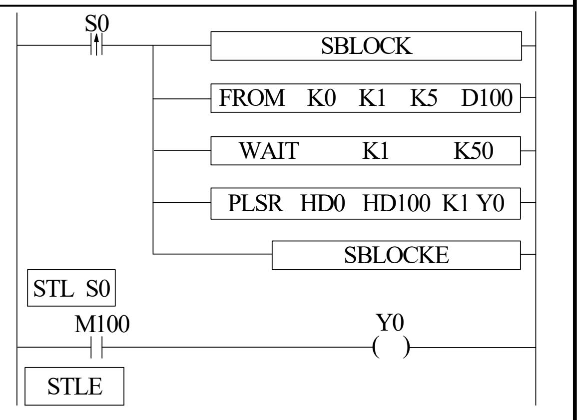


(6)BLOCK is not recommended to put in the STL, because if one STL ends, while the BLOCK doesn't end, then big problem will happen.

NO(×)



YES(√)



(7)Label Kind type cannot be used in the block

Sign P, I cannot be used in block. Even they can be added in block, but they do not work in fact.

9-6 BLOCK related instructions

9-6-1 Instruction explanation

Stop running the BLOCK [SBSTOP]

1)Summary

Stop the instructions running in the block

[SBSTOP]			
16 bits	SBSTOP	32 bits	-
Condition	NO,NC coil and pulse edge	Suitable types	XD, XL
Hardware		Software	V3.2

2)Operands

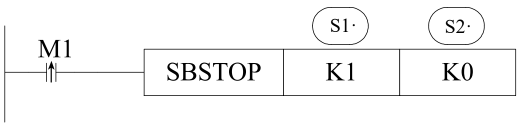
Operand	Function	Type
S1	The number of the BLOCK	16bits, BIN
S2	The mode to stop the BLOCK	16bits, BIN

3)Suitable component

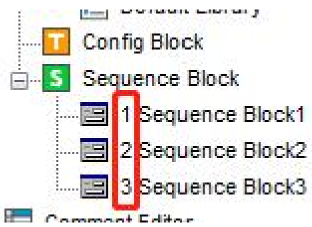
Operands	Word soft elements											Bit soft elements						
	System								Constant	Module		System						
	D	FD	TD	CD	DX	DY	DM	DS	K/H	ID	QD	X	Y	M	S	T	C	Dnm
S1	●								●									
S2									●									

*Notes: D includes D, HD; TD includes TD, HTD; CD includes CD, HCD, HSCD, HSD; DM includes DM, DHM; DS includes DS, DHS. M includes M,HM,SM； S includes S,HS； T includes T,HT； C includes C, HC.

Function



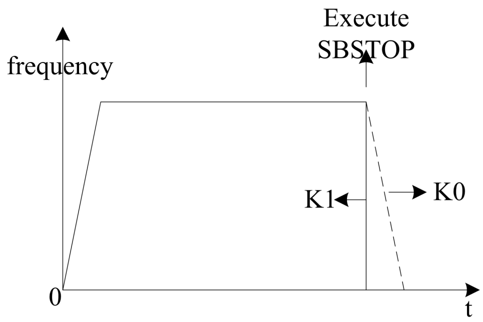
S1 is the block number of sequence block. The block number is unique and cannot be changed. It can be viewed in the left engineering bar as follows.



S2 is the mode for BLOCK stop, operand: K0, 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.



K2: Destructive slow stop BLOCK, that is, when the pulse is being sent, the SBSTOP condition holds, then the pulse will slow down along the slope, without to use with the SBGOON instruction, so the remaining instructions will not be executed. After executing this instruction, the BLOCK can be restarted. (Note: K2 mode is only supported by V3.4.2 and above PLC)
K3: Destructive immediate stop BLOCK, that is, when the SBSTOP condition is met while the pulse is being sent, the execution of the pulse instruction in BLOCK will be immediately stopped without the need to use it in conjunction with the SBGOON instruction, so the remaining instructions will no longer be executed. After executing this instruction, the BLOCK can be restarted. (Note: K3 mode is only supported by PLCs with firmware versions V3.4.6 and above, or XD5E/XDME with firmware versions V3.5.3 and above.)

Continue running the BLOCK[SBGOON]

1)Summary

This instruction is opposite to SBSTOP. To continue running the BLOCK.

[SBGOON]			
16 bits	SBGOON	32 bits	-
Condition	Pulse edge	Suitable types	XD, XL
Hardware	-	Software	V3.2

2)Operands

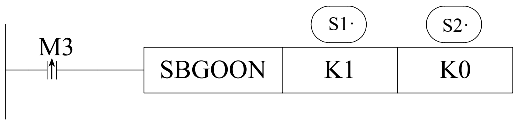
Operand	Function	Type
S1	The number of the BLOCK	16 bits, BIN
S2	The mode to continue running the BLOCK	16 bits, BIN

3)Suitable component

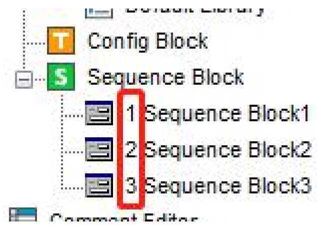
Operands	Word soft elements											Bit soft elements						
	System								Constant	Module		System						
	D	FD	TD	CD	DX	DY	DM	DS	K/H	ID	QD	X	Y	M	S	T	C	Dn.m
S1	●								●									
S2									●									

*Notes: D includes D, HD; TD includes TD, HTD; CD includes CD, HCD, HSCD, HSD; DM includes DM, DHM; DS includes DS, DHS. M includes M,HM,SM； S includes S,HS； T includes T,HT； C includes C, HC.

Function



S1 is the sequential function block number, which is unique and cannot be changed. It can be viewed in the left engineering column at the following location:

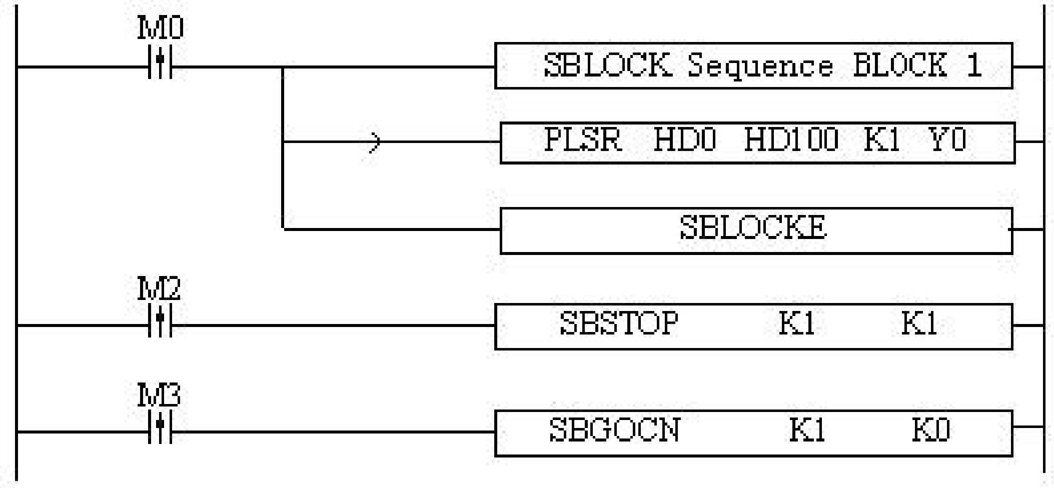


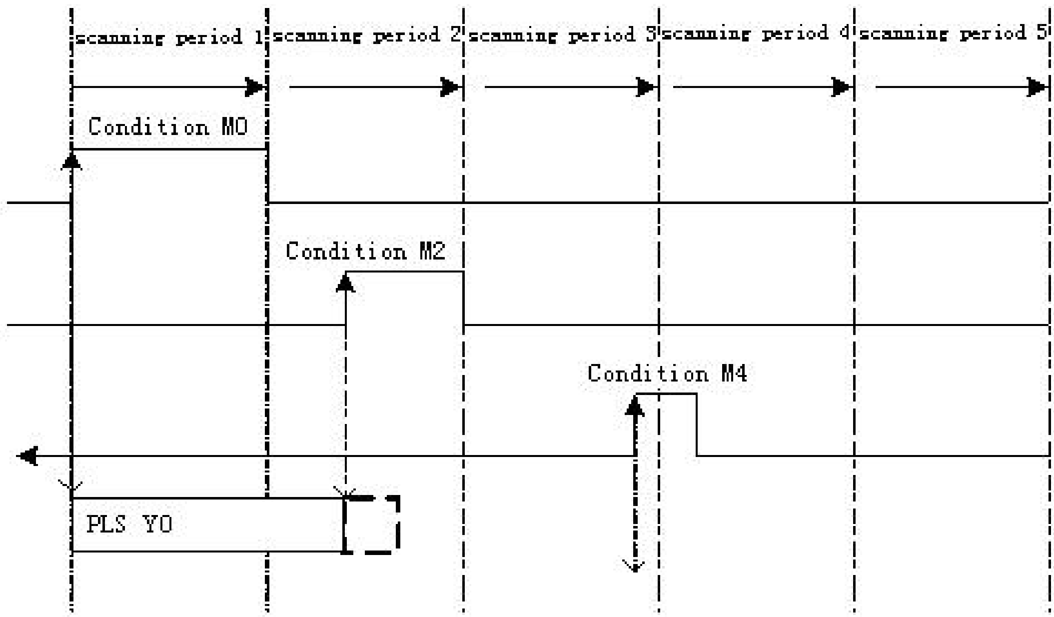
S2 is the mode to continue running the BLOCK. Operand: K0, K1.
K0: continue running the instructions in the BLOCK.
For example, if pulse outputting stopped last time, SBGOON will continue outputting the rest pulse;
K1: continue running the BLOCK, but abandon the instructions have not finished last time. Such as the pulse output instruction, if the pulse has not finished last time, SBGOON will not continue outputting this pulse but go to the next instruction in the BLOCK.

This instruction only applies to PLSR instructions in BLOCK, and can only send the remaining pulses for interpolation instructions, which can not be skipped.

9-6-2 The timing sequence of the instructions

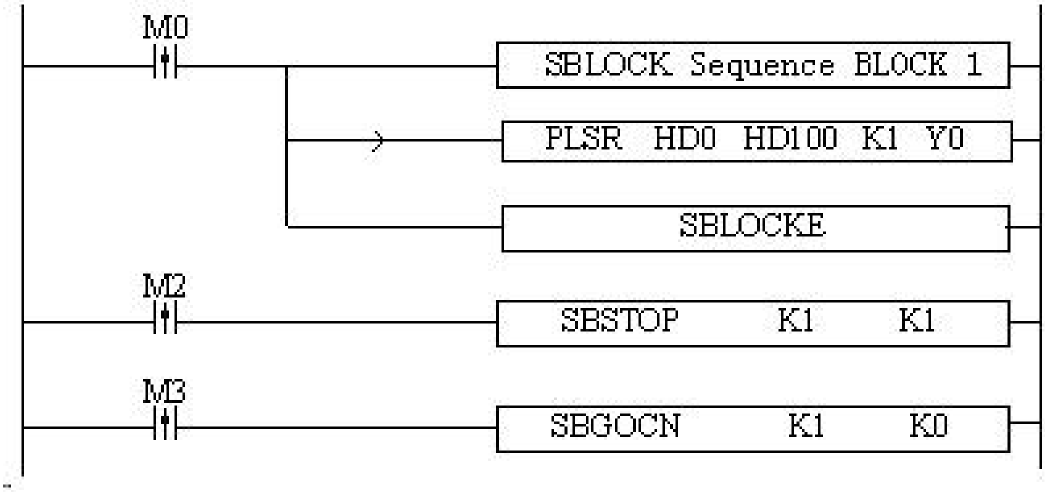
SBSTOP (K1 K1) +SBGOON (K1 K1)

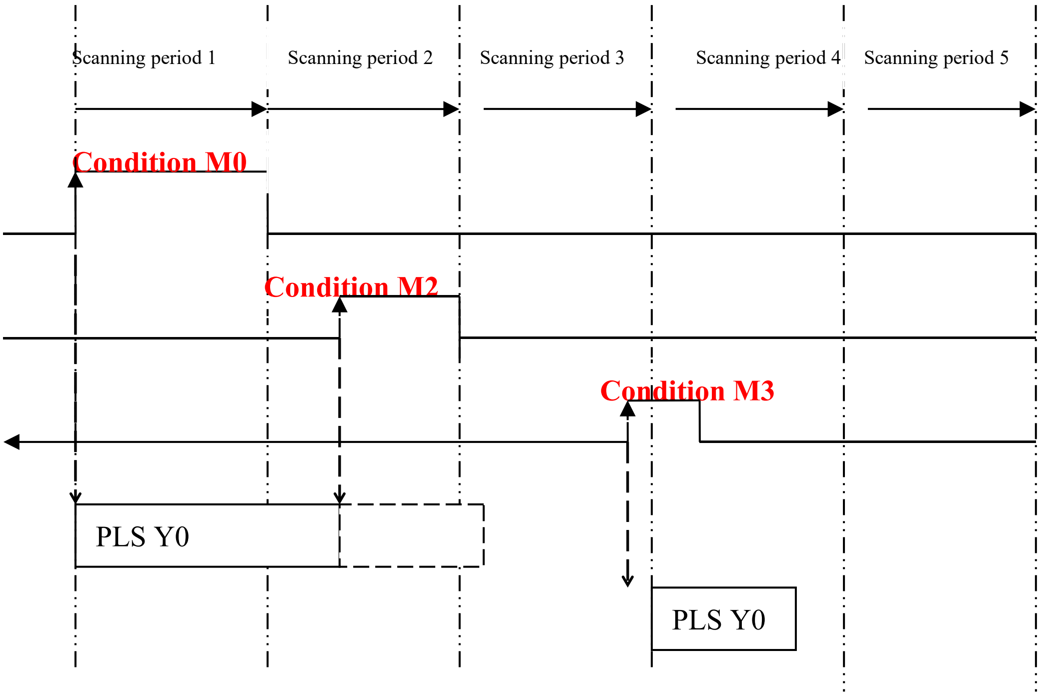




When M0 is from OFF→ON, run “PLSR HD0 HD100 K1 Y0” in the BLOCK to output the pulse;
When M2 is from OFF→ON, the BLOCK stops running at once;
When M4 is from OFF→ON, abandon the rest pulse.

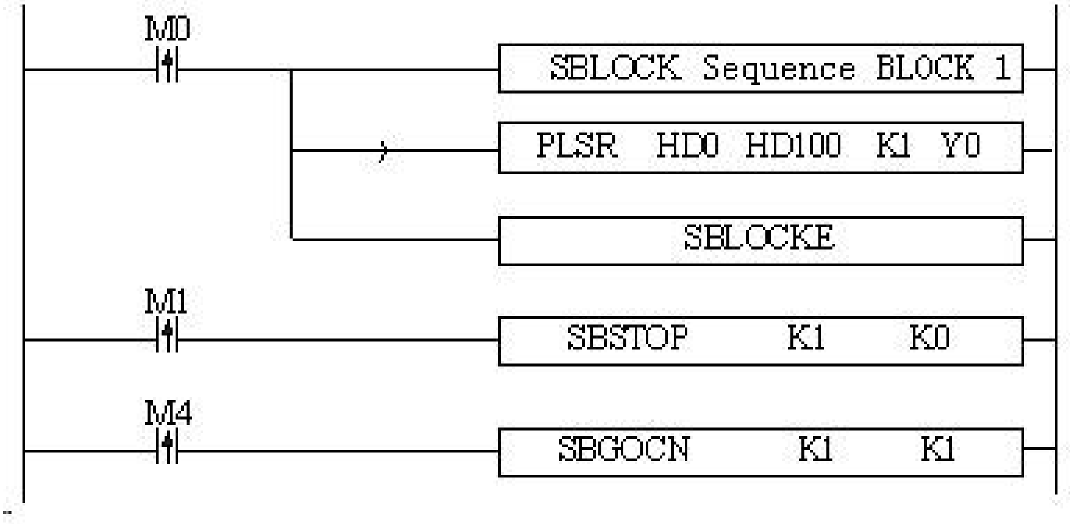
SBSTOP(K1 K1)+SBGOON(K1 K0)

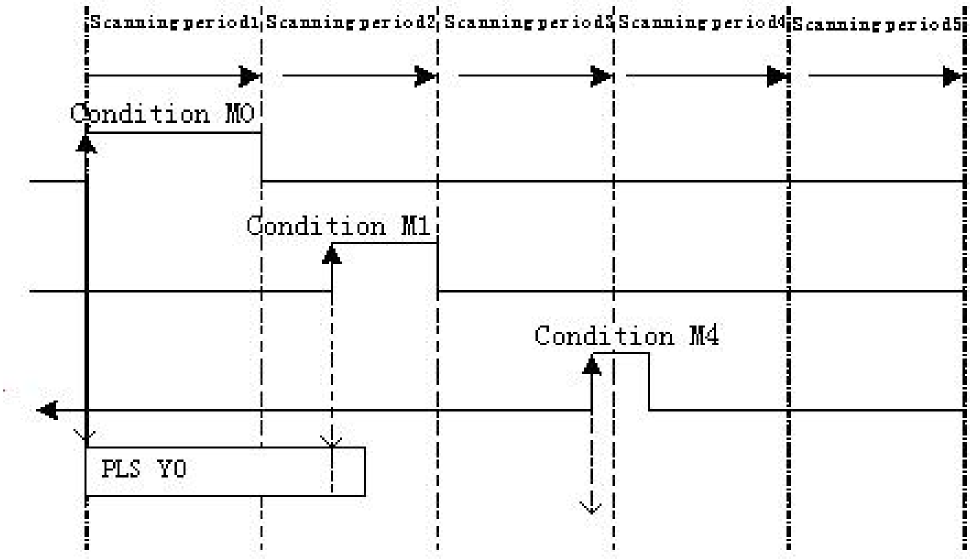




When M0 is OFF→ON, run ‘PLSR HD0 HD100 K1 Y0’ in the BLOCK to output the pulse;
When M2 is OFF→ON, the BLOCK stops running, the pulse output stops at once;
When M3 is OFF→ON, output the rest pulses.

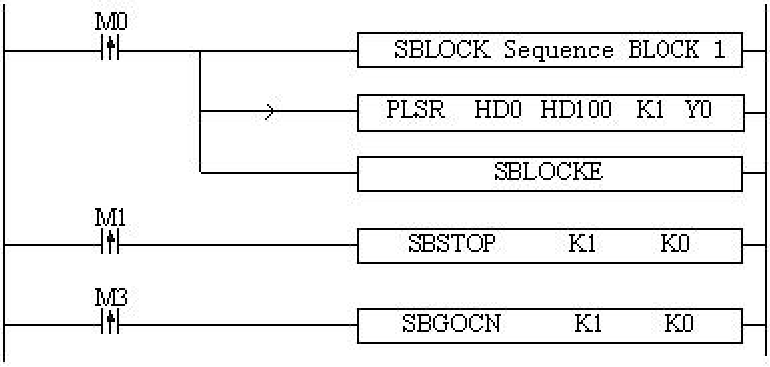
SBSTOP(K1 K0)+SBGOON(K1 K1)

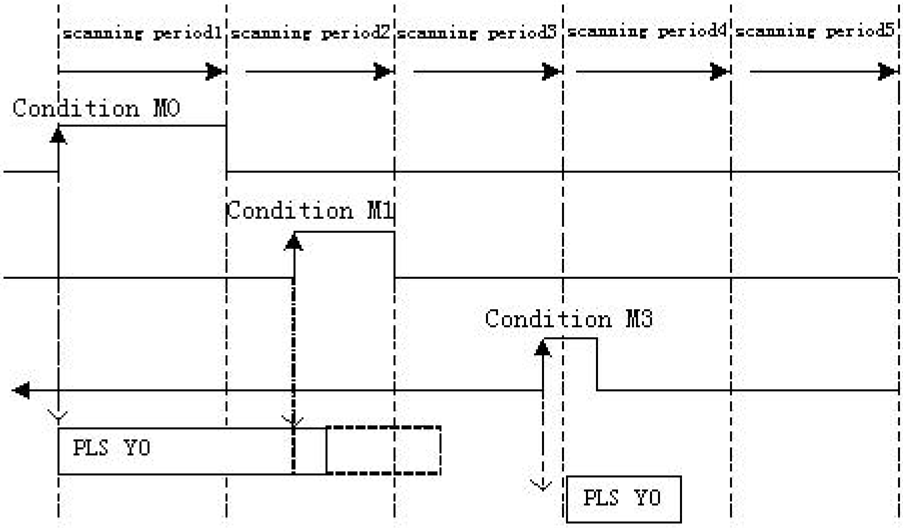




When M0 is from OFF→ON, run ‘PLSR HD0 HD100 K1 Y0’ in the BLOCK to output the pulse;
When M1 is from OFF→ON, stop running the BLOCK, the pulse will stop slowly with slope;
When M4 is from OFF→ON, abandon the rest pulses.

SBSTOP(K1 K0)+SBGOON(K1 K0)





When M0 is from OFF→ON, run ‘PLSR HD0 HD100 K1 Y0’ in the BLOCK to output the pulse;
When M1 is from OFF→ON, suspend running the BLOCK, the pulse will stop slowly with slope;
When M3 is from OFF→ON, output the rest pulses.

Please note that by the SBSTOP stops the pulse with slope, there may be still some pulses; in this case, if run SBGOON K1 K0 again, it will output the rest of the pulses.

9-7 BLOCK flag bit and register

1. BLOCK flag bit:

Address	Function	Explanation
SM300	BLOCK1 running flag	1: running 0: not running
SM301	BLOCK2 running flag	
SM302	BLOCK3 running flag	
…….	…….	
……..	…….	
SM399	BLOCK100 running flag	

2. BLOCK flag register:

Address	Function	Explanation
SD300	BLOCK1 running instruction	BLOCK use this value when monitoring
SD301	BLOCK2 running instruction	
SD302	BLOCK3 running instruction	
…….	…….	
……..	…….	
SD399	BLOCK100 running instruction	

If GBLOCK is used, it will occupy SM399 and SD399.

10 Special Function Instructions

This chapter mainly introduces PWM (pulse width modulation), FRQM, precise timing, interruption etc.

Special Function Instructions List:

Mnemonic	Function	Circuit and soft components	Chapter
Pulse Width Modulation, Frequency Detection			
PWM	Output pulse with the specified duty cycle and frequency	PWM S1 S2 D	10-1
FRQM	Fixed pulses frequency measurement	FRQM S1 D S2 S3	10-2
Time			
STR	Precise Time	STR D1 D2	10-3
Interruption			
EI	Enable Interruption	EI	10-4-1
DI	Disable Interruption	DI	10-4-1
IRET	Interruption Return	IRET	10-4-1

10-1 Pulse Width Modulation [PWM]

1) Summary

Instruction to realize PWM pulse width modulation

PWM pulse width modulation [PWM]			
16 bits instruction	-	32 bits instruction	PWM
execution condition	normally ON/OFF coil	suitable models	XD/XL (except XD1/XL1/XDH/XLH)
hardware requirement	-	software requirement	-

2)Operands

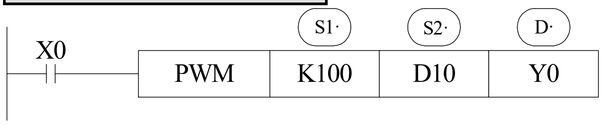
Operands	Function	Type
S1	specify the duty cycle value or soft component's ID number	32 bits, BIN
S2	specify the output frequency or soft component's ID number	32 bits BIN
D	specify the pulse output port	bit

3)Suitable Soft Components

Operands	Word soft elements											Bit soft elements						
	System								Constant	Module		System						
	D	FD	TD	CD	DX	DY	DM	DS	K/H	ID	QD	X	Y	M	S	T	C	Dnm
S1	●	●	●	●					●									
S2	●	●	●	●					●									
D													●					

*Notes: D includes D, HD; TD includes TD, HTD; CD includes CD, HCD, HSCD, HSD; DM includes DM, DHM; DS includes DS, DHS. M includes M,HM,SM； S includes S,HS； T includes T,HT； C includes C, HC.

Function and Action



Duty cycle **n**: 1~65535.

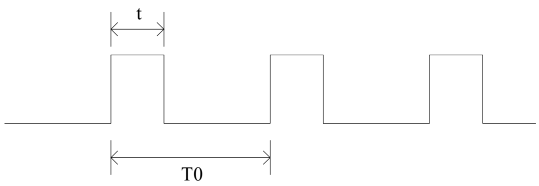
Output pulse **f**: 1~100KHz

XD series PLC PWM output need transistor type terminal:

PLC model	PWM output terminal
XD2-16T/RT -24T/RT -32T/RT -48T/RT -60T/RT	Y0, Y1
XD3-16T/RT-24T/RT-24T4-32T/RT-32T4-42T/RT-48T/RT -60T/RT	Y0, Y1
XD5-16T/RT-24T/RT-32T/RT-42T-48T/RT -60T/RT-80T	Y0, Y1
XD5-24T4 -32T4-48T4 -48T6-60T4 -60T6-60T10	Y0, Y1, Y2, Y3
XD5-24D2T2	Y0, Y2, Y4, Y6
XD5-48D4T4	Y14, Y16
XDM-24T4 -32T4 -60T4 -60T10	Y0, Y1, Y2, Y3
XDC-24T -32T -48T -60T	Y0, Y1
XD3E-24T	Y0, Y1
XD5E-24T -30T -48T -60T	Y0, Y1

PLC model	PWM output terminal
XD5E-30T4 -60T4 -60T6 -60T10	Y0, Y1, Y2, Y3
XDME-30T4 -60T4 -60T10	Y0, Y1, Y2, Y3
XDH-30A16(L)-60T4-60A32-60A64	Y0, Y1, Y2, Y3
XL3-16T/32T	Y0, Y1
XL5-16T/32T	Y0, Y1
XL5E-16T/32T	Y0, Y1
XL5-32T4, XL5E-32T4/64T6, XLME-32T4	Y0, Y1, Y2, Y3
XL5-64T10, XL5E-64T10, XLME-64T10	Y0, Y1, Y2, Y3
XL5N-32T	Y0, Y1
XL5H-24A8L	Y0, Y1
XLH-24A16(L)-30A32(L)	Y0, Y1, Y2, Y3

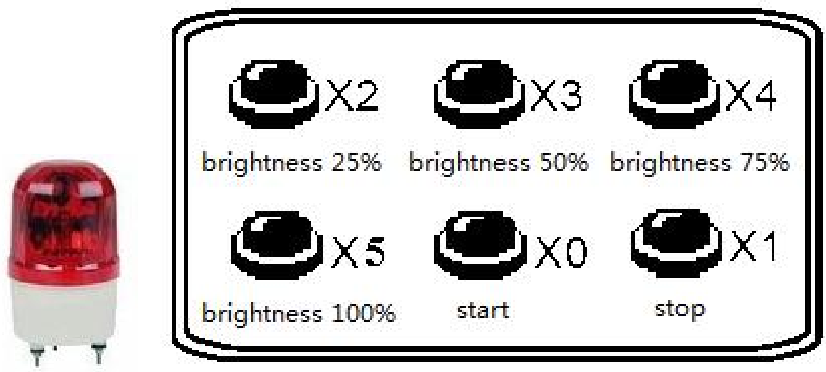
- Duty cycle of **PWM** output =n /65535×100%
- PWM use the unit of 0.1Hz, so when set S2 frequency, the set value is 10 times of the actual frequency (10f). E.g.: to set the frequency as 72 KHz, and then set value in S2 is 720000.
- When X0 is ON, output PWM wave; When X0 is OFF, stop output. PMW output doesn't have pulse accumulation.



In the left graph:
T0=1/f
t/T0=n/65535

Note: it needs to connect 1K ohm amplification resistor between output terminal and common terminal when using PWM instruction.
XDH/XLH models require firmware version V3.7.2 or higher to support this function.

Example



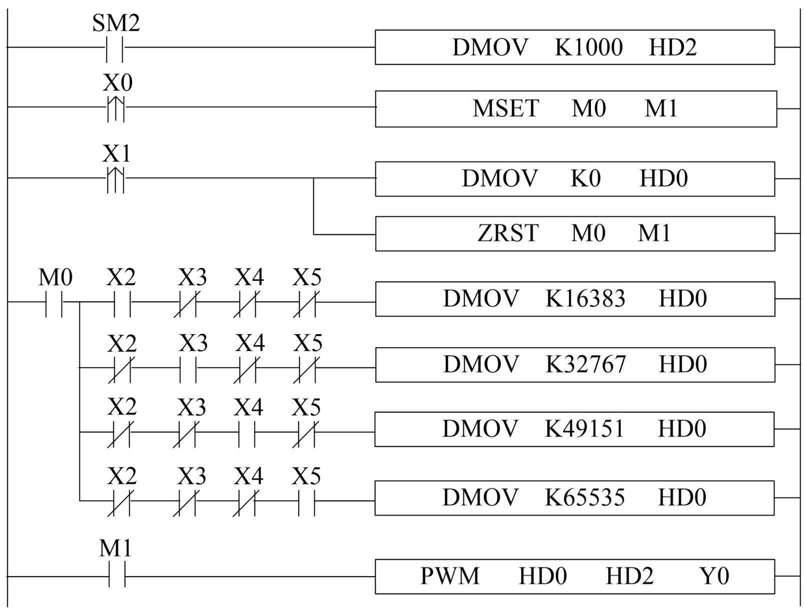
There is a LED drived by DC24V. It needs to control the brightness of the LED. In order to decrease the power loss of wave collector, turn ON the switch at the moment it is OFF, then turn it OFF. This process will cycle. Connet a transistor between the power supply and LED. The pulse signal will input from the transistor base terminal. The current between base and emitter is pulse. The LED input voltage is proportional to the duty ratio. The LED input

voltage will be changed by changing the duty ratio. There are many methods to change the value. The normal way is pulse width modulation (PWM) which means only changing the ON holding time but not changing the ON frequency.
This example applies the PWM technology to the LED brightness adjustment. The controller can accpet 24V PWM control signal. The brightness range includes 25%, 50%, 75%, 100%. The brightness is controlled by the PWM duty ratio.

Element explanation:

PLC component	Explanation	Mark
X0	Start button, X0 is ON when pressed.	
X1	Stop button, X1 is ON when pressed.	
X2	25% brightness button, X2 is ON when pressed.	
X3	50% brightness button, X3 is ON when pressed.	
X4	75% brightness button, X4 is ON when pressed.	
X5	100% brightness button, X5 is ON when pressed.	
HD0	PWM duty ratio register	
HD2	PWM frequency register	Defaulted 100Hz

Program:



Program explanation:

1. HD0 will control the LED voltage. The voltage = 24*HD0/65535, pulse output frequency is 100Hz.
2. Press start button, X0 is ON, M0, M1 is ON, the LED brightness adjustment starts.
3. X2 is ON, HD0=16383, HD0/32768=0.25, the LED brightness is 25%.
4. X3 is ON, HD0=32767, HD0/32768=0.5, the LED brightness is 50%.
5. X4 is ON, HD0=49151, HD0/32768=0.75, the LED brightness is 75%.

6. X5 is ON, HD0=65535, HD0/32768=1, the LED brightness is 100%.
7. Press shut down button, X1 is ON, HD0 is reset, shut down the PWM trigger condition, LED voltage is 0V.

10-2 Frequency measurement [FRQM]

1) Summary

Measure the frequency.

Frequency measurement [FRQM]			
16 bits instruction	-	32 bits instruction	FRQM
execution condition	Normally ON OFF coil	suitable models	XD/XL (except XD1/XL1/XDH/XLH)
hardware requirement	-	software requirements	-

2)Operands

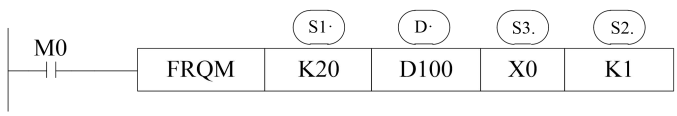
Operands	Function	Type
S1	Sampling pulse numbers	16 bits, BIN
S2	The display precision	16bits, BIN
D	Measurement result	32 bits, BIN
S3	Pulse input terminal	bit

3) Suitable Soft Components

Operands	Word soft elements											Bit soft elements						
	System								Constant	Module		System						
	D	FD	TD	CD	DX	DY	DM	DS	K/H	ID	QD	X	Y	M	S	T	C	Dnm
S1	•	•	•	•					•									
S2	•	•	•	•					•									
D													•					

*Notes: D includes D, HD; TD includes TD, HTD; CD includes CD, HCD, HSCD, HSD; DM includes DM, DHM; DS includes DS, DHS. M includes M,HM,SM； S includes S,HS； T includes T,HT； C includes C, HC.

Function and Action



- The sampling pulse numbers can be adjusted according to the frequency, the higher the frequency, the bigger the sampling pulse numbers.
- Measurement result, the unit is Hz.

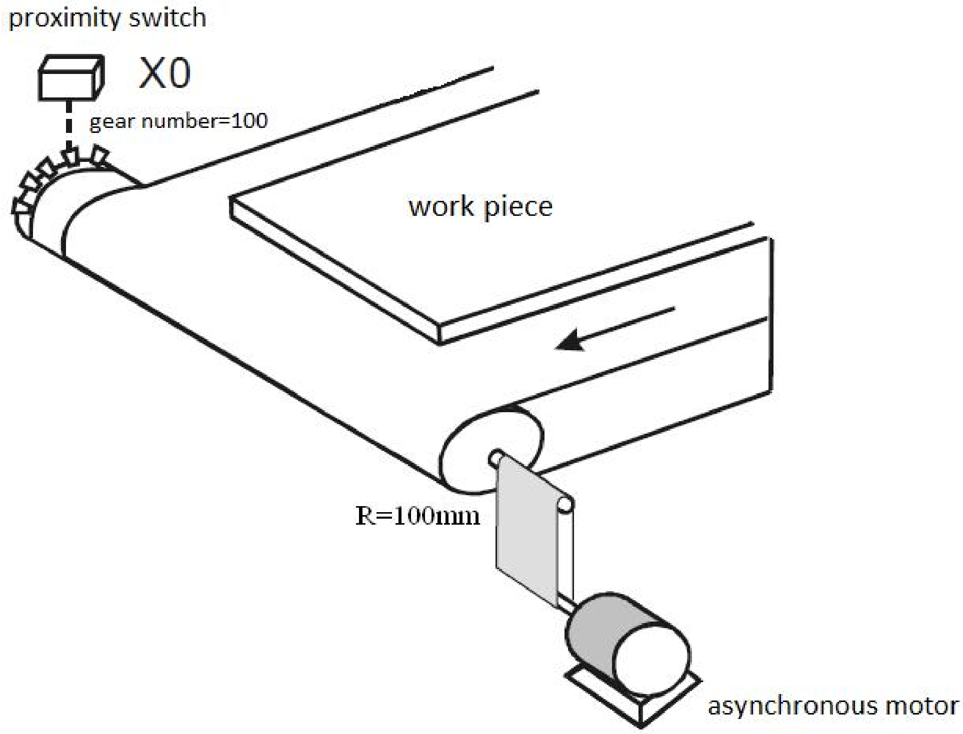
- Display resolution: only can set to 1, 10, 100, 1000, 10000.
- When M0 is ON, FRQM collects 20 pulses from X0, and records the sampling time. The result of sampling numbers dividing by sampling time will be saved in D100. The measurement process will repeat. If the measurement freqeuncy is less than the measurement range, the result is 0.
- The measurement precision is 0.001%.

The pulse input terminal for FRQM:

<table>
<tr><th colspan="2">Model</th><th>X terminal</th><th>Frequency measurement upper limit (Hz)</th></tr>
<tr><td rowspan="3">XD2 series</td><td>16 points</td><td>X0, X3, X6</td><td>10K</td></tr>
<tr><td rowspan="2">24/32/48/60 points</td><td>X0, X3</td><td>80K</td></tr>
<tr><td>X6</td><td>10K</td></tr>
<tr><td rowspan="4">XD3 series</td><td rowspan="2">16/24/32 points</td><td>X0</td><td>80K</td></tr>
<tr><td>X3, X6</td><td>10K</td></tr>
<tr><td rowspan="2">42/48/60 points</td><td>X0, X3</td><td>80K</td></tr>
<tr><td>X6</td><td>10K</td></tr>
<tr><td rowspan="2">XD3E series</td><td rowspan="2">24 points</td><td>X0, X3</td><td>80K</td></tr>
<tr><td>X6</td><td>10K</td></tr>
<tr><td rowspan="4">XD5 series</td><td rowspan="2">16/24/32/42/48/60/80 points</td><td>X0, X3</td><td>80K</td></tr>
<tr><td>X6</td><td>10K</td></tr>
<tr><td>24T4/32T4/48T4/48D4T4/60T4</td><td>X0, X3, X6, X11</td><td>80K</td></tr>
<tr><td>24D2T2/48T6/60T6/60T10</td><td>X0, X3, X6, X11</td><td>80K</td></tr>
<tr><td rowspan="2">XDM series</td><td>24T4/32T4/60T4</td><td>X0, X3, X6, X11</td><td>80K</td></tr>
<tr><td>60T10</td><td>X0, X3, X6, X11</td><td>80K</td></tr>
<tr><td>XDC series</td><td>24/32/48/60 points</td><td>X0, X3, X6, X11</td><td>80K</td></tr>
<tr><td rowspan="3">XD5E series</td><td rowspan="2">24/30/48/60 points</td><td>X0, X3</td><td>80K</td></tr>
<tr><td>X6</td><td>10K</td></tr>
<tr><td>30T4/60T4/60T6/60T10</td><td>X0, X3, X6, X11</td><td>80K</td></tr>
<tr><td>XDME series</td><td>30T4/60T4/60T10</td><td>X0, X3, X6, X11</td><td>80K</td></tr>
<tr><td rowspan="2">XL3 series</td><td rowspan="2">16/32 points</td><td>X0</td><td>80K</td></tr>
<tr><td>X3, X6</td><td>10K</td></tr>
<tr><td rowspan="3">XL5 series</td><td rowspan="2">16/32 points</td><td>X0, X3</td><td>80K</td></tr>
<tr><td>X6</td><td>10K</td></tr>
<tr><td>32T4/64T10</td><td>X0, X3, X6, X11</td><td>80K</td></tr>
<tr><td rowspan="3">XL5E series</td><td rowspan="2">16/32 points</td><td>X0, X3</td><td>80K</td></tr>
<tr><td>X6</td><td>10K</td></tr>
<tr><td>32T4/64T6/64T10</td><td>X0, X3, X6, X11</td><td>80K</td></tr>
<tr><td rowspan="2">XLME series</td><td>32T4</td><td>X0, X3, X6, X11</td><td>80K</td></tr>
<tr><td>64T10</td><td>X0, X3, X6, X11</td><td>150K</td></tr>
<tr><td rowspan="2">XL5H series</td><td rowspan="2">24A8L</td><td>X0, X3</td><td>80K</td></tr>
<tr><td>X6</td><td>10K</td></tr>
<tr><td rowspan="2">XL5N series</td><td rowspan="2">32 points</td><td>X0, X3</td><td>80K</td></tr>
<tr><td>X6</td><td>10K</td></tr>
</table>

Example

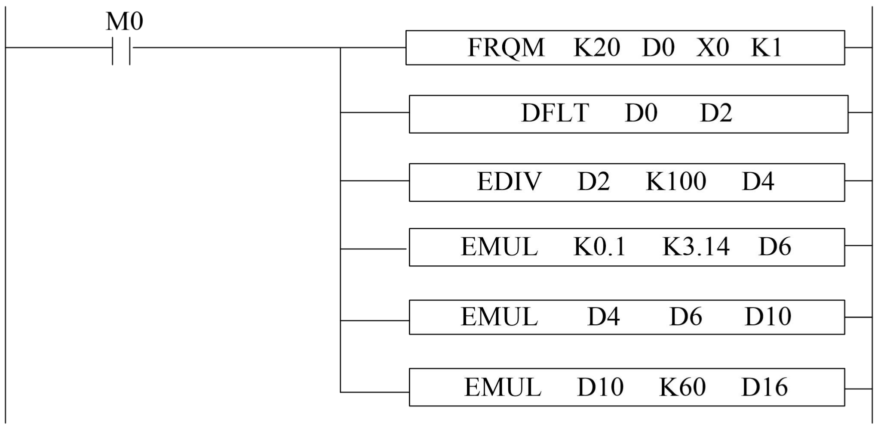
Asynchronous motor drives the conveyor to transfer the work piece. It needs to real-time display the work piece moving speed. The diameter of the transmission shaft is 100mm, the gear numbers on the transmission shaft are 100, the speed unit is m/min.



Component explanation:

PLC component	Control explanation	Mark
X0	Proximity switch, to count the gear numbers	
M0	Start signal	
D16	Speed register (float number)	

Program:



Program explanation:

1. Set ON the start signal M0, to run the frequency meansurement program
2. Transform the frequency to float number, then it is divided by 100 (gear numbers per

rotation), the result is shaft rotate numbers per second (float number).

3. Calculate the diameter of the transmission shaft and save in register D6 (float number), then calculate the transfer distance per second and save in D10 (float number).
4. the transfer distance per second multiply by 60 is the speed (m/min).

10-3 Precise Timing [STR]

1) Summary

Read and stop precise timing when precise timing is executed

Precise timing[STR]			
16 bits instruction	-	32 bits instruction	STR
execution condition	edge activation	suitable models	XD/XL (except XDH,XLH)
hardware requirement	-	software requirements	-

2)Operands

Operands	Function	Type
D1	Timer Number	bit
D2	specify timer's value or soft component's ID number	32 bits, BIN

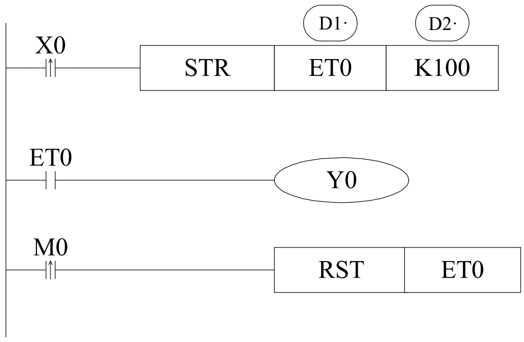
3)Suitable Soft Components

Operands	Word soft elements											Bit soft elements						
	System								Constant	Module		System						
	D	FD	TD	CD	DX	DY	DM	DS	K/H	ID	QD	X	Y	M	S	T	C	Dn.m
D																●		
D1																●		
D2	●	●	●	●					●									

*Note: D includes D, HD; TD includes TD, HTD; CD includes CD, HCD, HSCD, HSD; DM includes DM, DHM; DS includes DS, DHS. M includes M, HM, SM; S includes S and HS; T includes T and HT; C includes C and HC.

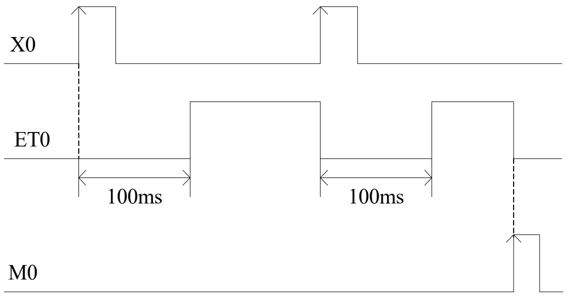
Function and Action

<Precise timing>, <Precise timing reset>



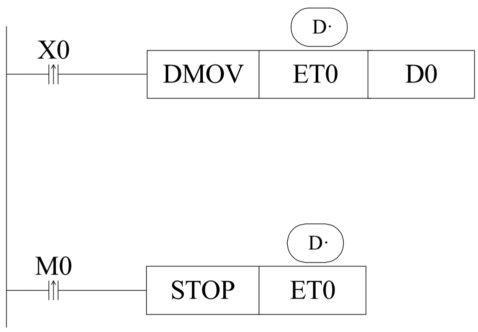
D1· :Timer's number. Range: ET0~ET30 (ET0, ET2, ET4……all number should be even)
D2· :Timing value

- Precise timer works in unit of 1ms.
- Precise timer32 bits, the counting range is 0~+2,147,483,647.
- When executing STR, the timer will be reset before start timing.
- When X0 turns from OFF to ON, ET0 starts timing. ET0 will be reset and keep its value 100 when accumulation time reaches 100ms; If X0 again turns from OFF to ON, timer T600 turns from ON to OFF， restart to time, when time accumulation reaches 100ms, T600 reset again. See graph below:



When the pre-condition of STR is normally open/closed coil, the precise timer will set ON immediately when the timing time arrives and reset the timing, and cycle back and forth.

<read the precise timing>, <stop precise time>



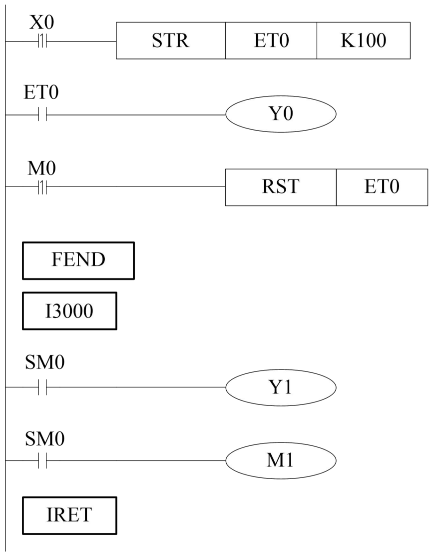
- When X0 changes from OFF to ON, move the current precise timing value into D0 immediately, it will not be affected by the scan cycle;
- When M0 changes from OFF to ON, execute STOP instruction immediately, stop precise timing and refresh the count value in ETD0. It will not be affected by the scan cycle;

PreciseTiming Interruption

- When the precise timing reaches the count value, it will generate an interruption tag, interruption subprogram will be executed.
- Can start the precise timing in precise timing interruption;
- Every precise timer has its own interruption tag, as shown below:

Interruption Tag corresponding to the Timer:

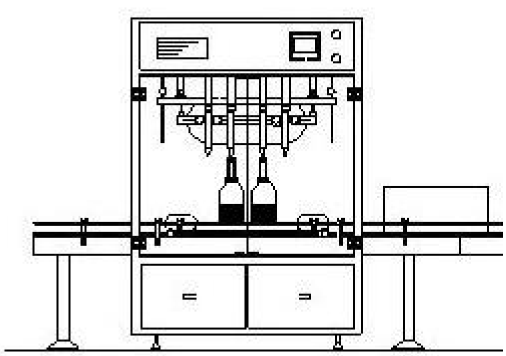
Timer's No	Interruption Tag	Timer's No	Interruption Tag
ET0	I3000	ET10	I3005
ET2	I3001	ET12	I3006
ET4	I3002	……	……
ET6	I3003	ET22	I3011
ET8	I3004	ET24	I3012



When X0 changes from OFF to ON, ET0 will start timing. And ET0 reset when accumulation time is up to 100ms; meantime generates an interruption, the program jumps to interruption tag I3000 and execute the subprogram.

Example 1

The filling machine controls the filling capacity by controlling the liquid valve open time (it is 3000ms in this application). To improve the filling capacity precision, the liquid valve open time can be controlled by precise timing.

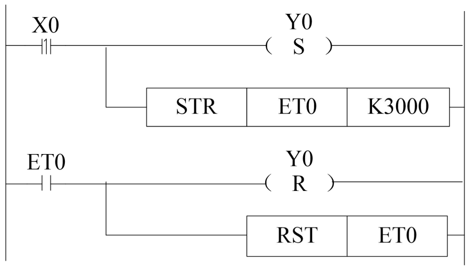


Filling machine

Component explanation:

PLC component	Control explanation	Mark
X0	Start button, X0 is ON when the button is pressed	
ET0	Precise timer	
Y0	Control the liquid valve, Y0 ON when the valve opened, Y0 OFF when the valve closed	

Program:

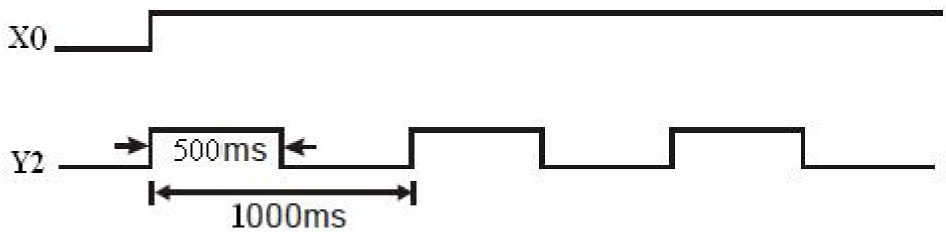


Program explanation:

1. When X0 is ON, the liquid valve Y0 and precise timer ET0 open at once.
2. Shut down the liquid valve Y0 and precise timer ET0 when the time arrived.

Example 2

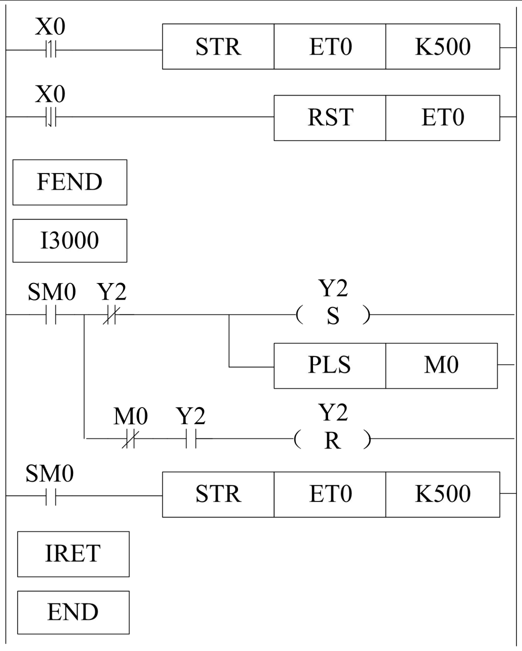
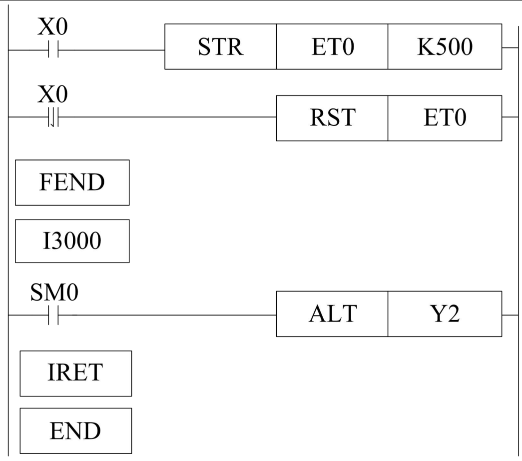
The precise timer interruption can produce the following pulse wave. The Y2 ON time is 500ms, the pulse period is 1000ms.



Component explanation:

PLC component	Control explanation	Mark
X0	Start button, X0 is ON when button is pressed	
Y2	Pulse output terminal	
M0	Internal auxiliary coil	
ET0	Precise timer	

Program:

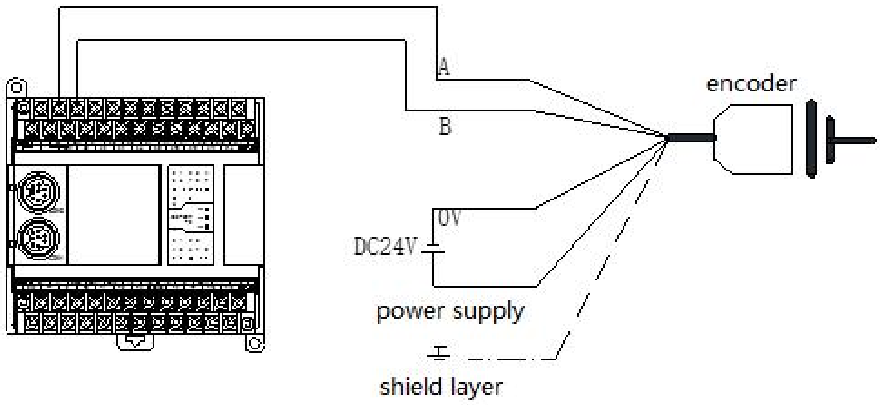
Method 1	Method 2
	

Program explanation:

1. When X0 is ON, the precise timer interruption will work, Y2 will output the pusle wave.
2. When X0 is OFF, shut down the precise timer interruption, Y2 stop outputting.

Example 3

As the FRQM calculating the time for fixed pulse numbers, we will change the way to calculate the pulse numbers in fixed time.

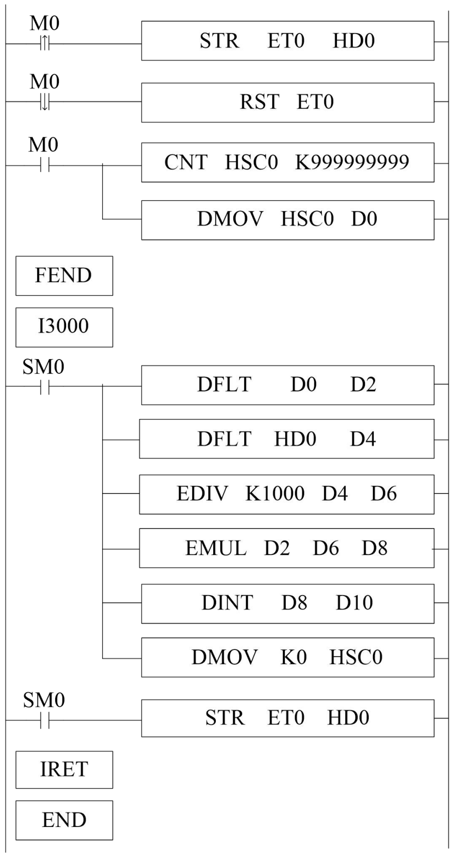


Component explanation:

PLC component	Control explanation	Mark
M0	Start button, X0 is ON when pressed	
ET0	Precise timer	
HD0	Precise timer setting value (unit: ms)	
HSC0	High speed counter	

D10	The measured frequency (unit: s)	

Program:



Program explanation:

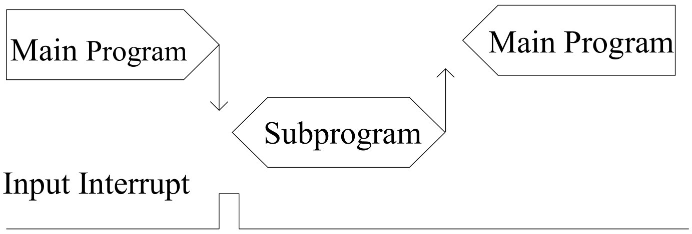
1. Set the high speed counter sampling period register HD0, the unit is ms.
2. Set ON M0 to start the precise timer interruption and high speed counter, calcuate the frequency
3. The frequency range is 0-80KHz, the precision is 0.005%.

10-4 Interruption [EI], [DI], [IRET]

XD/XL series PLC have interruption function, including external interruption and timing interruption. By interruption function we can deal with some special programs. This function is not affected by the scan cycle.

10-4-1 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.



Note: The external interruption of XC series PLC cannot be activated by rising edge and falling edge at the same time; but XD/XL series PLC supports rising edge and falling edge activation meantime.

External Interruption's Port Definition

XD/XL series 16 I/O

Input terminal	Pointer No. Rising Interruption	Pointer No. Falling interruption	Disable the interruption instruction
X2	I0000	I0001	SM050
X3	I0100	I0101	SM051
X4	I0200	I0201	SM052
X5	I0300	I0301	SM053
X6	I0400	I0401	SM054
X7	I0500	I0501	SM055

XD series 10 I/O

Input terminal	Pointer No. Rising Interruption	Pointer No. Falling interruption	Disable the interruption instruction
X2	I0000	I0001	SM050
X3	I0100	I0101	SM051
X4	I0200	I0201	SM052

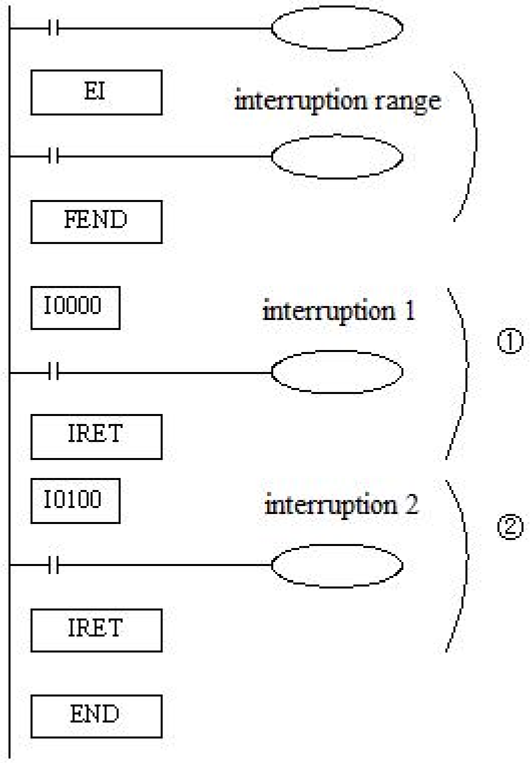
XD/XL series 24~64 I/O

Input terminal	Pointer No. Rising Interruption	Pointer No. Falling interruption	Disable the interruption instruction
X2	I0000	I0001	SM050
X3	I0100	I0101	SM051
X4	I0200	I0201	SM052
X5	I0300	I0301	SM053
X6	I0400	I0401	SM054
X7	I0500	I0501	SM055
X10	I0600	I0601	SM056
X11	I0700	I0701	SM057
X12	I0800	I0801	SM058
X13	I0900	I0901	SM059

Note: when the interruption ban coil is ON, the external interruption will not execute. External interrupt terminals that have been used as external interrupt signals in the program cannot be used as the origin of ZRN and EXT signals for Z-phase and PLSR.

Interruption Instruction

Enable Interruption [EI], Disable Interruption [DI], Interruption Return [IRET]



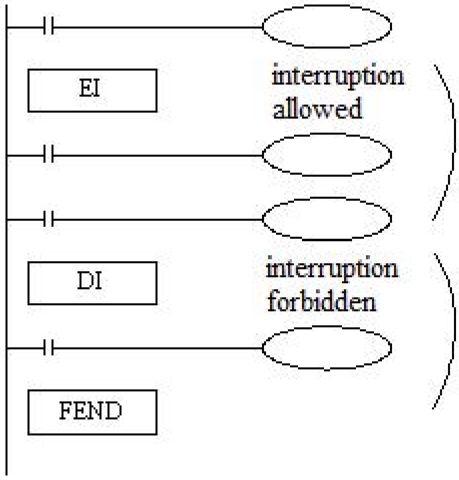
- If use EI instruction to allow interruption, then when scanning the program, if interruption input changes from OFF to ON, then execute subroutine ①、②. Return to the original main program.
- Interruption pointer (I****) should be behind FEND instruction;
- PLC is usually on the status that allows interruption.

Note:

(1) In interrupt subroutine, only simple instructions such as set, reset, transmission and operation can be written, which can be executed in a scanning cycle. Other instructions such as sending pulses, timing (except for precise timing), communication and other instructions that need to be continuously executed are not supported.

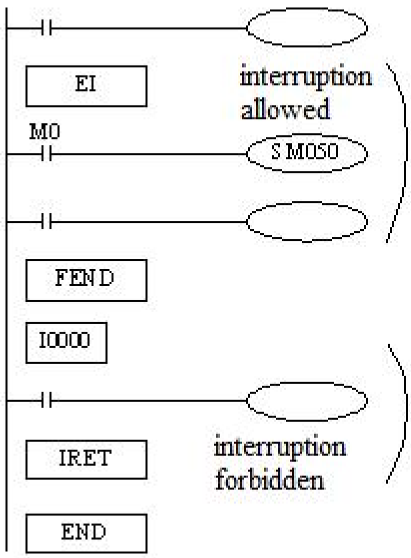
(2) There are a total of 20 timed interrupts (I40**~I59**), and only one time can be used for the same timed interrupt. For example, if I4010 (timed interrupt executed every 10ms) is used for interrupt I40**, then I4050 (timed interrupt executed every 50ms) cannot be used. It can be written as I4150.

Interruption's Range Limitation



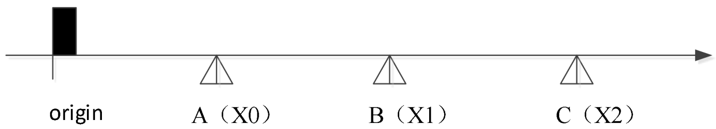
- By programming DI instruction, can set interruption disabled area;
- Allow interruption input between EI~DI
- If interruption forbidden is not required, please program only with EI, and program with DI is not required.

Disable the Interruption



- Every input interruption is equipped with special relays (SM50~SM69) to disable interruption.
- In the left program, if use M0 to set SM50 "ON", then disable the interruption 0.

Example 1



The positions of A, B, C are unknown. The speed of the three segments are different. The application can be perform by PLSF instruction and external interruption. We can install three proximity switch at postion A, B, C, and connect the signal to PLC input terminal X0, X1, X2. (suppose X0, X1, X2 are external interruption terminal, the related rising edge interruption ID are I0000, I0100, I0200. The PLC external interruption terminal please refer to "external interruption terminal definition). The pulse terminal is Y0, the direction terminal is Y2. To improve the speed changing precision, the acceleration and deceleartion time are 0. The speed will switch by external interruption.

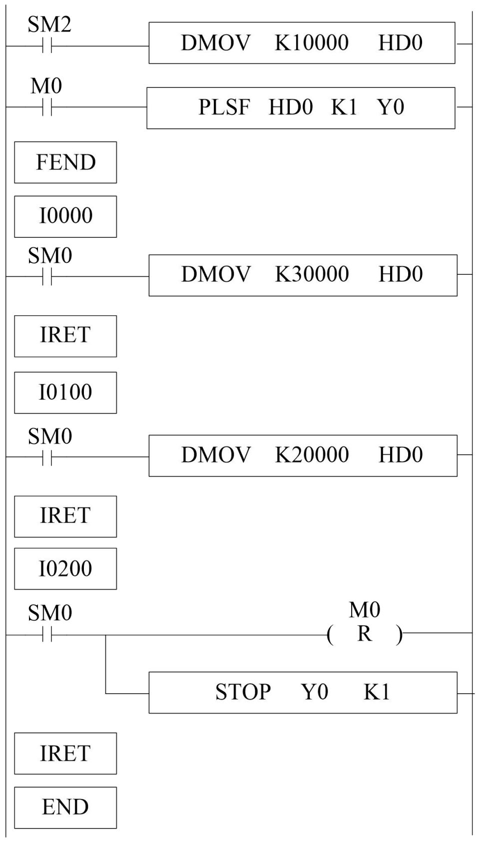
Segment	Frequency setting value (Hz)	Pulse numbers
Origin ---- A	10000	999999999
A---- B	30000	999999999
B ----- C	20000	999999999
Acceleration and deceleratoin time	0	

Note: as the pulse numbers of each segment is unknown, the pulse numbers should set large enough to ensure the object can move to the proximity switch. The STOP instruction will be run by external interruption when the object gets to position C.

Component explanation

PLC component	Control explanation	Mark
M0	Start button, PLSF will send pulse when the button is pressed	
HD0	the PLSF pulse frequency register	

Program

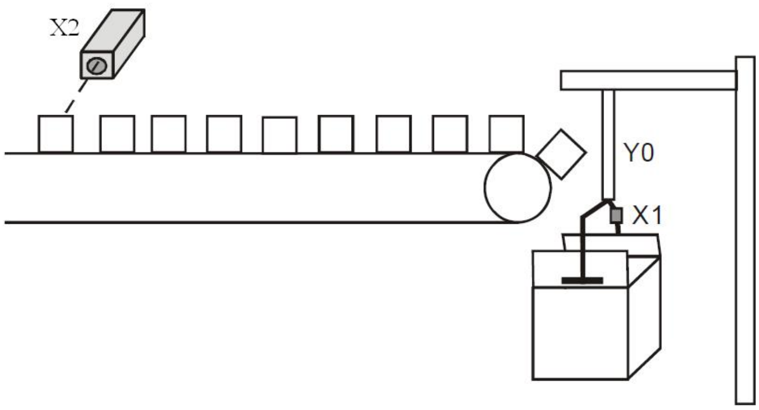


Program explanation

1. SM2 is ON, set HD0 to 10000, set on M0, PLSF instruction will send 10000Hz pulse, the object will move from origin to A.
2. When the object touches A, X0 will be ON at once, the external interruption I0000 will work, HD0 is set to 30000, the object will move from A to B with the speed of 30000Hz.
3. When the object touches B, X1 will be ON at once, the external interruption I0100 will work, HD0 is set to 20000, the object will move from B to C with the speed of 20000Hz.
4. When the object touches C, X2 will be ON at once, the external interruption I0200 will work, M0 is set OFF, the pulse sending will stop at once.

Example 2

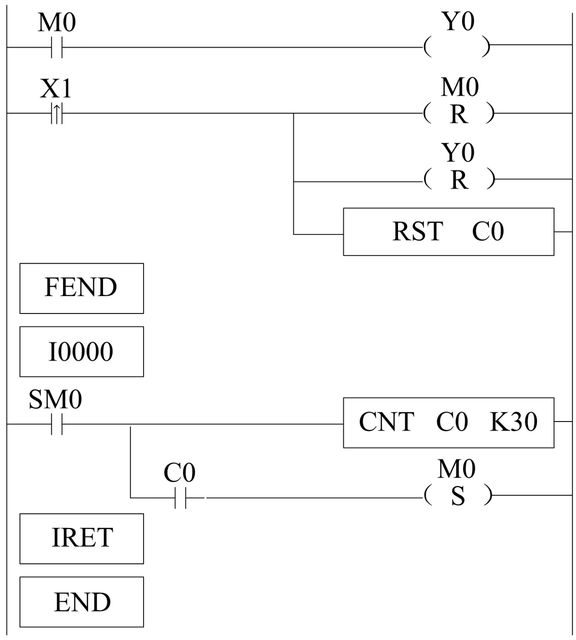
The diagram is the product packing machine. The robot will pack the product when 30 products are detected, the robot and counter will be reset after packing completed. To improve the working efficiency, the product sending speed is very fast, the sensor X2 detects the product time is 8ms, PLC input terminal filter time is 10ms, the normal counter cannot detect the products. We can use the external interruption to count the products.



Component explanation:

PLC component	Control explanation	Mark
X2	Product counting photoelectric sensor, X2 is ON when the product is detected	
X1	Robot action complete sensor, X1 is ON when the action is completed	
C0	16-bit counter	
Y0	Robot	

Program:



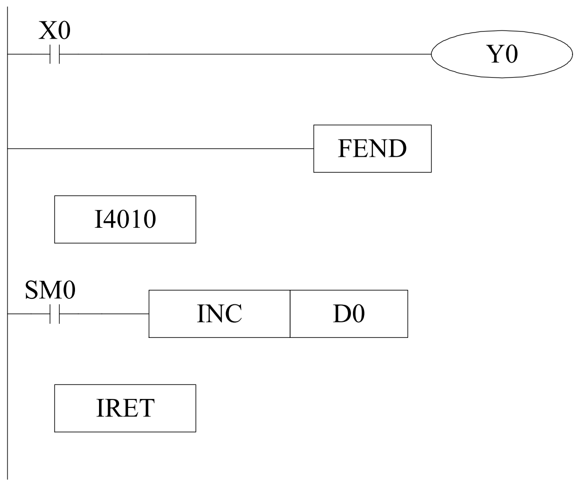
Program explanation:

1. In the external interruption program, count the X2 input, when the X2 is 30, set ON M0
2. In the main program, it controls the Y0 according to the M0 state.
3. When the robot action is completed, X1 changes from OFF to ON once, RST works, Y0 and C0 are reset, M0 is OFF, wait for the next packing process.

10-4-2 Timing Interruption

Function and Action

Under the circumstance that the main program execution cycle is very long, when you have to handle with special program or execute specific program every once in a while when program is scanning in sequence control, the timing interruption is very useful. It is not affected by PLC scan cycle and executes timing interruption subroutine every N ms.



- Timing interruption is open status in default, just like other interruption subroutines, it should be written behind the main program, starts with I40xx, ends with IRET.
- There are 20 channels of timing interruptions, representation: I40**~I59**('**'means interruption time; Unit is ms. E.g: I4010 means executing once the first timing interruption per 10ms.

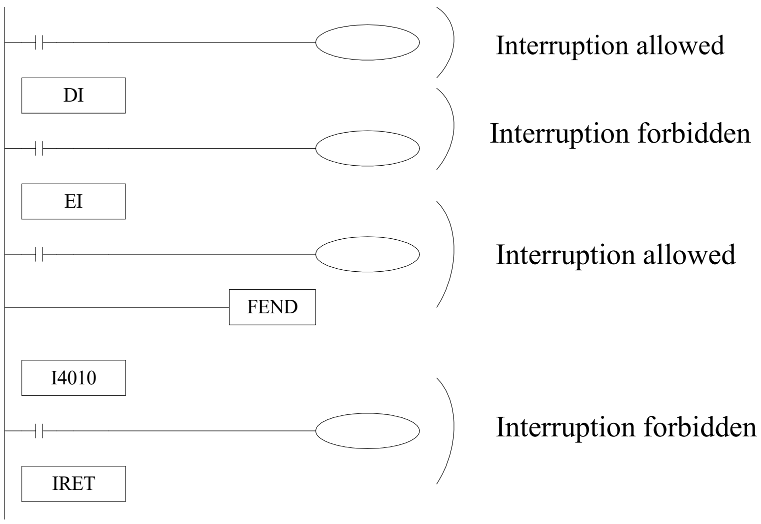
Interruption No

XD, XL series timing interruption:

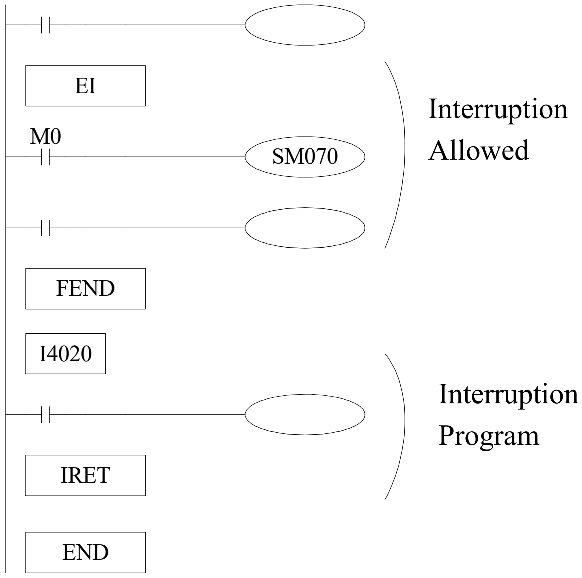
Interruption number	Interruption ban instruction	Interruption number	Interruption ban instruction	Explanation
I40**	SM070	I50**	SM080	** means the timing interruption time, the range is 1~99, the unit is ms. **Note:** **1, only I59** timing interrupt can support 100us .** **2. Time-base selection function is only supported by PLC firmware version V3.4.6 or later.**
I41**	SM071	I51**	SM081	
I42**	SM072	I52**	SM082	
I43**	SM073	I53**	SM083	
I44**	SM074	I54**	SM084	
I45**	SM075	I55**	SM085	
I46**	SM076	I56**	SM086	
I47**	SM077	I57**	SM087	
I48**	SM078	I58**	SM088	
I49**	SM079	I59**	SM089	

Interruption range's limitation

- Timing interruption is usually on 'allow' status.
- Can set interruption allow and forbidden area with EI、DI instructions. As shown in below pictures, all timing interruptions are forbidden between DI and EI, and allowed beyond DI~EI.



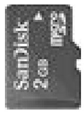
Interruption Forbidden



- The first 3CH timing interruptions are equipped with special relays (SM070~SM079).
- In the left example, if use M0 to set SM070 "ON", then forbid timing interruption forbidden.

10-5 SD card reading and writing

XD5 (except XD5-16), XDM series PLC body can be extended with an SD card for data storage and backup. The SD card slot is located on the CPU board of the PLC. When using, you need to lift the BD cover plate and insert the SD card into the card slot.



The SD card is not installed when the PLC leaves the factory. Users need to bring their own microSD (TF Card), and the card capacity must not be greater than 32GB.
Before installing the SD card on the PLC, please use the card reader to format it into FAT32 format on the computer.
Note:
1. The PLC of Ethernet model does not support SD card.
2. The use of SD card conflicts with some communication ports of PLC. XD5-32/60, XD5/XDM-32T4/60T4 conflict with COM4, and XD5-60T6/T10, XDM-60T10 conflict with COM4 and COM5.

10-5-1 Document content and format

SD card supports four data types, including Single Word (W), Double Word (DW), Floating point (Fm.n) and Character (Sx).
As shown in the following figure, the first row in the excel file declares the data type:

	A	B	C	D	E	F	G
1	w	dw	s8	f4.15	dw	w	dw
2	-32765	-32770	hellbaby	1237.20100156164	30000	999	3121
3	454	-91877301	testh	2351.25150102545	-454532088	-15453	124522

The data range and occupied space of each type are shown in the following table

Data type	W	DW	Fm.n(m<=15,n<=15)	Sx(x<=16)
Data range	-32768~32767	-2147483648~2147483647	-18446742974197923840~18446742974197923840	\
Number of characters occupied in SD card	6	11	m+1+n	2*x
Number of WORD	1	2	2	x

Notes:
※ 1: when the actual length of the data is less than the number of characters stored in the SD card, the space from the left is used to fill in. For example, a single word data is 454, which is less than the 6 characters occupied by W type, so fill 6-3=3 spaces from the left, and the actual occupation is "∟ ∟ ∟ 454" (∟ represents spaces).
※ 2: when Fm.n is a negative number, the sign bit "-" also occupies a character. For example, the defined floating-point type is F5.3. After writing the data "-12345.123" to the SD card, the least significant bit will be deleted and the data will become "-12345.12".
※ 3: the x of the character Sx represents the word length, not the character length.

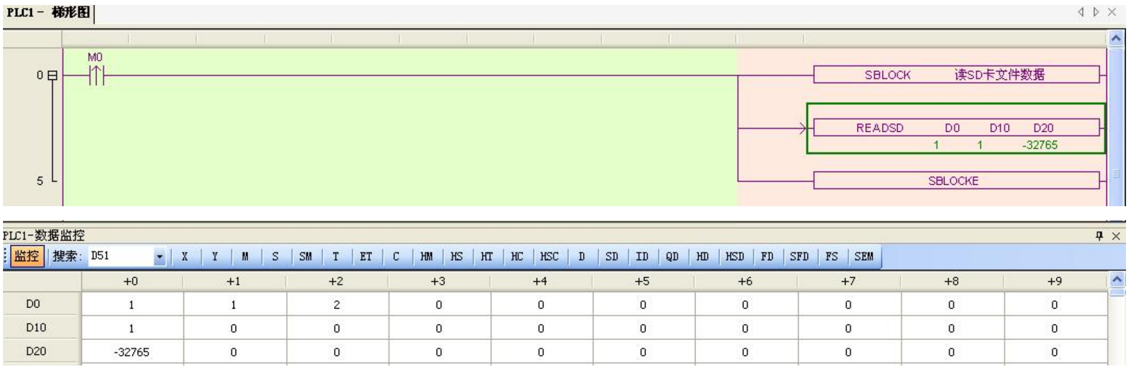
※ 4: Due to the fact that Excel does not display the current number of spaces in front and line breaks behind, it is recommended to use Notepad on the SD card for editing to see more clearly.

10-5-2 File name and storage location

SD card supports storing ".csv" format files, which must be stored in the root directory of SD card.
All ".csv" files must be named after "dataxxx.csv", and "xxx" is the index number of the file, ranging from 001 to 999. When xxx is less than 100, add 0 from the left to supplement. If the index number of the file is 1, the file is correctly named "data001.csv".

10-5-3 Read/write SD card

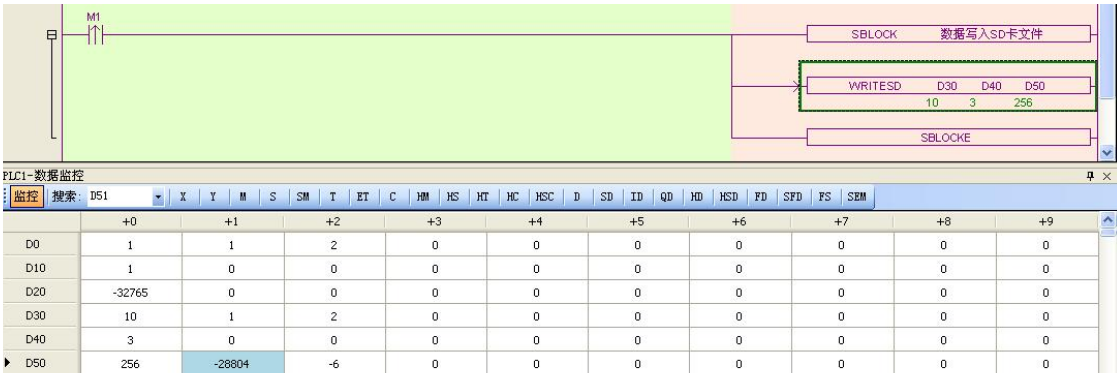
The instruction to read and write SD card is in the sequential function block block. Open the block function interface and insert "read and write SD module". The following is an introduction to the instructions for reading and writing SD cards.
1) Read SD card
Add an SD card reading instruction in the sequence function block, as shown in the following figure:



In the above figure, the SD card reading instruction is: READSD D0 D10 D20, and the function meanings of each parameter are as follows:
D0: file index number / column / line beginning address, D0~D2 refers to file index number, column number and line number respectively.
As shown in the data monitoring above, D0=1, D1=1, D2=2, indicate:
The index number of the file is 1: that is, the file name is data001.csv.
Column number 1: column 1.
Line number is 2: Line 2 (data line 2).
D10: number of words to read data.
As shown in the data monitoring above, D10=1 means reading 1 word data.
D20: the read data is stored in the first address of the PLC body.
As shown in the data monitoring above, the data read in D20 is -32765.

Therefore, the specific meaning of the above program is to read the file name data001 under the root directory of the SD card CSV, and save the read data in register D20.
If the number of columns is 0, the number of words read from the data indicates the number of rows read. For example, when D0=1, D1=0, D2=1, D10=5, it means that the data of 5 lines is read from the first line of the file data001.csv and stored in the register starting from D20.

2)write SD card
Add a write SD card instruction in the sequence function block, as shown in the following figure:



	+0	+1	+2	+3	+4	+5	+6	+7	+8	+9
D0	1	1	2	0	0	0	0	0	0	0
D10	1	0	0	0	0	0	0	0	0	0
D20	-32765	0	0	0	0	0	0	0	0	0
D30	10	1	2	0	0	0	0	0	0	0
D40	3	0	0	0	0	0	0	0	0	0
D50	256	-28804	-6	0	0	0	0	0	0	0

The SD card instruction : WRITESD D30 D40 D50, and the function meanings of each parameter are as follows:
D30: file index number / column / line beginning address, D30~D32 represent file index number, column number and line number respectively.
As shown in the data monitoring above, D30=10, D1=1, D2=2 mean:
The index number of the file is 10: that is, the file name is data010.csv.
Column number 1: column 1.
Line number is 2: Line 2 (data line 2).
D40: number of words to read data.
As shown in the data monitoring above, D40=3 indicates the data written into 3 words.
D50: the written data is stored in the first address of the PLC body.
As shown in the data monitoring above, write the register data starting from D50 into the SD card.
Therefore, the specific meaning of the above program is: write the 3 word data starting from register D50 to the root directory of SD card, and the file name is data010.csv at column 1 and row 2.
If the number of columns is 0, the number of words written to the data represents the number of rows written. For example, when D30=2, D31=0, D32=3, D40=5, it means that the five rows of data starting from D50 are written into the third to seventh rows of data002.csv.

10-5-4 Notes

1)Only visible characters (letters and numbers) are supported in the character type, that is, the characters of the [32,126] range in the ASCII code table.

Exceptions:

- Support end character in invisible characters
- Don't support "," in visible characters

2)Csv file restrictions

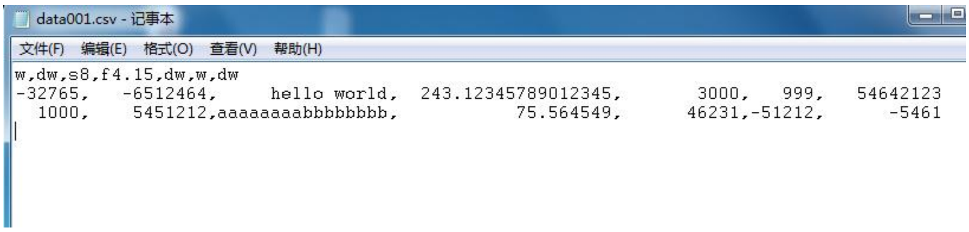
- The number of columns in .csv file cannot exceed 20.
- The characters of the file data line cannot exceed 512 characters(including commas and line breaks

3)Limit the number of reading and writing data of word

A piece of data cannot be read only part of it. For example, the format is defined as W, DW, S8. If you read from the first column W and the number of words read is 10, S8 cannot be read completely, and the program will report an error (the correct number of words is 3, or 11). When the program judges that the parameters are wrong, it will not read or write the SD car.

4) Create CSV file restrictions

- if you use the notepad provided by Windows to create a CSV file, the end of the last line must be wrapped, and the data is created strictly according to the type definition (table in section 10-5-1), as shown in the following figure:



- existing CSV files can only be used after being converted by using the format conversion tool in section 10-5-5.

5) Limit in read and write

In the read instruction, if the specified number of lines exceeds the actual number of lines in the file, the program will report an error.

6) Write process description

In the write instruction, if the specified number of lines exceeds the actual number of lines of the file, the file will be extended. During the expansion process, insert spaces in the corresponding positions according to the column data type, leaving a fixed space for inserting data later.

Notes:

If the number of lines written is greater than the actual number of lines in the file, the file will be expanded. The file expansion speed is very slow, which will affect the PLC scanning cycle and cause the watchdog to trigger. This is not recommended.

7) SD card status information

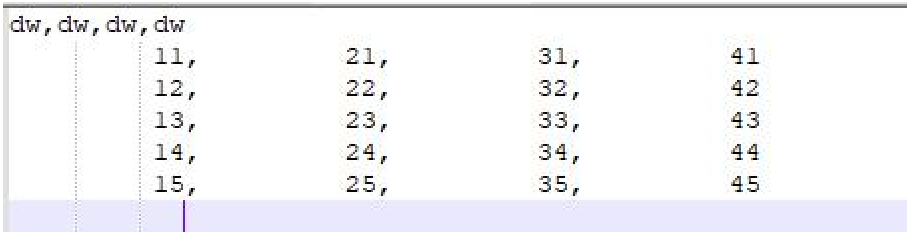
The status information of the SD card can be viewed through the special register SD453, as shown in the following table:

SD453	Explanation	Reason
0	Operational successful	
1	Reserve	
2	Reserve	
3	Reserve	
4	Read/write files does not exist	File doed not exist
5	Read file to end	End of file reached
6	Reading file	There are tasks reading files
7	Writing file	There are tasks writing files
8	Read/write error	Unplug SD card while reading and writing
9	Insufficient SD card space	Insufficient SD card space
10	Reserve	
11	FAT32 error	SD card is not inserted properly, or the SD card is not formatted as FAT32
12	Reserve	
13	Reserve	
14	SD card cannot be initialized	SD card is not inserted properly
15	Reserve	
16	Reserve	
17	Reserve	
18	Reserve	
19	SD card cannot be detected	SD card not inserted
20	Error in reading and writing parameters	Check index，row，line，wordcnt
21	The read/write data does not conform to the format definition	Miss data type definition
22	Wrong data type of file	Appear type definitions other than W,DW,Sn,Fx.y
23	The data type of the file is not defined	Floating point and character data do not meet the definition
24	Illegal file name	Index>999
25	Illegal column index	column index is greater than the number of file columns
26	Illegal row index	Row index is less than 1
27	Illegal read and write words	The number of words read and write is less than 1
28	Illegal read and write words	Word number cannot read data completely
29	The number of characters in the file data line exceeds the limit	Data line characters exceed 512 characters
30	Other errors	

10-5-5 Application example

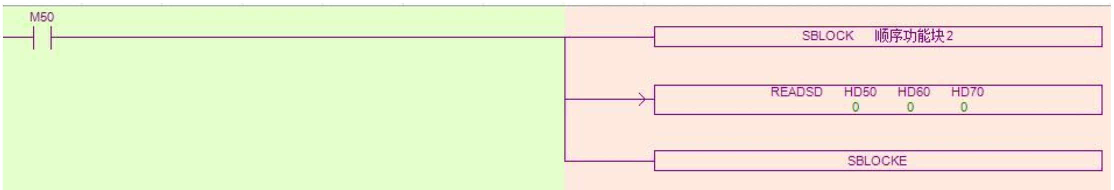
Operation case 1: The data in the document has the same format

1. Create a new CSV format table, it is recommended to right-click and open it in Notepad to visually see the placeholders in front, and name it data010.csv.
2. For each data type, it is necessary to correspond with the number of characters. For example, writing a two character integer dw will take up 11 characters and needs to be written as ∟∟∟∟∟∟∟∟∟ 11, where ∟∟∟ represents a space character, as shown in the figure:

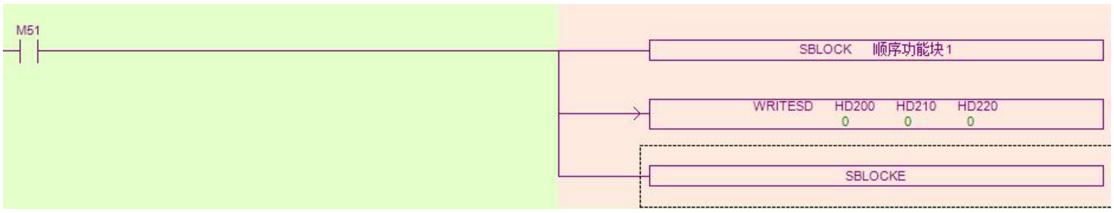


Note: When creating a new file, it is necessary to wrap the line after the last digit and add an 11 bits space symbol (provided based on the actual type of data in the file, and if it is a single character register, add a 7-bit space symbol) to ensure that the data can be read correctly.

3. Write the instruction in the form shown in the following figure:



Read SD card instruction



Write SD card instruction

HD50/HD200- corresponds to the ID number in the file name, HD51- corresponds to the modified column number, HD52- corresponds to the modified row number, set HD50 to 10, HD51 set to 1, and HD52 set to 1, that is, the first column and first row;
HD60/HD210- corresponds to the total number of WORD files that need to be read, requiring a total of 10 double word data to be modified. Each double word data occupies 2 words, and HD210 fills in 20. The same applies to reading data;
HD220-HD238 corresponds to the data that needs to be written to the SD card, for example, the first 10 data need to be changed to all 99 data, and the same applies to reading data;
It is recommended to monitor SD453 (SD card error message) before conducting the command. If SD453 does not report an error code after conducting the command, it means that the writing is successful.

4. Open the file in the SD card again to check the data status. Currently, the SD card data is written in the order of the first column, first row ->second column, first row ->third column,

first row ->fourth column, first row ->first column, second row ->second column, second row ->third column, second row ->first column, third row ->second column, third row, as shown in the following figure.

```
dw,dw,dw,dw
        99,          99,          99,          99
        99,          99,          99,          99
        99,          99,          33,          43
        14,          24,          34,          44
        15,          25,          35,          45
```

Operation case 2: The data in the document is in different formats

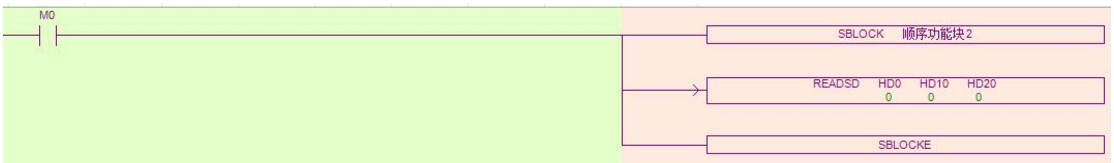
1. Create a new CSV format table, it is recommended to right-click and open it in Notepad to visually see the placeholders in front, and name it data006.csv
2. For each data type, it is necessary to correspond with the number of characters. For example, when writing a floating-point number f4.2, it needs to be written as ∟∟∟ 1.10, where ∟ represents a space character, as shown in the figure:

```
w,dw,f2.2,s2
    11,          21,  3.1,  41
    12,          22,  3.2,  42
    13,          23,  3.3,  43
    14,          24,  3.4,  44
    15,          25,  3.5,  45
```

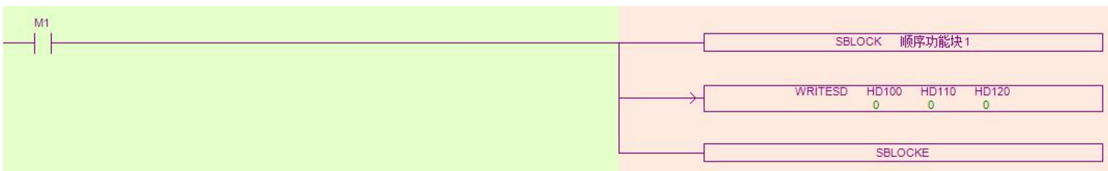
	w	dw	f2.2	s2
The number of characters occupied separately	6	11	5	4

Note: The data types in this file are in different formats, and a line break needs to be added after the last data.

3. Write the instruction in the form shown in the following figure:



Read SD card instruction



Write SD card instruction

HD0/HD100- corresponds to the ID number in the file name, that is, the data in HD0 is 006, HD1- corresponds to the modified column number, HD1 is 2, HD2- corresponds to the modified row number, and HD2 is 1, which is the first row of the second column; HD10/HD110- corresponds to the total number of WORD characters that need to be modified, including 1 double word data, 1 floating-point data, and 1 character. HD10 is 6, and the same applies to reading data, ensuring consistency in data type;

	w	dw	f2.2	s2
Number of words occupied separately	1	2	2	2

HD120-HD124 corresponds to the data that needs to be written to the SD card. For example, in the file, 21, 3.1, and 41 are changed to 28, 3.8, and 48 in sequence. The data type of HD120 is double character, and the value is 28. The data type of HD122 is floating-point, and the value is 3.8. The data type of HD124 is character format, and value is 48. The same applies to reading data, and the data type needs to be kept consistent;

It is recommended to monitor SD453 (SD card error message) before conducting the command. If SD453 does not report an error code after conducting the command, it means that the writing is successful.

4. Open the file on the SD card again to check the data status. Currently, the SD card data is written in the order of second column, first row ->third column, first row ->fourth column, first row ->, as shown in the following figure.

```
w,dw,f2.2,s2
    11,         28,  3.8,  48
    12,         22,  3.2,  42
    13,         23,  3.3,  43
    14,         24,  3.4,  44
    15,         25,  3.5,  45
```

10-6 Multi station control[MSC]

1) Summary

Grab the encoder value according to the trigger input, calculate and save the entry value and exit value of the workpiece in each station, compare the stored value of each workpiece in each station with the current value of the encoder, and output the comparison result.

Multi station control[MSC]			
16 bits	-	32 bits	MSC
Execution condition	Normally ON/OFF	Suitable Models	XD5, XDM, XDH, XD5E, XDME, XL5E, XLME, XL5N, XL5H, XLH
Hardware requirement	-	Software requirement	-

2) Operands

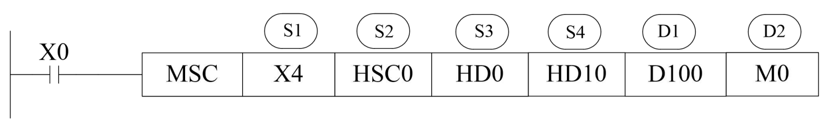
Operands	Function	Model
S1	Specify the software component address number of the command trigger input point	bit
S2	Specify high speed counter number	32 bits, BIN
S3	Specify the number of stations and workpieces, and the first address number of the register of the filtering time	16 bits, BIN
S4	Specify the first address number of the register for the reference value and the deviation value	32 bits, BIN
D1	Specify the number of the first address of the register storing the index value and the comparison value	16/32 bits,BIN
D2	Specify the software component address number of the output result	bit

3) Suitable soft component

Operands	Word soft elements											Bit soft elements						
	System								constant	Module		System						
	D	FD	TD	CD	DX	DY	DM	DS	K/H	ID	QD	X	Y	M	S	T	C	Dn.m
S1												●						
S2	Only HSC																	
S3	●																	
S4	●																	
D1	●																	
D2													●	●				

*Note: D includes D, HD. TD includes TD, HTD. CD includes CD, HCD, HSCD, HSD. DM includes DM, DHM. DS includes DS, DHS. M includes M, HM, SM. S includes S and HS. T includes T and HT. C includes C and HC.

Function and action



- S1: it is the command trigger input point, which can select the external interrupt input point or ordinary input point, trigger the command at the rising edge and falling edge, and grab the encoder value.
- S2: it is the number of the high-speed counter used together, which is used for encoder signal input. The high-speed counting mode is single-phase incremental mode.
- S3: three 16 bits registers (single word) are occupied continuously to set the number of stations, the number of workpieces, and the filtering time. It is recommended to use the power-off holding register.

  The specific register allocation is as follows:

  S3: set the number of stations, recorded as n, range: 1~32;

  S3+1: set the maximum number of workpieces that can be processed, recorded as m, range: 1~64;

  S3+2: set the filtering time, range: 0~32767, unit: ms. This parameter can be used to prevent errors caused by mechanical jitter. If the filtering time is set to 0, it means no filtering. If it is less than 0, it will be treated as 0. Assuming that the filtering time is set to t and the trigger input point is X4, the capture of the input signal adopts the following methods:

Rising edge: after X4 off state is maintained for at least t ms, the first detected rising edge is the trigger signal;

Falling edge: after the X4 on state is maintained for at least t ms, the first falling edge detected is the trigger signal.

- S4: 3n 32-bit registers (double words) are occupied continuously, which are used to set the reference value, workpiece entry deviation value and workpiece departure deviation value of each station. Each parameter occupies 2 registers continuously. It is recommended to use the power-off holding register. The specific register address allocation is as follows:

Name	Station 1	Station 2	……	Station n
Reference value(double word)	S4	S4+2	……	S4+(n-1)×2
Workpiece entry deviation value(double word)	S4+2n	S4+2n+2	……	S4+(2n-1)×2
Workpiece departure deviation value(double word)	S4+4n	S4+4n+2	……	S4+(4n-1)×2

◆ When the reference value of a station is set to 0, it means that the station does not operate.

◆ The workpiece entry deviation value and the workpiece departure deviation value are mainly used for position calibration. When the encoder value of the workpiece entering and leaving the corresponding station is found to be inconsistent with the setting during actual use, it can be calibrated by adjusting the workpiece entry deviation value and the workpiece departure deviation value. For example, the reference value of station 1 is set to 1000, which means that the workpiece enters station 1 after triggering the rising edge of X4 through 1000 high-speed count values. If in actual use, the workpiece enters station 1 with only 990 high-speed count values, the workpiece entry deviation value can be set to -10.

- D1: continuously occupy 2n 16 bits registers (single word), 2m × n 32-bit registers (double word) are used to store the workpiece forward index value, follow index value, entry comparison value and departure comparison value of each station. The specific register address allocation is as follows:

Name	Station 1	Station 2	……	Station n
Forward index value(word)	D1	D1+1	……	D1+(n-1)
Follow index value(word)	D1+n	D1+(n+1)	……	D1+(2n-1)
Workpiece 1 entry comparison value(double word)	D1+2n	D1+2n+2	……	D1+2n+2(n-1)
Workpiece 1 departure comparison value (double word)	D1+4n	D1+4n+2	……	D1+4n+2(n-1)
……	……	……	……	……
Workpiece m entry comparison value(double word)	D1+4mn-2n	D1+4mn-2n+2	……	D1+4mn-2
Workpiece mdeparture comparison value(double word)	D1+4mn	D1+4mn+2	……	D1+4mn+2(n-1)

Note: D1 occupies a large storage area, please confirm whether the register space is enough. If it is not enough, PLC will only store data in the effective area, and will not generate alarms and prompts.

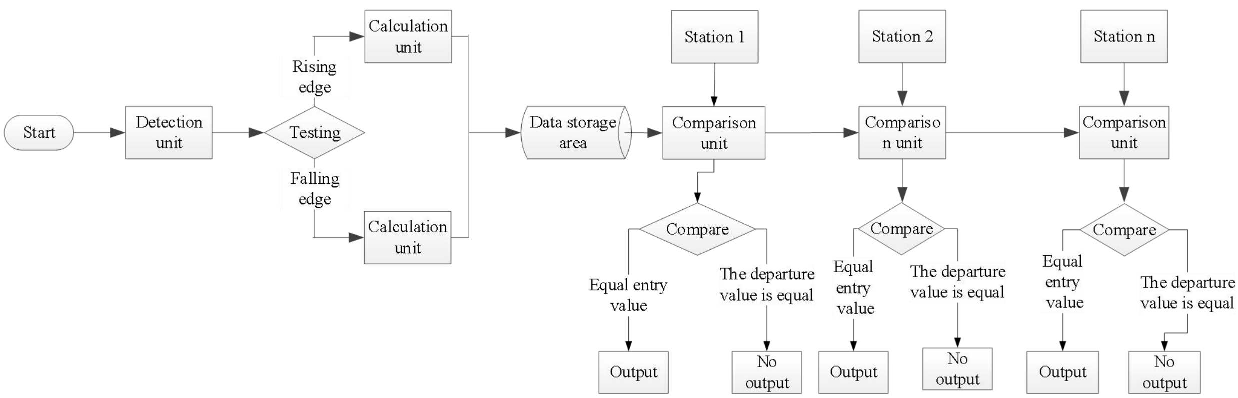
- ◆ When the entry comparison value and the exit comparison value of a station are both 0, it means that the comparison action of the station is not executed.
- ◆ The forward index value will automatically increase by 1 when triggering each rising and falling edge of the input signal (if the filtering time is > 0, wait for the filtering time and then increase by 1), and the method of cyclic accumulation is adopted. For example, when the maximum number of workpieces processed is m = 10, the forward index value will cycle by 0, 1, 2, 3... 19, 0, 1, 2, 3... 19 (the initial value is 0). Since the forward index value will increase by 1 at both rising and falling edges, the maximum forward index value is 2*m。

Note: the following index value will be judged before adding 1 to the forward index value. If the value after adding 1 is equal to the following index value, the forward index value will not be accumulated and the comparison value this time will be recorded.

- ◆ Follow the index value will automatically add 1 when the workpiece enters and leaves the station. Generally, after the workpiece has completed a station, the following index value of the corresponding station is even.
- ◆ The entry comparison value is automatically calculated and stored in the D1 data area when the corresponding workpiece triggers the rising edge of the input signal. The entry comparison value of the station is generally:

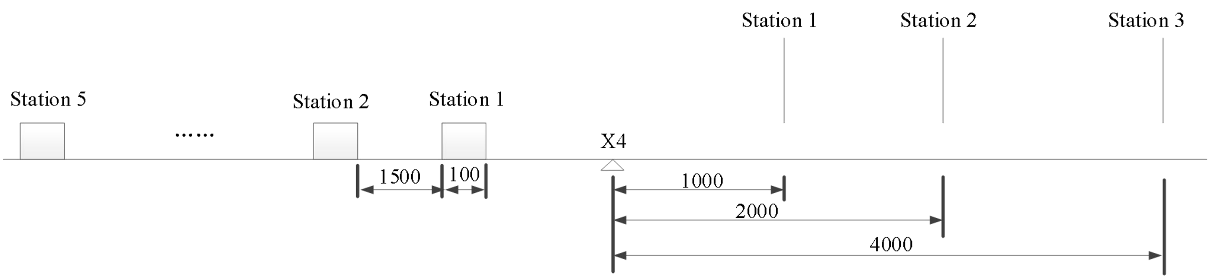
- ◆ The comparison value of workpiece m entering station n = the grab count value of workpiece m (at the rising edge) + the reference value of station n + the workpiece entering deviation value of station n.
- ◆ The departure comparison value is automatically calculated and stored in the D1 data area when the falling edge of the input signal is triggered by the corresponding workpiece. The departure comparison value of the station is generally:
- ◆ The comparative value of workpiece m leaving station n = the grab count value of workpiece m (at the falling edge) + the reference value of station n + the workpiece leaving deviation value of station n.

- D2: continuously occupy n coils (corresponding to the number of n stations), and only Y and M coil outputs can be specified to judge whether the corresponding workpiece enters and leaves the station. When the command is executed, each station will judge whether the corresponding workpiece enters and leaves the station according to the set comparison value according to the follow index value. When the real-time count value of the corresponding workpiece is ≥ the enter comparison value, the corresponding output point is set to on, and the follow index value is automatically increased by 1; When the real-time count value of the corresponding workpiece ≥ leaves the comparison value, the corresponding output point is set to off, and the following index value automatically increases by 1, but it will not exceed the forward index value.
- There is no limit on the number of times MSC instructions are used, but if the same high counter needs to be used in the program, each instruction must be placed in a different process, and only one instruction can be executed at a time.
- Before the instruction is executed, please confirm whether the high-speed counter used overflows (it can be judged by the high-speed count overflow flag bit sm130, etc.) and make corresponding treatment.
- When the precondition of MSC is disconnected and reconnected, the values in D1 and D2 storage areas will be cleared to 0 and set to off.
- The basic process of instruction processing is as follows:

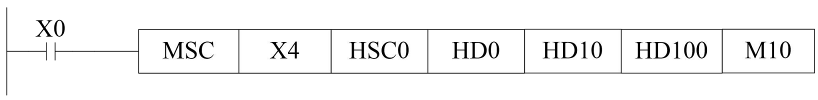


Note:
(1) When the given parameter for the number of workers or workpieces in the MSC instruction is 0 or exceeds the limit, it will cause the PLC to report an error (the instruction does not report an error). At this time, the instruction conduction condition needs to be disconnected, and the PLC needs to be stopped to ensure that it is in a safe stop state. Then restart the PLC to ensure that the instruction parameters are correct before the instruction can run normally.
(2) When the workpiece is fed too fast, it will cause an error: MSC multi station control workpiece feeding too fast. For example, a production line has only three workstations set, but the first workpiece has not been processed yet. The second and third workpieces are fed into the line, and this error occurs when the fourth workpiece is detected.
(3) When MSC multi station control workpiece feeding is too fast, it may result in only generating entry comparison values without generating exit comparison values, leading to the situation where the output coil of the workpiece remains ON after leaving the station.
(4) Modifying the deviation value of the corresponding workstation during operation takes effect in real-time.

Procedure example
For example, the existing five workpieces need to be processed through three stations. The trigger input signal is X4, the encoder signal input point is X0 (the corresponding high-speed counter is HSC0), the width of each workpiece is 100, the distance between workpieces is 1500, the distance between workpieces is 1000, the distance between workpieces is X4, 2000, and the distance between workpieces is 4000.



The procedure is as follows:



Soft component address	Function description
X4	Trigger input point
HSC0	High speed counting input point, receiving encoder signal
HD0	Number of stations, HD0=3 in the example
HD1	The maximum number of workpieces that can be processed , HD1=4 in the example
HD2	Filter time， HD2=300ms in the example
HD10(double word)	The reference value of station 1.HD10=1000 here in the example
HD12(double word)	The reference value of station 2.HD12=2000 here in the

	example
HD14(double word)	The reference value of station 3.HD14=4000 here in the example
HD16(double word)	The workpiece of station 1 enters the deviation value and is set to 0
HD18(double word)	The workpiece of station 2 enters the deviation value and is set to 0
HD20(double word)	The workpiece of station 3 enters the deviation value and is set to 0
HD22(double word)	The workpiece departure deviation value of station 1 is set to 0
HD24(double word)	The workpiece departure deviation value of station 2 is set to 0
HD26(double word)	The workpiece departure deviation value of station 3 is set to 0

Output result address assignment:

Name	Station 1	Station 2	Station 3
Forward index value(single word)	HD100	HD101	HD102
Follow index value(single word)	HD103	HD104	HD105
Workpiece 1 enters the comparison value(double word)	HD106	HD108	HD110
Workpiece 1 leaves the comparison value(double word)	HD112	HD114	HD116
Workpiece 2 entry comparison value(double word)	HD118	HD120	HD122
Workpiece 2 departure comparison value(double word)	HD124	HD126	HD128
Workpiece 3 entry comparison value(double word)	HD130	HD132	HD134
Workpiece 3 departure comparison value(double word)	HD136	HD138	HD140
Workpiece 4 entry comparison value(double word)	HD142	HD144	HD146
Workpiece 4 departure comparison value(double word)	HD148	HD150	HD152
Output flag	M10	M11	M12

Program execution results:

Assuming that the high-speed count value when workpiece 1 triggers the rising edge of X4 is 1000, the forward index value, follow index value, workpiece entry comparison value and workpiece departure comparison value of each station are shown in the following table:

Parameter		Station 1	Station 2	Station 3
Workpiece 1	Forward index value	X4 rising edge:1	X4 rising edge:1	X4 rising edge:1
		X4 falling edge: 2	X4 falling edge: 2	X4 falling edge: 2
	Follow index value	M10 rising edge: 1	M11 rising edge: 1	M12 rising edge: 1
		M10 falling edge: 2	M11 falling edge: 2	M12 falling edge: 2
	Entry comparison value	HD106=2000	HD108=3000	HD110=5000
	Departure comparison value	HD112=2100	HD114=3100	HD116=5100
Workpiece 2	Forward index value	X4 rising edge:3	X4 rising edge:3	X4 rising edge:3
		X4 falling edge:4	X4 falling edge:4	X4 falling edge:4
	Follow index value	M10 rising edge: 3	M11 rising edge: 3	M12 rising edge: 3
		M10 falling edge: 4	M11 falling edge: 4	M12 falling edge: 4
	Entry comparison value	HD118=3600	HD120=4600	HD122=6600
	Departure comparison value	HD124=3700	HD126=4700	HD128=6700
Workpiece 3	Forward index value	X4 rising edge:5	X4 rising edge:5	X4 rising edge:5
		X4 falling edge:6	X4 falling edge:6	X4 falling edge:6
	Follow index value	M10 rising edge:5	M11 rising edge:5	M12 rising edge:5
		M10 falling edge: 6	M11 falling edge: 6	M12 falling edge: 6
	Entry comparison value	HD130=5200	HD132=6200	HD134=8200
	Departure comparison value	HD136=5300	HD138=6300	HD140=8300
Workpiece 4	Forward index value	X4 rising edge:7	X4 rising edge:7	X4 rising edge:7
		X4 falling edge:0	X4 falling edge:0	X4 falling edge:0
	Follow index value	M10 rising edge:7	M11 rising edge:7	M12 rising edge:7
		M10 falling edge: 0	M11 falling edge: 0	M12 falling edge: 0
	Entry comparison value	HD142=6800	HD144=7800	HD146=9800
	Departure comparison value	HD148=6900	HD150=7900	HD152=9900
Workpiece 5	Forward index value	X4 rising edge:1	X4 rising edge:1	X4 rising edge:1
		X4 falling edge:2	X4 falling edge:2	X4 falling edge:2
	Follow	M10 rising edge:1	M11 rising	M12 rising edge:1

	index value		edge:1	
		M10 falling edge: 2	M11 falling edge: 2	M12 falling edge: 2
	Entry comparison value	HD106=8400	HD108=9400	HD110=11400
	Departure comparison value	HD112=8500	HD114=9500	HD116=11500

Note: Once X0 is disconnected and reconnected, all the data in the above table will be cleared to 0.

11 Common Questions and Answers

This chapter mainly introduces XD/XL series PLC common questions and answers.

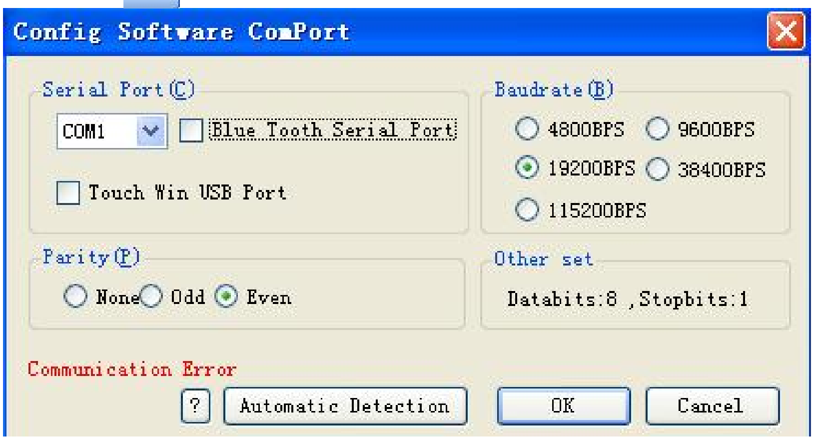
Q1: How to connect PLC with PC?

A1: XD series PLC with firmware version V3.2 or later supports USB port, COM1 port (RS232), and COM2 port (RS485) to connect. Ethernet PLC also supports Ethernet port (RJ45) to connect.

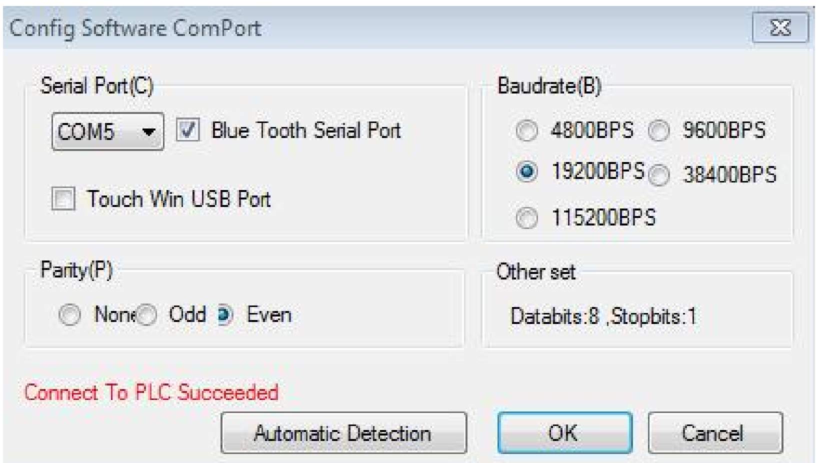
1. Connect via USB port and PC (refer to Section 6-1-1)

2. Connect via COM1 port (RS232) and PC

If your PC is desktop computer, you can use our company special DVP or XVP cables to connect PC and PLC (Usually PORT1) as general commercial desktop computer has 9 needle serial port. After connecting DVP correctly, power on PLC, click 'Config Software ComPort'  the following window will jump out:



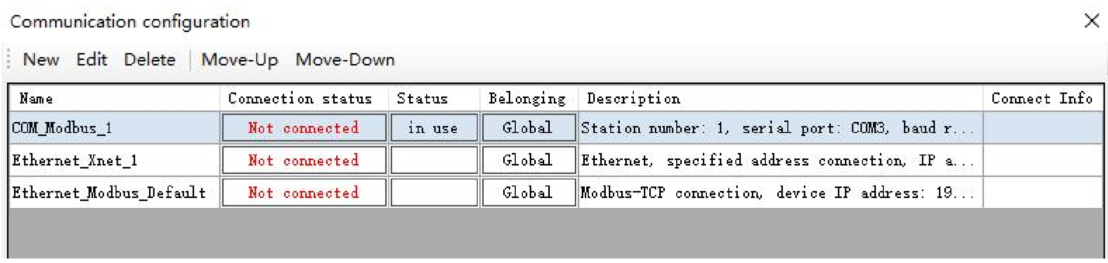
Choose correct communication serial port according to your PC actual serial port.; baud rate selects 19200BPS, parity check selects even parity, 8 data bits, 1 stop bit; you can also click 'check' button directly in the window, and communication parameters will be selected by PLC itself. 'Connect to PLC succeed' will be displayed on the left bottom of window as below:



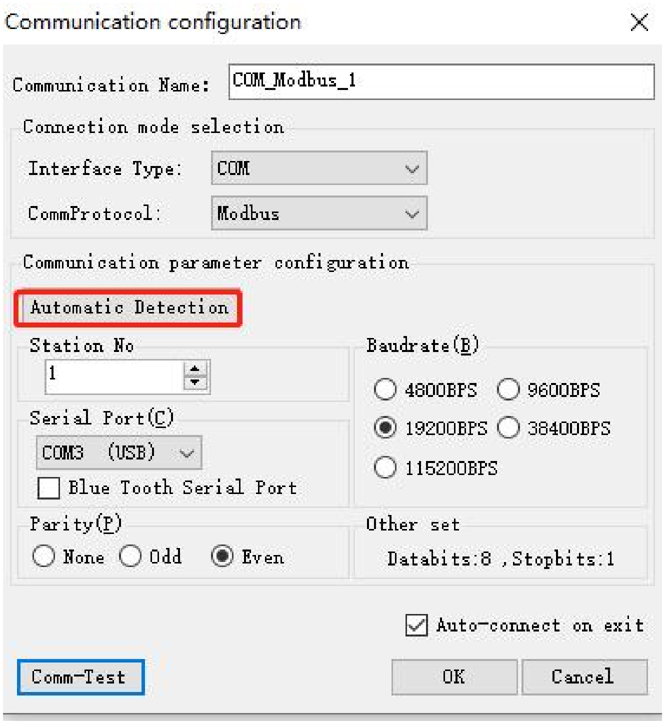
Then it means that PLC has been connected to PC successfully!
Usage method of notebook PC with 9-pin serial port is the same with desktop PC's.
If the notebook does not have 9-pin serial port, users can use USB converter to realize connection between PLC and notebook USB port. Make sure to install USB converter drive software (Xinje special USB converter module COM-USB is recommended， USB converter drive software can be downloaded on Xinje official website)!

3. Connect via COM2 port (RS485) and PC

If the computer is equipped with 9-pin serial port, it can connect the PC with PLC (usually com2 port) through RS485 serial conversion module and XVP cable. If the computer has only USB interface, it can be connected through USB to RS485 cable.
When the wiring is correctly connected, power on the PLC,click 'Config Software ComPort' , and the following window will pop up:



Choose correct communication serial port according to your PC actual serial port.; baud rate selects 19200BPS, parity check selects even parity, 8 data bits, 1 stop bit; you can also click 'check' button directly in the window, and communication parameters will be selected by PLC itself. 'Connect to PLC succeed' will be displayed on the left bottom of window as below.



4.Connect via RJ45 port

1)Computer configuration

After the network cable is plugged in, open "control panel" → "network and Internet" → "network connection".

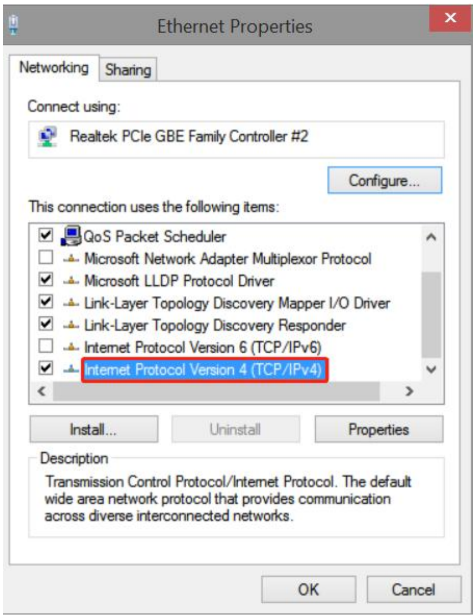
Find the Ethernet that has been successfully connected. Right click the Ethernet and click properties. The Ethernet properties interface pops up. Then follow the steps below:

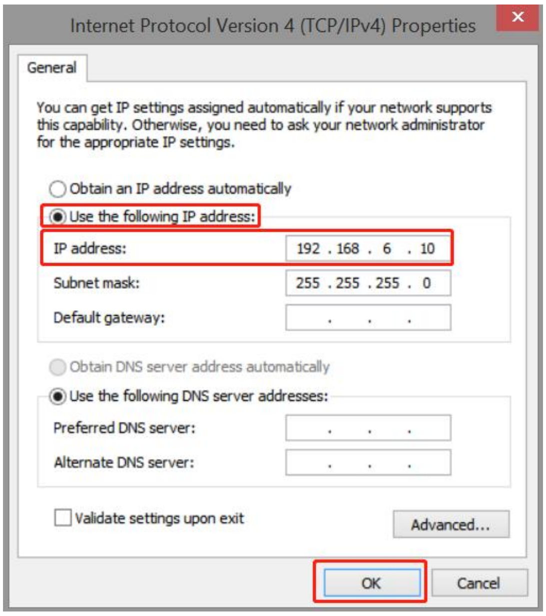
(1) Double click "Internet Protocol Version 4 (TCP/IPV4)".

(2) Select "use the following IP address".

(3) Set IP address: 192.168.6.xxx, "xxx" can be set arbitrarily (except 6).

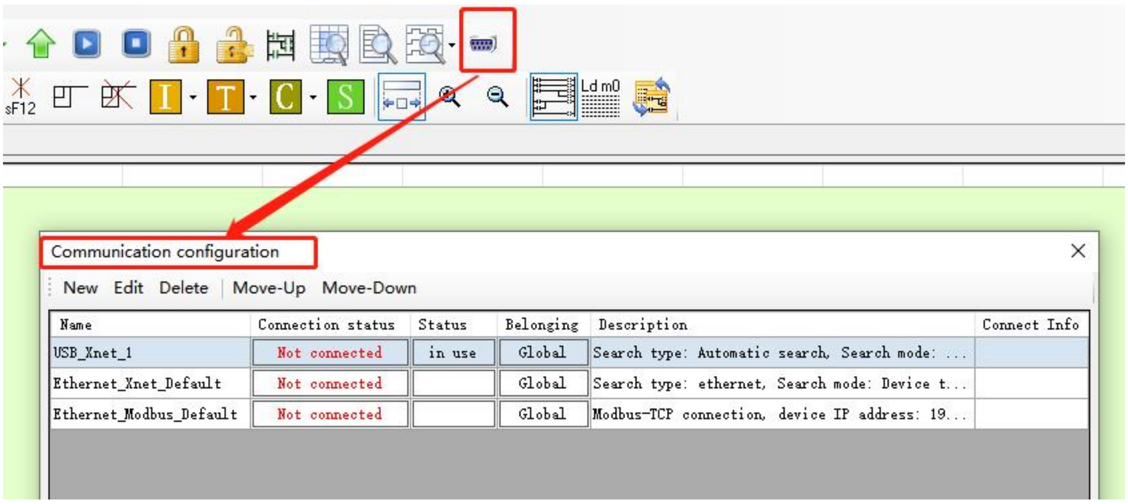
Note: The last digit of the computer address and the IP address of the PLC device cannot be set repeatedly.





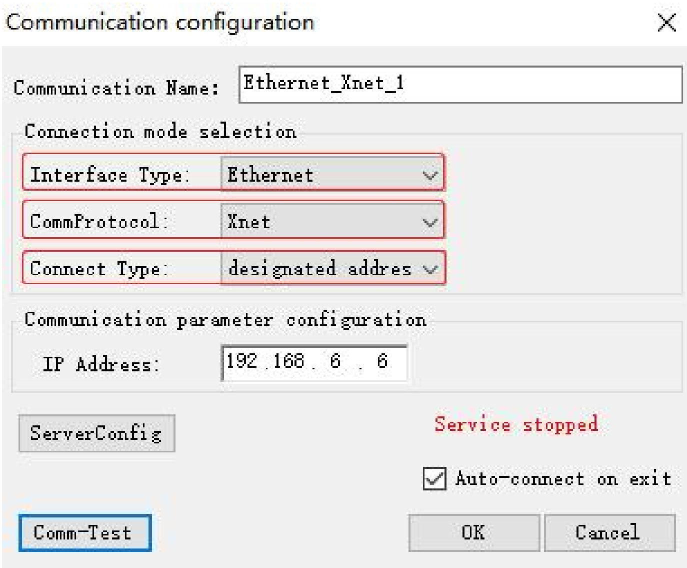
2)PLC configuration

After checking the wiring and Ethernet configuration,open XDPPRO programming tool→click communication configuration→double click Ethernet-Xnet.



Configure according to the following figure:

Choose Xnet protocol, the IP address is your PLC IP address. Click 【Comm-Test】, ‘Connect to PLC succeed’ will be displayed



Click OK after configuration and select " in use" for corresponding status.

Q2：PC cannot connect PLC via RS232 port, it shows offline status?

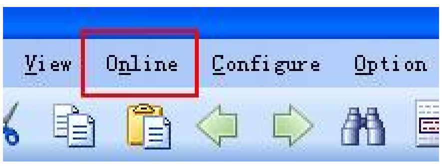
A2：

Several possible reasons：

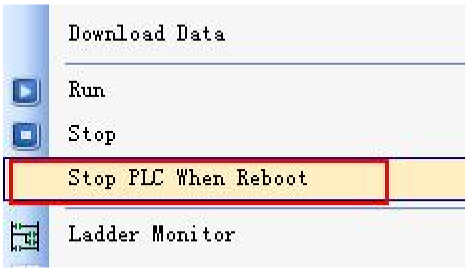
Users may changed the communication parameters of PORT1 in PLC (Do not change Port1 communication parameters, or it may lead to connection between PC and PLC failure!)
USB converter driver software was installed incorrectly or USB converter cable is not good
PORT1 communication of PLC is damaged.
The download communication cable brand is not Xinje XVP cable.

Solutions:

At first, use Xinje XVP cable to connect PC and PLC;
After confirming the connection cable is the Xinje special XVP cable and USB convertor has been used, you can use it to try to connect desktop PC with 9-needle serial port to PLC. If the desktop PC can be connected correctly, please change the USB converter cable with higher performance or install the USB converter serial driver software again.
If PLC can not connect with desktop computer correctly either, you can use ‘stop PLC when reboot’ function to stop PLC and recover the PLC to factory setting, operating method is as follow:
Power on PLC and connect PLC by DVP cables, then click ‘online’ button on PLC editing software menu;

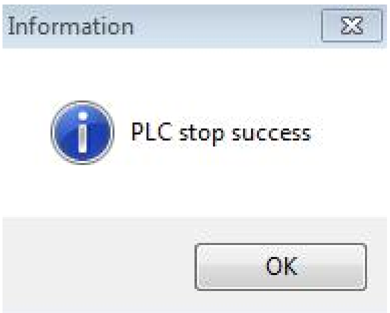


Click ‘Stop when PLC reboot’ from the drop-down menu;

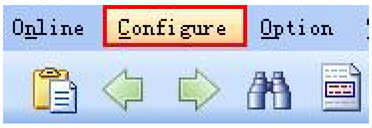


Following window will jump out;

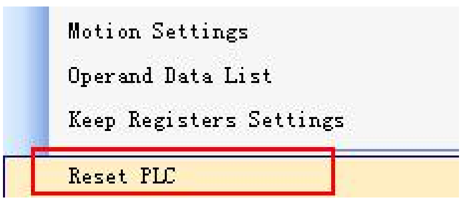
By this time, cut off PLC power for 2-3s and power on again, then a ‘PLC has been stopped successfully’ window will normally jump out; if the window do not jump out after power on, try again a few times until the information window of successful stop jump out.



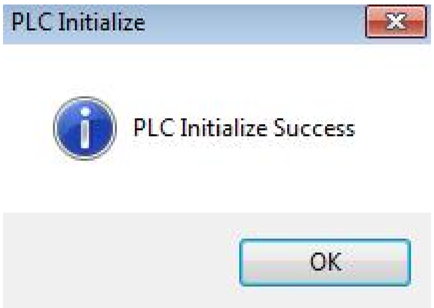
Then click ‘configure’ button ;



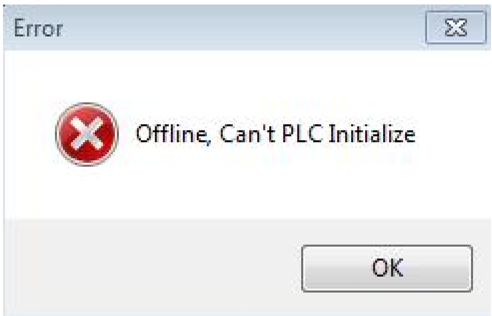
Click ‘Reset PLC’ in the drop-down menu;



By this time, ‘Reset PLC’ information window will jump out and it means that all steps of ‘Stop when PLC reboot’ have been finished.



If initialize PLC unsuccessfully after you trying a few times or the following window jumps out after clicking 'Reset PLC':



In both cases, use PLC system update tool to update PLC system, and PLC and PC will be connected successfully if system is updated (For more steps about system update, please refer to Q3 related content).

If update of the desktop computer with 9-pin serial port fails, it is very likely that PLC communication port is damaged, and please contact manufacturer or agent.

Q3：XD/XL series PLC system upgrade

A3：

When does PLC need update usually?

PLC software is in a continuous upgrade stage; if software and hardware version do not match, PLC will not support those upgraded function. About which PLC version the instruction support, please refer to instruction summary in this manual or appendix 2 'special function version requirement';

When users change the communication parameters, PLC and PC can not connect.

When users use 'program confidential download' function, however, forget the password (Note：PLC program will disappear after system update！).

How to update XD/XL series PLC？

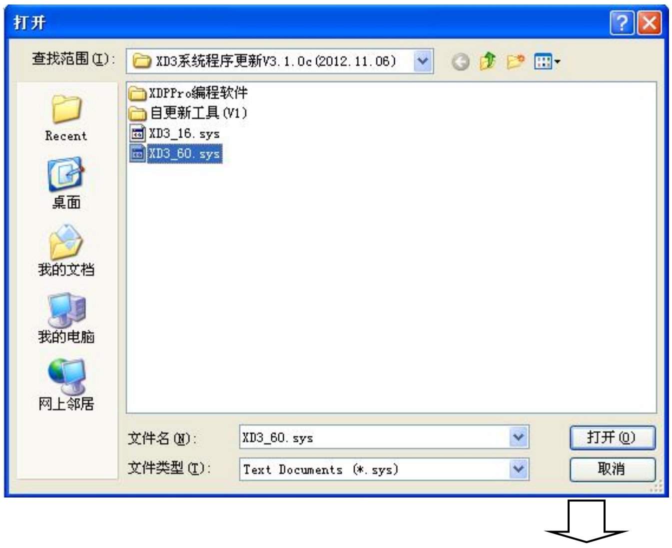
PLC update tool:

'XD series PLC download program tool' and 'system file' (*.sys file)

Close all the programs which may occupy the serial port

Cut off the power of PLC, open the XD series update tool (if user use this tool at the first time, please open the enrollment first)

Click "Open File", choose the PLC model for updating.**(Note: XD3_16.sys fit for PLC model XD3-16, XD3_60.sys fit for PLC model XD3-32 and XD3-60):**



Set the parameters:

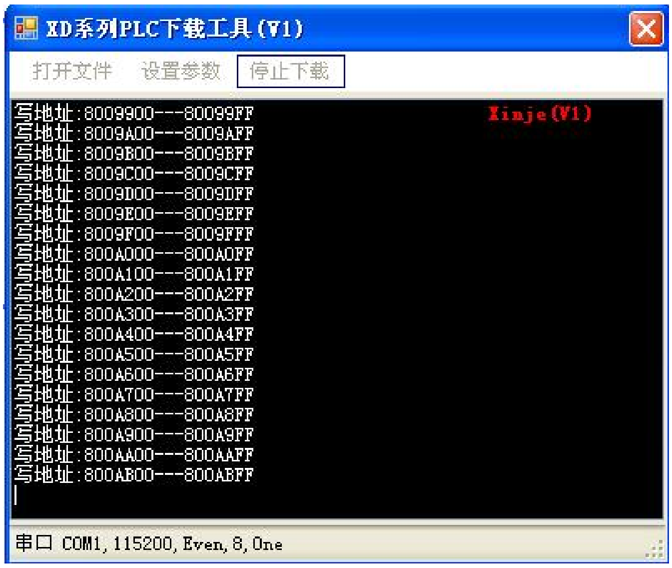
Click “set parameter”, it will show the parameter window:

Note: set the com port, the baud rate is default setting, no need to change.
Click “download”, the window will show below words:

Power on the PLC, the update tool will show below words:

Cut off the power of PLC, connect the short jumper, then power on the PLC again.

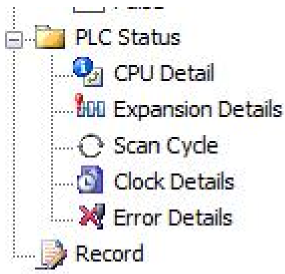
PLC start to update, the updating will take few minutes.

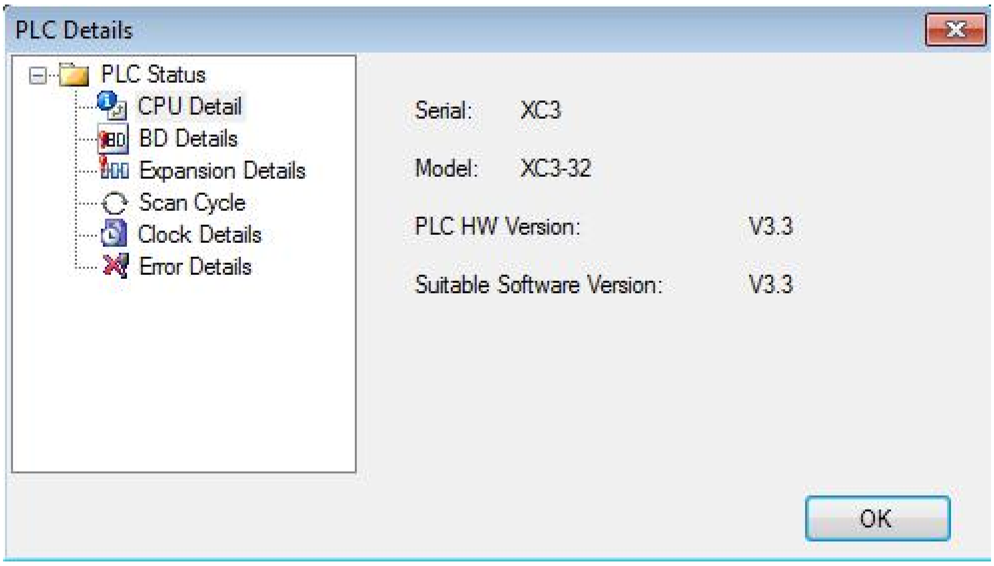


After finishing the update, cut off the PLC power, take off the short jumper, then power on the PLC again.

PLC hardware version

The PLC hardware version can be seen in "CPU detail" on the left window in XDPpro software (PLC online status)





Short jumper

XD, XL series PLC no need to short the jumper when updating.

Note:

Do not cut the power of PLC when it is updating. If it show the error "send data failed, ID not match…) please contact us for help.

The PLC program will be deleted after updating.

Q4: The bit soft component function.

A4:

Continuous 16 coils consist of a word, E.g: DM0 a word consist of 16 coils (bits) M0~M15 is as below:

DM0:

M15	M14	M13	M12	M11	M10	M9	M8	M7	M6	M5	M4	M3	M2	M1	M0

We can use bit in the register directly.

Example 1:

When M100 is from OFF to ON, M0 M1 are ON, M2—M15 are OFF

The other mode is bit operation of fixed register. E.g: D0.0 is the first bit of16 bits in register D0. Similarly, D0.1 is the second bit and so on, as shown below:

D0:

D0.15	D0.14	D0.13	D0.12	D0.11	D0.10	D0.9	D0.8	D0.7	D0.6	D0.5	D0.4	D0.3	D0.2	D0.1	D0.0

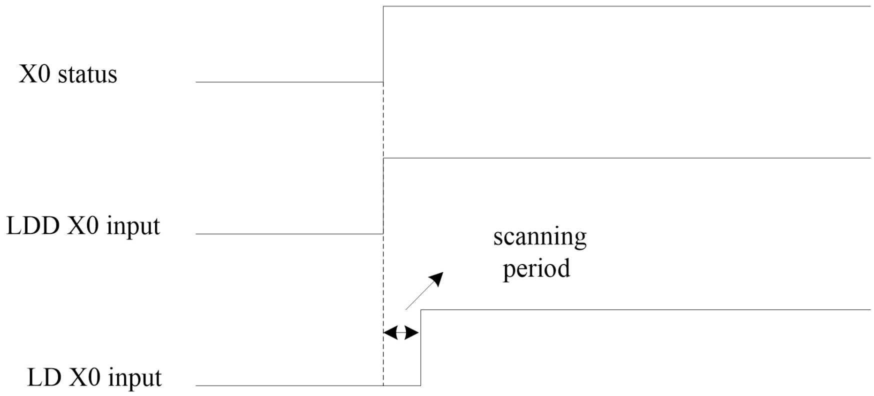
Similarly, we can use bit in register D0.

Q5: What's the use of execution instruction LDD/OUTD etc?

A5:

When PLC executes program, state of input point state will map to image register. From then on, PLC will refresh input state at the beginning of every scan cycle; if we use LDD instruction, then the state of input point will not need map to image register; the same with output point (OUTD).

LDD/OUTD instruction usually apply to the occasion that I/O need refresh immediately, which makes the state of input and output avoid the influence of the scan cycle.



Input point X0 sequence chart of LDD and LD

Q6: Why the output LED keeps flashing when using ALT instruction?

A6:

For ALT and many calculation instructions, these instructions will execute every scanning period when the condition is fulfilled (for example, the condition is normal ON coil). We recommend that the condition is rising edge or falling edge.

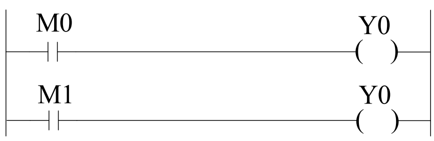
Q7: Why the M and Y cannot output sometime?

A7:

Output mainly has two ways: 1. OUT instruction; 2. SET instruction. The coil will keep outputting if there is no RST instruction.

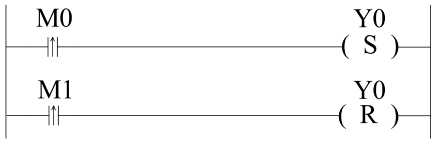
Usually in the program, one coil M or Y should use the same output way. Otherwise, the coil cannot output.

For example:



M0 is ON,M1 is OFF, Y0 cannot output
M0 is OFF, M0 is ON, Y0 will output
Reason: two different coils drive the same output coil

Y0 will be ON for one scanning period



M0 is ON, Y will keep outputting
M1 is ON, Y0 is OFF

Q8: Check and change the button battery in the PCB of PLC

A8:

The rated voltage of button battery is 3V. The voltage can be measured by multimeter. If the value of power-loss retentive register is very large, it means the battery is low. Please change the button battery.Users can use SM5 and SD5 to detect the power of button batteries in order to facilitate timely replacement of batteries. See Appendix 1 and Appendix 2 for details.

Q9: Communicate with SCADA software

A9:

If there is no choice for XD/XL series PLC in SCADA software, please choose Modbus-RTU protocol and communicate through RS485 port. Please refer to XD/XL series PLC instruction manual chapter 6.

Q10: MODBUS Communication

A10:

First of all, please ensure that the A and B terminals on the PLC are correctly connected with the RS485 communication terminals of other devices. To modify the parameters of the PORT 2 of the PLC, the following methods are adopted:
Method 1: Configuration by configuration parameter instruction
For specific instructions, please refer to Chapter 6, Communication Functions of this manual. The communication parameter settings of different devices are generally different, so it is important to choose the correct frequency setting mode of communication devices, make clear the corresponding MODBUS communication address and function code, and some communication devices need a given operation signal before displaying the setting frequency.
Method 2: Configuration through control panel (refer to Chapter 6 Communication Function of this manual for specific configuration method).

Q11: The LED light of XD/XL series PLC (PWR/RUN/ERR)

A11:

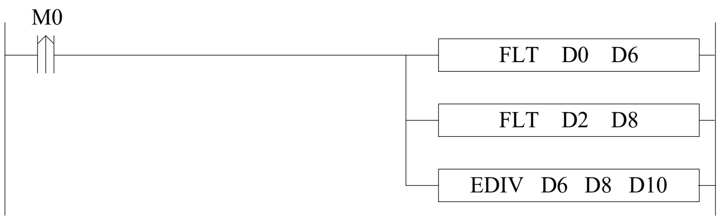
LED light	Problem	Solution
PWR shining, other LED off.	1. I/O PCB has short circuit 2. load is too large for 24V 3. not click RUN for program	Check I/O terminal, if there is short circuit. If the load is too large for 24V power supply. Make sure the program is

		running inside PLC. Contact us for help.
Three LED all OFF	1. PLC input power supply has short circuit 2. PLC power PCB damaged	Check the input power supply of PLC. Contact us for help.
PWR and ERR light	1. PLC input voltage is not stable 2. there is dead loop in the program 3. PLC system has problem	Check the power supply voltage, check if there is dead loop in the program. Update the hardware of PLC. Contact us for help.

Q12: The result is not correct when doing floating operation

A12:

Please transform the integer to floating number. For example: EDIV D0 D2 D10. If the value of D0 and D2 is integer, the result will has error (D10). Please use below instruction to transform the integer to floating number.

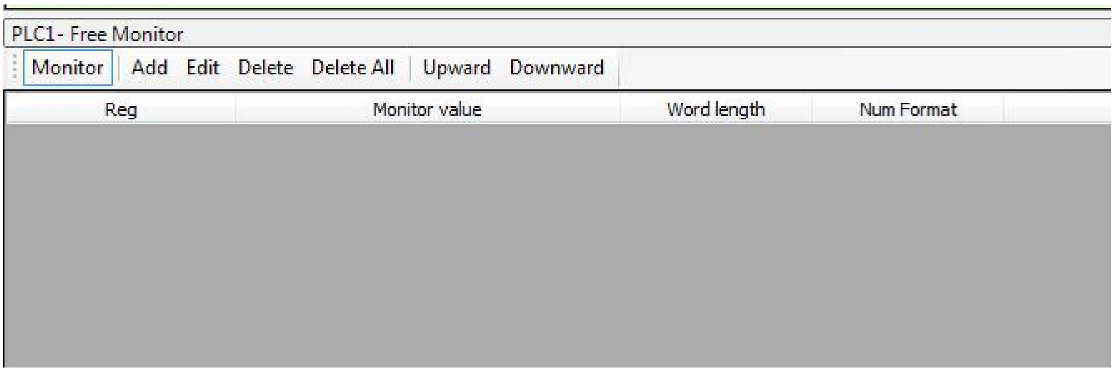


Q13: Why the floating numbers become messy code in online ladder monitor window?

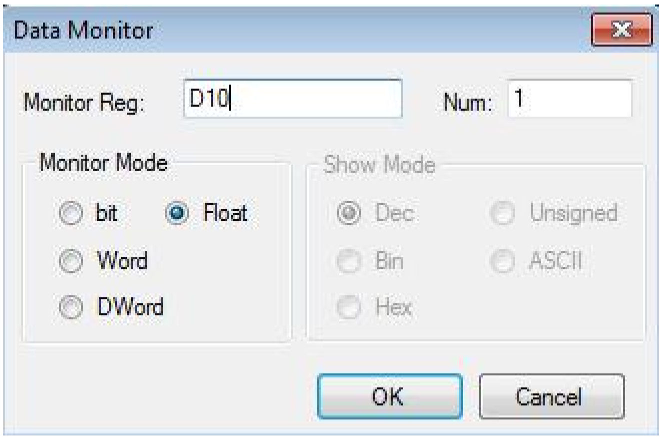
A13:

As the floating number cannot be displayed in online ladder monitoring, please monitor the floating number in free monitor function.

Open XDPpro software, click online/free monitor. The following window will pop up:



Click “add” in the window, the following window will pop up. Set the monitor mode to “float”. Monitor register set to D10. Then click ok.



Q14: Why data errors after using DMUL instructions?

A14:

DMUL operation instruction is 32 bits*32 bits=64 bits operation, the result occupies 4 words, such as: EMUL D0 D2 D10, two multiplier both are 32bit (D1,D0) and (D3, D2), the result is 64 bits (D13, D12, D11, D10), so D10~D13 will be occupied. If these data registers are used latter, operation will error.

Q15: Why the output point action errors after PLC running for a while?

A15:

It's possible that output terminal is loose, please check.

Q16: Why expansion module does not work while power indicator is ON?

A16:

It is likely the connection of module strips and PLC pins or CPU is not good. Compare the CPU and expansion in cross contrast way to find the problems.

Q17: Why the signal input but cannot see the high speed counter working?

A17:

If high-speed counting is to be carried out, in addition to connecting high-speed pulse to the input of high-speed counting of PLC, the corresponding high-speed counting program should be written with functional instructions. For details, please refer to the relevant content of Chapter 5 of this manual.

Q18: C language advantages compared to ladder chart?

A18:

(1) XD/XL series PLC supports almost all C language functions. When it comes to complex mathematical operations, the advantage of C language is more obvious.
(2) Enhance the confidentiality of the program (when using file-advanced storage mode, C language can not upload);
(3) C language function block can be called in many places and different files, which greatly improves the efficiency of programmers.

Q19: What's PLC output terminal A, B?

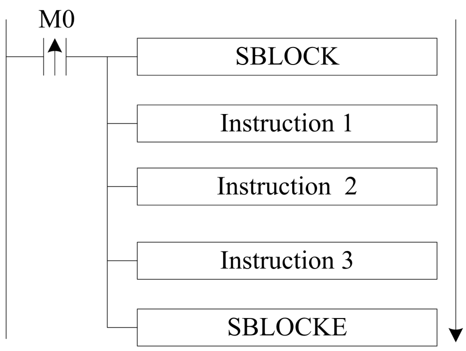
A19:

PLC output terminal A, B are RS485 terminals of PORT2 on PLC.

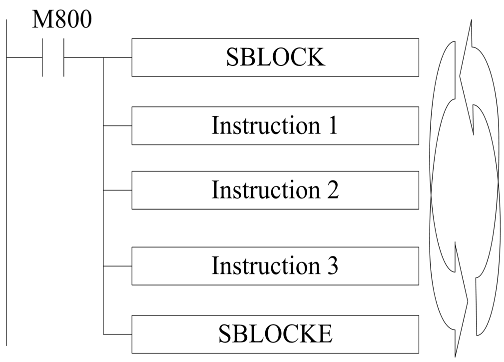
Q20: What's the difference of sequence function BLOCK trigger condition: rising edge triggered and normally closed conduction?

A20:

Rising edge triggered: when the condition is triggered, block executes in order from top to bottom; Normally closed conduction: when the condition is triggered, Block will execute in order from top to bottom, return to the top and execute again until the normally closed conduction breaks off. The cycle stops when the last one finished.



From up to down, run the instruction one by one



from up to down, cyclic run the instruction

Q21: What are the download modes of XD/XL series PLC and what are their characteristics?

A21:

XD/XL series PLC has three download modes, which are:

Common download mode

In this mode, you can easily download the program from the computer to the PLC or upload the program from the PLC to the computer. It will be very convenient to use this mode when debugging the equipment.

Password Download Mode

You can set a password for the PLC. When you upload the program from the PLC to the computer, you need to enter the correct password. In the advanced password option, you can also check the function of "download the program needs to be decrypted first" (Note: This operation is dangerous, if you forget the password, your PLC will be locked!). This download mode is suitable for users when they need to keep the device program secret and they can call out the device program at any time.

Secret download mode

In this mode, the program on the computer can be downloaded to the PLC, no matter what way the user can upload the program in the PLC to the computer; at the same time, the user program can be downloaded confidentially, which can occupy less internal resources of the PLC, greatly increase the program capacity of the PLC, and can have a faster download speed; after using this download mode, the program will be completely unable to recover.

Q22: What kinds of confidentiality methods do XD/XL series PLCs have?

A22:

Xinje PLC has three methods of confidentiality: (1) importing and exporting downloaded files; (2) secret downloading; (3) password downloading.

Import and export download files: After saving the PLC program in this way, users can download and use the program, but they can not view and edit the program.

Secret download: After secret downloading to PLC, the program and data in PLC will not be uploaded, indicating that "the program does not exist".

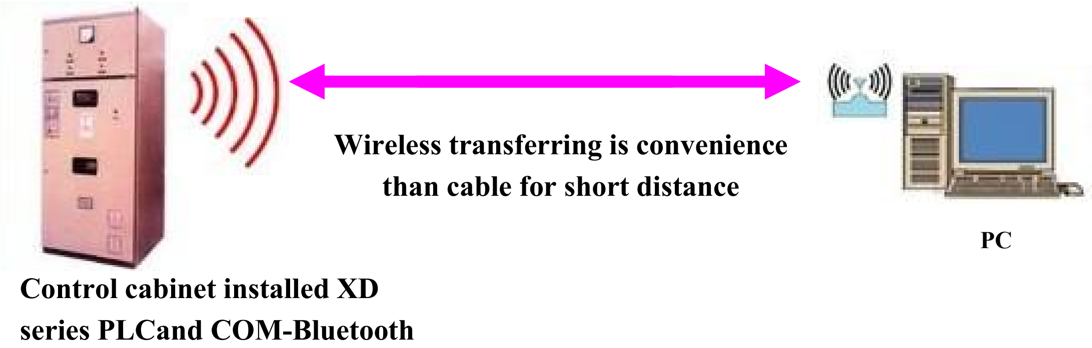
Password download: If you download the program that has set the password to the PLC, you need to input the correct password when uploading the PLC program; if you check "download program needs to be decrypted first", you also need to input the correct password when downloading the new program to the PLC. Under this mode, you can not modify the clock information of the PLC, and the confidentiality is stronger.

Q23: What’s the advantage that XD series PLC replaces DVP download cable with Bluetooth?

A23:

XD series PLC Bluetooth function can perform PLC program download and upload, monitor and Twin configuration software online simulation. The Bluetooth can replace the cable to transfer the data.

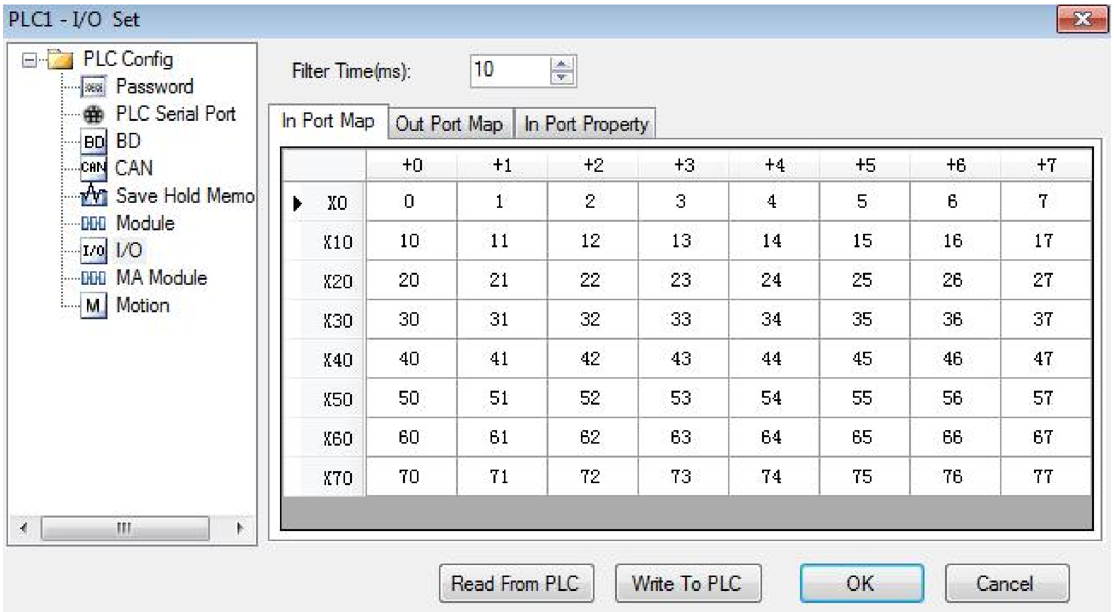
Note: COM-Bluetooth only fit for XINJE PLC.



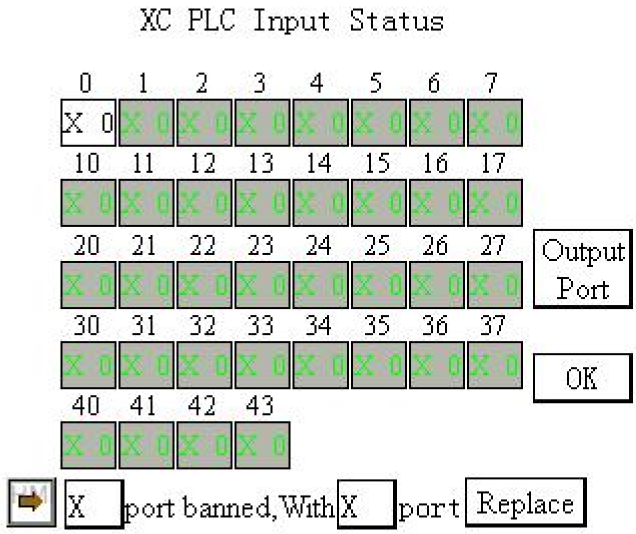
Q24: PLC I/O terminalexchanging

A24:

Sometime the PLC I/O terminals are broken. User don't have to change the program, PLC I/O terminal exchanging function can solve the problem. User can exchange the terminal through XINJE Touchwin HMI. Open Touchwin software, jump to screen no. 60004 (X terminals) or screen no. 60005 (Y terminals) to set the I/O exchanging.



	+0	+1	+2	+3	+4	+5	+6	+7
X0	0	1	2	3	4	5	6	7
X10	10	11	12	13	14	15	16	17
X20	20	21	22	23	24	25	26	27
X30	30	31	32	33	34	35	36	37
X40	40	41	42	43	44	45	46	47
X50	50	51	52	53	54	55	56	57
X60	60	61	62	63	64	65	66	67
X70	70	71	72	73	74	75	76	77



Touchwin HMI I/O terminal exchanging screen

Q25: What's the function of XD/XL series PLC indirect addressing?

A25:

Adding offset suffix after coils and data registers (Such as X3[D100], M10[D100], D0[D100]) can realize indirect addressing function; such as D100=9, X3[D100] represents X14， M10[D100] represents M19, D0[D100] represents D9; It usually applies to large number of bit and register operation and storage.

Q26: How does XD/XL series PLC connect to the network?

A26:

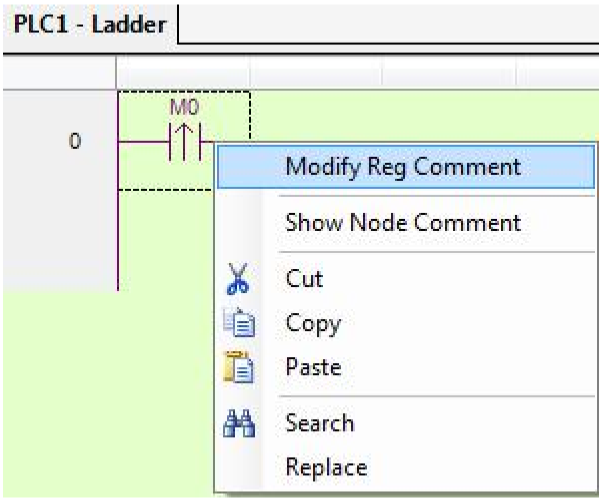
XD/XL series PLC can connect to network by Xinje T-BOX, G-BOX, W-BOX, S-BOX, A-BOX expansion modules or expansion BD boards which have their own communication characteristics. Details please refer to the user manual of communication module or BD board.

Q27: How to add soft element andline note in XDppro software?

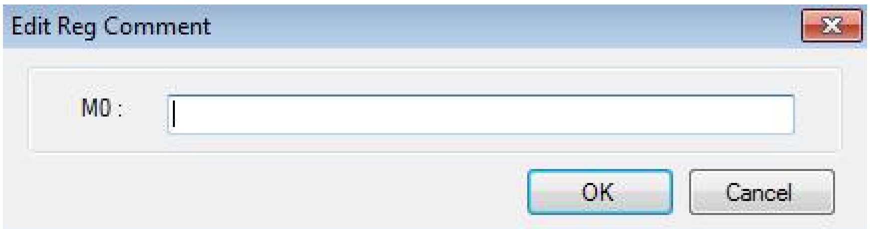
A27:

Soft element note

Open XDPpro software, and move the mouse to the corresponding soft element and right click the mouse, then menu will pop out:

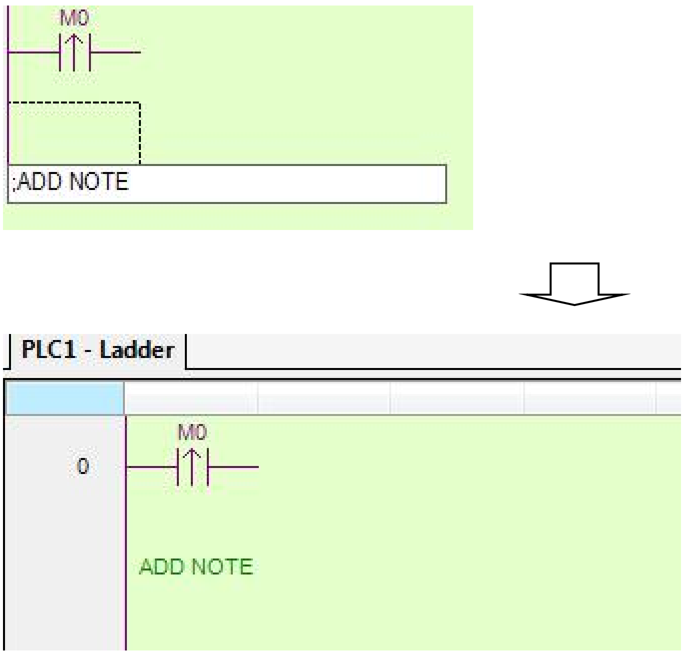


Click “Modify reg comment” to add element notes in below window:



Line note

Line note starts from “;”. Double click the line, then input semicolonand the contents.



Q28: Do not have clock function?Why is the clock inaccurate?

A28:

XD/XL series PLC clock function is optional, and if you want to buy the PLC with clock function, please confirm when purchasing. Otherwise, the default PLC when it leaves factory does not have clock function.

If you use a PLC with clock function, check whether the value in register SD13-SD19 is decimal. If not, you need to convert it into decimal through BIN or TRD instructions.

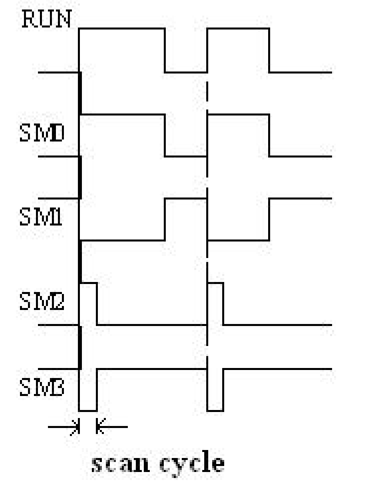
There are some errors in the clock of XD/XL series PLC. The error is about ±5 minutes per month. Please calibrate it by HMI or directly in the PLC program.

Appendix Special soft components

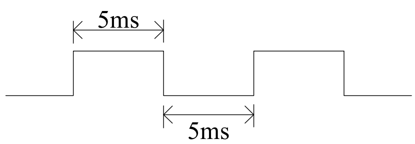
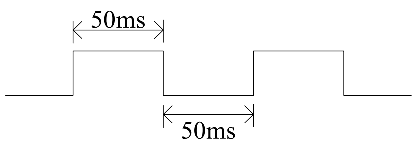
Appendix mainly introduces the functions of XD/XL series PLC special soft element, data register, FlashROM and the address distribution of expansions for users to search.

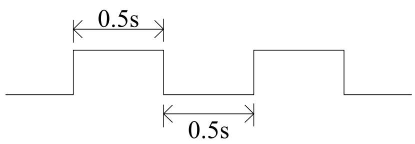
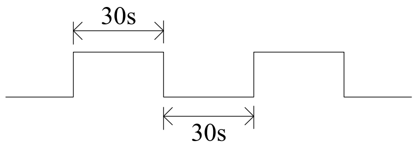
Appendix 1 Special Auxiliary Relay

Initial Status(SM0-SM7)

ID	Function	Description	
SM000	Coil ON when running		SM000 keeps ON when PLC running
SM001	Coil OFF when running		SM001 keeps OFF when PLC running
SM002	Initial positive pulse coil		SM002 is ON in first scan cycle
SM003	Initial negative pulse coil		SM003 is OFF in first scan cycle
SM004	PLC running error	When SM4 sets ON, it indicates that there is an error in the operation of PLC. (Firmware version V3.4.5 and above supports this function by PLC)	
SM005	Battery low alarm coil	When the battery voltage is less than 2.5V, SM5 will put ON (at this time, please replace the battery as soon as possible, otherwise the data will not be maintained)	
SM007	Online download busy flag	Online download write back flash judge busy flag bit	
SM008	Capacitor charging completion sign	Capacitor charging completion (XDH, XLH, XG2 support)	

Clock(SM11-SM14)

ID	Function	Description
SM011	10ms frequency cycle	
SM012	100ms frequency cycle	

SM013	1s frequency cycle	
SM014	1min frequency cycle	

Mark(SM20-SM22)

ID	Function	Description
SM020	Zero bit	SM020 is ON when plus/minus operation result is 0
SM021	Borrow bit	SM021 is ON when minus operation overflows
SM022	Carry bit	SM022 is ON when plus operation overflows

PC Mode(SM30~M34)

ID	Function	Description
SM030	PLC initialize	Factory reset
SM032	Retentive register reset	When SM032 is ON, ON/OFF mapping memory of HM, HS and current values of HT, HC, HD will be reset.
SM033	Clear user's program	When SM033 is ON, all PLC user's program will be cleared.
SM034	All output forbidden	When SM034 is ON, all PLC external contacts will be set OFF.

Stepping Ladder

ID	Function	Description
SM040	The process is running	Set ON when the process is running

Interruption ban(SM50-SM90)

ID	Address	Function	Description
SM050	I0000/I0001	Forbid input interruption 0	After executing EI instruction, the input interruption couldn't act independently when M acts, even if the interruption is allowed. E.g.: when SM050 is ON,I0000/I0001 is forbidden.
SM051	I0100/I0101	Forbid input interruption 1	
SM052	I0200/I0201	Forbid input interruption 2	
SM053	I0300/I0301	Forbid input interruption 3	
SM054	I0400/I0401	Forbid input interruption 4	
……	……	……	
SM069	I1900/I1901	Forbid input interruption 19	
SM070	I40**	Forbid timing interruption 0	After executing EI instruction, the timing interruption couldn't act independently when M acts, even if the interruption is allowed.
SM071	I41**	Forbid timing interruption 1	
SM072	I42**	Forbid timing interruption 2	
SM073	I43**	Forbid timing interruption 3	
SM074	I44**	Forbid timing interruption 4	
……	……	……	
SM089	I59**	Forbid timing interruption 19	
SM090		Forbid all interruptions	Forbid all interruptions

High Speed Ring Counter(SM99)

address	Function	Note
SM099	High Speed Ring Counting enable	SM99 set ON，SD99 add one per 0.1ms, cycle between0 and 32767

High speed count complete(SM100-SM109)

Address	Function	Note
SM100	HSC0 count complete flag(100 segments)	
SM101	HSC2 count complete flag(100 segments)	
SM102	HSC4 count complete flag(100 segments)	
SM103	HSC6 count complete flag(100 segments)	
SM104	HSC8 count complete flag(100 segments)	
SM105	HSC10 count complete flag(100 segments)	
SM106	HSC12 count complete flag(100 segments)	
SM107	HSC14 count complete flag(100 segments)	
SM108	HSC16 count complete flag(100 segments)	
SM109	HSC18 count complete flag(100 segments)	

High speed counter direction(SM110-SM119)

Address	Function	Note
SM110	HSC0 direction flag	
SM111	HSC2 direction flag	
SM112	HSC4 direction flag	
SM113	HSC6 direction flag	
SM114	HSC8 direction flag	
SM115	HSC10 direction flag	
SM116	HSC12 direction flag	
SM117	HSC14 direction flag	
SM118	HSC16 direction flag	
SM119	HSC18 direction flag	

High speed counter error(SM120-SM129)

address	Function	Note
SM120	HSC0 error flag	
SM121	HSC2 error flag	
SM122	HSC4 error flag	
SM123	HSC6 error flag	
SM124	HSC8 error flag	
SM125	HSC10 error flag	
SM126	HSC12 error flag	
SM127	HSC14 error flag	
SM128	HSC16 error flag	
SM129	HSC18 error flag	

High peed counter overflow flag (SM130~SM139)

address	Function	Note
SM130	HSC0 overflow flag	
SM131	HSC2 overflow flag	
SM132	HSC4 overflow flag	
SM133	HSC6 overflow flag	
SM134	HSC8 overflow flag	
SM135	HSC10 overflow flag	
SM136	HSC12 overflow flag	
SM137	HSC14 overflow flag	
SM138	HSC16 overflow flag	
SM139	HSC18 overflow flag	

Communication(SM140-SM193)

	Address	Function	Note
Serial port 0	SM140	Modbus instruction execution flag	When the instruction starts to execute, set ON When execution is complete, set OFF
	SM141	X-NET instruction execution flag	When the instruction starts to execute, set ON When execution is complete, set OFF
	SM142	Free format communication sending flag	When the instruction starts to execute, set ON When execution is complete, set OFF
	SM143	Free format communication receive complete flag	When receiving a frame of data or receiving data timeout, set ON. Require user program to set OFF
Serial port 1	SM150	Modbus instruction execution flag	Same to SM140
	SM151	X-NET instruction execution flag	Same to SM141
	SM152	Free format communication sending flag	Same to SM142
	SM153	Free format communication receive complete flag	Same to SM143
Serial port 2	SM160	Modbus instruction execution flag	Same to SM140
	SM161	X-NET instruction execution flag	Same to SM141
	SM162	Free format communication sending flag	Same to SM142
	SM163	Free format communication receive complete flag	Same to SM143
Serial port 3	SM170	Modbus instruction execution flag	Same to SM140
	SM171	X-NET instruction execution flag	Same to SM141
	SM172	Free format communication sending flag	Same to SM142
	SM173	Free format communication receive complete flag	Same to SM143
Serial port 4	SM180	Modbus instruction execution flag	Same to SM140
	SM181	X-NET instruction execution flag	Same to SM141
	SM182	Free format communication sending flag	Same to SM142
	SM183	Free format communication receive complete flag	Same to SM143

Serial port 5	SM190	Modbus instruction execution flag	Same to SM140
	SM191	X-NET instruction execution flag	Same to SM141
	SM192	Free format communication sending flag	Same to SM142
	SM193	Free format communication receive complete flag	Same to SM143

Sequence Function BLOCK(SM300-SM399)

ID	Function	Description
SM300	BLOCK1 running flag	SM300 will be ON when block1 is running
SM301	BLOCK2 running flag	SM301 will be ON when block2 is running
SM302	BLOCK3 running flag	SM302 will be ON when block3 is running
SM303	BLOCK4 running flag	SM303 will be ON when block4 is running
SM304	BLOCK5 running flag	SM304 will be ON when block5 is running
SM305	BLOCK6 running flag	SM305 will be ON when block6 is running
……	……	
SM396	BLOCK97 running flag	SM396 will be ON when block97is running
SM397	BLOCK98 running flag	SM397 will be ON when block98 is running
SM398	BLOCK99 running flag	SM398 will be ON when block99 is running
SM399	BLOCK100 running flag	SM399 will be ON when block100 is running

Error check(SM400-SM415)

ID	Function	Description
SM400	I/O error	ERR LED keeps ON, PLC don not run and output, check when power on
SM401	Expansion module communication error	
SM402	BD communication error	
……		
SM405	No user program	Internal code check wrong
SM406	User program error	Implement code or configuration table check wrong
SM407	SSFD check error	ERR LED keeps ON, PLC don not run and output, check when power on
SM408	Memory error	Can not erase or write Flash
SM409	Calculation error	
SM410	Offset overflow	Offset exceeds soft element range
SM411	FOR-NEXT overflow	Reset when power on or users can also reset by hand.
SM412	Invalid data fill	When offset of register overflows, the return value will be SM372 value
SM413	Encrypted checksum error	
SM414	FLASH data error	

SM415	RTC real time clock error flag bit	RTC time and date verification failed

Error Message(SM450-SM463)

ID	Function	Description
SM450	System error check	
SM451	Hardfault interrupt flag	
……		
SM453	SD card error	
SM454	Power supply is cut off	
SM455	Power down keeps data error	
SM456	Online download error flag bit	
……		
SM460	Extension module ID not match	
SM461	BD/ED module ID not match	
SM462	Extension module communication overtime	
SM463	BD/ED module communication overtime	
SM464	The expansion module communication data overflow	
SM465	The BD/ED module communication data overflow	

Expansion Modules, BD Status(SM500)

ID	Function	Description
SM500	Module status read is finished	

Appendix 2 Special Data Register

Battery (SD5~SD7)

ID	Function	Description
SD005	Battery register	It will display 100 when the battery voltage is 3V, if the battery voltaeg is lower than 2.5V, it will display 0, it means please change new battery at once, otherwise the data will lose when PLC power off.

Clock(SD10-SD019)

ID	Function	Description
SD010	Current scan cycle	100us， us is the unit
SD011	Min scan time	100us， us is the unit
SD012	Max scan time	100us， us is the unit
SD013	Second (clock)	0~59 (BCD code)
SD014	Minute (clock)	0~59 (BCD code)

SD015	Hour (clock)	0~23 (BCD code)
SD016	Day (clock)	1~31 (BCD code)
SD017	Month (clock)	1~12 (BCD code)
SD018	Year (clock)	2000~2099 (BCD code)
SD019	Week (clock)	0(Sunday)~6(Saturday)(BCD code)

Flag (SD020-SD031)

ID	Function	Note
SD020	Model type	
SD021	model(low-8)series(high-8)	
SD022	Compatiable system version(low)system version(high)	
SD023	Compatiable model version(low)model version(high)	
SD024	Model info	
SD025	Model info	
SD026	Model info	
SD027	Model info	
SD028	Suitable software version	
SD029	Suitable software version	
SD030	Suitable software version	
SD031	Suitable software version	

Step ladder(SD040)

ID	Function	Description
SD40	Flag of the executing process S	

Step ladder(SD099)

ID	Function	Description
SD99	High speed ring counter	When SM99 is set to on, SD99 adds 1 every 0.1ms, and circulates between 0 and 32767.

High Speed Counting(SD100-SD109)

ID	Function	Description
SD100	Current segment (No. n segment)	HSC00
SD101	Current segment (No. n segment)	HSC02
SD102	Current segment (No. n segment)	HSC04
SD103	Current segment (No. n segment)	HSC06
SD104	Current segment (No. n segment)	HSC08
SD105	Current segment (No. n segment)	HSC10

SD106	Current segment (No. n segment)	HSC12
SD107	Current segment (No. n segment)	HSC14
SD108	Current segment (No. n segment)	HSC16
SD109	Current segment (No. n segment)	HSC18

High speed counter error(SD120-SD129)

ID	Function	Note
SD120	HSC0 error info	
SD121	HSC2 error info	
SD122	HSC4 error info	
SD123	HSC6 error info	
SD124	HSC8 error info	
SD125	HSC10 error info	
SD126	HSC12 error info	
SD127	HSC14 error info	
SD128	HSC16 error info	
SD129	HSC18 error info	

communication(SD140~SD199)

	ID	Function	Note
Serial port 0	SD140	Modbus read write instruction execution result	0: correct 100: receive error 101: receive overtime 180: CRC error 181: LRC error 182: station error 183: send buffer overflow 400: function code error 401: address error 402: length error 403: data error 404: slave station busy 405: memory error(eraseFLASH)
	SD141	X-Net communication result	0: correct 1: communication overtime 2: memory error 3: receive CRC error
	SD142	Free format communication send result	0: correct 410: free format send buffer overflow
	SD143	Free format communication receive result	0: correct 410: send data length overflow 411: receive data short 412: receive data long 413: receive error 414: receive overtime

			415: no start character 416: no end character
	SD144	Free format communication receive data numbers	In bytes, there are no start and stop characters
	……		
	SD149		
Serial port 1	SD150	Modbus read write instruction execution result	0: correct 100: receive error 101: receive overtime 180: CRC error 181: LRC error 182: station error 183: send buffer overflow 400: function code error 401: address error 402: length error 403: data error 404: slave station busy 405: memory error(eraseFLASH)
	SD151	X-Net communication result	0: correct 1: communication overtime 2: memory error 3: receive CRC error
	SD152	Free format communication send result	0: correct 410: free format send buffer overflow
	SD153	Free format communication receive result	0: correct 410: send data length overflow 411: receive data short 412: receive data long 413: receive error 414: receive overtime 415: no start character 416: no end character
	SD154	Free format communication receive data numbers	In bytes, there are no start and stop characters
	……		
	SD159		
Serial port 2	SD160	Modbus read write instruction execution result	0: correct 100: receive error 101: receive overtime 180: CRC error 181: LRC error 182: station error 183: send buffer overflow 400: function code error 401: address error 402: length error 403: data error 404: slave station busy 405: memory error(eraseFLASH)

	SD161	X-Net communication result	0: correct 1: communication overtime 2: memory error 3: receive CRC error
	SD162	Free format communication send result	0: correct 410: free format send buffer overflow
	SD163	Free format communication receive result	0: correct 410: send data length overflow 411: receive data short 412: receive data long 413: receive error 414: receive overtime 415: no start character 416: no end character
	SD164	Free format communication receive data numbers	In bytes, there are no start and stop characters
	……		
	SD169		
Serial port 3	SD170~SD179		
Serial port 4	SD180~SD189		
Serial port 5	SD190~SD199		

Sequence Function Block(SD300-SD399)

ID	Function	Description
SD300	Executing instruction of BLOCK1	The value will be used when BLOCK monitors
SD301	Executing instruction of BLOCK2	The value will be used when BLOCK monitors
SD302	Executing instruction of BLOCK3	The value will be used when BLOCK monitors
SD303	Executing instruction of BLOCK4	The value will be used when BLOCK monitors
SD304	Executing instruction of BLOCK5	The value will be used when BLOCK monitors
SD305	Executing instruction of BLOCK6	The value will be used when BLOCK monitors
……	……	……
SD396	Executing instruction of BLOCK97	The value will be used when BLOCK monitors
SD397	Executing instruction of BLOCK98	The value will be used when BLOCK monitors
SD398	Executing instruction of BLOCK99	The value will be used when BLOCK monitors
SD399	Executing instruction of BLOCK100	The value will be used when BLOCK monitors

Error Check(SD400-SD413)

ID	Function	Note
SD400		
SD401	Extension module no. of communication error	Means module no.n is error
SD402	BD/ED module no. of communication error	
SD403	FROM/TO error type	
SD404	PID error type	
SD405	No user program	
SD406	User program error type	
SD407	SSDF error type	
SD408	Erasure flash error type	
SD409	Calculation error code	1: divide by 0 error 2: MRST, MSET front operand address less than back operand 3: ENCO, DECO data bits of encoding and decoding instructions exceed the limit. 4: BDC code error 7: Radical sign error
SD410	The number of offset register D when offset crosses the boundary	
SD411		
SD412	Invalid data fill value (low16 bits)	
SD413	Invalid data fill value (high16 bits)	
SD414	Flash register data error type	
SD415	RTC real time clock error type	1: The RTC power supply has a low voltage condition and needs to be rewritten 2: RTC writes data, and the clock chip does not respond to the ACK signal 3: Write illegal time date data

Error Check(SD450-SD465)

ID	Function	Description
SD450	1：Watchdog act (Default 200ms) 2：Control block application fail 3：Visit illegal address	
SD451	Hardware error type: 1：Register error 2：Bus error 3：Usage error	
SD452	Hardware error	
SD453	SD card error	
SD454	Power-off time	
SD455		

SD456		
SD460	Extension module ID not match	
SD461	BD/ED module ID not match	
SD462	Extension module communication overtime	
SD463	BD/ED module communication overtime	
SD464	Communication data overflow of expansion module number	
SD465	BD/ED module number communication data overflow	

Expansion Modules, BD Status(SD500-SD516)

ID	Function	Description
SD500	Module number Expansion modules： #10000~10015 BD：#20000~20001 ED: #30000	
SD501~516	Expansion module, BD /ED status	16 registers

Module info(SD520-SD823)

ID	Function	Explanation	Note
SD520~SD535	Extension module info	Extension module 1	Each extension module, BD, ED occupies 16 registers
……	……	……	
SD760~SD775	Extension module info	Extension module 16	
SD776~SD791	BD module info	BD module 1	
SD792~SD807	BD module info	BD module 2	
SD808~SD823	ED module info	ED module 1	

Expansion Module Error Information

ID	Function	Description	
SD860	Error times of module read		Expansion module 1
SD861	Error types of module read	Module address error. Module accepted data length error. Module CRC parity error when PLC is accepting data. Module ID error. Module overtime error.	
SD862	Error times of module write		
SD863	Error types of module		

	write		
SD864	Error times of module read		Expansion module 2
SD865	Error types of module read	Module address error. Module accepted data length error. Module CRC parity error when PLC is accepting data. Module ID error. Module overtime error.	
SD866	Error times of module write		
SD867	Error types of module write		
……			
SD920	Error times of module read		Expansion module 16
SD921	Error types of module read	Module address error. Module accepted data length error. Module CRC parity error when PLC is accepting data. Module ID error. Module overtime error.	
SD922	Error times of module write		
SD923	Error types of module write		
SD930	Current ED read/write duration	100us,unit: us	ED module 1
SD931	Minimum ED read/write duration		
SD932	Maximum ED read/write duration		
SD933	ED read/write error duration		
SD934	Total number of ED read and write (low16 bits)		
SD935	Total number of ED reads and writes (high16 bits)		
SD936	Current BD1 read/write duration	100us,unit: us	BD module 1
SD937	Minimum BD1 read/write duration		
SD938	Maximum BD1 read/write duration		
SD939	BD1 read/write error times		
SD940	Total number of BD1 read and write (low16 bits)		
SD941	Total number of BD1 reads and writes (high16 bits)		
SD942	Current BD2 read/write duration		BD module 2
SD943	Minimum BD2 read / write duration read module error type		

SD944	Maximum BD1 read/write duration		
SD945	BD1 read/write error times		
SD946	Total number of BD2 read and write (low16 bits)		
SD947	Total number of BD2 reads and writes (high16 bits)		

Version info(SD990~SD993)

ID	Function	Explanation	Note
SD990	Firmware version date	Low 16-bit	
SD991	Firmware version compilation date	High 16-bit	
SD992	FPGA version compilation date	Low 16-bit	
SD993	FPGA version compilation date	High 16-bit	

Special function(HSD50~HSD60)

ID	Function	Description
HSD50	Keep data write back time after power failure※1	Single word,unit:1ms
HSD51	Power failure detection	CPU working time after power failure,unit:100us
HSD52	Last PLC operation time(low16 bits)	Double word,unit:1s
HSD53	Last PLC operation time(high16 bits)	
HSD54	Current PLC operation time (low16 bits)	Double word,unit:1s
HSD55	Current PLC operation time (high16 bits)	
HSD58	Flash register erasure count	

Error record(HSD80~HSD179)

ID	Function	Description
HSD79	Error list index value	
HSD80~HSD84	Article 1 error message	
HSD85~HSD89	Article 2 error message	
HSD90~HSD94	Article 3 error message	
HSD95~HSD99	Article 4 error message	
HSD100~HSD104	Article 5 error message	

HSD105~HSD109	Article 6 error message	(1) XDC series PLC only supports the storage of 4 error history messages; (2) This function requires programming software version v3.5.3 (20190326) and above. (3) H motion mode supports up to 20 error messages, and C motion mode supports up to 4 error messages.
HSD110~HSD114	Article 7 error message	
HSD115~HSD119	Article 8 error message	
HSD120~HD124	Article 9 error message	
HSD125~HSD129	Article 10 error message	
HSD130~HD134	Article 11 error message	
HSD135~HSD139	Article 12 error message	
HSD140~HD144	Article 13 error message	
HSD145~HSD149	Article 14 error message	
HSD150~HD154	Article 15 error message	
HSD155~HSD159	Article 16 error message	
HSD160~HSD164	Article 17 error message	
HSD165~HSD169	Article 18 error message	
HSD170~HSD174	Article 19 error message	
HSD175~HSD179	Article 20 error message	
HSD180~HSD184	Article 21 error message	H motion mode is extended to 76 error messages (firmware version v3.7.2 and above support)
HSD185~HSD189	Article 22 error message	
……		
HSD450~HSD454	Article 75 error message	
HSD455~HSD459	Article 76 error message	

Notes:

※ 1: HSD50 is "maintain data write back time after power failure" in v3.7.2 and above.

Appendix 3 Special Flash Register

Special FLASH data register SFD

*** means it works only after repower on the PLC**

I filtering

ID	Function	Description
SFD0*	Input filter time	
SFD2*	Watchdog run-up time, default value is 200ms	

Special function configuration

ID	Function	Description
SFD3*	Special function configuration(default:0x0000)	Bit0:Power down memory register exception handling. 0: the system clears it; 1: No processing. Bit1: Execute user program in external interrupt subroutine. 0: execute in task; 1: Execute in interrupt (in this mode, the user interrupt subroutine cannot contain C language function block). This mode is generally used in occasions that require high real-time performance of external signals. Bit2: whether to raise the external interrupt priority. 0: not raise; 1: raise (raise to the highest).

I Mapping

ID	Function	Description	
SFD10*	I00 corresponds to X**	Input terminal 0 corresponds to X** number	0xFF means terminal bad，0xFE means terminal idle
SFD11*	I01 corresponds to X**		
SFD12*	I02 corresponds to X**		
……	……		
SFD73*	I77 corresponds to X**	Default value is 77(Octonary)	

O Mapping

ID	Function	Description	
SFD74*	O00 corresponds to Y**	Output terminal 0 correspond to Y** number，Default value is 0	0xFF means terminal bad，0xFE means terminal idle
……	……		
SFD137 *	O77 corresponds to Y**	Default value is 77(Octonary)	

I Attribute

ID	Function	Description	
SFD138*	I00 attribute	Attribute of input terminal 0	0：positive logic others：negative logic
SFD139*	I01 attribute		
……	……		
SFD201*	I77 attribute		

High Speed Counting

ID	Function	Description
SFD310	HSC0 single phase counting edge configuration	0: rising edge count, 1: Falling edge count, 2: Both rising and falling edges are counted
SFD311	HSC2 single phase counting edge configuration	0: rising edge count, 1: Falling edge count, 2: Both rising and falling edges are counted
SFD312	HSC4 single phase counting edge configuration	0: rising edge count, 1: Falling edge count, 2: Both rising and falling edges are counted
SFD313	HSC6 single phase counting edge configuration	0: rising edge count, 1: Falling edge count, 2: Both rising and falling edges are counted
SFD320	HSC0 frequency times	2：2 times frequency; 4: 4 times frequency(effective at AB phase counting mode)
SFD321	HSC2 frequency times	Ditto
SFD322	HSC4 frequency times	Ditto
SFD323	HSC6 frequency times	Ditto
SFD324	HSC8 frequency times	Ditto
SFD325	HSC10 frequency times	Ditto
SFD326	HSC12 frequency times	Ditto
SFD327	HSC14 frequency times	Ditto
SFD328	HSC16 frequency times	Ditto
SFD329	HSC18 frequency times	Ditto
SFD330	Bit selection of HSC absolute and relative(24 segment)	bit0 corresponds to HSC0，bit1corresponds to HSC2，and so on, bit9 corresponds to HSC18 0：relative 1：absolute
SFD331	Interrupt circulating of 24 segments high speed counting	bit0 corresponds to HSC0，bit1corresponds to HSC2，and so on, bit9 corresponds to HSC18 0：single 1：loop
SFD332	CAM function	bit0 corresponds to HSC0，bit1corresponds to HSC2，and so on, bit9 corresponds to HSC18 0：do not support CAM function 1：support CAM function

Expansion Module Configuration

ID	Function	Explanation
SFD340	Extension module configuration status(#1#2)	Configuration Status of Extension Modules 1 and 2
SFD341	Extension module configuration status(#3#4)	Configuration Status of Extension Modules 3 and 4
……	……	……
SFD347	Extension module configuration status(#15#16)	Configuration Status of Extension Modules 15 and 16
SFD348	BD module configuration status(#1#2)	Configuration Status of BD Modules 1 and 2
SFD349	ED module configuration status(#1)	Configuration Status of ED Module 1
SFD350 ⋮ SFD359	Extension module configuration	Configuration of Extension Module 1
SFD360 ⋮ SFD369	Extension module configuration	Configuration of Extension Module 2
⋮	⋮	
SFD500 ⋮ SFD509	Extension module configuration	Configuration of Extension Module 16
SFD510 ⋮ SFD519	BD module configuration	Configuration of BD Module 1
SFD520 ⋮ SFD529	BD module configuration	Configuration of BD Module 2
SFD530 ⋮ SFD539	ED module configuration	Configuration of ED Module 1

Communication

ID	Function	Note
SFD600	COM1 free format communication buffer bit numbers	0: 8-bit 1: 16-bit
SFD610	COM2 free format communication buffer bit numbers	0: 8-bit 1: 16-bit
SFD620	COM3 free format communication buffer bit numbers	0: 8-bit 1: 16-bit
SFD630	COM4 free format communication buffer bit numbers	0: 8-bit 1: 16-bit

SFD640	COM5 free format communication buffer bit numbers	0: 8-bit 1: 16-bit

Appendix 4 PLC resource conflict table

When PLC is used in practice, conflicts may arise because some resources are used at the same time. This section will list the resources that may cause conflicts in each PLC model. This part mainly refers to high-speed counting, accurate timing and pulse output.

Accurate timing	High speed counting				Pulse output
XD2-16, XD3-16, XD5-16, XL3-16/32,XL5-16,XL5E-16					
ET0	-	-	-	-	-
ET2					
ET4					
ET6					
ET8	HSC0				
ET10		HSC2			
ET12			HSC4		
ET14					Y0
ET16					Y0
ET18					Y1
ET20					Y1
ET22					
ET24					
XD3-24/32/48/60, ZG3-30					
ET0					
ET2					
ET4					
ET6					
ET8					
ET10					
ET12	HSC0				
ET14		HSC2			
ET16			HSC4		
ET18					Y0
ET20					Y0
ET22					Y1
ET24					Y1
XD5-24/32/48/60, XDM-24/32/48/60, XD5E-24/30/48/60, XDME-30/60, XL5-32, XL5E-32/64					
ET0	-	-	-	-	-
ET2				HSC6	
ET4			HSC4		
ET6		HSC2			
ET8	HSC0				

ET10					Y3
ET12					Y3
ET14					Y2
ET16					Y2
ET18					Y1
ET20					Y1
ET22					Y0
ET24					Y0
XDC-24/32/48/60					
ET0	-	-	-	HSC6	-
ET2			HSC4		
ET4		HSC2			
ET6	HSC0				
ET8					Y3
ET10					Y3
ET12					Y2
ET14					Y2
ET16					Y1
ET18					Y1
ET20					Y0
ET22					Y0
ET24					

※1：This form should be read horizontally. Any two resources in each row cannot be used at the same time. Otherwise, it will cause conflict.

Appendix 5 PLC function configuration list

This part is used to check each model's configurations. Via this table, we can judge products type easily.

○ Selectable × Not support √ Support

Model	USB port	232 port	485 port	RJ45 port	Extension module	Extension BD	Left extension ED	HSC channel Single/AB phase		Pulse output (T/RT)		External interruption
								OC	Differential	normal	Differential	
XD1 series												
XD1-10	×	2	×	×	×	×	×	×	×	×	×	6
XD1-16	×	2	×	×	×	×	×	×	×	×	×	6
XD1-24	×	2	1	×	×	×	×	×	×	×	×	10
XD1-32	×	2	1	×	×	×	×	×	×	×	×	10
XD2 series												
XD2-16	×	2	1	×	×	×	1	3	×	2	×	6
XD2-24	×	2	1	×	×	1	1	3	×	2	×	10
XD2-32	×	2	1	×	×	1	1	3	×	2	×	10
XD2-42	×	2	1	×	×	1	1	3	×	2	×	10
XD2-48	×	2	1	×	×	2	1	3	×	2	×	10
XD2-60	×	2	1	×	×	2	1	3	×	2	×	10

Model	USB port	232 port	485 port	RJ45 port	Extension module	Extension BD	Left extension ED	HSC channel Single/AB phase		Pulse output (T/RT)		External interruption
								OC	Differential	normal	Differential	
XD3 series												
XD3-16	1	1	1	×	10	×	1	3	×	2	×	6
XD3-24	1	1	1	×	10	1	1	3	×	2	×	10
XD3-24T4	1	1	1	×	10	1	1	3	×	4	×	10
XD3-32	1	1	1	×	10	1	1	3	×	2	×	10
XD3-32T4	1	1	1	×	10	1	1	3	×	4	×	10
XD3-42	1	1	1	×	10	1	1	3	×	4	×	10
XD3-48	1	1	1	×	10	2	1	3	×	2	×	10
XD3-60	1	1	1	×	10	2	1	3	×	2	×	10
XD5 series												
XD5-16	1	1	1	×	16	×	1	3	×	2	×	6
XD5-24	1	1	1	×	16	1	1	3	×	2	×	10
XD5-32	1	1	1	×	16	1	1	3	×	2	×	10
XD5-42	1	1	1	×	16	1	1	3	×	2	×	10
XD5-48	1	1	1	×	16	2	1	3	×	2	×	10
XD5-60	1	1	1	×	16	2	1	3	×	2	×	10
XD5-80	1	1	1	×	16	2	1	3	×	2	×	10
XD5-24T4	1	1	1	×	16	1	1	4	×	4	×	10
XD5-24D2T2	1	1	1	×	16	1	1	2	2	2	2	10
XD5-32T4	1	1	1	×	16	1	1	4	×	4	×	10
XD5-48T4	1	1	1	×	16	2	1	4	×	4	×	10
XD5-48D4T4	1	1	1	×	16	2	1	4	4	4	4	10
XD5-48T6	1	1	1	×	16	2	1	6	×	6	×	10
XD5-60T4	1	1	1	×	16	2	1	4	×	4	×	10
XD5-60T6	1	1	1	×	16	2	1	6	×	6	×	10
XD5-60T10	1	1	1	×	16	2	1	10	×	10	×	10
XDM series												
XDM-24T4	1	1	1	×	16	1	1	4	×	4	×	10
XDM-32T4	1	1	1	×	16	1	1	4	×	4	×	10
XDM-60T4	1	1	1	×	16	2	1	4	×	4	×	10
XDM-60T4L	1	1	1	×	16	2	1	4	×	4	×	10
XDM-60T10	1	1	1	×	16	2	1	10	×	10	×	10
XDC series												
XDC-24	×	2	1	×	16	1	1	4	×	2	×	10
XDC-32	×	2	1	×	16	1	1	4	×	2	×	10
XDC-48	×	2	1	×	16	2	1	4	×	2	×	10
XDC-60	×	2	1	×	16	2	1	4	×	2	×	10
XD3E series												
XD3E-24	×	1	1	2	10	1	1	3	×	2	×	10
XD5E series												
XD5E-24	×	1	1	2	16	1	1	3	×	2	×	10
XD5E-30	×	1	1	2	16	1	1	3	×	2	×	10

Model	USB port	232 port	485 port	RJ45 port	Extension module	Extension BD	Left extension ED	HSC channel Single/AB phase		Pulse output (T/RT)		External interruption
								OC	Differential	normal	Differential	
XD5E-30T4	×	1	1	2	16	1	1	4	×	4	×	10
XD5E-48	×	1	1	2	16	2	1	3	×	2	×	10
XD5E-60	×	1	1	2	16	2	1	3	×	2	×	10
XD5E-60T4	×	1	1	2	16	2	1	4	×	4	×	10
XD5E-60T6	×	1	1	2	16	2	1	6	×	6	×	10
XD5E-60T10	×	1	1	2	16	2	1	10	×	10	×	10
XDME series												
XDME-30T4	×	1	1	2	16	1	1	4	×	4	×	10
XDME-60T4	×	1	1	2	16	2	1	4	×	4	×	10
XDME-60T10	×	1	1	2	16	2	1	10	×	10	×	10
XDH series												
XDH-30A16	×	1	1	2	16	×	1	4	×	4	×	10
XDH-30A16L	×	1	1	2	16	×	1	4	×	4	×	10
XDH-60A32	×	1	1	2	16	1	1	4	×	4	×	10
XDH-60A64	×	1	1	2	16	1	1	4	×	4	×	10
XDH-60T4	×	1	1	2	16	1	1	4	×	4	×	10
XL1 series												
XL1-16	×	2※1	1	×	×	×	×	×	×	×	×	6
XL1-16T-U	1	1	1	×	×	×	×	×	×	×	×	6
XL3 series												
XL3-16	1	1	1	×	10	×	1	3	×	2	×	6
XL3-32	1	1	1	×	10	×	1	3	×	2	×	10
XL5 series												
XL5-16	1	1	1	×	16	×	1	3	×	2	×	6
XL5-32	1	1	1	×	16	×	1	3	×	2	×	10
XL5-32T4	1	1	1	×	16	×	1	4	×	4	×	10
XL5-64T10	1	1	1	×	16	×	1	10	×	10	×	10
XL5E series												
XL5E-16	×	1	1	2	16	×	1	3	×	2	×	6
XL5E-32	×	1	1	2	16	×	1	3	×	2	×	10
XL5E-32T4	×	1	1	2	16	×	1	4	×	4	×	10
XL5E-64T6	×	1	1	2	16	×	1	6	×	6	×	10
XL5E-64T10	×	1	1	2	16	×	1	10	×	10	×	10
XL5N series												
XL5N-32	×	1	1	2	16	×	1	3	×	2	×	10
XLME series												
XLME-32T4	×	1	1	2	16	×	1	4	×	4	×	10
XLME-64T10	×	1	1	2	16	×	1	10	×	10	×	10
XLH series												
XLH-24A16	×	1	1	2	16	×	1	4	×	4	×	10
XLH-24A16L	×	1	1	2	16	×	1	4	×	4	×	10

Model	USB port	232 port	485 port	RJ45 port	Extension module	Extension BD	Left extension ED	HSC channel Single/AB phase		Pulse output (T/RT)		External interruption
								OC	Differential	normal	Differential	
XL5H series												
XL5H-24A8L	×	1	1	2	16	×	1	3	×	2	×	10

Note:

1. The XL1-16T with hardware version below H4 has only one RS232 port (COM1).

2: All models are equipped with clock function as standard.

Wechat ID

WUXI XINJE ELECTRIC CO., LTD.

No.816, Jianzhu West Road, Binhu District, Wuxi City, Jiangsu Province, China
Tel: 400-885-0136
Fax: 86-510-85111290
Email: fiona.xinje@vip.163.com
www.xinje.com