XD3/XDM series PLC
User manual [Instruction]

WUXI XINJE ELECTRIC CO., LTD.

Data No. PC11 20150510 3.3

XINJE

XD3/XDM series PLC
User manual［Instruction］

- Basic explanation

Thank you for purchasing Xinje XD3/XDM series PLC.

This manual mainly introduces XD3/XDM 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

XD3/XDM 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

XD3/XDM 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. XD3/XDM series added new register FS. (For different XD3/XDM 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

XD3/XDM series PLC has enough basic instructions including basic sequential control, data moving and comparing, arithmetic operation, logic control, data loop and shift etc. XD3/XDM series PLC also support interruption, high speed pulse, frequency testing, precise time, PID control and so on.

C Language Function Block

XD3/XDM series PLC support C language; users can call the C program in ladder chart. This function improves the programming efficiency.

Stop PLC when reboot

XD3/XDM 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

XD3/XDM 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

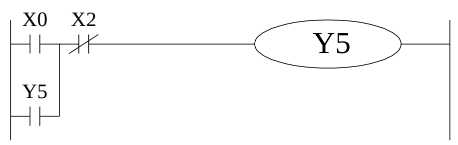
XD3/XDM series PLC support two types of programming language:

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;

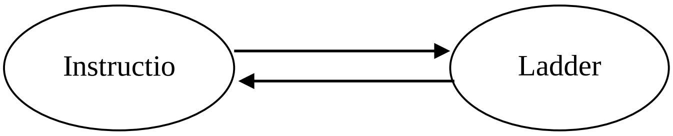
E.g.: step	instruction	operand
0	LD	X000
1	OR	Y005
2	ANI	X002
3	OUT	Y005

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 PLC status online.
E.g.:



1-2-2. Alternation

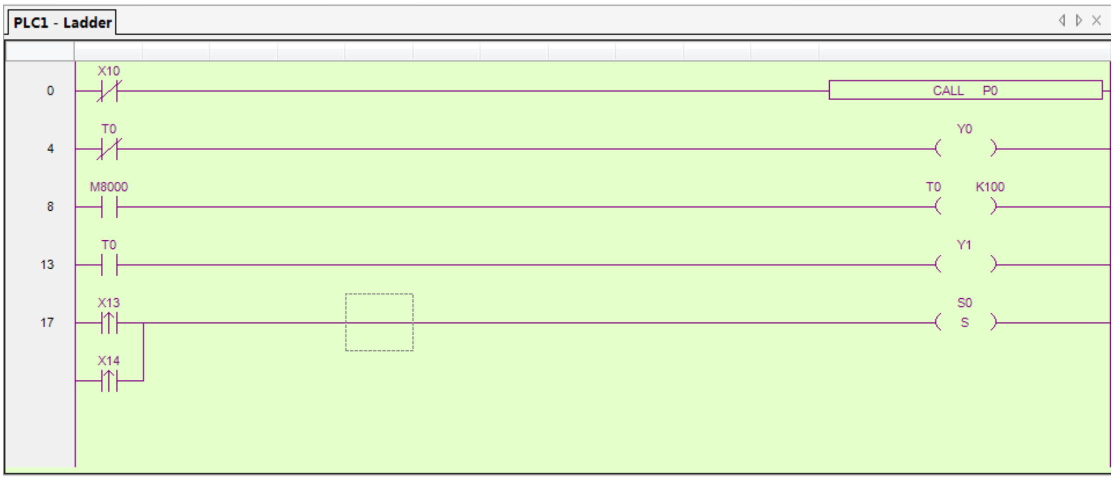
The two kinds of programming language can be transformed to each other.



1-3. Programming mode

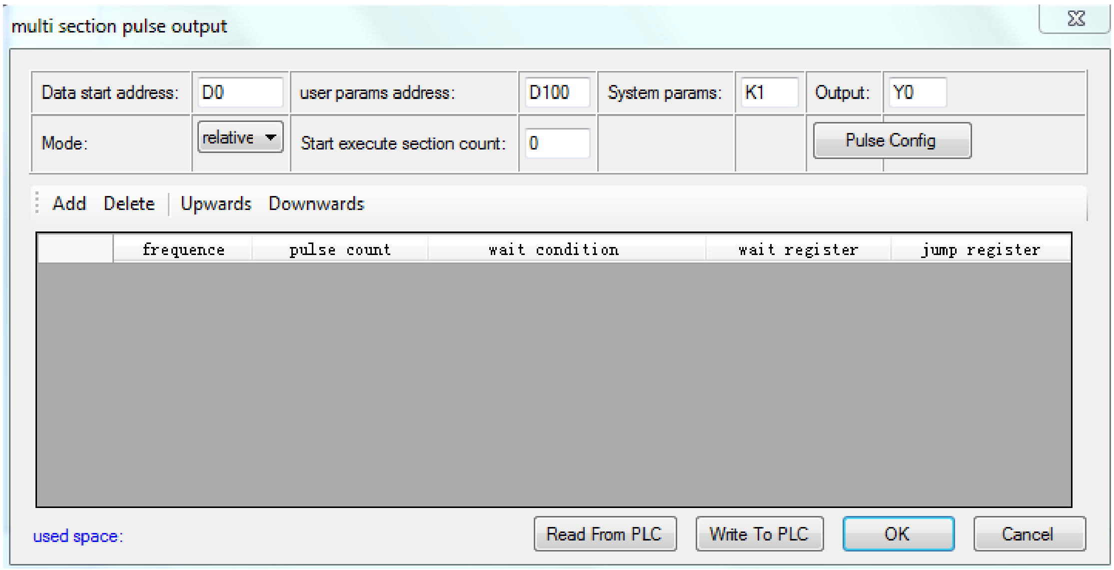
Direct Input

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 XD3/XDM 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 Assignment Principle

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…

Up to 10 extension modules can be connected to the XD3 main unit (XDM can connect 16 modules).

Extension BD board: BD 1 starts from X20000; BD 2 starts from X20100…. Up to 1 BD can be connected to the main unit.

Using notes

The input filter of input relay is digital one; user can change the filter parameters.

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…

Up to 10 extension modules can be connected to the XD3 main unit (XDM can connect 16 modules).

Extension BD board: BD 1 starts from Y20000; BD 2 starts from Y20100…. Up to 1 BDs can be connected to the main unit.

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 decimal form.
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 counter will 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 XD3/XDM series PLC are 16 bits (the highest bit is sign bit), combine two data registers together is for 32 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 time when writing. Frequently writing can cause permanent damage for FD.

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.

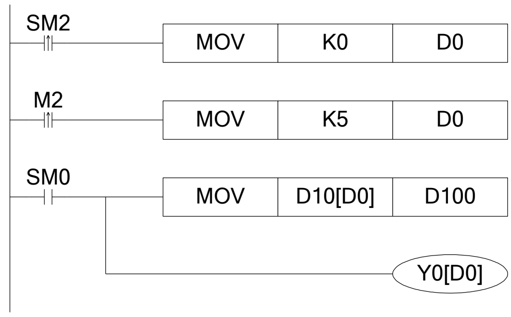
2-2. Structure of Soft Components

2-2-1. Structure of Memory

In XD3/XDM series PLC, there are many registers. Besides D, HD, FlashROM registers, we can also combine bit to register.

Data Register D, HD

For common use, 16 bits
For common use, 32 bits (combine two continuous 16-bits registers)
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.

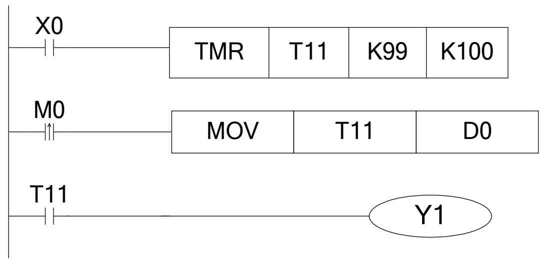
Timer T, HT/Counter C, HC

For common usage, 16 bits, represent the current value of timer/counter;

For common usage, 32 bits, (combine two continuous 16 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, 16 bits, (combine two continuous 16 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, (combine 16 bits)

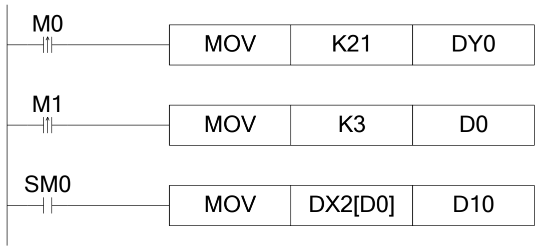
The soft components which can be combined to words are: X, Y, M, S, T, C, HM, HS, HT, HC.

Format: add “D” in front of soft components, like DM10, represents a 16-bits register from M10~M25

Get 16 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

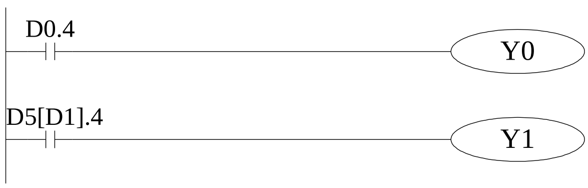
Composed by bit of register, support register D

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

The soft components range of XD3 main unit and extension module:

	Name	Range					Points				
		16	24	32	48	60	16	24	32	48	60
X	Input	X0~X7	X0~X15	X0~X21	X0~X33	X0~X43	8	14	18	28	36
Y	Output	Y0~Y7	Y0~Y11	Y0~Y15	Y0~Y23	Y0~Y27	8	10	14	20	24

X	Input ※3	X10000～10077(#1 expansion module) …… X11100～11177(#10 expansion module)	640
Y	Output ※3	Y10000～10077(#1 expansion module) …… Y11100～11177(#10 expansion module)	640
X	Input ※4	X20000～20077(#1 expansion BD)	64
Y	Output ※4	Y20000～20077(#1 expansion BD)	64
M	Internal register	M0~M7999	8000
HM		HM0~HM959※1	960
SM		Special register 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 timing 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 register SD0~SD2047	2048
HSD		Special register HSD0~HSD499※2	500
FD	FlashROM register	FD0~FD6143	6144
SFD		Special register SFD0~SFD1999※2	2000
FS	Special secure register	FS0~FS47	48
ID※5	Main unit	ID0~99	100
	Expansion module	ID10000～10099(#1 expansion module) …… ID10900～10999(#10 expansion module)	1000
	Expansion BD	ID20000～20099(#1 expansion BD)	100
QD※6	Main unit	QD0~99	100
	Expansion module	QD10000～10099(#1 expansion module) …… QD10900～10999(#10 expansion module)	1000
	Expansion BD	QD20000～20099(#1 expansion BD)	100
SEM	Sequence function block instruction WAIT special coil	SEM0~SEM31	32

The soft components range of XDM main unit and extension module:

	Name	Range			Points		
		24	32	60	24	32	60
X	Input	X0~X15	X0~X21	X0~X43	14	18	36
Y	Output	Y0~Y11	Y0~Y15	Y0~Y27	10	14	24

X	Input ※3	X10000～10077(#1 expansion module) …… X11700～11777(#16 expansion module)	1024
Y	Output ※3	Y10000～10077(#1 expansion module) …… Y11700～11777(#16 expansion module)	1024
X	Input ※4	X20000～20077(#1 expansion BD)	64
Y	Output ※4	Y20000～20077(#1 expansion BD)	64
M	Internal register	M0~M20479	20480
HM		HM0~HM6143※1	6144
SM		Special register SM0~SM4095※2	4096
S	Flow	S0~S7999	8000
HS		HS0~HS999※1	1000
T	Timer	T0~T4095	4096
HT		HT0~HT1023※1	1024
ET		Precise timing ET0~ET39	40
C	Counter	C0~C4095	4096
HC		HC0~HC1023※1	1024
HSC		High speed counter HSC0~HSC39	40
D	Data register	D0~D20479	20480
HD		HD0~HD24999※1	25000
SD		Special register SD0~SD4095	4096
HSD		Special register HSD0~HSD1023※2	1024
FD	FlashROM register	FD0~FD8191	8192
SFD		Special register SFD0~SFD4095※2	4096
FS	Special secure register	FS0~FS255	256
ID※5	Main unit	ID0~99	100
	Expansion module	ID10000～10099(#1 expansion module) …… ID11500～11599(#16 expansion module)	1600
	Expansion BD	ID20000～20099(#1 expansion BD)	100
QD※6	Main unit	QD0~99	100
	Expansion module	QD10000～10099(#1 expansion module) …… QD11500～11599(#16 expansion module)	1600
	Expansion BD	QD20000～20099(#1 expansion BD)	100
SEM	Sequence function block instruction WAIT special coil	SEM0~SEM127	128

※1: Power-off retentive range, the range cannot be changed.

※2: For system special use (not power-off retentive), they cannot be used for other way. Please refer to appendix Special soft components.

※3: Extension module I/O addresses assignment (octal), XD3 can connect 10 modules, XDM can connect 16 modules.

※4: Extension BD I/O addresses assignment (octal), up to 1 BD can be extended.

※5: Analog input addresses.
※6: Analog output addresses.

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

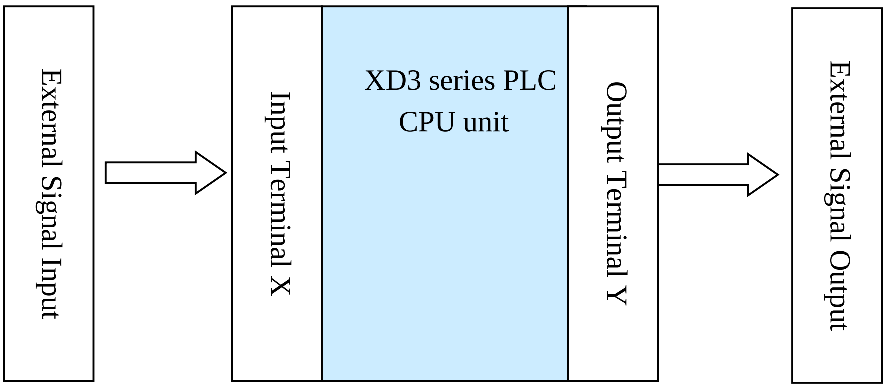
Number List

XD3/XDM series PLC input/output are all in octal form, each series numbers are listed below:

PLC	Name	Range					Points				
		16	24	32	48	60	16	24	32	48	60
XD3	X	X0~X7	X0~X15	X0~X21	X0~X33	X0~X43	8	14	18	28	36
	Y	Y0~Y7	Y0~Y11	Y0~Y15	Y0~Y23	Y0~Y27	8	10	14	20	24

PLC	Name	Range			Points		
		24	32	60	24	32	60
XDM	X	X0~X15	X0~X21	X0~X43	14	18	36
	Y	Y0~Y11	Y0~Y15	Y0~Y27	10	14	24

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

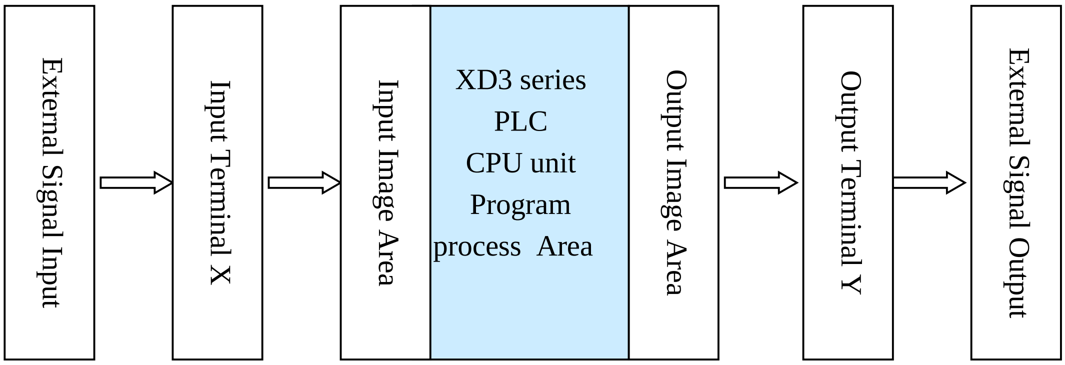
The input relays which are not connected with external devices can be seemed to fast internal relays

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 XD3/XDM series PLC are all in decimal form, please see the following table:

PLC	Name	Range		
		Normal	Power-off retentive	Special
XD3	M	M000~M7999	HM0-HM959	SM0~SM2047
XDM		M000~M20479	HM0-HM6143	SM0~SM4095

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 for special 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;

2-6. Status Relay (S, HS)

Address List

Status relays addresses are in form of decimal, the address are shown below:

PLC	Name	Range	
		Normal	Power-off retentive
XD3	S	S000~S1023	HS0~HS127
XDM		S000~S7999	HS0~HS999

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

2-7. Timer (T, HT)

Address List

The timer addresses are in the form of decimal; please see the following table:

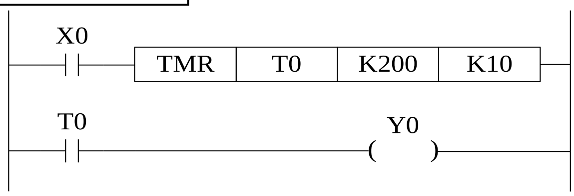
Series	Name	Range		
		Normal	Power-off retentive	Precise timing
XD3	T HT ET	T0~T575	HT0~HT95	ET0~ET31
XDM		T0~T4095	HT0~HT1023	ET0~ET39

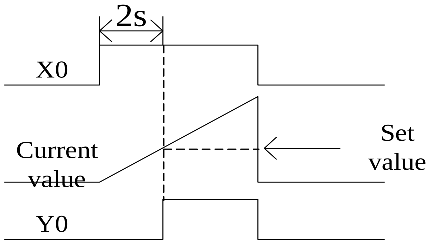
Function

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).

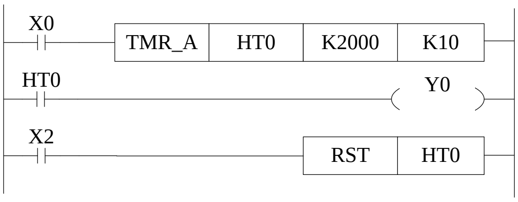
Normal type

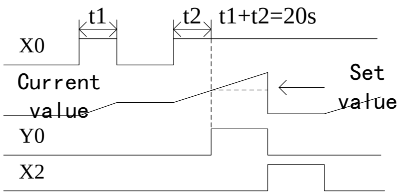




If X0 is ON, then T0 accumulates 10ms pulse based on the current value; when the accumulation value reaches the set value K200, the timer output activates. I.e. the output activates 2s later. If X0 is OFF, the timer resets, the output resets;

Accumulation type





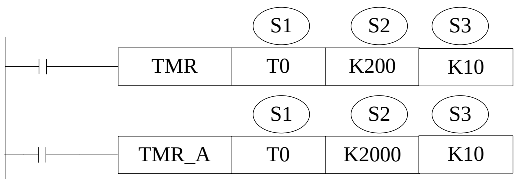
If X0 is ON, HT0 accumulates the 10ms pulse based on the current value. When the accumulation value reaches the set value K2000, the timer output activates.

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:

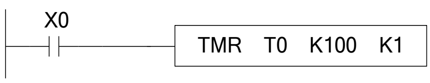
S1: timer (T0, HT10)

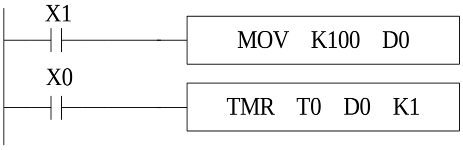
S2: set time (such as K100)

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

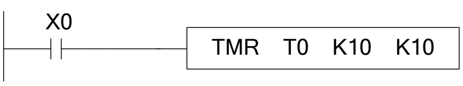


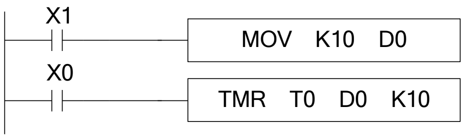


Set value is constant K

set value is register D

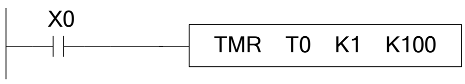
(2) Time unit is 10ms, set time is K10, the real time is 10ms*10=0.1s

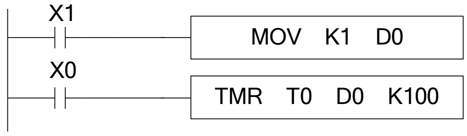




Set value is constant K set value is register D

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

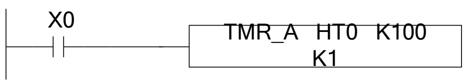


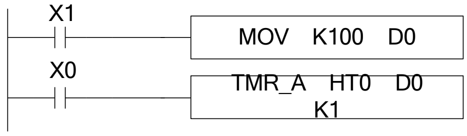


Set value is constant K set value is register D

Power-off retentive, accumulation

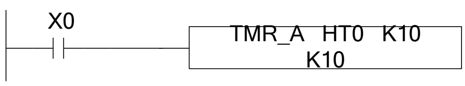
(1) Time unit is 1ms, set time is K100, the real time is 1ms *100=0.1s

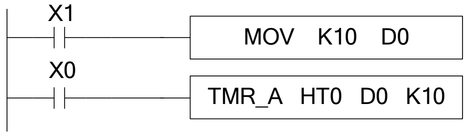




Set value is constant K set value is register D

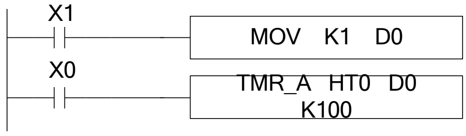
(2) Time unit is 10ms, set time is K10, the real time is 10ms*10=0.1s





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

The TMR is not accumulation timer instruction; TMR_A is accumulation timer instruction. The time unit includes K1, K10 and K100. Please don’t write other time unit otherwise the timer instruction will not run.

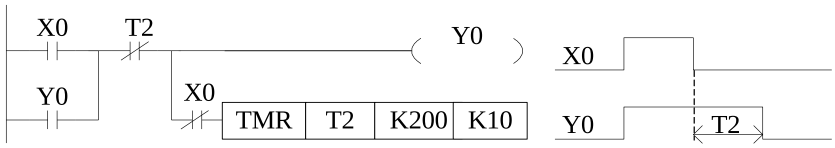
Time value

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.

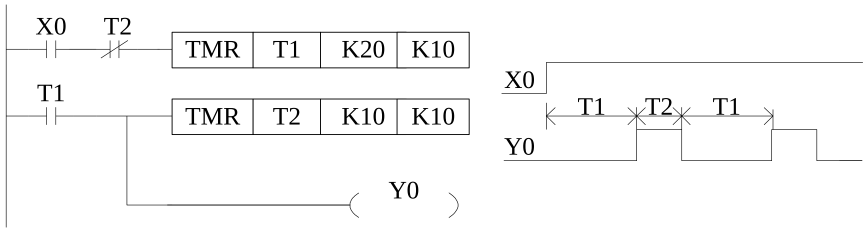
Application

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.

2-8. Counter (C, HC)

Number list

The counter addresses are in decimal; please see the following table for details:

Series	Name	Range		
		Normal	Power-off retentive	High speed counter
XD3	C HC HSC	C0~C575	HC0~HC95	HSC0~HSC31
XDM		C0~C4095	HC0~HC1023	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: XD3 series counters can be 16 or 32 bits count up/down mode. The mode is appointed by the instruction.

Counter features

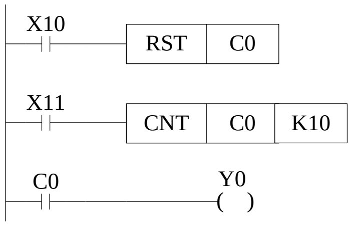
Item	16-bit counter	32-bit counter
Count direction	Count down/up	Count up/down
Set value	0~32,767	-2,147,483,648~+2,147,483,647
Set value type	Constant K or register	Constant K or a couple of registers
Count value	The value will not change when reaching the max or min value	The value will not change 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

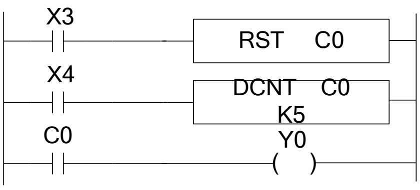
The set value range of 16-bit count-up counter is K1~K32,767 (decimal). K0 and K1 have the same function. They mean the counter output will act at the first counting.
If the PLC power supply is cut off, common counter value will be reset. The power-off retentive counter value will be kept.

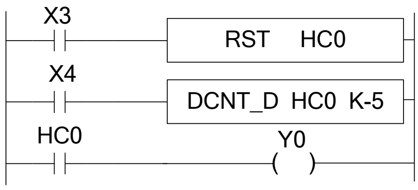


The counter C0 increases one when the X11 drives once. When C0 value reaches 10, the output acts. Then X11 drives again, C0 will continue increase one.
If X10 is ON, the C0 and output will be reset.
The counter set value can be constant K or register. For example, if D10 is 123, the set value is equal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

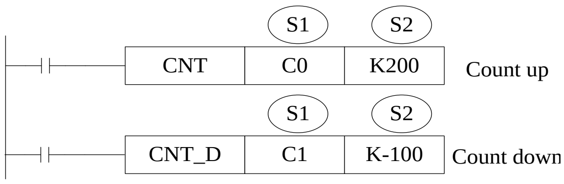
32-bit counter can be seemed to 32-bit register.

Counter set value

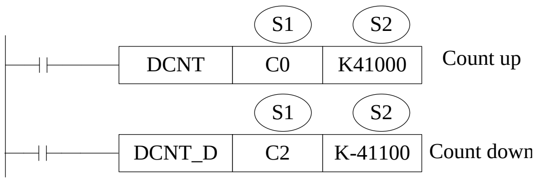
The set value contains two conditions: 16-bit and 32-bit. The counter types include common counter (C) and power-off retentive counter (HC).

Count instruction:

16-bit counter:



32-bit counter:



Reset instruction:

16-bit counter:

32-bit counter:

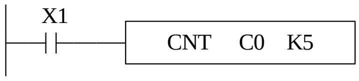
S1: counter (such as C0, HC10)

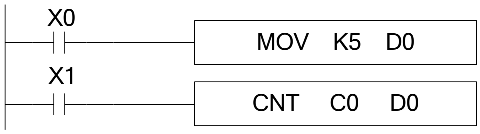
S2: counter set value (such as K100)

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)

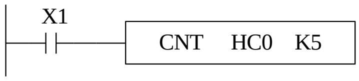
《set value is constant K》 《set value is register 》

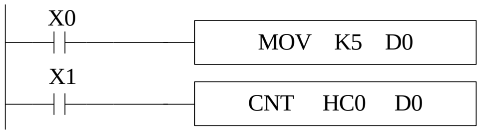




16-bit counter (power-off retentive, count up)

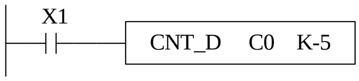
《set value is constant K》 《set value is register 》

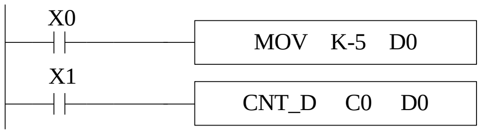




16-bit counter (common, count down)

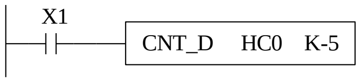
《set value is constant K》 《set value is register 》

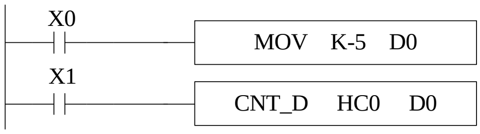




16-bit counter (power-off retentive, count down)

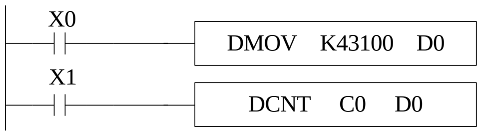
《set value is constant K》 《set value is register 》





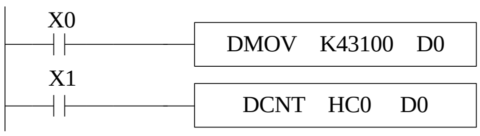
32-bit counter (common, count up)

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



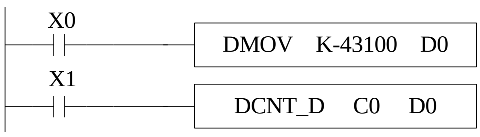
32-bit counter (power-off retentive, count up)

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



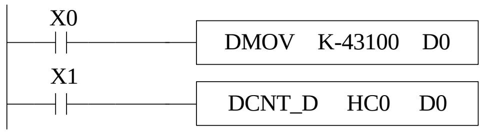
32-bit counter (common, count down)

《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 》



Count value

16-bit count up counter

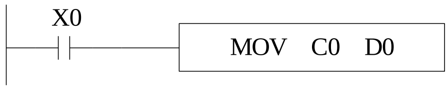
The count range is 0~32767.
When the count value reaches 32767, the counter stops working and keeps the state.

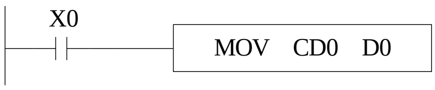
16-bit count down counter

The count range is -32768~0. When the count value reaches -32768, the counter stops working and keeps the state.

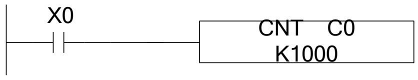
32-bit count up/down counter

The count range is -2,147,483,648 ~+2,147,483,647. When the count value reaches K2,147,483,647, it will become K-2,147,483,648. When the count value reaches K-2,147,483,648, it will become K2,147,483,647. The ON/OFF state of counter will change with the count value.





The two instructions have the same function. C0 is seemed to register in the first instruction.



The highest frequency of this instruction is related to the filter parameter and PLC scanning period. The max frequency it can count will be 500Hz. If the frequency is larger than 500Hz, please use high speed counter HSC0-HSC30.

High speed counter HSC0: the frequency input terminal is X0. The high speed counter will not be affected by input filter response delay time and PLC scanning period. Please refer to chapter 5 for details.

2-9. Data register (D, HD)

Address list

The data register of XD3/XDM series PLC is in decimal format. Please see the following table:

Series	Name	Range			
		Normal	Power-off retentive	Special	Special power-off retentive
XD3	D	D0~D7999	HD0~HD999	SD0~SD2047	HSD0~HSD499
XDM		D0~D20479	HD0~HD6143	SD0~SD4095	HSD0~HSD1023

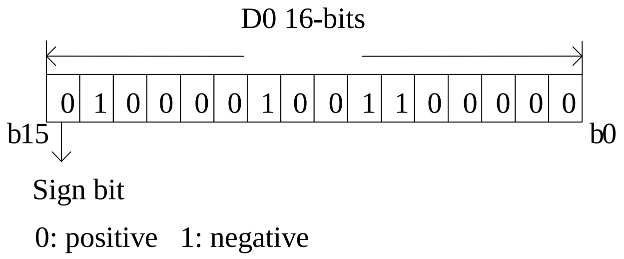
Structure

Data register is used to store data; it includes 16 bits(the higheset bit is sign bit) and 32 bits. (32 bits contains two registers, the highest bit is sign bit)

16 bits

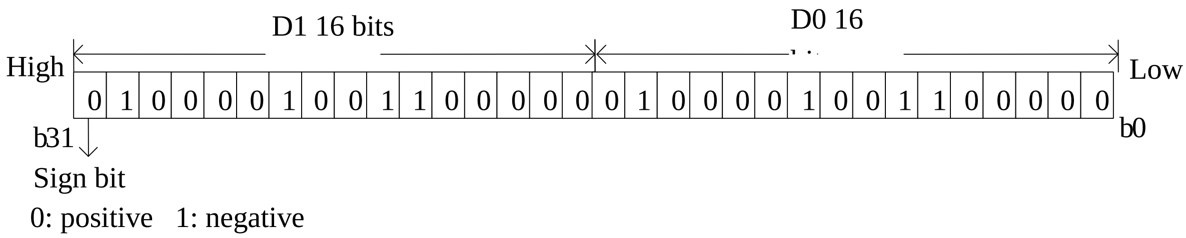
16-bits register range is -32,768 ~ +32,767

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



32 bits

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

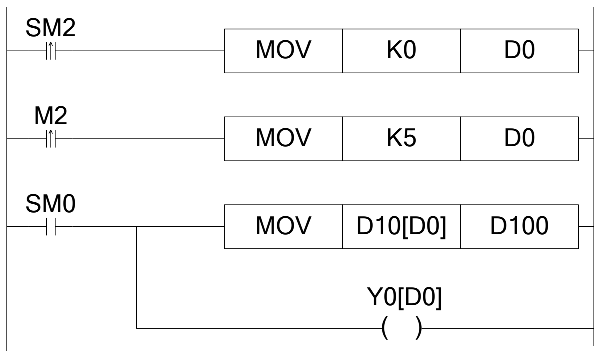
Used as offset (indirect appoint)

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;

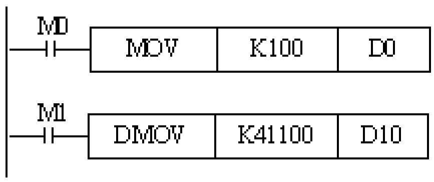
When M2 is from OFF→ON, D0=5, D100=D15, Y5 is ON.

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

Example

Data register D can deal with many kinds of data.

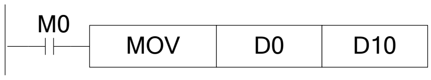
Data storage



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

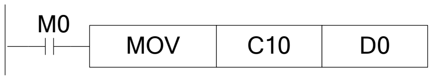
When M1 is ON, write 41100 into D11,D10 (32bits value)

Data transfer



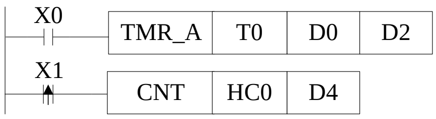
When M0 is ON, transfer the value of D10 to D0

Read the timer and counter



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

As the set value of timer and counter

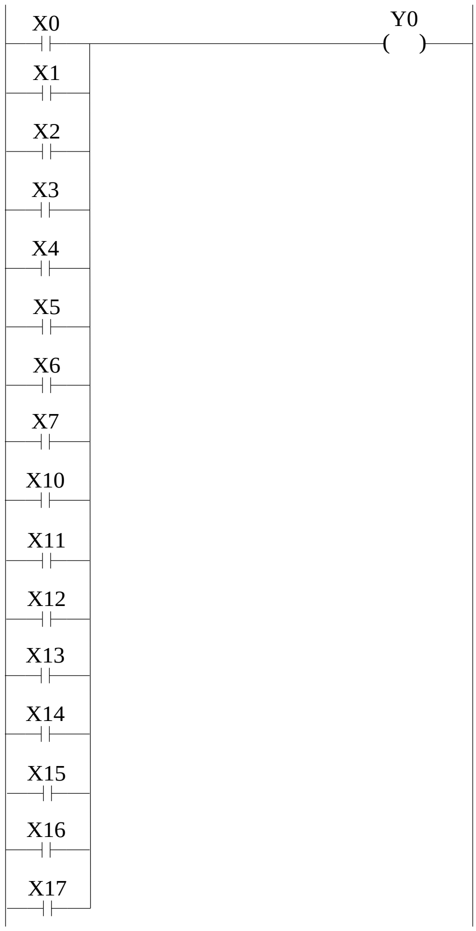


When X0 is ON, T10 starts to work, T0 will set ON when D0 value is equal to timer value, time unit is D2.

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

2-9-1． Word consist of bits

One of the coils from X0 to X17 is ON, Y0 will be ON.
Programming method one:

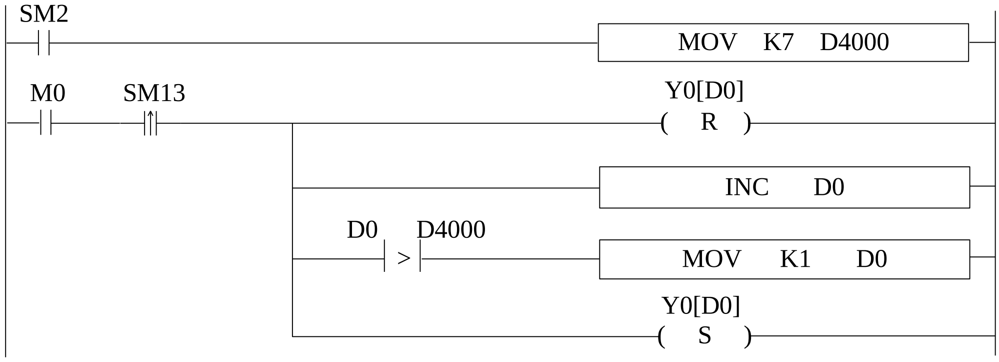


Programming method two: (application of word consists of bits)

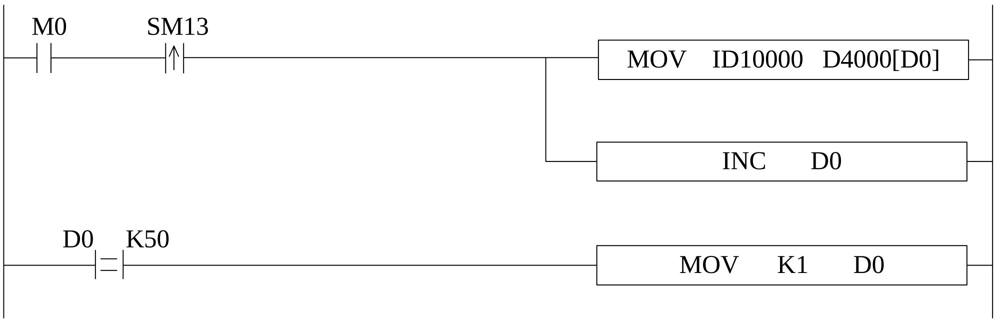
2-9-2． Offset application

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. Constant

Data process

XD3/XDM 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
XD3/XDM 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)
XD3/XDM 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.

2-11. 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
XD3, XDM	P	P0~P9999

Model	Sign	Address			
		External interruption			Timer interruption
		Input	Rising interruption	Falling interruption	
XD3-16	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	Sign	Address			
		External interruption			Timer interruption
		Input	Rising interruption	Falling interruption	
	I	X2	I0000	I0001	

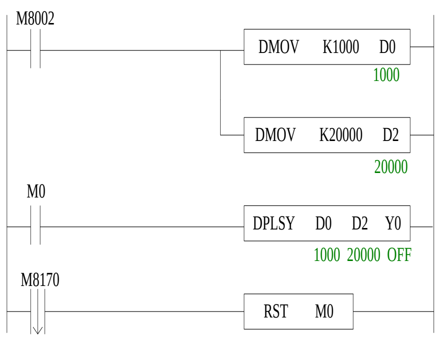
XD3-24/32/48/60		X3	I0100	I0101	There are 20 timer interruptions. From I40** to I59**. “**” means the timeof timer interruption, the unit is ms.
		X4	I0200	I0201	
		X5	I0300	I0301	
		X6	I0400	I0401	
		X7	I0500	I0501	
		X10	I0600	I0601	
		X11	I0700	I0701	
		X12	I0800	I0801	
		X13	I0900	I0901	

Model	Name	Range			
		External interruption			Timing interruption
		Input	Rising interruption	Falling interruption	
XDM-24/32/60T4/60T10	I	X2	I0000	I0001	There are 20 timer interruptions. From I40** to I59**. “**” means the time of 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

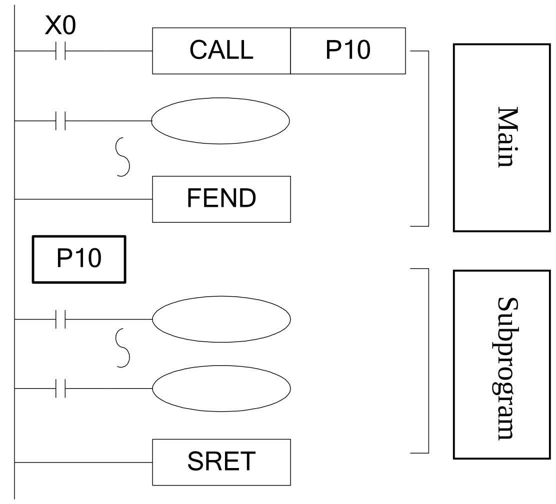
P is usually used in flow; it is used together with CJ (condition jump), CALL (call subprogram), etc.

Condition Jump CJ



If coil X0 is ON, jump to the program after 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

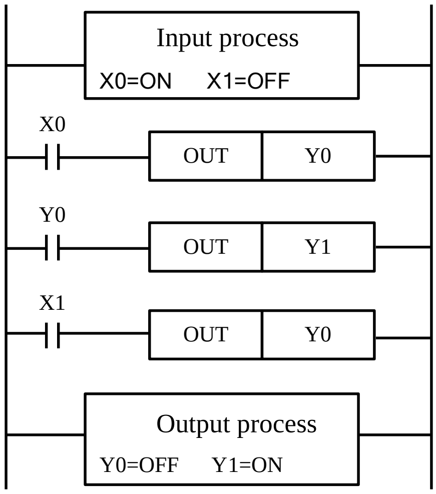
Tag I is usually used in interruption, including external interruption, time interruption etc. It often works together with IRET (interruption return), EI (enable interruption), DI (disable interruption);
External interruption
Accept the input signal from the special input terminals, not affected by 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 is different 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.

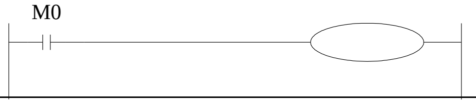
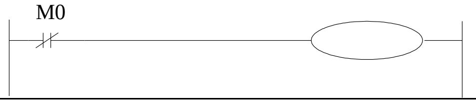
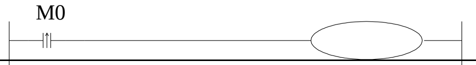
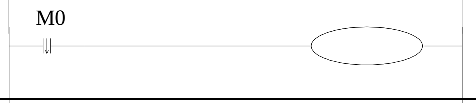
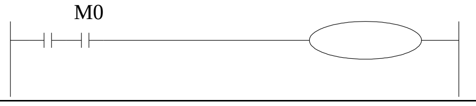
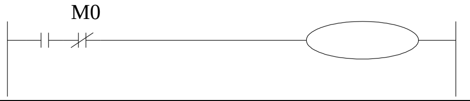
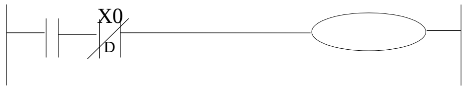
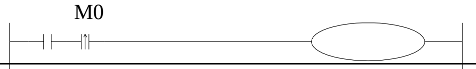
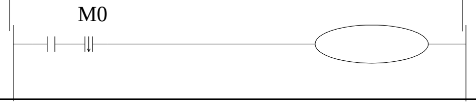
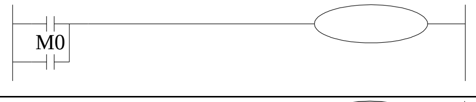
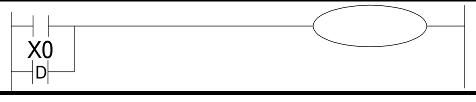
When executing dual output (use dual coil), the after one is act in priority.

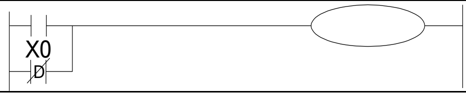
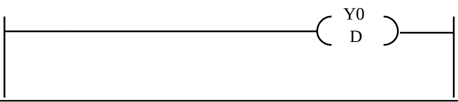
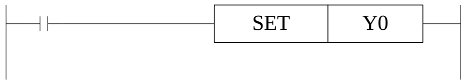
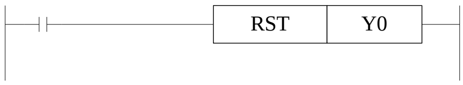
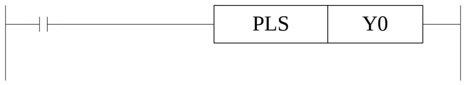
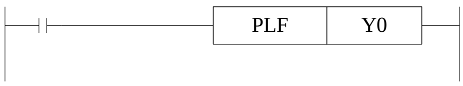
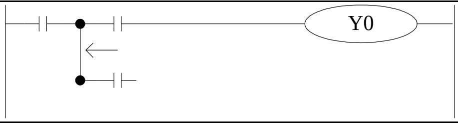
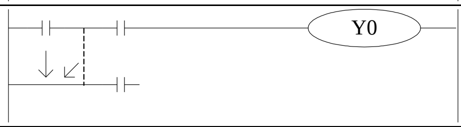
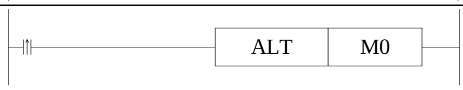
3 Basic Program Instructions

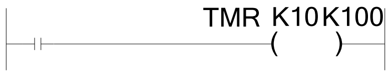
This chapter introduces the basic instructions and their functions.

3-1. Basic Instructions List

XD3, XDM series support all the basic instructions:

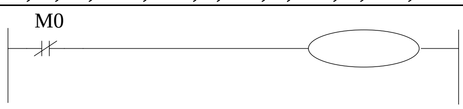
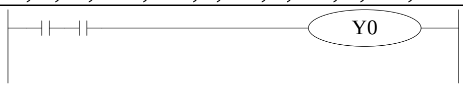
Mnemonic	Function	Format and Device	**Chapter**
LD	Initial logical operation contact type NO (normally open)		3-2
LDD	Read the status from the contact directly		3-6
LDI	Initial logical operation contact type NC (normally closed)		3-2
LDDI	Read the normally closed contact directly		3-6
LDP	Initial logical 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) contacts		3-4
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
PLS	Rising edge pulse		3-11
PLF	Falling/trailing edge pulse		3-11
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
END	Force the current program scan to end		3-14
GROUP	Group		3-15

GROUPE	Group End		3-15
TMR	Time		2-7

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
LDI (negative)	Initial logic operation contact type NC (Normally Closed)	 Devices: X,Y,M,HM,SM,S,HS,T,HT,C,HC,Dn.m
OUT (OUT)	Final logic operation type drive coil	 Operands: X,Y,M,HM,SM,S,HS,T,HT,C,HC,Dn.m

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.

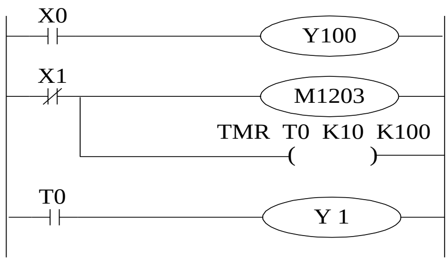
OUT instruction can drive the output relays, auxiliary relays, status, timers, and counters. But this instruction can't be used for the input relays

For coil T and C, please set constant K or register D when using OUT.

The following table shows the constant K setting range, actual timer constant, program step relative to OUT instruction (include the setting value).

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

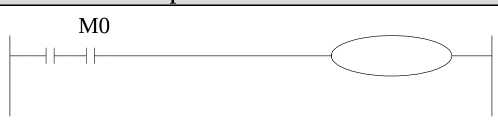
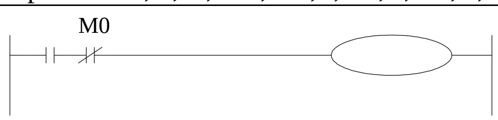
Program




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

3-3. [AND] , [ANI]

Mnemonic and Function

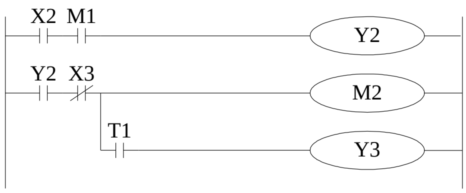
Mnemonic	Function	Format and Operands
AND (and)	Normal open contactor in series	 Operand: X,Y,M,HM,SM,S,HS,T,HT,C,HC,Dn.m
ANI (and reverse)	Normal close contactor in series	 Operand: X,Y,M,HM,SM,S,HS,T,HT,C,HC,Dn.m

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 output can repeat as long as the output order is correct. There’s no limit for the serial connected contactors and follow-on output times.

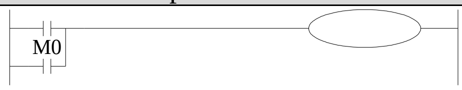
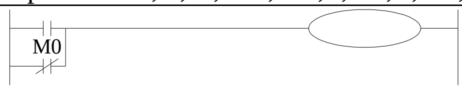
Program



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

3-4. [OR] , [ORI]

Mnemonic and Function

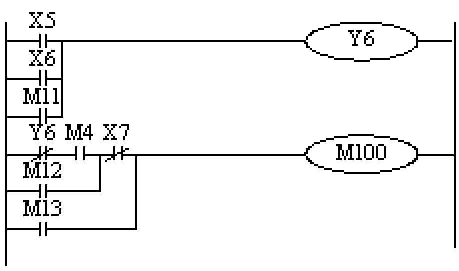
Mnemonic	Function	Format and Operands
OR (OR)	Parallel connection of NO (Normally Open) contactors	 Operand: X,Y,M,HM,SM,S,HS,T,HT,C,HC,Dn.m
ORI (OR reverse)	Parallel connection of NC (Normally Closed) contactors	 Operand: X,Y,M,HM,SM,S,HS,T,HT,C,HC,Dn.m

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;

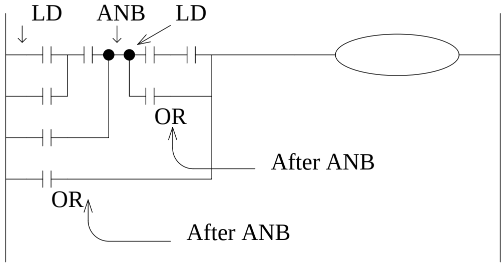
OR and ORI start from the instruction step, parallel connect with the LD and LDI instruction step introduced before. There is no limit for the parallel connect times.

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]

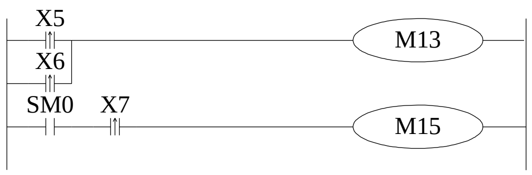
Mnemonic and Function

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

Statements

LDP, ANDP, ORP will be ON for one scanning period when the signal rising pulse is coming (OFF→ON)
LDF, ANDF, ORF will be ON for one scanning period when the signal falling pulse is coming (ON→OFF)

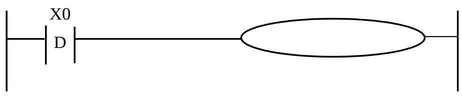
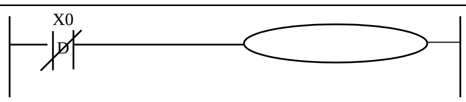
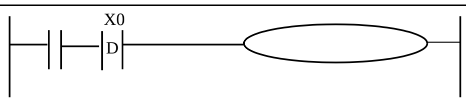
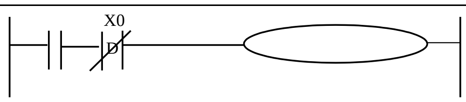
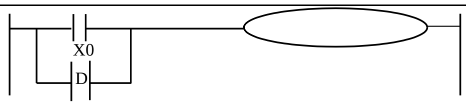
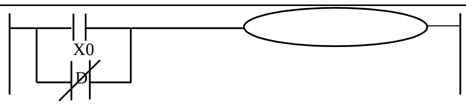
Program

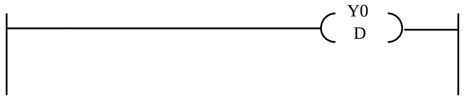



```
LDP     X5
ORP     X6
OUT     M13
LD      M8000
ANDP    X7
OUT     M15
```

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

Mnemonic and Function

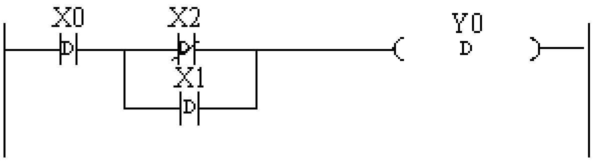
Mnemonic	Function	Format and Operands
LDD	Read the status from the contact directly	 Devices: X
LDDI	Read the normally closed contact directly	 Devices: X
ANDD	Read the status from the contact directly	 Devices: X
ANDDI	Read the normally closed contact directly	 Devices: X
ORD	Read the status from the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

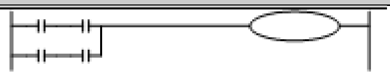
Program



LDD	X0
LDDI	X2
ORD	X2
ANB	
OUTD	Y0

3-7. [ORB]

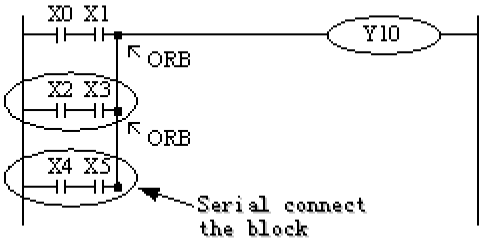
Mnemonic and Function

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

Statements

Two or more contactors is 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:

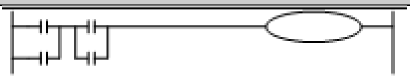
LD	X0
AND	X1
LD	X2
AND	X3
ORB	
LD	X4
AND	X5
ORB	
OUT	Y10

Non-preferred programming method:

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

3-8. [ANB]

Mnemonic and Function

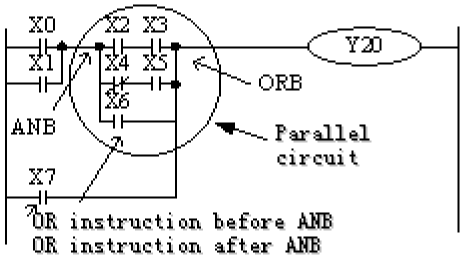
Mnemonic	Function	Format and Devices
ANB (And Block)	Serial connection of parallel circuits	 Devices: none

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.

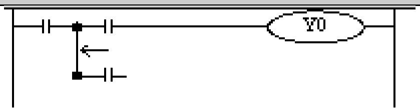
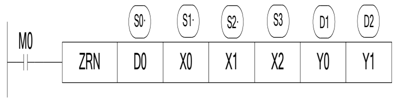
Program



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

3-9. [MCS] , [MCR]

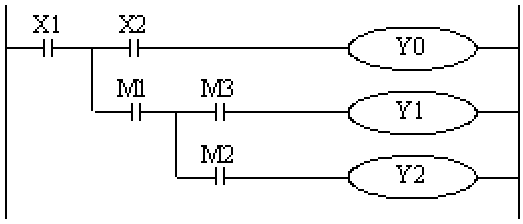
Mnemonic and Function

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.

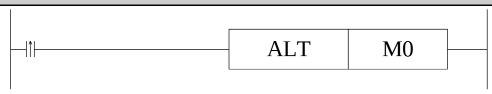
Program



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

3-10. [ALT]

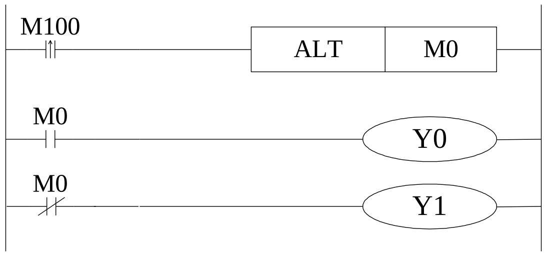
Mnemonic and Function

Mnemonic	Function	Format and Devices
ALT (Alternate)	Alternate the coil	 Coil: X,Y,M,HM,SM,S,HS,T,HT,C,HC,Dn.m

Statements

The status of the coil is reversed after using ALT (ON to OFF, OFF to ON).

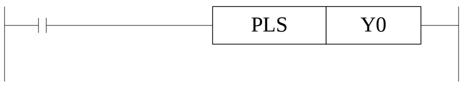
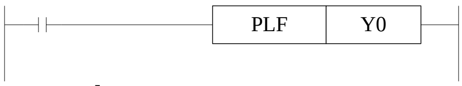
Program



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

3-11. [PLS] , [PLF]

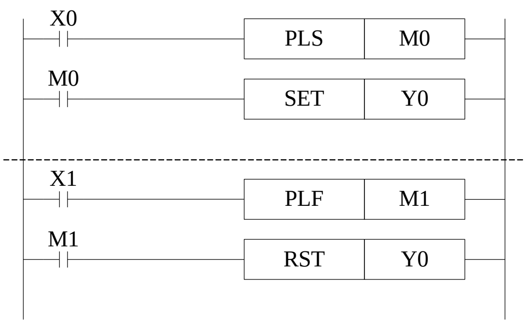
Mnemonic and Function

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

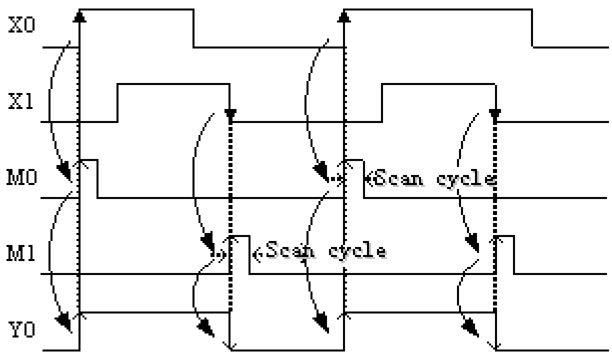
Statements

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

Program




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



3-12. [SET], [RST]

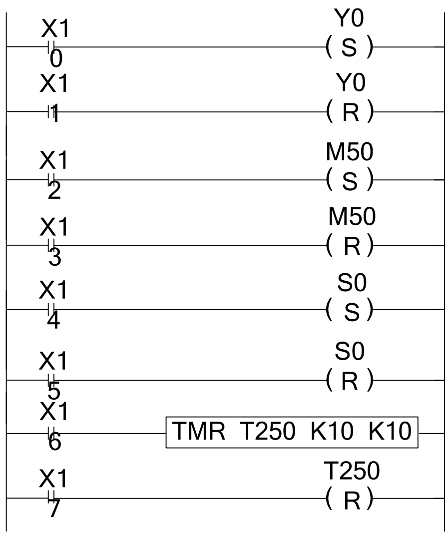
Mnemonic and Function

Mnemonic	Function	Format and Devices
SET (Set)	Set a bit device permanently ON	SET Y0 Operand: X,Y,M,HM,SM,S,HS,T,HT,C,HC,Dn.m
RST (Reset)	Reset a bit device permanently OFF	RST Y0 Operand: X,Y,M,HM,SM,S,HS,T,HT,C,HC,Dn.m

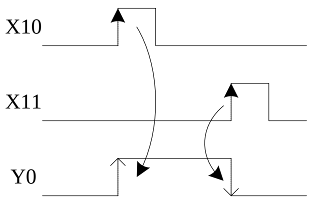
Statements

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
```

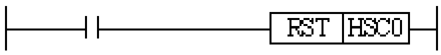


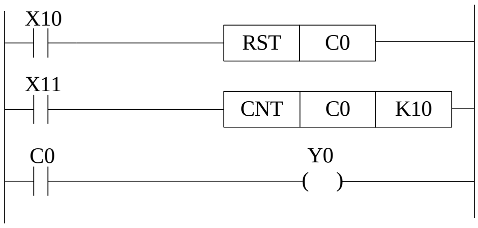
3-13. 【CNT】【CNT_D】【DCNT】【DCNT_D】【RST】for the counters

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

Internal counter programming

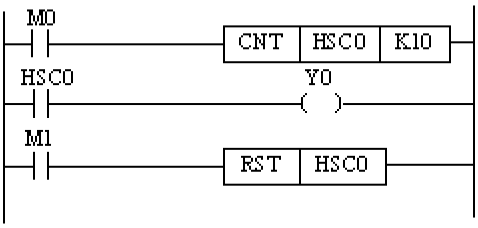
RST Reset	Reset the output coil, clear the current count value	 Operand: C, HC, HSC



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.

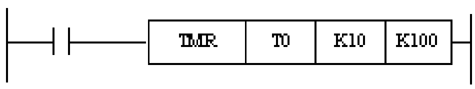
High speed counter programming

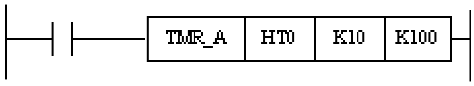
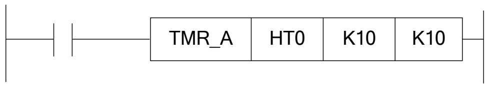


Increase count the OFF to ON times of M0.
When the count value reaches set value (value of K or D), the count coil will be ON.
When M1 is ON, the count coil of HSC0 reset, the current value becomes 0.

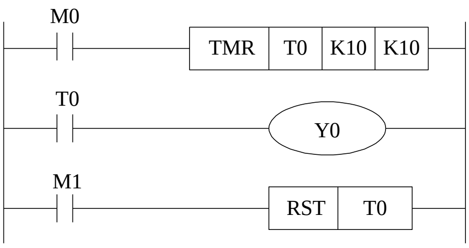
3-14. [TMR], [TMR-A] for timers

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

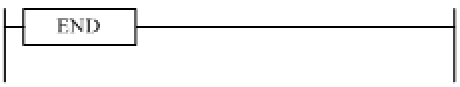


When M0 is ON, T0 starts to timing. When T0 reaches K10, T0 coil is ON. Then T0 continues timing. When M1 is ON, reset the T0.

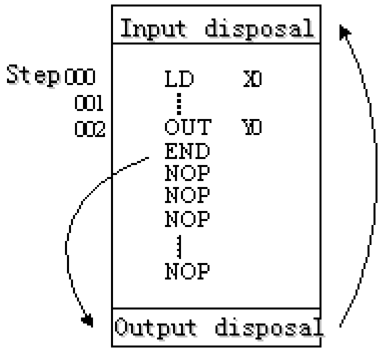
Power-off retentive timer will keep the current value and counter coil status when the power is off.

3-15. [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.)

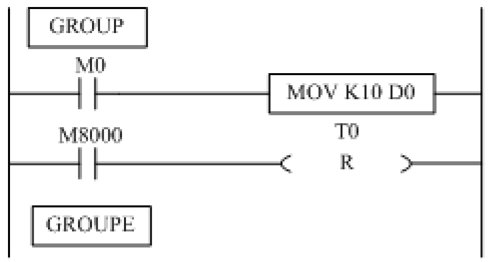
3-16. [GROUP] , [GROUPE]

Mnemonic and Function

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

Statements

GROUP and GROUPE should used in pairs.
GROUP and GROUPE don't have practical meaning; they are used to optimize the program structure. So, add or delete these instructions doesn't 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.

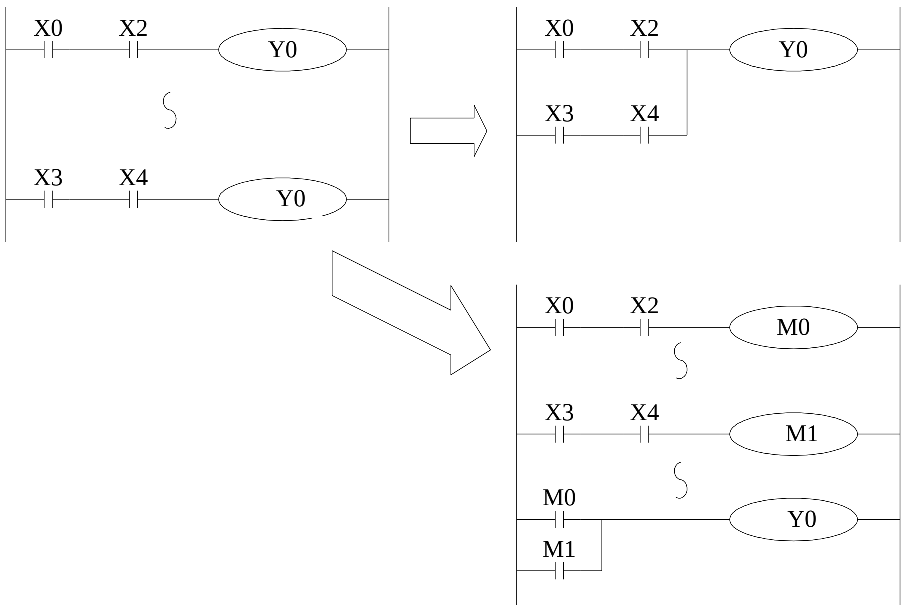
Program's executing sequence

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 XD3, XDM 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
FLT	16 bits integer converts to float point	FLT S D	4-8-2
DFLT	32 bits integer converts to float point	DFLT S D	4-8-2
FLTD	64 bits integer converts to float point	FLTD S D	4-8-2
INT	Float point converts to integer	INT S D	4-8-3
BIN	BCD converts to binary	BIN S D	4-8-4
BCD	Binary converts to BCD	BCD S D	4-8-5

ASCI	Hex. converts to ASCII	ASCI S D n	4-8-6
HEX	ASCII converts to Hex.	HEX S D n	4-8-7
DECO	Coding	DECO S D n	4-8-8
ENCO	High bit coding	ENCO S D n	4-8-9
ENCOL	Low bit coding	ENCOL S D n	4-8-10
GRY	Binary to Gray code	GRY S D	4-8-11
GBIN	Gray code to binary	GBIN S D	4-8-12
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

4-2. Reading Method of Applied Instructions

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

Summary

ADDITION [ADD]			
16 bits	ADD	32 bits	DADD
Execution condition	Normally ON/OFF, Rising/Falling edge	Suitable Models	XD3, XDM
Hardware requirement	-	Software requirement	-

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

Suitable Soft Components

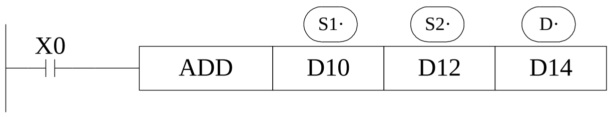
Word

Operand	System								Constant	Module	
	D*	FD	TD*	CD*	DX	DY	DM*	DS*	K/H	ID	QD
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 on 32 bits operation, low 16 bits of 32-bit register are assigned, the register address close to the low 16 bits register will be assigned to high 16 bits of 32-bit register. Even number is recommended for the low 16 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.

Related flag

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

Notes

The assignment of the data
The data register of XD3 series PLC is a single word (16 bit) 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 bit) 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 represent 32 bits instruction
Add letter "D" before 16 bits instruction to represent 32 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]

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	XD3, XDM
Hardware requirement	-	Software requirement	-

Operands

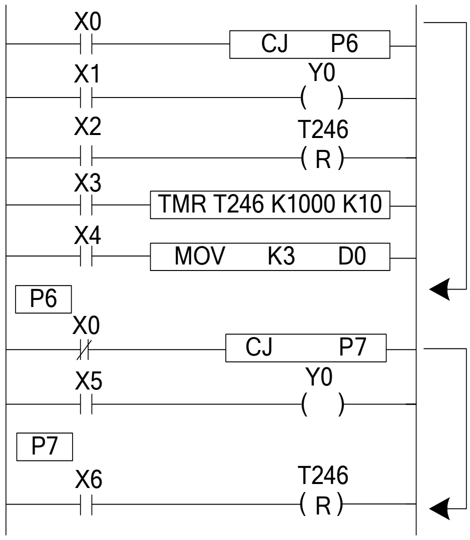
Operands	Function	Data Type
Pn	Jump to the target (with pointer Nr.) P (P0~P9999)	Pointer's Nr.

Suitable Soft Components

	Pointer	
Other	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]

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	XD3, XDM
Hardware requirement	-	Software requirement	-
Subroutine Return [SRET]			
16 bits	SRET	32 bits	-
Execution condition	-	Suitable Models	XD3, XDM
Hardware requirement	-	Software requirement	-

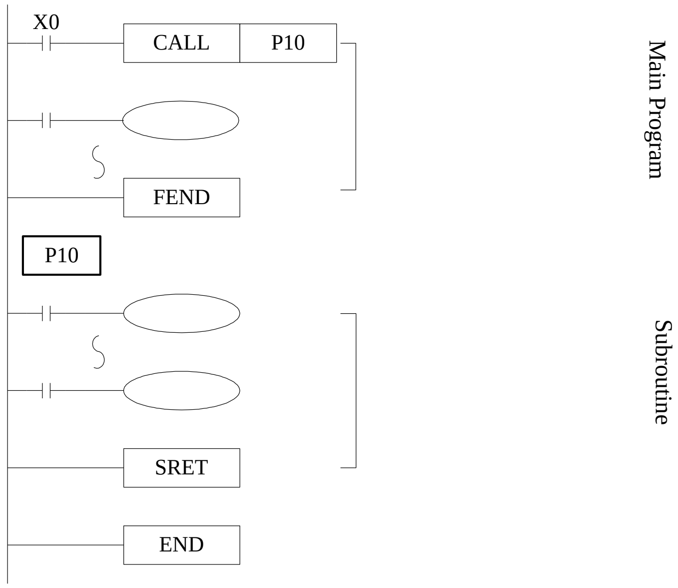
Operands

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

Suitable Soft Components

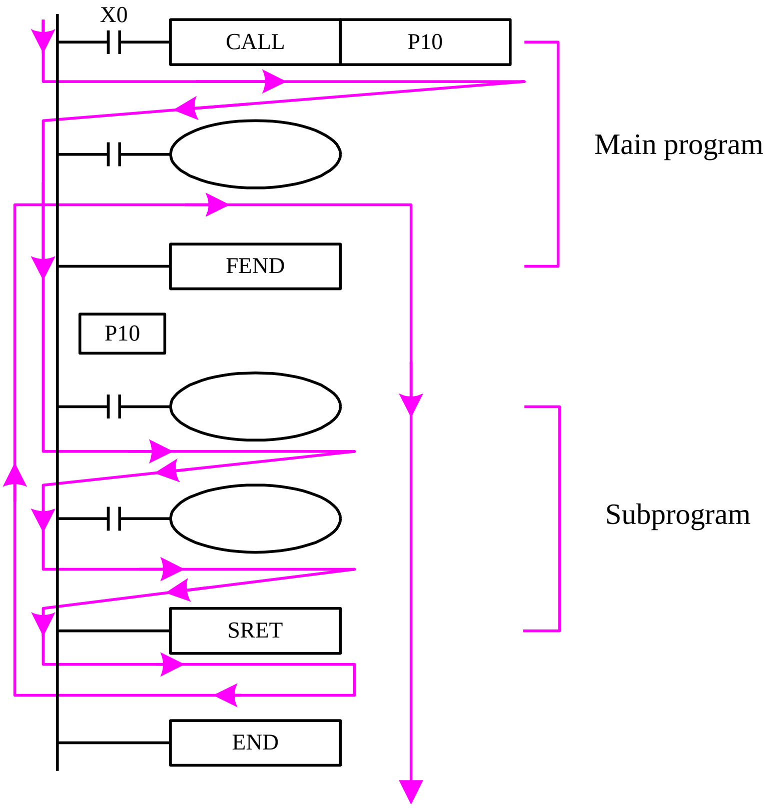
Others	Pointer	
	P	I
	●	

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.

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]

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	XD3, XDM
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	XD3, XDM
Hardware requirement	-	Software requirement	-
Flow starts [STL]			
16 bits	STL	32 bits	-

Execution condition	-	Suitable Models	XD3, XDM
Hardware requirement	-	Software requirement	-
Flow ends [STLE]			
16 bits	STLE	32 bits	-
Execution condition	-	Suitable Models	XD3, XDM
Hardware requirement	-	Software requirement	-

operands

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

3.Suitable Soft Components

Bit

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

*Note: M includes M, HM and SM; S includes S, HS; T includes T and HT; C includes C and 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.

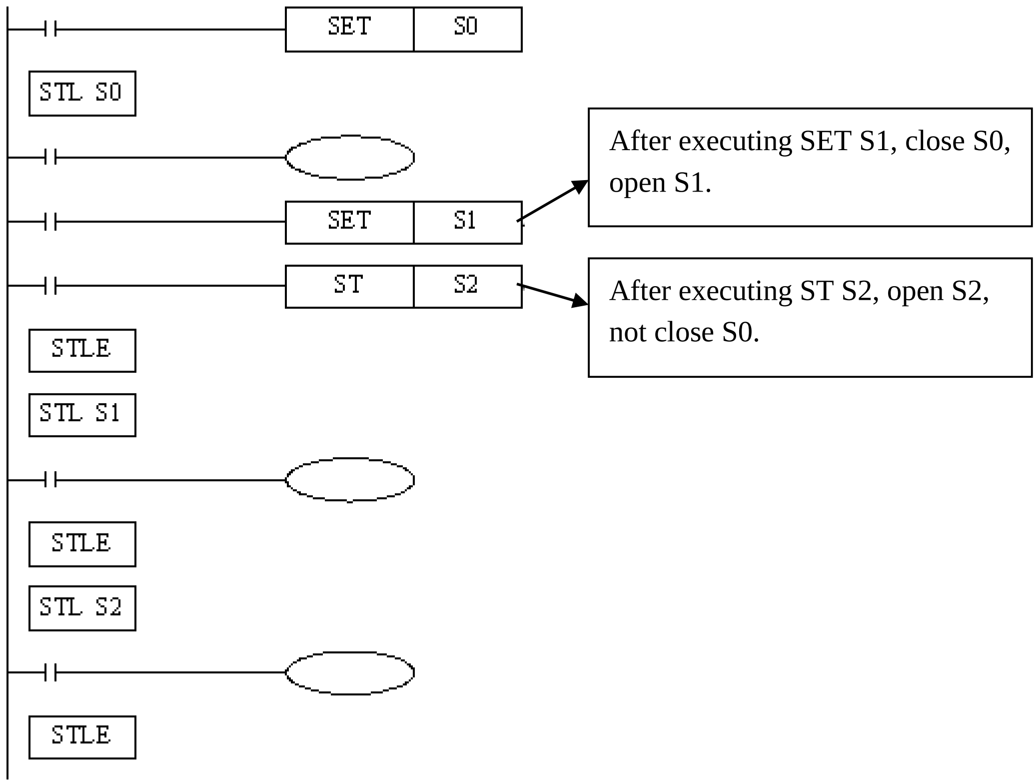
In flow S0, SET S1 close the current flow S0, open flow S1.

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.

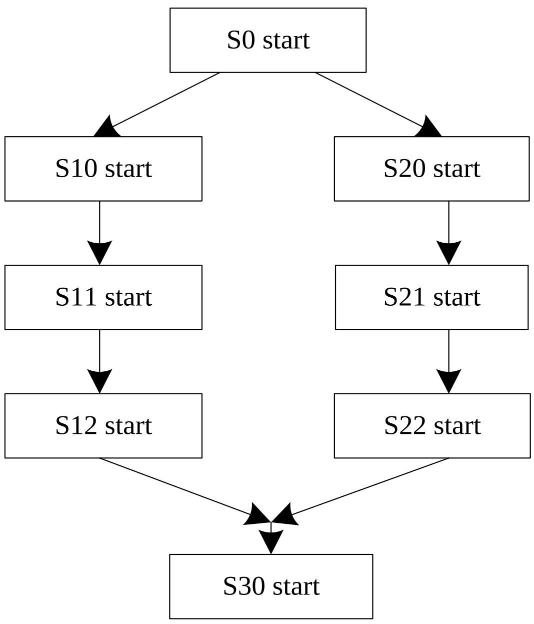
After executing **SET Sxxx** instruction and jump to the next flow, the pulse instructions in the former flow will be closed. (including one-segment, multi-segment, relative or absolute, return to the origin)

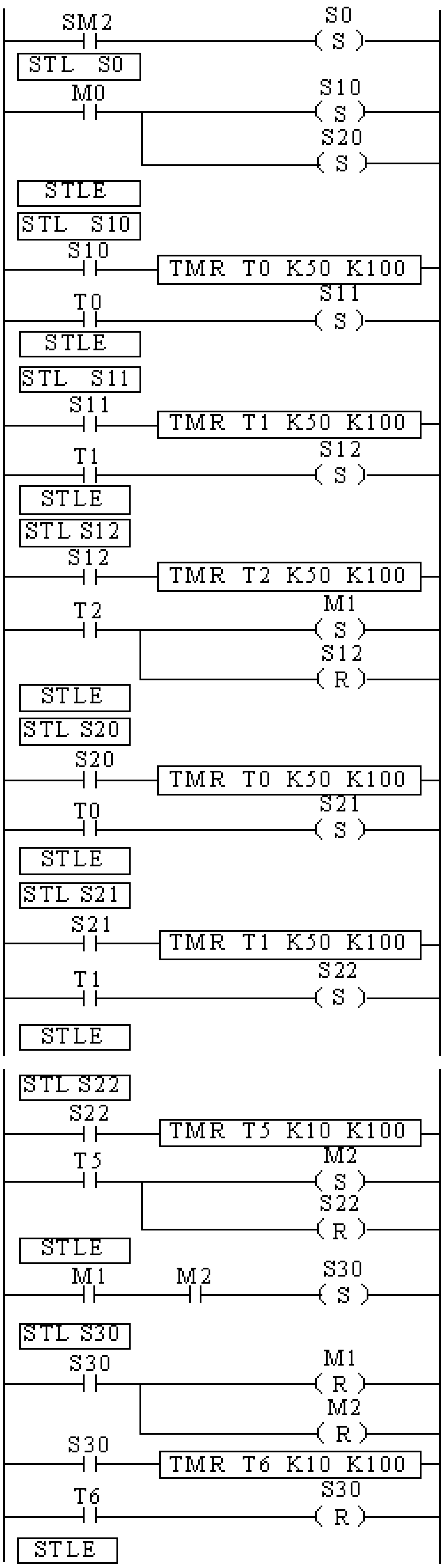


Example

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

Program diagram:





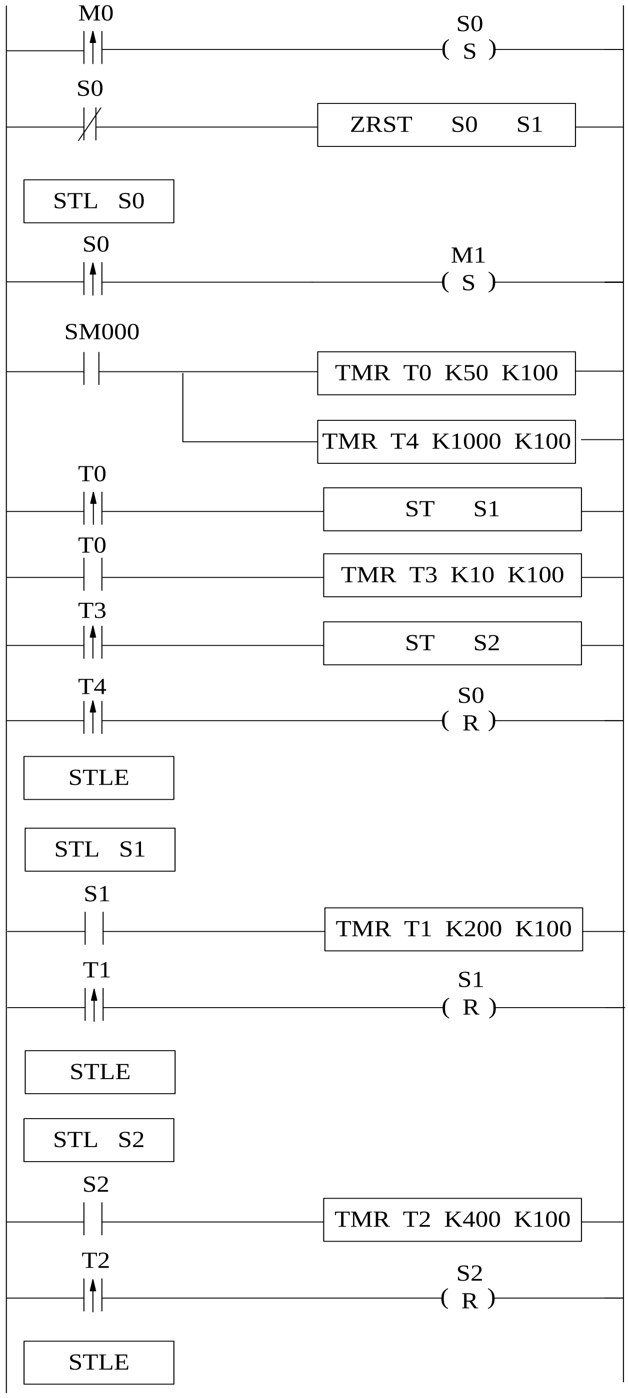
The program explanation:
When SM2 is ON, set ON flow S0. When M0 is ON, set ON flow S10 and S20.

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]

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	XD3, XDM
Hardware requirement	-	Software requirement	-
Loop ends [NEXT]			
16 bits	NEXT	32 bits	-
Execution condition	Normally ON/OFF, Rising/Falling edge	Suitable Models	XD3, XDM
Hardware requirement	-	Software requirement	-

Operands

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

Suitable Soft Components

Word

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

*Notes: D includes D, HD; TD includes TD, HTD; CD includes CD, HCD, HSCD, HSD; DM includes DM, DHM; DS includes DS, DHS.

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.

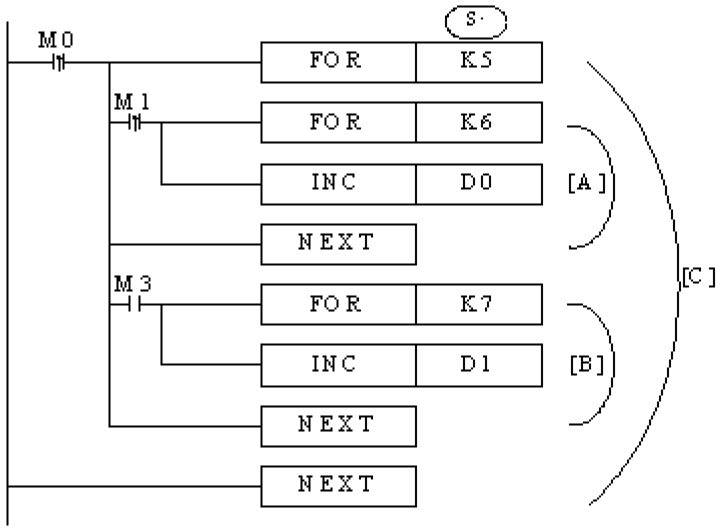
Between FOR and NEXT, LDP, LDF instructions are effective for one time. Every time when M0 turns from OFF to ON, and M1 turns from OFF to ON, [A] loop is executed 6 times.

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

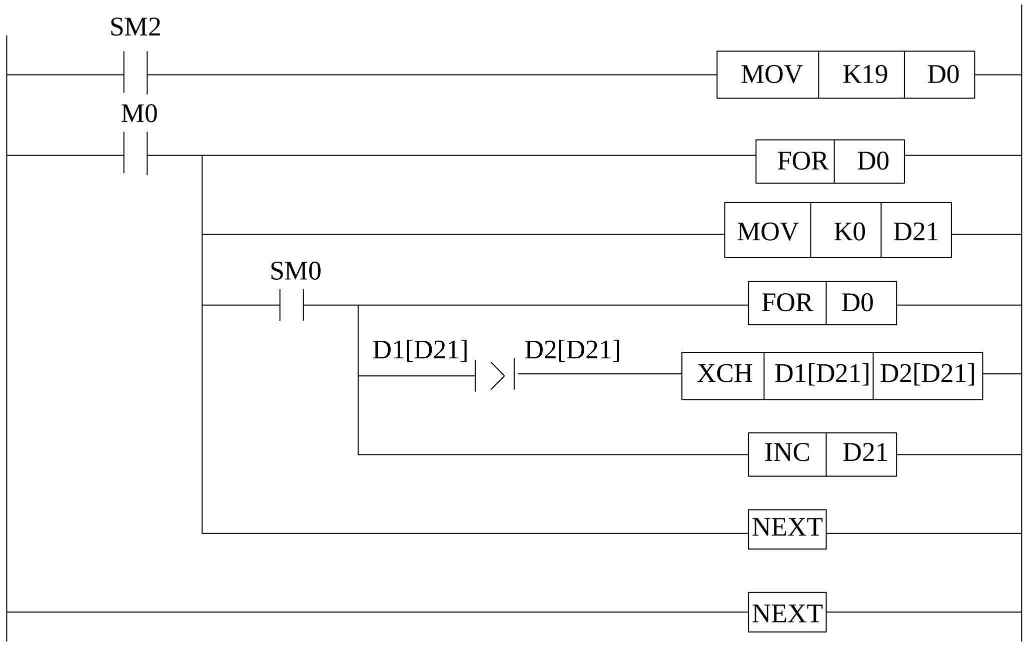
If there are many loop times, the scan cycle will be prolonged. Monitor timer error may occur, please note this.

If NEXT is before FOR, or no NEXT, or NEXT is behind 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]

Summary

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

main program ends [FEND]			
Execution condition	-	Suitable Models	XD3, XDM
Hardware requirement	-	Software requirement	-
program ends [END]			
Execution condition	-	Suitable Models	XD3, XDM
Hardware requirement	-	Software requirement	-

Operands

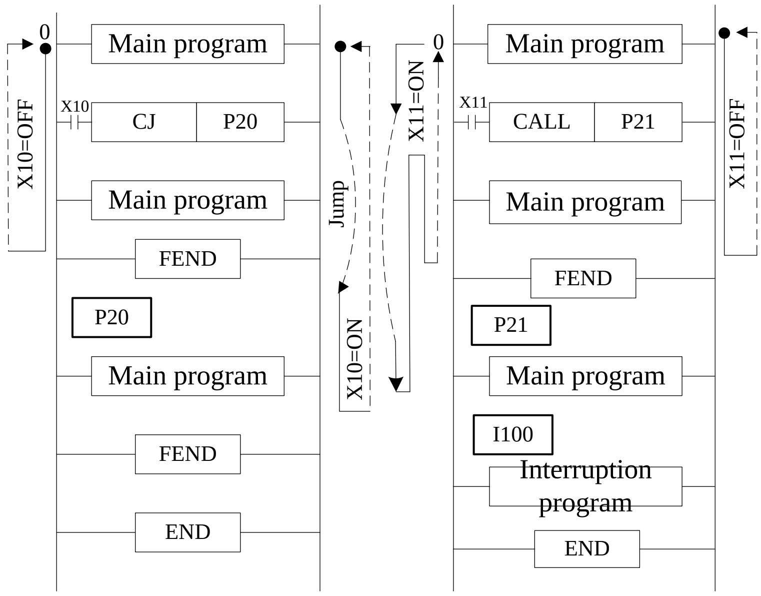
Operands	Function	Data Type
None	-	-

Suitable Soft Components

None

Description

Even though [FEND] instruction represents the end of the main program, the function is same to END to process 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]

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	XD3, XDM
Hardware requirement	-	Software requirement	-

2. Operands

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

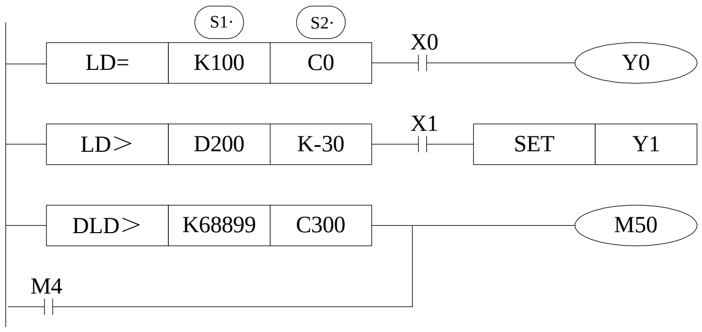
3. Suitable soft components

	Operand	System								Constant	Module	
Word		D*	FD	TD*	CD*	DX	DY	DM*	DS*	K/H	ID	QD
	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.

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 using 16 bits instruction, the program or operation will be error.

4-4-2. Serial Compare [AND]

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	XD3, XDM
Hardware requirement	-	Software requirement	-

Operands

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

suitable soft components

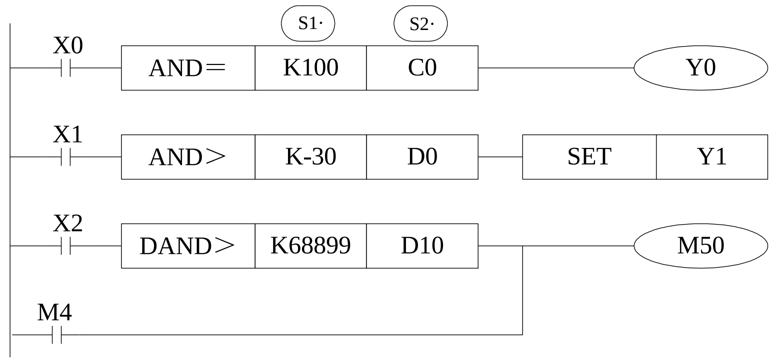
Word

Operand	System								Constant	Module	
	D*	FD	TD*	CD*	DX	DY	DM*	DS*	K/H	ID	QD
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.

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 using 16 bits instruction, the program or operation will be error.

4-4-3. 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	XD3, XDM
Hardware requirement	-	Software requirement	-

2. Operands

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

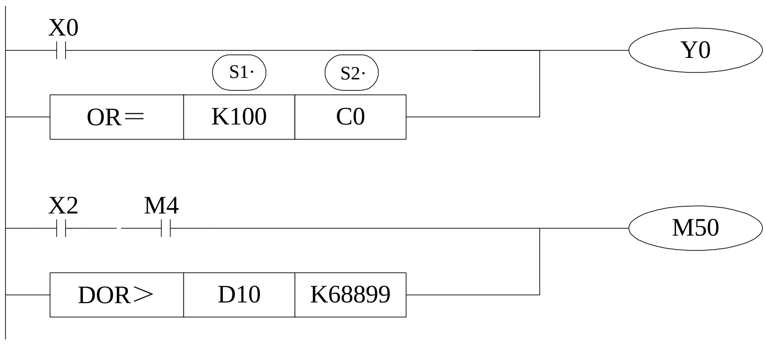
3. Suitable soft components

Word	Operand	System								Constant	Module	
		D*	FD	TD*	CD*	DX	DY	DM*	DS*	K/H	ID	QD
	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.

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)

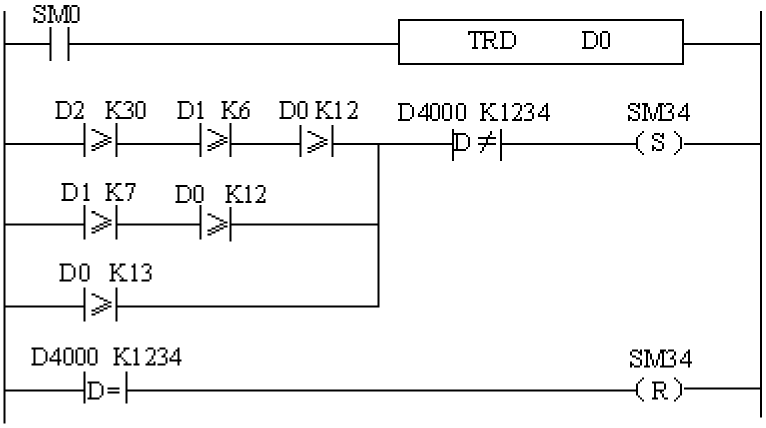


Note Items

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 using 16 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
```

4-5. 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]

1. Summary

Compare the two data, output the result.

Data compare [CMP]			
16 bits	CMP	32 bits	DCMP
Execution condition	Normally ON/OFF, rising/falling edge	Suitable Models	XD3, XDM
Hardware requirement	-	Software requirement	-

2. Operands

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

3. Suitable soft component

Word

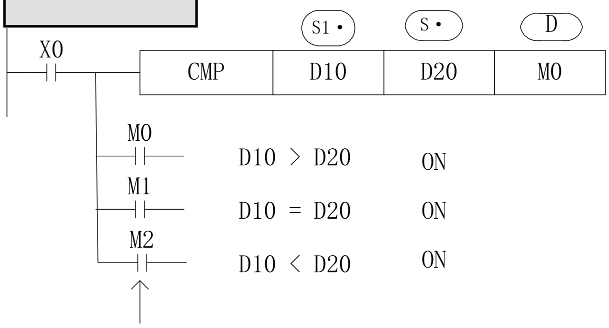
Operand	System								Constant	Module	
	D*	FD	TD*	CD*	DX	DY	DM*	DS*	K/H	ID	QD
S1	●	●	●	●	●	●	●	●	●		
S	●	●	●	●	●	●	●	●	●		

Bit

Operand	System						
	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



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.

4-5-2. Data zone compare [ZCP]

1. Summary

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

Data Zone compare [ZCP]			
16 bits	ZCP	32 bits	DZCP
Execution condition	Normally ON/OFF, rising/falling edge	Suitable Models	XD3, XDM
Hardware requirement	-	Software requirement	-

2. Operands

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

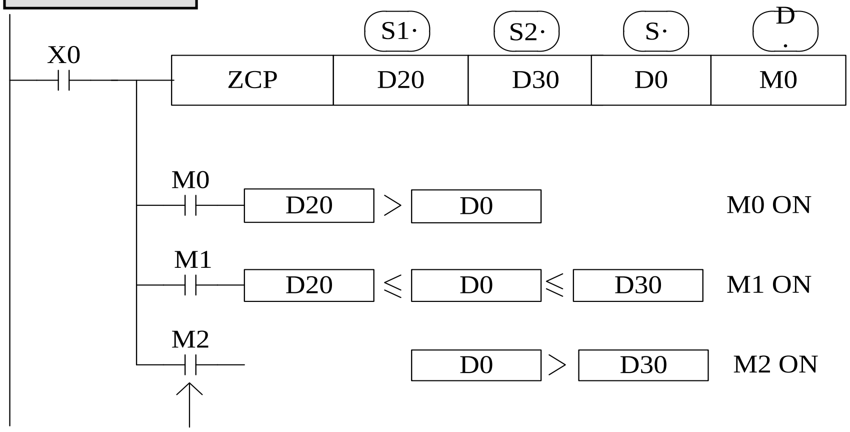
3. Suitable soft components

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

	Operand	System						
		X	Y	M*	S*	T*	C*	Dn.m
Bit	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]

1. Summary

Move the specified data to the other soft components

MOV [MOV]			
16 bits	MOV	32 bits	DMOV
Execution condition	Normally ON/OFF, rising/falling edge	Suitable Models	XD3, XDM
Hardware requirement	-	Software requirement	-

2. Operands

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

3. Suitable soft component

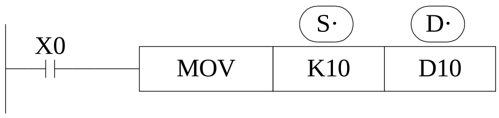
Word

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

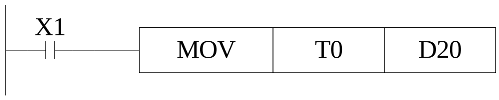
*Notes: D includes D, HD; TD includes TD, HTD; CD includes CD, HCD, HSCD, HSD; DM includes DM, DHM; DS includes DS, DHS.

Description

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

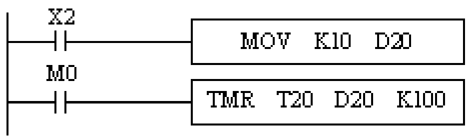


<read the counter or timer current value>



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

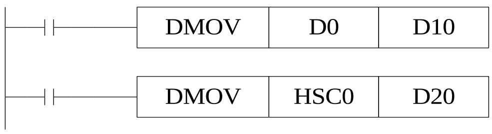
<indirect set the timer value>



(K10) (D20)
D20=K10

< Move the 32bits data >

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



(D1, D0) → (D11, D10)
(the current value of HSC0) → (D21, D20)

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	XD3, XDM
Hardware requirement	-	Software requirement	-

2. Operands

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

3. Suitable soft components

Word

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

Bit

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

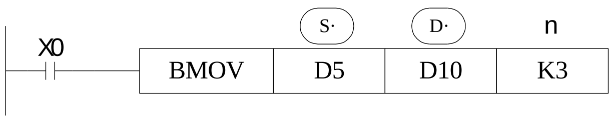
*Notes: D includes D, HD; TD includes TD, HTD; CD includes CD, HCD, HSCD, HSD; DM includes DM, DHM; DS includes DS, DHS.

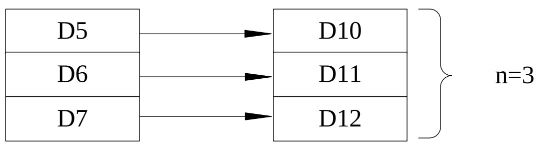
M includes M, HM, SM; S includes S, HS; T includes T and HT; C includes C and HC.

Description

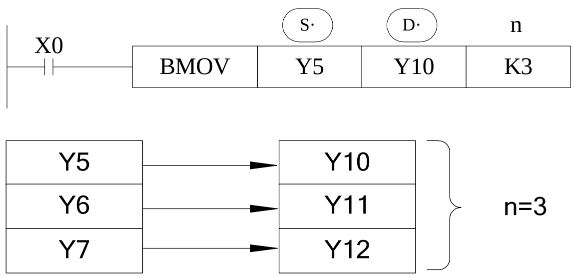
Move the source data block to the target data block. The data quantity is n.

<word move>

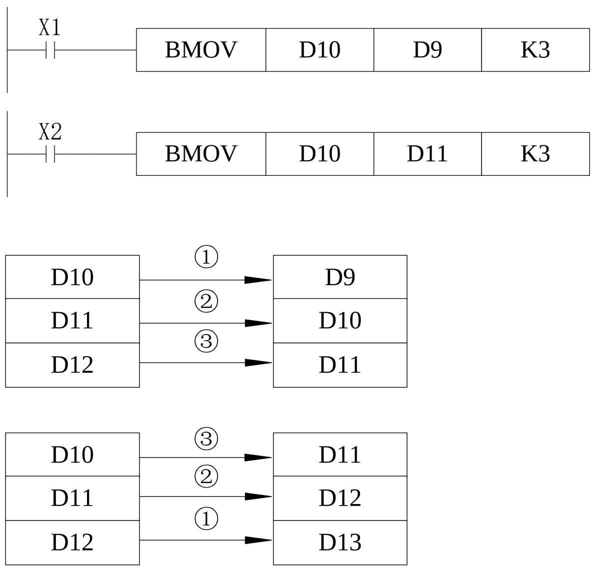




<bit move>



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



4-5-5. Data block Move [PMOV]

1. Summary

Move the specified data block to the other soft components

Data block mov[PMOV]			
16 bits	PMOV	32 bits	-
Execution condition	Normally ON/OFF coil, rising/falling edge	Suitable Models	XD3, XDM
Hardware requirement	-	Software requirement	-

2. Operands

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

3. Suitable soft components

Word

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

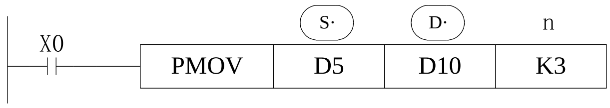
Bit

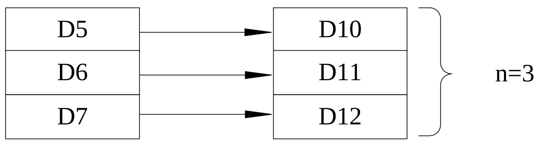
Operand	System						
	X	Y	M*	S*	T*	C	Dn.m
S	●	●	●				
D	●	●	●				

*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 and HT; C includes C and 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]

1. Summary

Move the specified data to the other soft components

Fill Move [FMOV]			
16 bits	FMOV	32 bits	DFMOV
Execution condition	Normally ON/OFF, rising/falling edge	Suitable Models	XD3, XDM
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	16/32 bits, BIN;

3. Suitable soft component

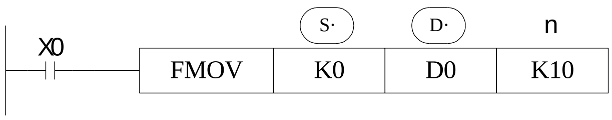
Word

Operand	System								Constant	Module	
	D*	FD	TD*	CD*	DX	DY	DM*	DS*	K/H	ID	QD
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.

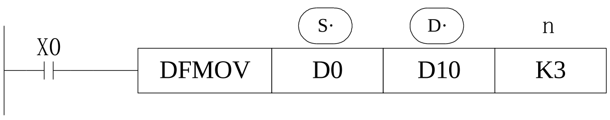
Description

<16 bits instruction>



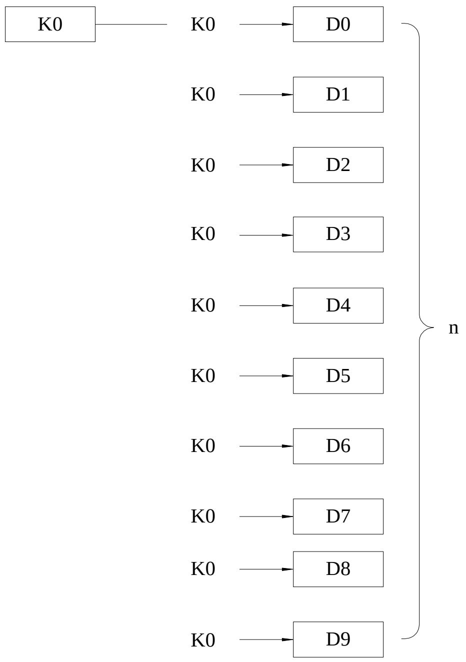
Move K0 to D0~D9, copy a single data device to a range of destination device
Move the source data to target data, the target data quantity is n
If the set range exceeds the target range, move to the possible range

<32 bits instruction >

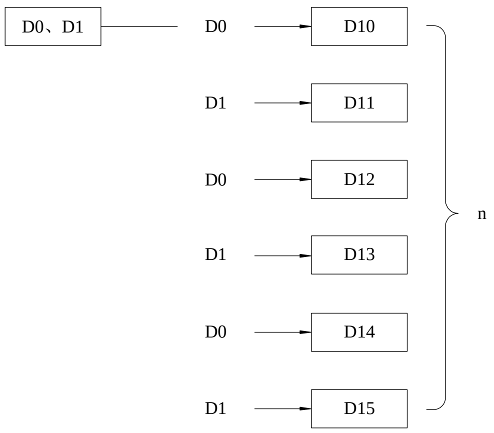


Move D0.D1 to D10.D11:D12.D13:D14.D15.

<16 bits Fill Move >



<32 bits Fill move>



4-5-7. Floating move [EMOV]

Summary

Move the float number to target address

Floating move [EMOV]			
16 bits	-	32 bits	EMOV
Execution condition	Normally on/off, edge trigger	Suitable models	XD3, XDM
Hardware	-	Software	-

Operands

Operand	Function	Type
S	Source soft element address	32 bits, BIN
D	Destination soft element address	32 bits, BIN

Suitable soft element

Word

Operand	System								Constant	Module	
	D*	FD	TD*	CD*	DX	DY	DM*	DS*	K/H	ID	QD
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.

Description

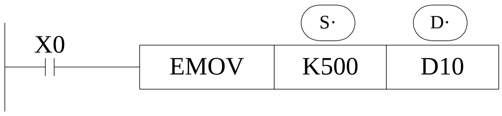
<32 bits instruction>

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.

4-5-8. FlashROM Write [FWRT]

1. Summary

Write the specified data to FlashRom register.

FlashROM Write [FWRT]			
16 bits	FWRT	32 bits	DFWRT
Execution condition	rising/falling edge	Suitable Models	XD3, XDM
Hardware requirement	-	Software requirement	-

2. Operands

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

3. Suitable soft components

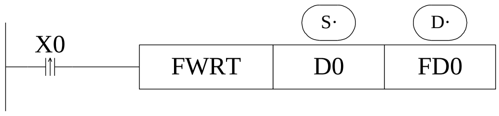
Word

Operand	System								Constant	Module	
	D*	FD	TD*	CD*	DX	DY	DM*	DS*	K/H	ID	QD
S	●	●	●	●	●	●	●	●	●		
D		●									
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.

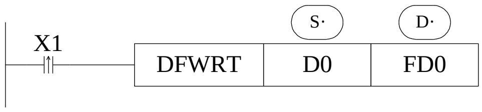
Description

< Written of single word >



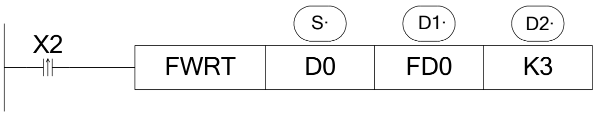
Write value from D0 to FD0

<Written of double words>



Write value from D0,D1 to FD0,FD1

<Written of multi-word>



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

※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 150ms, 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]

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	XD3, XDM
Hardware requirement	-	Software requirement	-

2. Operands

Operands	Function	Data Type
D1	Start soft element address	bit

D2	End soft element address	bit

3. Suitable soft components

Bit

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

*Notes: M includes M, HM, SM; S includes S and HS; T includes T and HT; C includes C and 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]

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	XD3, XDM
Hardware requirement	-	Software requirement	-

2. Operands

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

3. Suitable soft components

Word

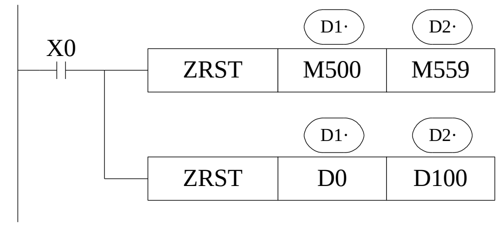
Operand	System								Constant	Module	
	D*	FD	TD*	CD*	DX	DY	DM*	DS*	K/H	ID	QD
D1	●				●	●	●				
D2	●			●	●	●	●				

Bit

Operand	System						
	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 and HS; T includes T and HT; C includes C and 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 and low byte of specified register

High and low byte swap [SWAP]			
16 bits	SWAP	32 bits	-
Execution condition	Falling or rising pulse edge	Suitable Models	XD3, XDM
Hardware requirement	-	Software requirement	-

2. Operands

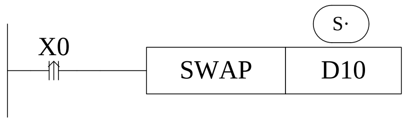
Operands	Function	Data Type
S	The address of the soft element	16 bits; BIN

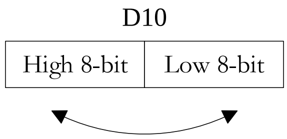
3. Suitable soft components

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

*Notes: D includes D, HD; TD includes TD, HTD; CD includes CD, HCD, HSCD, HSD; DM includes DM, DHM; DS includes DS, DHS.

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.

4-5-12. Exchange [XCH]

1. Summary

Exchange the data in two soft element

Exchange [XCH]			
16 bits	XCH	32 bits	DXCH
Execution condition	Rising or falling pulse edge	Suitable Models	XD3, XDM
Hardware requirement	-	Software requirement	-

2. Operands

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

3. Suitable soft component

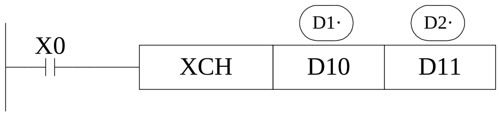
Word

Operand	System								Constant	Module	
	D*	FD	TD*	CD*	DX	DY	DM*	DS*	K/H	ID	QD
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.

Description

<16 bits instruction>

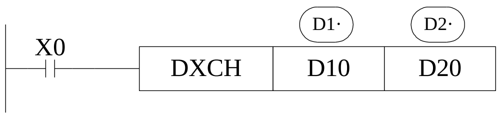


Before (D10) =100 → After (D10) =101
(D11) =101 (D11) =100

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]

1. Summary

Add two numbers and store the result

Add [ADD]			
16 bits	ADD	32 bits	DADD
Execution condition	Normal ON/OFF/falling or rising pulse edge	Suitable Models	XD3, XDM
Hardware requirement	-	Software requirement	-

2. Operands

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

3. Suitable soft components

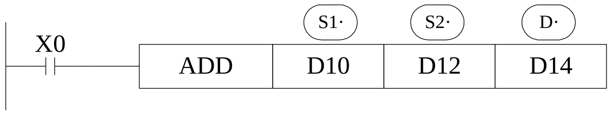
Word

Operand	System								constant	Module	
	D*	FD	TD*	CD*	DX	DY	DM*	DS*	K/H	ID	QD
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.

Description

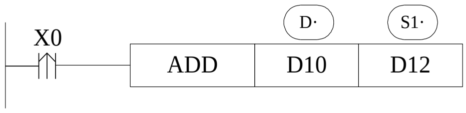
< Three operands>



(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 limit) or 2147483647 (32 bits limit), the carry flag acts. (refer to the next page). If the result exceeds –323768（16 bits limit）or –2147483648 (32 bits limit), the borrow flag acts (refer to the next page).
When doing 32 bits operation, word device’s low 16 bits are assigned; the device close to the preceding device’s is the high bits. To avoid ID repetition, we recommend you assign device’s ID to be even number.
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>



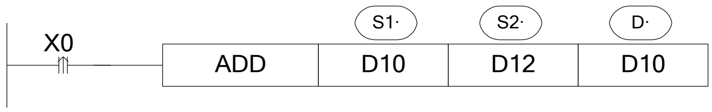
(D10)＋ (D12) → (D10)

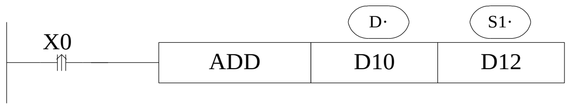
Two source data do binary addition and send the result to addend data address. Each data’s highest bit is the sign bit, 0 stands for positive, 1 stands for negative. All calculations are algebraic processed. (5+ (-8) =-3)
If the result of a calculation is “0”, the “0” flag acts. If the result exceeds 323767 (16 bits limit) or 2147483647 (32 bits limit), the carry flag acts. (refer to the next page). If the result

exceeds –323768（16 bits limit）or –2147483648 (32 bits limit), the borrow flag acts (refer to the next page).

When doing 32 bits operation, word device’s low 16 bits are assigned; the device close to the preceding device’s is the high bits. To avoid ID repetition, we recommend you assign device’s ID to be even number.

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
SM020	Zero	ON: the calculate result is zero OFF: the calculate result is not zero
SM021	Borrow	ON: the calculate result is over -32768(16 bit) or -2147483648(32bit) OFF: the calculate result is less than -32768(16 bit) or -2147483648(32bit)
SM022	Carry	ON: the calculate result is over 32768(16 bit) or 2147483648(32bit) OFF: the calculate result is less than 32768(16 bit) or 2147483648(32bit)

4-6-2．Subtraction [SUB]

1. Summary

Two numbers do subtraction, store the result

Subtraction [SUB]			
16 bits	SUB	32 bits	DSUB
Execution condition	Normally ON/OFF/rising or falling pulse edge	Suitable Models	XD3, XDM
Hardware requirement	-	Software requirement	-

Operands

Operands	Function	Data Type
Three operands		
S1	The sub operation data address	16 bits /32 bits,BIN
S2	The sub operation data address	16 bits /32 bits,BIN
D	The result address	16 bits /32 bits,BIN

Two operands		
D	Be subtracted data and result address	16 bits /32 bits,BIN
S1	Subtract data address	16 bits /32 bits,BIN

Suitable soft component

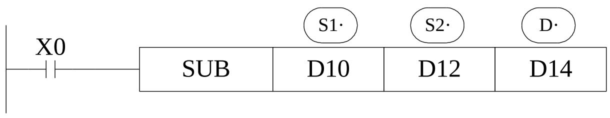
Word

Operand	System								Constant	Module	
	D*	FD	TD*	CD*	DX	DY	DM*	DS*	K/H	ID	QD
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.

Description

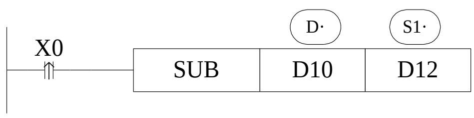
<Three operands>



(D10) — (D12) → (D14)

S1· appoint the soft unit's content, subtract the soft unit's content appointed by S2· in the format of algebra. The result will be stored in the soft unit appointed by D· (5-(-8)=13).
The action of each flag, the setting method of 32 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.

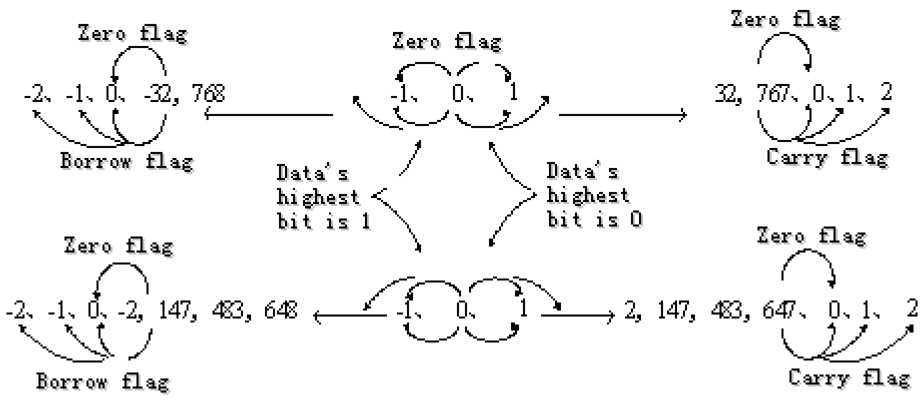
<Two operands>



(D10) — (D12) → (D10)

S1· appoint the soft unit's content, subtract the soft unit's content appointed by S2· in the format of algebra. The result will be stored in the soft unit appointed by D· (5-(-8)=13)
The action of each flag, the setting method of 32 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:



4-6-3. Multiplication [MUL]

1. Summary

Multiply two numbers, store the result

Multiplication [MUL]			
16 bits	MUL	32 bits	DMUL
Execution condition	Normally ON/OFF / pulse edge	Suitable Models	XD3, XDM
Hardware requirement	-	Software requirement	-

2. Operands

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

3. Suitable soft component

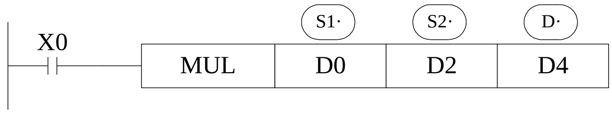
Word

Operand	System								Constant	Module	
	D*	FD	TD*	CD*	DX	DY	DM*	DS*	K/H	ID	QD
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.

Description

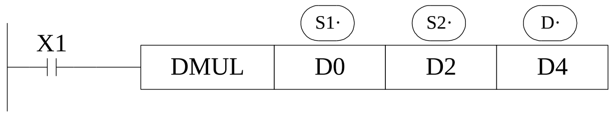
<16 bits Operation>



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 of 32 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 use 32 bits operation, the result is stored at the
bits.
Even use word device, 64 bits results can't be monitored.
Please change to floating value operation for this case.

4-6-4. Division [DIV]

1. Summary

Divide two numbers and store the result

Division [DIV]			
16 bits	DIV	32 bits	DDIV
Execution condition	Normally ON/OFF, rising/falling edge	Suitable Models	XD3, XDM
Hardware requirement	-	Software requirement	-

2. Operands

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

3. Suitable soft components

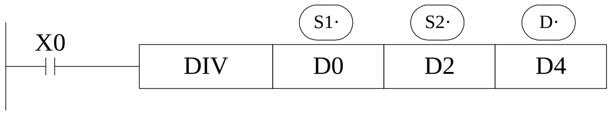
Word

Operand	System								Constant	Module	
	D*	FD	TD*	CD*	DX	DY	DM*	DS*	K/H	ID	QD
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.

Description

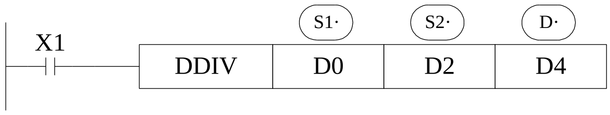
<16 bits operation >



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, D· and the next address appoint the soft component of the result 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.

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

1. Summary

Increase or decrease the number

Increase one [INC]			
16 bits	INC	32 bits	DINC
Execution condition	Normally ON/OFF, rising/falling edge	Suitable Models	XD3, XDM
Hardware requirement	-	Software requirement	-
Decrease one [DEC]			
16 bits	DEC	32 bits	DDEC
Execution condition	Normally ON/OFF, rising/falling edge	Suitable Models	XD3, XDM
Hardware requirement	-	Software requirement	-

2. Operands

Operands	Function	Data Type
D	The increase or decrease data address	16 bits / 32bits,BIN

3. Suitable soft components

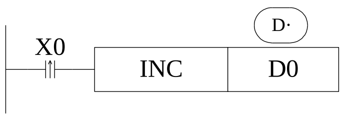
Word

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

*Notes: D includes D, HD; TD includes TD, HTD; CD includes CD, HCD, HSCD, HSD; DM includes DM, DHM; DS includes DS, DHS.

Description

< Increment [INC]>

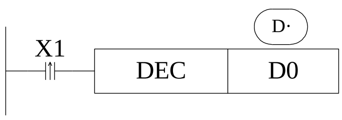


(D0) ＋1→(D0)

D· will increase one when X0 is ON.

For 16 bits operation, when +32767 increase one, it will become -32768; for 32 bits operation, +2147483647 increases one is -2147483647. 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.

4-6-6. Mean [MEAN]

1. Summary

Get the mean value of data

Mean [MEAN]			
16 bits	MEAN	32 bits	DMEAN
Execution condition	Normally ON/OFF, rising/falling edge	Suitable Models	XD3, XDM
Hardware requirement	-	Software requirement	-

2. Operands

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

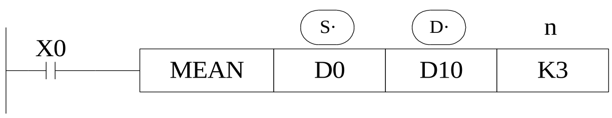
3. Suitable soft components

Word

Operand	System								Constant	Module	
	D*注	FD	TD*注	CD*注	DX	DY	DM*注	DS*注	K/H	ID	QD
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.

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], Logic OR[WOR], Logic Exclusive OR [WXOR]

1. Summary

Do logic AND, OR, XOR for data

Logic AND [WAND]			
16 bits	WAND	32 bits	DWAND
Execution condition	Normally ON/OFF, rising/falling edge	Suitable Models	XD3, XDM
Hardware requirement	-	Software requirement	-
Logic OR[WOR]			
16 bits	WOR	32 bits	DWOR
Execution condition	Normally ON/OFF, rising/falling edge	Suitable Models	XD3, XDM
Hardware requirement	-	Software requirement	-
Logic Exclusive OR [WXOR]			
16 bits	WXOR	32 bits	DWXOR
Execution condition	Normally ON/OFF, rising/falling edge	Suitable Models	XD3, XDM
Hardware requirement	-	Software requirement	-

2. Operands

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

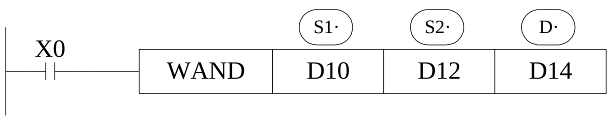
3. Suitable soft components

Word	Operand	System								Constant	Module	
		D*	FD	TD*	CD*	DX	DY	DM*	DS*	K/H	ID	QD
	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.

Description

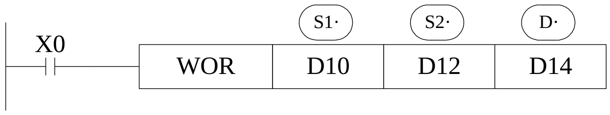
< Logic AND >



0&0=0 0&1=0

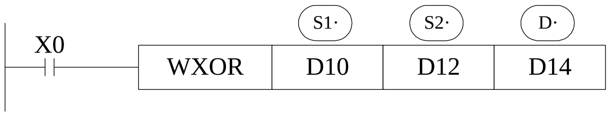
1&0=0 1&1=1

< Logic OR >



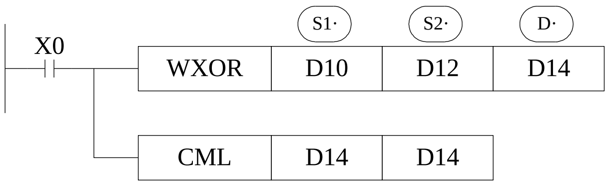
0 or 0=0 0 or 1=1
1 or 0=1 1 or 1=1

< Logic WXOR >



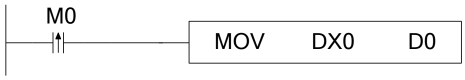
0 xor 0=0 0 xor 1=1
1 xor 0=1 1 xor 1=0

If use this instruction along with CML instruction, XOR NOT executed.



Example 1:

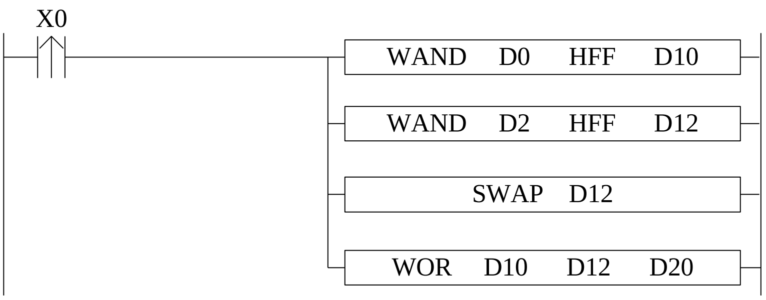
The 16 bits data is composed by X0~X7, and store in D0.



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

4-6-8. Logic converse [CML]

1. Summary

Logic converse the data

Converse [CML]			
16 bits	CML	32 bits	DCML
Execution condition	Normally ON/OFF, rising/falling edge	Suitable Models	XD3, XDM
Hardware requirement	-	Software requirement	-

2. Operands

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

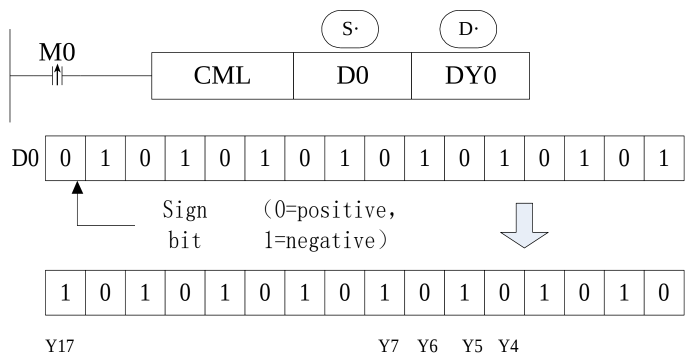
3. Suitable soft components

Word

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

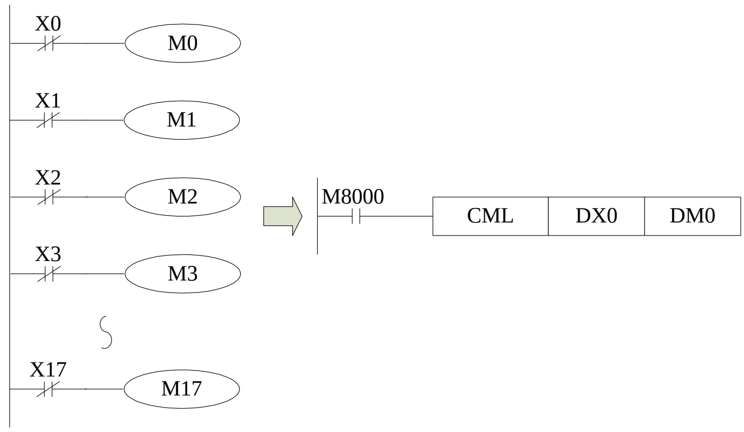
*Notes: D includes D, HD; TD includes TD, HTD; CD includes CD, HCD, HSCD, HSD; DM includes DM, DHM; DS includes DS, DHS.

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]

1. Summary

Get the negative data

Negative [NEG]			
16 bits	NEG	32 bits	DNEG
Execution condition	Normally ON/OFF, rising/falling edge	Suitable Models	XD3, XDM
Hardware requirement	-	Software requirement	-

2. Operands

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

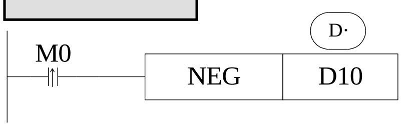
3. Suitable soft components

Word

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

*Notes: D includes D, HD; TD includes TD, HTD; CD includes CD, HCD, HSCD, HSD; DM includes DM, DHM; DS includes DS, DHS.

Description



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

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], Arithmetic shift right [SHR]

1. Summary

Do arithmetic shift left/right for the numbers

Arithmetic shift left [SHL]			
16 bits	SHL	32 bits	DSHL
Execution condition	Normally ON/OFF, rising/falling edge	Suitable Models	XD3, XDM
Hardware requirement	-	Software requirement	-
Arithmetic shift right [SHR]			
16 bits	SHR	32 bits	DSHR
Execution condition	Normally ON/OFF, rising/falling edge	Suitable Models	XD3, XDM
Hardware requirement	-	Software requirement	-

2. Operands

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

3. Suitable soft components

Word

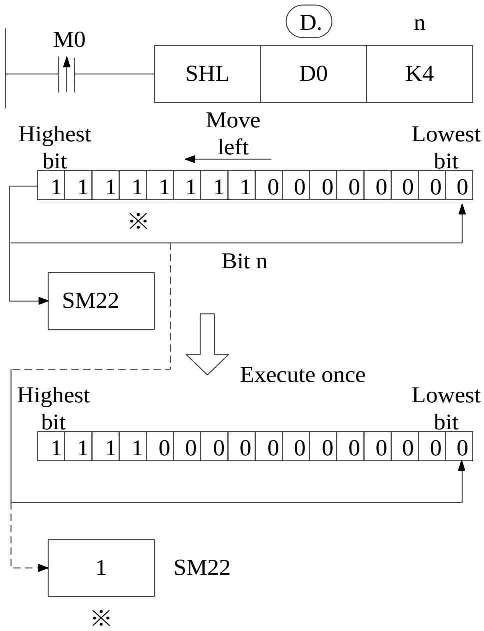
Operand	System								Constant	Module	
	D*	FD	TD*	CD*	DX	DY	DM*	DS*	K/H	ID	QD
D	●		●	●		●	●	●			
n									●		

*Notes: D includes D, HD; TD includes TD, HTD; CD includes CD, HCD, HSCD, HSD; DM includes DM, DHM; DS includes DS, DHS.

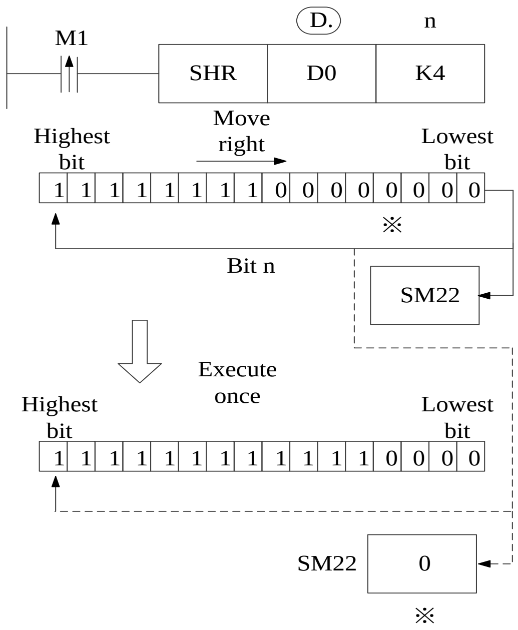
Description

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], Logic shift right [LSR]

1. Summary

Do logic shift right/left for the data

Logic shift left [LSL]			
16 bits	LSL	32 bits	DLSL
Execution condition	Normally ON/OFF, rising/falling edge	Suitable Models	XD3, XDM
Hardware requirement	-	Software requirement	-
Logic shift right [LSR]			
16 bits	LSR	32 bits	DLSR
Execution condition	Normally ON/OFF, rising/falling edge	Suitable Models	XD3, XDM
Hardware requirement	-	Software requirement	-

2. Operands

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

3. Suitable soft components

Word

Operand	System								Constant	Module	
	D*	FD	TD*	CD*	DX	DY	DM*	DS*	K/H	ID	QD
D	●		●	●		●	●	●			
n									●		

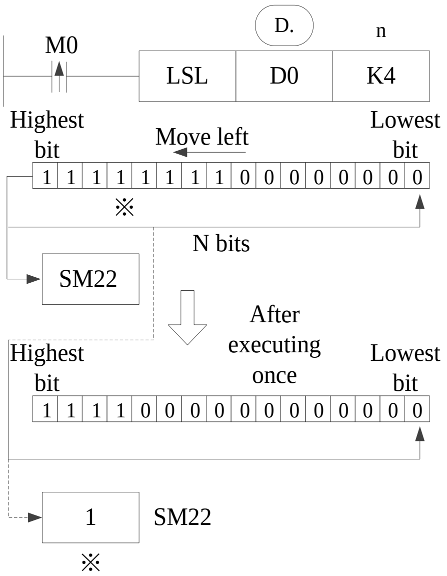
*Notes: D includes D, HD; TD includes TD, HTD; CD includes CD, HCD, HSCD, HSD; DM includes DM, DHM; DS includes DS, DHS.

Description

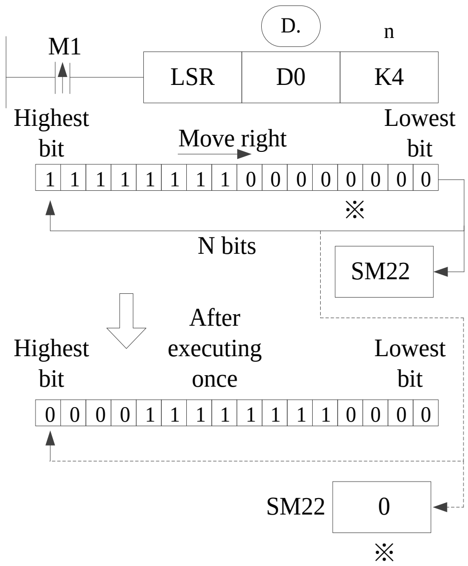
After executing LSL once, the lowest bit is filled with 0; the last bit is stored in carry flag. LSL meaning and operation are the same to SHL.

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 in the highest bit when moving, SHR all bits are moved.

< Logic shift left >



< Logic shift right >



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

1. Summary

Cycle shift left or right

Rotation shift left [ROL]			
16 bits	ROL	32 bits	DROL
Execution condition	rising/falling edge	Suitable Models	XD3, XDM
Hardware requirement	-	Software requirement	-
Rotation shift right [ROR]			
16 bits	ROR	32 bits	DROR
Execution condition	rising/falling edge	Suitable Models	XD3, XDM
Hardware requirement	-	Software requirement	-

2. Operands

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

3. Suitable soft components

Word

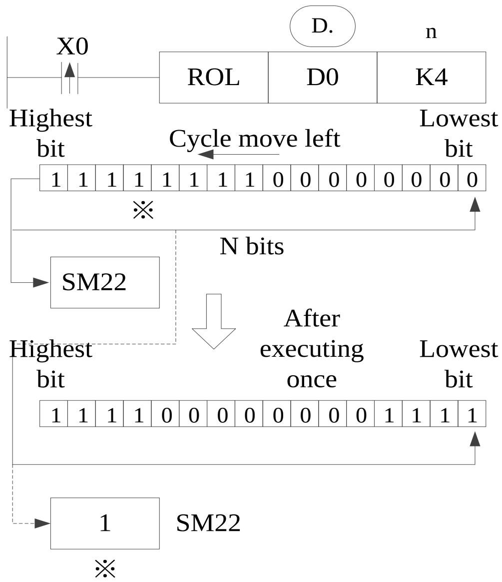
Operand	System								Constant	Module	
	D*	FD	TD*	CD*	DX	DY	DM*	DS*	K/H	ID	QD
D	●		●	●		●	●	●			
n									●		

*Notes: D includes D, HD; TD includes TD, HTD; CD includes CD, HCD, HSCD, HSD; DM includes DM, DHM; DS includes DS, DHS.

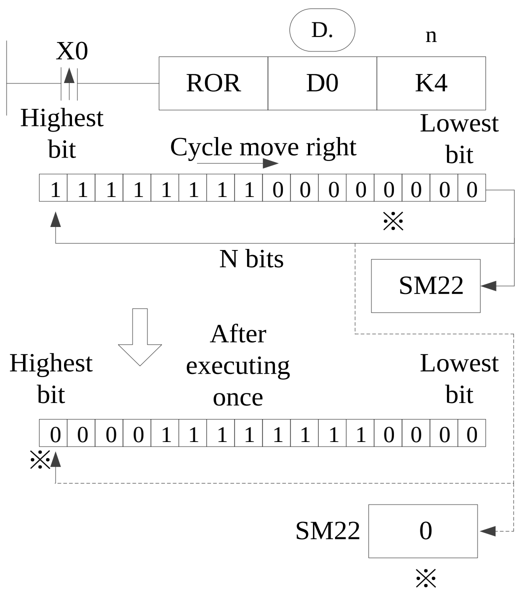
Description

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	DSFTL
Execution condition	rising/falling edge	Suitable Models	XD3, XDM
Hardware requirement	-	Software requirement	-

2. Operands

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

3. Suitable soft components

Word

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

Bit

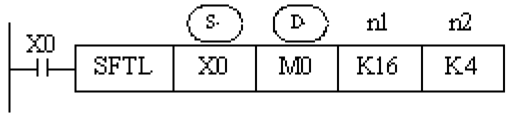
Operand	System						
	X	Y	M*	S*	T*	C*	Dn.m
S	●	●	●	●	●	●	
D		●	●	●	●	●	

*Notes: D includes D, HD; TD includes TD, HTD; CD includes CD, HCD, HSCD, HSD; DM includes DM, DHM; DS includes DS, DHS.

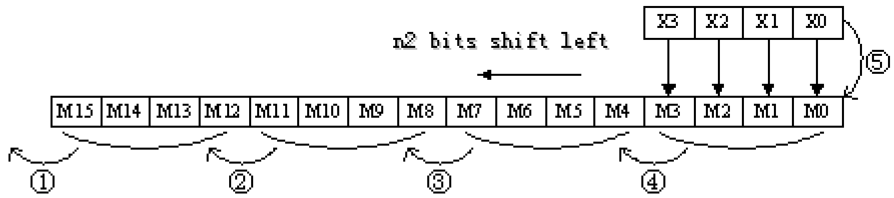
M includes M, HM, SM;   S includes S, HS;   T includes T, HT;   C includes C, HC.

Description

Move 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 1, the object will move 1 bit left when the instruction executes once.



M15~M12→Overflow
M11~M 8→M15~M 12
M 7~M 4→M11~M8
M 3~M 0→M7~M4
X 3~X 0→M3~M0



4-7-5. Bit shift right [SFTR]

1. Summary

Bit shift right

Bit shift right [SFTR]			
16 bits	SFTR	32 bits	DSFTR
Execution condition	rising/falling edge	Suitable Models	XD3, XDM
Hardware requirement	-	Software requirement	-

2. Operands

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

3. Suitable soft components

Word

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

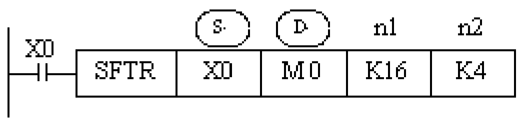
Bit

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

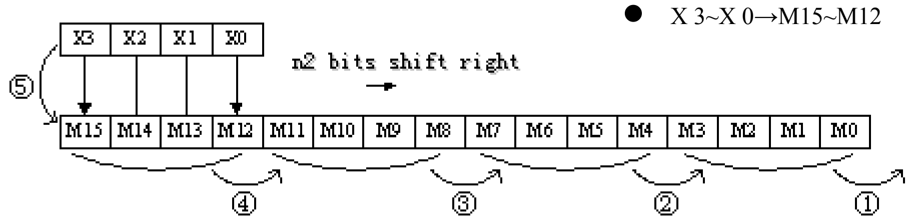
*Notes: D includes D, HD; TD includes TD, HTD; CD includes CD, HCD, HSCD, HSD;
DM includes DM, DHM; DS includes DS, DHS.
M includes M, HM, SM; S includes S, HS; T includes T, HT; C includes C, HC.

Description

Move 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.



- M 3~M 0→Overflow
- M 7~M 4→M3~M0
- M11~M 8→M7~M4
- M15~M12→M11~M8
- X 3~X 0→M15~M12



4-7-6. Word shift left [WSFL]

1. Summary

Word shift left

Word shift left [[WSFL]			
16 bits	WSFL	32 bits	-
Execution condition	rising/falling edge	Suitable Models	XD3, XDM
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	16 bits, BIN
n2	Word shift left times	16 bits, BIN

3. Suitable soft components

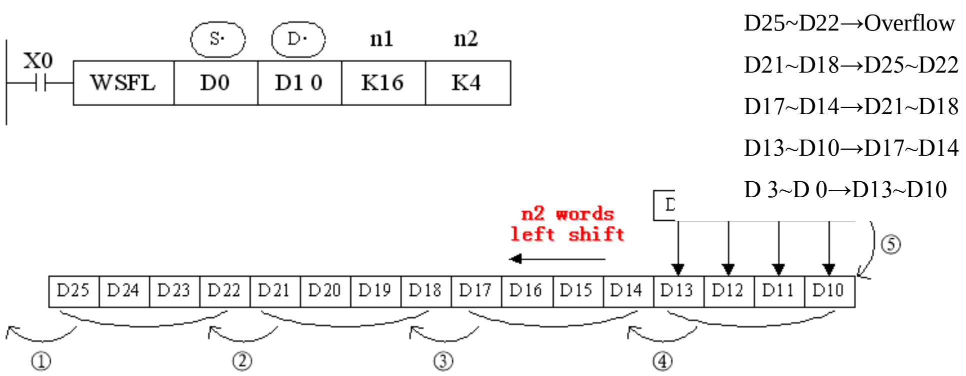
Word

Operand	System								Constant	Module	
	D*	FD	TD*	CD*	DX	DY	DM*	DS*	K/H	ID	QD
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.

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.



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	XD3, XDM
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	16 bits, BIN
n2	Shift right times	16 bits, BIN

3. Suitable soft components

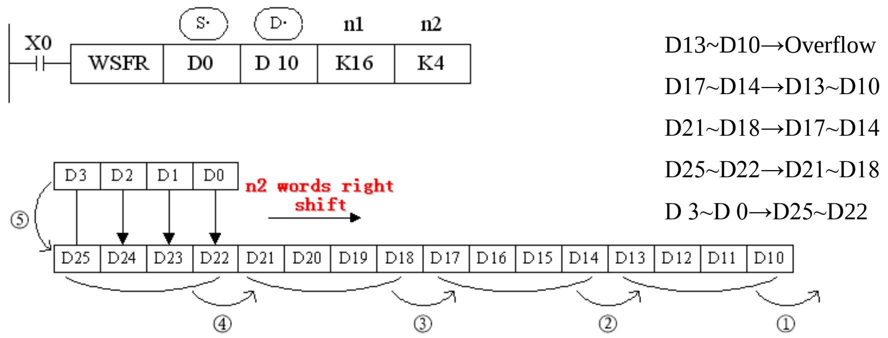
Word

Operand	System								Constant	Module	
	D*	FD	TD*	CD*	DX	DY	DM*	DS*	K/H	ID	QD
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.

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.



4-8. Data Convert

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

BIN	BCD convert to binary	4-8-4
BCD	Binary converts to BCD	4-8-5
ASCI	Hex. converts to ASCII	4-8-6
HEX	ASCII converts to Hex.	4-8-7
DECO	Coding	4-8-8
ENCO	High bit coding	4-8-9
ENCOL	Low bit coding	4-8-10
GRY	Binary converts to gray code	4-8-11
GBIN	Gray code converts to binary	4-8-12

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

1. Summary

Single word integer converts to double word integer [WTD]			
16 bits	WTD	32 bits	-
Execution condition	Normally ON/OFF, rising/falling edge	Suitable Models	XD3, XDM
Hardware requirement	-	Software requirement	-

2. Operands

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

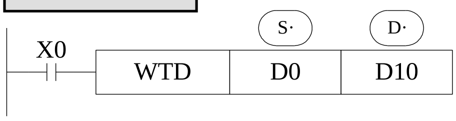
3. Suitable soft components

Word

Operand	System								Constant	Module	
	D*	FD*	TD*	CD*	DX	DY	DM*	DSV	K/H	ID	QD
S	●	●	●	●	●	●	●	●			
D	●		●	●		●	●	●			

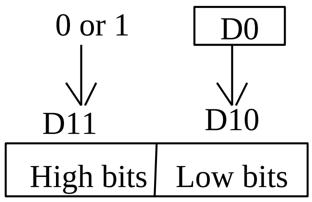
*Notes: D includes D, HD; TD includes TD, HTD; CD includes CD, HCD, HSCD, HSD; DM includes DM, DHM; DS includes DS, DHS.

Description



(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.

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

1. Summary

16 bits integer converts to float point [FLT]					
16 bits	FLT	32 bits	DFLT	64 bits	FLTD
Execution condition	Normally ON/OFF, rising/falling edge		Suitable Models	XD3, XDM	
Hardware requirement	-		Software requirement	-	

2. Operands

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

3. Suitable soft components

Word

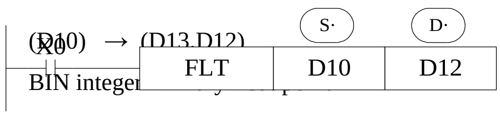
Operand	System									Constant	Module	
	D*	FD	ED	TD*	CD*	DX	DY	DM*	DS*	K/H	ID	QD
S	●	●								●		
D	●											

*Notes: D includes D, HD; TD includes TD, HTD; CD includes CD, HCD, HSCD, HSD; DM includes DM, DHM; DS includes DS, DHS.

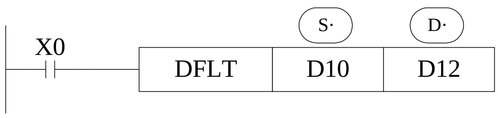
Description

<16 bits>

(D10) → (D13,D12)
BIN integer

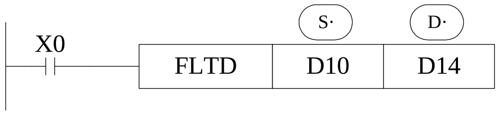


<32 bits >



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

<64 bits>



(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 the 64 bits integer to 32 bits floating value.

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

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

1. Summary

Floating point converts to integer [INT]			
16 bits	INT	32 bits	DINT
Execution condition	Normally ON/OFF, rising/falling edge	Suitable Models	XD3, XDM
Hardware requirement	-	Software requirement	-

2. Operands

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

3. Suitable soft components

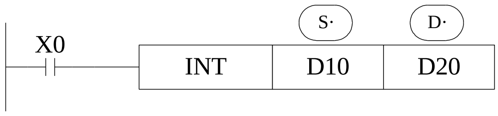
Word

Operand	System								Constant	Module	
	D*	FD	TD*	CD*	DX	DY	DM*	DS*	K/H	ID	QD
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; the word combined by bits.

Description

<16 bits>

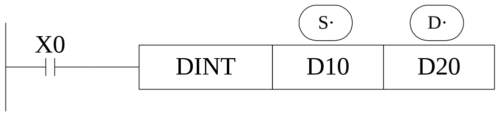


(D11,D10) → (D20)

Binary Float BIN integer

Give up the data after the decimal dot

<32 bits>



(D11,D10) → (D20,D21)

Binary Float BIN integer

Give up the data after the decimal dot

The binary source number is converted into a BIN integer and stored at the destination device.
Abandon the value behind the decimal point.
The inverse instruction is FLT.
When the result is 0, the flag bit is ON.
When converting, less than 1 and abandon it, zero flag is ON.
The result is over below data, the carry flag is ON.
16 bits operation: -32,768~32,767
32 bits operation: -2,147,483,648~2,147,483,647

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

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

1. Summary

BCD convert to binary [BIN]			
16 bits	BIN	32 bits	-
Execution condition	Normally ON/OFF, rising/falling edge	Suitable Models	XD3, XDM
Hardware requirement	-	Software requirement	-

2. Operands

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

3. Suitable soft components

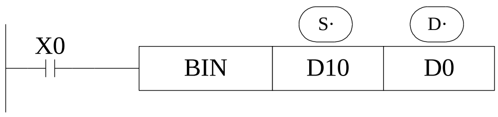
Word

Operand	System								Constant	Module	
	D*	FD	TD*	CD*	DX	DY	DM*	DS*	K/H	ID	QD
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.

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.

4-8-5. 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	XD3, XDM
Hardware requirement	-	Software requirement	-

2. Operands

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

3. Suitable soft components

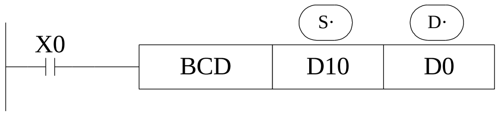
Word

Operand	System								Constant	Module	
	D*	FD	TD*	CD*	DX	DY	DM*	DS*	K/H	ID	QD
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.

Description

source (BIN)→destination (BCD)



This instruction can change the binary value to BCD code.

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

1. Summary

Hex. convert to ASCII [ASCI]			
16 bits	ASCI	32 bits	-
Execution condition	Normally ON/OFF, rising/falling edge	Suitable Models	XD3, XDM
Hardware requirement	-	Software requirement	-

2. Operands

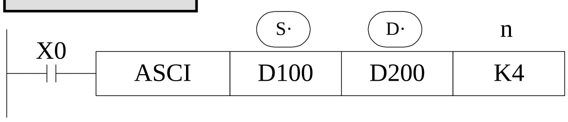
Operands	Function	Data Type
S	Source soft element address	2 bits, HEX
D	Target soft element address	ASCII code
n	Transform character quantity	16 bits, BIN

3. Suitable soft components

Word	Operand	System								Constant	Module	
		D*	FD	TD*	CD*	DX	DY	DM*	DS*	K/H	ID	QD
	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.

Description



Transform the source Hex data to ASCII code, and store in D·. The transformation chacters are n.

D· Will store one ASCII code.

The convert process is this

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

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

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

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

1. Summary

ASCII converts to Hex. [HEX]			
16 bits	HEX	32 bits	-
Execution condition	Normally ON/OFF, rising/falling edge	Suitable Models	XD3, XDM
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

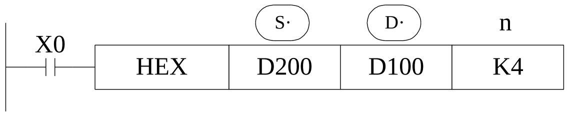
3. Suitable soft components

Word

Operand	System								Constant	Module	
	D*	FD	TD*	CD*	DX	DY	DM*	DS*	K/H	ID	QD
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.

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 down	30H	0
D200 up	41H	A
D201 down	42H	B
D201 up	43H	C
D202 down	31H	1
D202 up	32H	2
D203 down	33H	3
D203 up	34H	4
D204 down	35H	5

n (D·)	D102	D101	D100
1	Not change to be 0	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

n=k4

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			

4-8-8. Coding [DECO]

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	XD3, XDM

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

Word

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

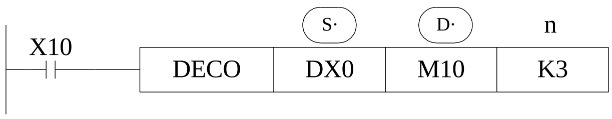
Bit

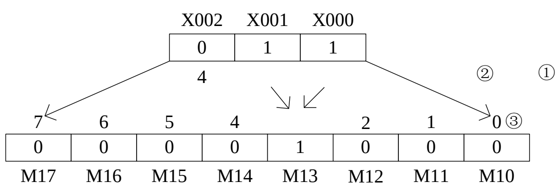
Operand	System						
	X	Y	M*	S*	T*	C*	Dn.m
D	●	●	●	●	●	●	

*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 and HT; C includes C and HC.

Description

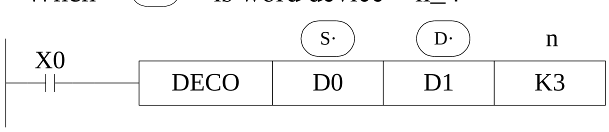
< When D· is bit unit > n≤16

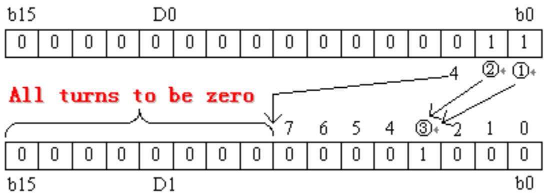




The source address is 1+2=3, so starts from M10; the third bit (M13) is 1. If the source is all 0, M10 is 1.
When n=0, no operation, beyond n=0~16, don't execute the instruction.
When n=16, if decoding command D· is soft unit, it's point is 2^{16}=65536.
When drive input is OFF, instructions are not executed, the decoding output keep on the state.

< When D· is word device > n≤4





Low n bits (n≤4) of source address are decoded to target address. n≤3, the high bit of target address all become 0.

When n=0, no operation, beyond n=0~14, don't execute the instruction.

4-8-9． High bit coding [ENCO]

1. Summary

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	XD3, XDM
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

Word

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

Bit

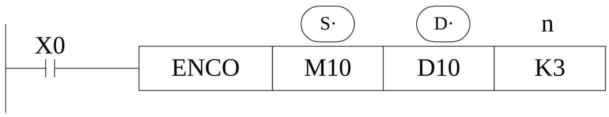
Operand	System						
	X	Y	M*	S*	T*	C*	Dn.m
S	●	●	●	●	●	●	

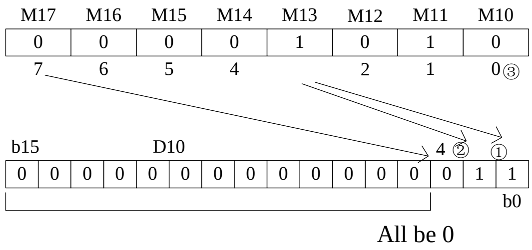
*Notes: D includes D, HD; TD includes TD, HTD; CD includes CD, HCD, HSCD, HSD; DM includes DM, DHM; DS includes DS, DHS.

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

Description

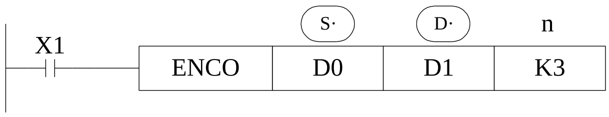
< When S· is bit device > n≤16

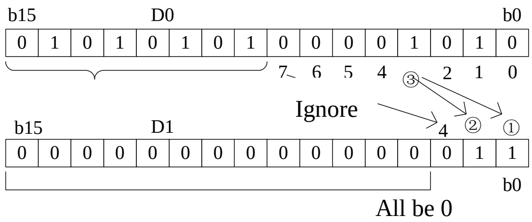




Igore the 1 of M11

< When S· is word device > n≤4





Ignore the 1 of b1

If many bits in the source address are 1, ignore the low bits. If source addresses are all 0, don’t execute the instructions.

When drive input is OFF, the instruction is not executed, encode output don’t change.

When n=16, if encode instruction S· is bit unit, it’s point number is 2^{16}=65536.

4-8-10. Low bit coding [ENCOL]

1. Summary

Find the lowest bit which is 1.

Low bit coding [ENCOL]			
16 bits	ENCOL	32 bits	-
Execution condition	Normally ON/OFF, rising/falling edge	Suitable Models	XD3, XDM
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

Word

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

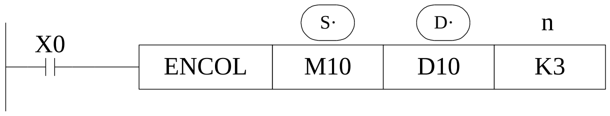
Bit

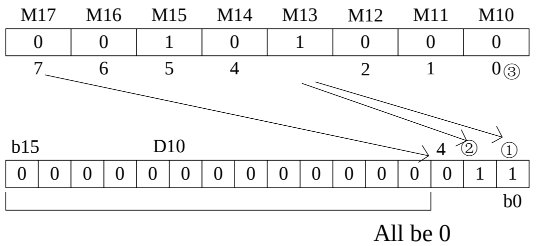
Operand	System						
	X	Y	M*	S*	T*	C*	Dn.m
S	●	●	●	●	●	●	

*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 and HT; C includes C and HC.

Description

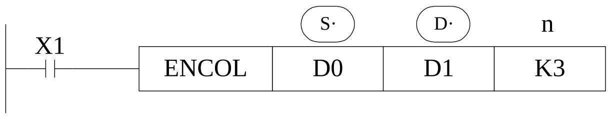
<if S· is bit device > n≤16

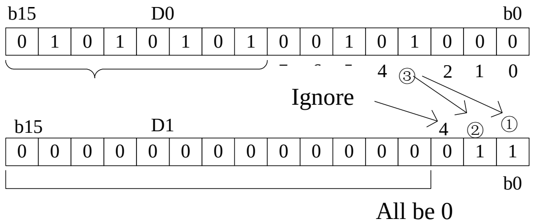




Ignore the 1 of M15

< if S· is word device> n≤4





Ignore the 1 of b5

If many bits in the source address are 1, ignore the high bits. If source address are all 0, don't execute the instructions.
When drive input is OFF, the instruction is not executed, encode output don't change
When n=16, if encode instruction S· is bit unit, it's point number is 2^{16}=65536.

4-8-11. Binary to Gray code [GRY]

1. Summary

Transform the binary data to gray code.

Binary to gray [GRY]			
16 bits	GRY	32 bits	DGRY
Execution condition	Normally ON/OFF, rising/falling edge	Suitable Models	XD3, XDM
Hardware requirement	-	Software requirement	-

2. Operands

Operands	Function	Data Type
S	Soft element address need coding	16bits/32bits, BIN
D	Soft element address to save coding result	16bits/32bits, BIN

3. Suitable soft components

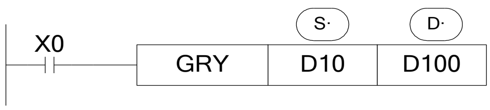
Word

Operand	System								Constant	Module	
	D*	FD	TD*	CD*	DX	DY	DM*	DS*	K/H	ID	QD
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.

Description

Source (BIN) → target (GRY)



b15 D10 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

b15 D100 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 has 32 bits mode DGRY, which can transform 32 bits gray code.

S· Range is 0~32,767 (16 bits instruction); 0~2,147,483,647 (32 bits instruction).

4-8-12. Gray code to binary [GBIN]

1. Summary

Transform the gray code to binary data.

Gray code to binary [GBIN]			
16 bits	GBIN	32 bits	DGBIN
Execution condition	Normally ON/OFF, rising/falling edge	Suitable Models	XD3, XDM
Hardware requirement	-	Software requirement	-

2. Operands

Operands	Function	Data Type
S	Soft element address need coding	16bits/32bits, BIN
D	Soft element address to save coding result	16bits/32bits, BIN

3. Suitable soft components

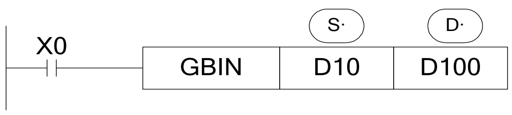
Word

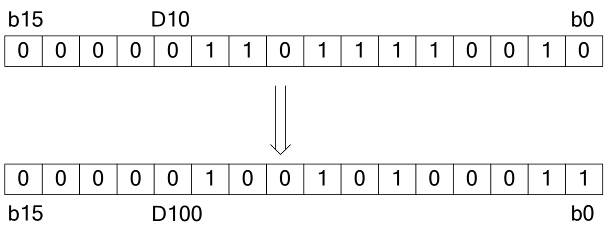
Operand	System								Constant	Module	
	D*	FD	TD*	CD*	DX	DY	DM*	DS*	K/H	ID	QD
S	●	●	●	●	●	●	●	●	●		
D	●		●	●		●	●	●			

*Notes: D includes D, HD; TD includes TD, HTD; CD includes CD, HCD, HSCD, HSD; DM includes DM, DHM; DS includes DS, DHS.

Description

Source (GRY) → target (BIN)





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 has 32 bits mode DBIN, which can transform 32 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]

1. Summary

Floating Compare [ECMP]			
16 bits	-	32 bits	ECMP
Execution condition	Normally ON/OFF, rising/falling edge	Suitable Models	XD3, XDM
Hardware requirement	-	Software requirement	-

2. Operands

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

3. Suitable soft components

Word

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

Bit

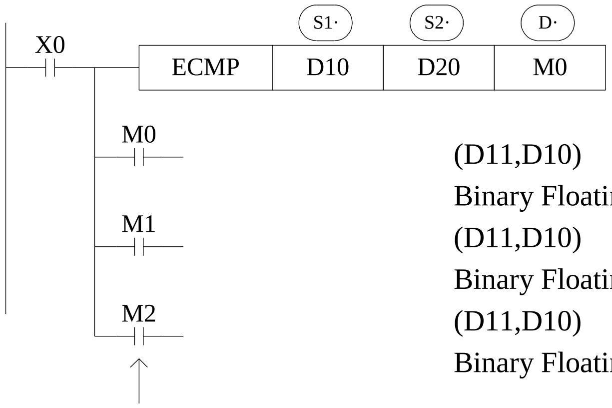
Operand	System						
	X	Y	M*	S*	T*	C*	Dn.m
D		●	●	●			

*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 and HS; C includes C and HC.

Description

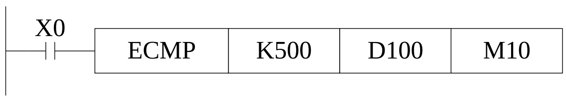
(D11, D10) : (D21, D20) → M0,M1,M2
Binary Floating Binary Floating



(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.
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 to floating Binary 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	XD3, XDM
Hardware requirement	-	Software requirement	-

2. Operands

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

3. Suitable soft components

Word

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

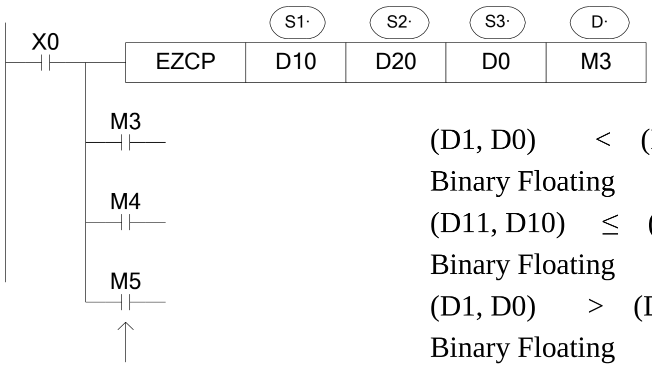
Bit

Operand	System						
	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 and HS; C includes C and HC.

Description

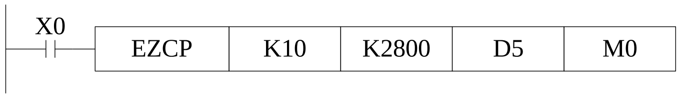
Compare the source data with the range



(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.
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 float numbers, otherwise the result will be error.

4-9-3. Floating Addition [EADD]

1. Summary

Floating Add [EADD]			
16 bits	-	32 bits	EADD
Execution condition	Normally ON/OFF, rising/falling edge	Suitable Models	XD3, XDM
Hardware requirement	-	Software requirement	-

2. Operands

Operands	Function	Data Type
S1	Addition operation data address	32 bits, BIN
S2	Addition operation data address	32 bits, BIN
D	Result address	32 bits, BIN

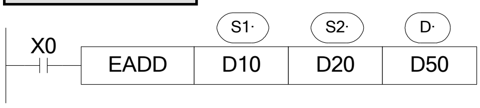
3. Suitable soft components

Word

Operand	System								Constant	Module	
	D*	FD	TD*	CD*	DX	DY	DM*	DS*	K/H	ID	QD
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.

Description



(D11, D10) + (D21, D20) → (D51, D50)

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.

X1

EADD	D100	K1234	D110

(K1234) + (D101, D100) → (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.

Note: the add value must be float numbers, otherwise the result will be error.

4-9-4. Floating Subtraction [ESUB]

1. Summary

Floating Sub [ESUB]			
16 bits	-	32 bits	ESUB
Execution condition	Normally ON/OFF, rising/falling edge	Suitable Models	XD3, XDM
Hardware requirement	-	Software requirement	-

2. Operands

Operands	Function	Data Type
S1	Subtraction operation data address	32 bits, BIN
S2	Subtraction operation data address	32 bits, BIN
D	Result address	32 bits, BIN

3. Suitable soft components

Word

Operand	System								Constant	Module	
	D*	FD	TD*	CD*	DX	DY	DM*	DS*	K/H	ID	QD
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.

Description

		S1·	S2·	D·
X0	ESUB	D10	D20	D50

(D11, D10) － (D21,D20) → (D51,D50)
Binary Floating Binary Floating Binary Floating

The binary floating value S1 subtract S2, the result is stored in the target address.
If a constant K or H used as source data, the value is converted to floating point before the subtraction operation.

X1	ESUB	K1234	D100	D110

(K1234) － (D101, D100) → (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.

4-9-5. Floating Multiplication [EMUL]

1. Summary

Floating Multiply [EMUL]			
16 bits	-	32 bits	EMUL
Execution condition	Normally ON/OFF, rising/falling edge	Suitable Models	XD3, XDM
Hardware requirement	-	Software requirement	-

2. Operands

Operands	Function	Data Type
S1	Multiplication operation data address	32 bits, BIN
S2	Multiplication operation data address	32 bits, BIN
D	Result address	32 bits, BIN

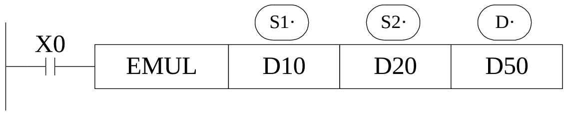
3. Suitable soft components

Word

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

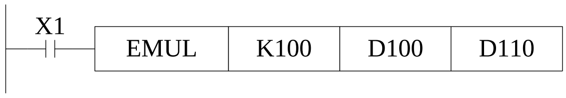
*Notes: D includes D, HD; TD includes TD, HTD; CD includes CD, HCD, HSCD, HSD; DM includes DM, DHM; DS includes DS, DHS.

Description



(D11, D10) × (D21, D20) → (D51, D50)
Binary Floating Binary Floating Binary Floating

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



(K100) × (D101, D100) → (D111, D110)
Binary converts to Floating Binary Floating Binary Floating

Note: the value must be floating numbers, otherwise the result will be error.

4-9-6. Floating Division [EDIV]

1. Summary

Floating Divide [EDIV]			
16 bits	-	32 bits	EDIV
Execution condition	Normally ON/OFF, rising/falling edge	Suitable Models	XD3, XDM
Hardware requirement	-	Software requirement	-

2. Operands

Operands	Function	Data Type
S1	Division operation data address	32 bits, BIN
S2	Division operation data address	32 bits, BIN
D	Result address	32 bits, BIN

3. Suitable soft components

Word

Operand	System								Constant	Module	
	D*	FD	TD*	CD*	DX	DY	DM*	DS*	K/H	ID	QD
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.

Description

		S1·	S2·	D·
X0	EDIV	D10	D20	D50

(D11, D10) ÷ (D21, D20) → (D51, D50)
Binary Floating Binary Floating Binary Floating

The floating point value of S1 is divided by the floating point value of S2. The result of the division is stored in D as a floating point value.
If a constant K or H used as source data, the value is converted to floating point before the division operation.

X1	EDIV	D100	K100	D110

(D101, D100) ÷ (K100) → (D111, D110)
Binary converts to Floating Binary Floating Binary Floating

The source data S2 is 0, the calculation will be error. The instruction will not work.
Note: the value must be floating numbers, otherwise the result will be error.

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	XD3, XDM
Hardware requirement	-	Software requirement	-

2. Operands

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

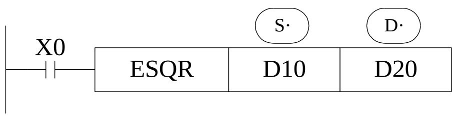
3. Suitable soft components

Word

Operand	System								Constant	Module	
	D*	FD	TD*	CD*	DX	DY	DM*	DS*	K/H	ID	QD
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.

Description

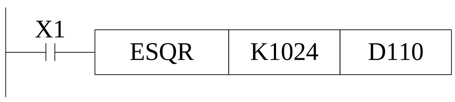


(D11, D10) → (D21, D20)

Binary Floating Binary Floating

A square root is performed on the floating point value S; the result is stored in D

If a constant K or H used as source data, the value is converted to floating point before the operation.



(K1024) → (D111, D110)

Binary converts to Floating Binary Floating

When the result is zero, zero flag activates.

Only when the source data is positive will the operation be effective. If S is negative then an error occurs and error flag SM409 is set ON, SD409=7, the instruction can’t be executed.

Note: the value must be floating numbers, otherwise the result will be error.

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	XD3, XDM
Hardware requirement	-	Software requirement	-

2. Operands

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

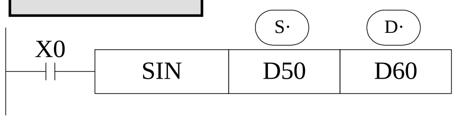
3. Suitable soft components

Word

Operand	System								Constant	Module	
	D*	FD	TD*	CD*	DX	DY	DM*	DS*	K/H	ID	QD
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.

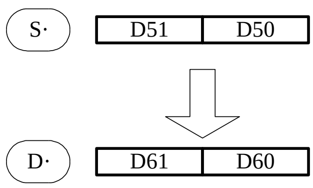
Description



(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 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	XD3, XDM
Hardware requirement	-	Software requirement	-

2. Operands

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

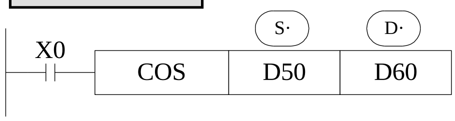
3. Suitable soft components

Word

Operand	System								Constant	Module	
	D*	FD	TD*	CD*	DX	DY	DM*	DS*	K/H	ID	QD
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.

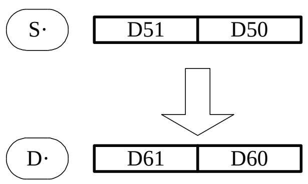
Description



(D51,D50) RAD → (D61,D60) COS

Binary Floating   Binary Floating

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



RAD value (angle× π/180)

Assign the binary floating value

COS value

Binary Floating

4-9-10． TAN [TAN]

1. Summary

TAN [TAN]			
16 bits	-	32 bits	TAN
Execution condition	Normally ON/OFF, rising/falling edge	Suitable Models	XD3, XDM
Hardware requirement	-	Software requirement	-

2. Operands

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

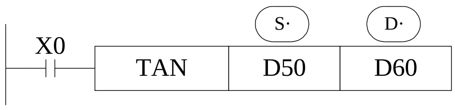
3. Suitable soft components

Word

Operand	System								Constant	Module	
	D*	FD	TD*	CD*	DX	DY	DM*	DS*	K/H	ID	QD
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.

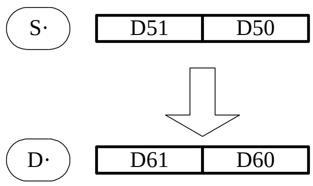
Description



(D51,D50) RAD → (D61,D60) TAN

Binary Floating Binary Floating

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



RAD value (angle× π/180)

Assign the binary floating value

TAN value

Binary Floating

4-9-11． ASIN [ASIN]

1. Summary

ASIN [ASIN]			
16 bits	-	32 bits	ASIN
Execution condition	Normally ON/OFF, rising/falling edge	Suitable Models	XD3, XDM
Hardware requirement		Software requirement	-

2. Operands

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

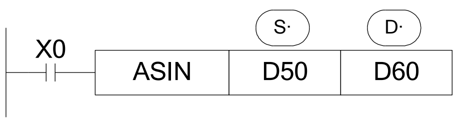
3. Suitable soft components

Word

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

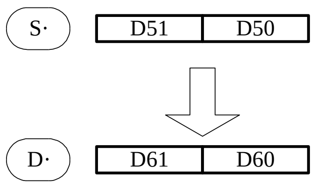
*Notes: D includes D, HD; TD includes TD, HTD; CD includes CD, HCD, HSCD, HSD; DM includes DM, DHM; DS includes DS, DHS.

Description



(D51, D50) ASIN → (D61, D60) RAD
Binary Floating Binary Floating

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



ASIN value
Binary Floating
RAD value (angle×π/180)
Assign the binary floating value

4-9-12. ACOS [ACOS]

1. Summary

ACOS [ACOS]			
16 bits	-	32 bits	ACOS
Execution condition	Normally ON/OFF, rising/falling edge	Suitable Models	XD3, XDM
Hardware requirement		Software requirement	-

2. Operands

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

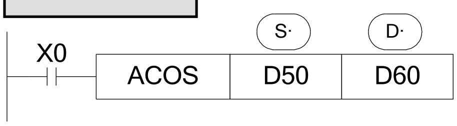
3. Suitable soft components

Word

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

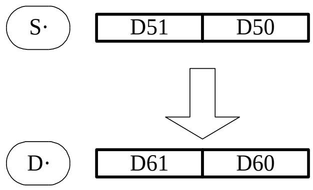
*Notes: D includes D, HD; TD includes TD, HTD; CD includes CD, HCD, HSCD, HSD; DM includes DM, DHM; DS includes DS, DHS.

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

4-9-13. ATAN [ATAN]

1. Summary

ATAN [ATAN]			
16 bits	-	32 bits	ACOS
Execution condition	Normally ON/OFF, rising/falling edge	Suitable Models	XD3, XDM
Hardware requirement		Software requirement	-

2. Operands

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

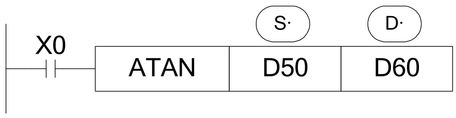
3. Suitable soft components

Word

Operand	System								Constant	Module	
	D*	FD	TD*	CD*	DX	DY	DM*	DS*	K/H	ID	QD
S	•	•			•	•	•	•	•		
D	•					•	•	•			

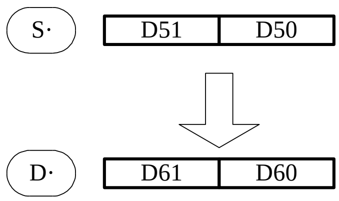
*Notes: D includes D, HD; TD includes TD, HTD; CD includes CD, HCD, HSCD, HSD; DM includes DM, DHM; DS includes DS, DHS.

Description



(D51,D50) ATAN → (D61,D60) RAD
Binary Floating Binary Floating

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



ATAN value
Binary Floating
RAD value (angle× π/180)
Assign the binary floating value

4-10．RTC Instructions

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

※1: To use the instructions, The Model should be equipped with RTC function;

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

1. Instruction Summary

Read the clock data:

Read the clock data: [TRD]			
16 bits	TRD	32 bits	-
Execution condition	Normally ON/OFF, rising/falling edge	Suitable Models	XD3, XDM
Hardware requirement		Software requirement	-

2. Operands

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

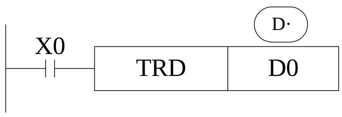
3. Suitable Soft Components

Word

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

*Notes: D includes D, HD; TD includes TD, HTD; CD includes CD, HCD, HSCD, HSD; DM includes DM, DHM; DS includes DS, DHS.

Description



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

Read PLC's real time clock according to the following format.

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	0 (Sun.)-6 (Sat.)	→	D6	Week

The RTC (real time clock) value is in BCD code format (SD013 to SD019). Please choose hex format to monitor the RTC value in XDPpro software. The value can be transformed to

decimal format by BIN instruction. After reading the RTC by TRD instruction, the value will show in decimal format.
After reading the RTC by TRD, the value becomes decimal value.
after executing TRD instruction, D0 to D6 are occupied.

4-10-2．Write Clock Data [TWR]

1. Instruction Summary

Write the clock data:

Write clock data [TWR]			
16 bits	-	32 bits	TWR
Execution condition	Normally ON/OFF, rising/falling edge	Suitable Models	XD3, XDM
Hardware requirement		Software requirement	-

2. Operands

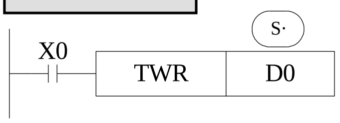
Operands	Function	Data Type
S	Write the clock data to the register	16 bits, BIN

3. Suitable Soft Components

Word	Operand	System								Constant	Module	
		D*	FD	TD*	CD*	DX	DY	DM*	DS*	K/H	ID	QD
	D	●		●	●	●	●	●	●			

*Notes: D includes D, HD; TD includes TD, HTD; CD includes CD, HCD, HSCD, HSD; DM includes DM, DHM; DS includes DS, DHS.

Description



Write the RTC value to the PLC.
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.

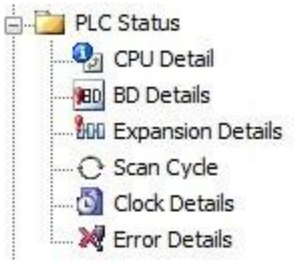
Data for clock setting

Unit	Item	Clock data
D0	Year	0-99
D1	Month	1-12
D2	Date	1-31
D3	Hour	0-23
D4	Minute	0-59
D5	Second	0-59
D6	Week	0 (Sun.)-6 (Sat.)

Special data register for real time clock t

Unit	Item
SD018	Year
SD017	Month
SD016	Date
SD015	Hour
SD014	Minute
SD013	Second
SD019	Week

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 choosing secret download program advance mode in XDPpro software, the RTC only can be changed through TWR instruction.
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.



4-10-2. Write Clock Data [TWR]

1. Instruction Summary

Compare three continuous clocks time.

Clock compare [TCMP]			
16 bits	TCMP	32 bits	-
Condition	Normally ON/OFF, rising/falling edge	Suitable model	XD3, XDM
Hardware	-	Software	-

2. operand

Operand	Function	Model
S1	The first clock soft component address	16 bits, BIN
S2	The second clock soft component address	16 bits, BIN
S3	The third clock soft component address	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

Word

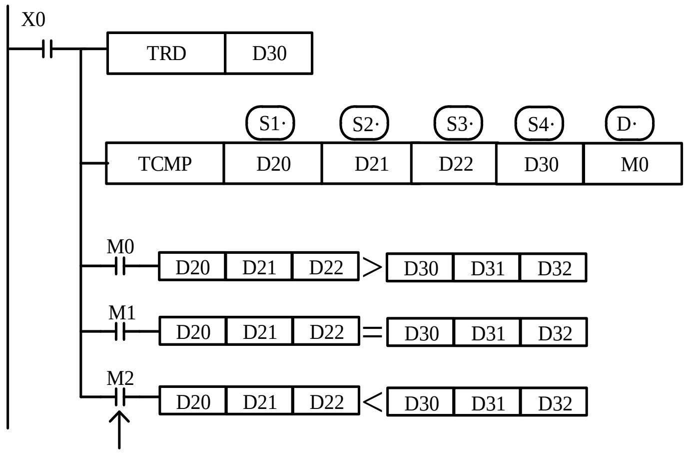
Operand	System								Constant	Module	
	D*	FD	TD*	CD*	DX	DY	DM*	DS*	K/H	ID	QD
S1	●	●			●	●	●	●	●		
S2	●	●			●	●	●	●	●		
S3	●	●			●	●	●	●	●		
S4	●	●			●	●	●	●	●		

Bit

Operand	System						
	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.

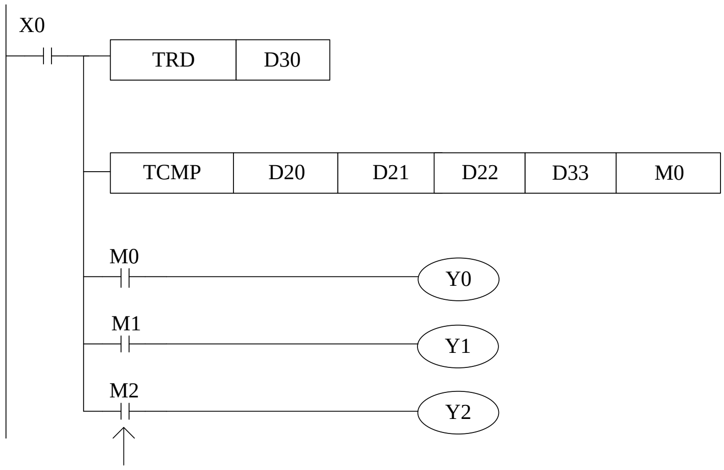
Description



Even X0=OFF to stop instruction TCMP, M0~M2 still keep the state before X0 become OFF.

TRD will read the present clock information in D30~D36 (year, month, day, hour, minute, second, week).
X0 from OFF to ON, TCMP worked. Compare the three registers starting from S4 to three registers S1, S2, S3 (year, month, day). When S1, S2, S3 is larger than S4 clock, M0 is ON. When S1, S2, S3 is equal to S4 clock, M1 is ON. When S1, S2, S3 is smaller than S4 clock, M2 is ON.
For example, the present clock is 15:32:49 7,30,2014 Wednesday. D30=14, D31=7, D32=30, D33=15, D34=32, D35=49, D36=3. If the setting time is 1,6,2015, D20=15, D21=1, D22=6, Then M0=ON. If the setting time is 7,31,2014, D20=14, D21=7, D22=31, then M1=ON. If the setting time is 6,31,2014, D20=14, D21=6, D22=31, then M2=ON.
Note: if S4 is D33, it means hour, minute, second, then S1, S2, S3 mean hour, minute, second. S4 can start from year, month, day, hour; cannot start from minute, second. The week cannot compare.

For example:



The present clock is 15:32:49 7,30,2014 Wednesday. So D30=14, D31=7, D32=30, D33=15, D34=32, D35=49, D36=3. If the setting time is 15:32:49, D20=15, D21=32, D22=49, so Y1=ON. If the setting time is 17:32:49, D20=17, D21=32, D22=49, so Y0=ON. If the setting time is 2:32:5, D20=2, D21=32, D22=5, so Y2=ON.

5 HIGH SPEED COUNTER (HSC)

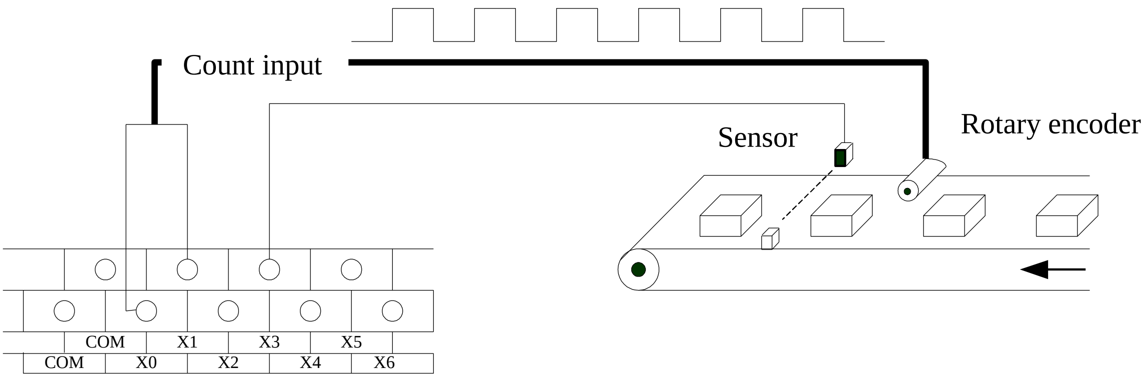
This chapter will introduce high speed counter's functions, including high speed count model, wiring method, read/write HSC value, reset etc.

Instructions List for HSC

Instruction name	Function	Instruction	Chapter
HSC read/write			
DMOV	HSC read	DMOV HSC0 D0	5-6-1
DMOV	HSC write	DMOV D4000 HSC0	5-6-2
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
CNT	24-segments single phase	CNT HSC0 K1000 D0	5-7-3
CNT_AB	24-segments AB phase	CNT_AB HSC0 K1000 D0	5-7-4
RST	HSC reset	RST HSC0	5-8

5-1. Functions Summary

XD3, XDM 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 80 KHz.

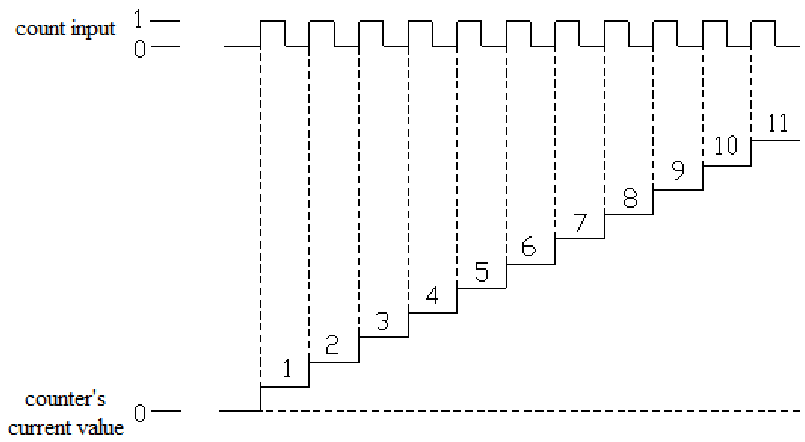


5-2. HSC Mode

XD3, XDM series high speed counter has two working mode: increasing mode and AB phase mode.

Increasing Mode

Under this mode, the count value increase at each pulse’s rising edge;

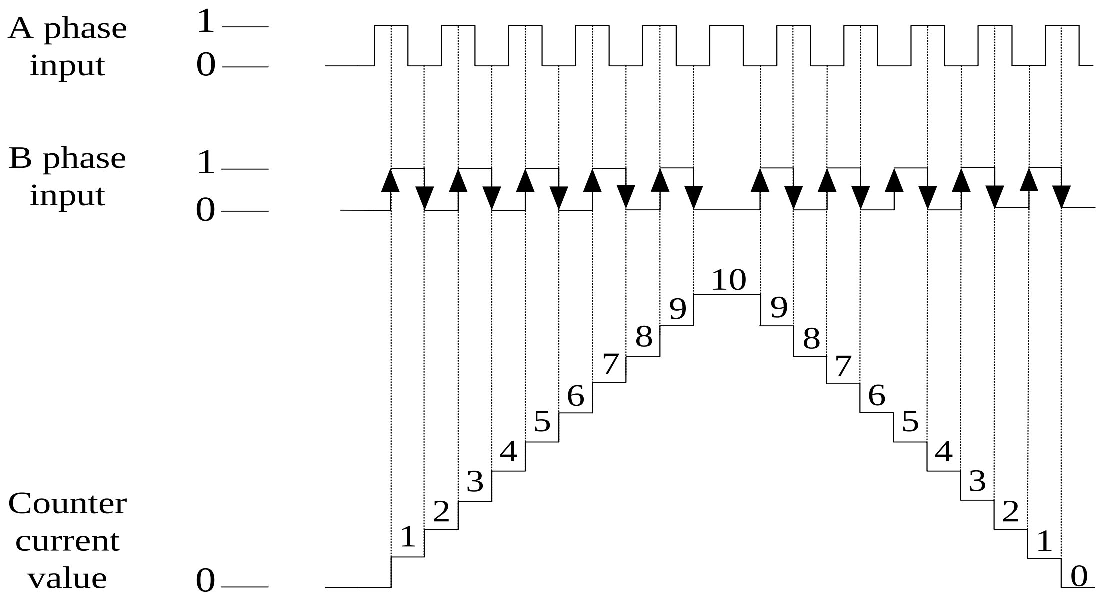


AB Phase Mode

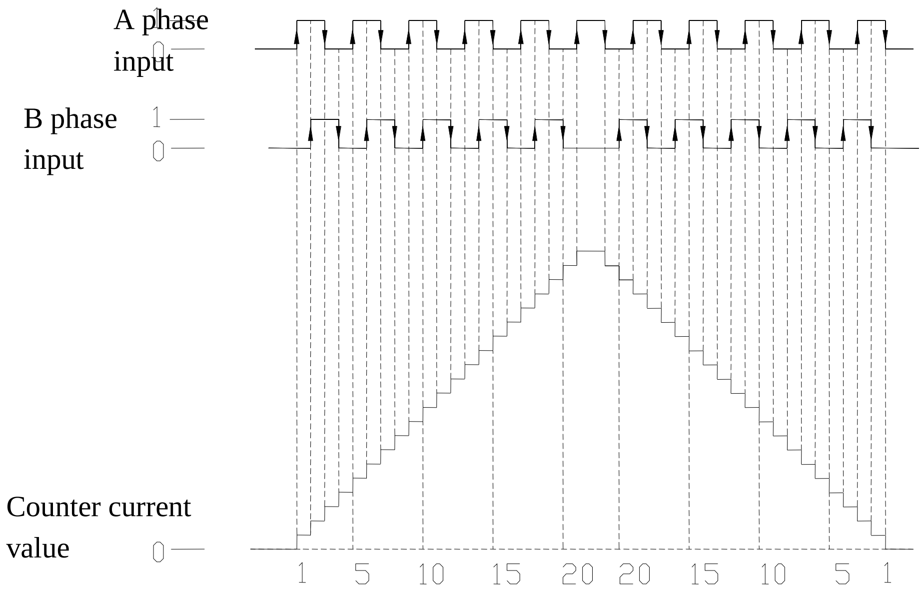
Under this mode, the HSC value increase or decrease according to two differential signal (A phase and B phase). According to the multiplication, we have 1-time frequency and 4-time frequency, but the default count mode is 4-time mode.

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

1-time Frequency



4-time Frequency



5-3. HSC Range

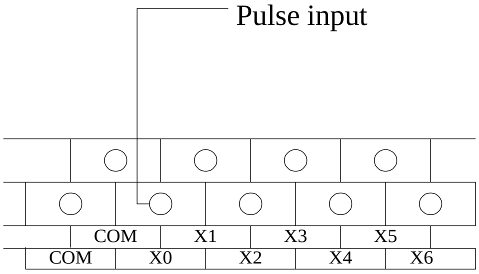
HSC's count range is: -2,147,483,648 ~ +2,147,483,647. If the count value overflows this range, then overflow or underflow appears;

Overflow means the count value jumps from +2,147,483,647 to -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.

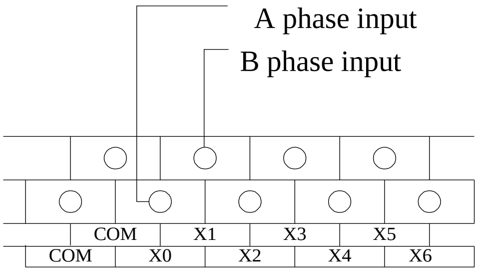
5-4. HSC Input Wiring

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 (counter HSC0)



AB phase mode (counter HSC0)



5-5. HSC ports assignment

Each letter's Meaning:

U	A	B	Z
Pulse input	A phase input	B phase input	Z phase pulse catching

Normally, X0 and X1 can accept 80 KHz and 50 KHz pulse in single phase mode and AB phase mode. Other terminals can accept 10 KHz and 5 KHz pulse in single phase mode and AB phase mode. X can use as normal input terminals when there are no high speed pulses input. In the following table, Frequency time 2 means 2-time frequency; 4 means 4-time frequency; 2/4 means 2-time and 4-time frequency.

XD3-16T/R/RT-E												
	Increasing mode							AB phase mode				
	HSC0	HSC2	HSC4	HSC6	HSC8	HSC10	HSC12	HSC0	HSC2	HSC4	HSC6	HSC8
Max frequency	80K	10K						50K	5K			

Frequency time								2/4	2/4			
Counter interruption	√	√						√	√			
X000	U							A				
X001								B				
X002								Z				
X003			U						A			
X004									B			
X005									Z			
X006												
X007												
X010												
X011												

XD3-24/32/48/60/T/R/RT-E												
	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		
Frequency time								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												

XDM-24T4/32T4/60T4-E												
	Increasing mode							AB phase mode				
	HSC0	HSC2	HSC4	HSC6	HSC8	HSC10	HSC12	HSC0	HSC2	HSC4	HSC6	HSC8
Max frequency	80K	80K	80K	80K				50K	50K	50K	50K	
Frequency time								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	

XDM-60T10 -E

	Incremental mode											
	HSC0	HSC2	HSC4	HSC6	HSC8	HSC10	HSC12	HSC14	HSC16	HSC18	HSC20	HSC22
Max frequency	80K	80K	80K	80K	80K	80K	80K	80K	10K	10K		
Frequency time												
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												

XDM-60T10 -E

	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	5K	5K		
Frequency time	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. Read/Write HSC value

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

5-6-1. Read HSC value [DMOV]

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	XD3, XDM
Hardware requirement		Software requirement	-

Operands

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

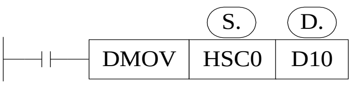
Suitable Soft Components

word

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

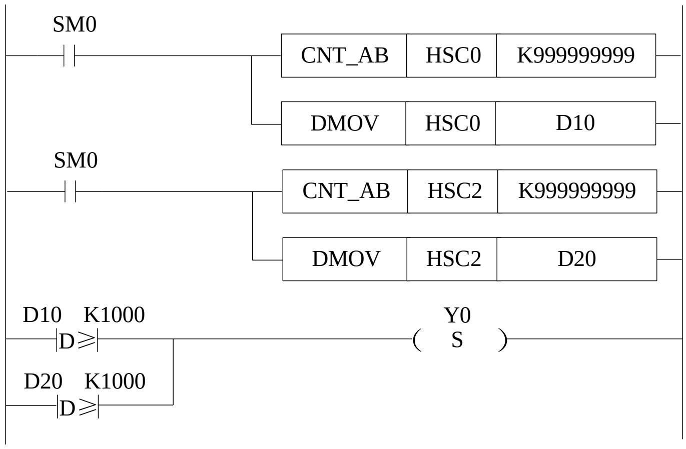
*Notes: D includes D, HD; TD includes TD, HTD; CD includes CD, HCD, HSCD, HSD; DM includes DM, DHM; DS includes DS, DHS.

FUNCTIONS AND ACTIONS



Move the counting value of HSC (dword) to the target register when the condition activates. DMOV will send the counting value to data register; this will improve the counting value precision.

Program example:



5-6-2. Write HSC value [DMOV]

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	XD3, XDM
Hardware requirement		Software requirement	-

operands

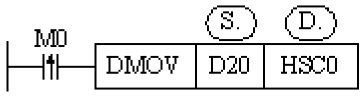
Operands	Function	Type
S	Specify HSC code	32 bits, BIN
D	Specify the read/written register	32 bits, BIN

suitable soft components

	Operand	System								Constant	Module	
word		D*	FD	TD*	CD*	DX	DY	DM	DS*	K/H	ID	QD
	S					●						
	D	●										

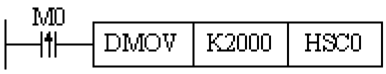
*Notes: D includes D, HD; TD includes TD, HTD; CD includes CD, HCD, HSCD, HSD; DM includes DM, DHM; DS includes DS, DHS.

FUNCTIONS AND ACTIONS



Move the data register value to HSC when the condition activates.
The HSC cannot join all the instructions except DMOV. Please use DMOV to send the HSC value to normal register then it can join other instructions.

Program example:



5-7. HSC Reset Mode

5-7-1. HSC no 100-segment single phase [CNT]

Summarization

HSC no 100-segment single phase counting instruction.

HSC no 100-segment single phase [CNT]			
16-bit instruction	-	32-bit instruction	CNT
Execution condition	Normal ON/OFF	Suitable model	XD3, XDM
Hardware requirements	-	Software requirements	-

Operand

Operand	Function	Type
S1	Set the HSC (for example: HSC0)	32 bits, BIN
S2	Set the compare value (K100, D0)	32 bits, BIN

Suitable soft components

Word

Operand	System								Constant	Module	
	D*	FD	TD*	CD*	DX	DY	DM	DS*	K/H	ID	QD
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.

Description

SM0	CNT	HSC0 (S1.)	D20 (S2.)

When HSC0 is counting, compare HSC counting value to D20, if they are equal, set on coil HSC0.

Program example:

M0	CNT	HSC0	K20000

5-7-2． HSC no 100-segment AB phase [CNT_AB]

Summarization

HSC no 100-segment AB phase counting instruction.

HSC no 100-segment AB phase [CNT_AB]			
16 bits instruction	-	32 bits instruction	CNT_AB
Execution condition	Normal ON/OFF	Suitable model	XD3, XDM
Hardware requirements	-	Software requirements	-

Operand

Operand	Function	Type
S1	Set the HSC (such as:HSC0)	32 bits, BIN
S2	Set the compare value (such as: K100, D0)	32 bits, BIN

Suitable soft components

Word

Operand	System								Constant	Module	
	D*	FD	TD*	CD*	DX	DY	DM	DS*	K/H	ID	QD
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.

Description

SM0	CNT_AB	HSC0 (S1.)	D20 (S2.)

When HSC0 is AB phase counting, compare HSC counting value to D20, if they are equal, set on coil HSC0.

Program example:

M0			
─┤├─	CNT_AB	HSC0	K20000

5-7-3．HSC 100-segment single phase [CNT]

Summarization

HSC 100-segment single phase counting instruction.

HSC 100-segment single phase [CNT]			
16 bits instruction	-	32 bits instruction	CNT
Execution condition	Normal ON/OFF	Suitable model	XD3, XDM
Hardware requirements	-	Software requirements	-

Operand

Operand	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 24-segment value	32 bits, BIN

Suitable soft components

	Operand	System								Constant	Module	
		D*	FD	TD*	CD*	DX	DY	DM	DS*	K/H	ID	QD
word	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.

Description

SM0	S1.	S2.	S3.
─┤├─ CNT	HSC0	D4000	D4100

When HSC0 is single phase counting, compare the HSC value to D4100, if HSC value is equal to 24-segment value, it will produce HSC interruption.

Program example:

M0			
─┤├─ CNT	HSC0	D4000	D4100

5-7-4. HSC 100-segment AB phase [CNT_AB]

Summarization

HSC 100-segment AB phase counting instruction.

HSC 100-segment AB phase [CNT_AB]			
16 bits instruction	-	32 bits instruction	CNT_AB
Execution condition	Normal ON/OFF	Suitable model	XD3, XDM
Hardware requirments	-	Software requirements	-

Operand

Operand	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 24-segment value	32 bits, BIN

Suitable soft components

Word

Operand	System								Constant	Module	
	D*	FD	TD*	CD*	DX	DY	DM	DS*	K/H	ID	QD
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.

Description

SM0		S1.	S2.	S3.
─┤├─	CNT_AB	HSC0	D4000	D4100

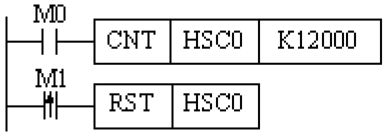
When HSC0 is AB phase counting, compare the HSC value to D4100, if HSC value is equal to 24-segment value, it will produce HSC interruption.

Program example:

M0				
─┤├─	CNT_AB	HSC0	D4000	D4100

5-8. AB Phase counter multiplication setting

HSC is software reset mode.



When M0 is ON, HSC0 is counting the pulse from X0 terminal. When M1 changes from OFF to ON, reset the HSC0, the counting value will be cleared.

5-9. AB Phase mode frequency time setting

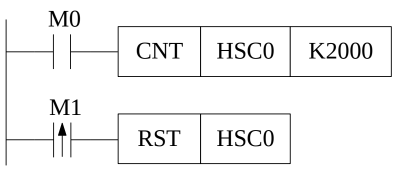
The frequency time can be set through special FLASH register for AB phase mode counting.

FLASH register	Function	Set value	Meaning
SFD320	HSC0 frequency times	2	2-time frequency
		4	4-time frequency
SFD321	HSC2 frequency times	2	2-time frequency
		4	4-time frequency
SFD322	HSC4 frequency times	2	2-time frequency
		4	4-time frequency
SFD323	HSC6 frequency times	2	2-time frequency
		4	4-time frequency
SFD324	HSC8 frequency times	2	2-time frequency
		4	4-time frequency
SFD325	HSC10 frequency times	2	2-time frequency
		4	4-time frequency
SFD326	HSC12 frequency times	2	2-time frequency
		4	4-time frequency
SFD327	HSC14 frequency times	2	2-time frequency
		4	4-time frequency
SFD328	HSC16 frequency times	2	2-time frequency
		4	4-time frequency
SFD329	HSC18 frequency times	2	2-time frequency
		4	4-time frequency

5-10. HSC Example

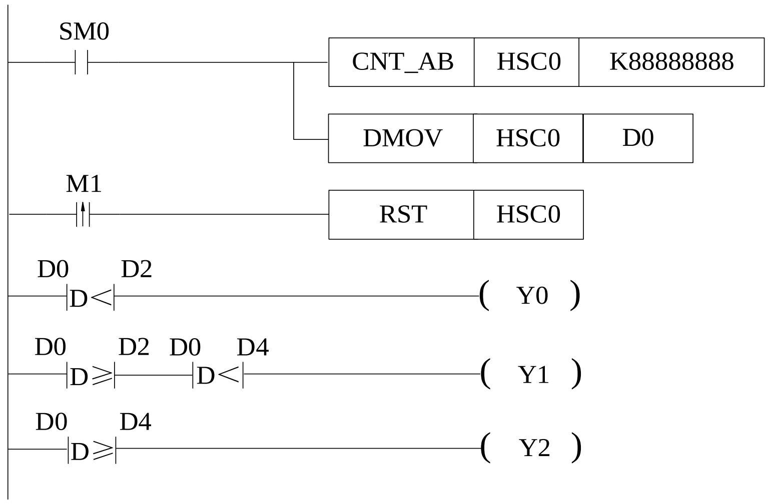
We make XD3-60 PLC as an example to introduce HSC programming method.

Increasing mode:



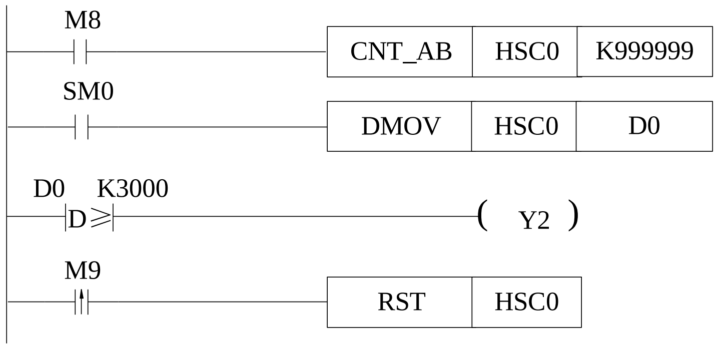
When M0 is ON, HSC0 counts the pulses of X0.

When the rising edge of M1 is coming, reset HSC0.

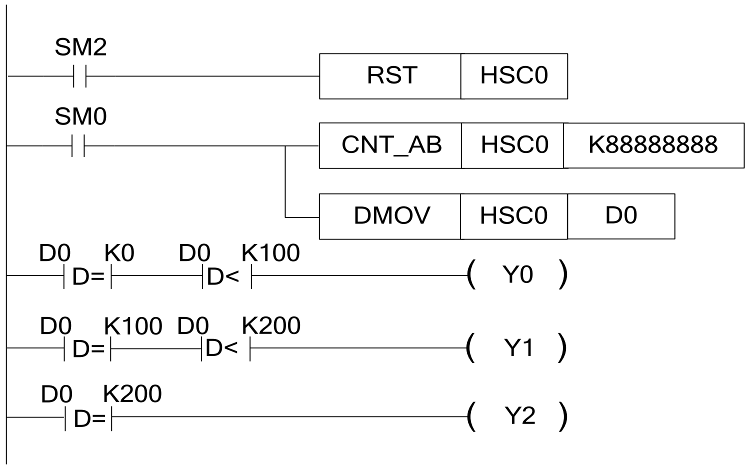


When SM0 is ON, set the count value of HSC to K88888888, read the HSC0 counting value to register D0 (dword).
When HSC0 counting vaue is less than D2 (D0<D2), Y0 is ON; when HSC0 counting value is in the range of D2 to D4 (D2≤ D0 <D4), Y1 is ON; when HSC0 counting value is larger than D4 (D0 ≥ D4), Y2 is ON.
When the rising edge of M1 is coming, reset HSC0.

AB phase mode:



When M8 is ON, HSC0 starts counting. The pulse input from X0 (A phase) and X1 (B phase).
If the counting value is over 3000, Y2 is ON.
When the rising edge of M9 is coming, reset HSC0.



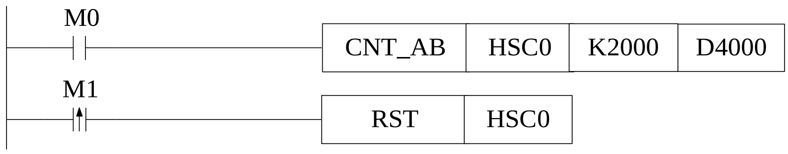
When the rising edge of SM2 is coming, reset HSC0 and the counting value is cleared.
When SM0 is ON, HSC0 starts counting; the set counting value is 88888888.
If the counting value is in the range of 0 to 100 ($0 \leq D0 < 100$), Y0 is ON; if the counting value is in the range of 100 to 200 ($100 \leq D0 < 200$), Y1 is ON; if the counting value is larger than 200 ($D0 \geq 200$), Y2 is ON.

5-11. HSC interruption

Some HSC (refer to chapter 5-5) has 100 segments 32-bit preset value. When the HSC difference value is equal to 100-segment preset value, the interruption will be produced.

5-11-1. Interruption instruction

(For the program about interruption, please refer chapter 5-11-4)




```
LD      M0                //HSC activates condition M0 (interruption count condition)
CNT_AB HSC0 K2000 D4000   //HSC value and set the start address of 100-segment
LDP     M1                //activate condition of HSC reset
RST     HSC0              //HSC and 100-segment reset (interruption reset)
```

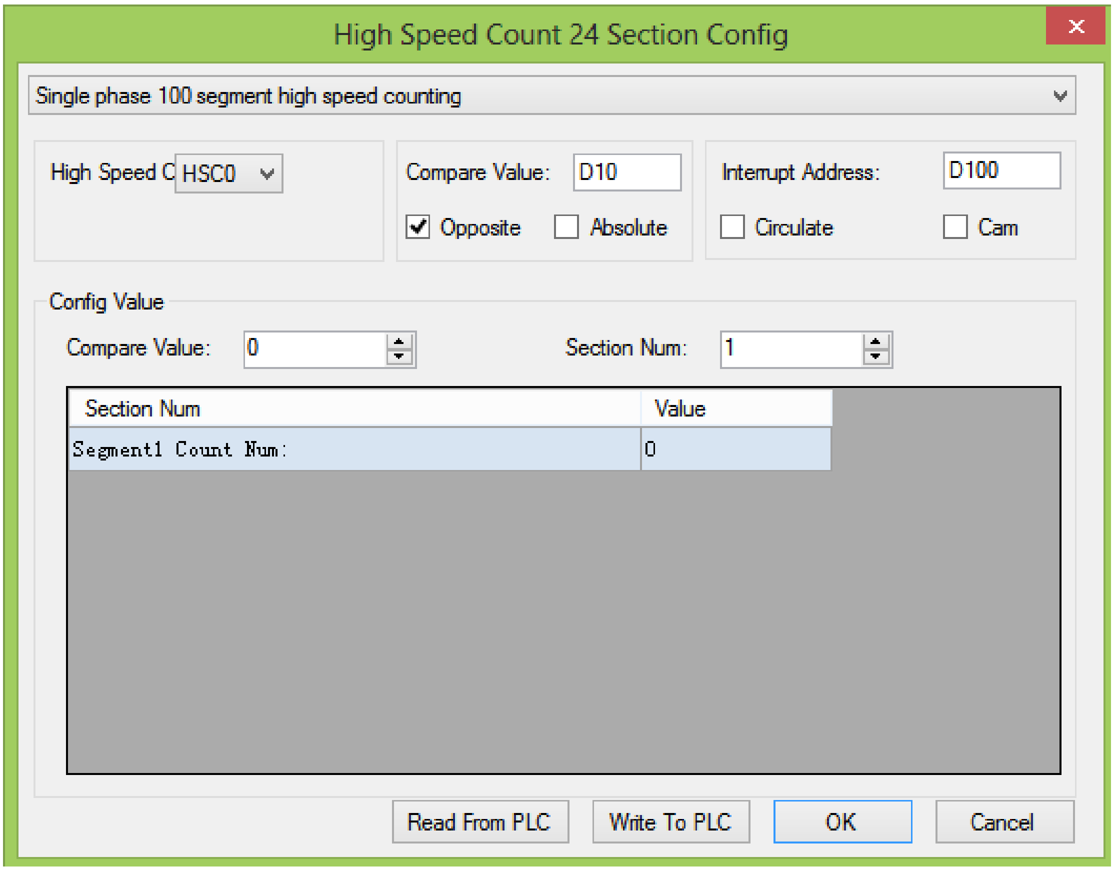
As shown in the above graph, data register D4000 is the start address of 100-segment preset value. The following addresses will save each 100-segment preset value in DWORD form.
Please pay attention when using HSC:
If certain preset value is 0, it means count interruption end at this segment;
Set the interruption preset value but not write the correspond interruption program is not allowed;
100-segment interruption of HSC occurs in order. If the first segment interruption doesn't happen, then the second segment interruption will not happen;

HSC CAM: after setting the 100-segment preset value, choose the HSC CAM function. When HSC counting value is equal to one of the preset value, the corresponding interruption will be executed. The same HSC CAM can be repeated when HSC counting value changes.

100-segment single phase and AB phase, HSC absolute and relative mode (refer to special register SFD330, SFD331), HSC CAM (refer to special register SFD332) can be configured in the following way:

Click the 

high speed counter config in XDPpro software. And configure the parameters in it.



5-11-2. 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
HSC0	I2000~I2099
HSC2	I2100~I2199
HSC4	I2200~I2299
HSC6	I2300~I2399
HSC8	I2400~I2499
HSC10	I2500~I2599
HSC12	I2600~I2699
HSC14	I2700~I2799

HSC16	I2800~I2899
HSC18	I2900~I2999

Define the preset value

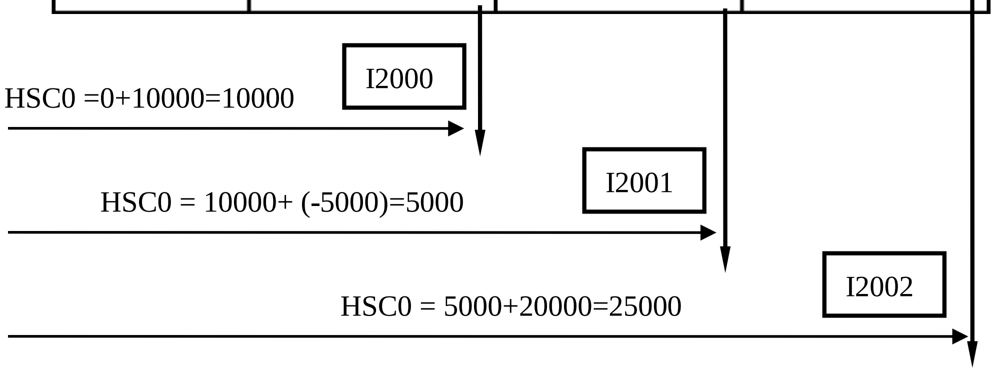
HSC 100-segment preset value is the difference value. When the counting value is equal to the difference of counting value and last preset value, it will produce the interruption. N interruption flags correspond to N interruption preset values. The (N+1) preset value is 0.

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 start to count, the counter's current value is 10000, it generates the first interruption I2000; the counter's current value is 5000, it generates the second interruption I2001; the counter's current value is 25000, it generates the third interruption I2002.

See graph below:

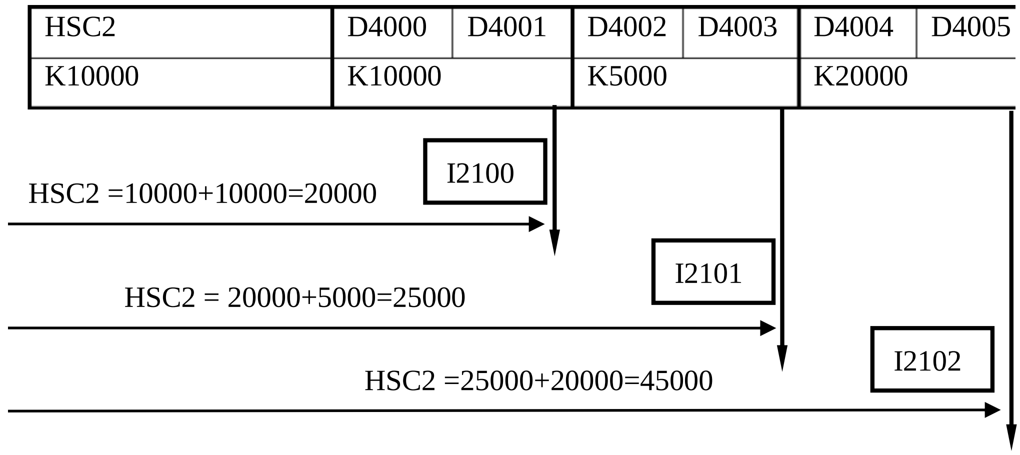
HSC0	D4000	D4001	D4002	D4003	D4004	D4005
K0	K10000		K-5000		K20000	



Example 2:

HSC2 current value is 10000, the segment one preset value is 10000, the preset value in segment 2 is 5000, the preset value in segment 3 is 20000. When start to count, the counter's current value is 20000, it generates the first interruption I2100; the counter's current value is 25000, it generates the second interruption I2101; the counter's current value is 45000, it generates the third interruption I2102.

See graph below:



5-11-3. 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.

reset the HSC

Reboot the HSC activate condition

Mode 2: Continuous loop

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

continuous back-forth movement

Generate cycle interruption according to the defined pulse

Via setting the special auxiliary relays SFD331, users can set the HSC interruption to be single loop mode or continuous loop mode. The continuous loop mode is only suitable for the relative counting. The detailed assignment is show below:

Address	HSC	Setting
Bit0	100 segments HSC interruption cycle (HSC0)	
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)	0: single loop
Bit5	100 segments HSC interruption cycle (HSC10)	1: continuous loop
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)	

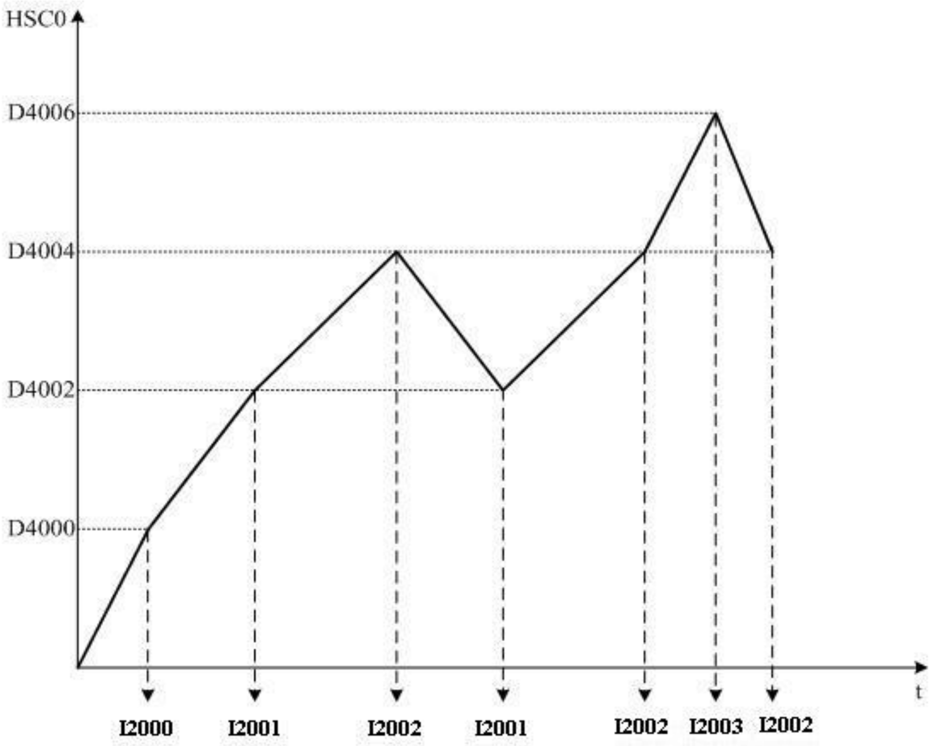
5-11-4. CAM function of high speed counter interruption

After setting all the interruption counter value, choose the high speed counter CAM function. When the high speed counter value is the same to the setting interruption value, execute the high speed counter interruption (the same to 100-segment high speed counter interruption flag). If the high speed counter value changes, it can executes the same CAM high speed interruption repeatly.

High speed counter CAM can perform the circular order interruption of normal electronic CAM, and single-cycle positive-negative-single-point multi-time interruption. It is applied in high-speed winding machine, packing machine and so on.

Example: CAM function

Four numbers are stored in the registers starting from D4000 (dword). Then the HSC0 starts to count. When the counting value is equal to any of the four numbers, it will produce the interruption. Please see the following diagram:



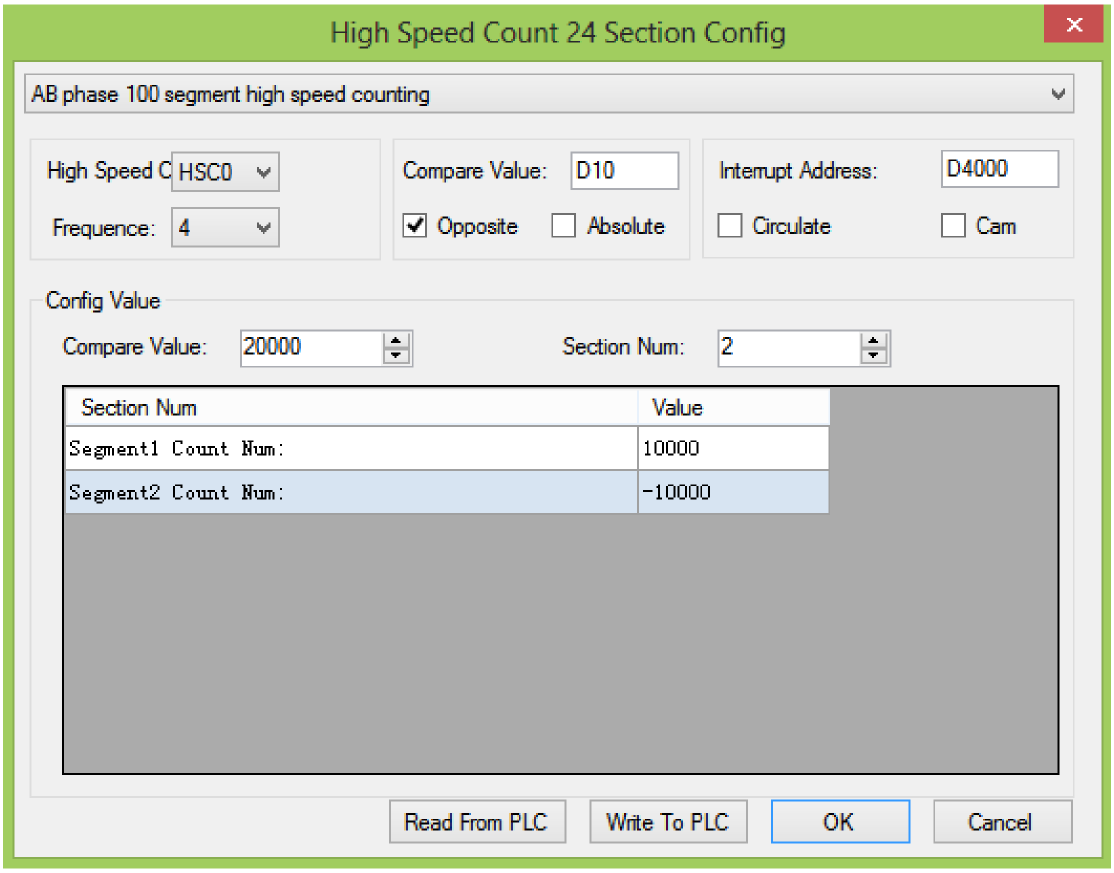
5-11-5. Application of HSC interruption

Application 1:

When M0 is ON, HSC0 starts counting from D4000. When it reaches the preset value, the interruption is produced. When the rising edge of M1 is coming, clear the HSC0.

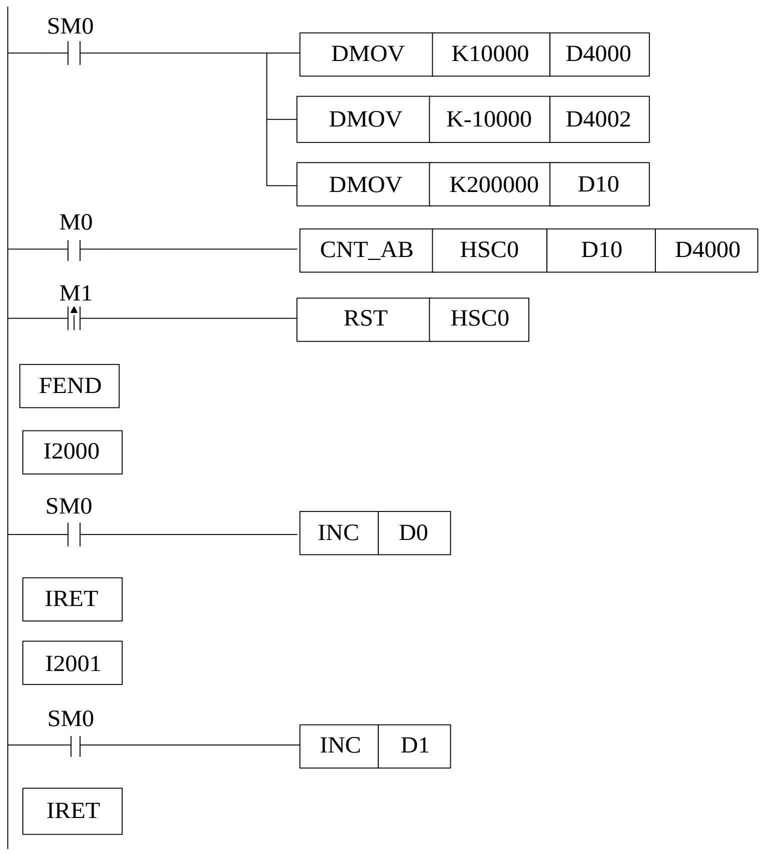
Method 1:

Configure the parameters through XDPpro software:



Configure item	**Function**
HSC	Choose HSC, the range is from HSC0 to HSC18
Frequency	Choose the HSC frequency times (2-time or 4-time)
Compare value	The value can be register or constant
Opposite 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	HSC CAM function available

Method 2: make the program

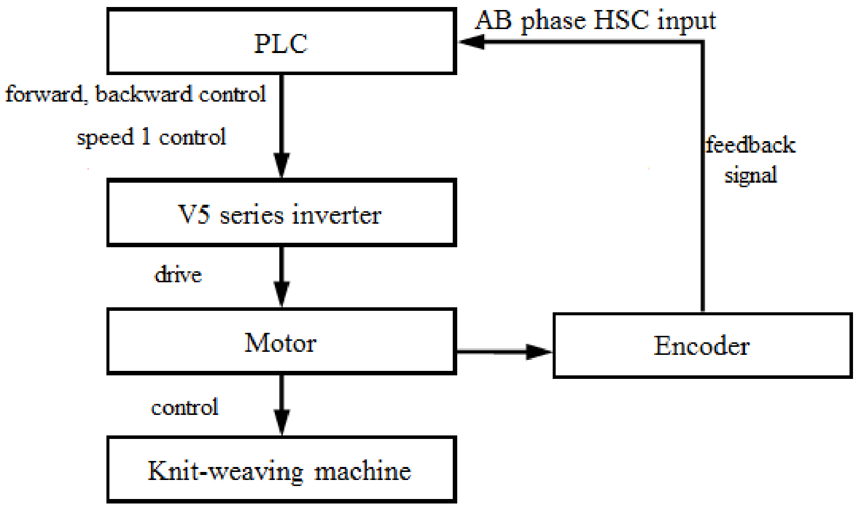


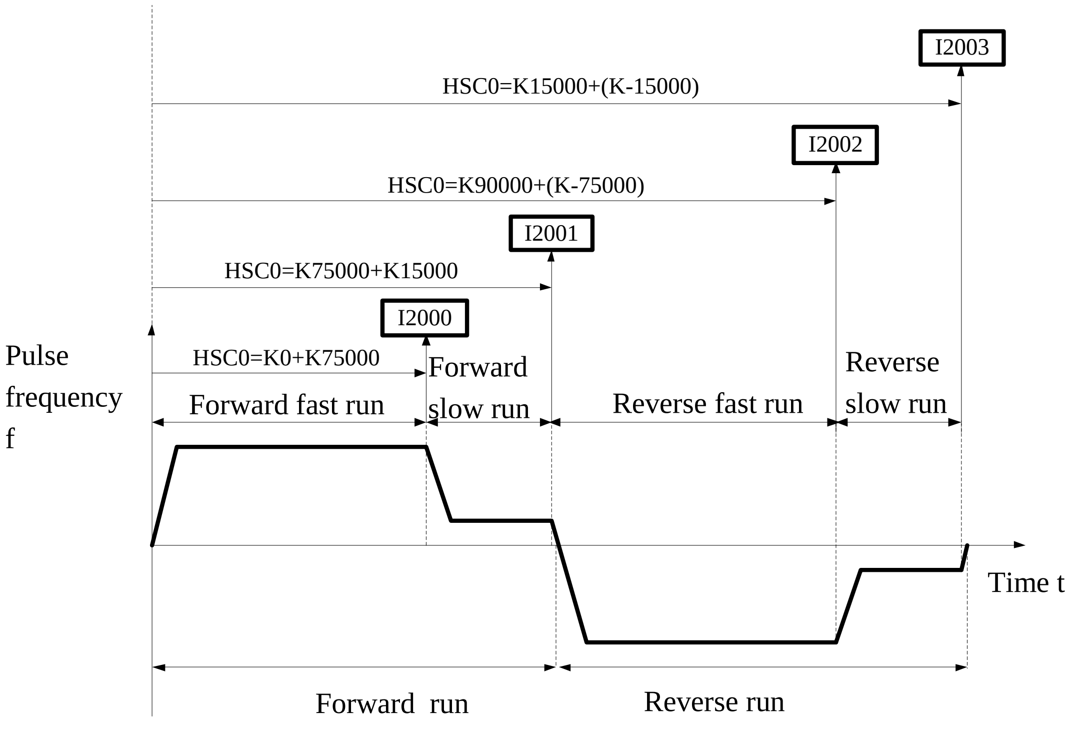
Instruction:

```
LD       SM0                    //SM000 is normal ON coil
DMOV K10000 D4000               //segment one preset value D4000 is 10000
DMOV K-10000 D4002              //segment 2 preset value D4002 is -10000
DMOV K200000   D10              //set HSC compare value
 LD      M0                     //HSC activate condition M0
CNT_AB  HSC0 D10 D4000          //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 normal ON coil
INC      D0                     //D0= D0+1
IRET                            //interruption return flag
I2001                           //segment 2 interruption flag
LD       SM0                    //SM0 is normal ON coil
INC      D1                     //D1= D1+1
IRET                            //interruption return flag
```

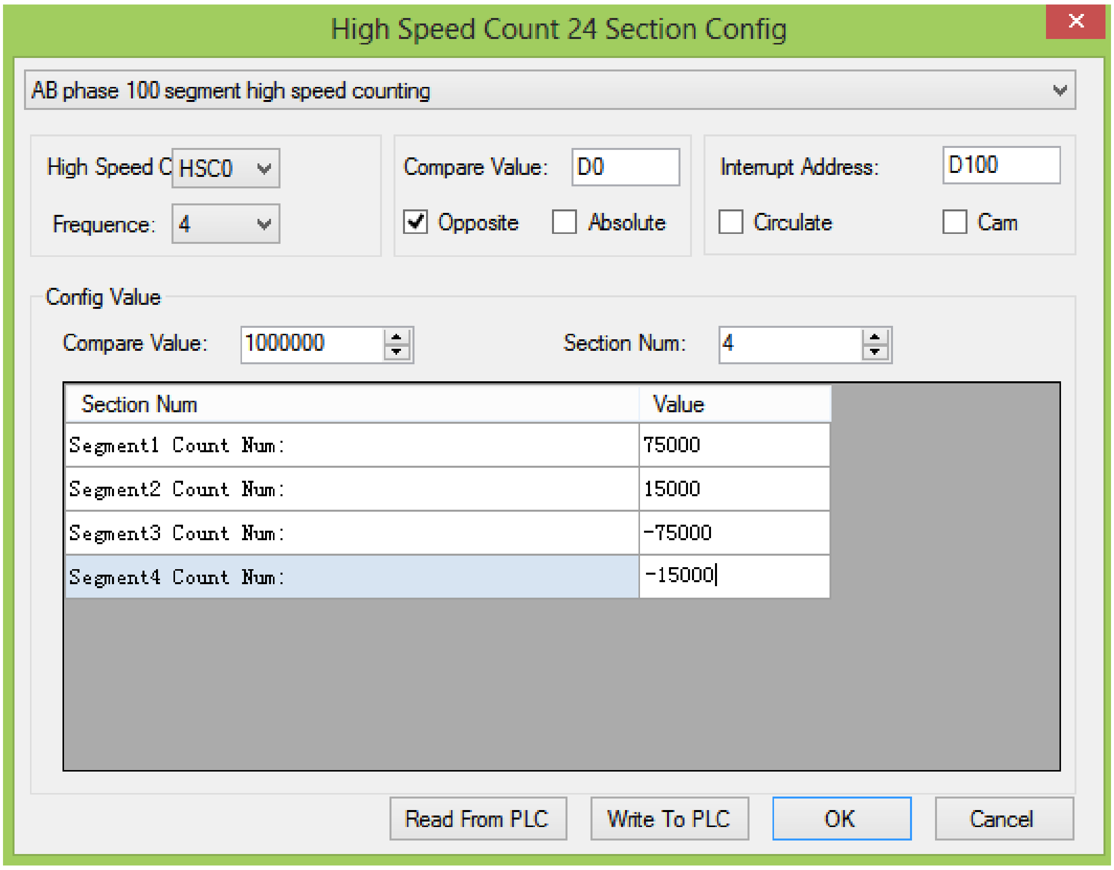
Application 2: knit-weaving machine (continuous loop mode)

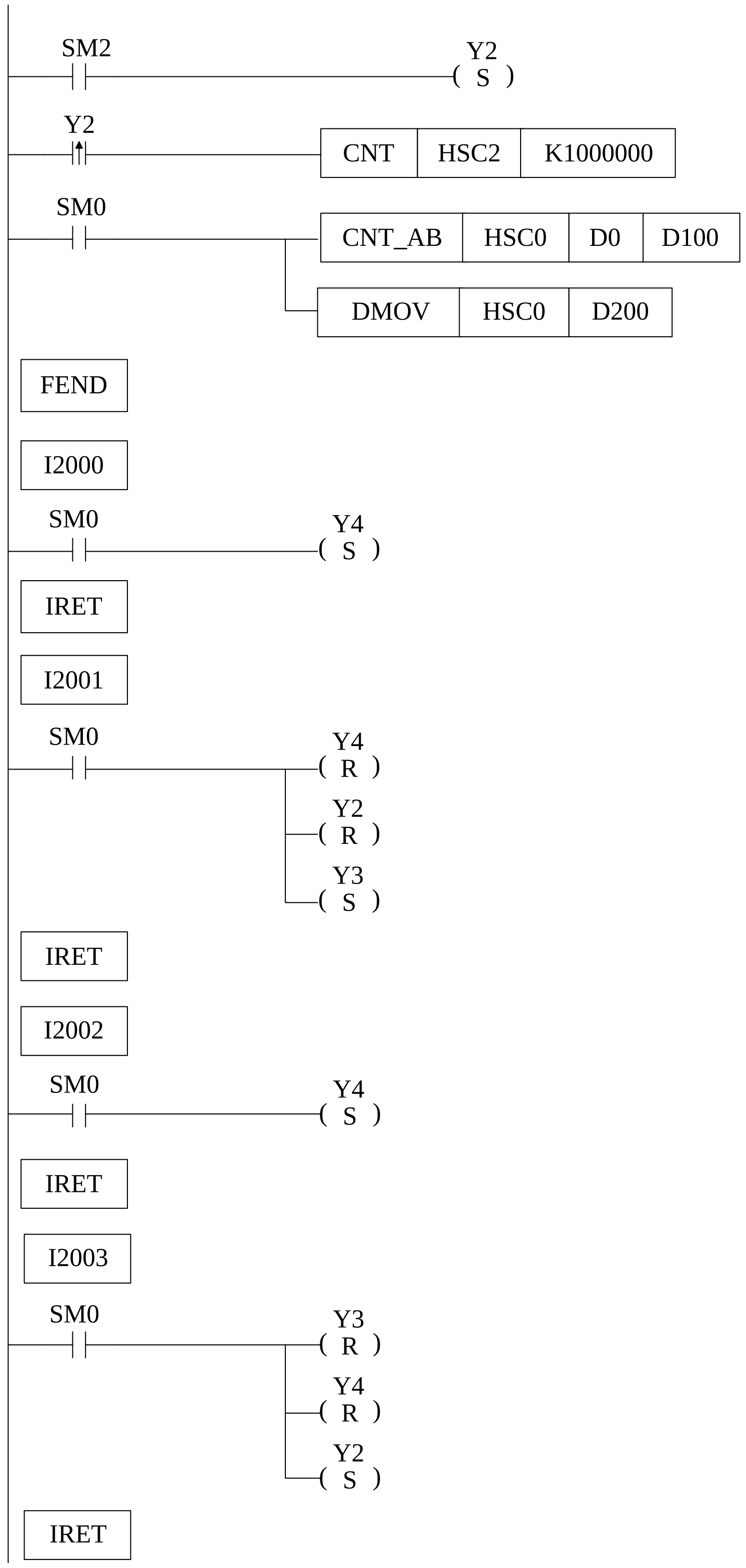
The machine principle: Control the inverter via PLC, thereby control the motor. Meantime, via the feedback signal from encoder, control the knit-weaving machine and the precise position.





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;


High Speed Count 24 Section Config
AB phase 100 segment high speed counting
High Speed C HSC0
Frequence: 4
Compare Value: D0
Opposite
Absolute
Interrupt Address: D100
Circulate
Cam
Config Value
Compare Value: 1000000
Section Num: 4
Section Num
Value
Segment1 Count Num: 75000
Segment2 Count Num: 15000
Segment3 Count Num: -75000
Segment4 Count Num: -15000
Read From PLC
Write To PLC
OK
Cancel



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 PULSE OUTPUT

This chapter will introduce the pulse function of XD3, XDM series PLC. The content includes pulse output instructions, input/output wiring, and notes, related coils and registers etc.

Pulse Output Instructions List

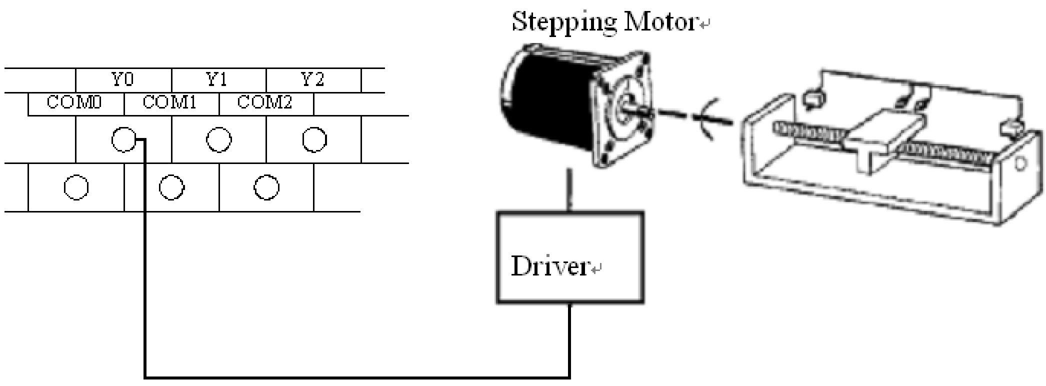
Instruction name	Function	Instruction	Chapter
Pulse output			
PLSR	Multi-segment pulse output	PLSR S0 S1 S2 D0	6-2-1
PLSF	Variable pulse output	PLSF S0 S1 S2 D0	6-2-2
ZRN	Return to mechanical origin	ZRN S0 D0	6-2-3
PLSMV	Refresh pulse quantity	PLSMV Yn	6-2-4
STOP	Stop pulse	STOP S0 S1	6-2-5

6-1. Functions Summary

Generally, XD3 series PLC have two pulse output channels. XDM series PLC has 4~10 channels pulse output. The pulse output modes include single direction pulse output without acceleration/deceleration, single direction pulse output with acceleration/deceleration, multi-segment double direction pulse output. The pulse frequency can up to 200 KHz.

Pulse output terminal:

PLC model	Pulse channel	Pulse output terminal
XD3-16T/RT, XD3-24T/RT, XD3-32T/RT, XD3-48T/RT, XD3-60T/RT	2	Y0, Y1
XDM-24T4, XDM-32T4, XDM-60T4	4	Y0, Y1, Y2, Y3
XDM-60T10	10	Y0, Y1, Y2, Y3, Y4, Y5, Y6, Y7, Y10, Y11



※1: Please use transistor output terminal for pulse output. Such as XD3-14T-E, XD3-60T-E.
※2: the pulse output terminal please see the above table. The direction terminal can be any one in the rest of the output terminals.
※3: the direction terminal will keep the pulse direction state after the pulse sending finished. For example, the pulse direction is positive, the terminal will be ON after pulse sending finished. If the pulse has direction output next time, the pulse will control the direction terminal. If the pulse without direction output next time, the direction terminal state can be controlled in the program.

6-2. Pulse Output Types and Instructions

6-2-1. Multi-segment pulse output [PLSR]

Instruction summarization

Multi-segment pulse output

Multi-segment pulse output [PLSR]			
16 bits instruction		32 bits instruction	PLSR
Execution condition	Rising/falling edge	Suitable model	XD3, XDM
Hardware requirements	-	Software requirements	-

Operand

Operand	Function	Type
S0	Pulse parameter start address	
S1	User parameter start address	
S2	System parameter block (1 ~4)	
D	Pulse output port	

Suitable soft components

Word

Operand	System								Constant	Module	
	D*	FD	TD*	CD*	DX	DY	DM*	DS*	K/H	ID	QD
S0	●	●	●	●	●	●	●	●			
S1	●	●	●	●	●	●	●	●			
S2	●	●							●		

Bit

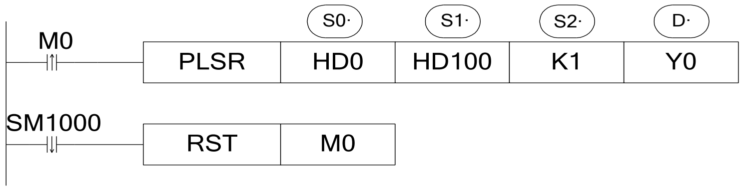
Operand	System						
	X	Y	M*	S*	T*	C*	Dn.m
D		●					

*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 and HT; C includes C and HC.

Description

《Instruction》



Pulse parameter address:

Address	**Contents**	**Mark**
S0+0 (dword)	Pulse segment quantity (1~100)	
S0+2 (8 words)	Reserved (8 words)	
S0+10 (dword)	Pulse frequency	Segment 1
S0+12 (dword)	Pulse quantity	
S0+14	bit15~bit8: wait condition 00: pulse sending end 01: wait time 02: wait signal 03: ACT time 04: EXT signal 05: EXT signal or pulse sending end bit7~bit0: wait condition register type 00: constant 01: D 02: HD 03: FD 04: X 05: M 06: HM	
S0+15 (dword)	Constant/register(waiting condition)	

S0+17	bit7~bit0: jump register type 00: constant 01: D 02: HD 03: FD	
S0+18 (dword)	Constant/register (jump register)	
…	…	…
S0+N*10+0 (dword)	Pulse frequency	
S0+N*10+2 (dword)	Pulse quantity	
S0+N*10+4	Wait condition, wait condition register type	
S0+N*10+5 (dword)	Constant or register (waiting condition)	Segment N
S0+N*10+7	Jump type, jump register type (waiting condition)	
S0+N*10+8 (dword)	Constant or register (jump register)	

Note:

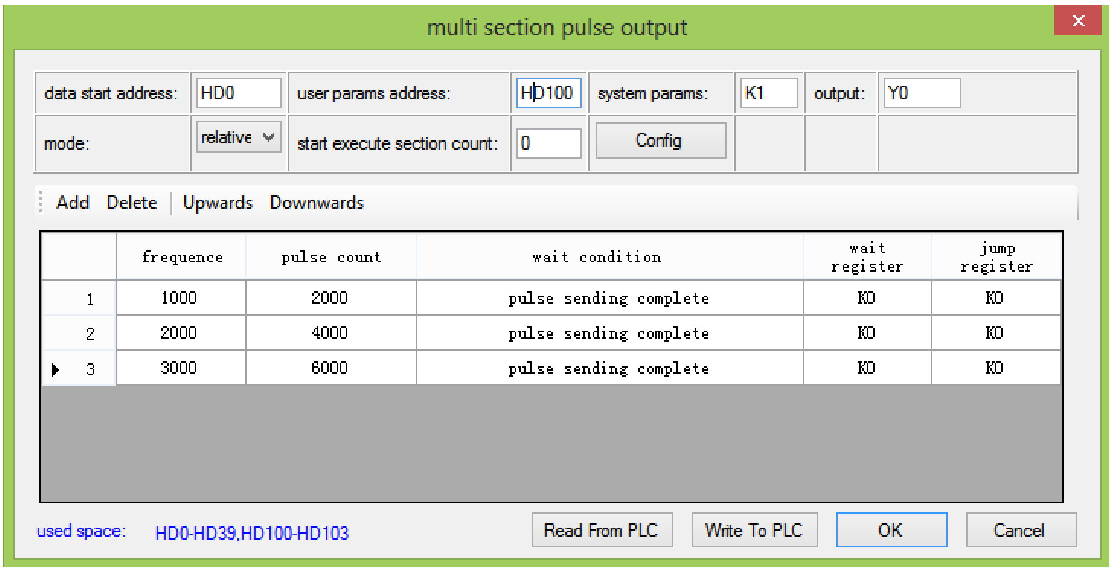
1. pulse frequency is positive value, the value increasing means acceleration, the value decreasing means deceleration, it is not related to the pulse direction. Pulse numbers can be positive or negative value, negative pulse number means reverse direction pulse.

2. The setting of PLSR each segment pulse frequency is effective immediately when the pulse is sending.

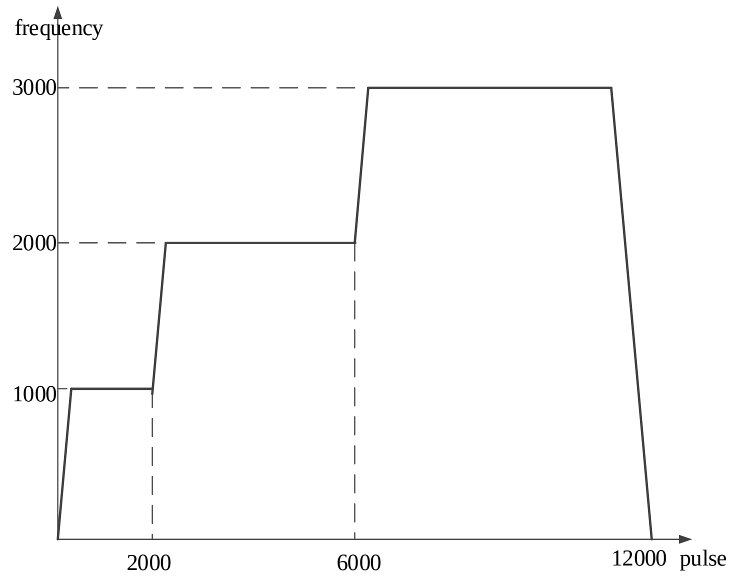
Waiting condition

Pulse sending end

Jump to the certain segment after executing this segment of pulse.



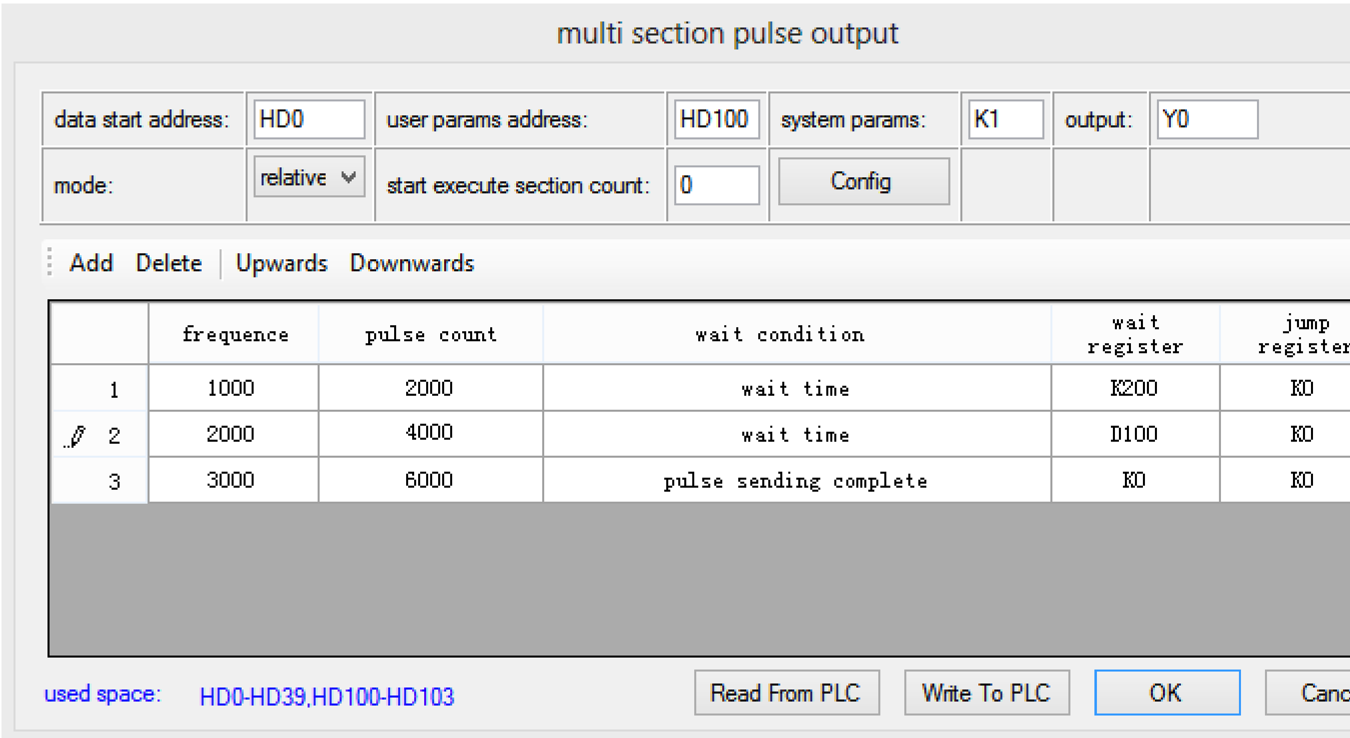
When the PLSR is triggered by edge, it sends 2000 pulses at 1000Hz, then jump to segment 2. It will send 4000 pulses at 2000Hz, then jump to segment 3. It will send 6000 pulses at 3000Hz.



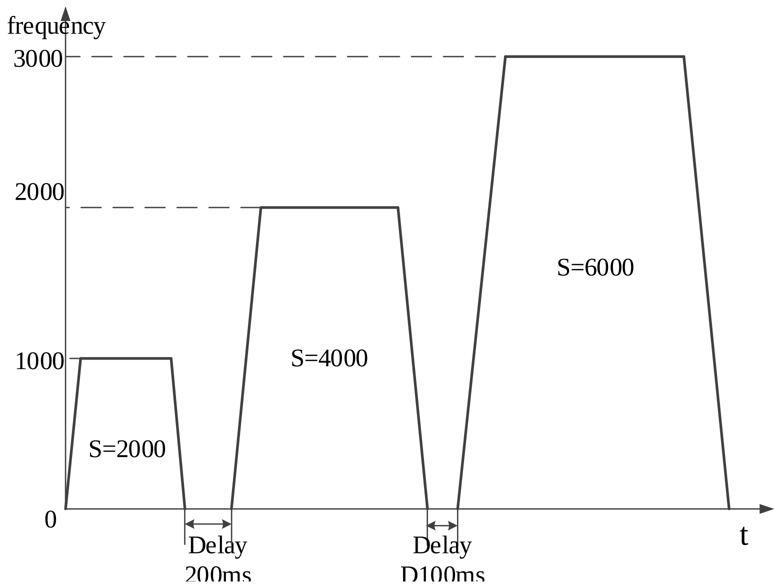
Note: acceleration and deceleration time can be set in the parameter table.

Wait time

Add time delay after current segment finished, and then jump to the certain segment. The delay time can be constant value, or set in register D, HD. The unit is ms.



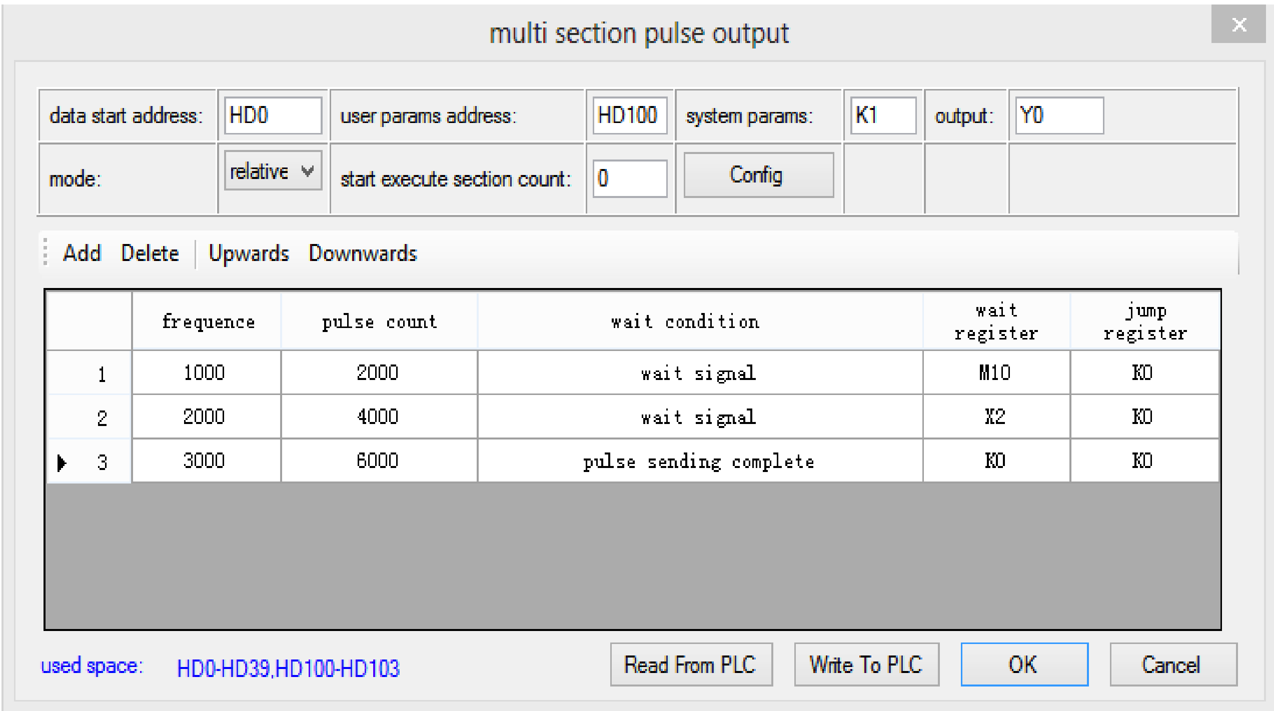
The PLSR is triggered by edge. It will send 2000 pulses at 1000Hz, then wait 200ms. Then it will send 4000 pulses at 2000Hz, then wait the time D100. Then it will send 6000 pulses at 3000Hz.



Note: acceleration and deceleration time can be set in the parameter table.

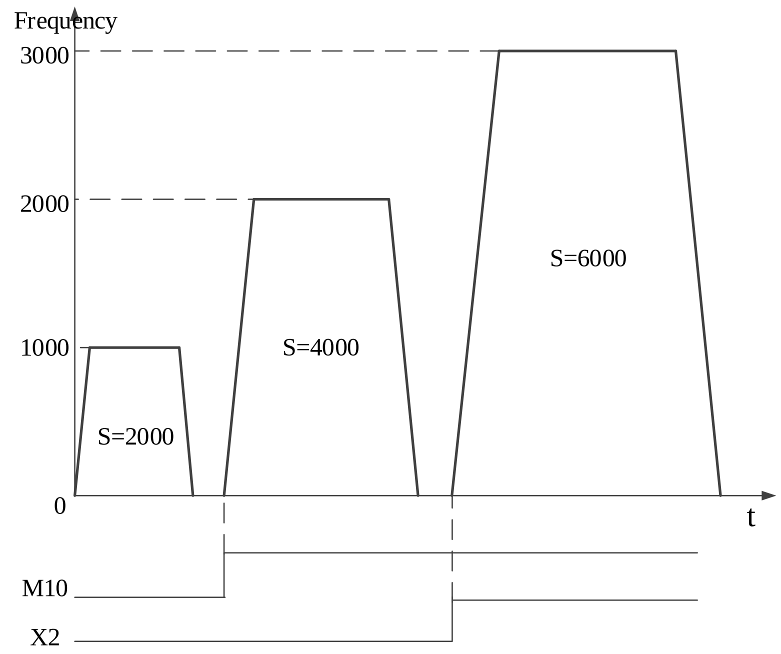
Wait signal

Wait for the wait signal after current segment finished. Jump to the certain segment when the signal arrives. The wait signal can be X, M, HM.



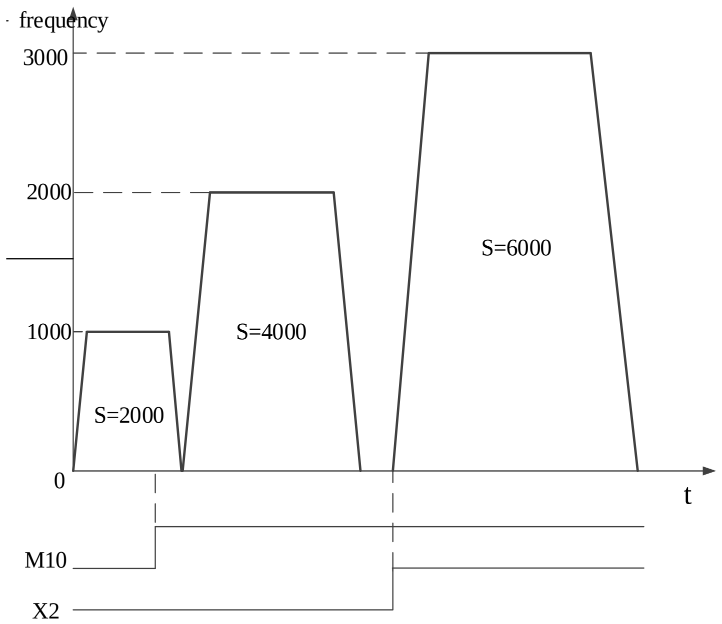
	frequence	pulse count	wait condition	wait register	jump register
1	1000	2000	wait signal	M10	K0
2	2000	4000	wait signal	X2	K0
3	3000	6000	pulse sending complete	K0	K0

The PLSR is triggered by edge. It will send 2000 pulses at 1000Hz. Then it will send 4000 pulses at 2000Hz when M10 from OFF to ON. Then it will send 6000 pulses at 3000Hz when X2 from OFF to ON.



Note:

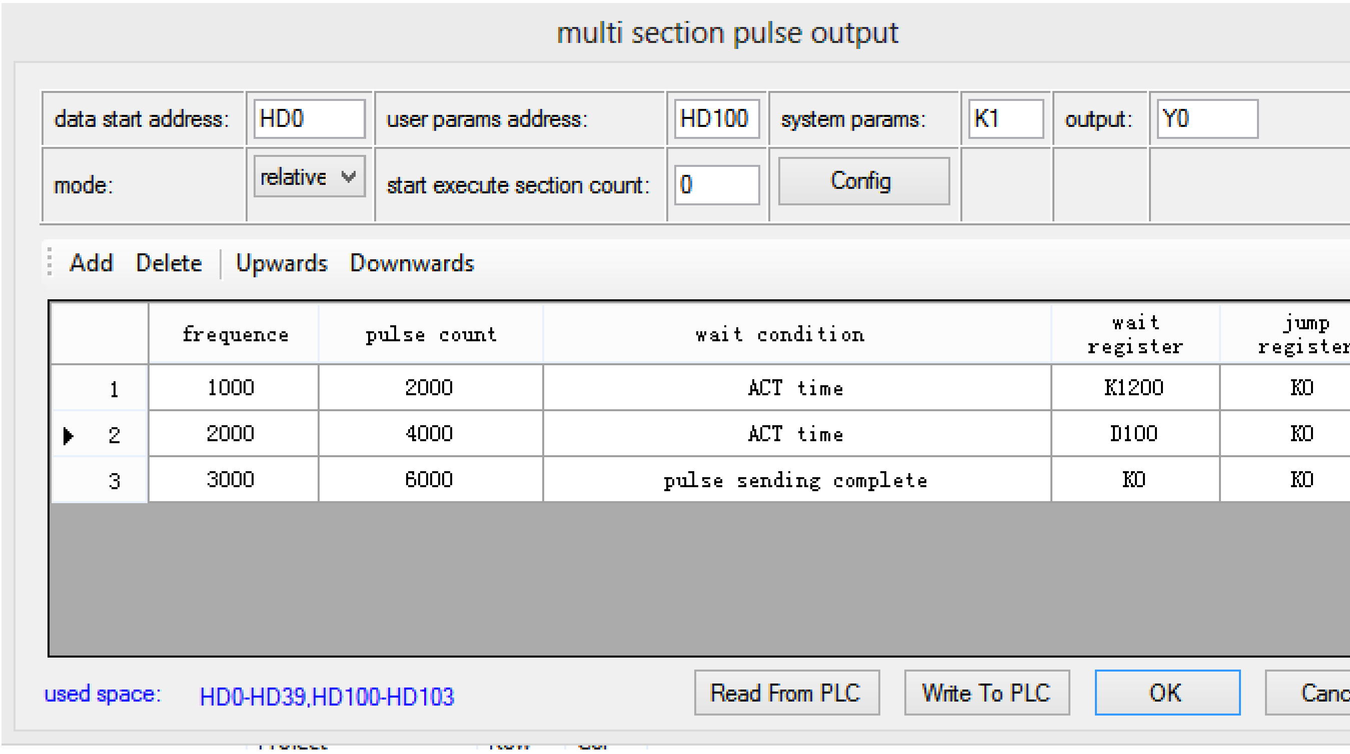
1. acceleration and deceleration time can be set in the parameter table.
2. the wait signal is from OFF to ON, but the present segment pulse is still sending, it will jump to the next segment when the present pulse sending completed.



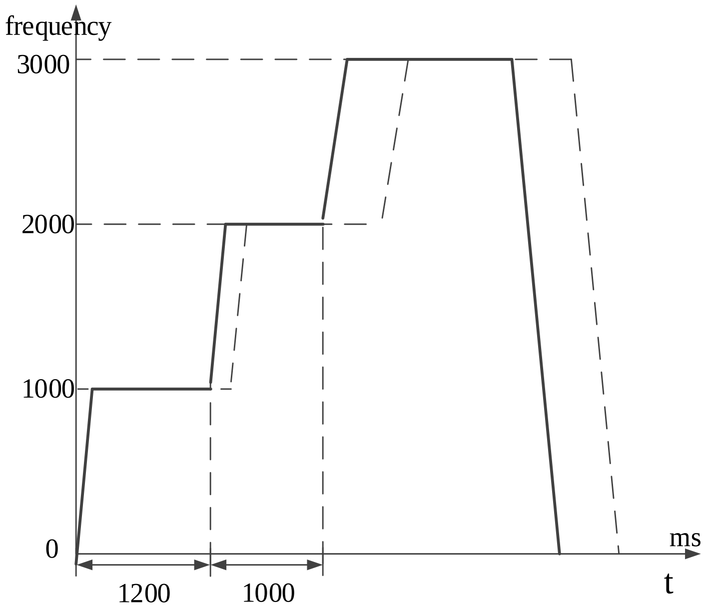
3. if the wait signal is not ON when the present pulses sending completed, it will not jump to the next segment until the wait signal is ON.

ACT time

Pulse sending for the ACT time, it will jump to the certain segment when the ACT time end. ACT time can be constant value or set in register D, HD, FD. The unit is ms.



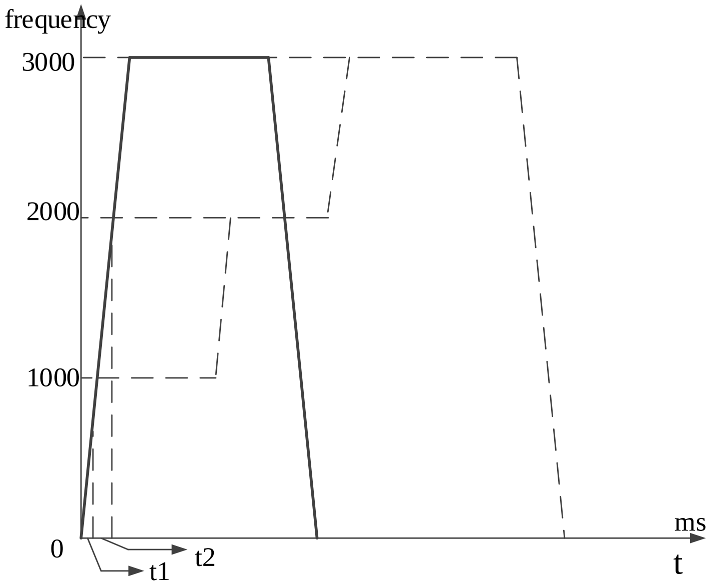
The PLSR is triggered by edge. It will send 2000 pulses at 1000Hz for 1200ms. Then it will send 4000 pulses at 2000Hz for D100 time. Then it will send 6000 pulses at 3000Hz.



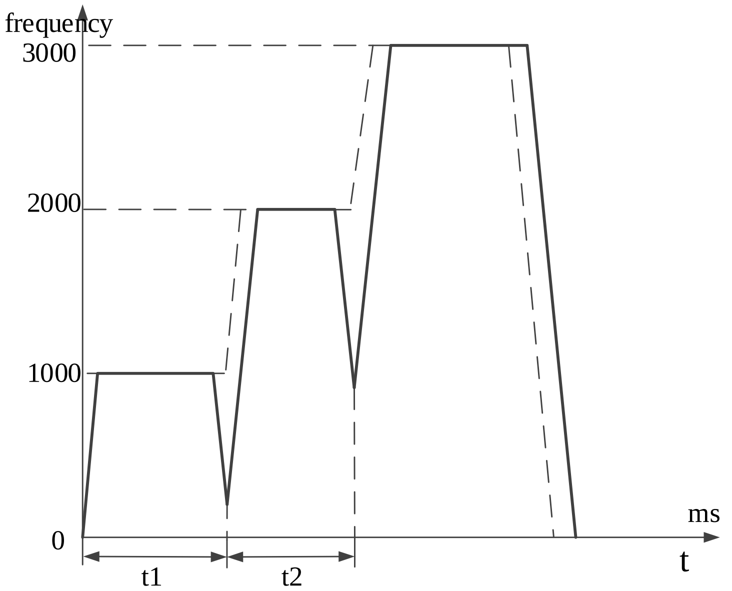
Note:

1. acceleration and deceleration time can be set in the parameter table.

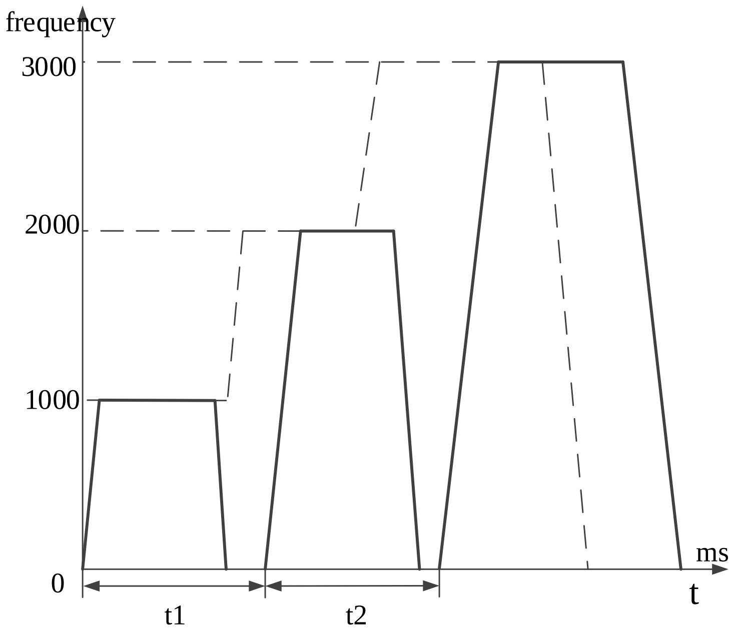
2. if the ACT time is short, the pulse is in accelerating. It will accelerate to next segment at the ACT time end point. The same to the third segment pulses.



3. if the ACT time is long, the pulse is in decelerating. It will accelerate to the next segment at the ACT time end point. The same to the third segment pulse.

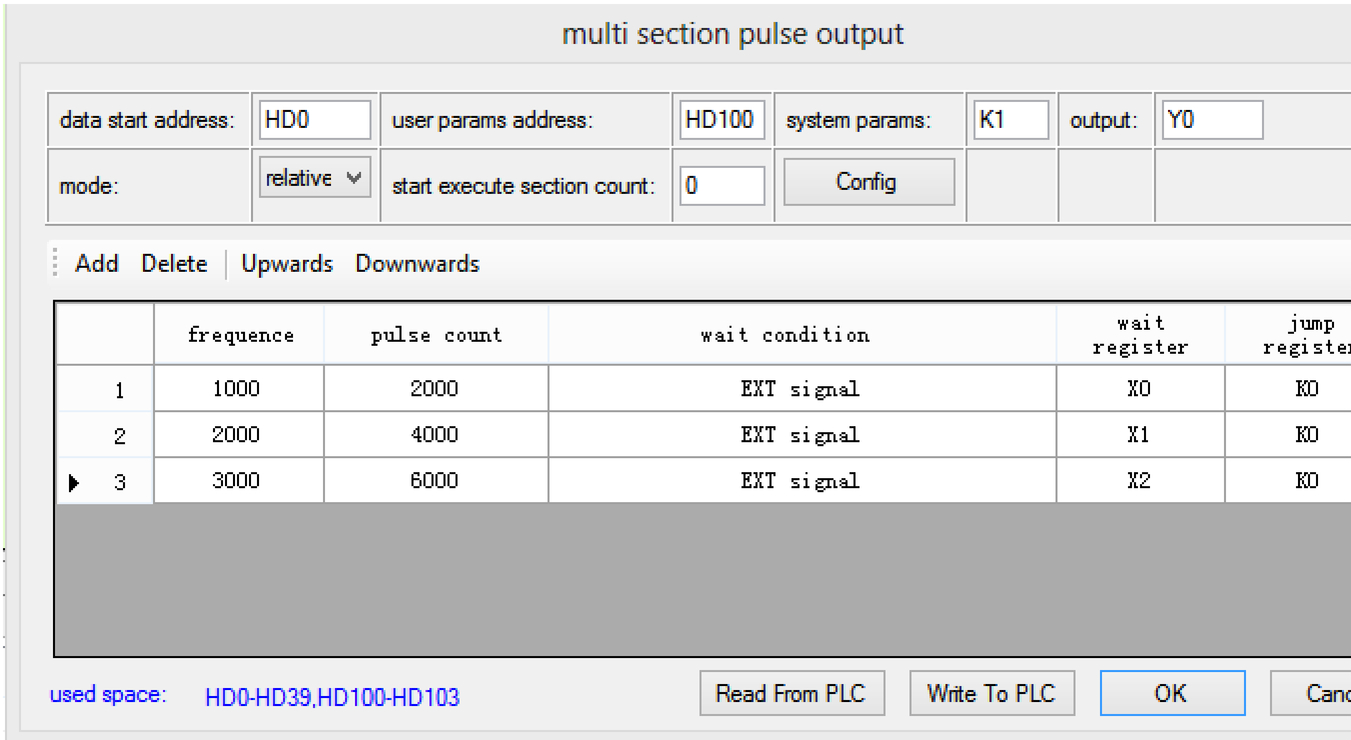


4. if the ACT time is too long, the pulse sending completed but the ACT time has not finished, it will start the next segment when the ACT time finishes.

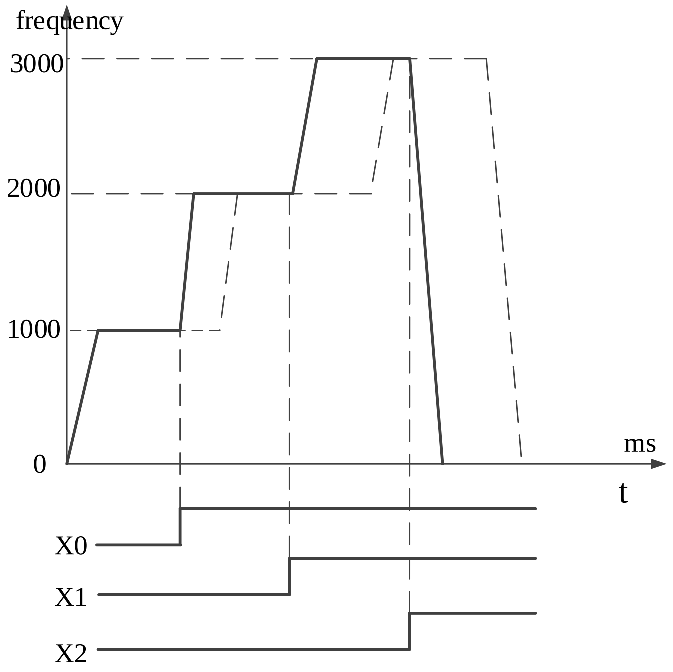


EXT signal

If EXT signal is activated (OFF to ON) when pulse is outputting,it will jump to the certain segment. If EXT signal is not activated when the pulse output finished, it will continue waiting the EXT signal. The EXT signal input is X (if the X is external interruption terminal, the response will faster).

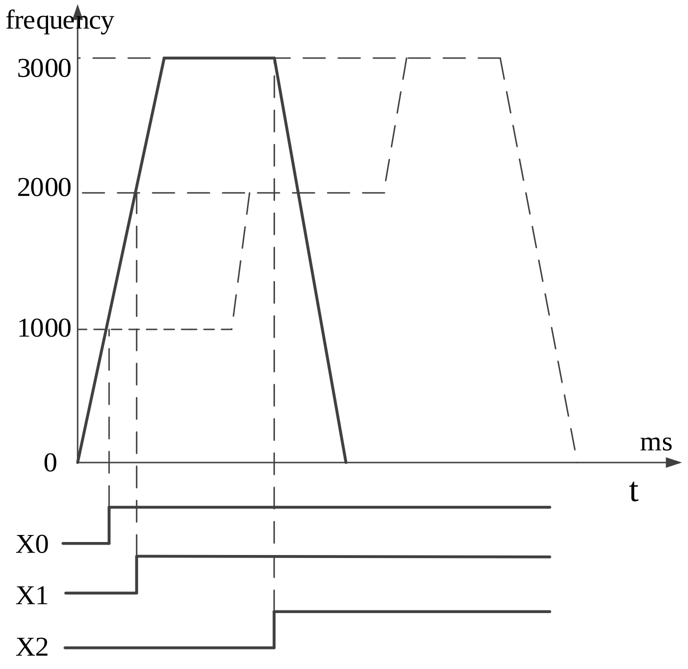


The PLSR is triggered by edge. It will send 2000 pulses at 1000Hz. Then it will send 4000 pulses at 2000Hz when X0 is ON. Then it will send 6000 pulses at 3000Hz when X1 is ON. Then the pulse will deceleration stop when the X2 is ON.

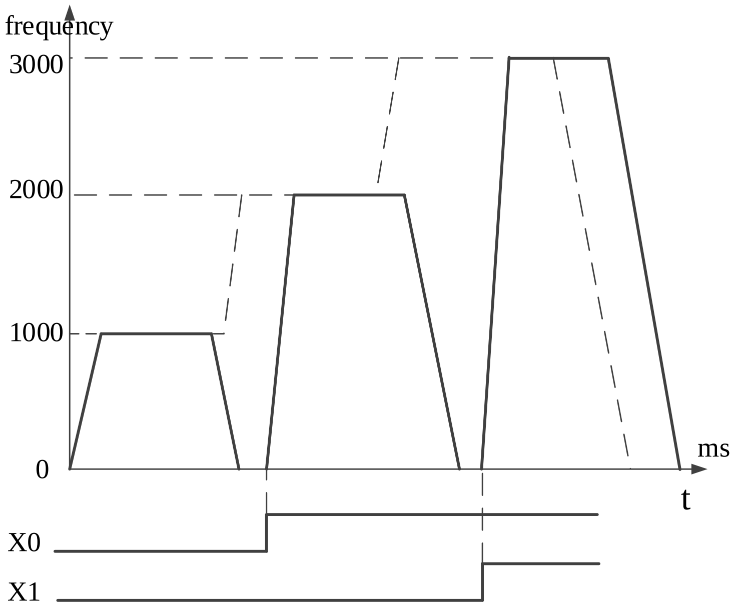


Note:

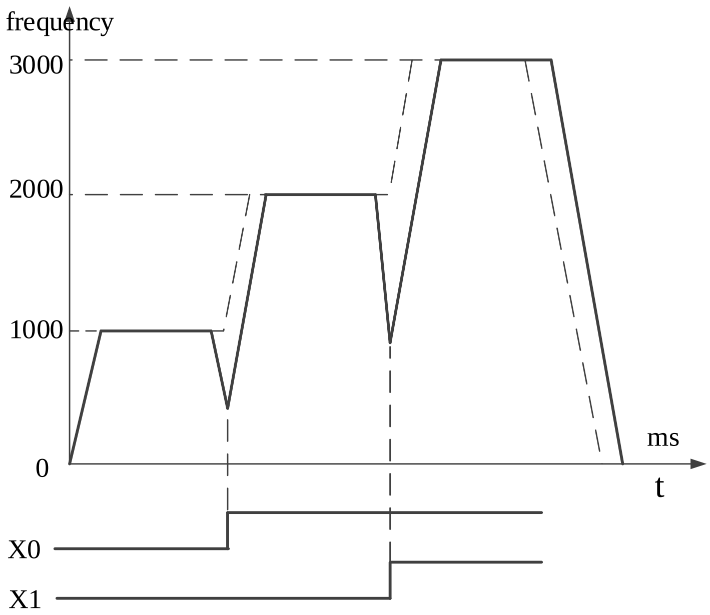
1. acceleration and deceleration time can be set in the parameter table.
2. if the EXT signal comes early, the pulse is in accelerating, it will accelerate to the next segment at the EXT activated position, the same to the third segment.



3. if the EXT signal comes too late, the present pulse already completed sending. It will go to the next segment when the EXT signal activate.

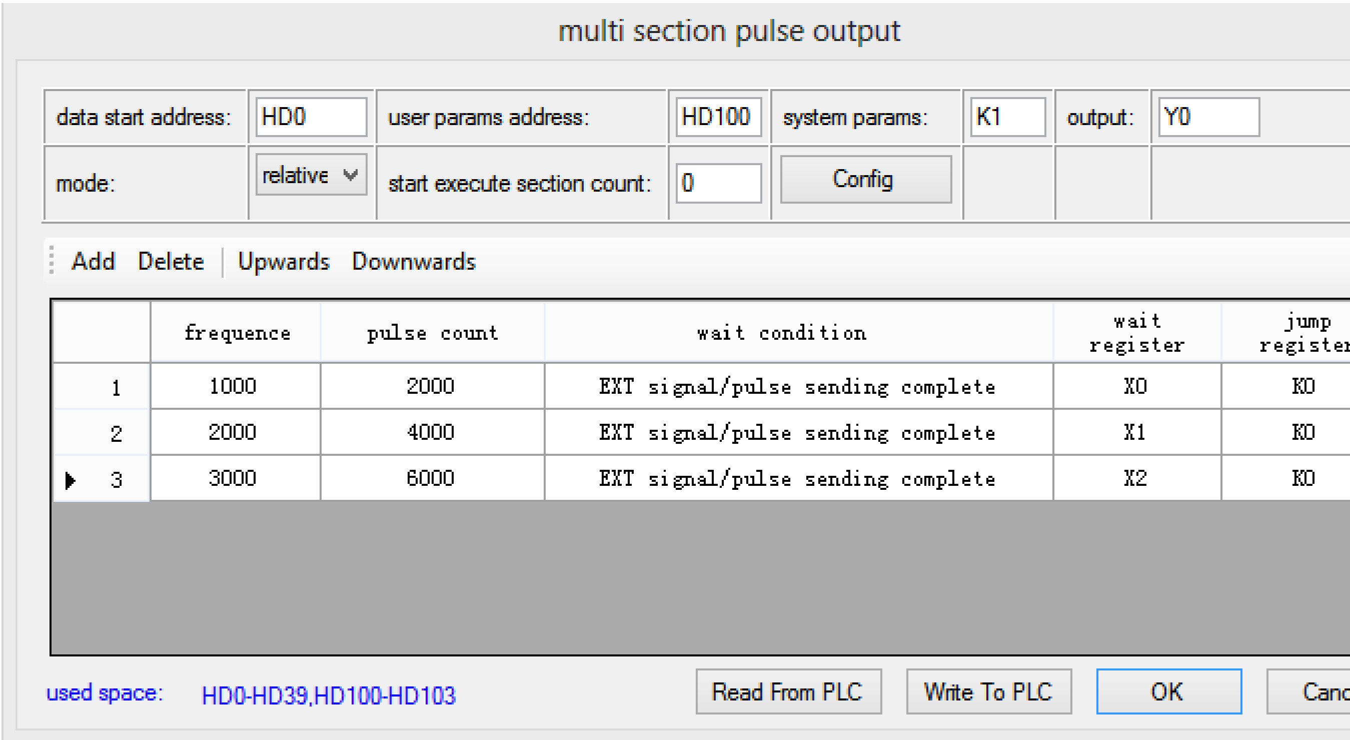


4. if the EXT signal comes too late, the pulse is in decelerating, it will accelerate to the next segment at the EXT signal activated position, the same to the third segment.



EXT signal or pulse sending complete

It will jump to the certain segment when the signal arrives or the pulse output finished. The present pulse is sending, it will go to the next segment when the EXT signal comes. Otherwise, it will go to the next segment when the present pulse sending completed.



EXT signal X0 is effective in the first segment. X1 is effective in the second segment. X2 is effective in the third segment.

Waiting register type

Constant

S0+N*10+5 (dword) the register value is constant. Range: K0~2147483647. Such as K2, K6, K3000.

D

S0+N*10+5 (dword) the register value is D address, such as D0, D200.

HD

S0+N*10+5 (dword) the register value is HD (power-off retentive register) address. Such as HD0, HD200.

FD

S0+N*10+5 (dword) the register value is FD (FLASH register) address. Such as FD0, FD200.

X

S0+N*10+5 (dword) the register value is X address. If the X is external interruption terminal, it is activated by the external interruption signal (the response time is faster). Such as X0, X6.

M

S0+N*10+5 (dword) the register value is M address. Such as M0, M200.

HM

S0+N*10+5 (dword) the register value is HM (power-off retentive coil) address. Such as HM0, HM200.

Jump register type

Constant

S0+N*10+8 (dword) the register value is constant. Range: K0~100. Such as K2, K6.

D

S0+N*10+8 (dword) the register value is D address. Such as D0, D200.

HD

S0+N*10+5 (dword) the register value is HD (power-off retentive register) address. Such as HD0, HD200.

FD

S0+N*10+5 (dword) the register value is FD (FLASH register) address. Such as FD0, FD200.

Note:

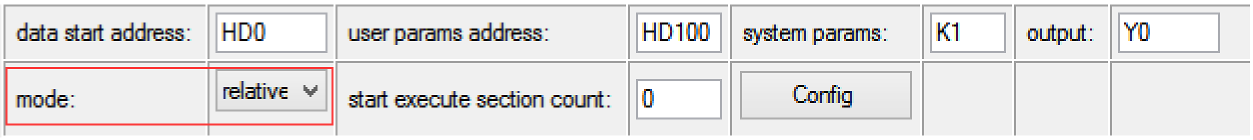
1. this parameter means the present pulse sending completed, it will jump to the next segment. For example: the present value is K6, it will jump to the sixth segment after the present pulse sending end.
2. the constant and register range is K0~100.
3. if the constant or register is 0, it will jump to the next segment in the configuration table.
4. if the constant or register value is the present segment, it will cyclic send the present segment pulse.

Pulse parameters start address

S1+0 (dword)	Pulse mode (0: relative mode; 1: absolute mode) *1
S1+2 (dword)	Pulse start segment (1~100) *2

Note:

*1. The pulse parameter is relative or absolute mode. The default mode is relative.



For example:

There are three segments of pulse. The first segment is 1000Hz, 2000 pulses; the second is 2000Hz, 4000 pulses; the third is 3000Hz, 6000 pulses.

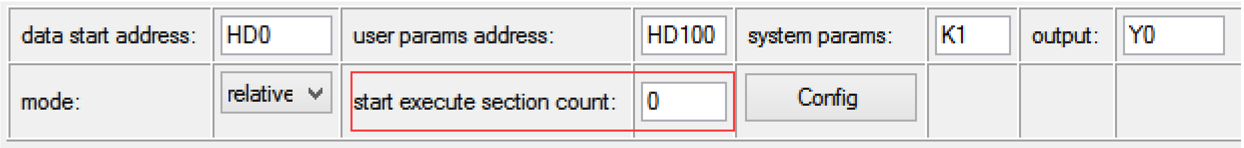
Relative mode:

	frequence	pulse count	wait condition	wait register	jump register
1	1000	2000	pulse sending complete	X0	K0
2	2000	4000	pulse sending complete	X1	K0
3	3000	6000	pulse sending complete	X2	K0

Absolute mode:

	frequence	pulse count	wait condition	wait register	jump register
1	1000	2000	pulse sending complete	X0	K0
2	2000	6000	pulse sending complete	X1	K0
3	3000	12000	pulse sending complete	X2	K0

*2 To set the start execution segment. It will start from the first segment when it is set to 0 or 1.



For example:
There are three segments of pulse. The first segment is 1000Hz, 2000 pulses; the second is 2000Hz, 4000 pulses; the third is 3000Hz, 6000 pulses. The start execute section count is 2. So the pulse will start from segment 2.

System parameters

User can set which group of parameters to use for the pulse. Each channel of pulse has 4 groups of parameters. each group of parameters include pulse speed, acceleration/deceleration time, max speed, start speed, end speed. PLSR can choose the suitable parameters to send the pulse.
Each channel pulse output has system parameters and 4 groups of parameters, user needs to set which group of parameter to use in pulse instruction S2- (constant, D, HD, FD).
Note: the following table is system parameters of frist channel of pulse (Y0). Other pulse channel parameters please refer to appendix 3.

SFD900	Pulse parameters	Bit 0: pulse output logic 0: positive logic(default setting) 1: negative logic, Bit 1: pulse direction logic 0: positive logic(default setting) 1: negative logic Bit 8: pulse unit 0: pulse quantity(default setting) 1: equivalent value	Public parameter	PULSE_1
SFD901	Reserved			
SFD902	Pulse quantity/ 1 rotation low 16 bits			
SFD903	Pulse quantity/ 1 rotation high 16 bits			
SFD904	Movement amount/1 rotation low 16 bits			
SFD905	Movement amount/1 rotation high 16 bits			
SFD906	Pulse direction terminal	The number of Y, 0xFF is no terminal		
SFD907	Direction delay time	Default value is 20, unit is ms		
SFD908	Gear clearance positive compensation			
SFD909	Gear clearance negative compensation			
SFD910	Electrical origin low 16 bits			

SFD911	Electrical origin high 16 bits			
SFD912	Machine back to origin parameters	Bit0: promixity switch state 0: normal open 1: normal close		
SFD913	Near signal terminal	Bit0~bit7: the X terminal number, 0xFF is no terminal		
SFD914	Z phase terminal	Bit0~bit7: the X terminal number, 0xFF is no terminal		
SFD915	Limit terminal	Bit7~bit0: limit 1 X terminal number, 0xFF is no terminal Bit15~bit8: limit 2 X terminal number, 0xFF is no terminal		
SFD916	Origin auxiliary signal terminal	Bit0~bit7: X terminal number, 0xFF is no terminal		
SFD917	CLR signal output terminal	Bit0~bit7: Y terminal number, 0xFF is no terminal		
SFD918	Back speed VH low 16 bits			
SFD919	Back speed VH high 16 bits			
SFD920	Back speed VL low 16 bits			
SFD921	Back speed VL high 16 bits			
SFD922	Creep speed low 16 bits			
SFD923	Creep speed high 16 bits			
SFD924	Mechanical origin low 16 bits			
SFD925	Mechanical origin high 16 bits			
SFD926	Z phase quantity			
SFD927	CLR signal delay time	Default value 20, unit: ms		
SFD936	G instruction which group of parameter choice	G instruction parameter group choice		
…				
SFD950	Pulse default speed low 16 bits	Send pulse with default speed when speed is 0	Group 1 parameters	
SFD951	Pulse default speed high 16 bits			
SFD952	Pulse default speed acceleration time			

SFD953	Pulse default speed deceleration time			
SFD954	Tween acceleration/deceleration time			
SFD955	Reserved			
SFD956	Max speed limit low 16 bits			
SFD957	Max speed limit high 16 bits			
SFD958	Start speed low 16 bits			
SFD959	Start speed high 16 bits			
SFD960	End speed low 16 bits			
SFD961	End speed high 16 bits			
SFD962	Curve acceleration time (ms)			
…				
SFD970	Pulse default speed low 16 bits	Send pulse with default speed when speed is 0	Group 2 parameters	
SFD971	Pulse default speed high 16 bits			
SFD972	Pulse default speed acceleration time			
SFD973	Pulse default speed deceleration time			
SFD974	Tween acceleration/deceleration time			
SFD975	Reserved			
SFD976	Max speed limit low 16 bits			
SFD977	Max speed limit high 16 bits			
SFD978	Start speed low 16 bits			
SFD979	Start speed high 16 bits			
SFD980	End speed low 16 bits			
SFD981	End speed high 16 bits			
SFD982	Curve acceleration time (ms)			
…				
SFD990	Pulse default speed low 16 bits	Send pulse with default speed when speed is 0	Group 3 parameters	
SFD991	Pulse default speed high 16 bits			
SFD992	Pulse default speed acceleration time			
SFD993	Pulse default speed deceleration time			
SFD994	Tween acceleration/deceleration time			
SFD995	Reserved			

SFD996	Max speed limit low 16 bits			
SFD997	Max speed limit high 16 bits			
SFD998	Start speed low 16 bits			
SFD999	Start speed high 16 bits			
SFD1000	End speed low 16 bits			
SFD1001	End speed high 16 bits			
SFD1002	Curve acceleration time (ms)			
…				
SFD1010	Pulse default speed low 16 bits	Send pulse with default speed when speed is 0	Group 4 parameters	
SFD1011	Pulse default speed high 16 bits			
SFD1012	Pulse default speed acceleration time			
SFD1013	Pulse default speed deceleration time			
SFD1014	Tween acceleration/deceleration time			
SFD1015	Reserved			
SFD1016	Max speed limit low 16 bits			
SFD1017	Max speed limit high 16 bits			
SFD1018	Start speed low 16 bits			
SFD1019	Start speed high 16 bits			
SFD1020	End speed low 16 bits			
SFD1021	End speed high 16 bits			
SFD1022	Curve acceleration time (ms)			
…				

Public parameters

- Pulse parameter setting----pulse output logic

Pulse output logic contains positive logic and negative logic.

Positive logic: pulse numbers set to positive value, send pulse in positive direction (for example Y0 axis accumulated pulse numbers register HSD0 keeps increasing), pulse direction terminal is ON. Pulse numbers set to negative value, send pulse in reverse direction (for example Y0 axis accumulated pulse numbers register HSD0 keeps decreasing), pulse direction terminal is OFF.

Negative logic: pulse numbers set to positive value, send pulse in reverse direction (for example Y0 axis accumulated pulse numbers register HSD0 keeps decreasing), pulse direction terminal is OFF. Pulse numbers set to negative value, send pulse in positive direction (for example Y0 axis accumulated pulse numbers register HSD0 keeps increasing), pulse direction terminal is ON.

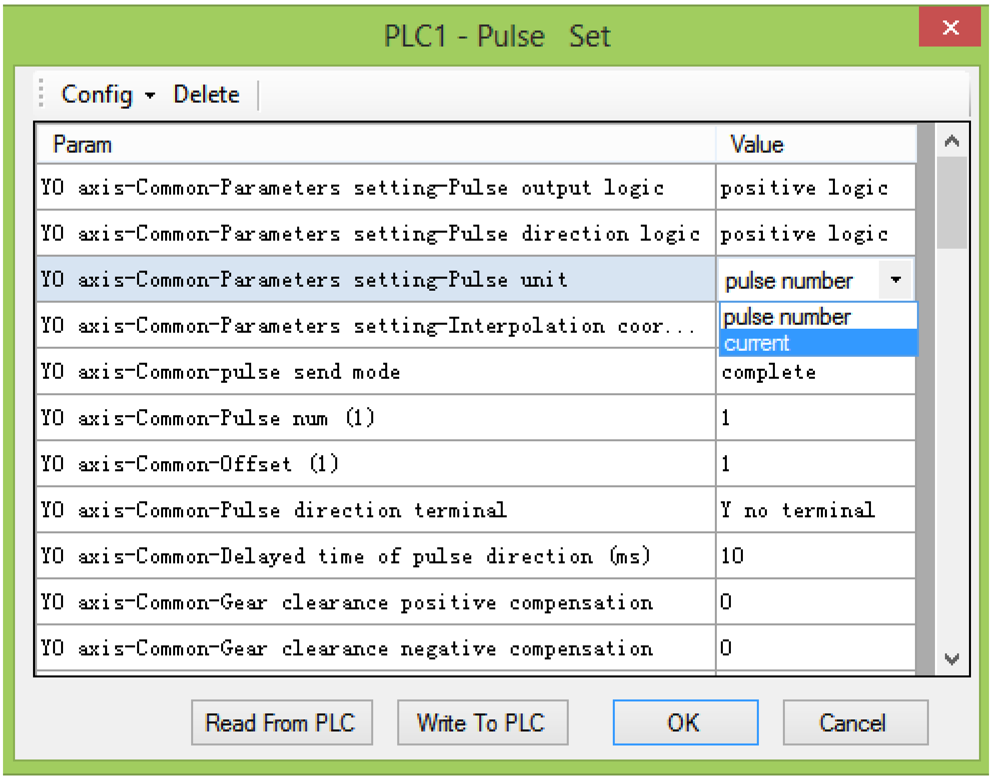
- Pulse parameters setting ---- pulse direction logic

Pulse direction logic contains positive logic and negative logic.
Positive logic: pulse numbers set to positive value, send pulse in positive direction (for example Y0 axis accumulated pulse numbers register HSD0 keeps increasing), pulse direction terminal is ON. Pulse numbers set to negative value, send pulse in reverse direction (for example Y0 axis accumulated pulse numbers register HSD0 keeps decreasing), pulse direction terminal is OFF.
Negative logic: pulse numbers set to positive value, send pulse in positive direction (for example Y0 axis accumulated pulse numbers register HSD0 keeps increasing), pulse direction terminal is OFF. Pulse numbers set to negative value, send pulse in reverse direction (for example Y0 axis accumulated pulse numbers register HSD0 keeps decreasing), pulse direction terminal is ON.

Note: the pulse direction will be ON when the pulse is sending, but the terminal will not be OFF after pulse sending end. When the pulse sends next time, it will auto-change the direction terminal state. If the pulse has no direction next time, please reset the direction terminal by manual.

- **Pulse parameters setting ---- pulse unit**

The pulse unit contains pulse numbers and equivalent.



Pulse number: it calculates as the pulse number in pulse configure table.

	frequence	pulse count	wait condition	wait register	jump register
1	1000	2000	pulse sending complete	K0	K0
▶ 2	2000	4000	pulse sending complete	K0	K0
3	3000	6000	pulse sending complete	K0	K0

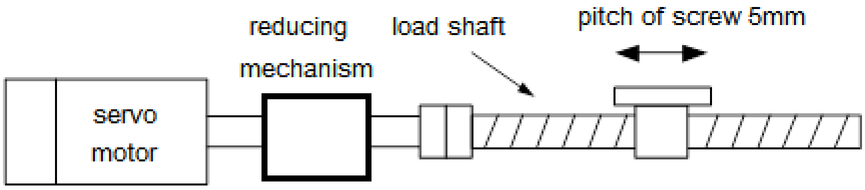
There are 3 segments in the pulse configure table. for example, the second segment will send 4000 pulses with the frequency of 2000Hz.

Equivalent: it calculates the pulse frequency and equivalent as the length unit. Please know the public parameter pulse number (one rotation) and movement (one rotation) before knowing the equivalent.

Public parameter ---- pulse number (one rotation)

The pulse numbers of transmission gear running one rotation. If there is reducing mechanism, the motor running one rotation doesn't mean the transmission gear running one rotation.

For example, the servo motor drives the lead screw through reducing mechanism. The servo drive is DS2-20P7-AS, the motor is MS-80ST-M02430□□-20P7 (encoder 2500 ppr), the servo drive electronic gear ratio is 1:1, the reducing ratio of reducing mechanism is 1:5, the pitch of screw is 5mm.



The pulse numbers of lead screw running one rotation:

$$50000 = 2500 * 4 * \frac{5}{1}$$

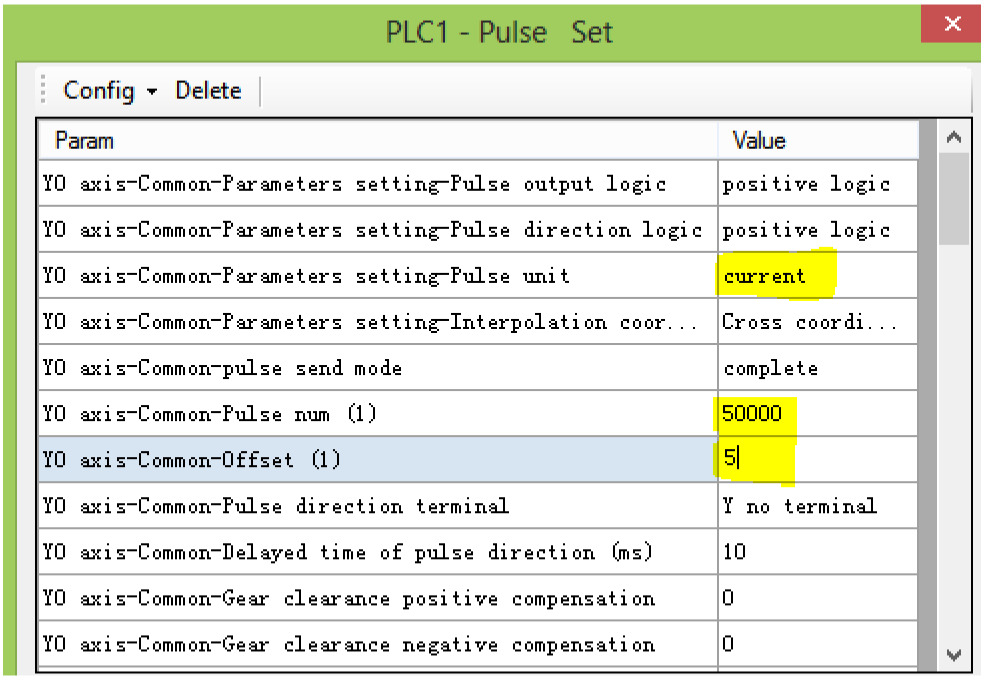
Public parameter ---- movement (one rotation)

The movement amounts of transmission mechanism running one rotation drives the object movement. For example, the lead screw means the screw pitch is 5mm, if it uses synchronous belt, the movement amounts is the circumference of synchronous belt transmission mechanism.

The following we will explain how to set the equivalent. We use the above mechanical structure to send 3 segments of pulse. The pulse configuration table is:

	frequence	pulse count	wait condition	wait register	jump register
▶ 1	10	20	pulse sending complete	K0	K0
2	15	30	pulse sending complete	K0	K0
3	20	40	pulse sending complete	K0	K0

There are 3 segments in the table (equivalent). For example, the second segment is moving 30mm with the speed 15mm/s. The public parameters are as the following.



The equivalent transforms to the corresponding frequency and pulse numbers:

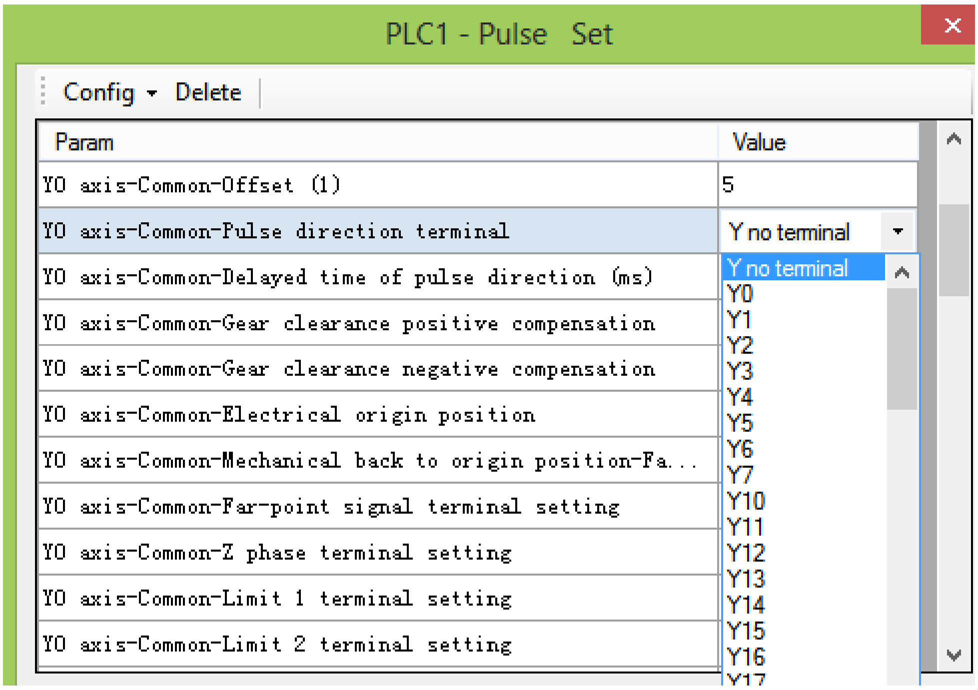
No.	Type	Frequency/speed	Pulse number/length
1	Equivalent	10mm/s	20mm
	Pulse number	100000pulse/s	200000 pulse
2	Equivalent	15mm/s	30mm
	Pulse number	150000pulse/s	300000 pulse
3	Equivalent	20mm/s	40mm
	Pulse number	200000pulse/s	400000 pulse

Note:

1. when it sets to pulse number, Y0 pulse accumulated register HSD0(double words) shows the pulse number, when it sets to equivalent, Y0 pulse accumulated register HSD0(double words) shows the pulse number, accumulated register HSD2 (double words) shows the accumulated equivalent lengths.
2. when it sets to equivalent, all other related parameters will run as equivalent; the equivalent length unit depends on the public parameter movement (one rotation). For example, movement (one rotation) is 6, the unit is mm, so other related length unit and speed unit are mm or mm/s.
3. when the setting is equivalent, please note that the frequency cannot over 200KHz.

- **Public parameter ---- pulse direction terminal**

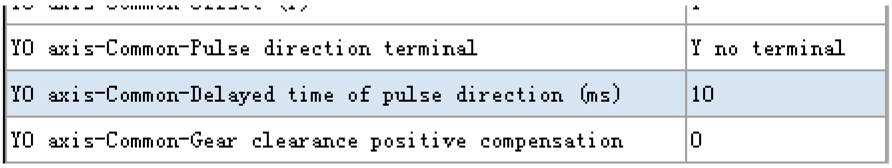
PLSR instruction direction terminal must be configured in the config table:



XD3 series PLC with transistor output terminal have two channels of pulse output (Y0 and Y1). The direction terminal can be any one except Y0 and Y1. When PLC sends pulse from Y0, Y1 has no pulse output, then Y1 can output pulse direction. When PLC sends pulse from Y1, Y0 has no pulse output, then Y0 can output pulse direction.
Note: please do not use the terminals exceed the output terminals on the PLC.

- **Pulse public parameter ---- pulse direction delay time (ms)**

It will set ON the direction terminal and delay the setting time, then the pulse will output. The delay time is pulse direction delay time (ms).

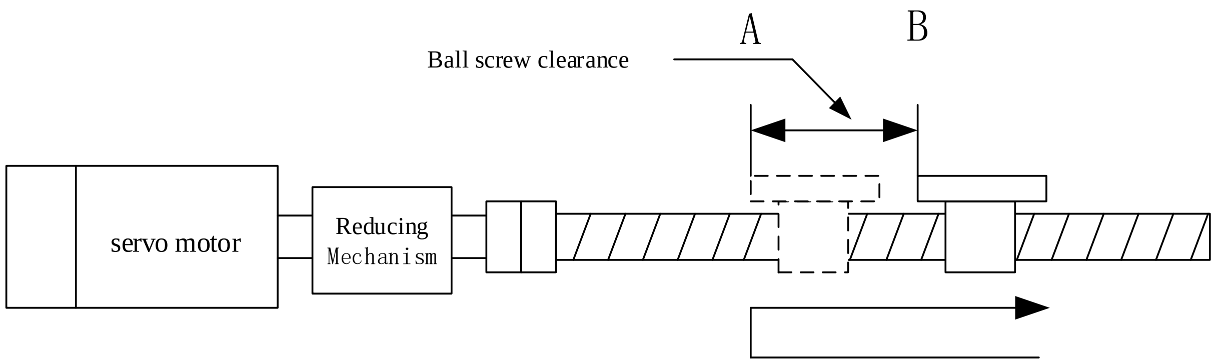


Y0 axis-Common-Pulse direction terminal	Y no terminal
Y0 axis-Common-Delayed time of pulse direction (ms)	10
Y0 axis-Common-Gear clearance positive compensation	0

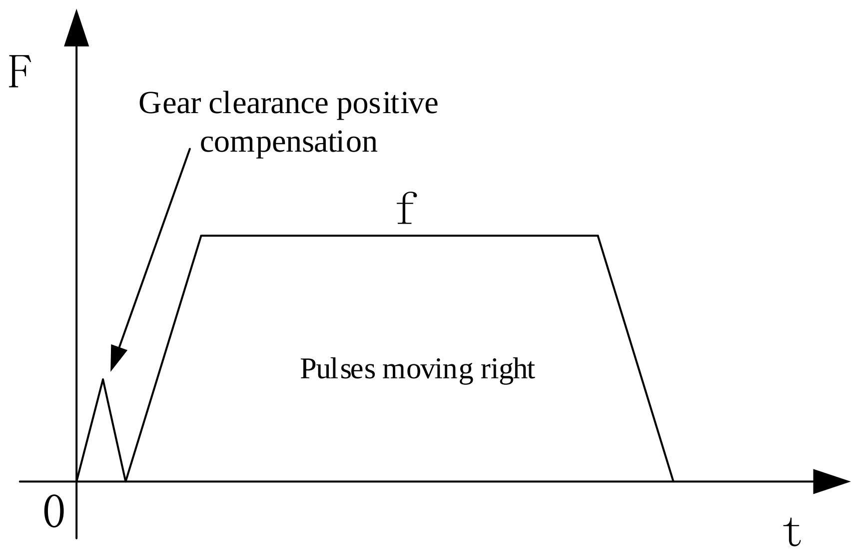
As the Y0 and Y1 terminal is high-speed optical coupler, and other terminal is normal optical coupler or relay output. The direction outputting will behind the pulse outputting. The direction terminal will be activated ahead and delay some time, then the pulse will output. This will avoid pulse output error caused by direction terminal lagging.
The default delay time is 10ms. Please adjust the time according to the direction terminal type. The delay time from short to long is Y0 and Y1, transistor output, relay output.

- **Pulse public parameter ---- gear clearance positive compensation**

There is contacting clearance between working table and ball screw. The working table switching from reverse movement to forward movement will cause the forward actual distance smaller than the setting distance. To clear up the distance error, it has the function of gear clearance positive compensation.



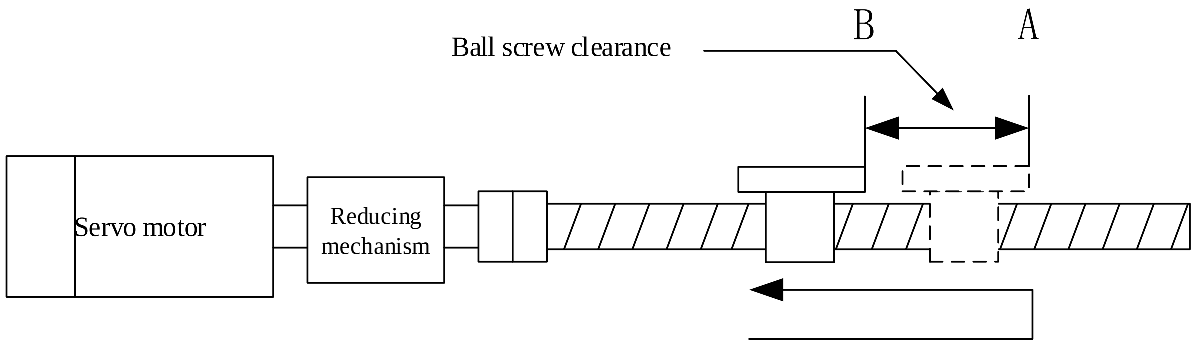
The working table moves from right to left. The working table will stop when its left side reaching the postion A. Then it moves from A to right. As the ball screw clearance, the working table will not move right with the beginning pulses. This will decrease the actual moving right distance. Without the ball screw clearance, the beginning pulses will make the working table moving from position A to B. To solve the problem, it needs to send some more pulses before sending the required moving right pulses.



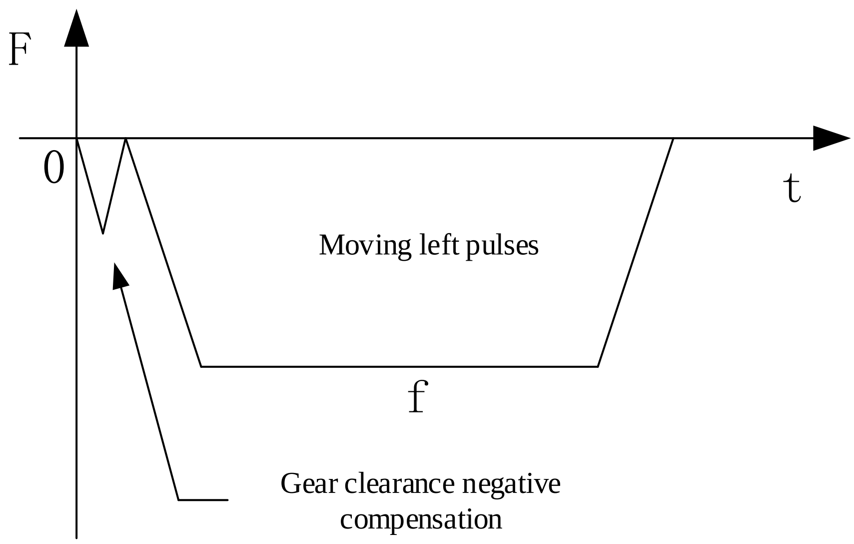
Note:
1. the gear clearance positive compensation will work when the pulse direction of present segment and last segment is opposite.
2. the gear clearance positive compensation and moving right pulses must send in different pulse segment.
3. the pulses of gear clearance positive compensation will not be accumulated in the pulse accumulated register. (Y0 register is HSD0)

- **Pulse public parameter ---- gear clearance negative compensative**

There is contacting clearance between working table and ball screw. The working table switching from forward movement to reverse movement will cause the reverse actual distance smaller than the setting distance. To clear up the distance error, it has the function of gear clearance negative compensation.



The working table moves from left to right. The working table will stop when its right side reaching the postion A. Then it moves from A to left. As the ball screw clearance, the working table will not move left with the beginning pulses. This will decrease the actual moving left distance. Without the ball screw clearance, the beginning pulses will make the working table moving from position A to B. To solve the problem, it needs to send some more pulses before sending the required moving left pulses.



Note:
1. the gear clearance negative compensation will work when the pulse direction of present segment and last segment is opposite.
2. the gear clearance negative compensation and moving left pulses must send in different pulse segment.
3. the pulses of gear clearance negative compensation will not be accumulated in the pulse accumulated register. (Y0 register is HSD0)

- **Pulse public parameter ---- electrical origin position**

This is not used in XD3 series PLC.

- **Pulse public parameter ---- origin switch state setting**

The mechanical origin switch state when returning to mechanical origin. Such as proximity switch, travel switch.

Y0 axis-Common-Electrical origin position	0
Y0 axis-Common-Mechanical back to origin position-Fa...	normally on
Y0 axis-Common-Far-point signal terminal setting	X no terminal
Y0 axis-Common-Z phase terminal setting	X no terminal

Normally OFF: The mechanical origin switch state is always OFF when returning to mechanical origin. When it touches the mechanical origin switch, the state will be ON.
Normally ON: The mechanical origin switch state is always ON when returning to mechanical origin. When it touches the mechanical origin switch, the state will be OFF.
Note: this function only fit for mechanical back to origin instruction ZRN.

- **Pulse public parameter ---- origin signal terminal setting**

The PLC input terminal connected to the mechanical origin switch when backing to the mechanical origin.

Y0 axis-Common-Mechanical back to origin position-Fa...	normally on
Y0 axis-Common-Far-point signal terminal setting	X no terminal
Y0 axis-Common-Z phase terminal setting	X no terminal

Note:
1. the input ID cannot over the real PLC input terminal range.
2. this function is only fit for mechanical back to origin instruction ZRN.

- **Pulse public parameter ---- Z phase terminal setting**

It will reverse move at low speed with the acceleration time slope until reaching the origin creeping speed. At the moment of leaving the origin it starts to count the Z phase input signal. This function is set the terminal connected to the Z phase input signal counting input.

Y0 axis-Common-Far-point signal terminal setting	X no terminal
Y0 axis-Common-Z phase terminal setting	X no terminal
Y0 axis-Common-Limit 1 terminal setting	X no terminal

Note: this function is only fit for mechanical back to origin instruction ZRN.

- **Pulse public parameter ---- limit 1 terminal setting and limit 2 terminal setting**

When backing to the mechanical origin, add protection terminal (such as travel switch) at both sides of the trip to avoid the working table out of the trip range.

Y0 axis-Common-Limit 1 terminal setting	X no terminal
Y0 axis-Common-Limit 2 terminal setting	X no terminal
Y0 axis-Common-Origin auxiliary signal X setting	X no terminal

Note:
1. the input ID cannot over the real PLC input terminal range.
2. this function is only fit for mechanical back to origin instruction ZRN.

- **Pulse public parameter ---- origin auxiliary signal X terminal setting**

The auxiliary input terminal helps to get the high precision when backing to the mechanical origin.

When the working table is far away from the origin, the ZRN instruction will approach the origin with fast returning speed VH. When the working table closes to the origin, it can switch the speed VH to VL by origin auxiliary signal. It can make the working table touching the origin with slow speed VL.

Y0 axis-Common-Limit 2 terminal setting	X no terminal
Y0 axis-Common-Origin auxiliary signal X setting	X no terminal
Y0 axis-Common-Zero clear CLR output setting	Y no terminal
Y0 axis-Common-Return speed VH (Hz)	0

Note:

1. the input ID cannot over the real PLC input terminal range.
2. this function is only fit for mechanical back to origin instruction ZRN.

- **Pulse public parameter ---- zero clear CLR output terminal setting and CLR signal delay time**

zero clear CLR output terminal setting : It outputs a signal when returning the mechanical origin end. This signal can be sent to other control device. For example, it outputs CLR signal to servo drive in order to clear the servo motor Error Counter when returning the mechanical origin end. Finally it copies the mechanical origin position to present position. Now the flyback action finished.

Y0 axis-Common-Origin auxiliary signal X setting	X no terminal
Y0 axis-Common-Zero clear CLR output setting	Y no terminal
Y0 axis-Common-Return speed VH (Hz)	0

CLR signal delay time: the CLR signal delay time after returning mechanical origin end. The unit is ms, the range is 0~32767.

Y0 axis-Common-Z phase num	0
Y0 axis-Common-CLR signal delayed time (ms)	20
Y0 axis-Common-grinding wheel radius(polar)	0

This function is only fit for mechanical back to origin instruction ZRN.

- **Pulse public parameter ---- return speed VH**

When it executes the ZRN instruction, the working table accelerates to return speed VH in order to decrease the time of backing to mechanical origin.

Y0 axis-Common-Zero clear CLR output setting	Y no terminal
Y0 axis-Common-Return speed VH (Hz)	0
Y0 axis-Common-Return speed VL (Hz)	0

This function is only fit for mechanical back to origin instruction ZRN.

- **Pulse public parameter ---- return speed VL**

When it executes the ZRN instruction, the working table touches the origin auxiliary signal with return speed VH, then it decelerates to return speed VL in order to improve the back to mechanical origin precision.

Y0 axis-Common-Return speed VH (Hz)	0
Y0 axis-Common-Return speed VL (Hz)	0
Y0 axis-Common-Creeping speed (Hz)	0

This function is only fit for mechanical back to origin instruction ZRN.

- **Pulse public parameter ---- creeping speed**

When it encounters the origin signal, the starting speed becomes zero, after the delay time it accelerates to creeping speed in opposite direction. It stops the creeping speed once the working table broken away from the origin signal. As the stop position is mechanical origin after the working table broken away from the origin signal, the creeping speed is normally small in order to improve the mechanical origin precision.

Y0 axis-Common-Return speed VL (Hz)	0
Y0 axis-Common-Creeping speed (Hz)	0
Y0 axis-Common-Mechanical zero position	0

This function is only fit for mechanical back to origin instruction ZRN.

- **Pulse public parameter ---- mechanical origin position**

The present position of working table when the returning mechanical origin end. Take Y0 axis as an example, set present position value in HSD0(double words) or HSD2(double words). Generally, the present value of mechanical origin position is 0, user can change it as needs. When the returning mechanical origin end, the corresponding register value will auto-change to set value.

Y0 axis-Common-Creeping speed (Hz)	0
Y0 axis-Common-Mechanical zero position	0
Y0 axis-Common-Z phase num	0

This function is only fit for mechanical back to origin instruction ZRN.

- **Pulse public parameter ---- Z phase numbers**

When it encounters the origin signal, the starting speed becomes zero, after the delay time it accelerates to creeping speed in opposite direction. After the working table broken away from the origin signal, it can count the servo motor Z phase pulses. When the count value reaches the Z phase pulse setting value, it stops the creeping speed immediately and cancel returning mechanical origin.

Y0 axis-Common-Mechanical zero position	0
Y0 axis-Common-Z phase num	0
Y0 axis-Common-CLR signal delayed time (ms)	20

This function is only fit for mechanical back to origin instruction ZRN.

- **Pulse public parameter ---- G instruction parameter group selection**

This parameter is only fit for XDM series PLC.

- ❖ **The first group of parameter (parameter group 2 to 4 please refer to group 1)**
- **Group 1 parameter ---- pulse default speed, pulse default acceleration time, pulse default deceleration time**

The three parameters can define the pulse acceleration and deceleration slope. The pulse default speed unit will be decided by pulse unit (pulse number or equivalent).

Y0 axis-group 1-Pulse default speed (Hz)	0
Y0 axis-group 1-Acceleration time of Pulse default s...	0
Y0 axis-group 1-Deceleration time of pulse default s...	0

Example 1: the pulse unit is pulse number, pulse default speed is 1000, pulse default acceleration time is 100ms, pulse default deceleration time is 200ms. It means when the pulse instruction is in acceleration stage, it costs 100ms when the pulse frequency increases1000Hz. When the pulse instruction is in deceleration stage, it costs 200ms when the pulse frequency decreases 1000Hz. If it accelerates from 0Hz to 5000Hz, the acceleration time is 5000Hz/1000Hz*100ms=500ms. If it decelerates from 5000Hz to 0Hz, the deceleration time is 5000Hz/1000Hz*200ms=1000ms.

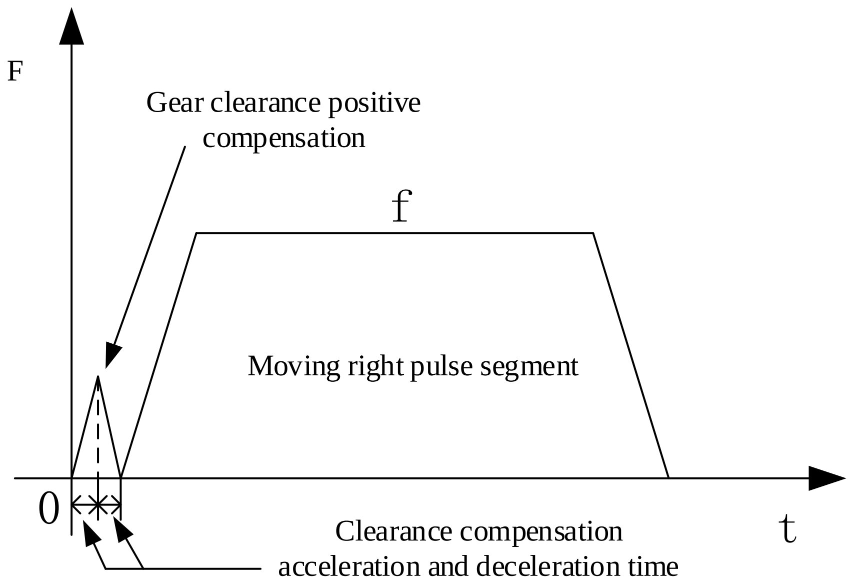
Example 2: the pulse unit is equivalent, the equivalent unit is mm. pulse default speed is 10, pulse default acceleration time is 100ms, pulse default deceleration time is 200ms. It means when the pulse instruction is in acceleration stage, it costs 100ms when the pulse frequency increases10mm/s. When the pulse instruction is in deceleration stage, it costs 200ms when the pulse frequency decreases 10mm/s. If it accelerates from 0Hz to 50mm/s, the acceleration time is 50mm/s/10mm/s*100ms=500ms, If it decelerates from 50mm/s to 0Hz, the deceleration time is 50mm/s/10mm/s*200ms =1000ms.

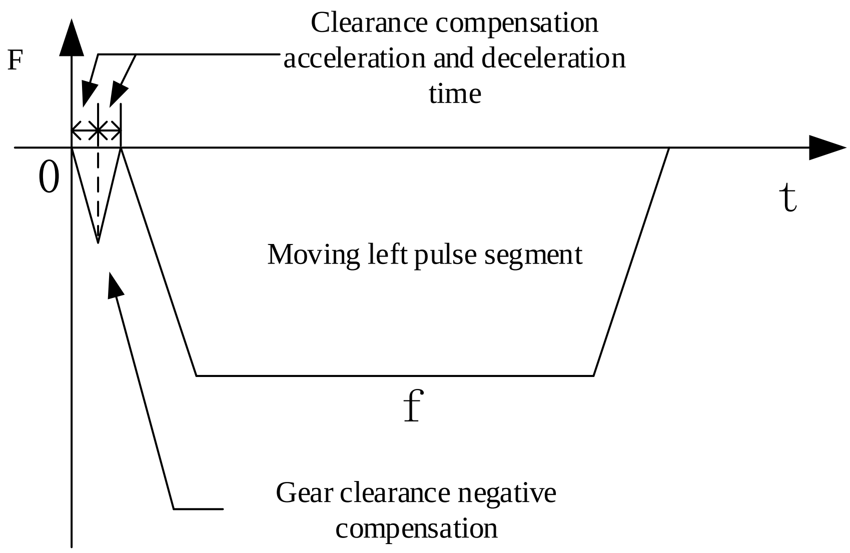
Note: the three parameters define the slope of deceleration and acceleration time. Each segment pulse will run as the slope.

- **First group of parameter ---- clearance compensation acceleration and deceleration time**

The acceleration and deceleration time of **gear clearance positive compensation** and **gear clearance negative compensation**. The acceleration and deceleration time is the same, they are **clearance compensation acceleration and deceleration time** whatever the setting clearance compensation is, the unit is ms.

Y0 axis-group 1-Deceleration time of pulse default s...	0
Y0 axis-group 1-Acceleration and deceleration time (ms)	0
Y0 axis-group 1-pulse acc/dec mode	linear acc/dec





Note: the acceleration and deceleration time is the same.

- **First group of parameter ---- max speed (Hz)**

When the pulse instruction is using the first group of parameters, the max pulse frequency cannot over the **max speed**. If it over the max speed, it will run as the max speed.

Y0 axis-group 1-pulse acc/dec mode	linear acc/dec
Y0 axis-group 1-Max speed (Hz)	0
Y0 axis-group 1-Initial speed (Hz)	0

Note:

1. The max speed unit is pulse number/second, this will not change as the pulse unit (pulse number or equivalent)
2. XD3 series PLC max output frequency is 200KHz. The max speed cannot over this value.

3. When the pulse unit is equivalent, the transformed pulse output frequency maybe too large and over the max speed, please pay attention.
4. make sure to set the max speed when using the pulse output instruction. Otherwise the pulse output instruction cannot work.

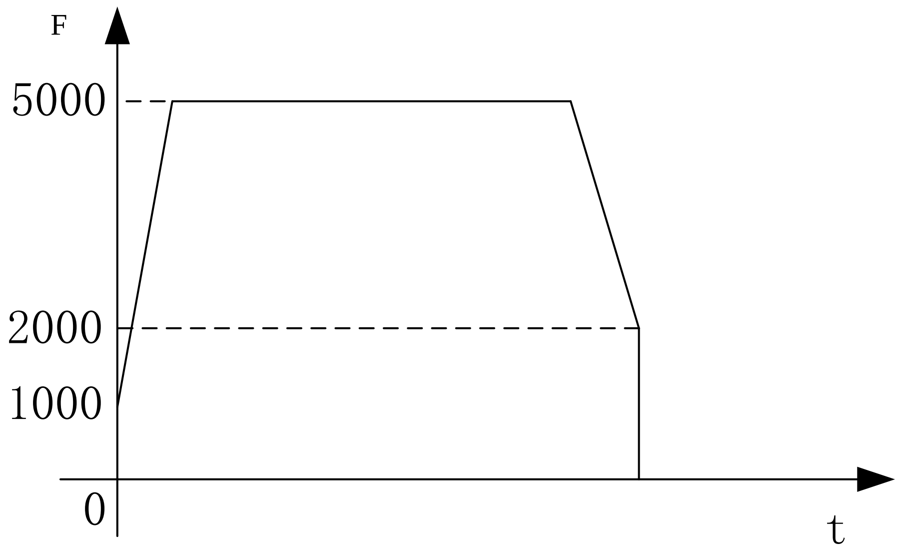
- **The first group of parameter ---- start speed/end speed (Hz)**

The startup frequency and end frequency of pulse instruction. Generally, the start speed and end speed is 0Hz. For some special occasions, the pulse instruction needs to accelerate from non-zero speed, and the pulse end speed is not 0.

Y0 axis-group 1-Initial speed (Hz)	0
Y0 axis-group 1-stop speed (Hz)	0
Y0 axis-group 2-Pulse default speed (Hz)	0

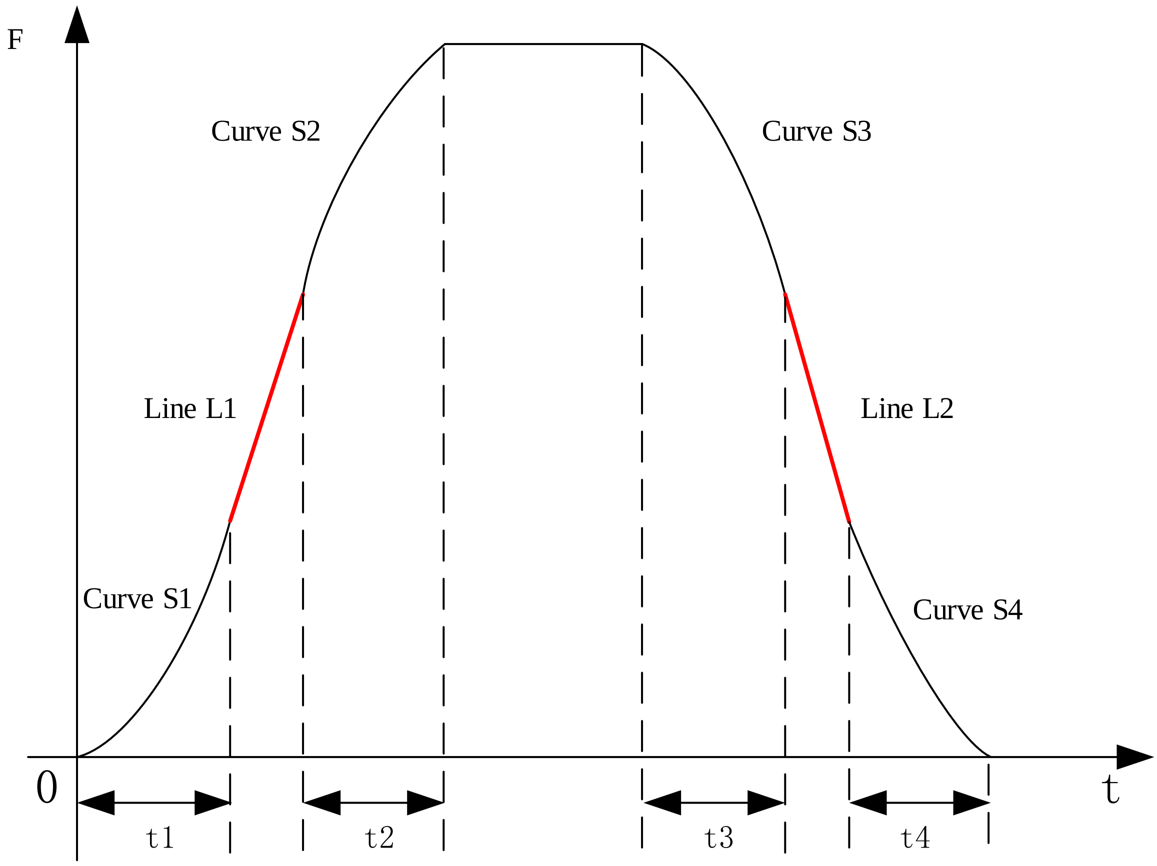
For example, the pulse number is 30000, start speed is 1000Hz, it costs 100ms to reach 5000Hz, it costs 50ms to decelerate to 2000Hz. The configuration table is as below:

Y0 axis-group 1-Initial speed (Hz)	1000
Y0 axis-group 1-stop speed (Hz)	2000



- **The first group of parameter ---- curve acceleration time (ms)**

The startup segment and stop segment can accelerate or decelerate as S-curve. The pulse wave is fit for stepper drive and servo motor. It can improve their running performance. The curve acceleration time unit is ms. This time is the following segment including S1, S3, S2, S4, L1, L2.



curve acceleration time=t1+t2, t1=t2; curve deceleration time=t3+t4, t3=t4. If the acceleration and deceleration time is t, t1+ t2=t= t3+t4, acceleration curve is formed by S1 and S2, deceleration curve is formed by S3 and S4.

Note:

1. when using the curve acceleration time, please keep the same of pulse default speed and pulse max speed; keep the same of pulse default acceleration time, pulse default deceleration time and curve acceleration time.

2. curve acceleration time is only fit for single segment pulse, but not fit for multi-segment S curve switching.

❖ **Parameter group 2 to 4 please refer to the first group**

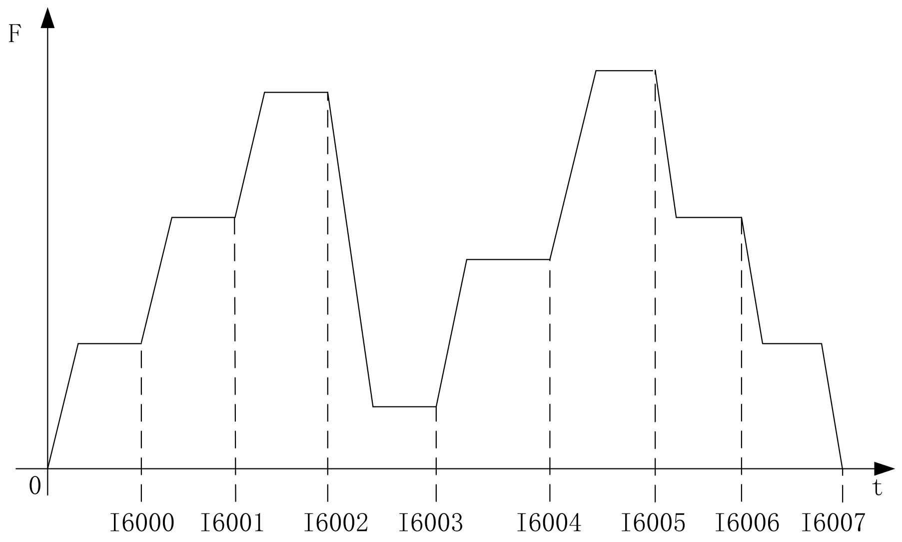
- **Pulse interruption flag**

PLSR has 100 segments interruptions. It produces an interruption after one pulse segment finished.

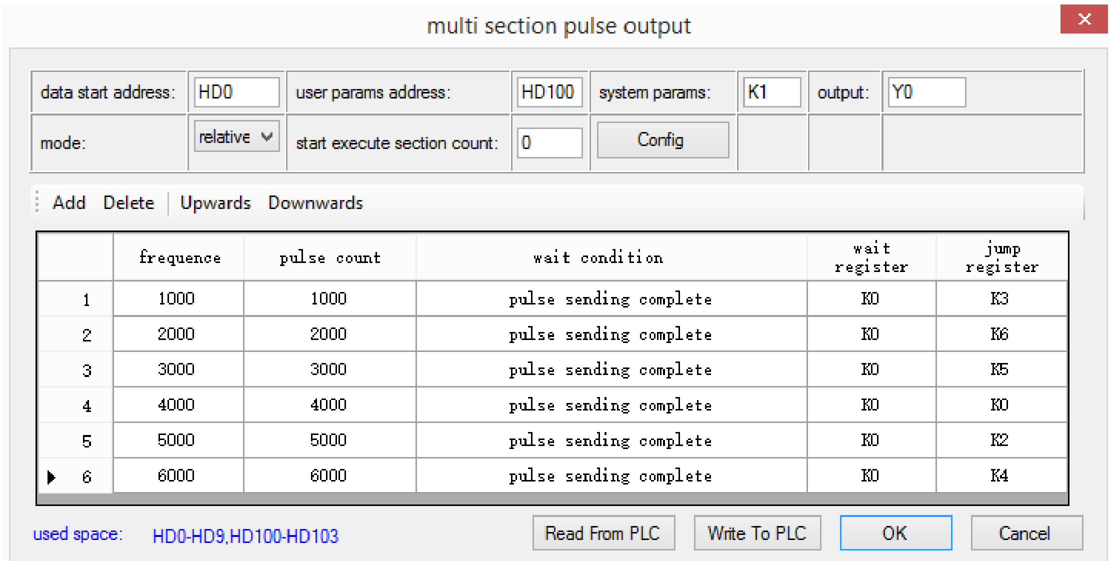
Note: each segment has a unique interruption flag whatever the jump setting in pulse configuration table. Once the pulse segment is executed, the interruption flag will be produced.

I60** (I6000~I6099)	PLS+0 (pulse)	Y0 pulse 100 segments interruption sublist address
I61**(I6000~I6099)	PLS+1 (pulse)	Y1 pulse 100 segments interruption sublist address
I62**(I6000~I6099)	PLS+2 (pulse)	Y2 pulse 100 segments interruption sublist address
I63**(I6000~I6099)	PLS+3 (pulse)	Y3 pulse 100 segments interruption sublist address
I64**(I6000~I6099)	PLS+4 (pulse)	Y4 pulse 100 segments interruption sublist address
I65**(I6000~I6099)	PLS+5 (pulse)	Y5 pulse 100 segments interruption sublist address
I66**(I6000~I6099)	PLS+6 (pulse)	Y6 pulse 100 segments interruption sublist address
I67**(I6000~I6099)	PLS+7 (pulse)	Y7 pulse 100 segments interruption sublist address
I68**(I6000~I6099)	PLS+8 (pulse)	Y8 pulse 100 segments interruption sublist address
I69**(I6000~I6099)	PLS+9 (pulse)	Y9 pulse 100 segments interruption sublist address

Example 1: configure 8 pulse segments and run from segment 1. The pulse output terminal is Y0. The interruption is shown as below:

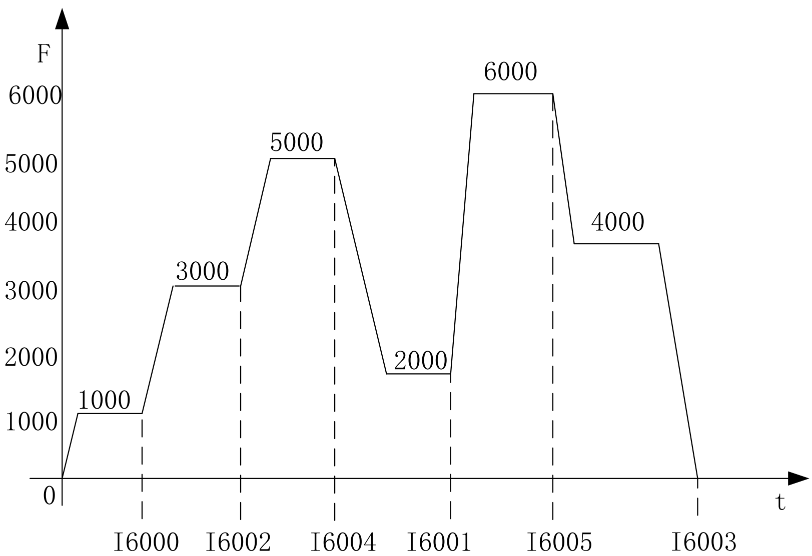


Example 2: configure 6 pulse segments, the pulse output terminal is Y0, but it is not continuous sending.

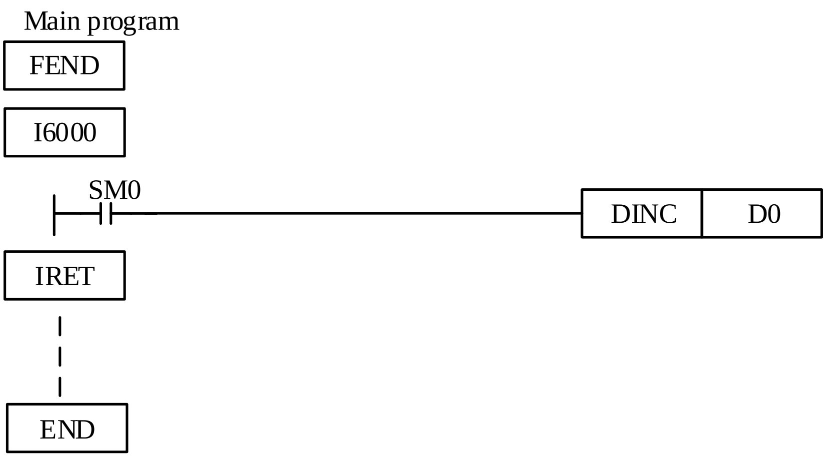


	frequence	pulse count	wait condition	wait register	jump register
1	1000	1000	pulse sending complete	K0	K3
2	2000	2000	pulse sending complete	K0	K6
3	3000	3000	pulse sending complete	K0	K5
4	4000	4000	pulse sending complete	K0	K0
5	5000	5000	pulse sending complete	K0	K2
6	6000	6000	pulse sending complete	K0	K4

As the above pulse configure table, the PLSR pulse sending order is segment 1, 3, 5, 2, 6, 4. The corresponding interruptions are I6000, I6002, I6004, I6001, I6005, I6003.

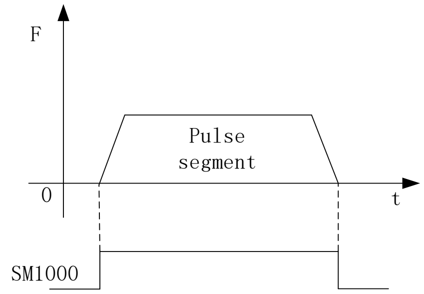


Note: the interruption program is the same to external interruption program



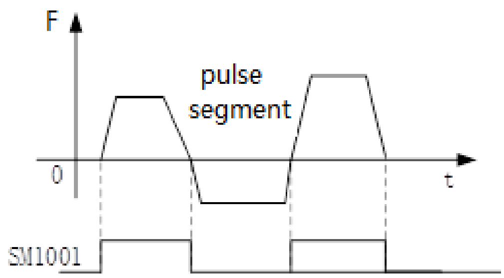
- **Pulse monitoring coil and register**
- Pulse sending flag

No.	Coil	Axis no.	Explanation
1	SM1000	PULSE_1	When the pulse is sending, the coil is ON, the coil is OFF when the pulse sending finished. To judge the pulse sending end by the falling edge of the coil signal.
2	SM1020	PULSE_2	
3	SM1040	PULSE_3	
4	SM1060	PULSE_4	
5	SM1080	PULSE_5	
6	SM1100	PULSE_6	
7	SM1120	PULSE_7	
8	SM1140	PULSE_8	
9	SM1160	PULSE_9	
10	SM1180	PULSE_10	



- Pulse sending direction flag

No.	Coil	Axis no.	Explanation
1	SM1001	PULSE_1	When the pulse number is positive value and sends positive direction pulse, the coil is ON. When the pulse number is negative value and sends negative direction pulse, the coil is OFF.
2	SM1021	PULSE_2	
3	SM1041	PULSE_3	
4	SM1061	PULSE_4	
5	SM1081	PULSE_5	
6	SM1101	PULSE_6	
7	SM1121	PULSE_7	
8	SM1141	PULSE_8	
9	SM1161	PULSE_9	
10	SM1181	PULSE_10	



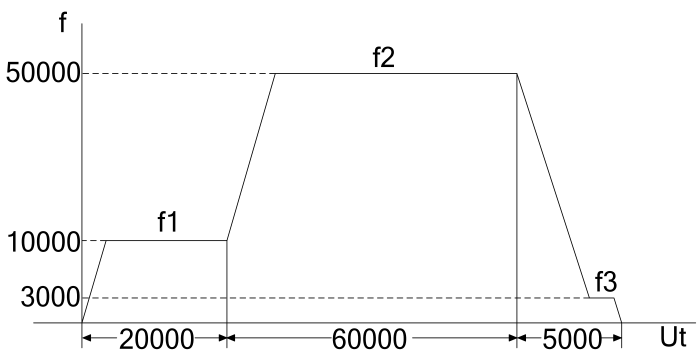
- High speed pulse special data register HSD (power-off retentive)

No.	Function	Explanation	Axis no.
HSD0	Accumulated pulse number low 16 bits	The unit is pulse number	PULSE_1
HSD1	Accumulated pulse number high 16 bits		

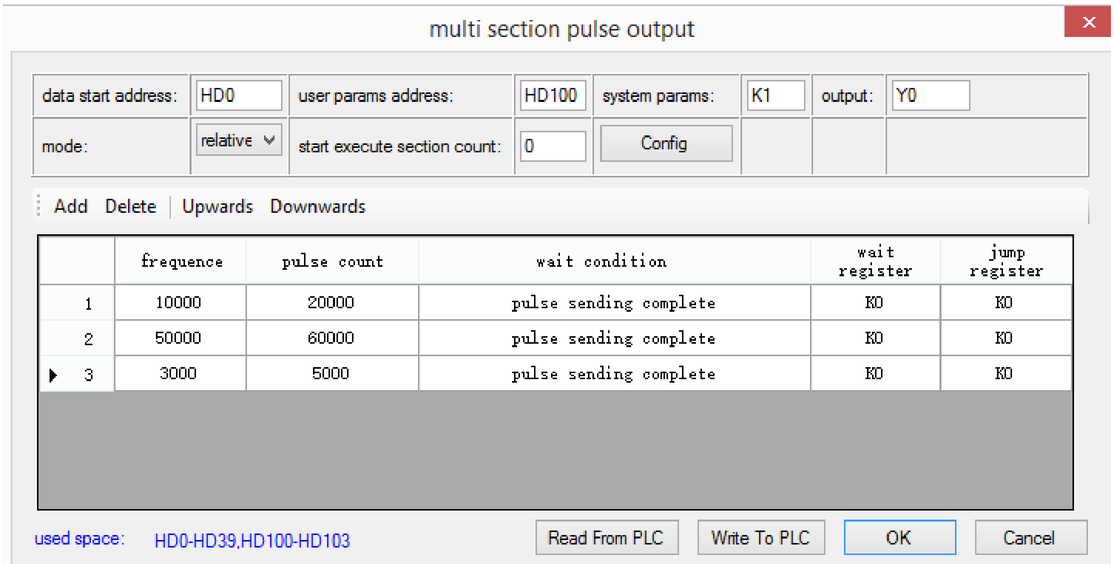
HSD2	Accumulated pulse number low 16 bits	The unit is pulse equivalent	
HSD3	Accumulated pulse number high 16 bits		
HSD4	Accumulated pulse number low 16 bits	The unit is pulse number	PULSE_2
HSD5	Accumulated pulse number high 16 bits		
HSD6	Accumulated pulse number low 16 bits	The unit is pulse equivalent	
HSD7	Accumulated pulse number high 16 bits		
HSD8	Accumulated pulse number low 16 bits	The unit is pulse number	PULSE_3
HSD9	Accumulated pulse number high 16 bits		
HSD10	Accumulated pulse number low 16 bits	The unit is pulse equivalent	
HSD11	Accumulated pulse number high 16 bits		
HSD12	Accumulated pulse number low 16 bits	The unit is pulse number	PULSE_4
HSD13	Accumulated pulse number high 16 bits		
HSD14	Accumulated pulse number low 16 bits	The unit is pulse equivalent	
HSD15	Accumulated pulse number high 16 bits		
HSD16	Accumulated pulse number low 16 bits	The unit is pulse number	PULSE_5
HSD17	Accumulated pulse number high 16 bits		
HSD18	Accumulated pulse number low 16 bits	The unit is pulse equivalent	
HSD19	Accumulated pulse number high 16 bits		
HSD20	Accumulated pulse number low 16 bits	The unit is pulse number	PULSE_6
HSD21	Accumulated pulse number high 16 bits		
HSD22	Accumulated pulse number low 16 bits	The unit is pulse equivalent	
HSD23	Accumulated pulse number high 16 bits		
HSD24	Accumulated pulse number low 16 bits	The unit is pulse number	PULSE_7
HSD25	Accumulated pulse number high 16 bits		
HSD26	Accumulated pulse number low 16 bits	The unit is pulse equivalent	

HSD27	Accumulated pulse number high 16 bits		
HSD28	Accumulated pulse number low 16 bits	The unit is pulse number	PULSE_8
HSD29	Accumulated pulse number high 16 bits		
HSD30	Accumulated pulse number low 16 bits	The unit is pulse equivalent	
HSD31	Accumulated pulse number high 16 bits		
HSD32	Accumulated pulse number low 16 bits	The unit is pulse number	PULSE_9
HSD33	Accumulated pulse number high 16 bits		
HSD34	Accumulated pulse number low 16 bits	The unit is pulse equivalent	
HSD35	Accumulated pulse number high 16 bits		
HSD36	Accumulated pulse number low 16 bits	The unit is pulse number	PULSE_10
HSD37	Accumulated pulse number high 16 bits		
HSD38	Accumulated pulse number low 16 bits	The unit is pulse equivalent	
HSD39	Accumulated pulse number high 16 bits		

Instruction mode



Pulse wave form



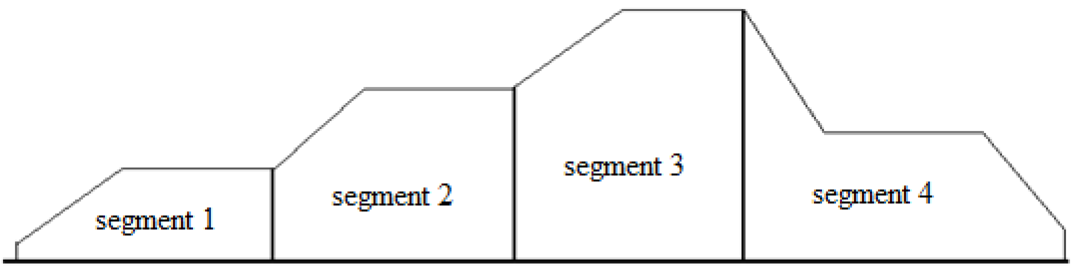
Pulse configuration table

Application

The following curves are set when the curve acceleration time is 0.

1. multi-segment pulse output

Pulse segment



Devide the pulse segment as the above.

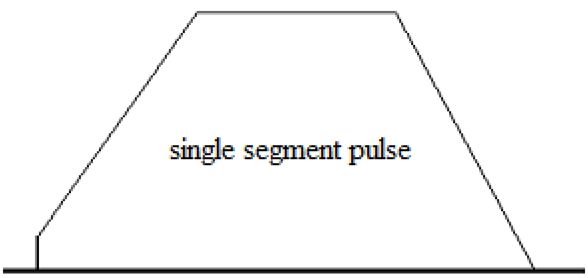
Except the last segment, other segments contain rising, falling or stable part.

The last segment contains rising, falling and stable part.

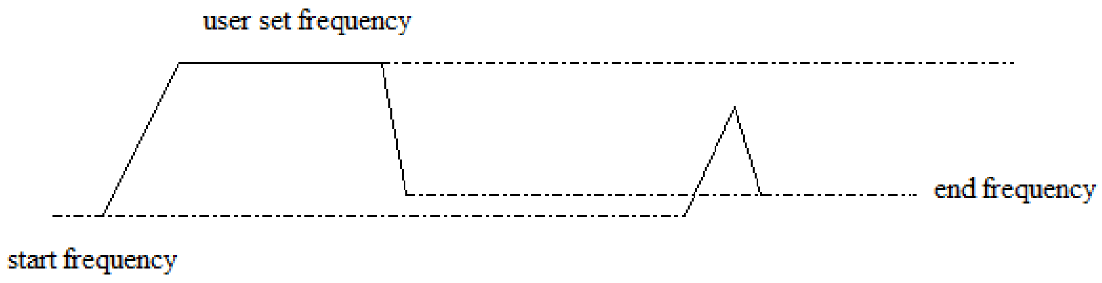
Single segment pulse wave form

There are enough pulse numbers

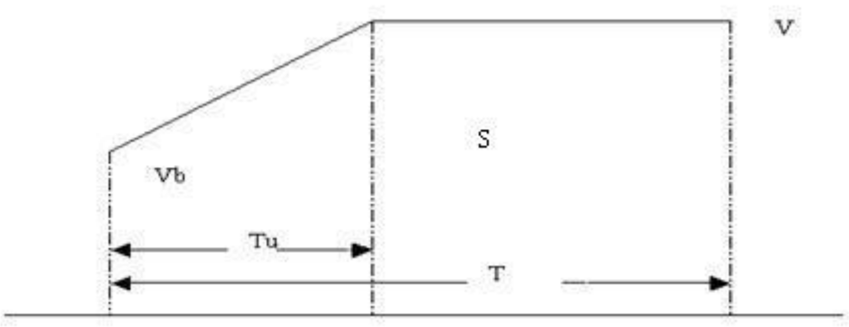
Pulse can up to the max frequency set by user, the wave form is ladder-shape



There is few pulses
Pulse wave form is triangle

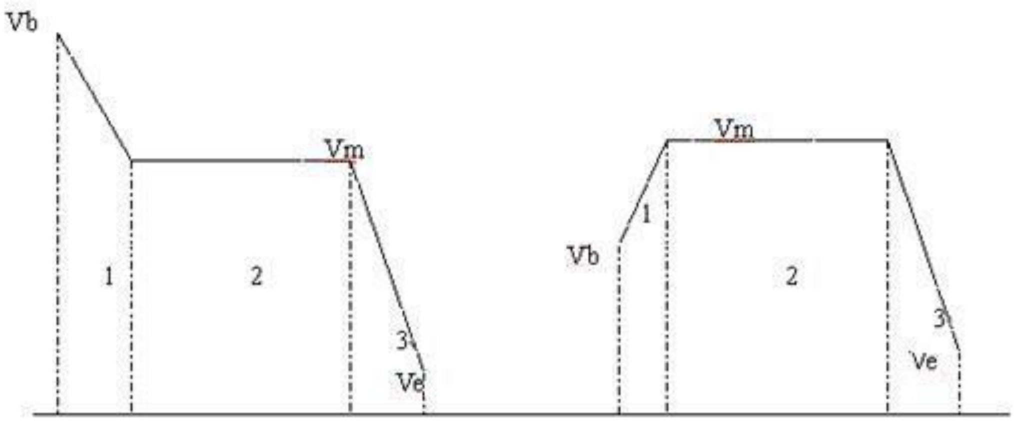


Send one segment of pulse



V: user set the frequency of current segment
S: pulse amounts of current segment
Vb: start frequency of current segment
T: sending pulse time of current segment
Tu: pulse rising/falling time (Tu = (V-VB) / K , K is slope)

The last segment



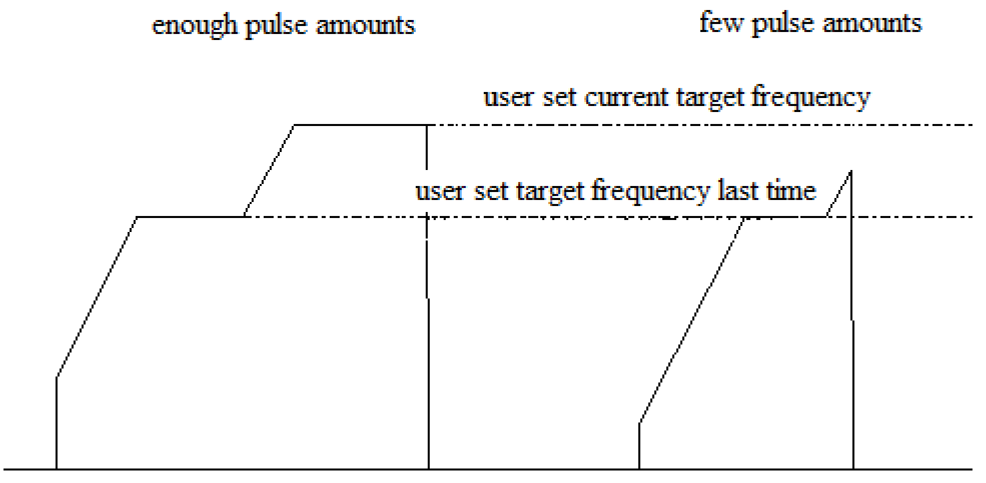
The last segment contains rising, falling and stable part.

Pulse amount is 0

If pulse amounts or frequency is 0, it will send pulse with default speed.

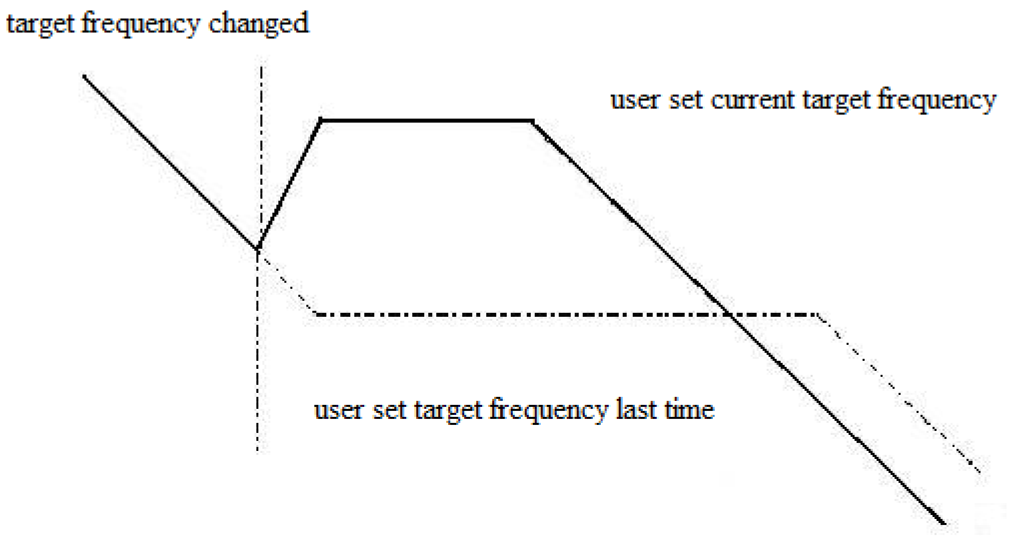
Modify the pulse frequency dynamically

Not the last segment



If user changes the current pulse frequency, it will get to the target frequency with the slope.

The last segment



If user changes the current frequency, PLC will send pulse as the new pulse wave.

Application 1

There are 3 pulse segments. Pulse channel is Y0. Pulse direction channel is Y2. All the parameters please see below tables.

Name	Frequency (Hz)	Pulse amounts
Segment 1	1000	2000
Segment 2	200	1000
Segment 3	2000	6000
Acceleration/deceleration	Frequency changes 1000Hz every 100ms	

Pulse parameters address:

Address	Explanation	Value
HD0 (dword)	Pulse segment quantity (1~100)	3
HD2 (8 words)	Reserved (8 words)	0
HD10 (dword)	Pulse frequency (segment 1)	1000
HD12 (dword)	Pulse quantity (segment 1)	2000
HD14	bit15~bit8: wait condition (segment 1) 00: pulse sending end 01: wait time 02: wait signal 03: ACT time 04: EXT signal 05: EXT signal or pulse sending end bit7~bit0: wait condition register type 00: constant 01: D 02: X 03: M	0
HD15 (dword)	Constant/register(waiting condition) (segment 1)	0
HD17	bit7~bit0: jump register type 00: constant 01: D	0
HD+18 (dword)	Constant/register (jump register) (segment 1)	0
HD+20 (dword)	Pulse frequency (segment 2)	200
HD+22 (dword)	Pulse quantity (segment 2)	1000
HD+24	Wait condition, wait condition register type (segment 2)	0
HD+25 (dword)	Constant or register (waiting condition) (segment 2)	0
HD+27	Jump type, jump register type (waiting condition) (segment 2)	0
HD+28 (dword)	Constant or register (jump register) (segment 2)	0
HD+30 (dword)	Pulse frequency (segment 3)	2000

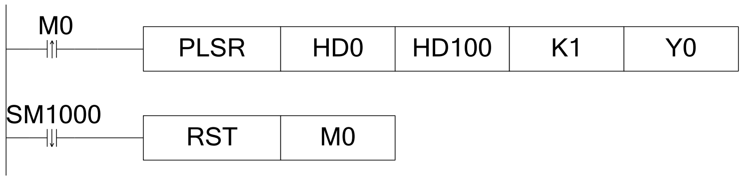
HD+32 (dword)	Pulse quantity (segment 3)	6000
HD+34	Wait condition，wait condition register type (segment 3)	0
HD+35 (dword)	Constant or register (waiting condition) (segment 3)	0
HD+37	Jump type, jump register type (waiting condition) (segment 3)	0
HD+38 (dword)	Constant or register (jump register) (segment 3)	0

System parameters address:

SFD900	Pulse parameters	Bit 0: pulse output logic 0: positive logic(default setting) 1: negative logic, Bit 1：pulse direction logic 0: positive logic(default setting) 1: negative logic Bit 8: pulse unit 0: pulse quantity(default setting) 1: equivalent value	0	Public parameters
SFD901	Reserved			
SFD902	Pulse quantity/ 1 rotation low 16 bits		0	
SFD903	Pulse quantity/ 1 rotation high 16 bits		0	
SFD904	Movement amount/1 rotation low 16 bits		0	
SFD905	Movement amount/1 rotation high 16 bits		0	
SFD906	Pulse direction terminal	The number of Y, 0xFF is no terminal	2	
SFD907	Direction delay time	Default value is 20, unit is ms	20	
SFD908	Gear clearance positive compensation		0	
SFD909	Gear clearance negative compensation		0	
SFD910	Electrical origin low 16 bits		0	
SFD911	Electrical origin high 16 bits		0	
SFD912	Machine back to origin parameters	Bit0: promixity switch state 0: normal open 1: normal close	0	
SFD913	Near signal terminal	Bit0~bit7: the X terminal number, 0xFF is no terminal	0xFF	
SFD914	Z phase terminal	Bit0~bit7: the X terminal number, 0xFF is no terminal	0xFF	
SFD915	Limit terminal	Bit7~bit0: limit 1 X terminal number, 0xFF is no terminal Bit15~bit8: limit 2 X terminal number, 0xFF is no terminal	FFFF	

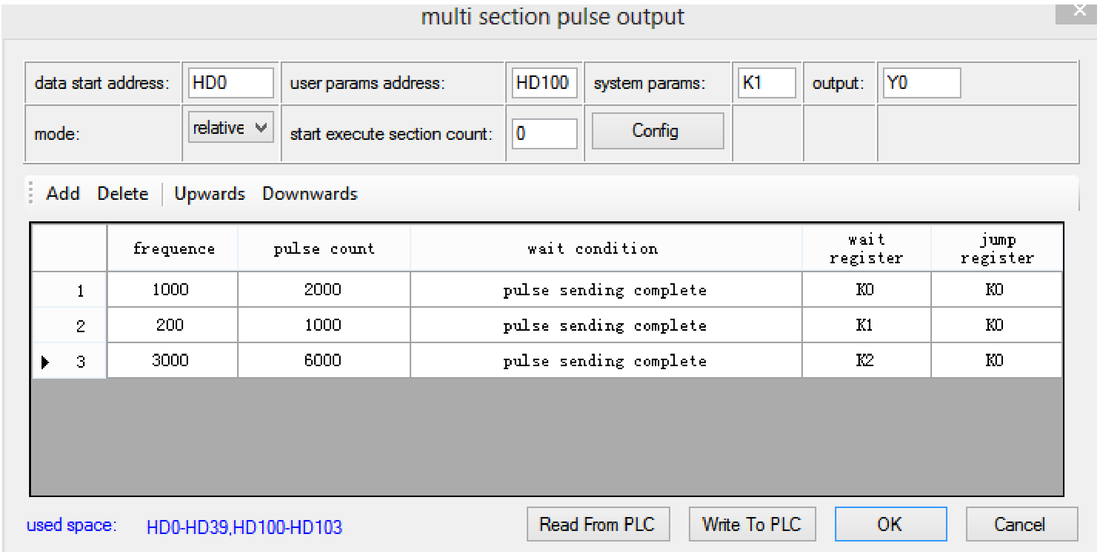
SFD916	Origin auxiliary signal terminal	Bit0~bit7: X terminal number, 0xFF is no terminal	0xFF	
SFD917	CLR signal output terminal	Bit0~bit7: Y terminal number, 0xFF is no terminal	0xFF	
SFD918	Back speed VH low 16 bits		0	
SFD919	Back speed VH high 16 bits		0	
SFD920	Back speed VL low 16 bits		0	
SFD921	Back speed VL high 16 bits		0	
SFD922	Creep speed low 16 bits		0	
SFD923	Creep speed high 16 bits		0	
SFD924	Mechanical origin low 16 bits		0	
SFD925	Mechanical origin high 16 bits		0	
SFD926	Z phase quantity		0	
SFD927	CLR signal delay time	Default value 20, unit: ms	20	
SFD936	G instruction which group of parameter choice	G instruction system parameter group choice	2	
…				
SFD950	Pulse default speed low 16 bits	Send pulse with default speed when speed is 0	1000	Group 1 parameters
SFD951	Pulse default speed high 16 bits		0	
SFD952	Pulse default speed acceleration time		100	
SFD953	Pulse default speed deceleration time		100	
SFD954	Tween acceleration/deceleration time		0	
SFD955	Reserved			
SFD956	Max speed limit low 16 bits		3392	
SFD957	Max speed limit high 16 bits		3	
SFD958	Start speed low 16 bits		0	
SFD959	Start speed high 16 bits		0	
SFD960	End speed low 16 bits		0	
SFD961	End speed high 16 bits		0	
SFD962	Curve acceleration time (ms)		0	
…				

Pulse instruction

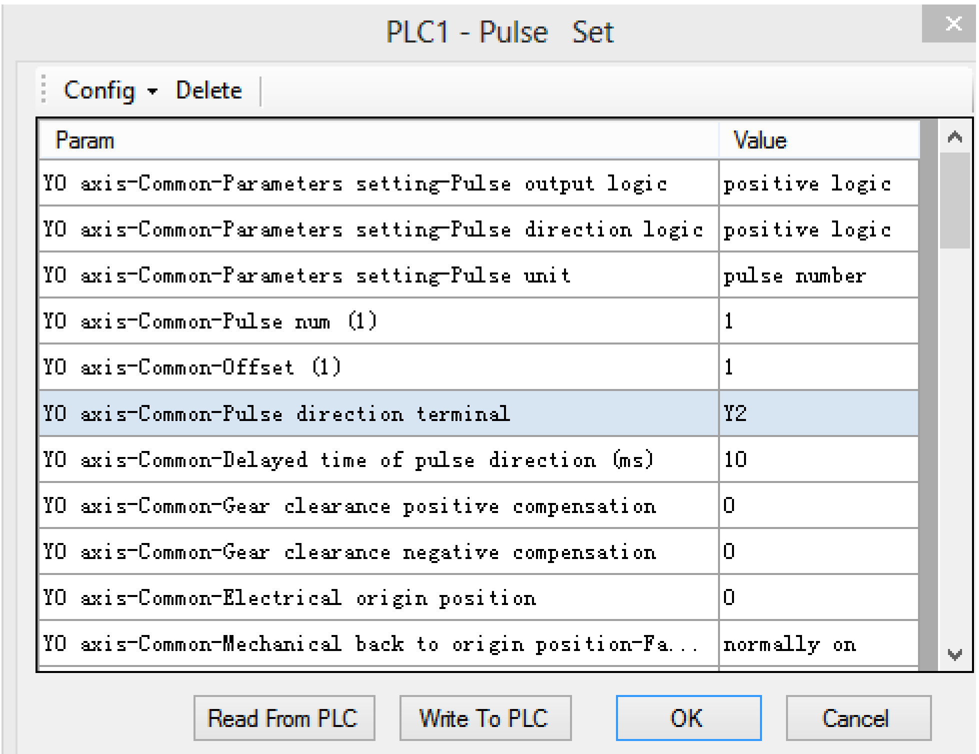


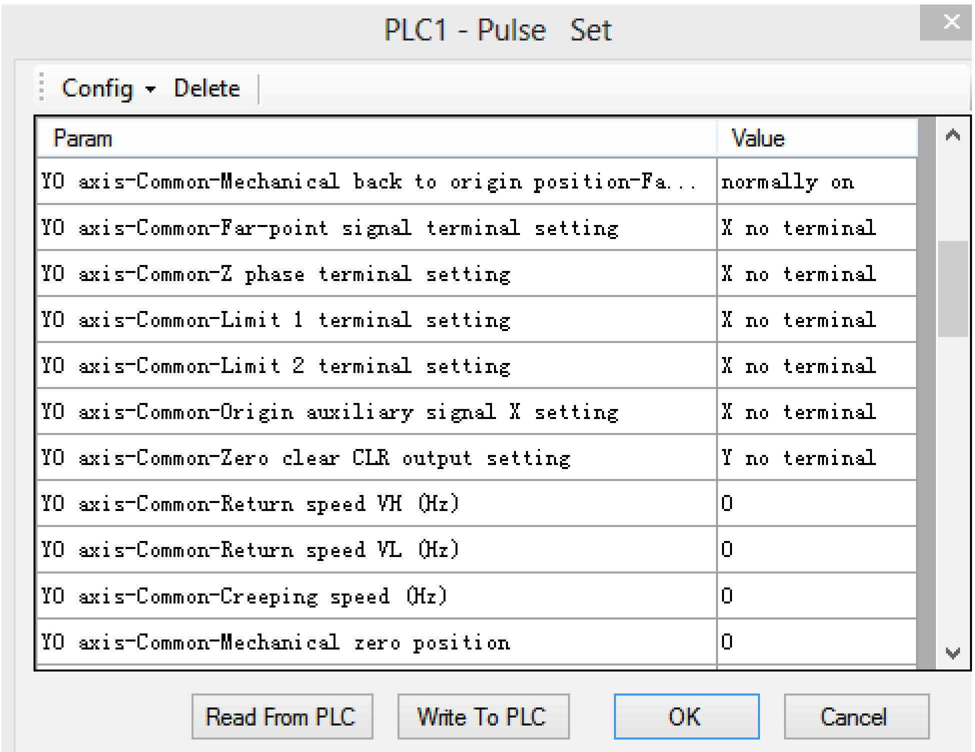
XDPpro software configuration:

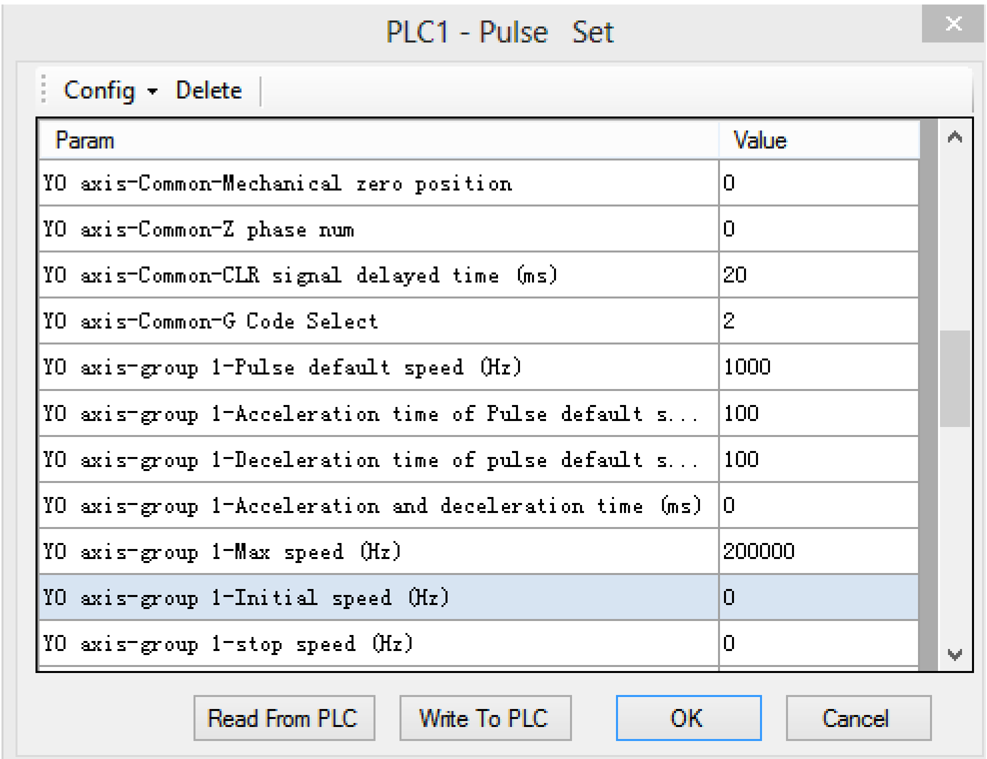
Pulse segment configuration

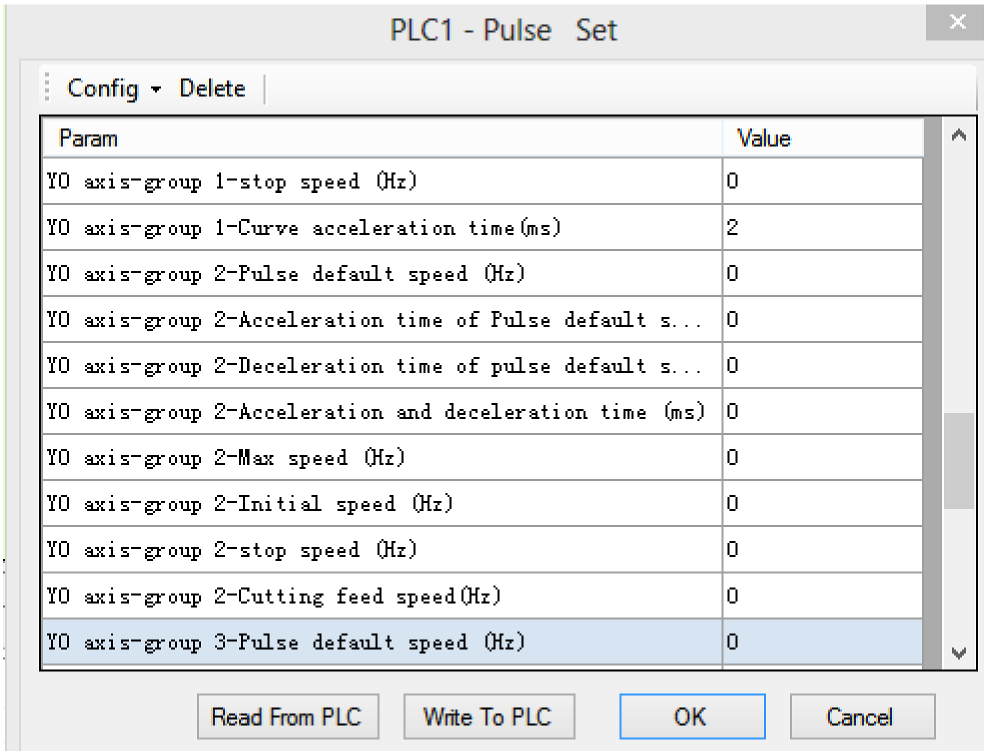


Pulse configuration parameters




PLC1 - Pulse Set
Config
Delete
Param
Value
Y0 axis-Common-Mechanical back to origin position-Fa...
normally on
Y0 axis-Common-Far-point signal terminal setting
X no terminal
Y0 axis-Common-Z phase terminal setting
X no terminal
Y0 axis-Common-Limit 1 terminal setting
X no terminal
Y0 axis-Common-Limit 2 terminal setting
X no terminal
Y0 axis-Common-Origin auxiliary signal X setting
X no terminal
Y0 axis-Common-Zero clear CLR output setting
Y no terminal
Y0 axis-Common-Return speed VH (Hz)
0
Y0 axis-Common-Return speed VL (Hz)
0
Y0 axis-Common-Creeping speed (Hz)
0
Y0 axis-Common-Mechanical zero position
0
Read From PLC
Write To PLC
OK
Cancel


PLC1 - Pulse Set
Config
Delete
Param
Value
Y0 axis-Common-Mechanical zero position
0
Y0 axis-Common-Z phase num
0
Y0 axis-Common-CLR signal delayed time (ms)
20
Y0 axis-Common-G Code Select
2
Y0 axis-group 1-Pulse default speed (Hz)
1000
Y0 axis-group 1-Acceleration time of Pulse default s...
100
Y0 axis-group 1-Deceleration time of pulse default s...
100
Y0 axis-group 1-Acceleration and deceleration time (ms)
0
Y0 axis-group 1-Max speed (Hz)
200000
Y0 axis-group 1-Initial speed (Hz)
0
Y0 axis-group 1-stop speed (Hz)
0
Read From PLC
Write To PLC
OK
Cancel



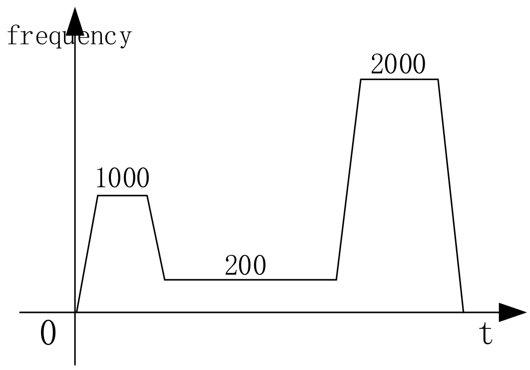
PLC1 - Pulse Set

Config Delete

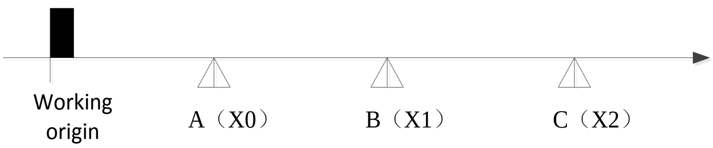
Param	Value
Y0 axis-group 1-stop speed (Hz)	0
Y0 axis-group 1-Curve acceleration time(ms)	2
Y0 axis-group 2-Pulse default speed (Hz)	0
Y0 axis-group 2-Acceleration time of Pulse default s...	0
Y0 axis-group 2-Deceleration time of pulse default s...	0
Y0 axis-group 2-Acceleration and deceleration time (ms)	0
Y0 axis-group 2-Max speed (Hz)	0
Y0 axis-group 2-Initial speed (Hz)	0
Y0 axis-group 2-stop speed (Hz)	0
Y0 axis-group 2-Cutting feed speed(Hz)	0
Y0 axis-group 3-Pulse default speed (Hz)	0

Read From PLC Write To PLC OK Cancel

Pulse curve diagram:



Application 2



As the above diagram, there are three segments. The distance between A, B, C is unknown. The distance of A-B, B-C, working oring-A are the same, but the moving speed is different. The instruction PLSR can make the function. Fisrt, install three proximity switches at A, B, C. then connect them to PLC terminal X0, X1, X2. Pulse output terminal is Y0, pulse direction terminal is Y2. All the parameters please refer to the following table.

Name	Frequency (Hz)	Pulse amounts
Working origin-A	1000	999999999
A-B	3000	999999999
B-C	2000	999999999
Acceleration deceleration time	Frequency changes 1000Hz every 100ms	

Note: as the pulse amounts of each segment is unknown, set a large value to make sure the object can move to the proximity switch. When the object reaches C, urgent stops the object with STOP instruction.

Pulse parameter address:

Address	Explanation	Value
HD0 (dword)	Pulse segment quantity (1~100)	3
HD2 (8 words)	Reserved (8 words)	0
HD10 (dword)	Pulse frequency (segment 1)	1000
HD12 (dword)	Pulse quantity (segment 1)	999999999
HD14	bit15~bit8: wait condition (segment 1) 00: pulse sending end 01: wait time 02: wait signal 03: ACT time 04: EXT signal 05: EXT signal or pulse sending end bit7~bit0: wait condition register type 00: constant 01: D 02: X 03: M	1026
HD15 (dword)	Constant/register(waiting condition) (segment 1)	0
HD17	bit7~bit0: jump register type 00: constant 01: D	0
HD+18 (dword)	Constant/register (jump register) (segment 1)	0
HD+20 (dword)	Pulse frequency (segment 2)	3000
HD+22 (dword)	Pulse quantity (segment 2)	999999999
HD+24	Wait condition, wait condition register type (segment 2)	1026
HD+25 (dword)	Constant or register (waiting condition) (segment 2)	1
HD+27	Jump type, jump register type (waiting condition) (segment 2)	0
HD+28 (dword)	Constant or register (jump register) (segment 2)	0
HD+30 (dword)	Pulse frequency (segment 3)	2000
HD+32 (dword)	Pulse quantity (segment 3)	999999999

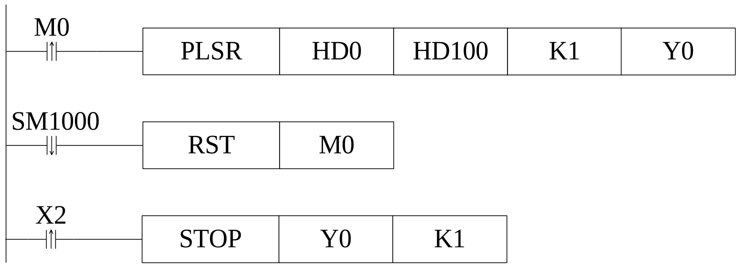
HD+34	Wait condition, wait condition register type (segment 3)	1026
HD+35 (dword)	Constant or register (waiting condition) (segment 3)	2
HD+37	Jump type, jump register type (waiting condition) (segment 3)	0
HD+38 (dword)	Constant or register (jump register) (segment 3)	0

System parameter address:

SFD900	Pulse parameters	Bit 0: pulse output logic 0: positive logic(default setting) 1: negative logic, Bit 1：pulse direction logic 0: positive logic(default setting) 1: negative logic Bit 8: pulse unit 0: pulse quantity(default setting) 1: equivalent value	0	Public parameter
SFD901	Reserved			
SFD902	Pulse quantity/ 1 rotation low 16 bits		0	
SFD903	Pulse quantity/ 1 rotation high 16 bits		0	
SFD904	Movement amount/1 rotation low 16 bits		0	
SFD905	Movement amount/1 rotation high 16 bits		0	
SFD906	Pulse direction terminal	The number of Y, 0xFF is no terminal	2	
SFD907	Direction delay time	Default value is 20, unit is ms	20	
SFD908	Gear clearance positive compensation		0	
SFD909	Gear clearance negative compensation		0	
SFD910	Electrical origin low 16 bits		0	
SFD911	Electrical origin high 16 bits		0	
SFD912	Machine back to origin parameters	Bit0: promixity switch state 0: normal open 1: normal close	0	
SFD913	Near signal terminal	Bit0~bit7: the X terminal number, 0xFF is no terminal	0xFF	
SFD914	Z phase terminal	Bit0~bit7: the X terminal number, 0xFF is no terminal	0xFF	
SFD915	Limit terminal	Bit7~bit0: limit 1 X terminal number, 0xFF is no terminal Bit15~bit8: limit 2 X terminal number, 0xFF is no terminal	FFFF	
SFD916	Origin auxiliary signal terminal	Bit0~bit7: X terminal number, 0xFF is no terminal	0xFF	

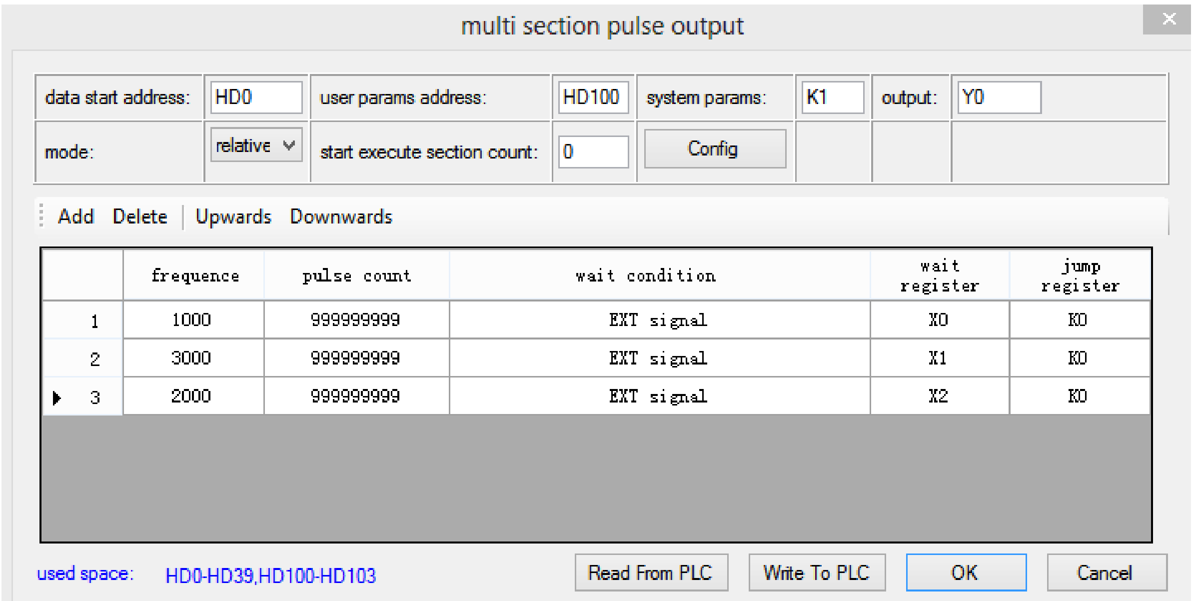
SFD917	CLR signal output terminal	Bit0~bit7: Y terminal number, 0xFF is no terminal	0xFF	
SFD918	Back speed VH low 16 bits		0	
SFD919	Back speed VH high 16 bits		0	
SFD920	Back speed VL low 16 bits		0	
SFD921	Back speed VL high 16 bits		0	
SFD922	Creep speed low 16 bits		0	
SFD923	Creep speed high 16 bits		0	
SFD924	Mechanical origin low 16 bits		0	
SFD925	Mechanical origin high 16 bits		0	
SFD926	Z phase quantity		0	
SFD927	CLR signal delay time	Default value 20, unit: ms	20	
SFD936	G instruction which group of parameter to use	G instruction system parameter group choice	2	
…				
SFD950	Pulse default speed low 16 bits	Send pulse with default speed when speed is 0	1000	Group 1 parameters
SFD951	Pulse default speed high 16 bits		0	
SFD952	Pulse default speed acceleration time		100	
SFD953	Pulse default speed deceleration time		100	
SFD954	Tween acceleration/deceleration time		0	
SFD955	Reserved			
SFD956	Max speed limit low 16 bits		3392	
SFD957	Max speed limit high 16 bits		3	
SFD958	Start speed low 16 bits		0	
SFD959	Start speed high 16 bits		0	
SFD960	End speed low 16 bits		0	
SFD961	End speed high 16 bits		0	
SFD962	Curve acceleration time (ms)		0	
…				

Pulse instruction:

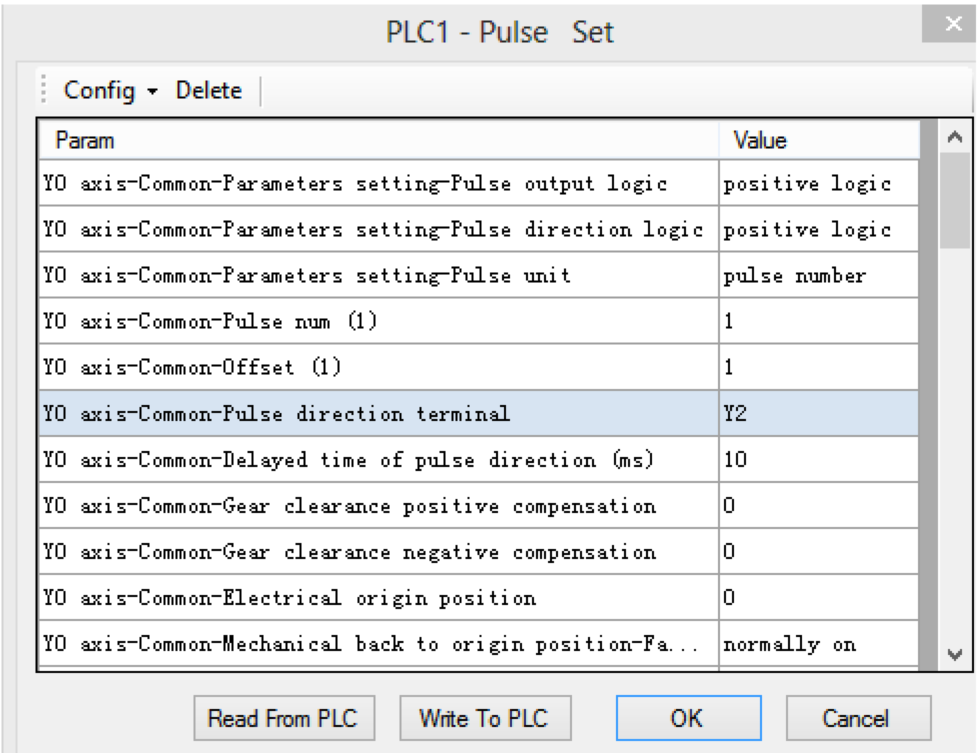


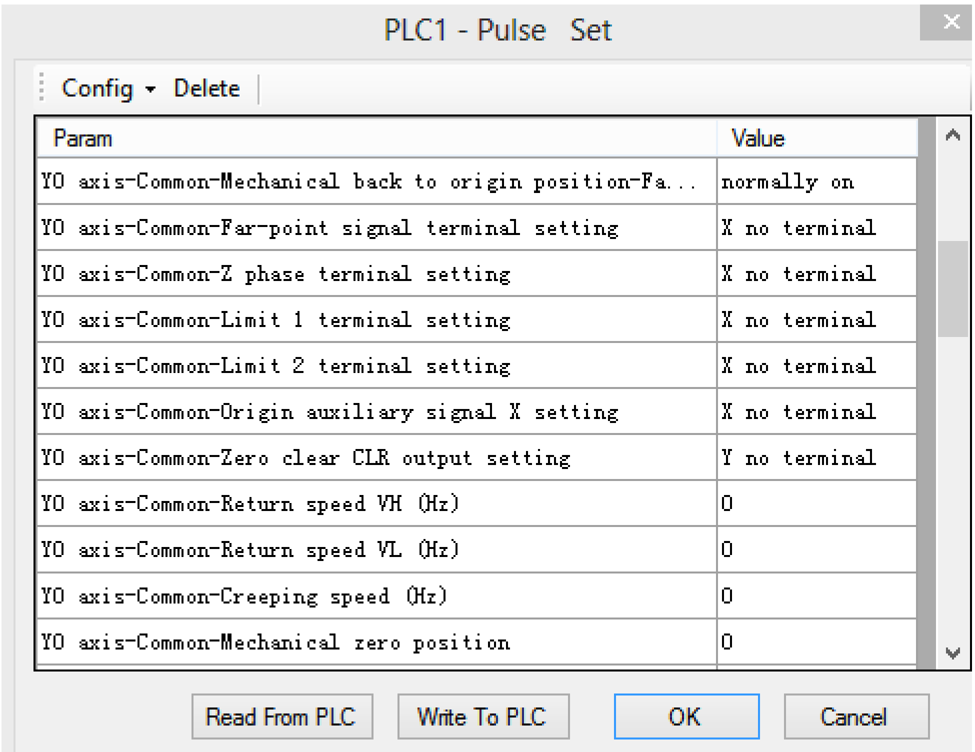
The configuration in the XDPpro software:

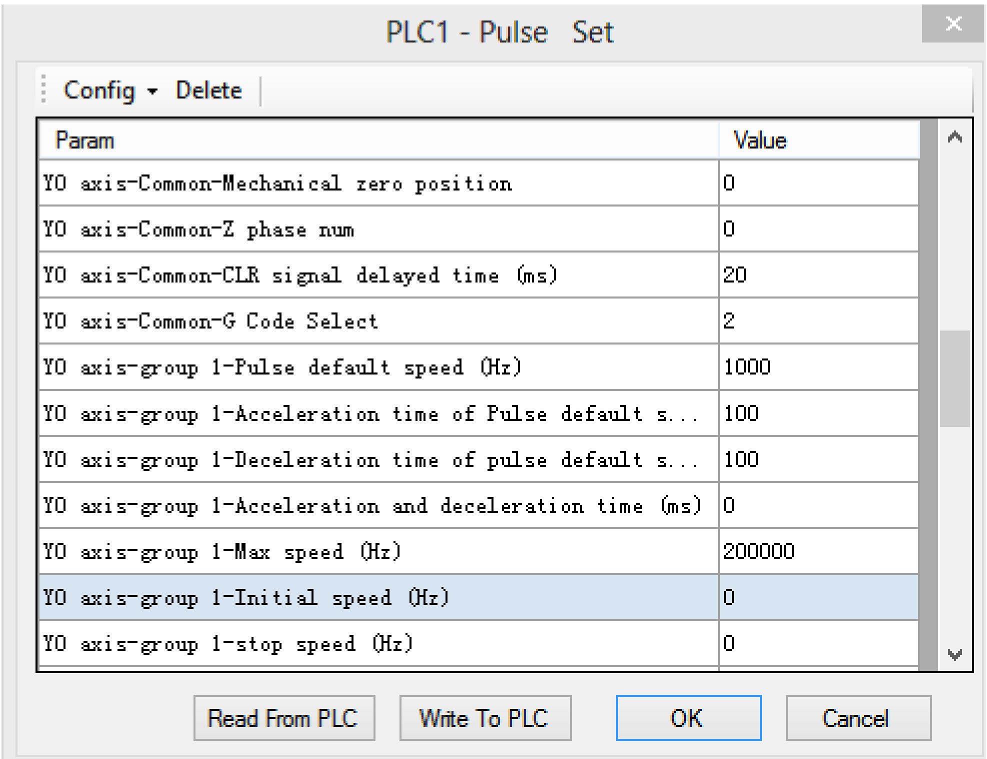
Pulse segment configuration

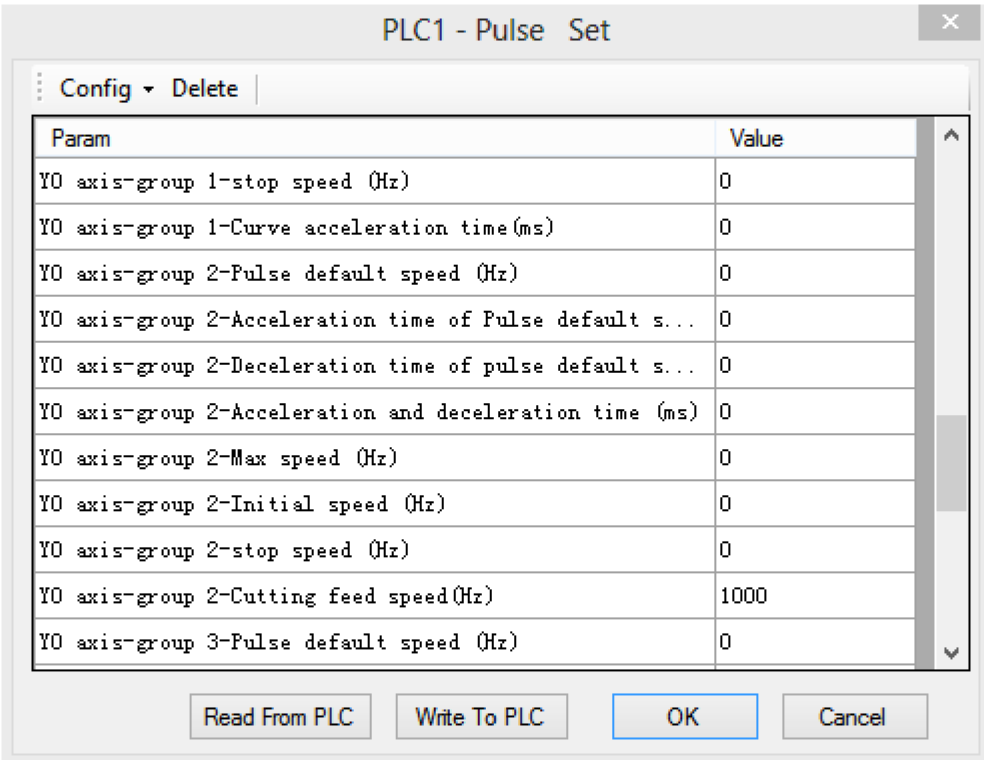


Pulse configuration

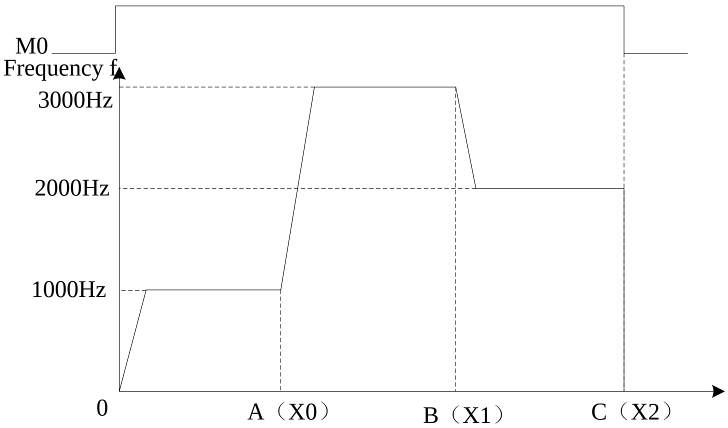



PLC1 - Pulse Set
Config Delete
Param
Value
Y0 axis-Common-Mechanical back to origin position-Fa...
normally on
Y0 axis-Common-Far-point signal terminal setting
X no terminal
Y0 axis-Common-Z phase terminal setting
X no terminal
Y0 axis-Common-Limit 1 terminal setting
X no terminal
Y0 axis-Common-Limit 2 terminal setting
X no terminal
Y0 axis-Common-Origin auxiliary signal X setting
X no terminal
Y0 axis-Common-Zero clear CLR output setting
Y no terminal
Y0 axis-Common-Return speed VH (Hz)
0
Y0 axis-Common-Return speed VL (Hz)
0
Y0 axis-Common-Creeping speed (Hz)
0
Y0 axis-Common-Mechanical zero position
0
Read From PLC
Write To PLC
OK
Cancel


PLC1 - Pulse Set
Config Delete
Param
Value
Y0 axis-Common-Mechanical zero position
0
Y0 axis-Common-Z phase num
0
Y0 axis-Common-CLR signal delayed time (ms)
20
Y0 axis-Common-G Code Select
2
Y0 axis-group 1-Pulse default speed (Hz)
1000
Y0 axis-group 1-Acceleration time of Pulse default s...
100
Y0 axis-group 1-Deceleration time of pulse default s...
100
Y0 axis-group 1-Acceleration and deceleration time (ms)
0
Y0 axis-group 1-Max speed (Hz)
200000
Y0 axis-group 1-Initial speed (Hz)
0
Y0 axis-group 1-stop speed (Hz)
0
Read From PLC
Write To PLC
OK
Cancel

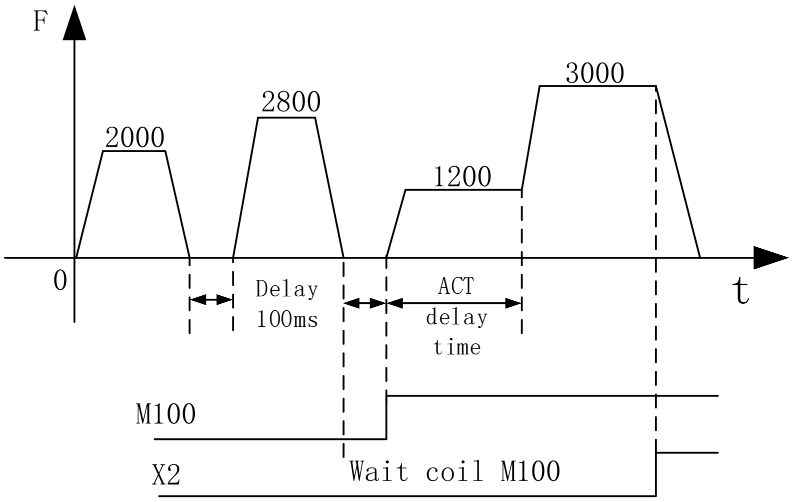


Pulse wave form:

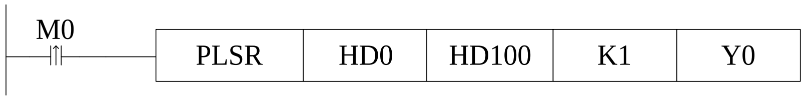


Application 3

There are four segments. Segment 1 frequency is 2000Hz, 3000 pulses, it delays 100ms after segment 1 then goes to segment 2. Segment 2 frequency is 2800Hz, pulse number is 4000. When M100 coil is ON, it goes to segment 3. Segment 3 frequency is 1200Hz, pulse number is 999999999, it delays ACT time 2s then goes to segment 4. Segment 4 frequency is 3000Hz, pulse numbers are 999999999, it deceleration stops the pulse when X2 is ON. The pulse acceleration slope is 80ms every 1000Hz, pulse deceleration slope is 120ms every 1000Hz. Pulse direction terminal is Y2.

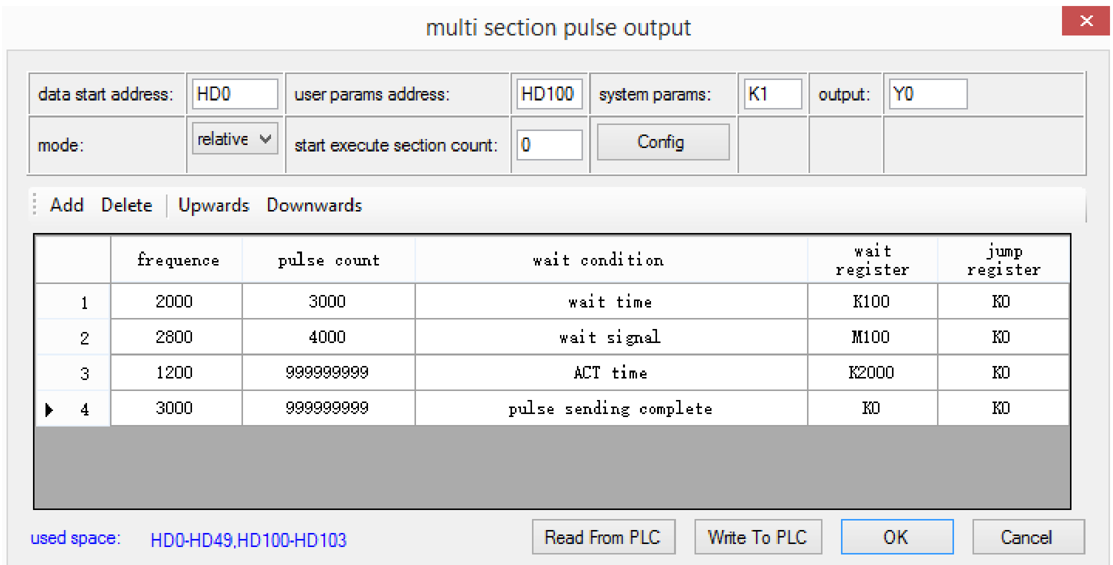


Pulse instruction:



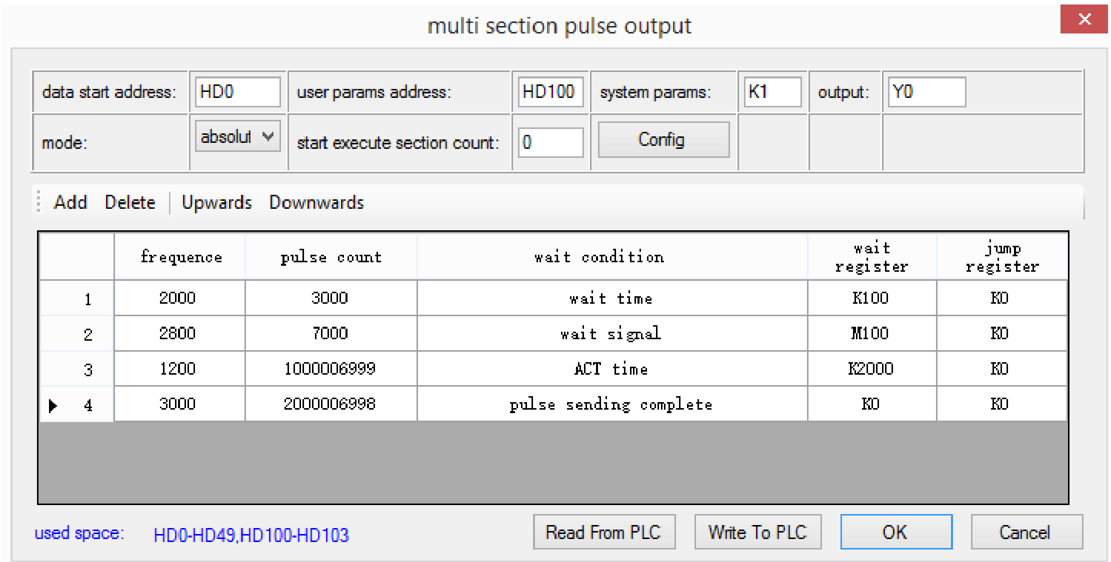
Method 1: set via XDPpro software

Pulse configureation table:



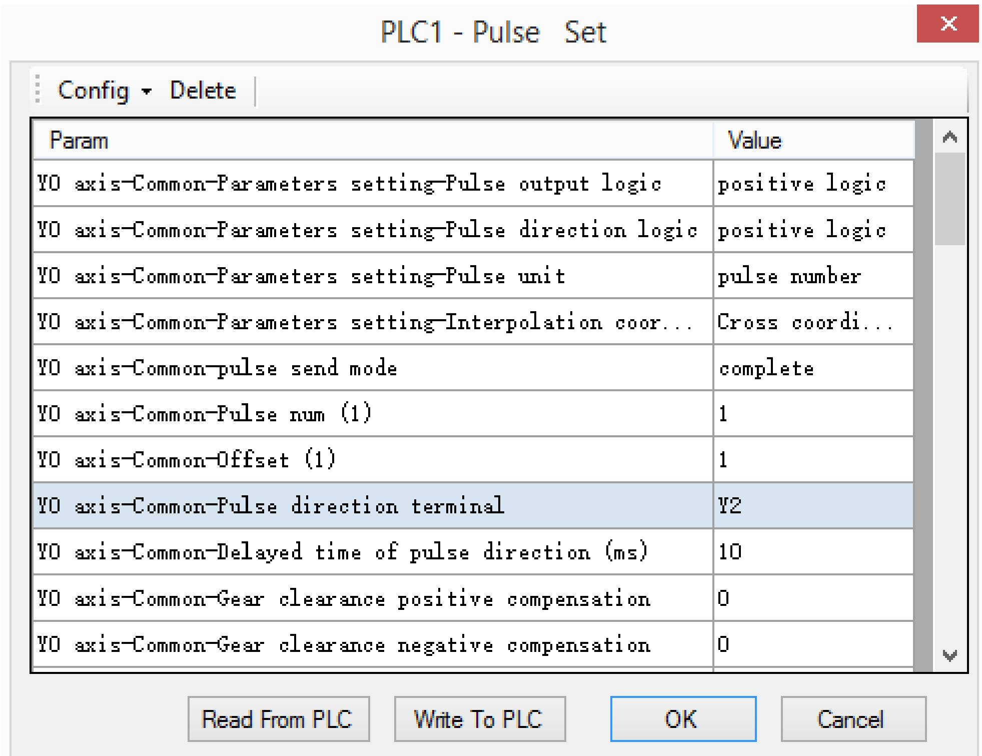
	frequence	pulse count	wait condition	wait register	jump register
1	2000	3000	wait time	K100	K0
2	2800	4000	wait signal	M100	K0
3	1200	999999999	ACT time	K2000	K0
4	3000	999999999	pulse sending complete	K0	K0

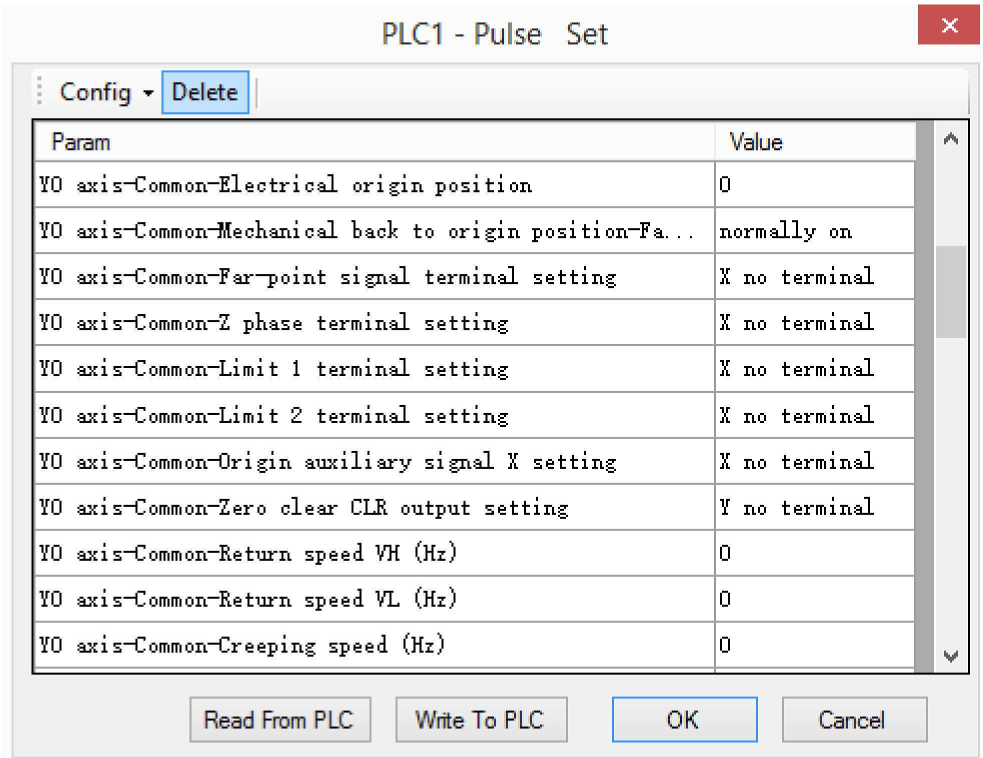
Relative mode

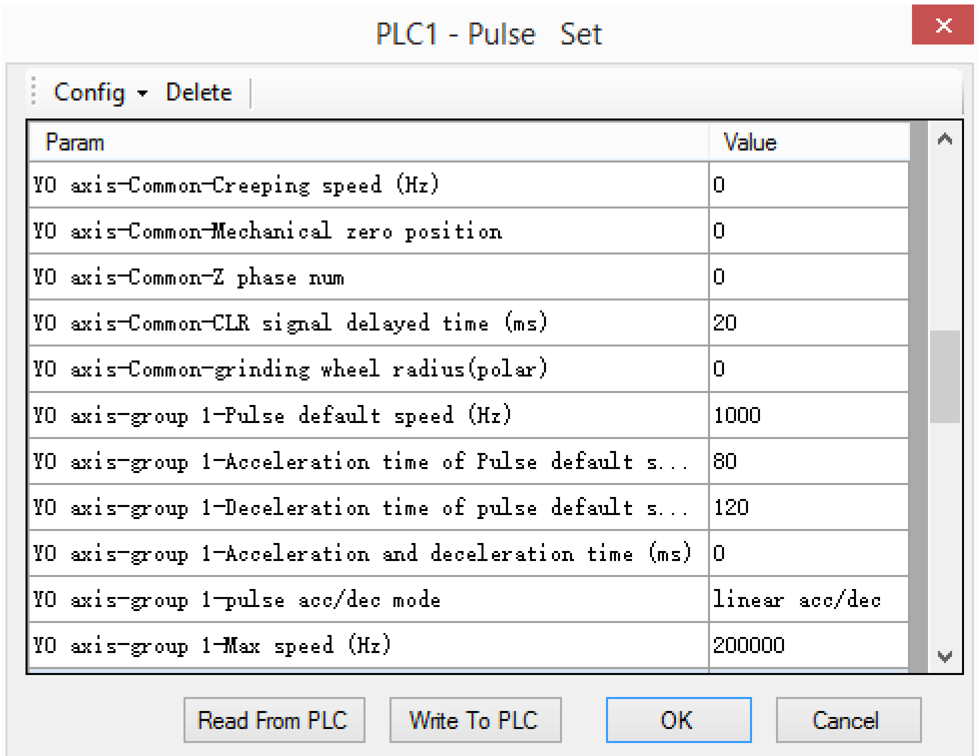


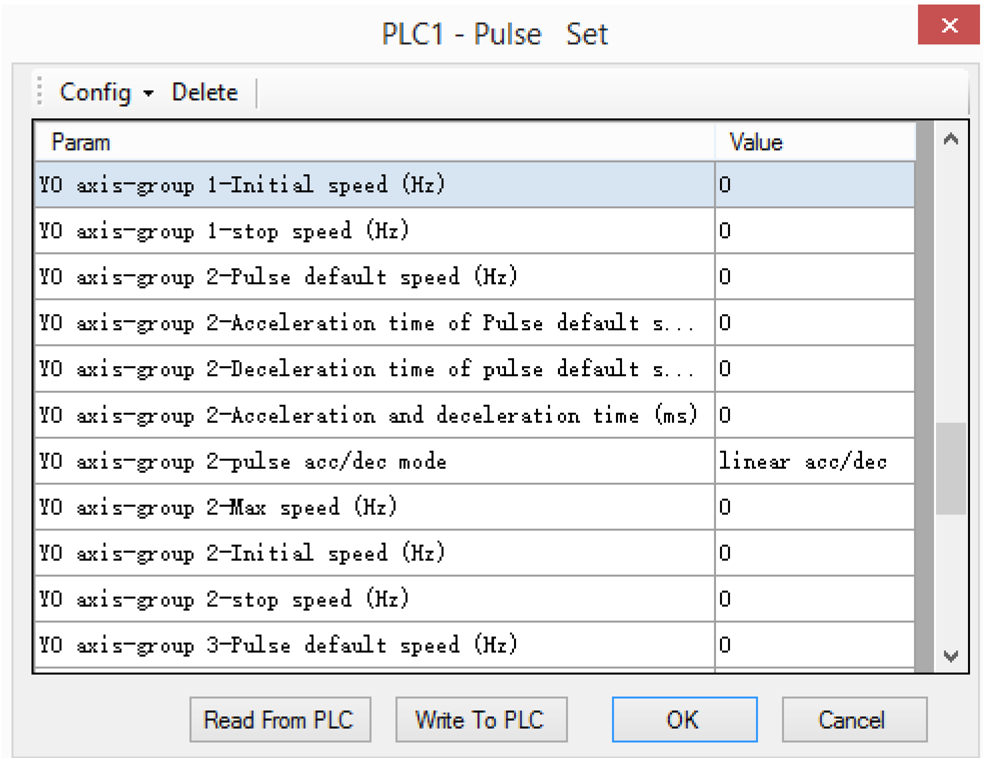
Absolute mode

System parameters configuration




PLC1 - Pulse Set
Config
Delete
Param
Value
Y0 axis-Common-Electrical origin position
0
Y0 axis-Common-Mechanical back to origin position-Fa...
normally on
Y0 axis-Common-Far-point signal terminal setting
X no terminal
Y0 axis-Common-Z phase terminal setting
X no terminal
Y0 axis-Common-Limit 1 terminal setting
X no terminal
Y0 axis-Common-Limit 2 terminal setting
X no terminal
Y0 axis-Common-Origin auxiliary signal X setting
X no terminal
Y0 axis-Common-Zero clear CLR output setting
Y no terminal
Y0 axis-Common-Return speed VH (Hz)
0
Y0 axis-Common-Return speed VL (Hz)
0
Y0 axis-Common-Creeping speed (Hz)
0
Read From PLC
Write To PLC
OK
Cancel


PLC1 - Pulse Set
Config
Delete
Param
Value
Y0 axis-Common-Creeping speed (Hz)
0
Y0 axis-Common-Mechanical zero position
0
Y0 axis-Common-Z phase num
0
Y0 axis-Common-CLR signal delayed time (ms)
20
Y0 axis-Common-grinding wheel radius(polar)
0
Y0 axis-group 1-Pulse default speed (Hz)
1000
Y0 axis-group 1-Acceleration time of Pulse default s...
80
Y0 axis-group 1-Deceleration time of pulse default s...
120
Y0 axis-group 1-Acceleration and deceleration time (ms)
0
Y0 axis-group 1-pulse acc/dec mode
linear acc/dec
Y0 axis-group 1-Max speed (Hz)
200000
Read From PLC
Write To PLC
OK
Cancel



Method 2: set the following registers

Pulse data address (relative mode)

Address	Explanation	Value
HD0 (dword)	Pulse segment quantity (1~100)	4
HD2 (8 words)	Reserved (8 words)	0
HD10 (dword)	Pulse frequency (segment 1)	2000
HD12 (dword)	Pulse quantity (segment 1)	3000
HD14	bit15~bit8: wait condition (segment 1) 00: pulse sending end 01: wait time 02: wait signal 03: ACT time 04: EXT signal 05: EXT signal or pulse sending end bit7~bit0: wait condition register type 00: constant 01: D 02: X 03: M	256
HD15 (dword)	Constant/register(waiting condition) (segment 1)	100
HD17	bit7~bit0: jump register type 00: constant 01: D	0
HD+18 (dword)	Constant/register (jump register) (segment 1)	0
HD+20 (dword)	Pulse frequency (segment 2)	2800

HD+22 (dword)	Pulse quantity (segment 2)	4000
HD+24	Wait condition, wait condition register type (segment 2)	515
HD+25 (dword)	Constant or register (waiting condition) (segment 2)	100
HD+27	Jump type, jump register type (waiting condition) (segment 2)	0
HD+28 (dword)	Constant or register (jump register) (segment 2)	0
HD+30 (dword)	Pulse frequency (segment 3)	1200
HD+32 (dword)	Pulse quantity (segment 3)	999999999
HD+34	Wait condition, wait condition register type (segment 3)	768
HD+35 (dword)	Constant or register (waiting condition) (segment 3)	2000
HD+37	Jump type, jump register type (waiting condition) (segment 3)	0
HD+38 (dword)	Constant or register (jump register) (segment 3)	0
HD+40 (dword)	Pulse frequency (segment 3)	3000
HD+42 (dword)	Pulse quantity (segment 3)	999999999
HD+44	Wait condition, wait condition register type (segment 3)	768
HD+45 (dword)	Constant or register (waiting condition) (segment 3)	2000
HD+47	Jump type, jump register type (waiting condition) (segment 3)	0
HD+48 (dword)	Constant or register (jump register) (segment 3)	0

System parameter address:

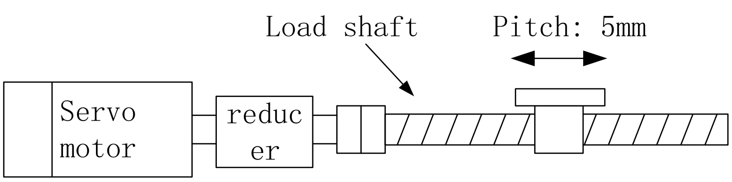
SFD900	Pulse parameters	Bit 0: pulse output logic 0: positive logic(default setting) 1: negative logic, Bit 1： pulse direction logic 0: positive logic(default setting) 1: negative logic Bit 8: pulse unit 0: pulse quantity(default setting) 1: equivalent value	0	Public parameter
SFD901	Reserved		0	
SFD902	Pulse quantity/ 1 rotation low 16 bits		1	
SFD903	Pulse quantity/ 1 rotation high 16 bits		0	
SFD904	Movement amount/1 rotation low 16 bits		1	

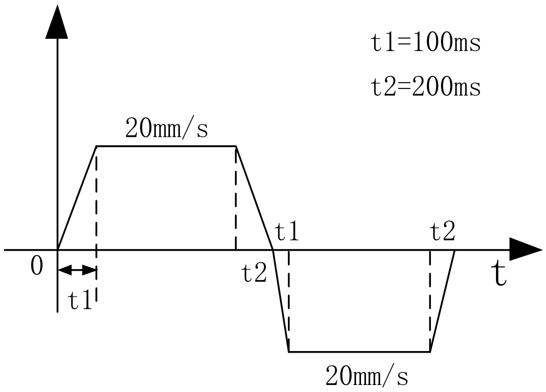
SFD905	Movement amount/1 rotation high 16 bits		0	
SFD906	Pulse direction terminal	The number of Y, 0xFF is no terminal	2	
SFD907	Direction delay time	Default value is 20, unit is ms	10	
SFD908	Gear clearance positive compensation		0	
SFD909	Gear clearance negative compensation		0	
SFD910	Electrical origin low 16 bits		0	
SFD911	Electrical origin high 16 bits		0	
SFD912	Machine back to origin parameters	Bit0: promixity switch state 0: normal open 1: normal close	0	
SFD913	Near signal terminal	Bit0~bit7: the X terminal number, 0xFF is no terminal	0xFF	
SFD914	Z phase terminal	Bit0~bit7: the X terminal number, 0xFF is no terminal	0xFF	
SFD915	Limit terminal	Bit7~bit0: limit 1 X terminal number, 0xFF is no terminal Bit15~bit8: limit 2 X terminal number, 0xFF is no terminal	FFFF	
SFD916	Origin auxiliary signal terminal	Bit0~bit7: X terminal number, 0xFF is no terminal	0xFF	
SFD917	CLR signal output terminal	Bit0~bit7: Y terminal number, 0xFF is no terminal	0xFF	
SFD918	Back speed VH low 16 bits		0	
SFD919	Back speed VH high 16 bits		0	
SFD920	Back speed VL low 16 bits		0	
SFD921	Back speed VL high 16 bits		0	
SFD922	Creep speed low 16 bits		0	
SFD923	Creep speed high 16 bits		0	
SFD924	Mechanical origin low 16 bits		0	
SFD925	Mechanical origin high 16 bits		0	
SFD926	Z phase quantity		0	
SFD927	CLR signal delay time	Default value 20, unit: ms	20	
SFD936	G instruction which group of parameter to use	G instruction system parameter group choice	2	
…				
SFD950	Pulse default speed low 16 bits	Send pulse with default speed when speed is 0	1000	Group 1

SFD951	Pulse default speed high 16 bits		0	
SFD952	Pulse default speed acceleration time		80	
SFD953	Pulse default speed deceleration time		120	
SFD954	Tween acceleration/deceleration time		0	
SFD955	Reserved		0	
SFD956	Max speed limit low 16 bits		3392	
SFD957	Max speed limit high 16 bits		3	
SFD958	Start speed low 16 bits		0	
SFD959	Start speed high 16 bits		0	
SFD960	End speed low 16 bits		0	
SFD961	End speed high 16 bits		0	
SFD962	Curve acceleration time (ms)		0	
…				

Application 4

There is a servo drive (electronic gear ratio is 1:1) and servo motor (encoder 2500ppr), the reducing ratio of reducer is 1:2. The pitch of ball screw is 10mm, the ball screw drives the working table which can moving left and right. It needs to move the working table from left to right by 200mm, then reverse move 200mm, the speed is 20mm/s, acceleration time is 100ms, the deceleration time is 200ms. Pulse direction terminal is Y2. The mechanical structure is shown as below:





Calculation:

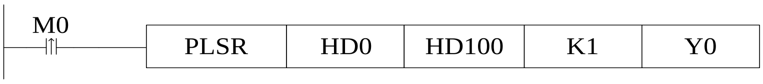
$$\text{pulse number (1 rotation)} = 20000 = 2500 * 4 * \frac{2}{1}$$

$$\text{movement (1 rotation)} = \text{pitch} = 10\text{mm}$$

$$20\text{mm/s} = \frac{20\text{mm}}{10\text{mm}} * 20000 = 40000\text{pulse/s}$$

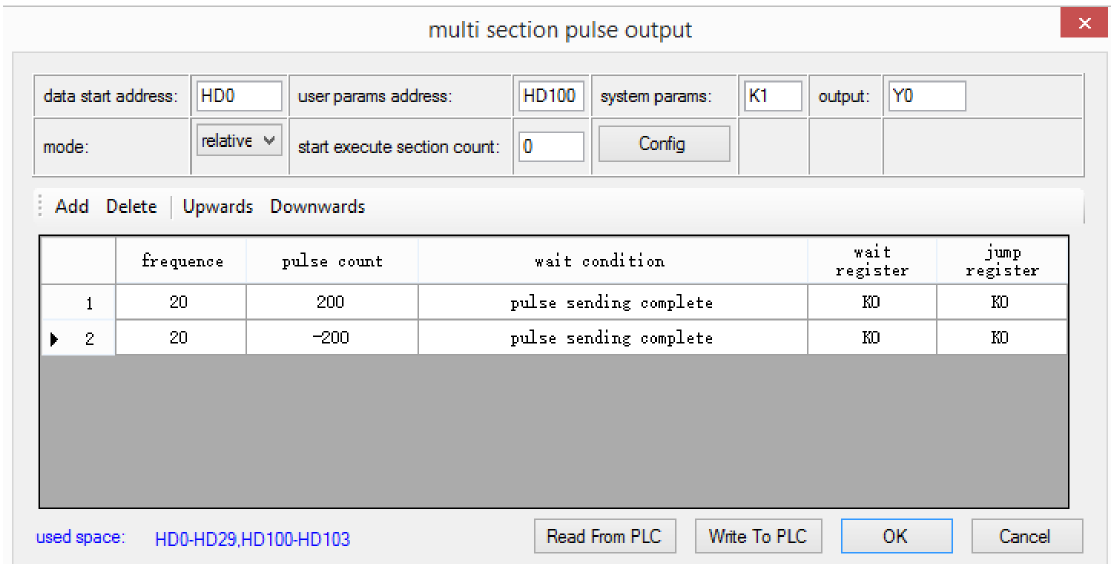
Max frequency is 40KHz/s, less than 200KHz/s, it can run normally.

Pulse instruction:

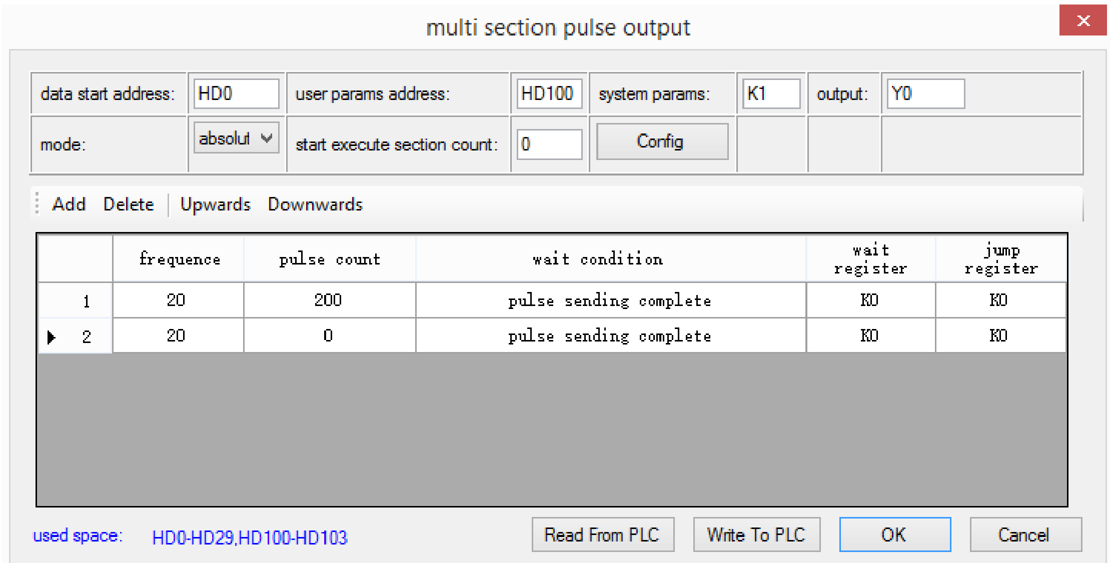


Method 1: set via XDPpro software

Pulse configuration table:

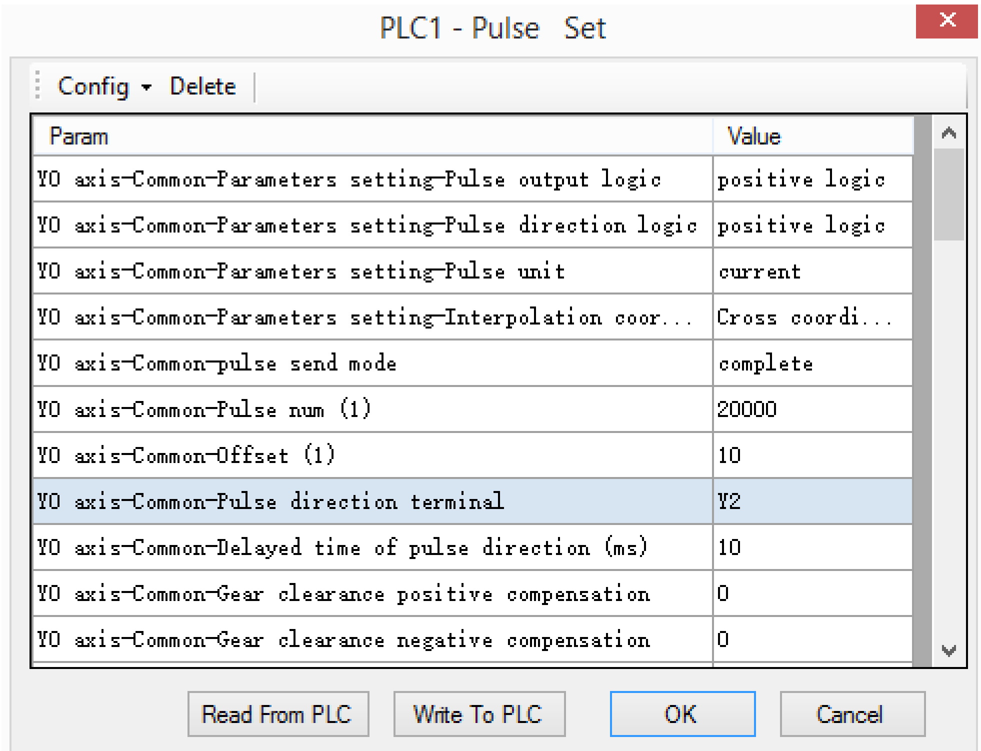


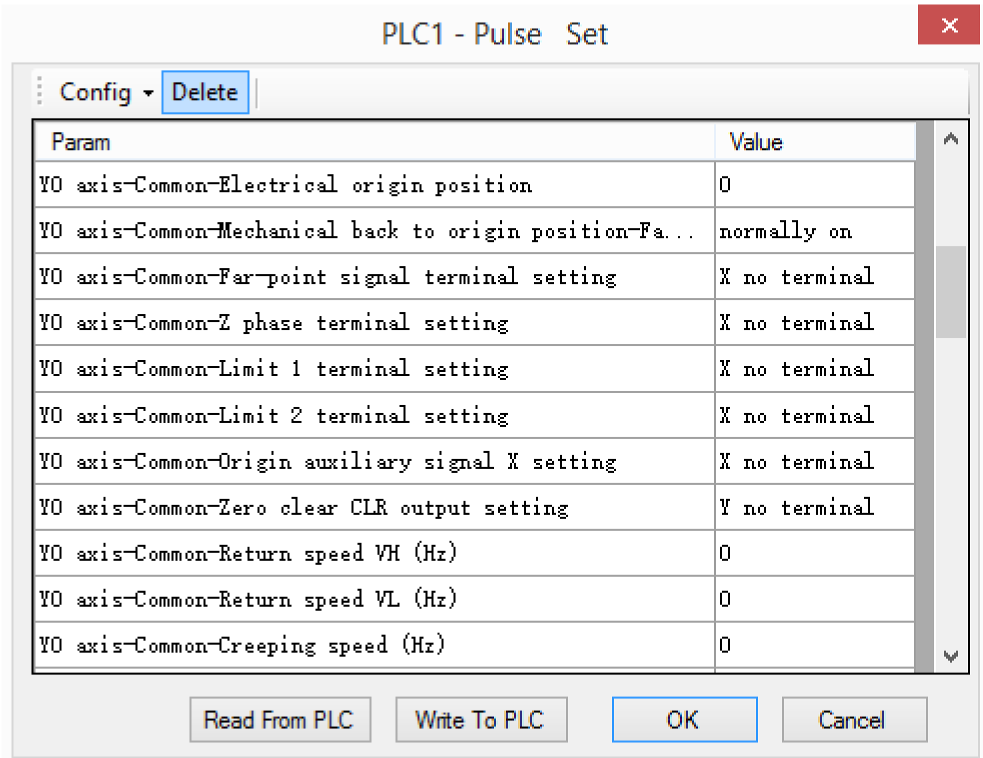
Relative mode



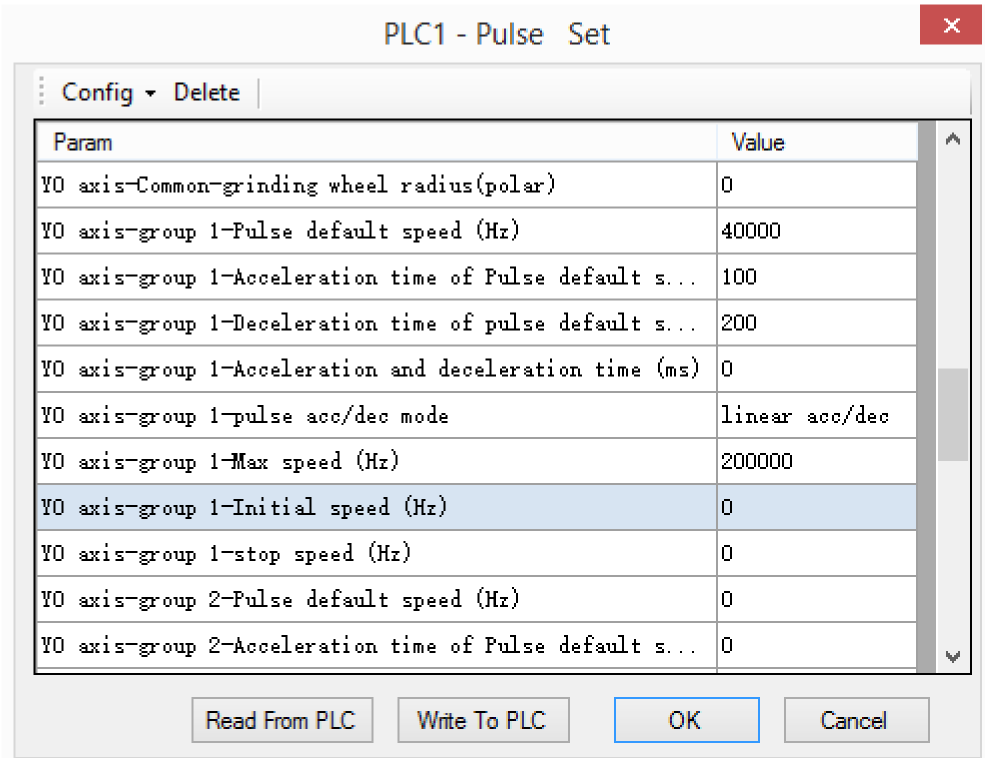
Absolute mode

System parameters

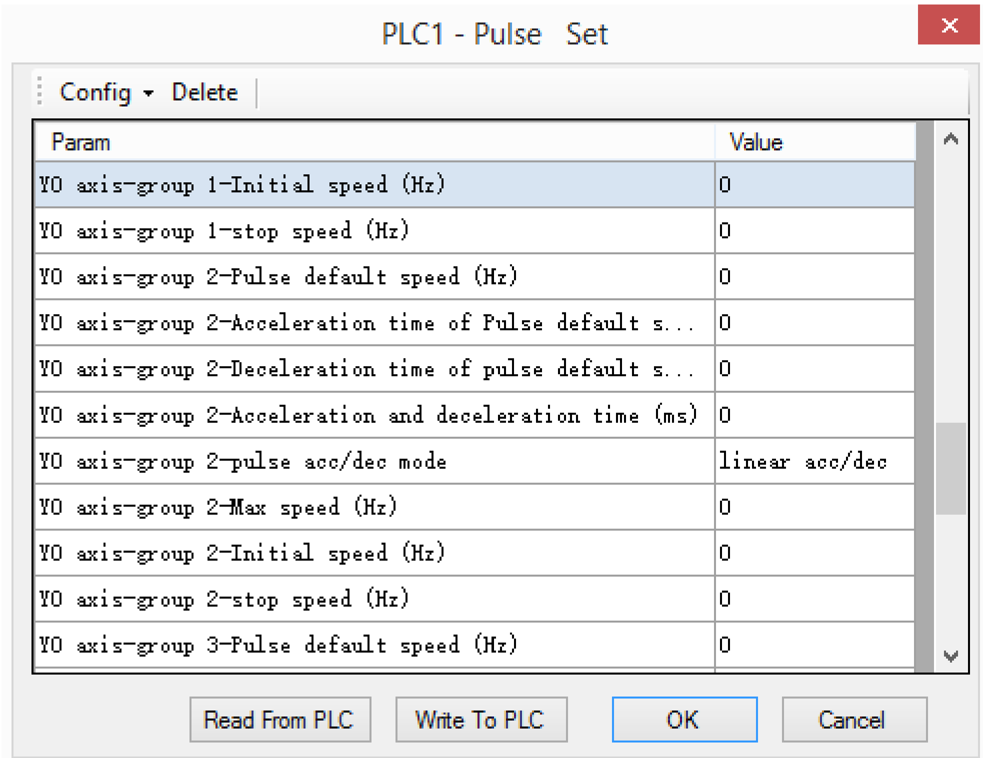




Param	Value
Y0 axis-Common-Electrical origin position	0
Y0 axis-Common-Mechanical back to origin position-Fa...	normally on
Y0 axis-Common-Far-point signal terminal setting	X no terminal
Y0 axis-Common-Z phase terminal setting	X no terminal
Y0 axis-Common-Limit 1 terminal setting	X no terminal
Y0 axis-Common-Limit 2 terminal setting	X no terminal
Y0 axis-Common-Origin auxiliary signal X setting	X no terminal
Y0 axis-Common-Zero clear CLR output setting	Y no terminal
Y0 axis-Common-Return speed VH (Hz)	0
Y0 axis-Common-Return speed VL (Hz)	0
Y0 axis-Common-Creeping speed (Hz)	0



Param	Value
Y0 axis-Common-grinding wheel radius(polar)	0
Y0 axis-group 1-Pulse default speed (Hz)	40000
Y0 axis-group 1-Acceleration time of Pulse default s...	100
Y0 axis-group 1-Deceleration time of pulse default s...	200
Y0 axis-group 1-Acceleration and deceleration time (ms)	0
Y0 axis-group 1-pulse acc/dec mode	linear acc/dec
Y0 axis-group 1-Max speed (Hz)	200000
Y0 axis-group 1-Initial speed (Hz)	0
Y0 axis-group 1-stop speed (Hz)	0
Y0 axis-group 2-Pulse default speed (Hz)	0
Y0 axis-group 2-Acceleration time of Pulse default s...	0



Method 2: set via pulse data reigsters

Pulse data register table (relative mode)

Address	Explanation	Value
HD0 (dword)	Pulse segment quantity (1~100)	2
HD2 (8 words)	Reserved (8 words)	0
HD10 (dword)	Pulse frequency (segment 1)	20
HD12 (dword)	Pulse quantity (segment 1)	200
HD14	bit15~bit8: wait condition (segment 1) 00: pulse sending end 01: wait time 02: wait signal 03: ACT time 04: EXT signal 05: EXT signal or pulse sending end bit7~bit0: wait condition register type 00: constant 01: D 02: X 03: M	0
HD15 (dword)	Constant/register(waiting condition) (segment 1)	0
HD17	bit7~bit0: jump register type 00: constant 01: D	0
HD+18 (dword)	Constant/register (jump register) (segment 1)	0
HD+20 (dword)	Pulse frequency (segment 2)	20

HD+22 (dword)	Pulse quantity (segment 2)	-200
HD+24	Wait condition, wait condition register type (segment 2)	0
HD+25 (dword)	Constant or register (waiting condition) (segment 2)	0
HD+27	Jump type, jump register type (waiting condition) (segment 2)	0
HD+28 (dword)	Constant or register (jump register) (segment 2)	0

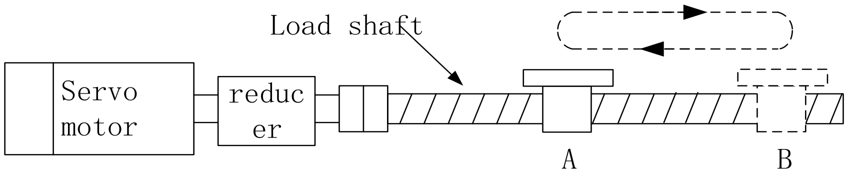
System parameter address:

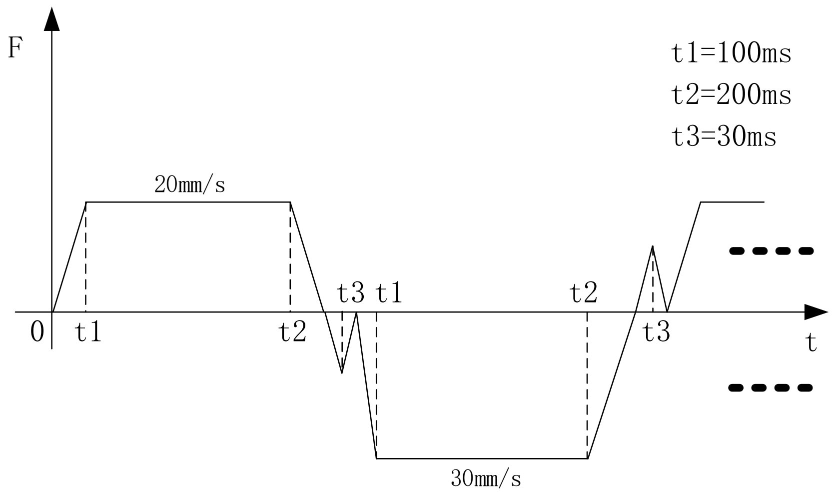
SFD900	Pulse parameters	Bit 0: pulse output logic 0: positive logic(default setting) 1: negative logic, Bit 1：pulse direction logic 0: positive logic(default setting) 1: negative logic Bit 8: pulse unit 0: pulse quantity(default setting) 1: equivalent value	256	Public parameter
SFD901	Reserved		0	
SFD902	Pulse quantity/ 1 rotation low 16 bits		20000	
SFD903	Pulse quantity/ 1 rotation high 16 bits		0	
SFD904	Movement amount/1 rotation low 16 bits		10	
SFD905	Movement amount/1 rotation high 16 bits		0	
SFD906	Pulse direction terminal	The number of Y, 0xFF is no terminal	2	
SFD907	Direction delay time	Default value is 20, unit is ms	10	
SFD908	Gear clearance positive compensation		0	
SFD909	Gear clearance negative compensation		0	
SFD910	Electrical origin low 16 bits		0	
SFD911	Electrical origin high 16 bits		0	
SFD912	Machine back to origin parameters	Bit0: promixity switch state 0: normal open 1: normal close	0	
SFD913	Near signal terminal	Bit0~bit7: the X terminal number, 0xFF is no terminal	0xFF	
SFD914	Z phase terminal	Bit0~bit7: the X terminal number, 0xFF is no terminal	0xFF	
SFD915	Limit terminal	Bit7~bit0: limit 1 X terminal number, 0xFF is no terminal Bit15~bit8: limit 2 X terminal number, 0xFF is no terminal	FFFF	

SFD916	Origin auxiliary signal terminal	Bit0~bit7: X terminal number, 0xFF is no terminal	0xFF	
SFD917	CLR signal output terminal	Bit0~bit7: Y terminal number, 0xFF is no terminal	0xFF	
SFD918	Back speed VH low 16 bits		0	
SFD919	Back speed VH high 16 bits		0	
SFD920	Back speed VL low 16 bits		0	
SFD921	Back speed VL high 16 bits		0	
SFD922	Creep speed low 16 bits		0	
SFD923	Creep speed high 16 bits		0	
SFD924	Mechanical origin low 16 bits		0	
SFD925	Mechanical origin high 16 bits		0	
SFD926	Z phase quantity		0	
SFD927	CLR signal delay time	Default value 20, unit: ms	20	
SFD936	G instruction which group of parameter to use	G instruction system parameter group choice	2	
…				
SFD950	Pulse default speed low 16 bits	Send pulse with default speed when speed is 0	40000	Group 1 parameters
SFD951	Pulse default speed high 16 bits			
SFD952	Pulse default speed acceleration time		100	
SFD953	Pulse default speed deceleration time		200	
SFD954	Tween acceleration/deceleration time		0	
SFD955	Reserved		0	
SFD956	Max speed limit low 16 bits		3392	
SFD957	Max speed limit high 16 bits		3	
SFD958	Start speed low 16 bits		0	
SFD959	Start speed high 16 bits		0	
SFD960	End speed low 16 bits		0	
SFD961	End speed high 16 bits		0	
SFD962	Curve acceleration time (ms)		0	
…				

例 5

There is a servo drive (electronic gear ratio is 1:1) and servo motor (encoder is 2500 ppr). The reducing ratio of reducer is 1:2. The pitch of ball screw is 5mm, the ball screw drives the working table which can move left and right. The working table will move back and forth between A and B. the distance between A and B is 200mm. the speed from A to B is 200mm/s, the speed from B to A is 30mm/s. the acceleration time is 100ms, the deceleration time is 200ms. The A→B→A route mechanical clearance error is 3mm, B→A→B route mechanical clearance error is 2mm. the pulse direction terminal is Y2.





Calculation:

$$\text{pulse number (1 rotation)} = 20000 = 2500 * 4 * \frac{2}{1}$$

$$\text{movement (1 rotation)} = \text{pitch} = 5\text{mm}$$

$$20\text{mm/s} = \frac{20\text{mm}}{5\text{mm}} * 20000 = 80000\text{pulse/s}$$

$$30\text{mm/s} = \frac{30\text{mm}}{5\text{mm}} * 20000 = 120000\text{pulse/s}$$

The acceleration time and deceleration time is same, but the max frequency is different, so the acceleration and deceleration slope is different.

Forward direction acceleration slope: it changes 80000Hz every 100ms

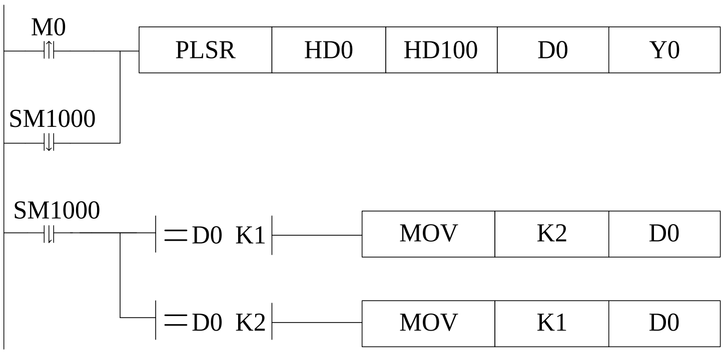
Forward direction deceleration slope: it changes 80000Hz every 200ms

Reverse direction acceleration slope: it changes 120000Hz every 100ms

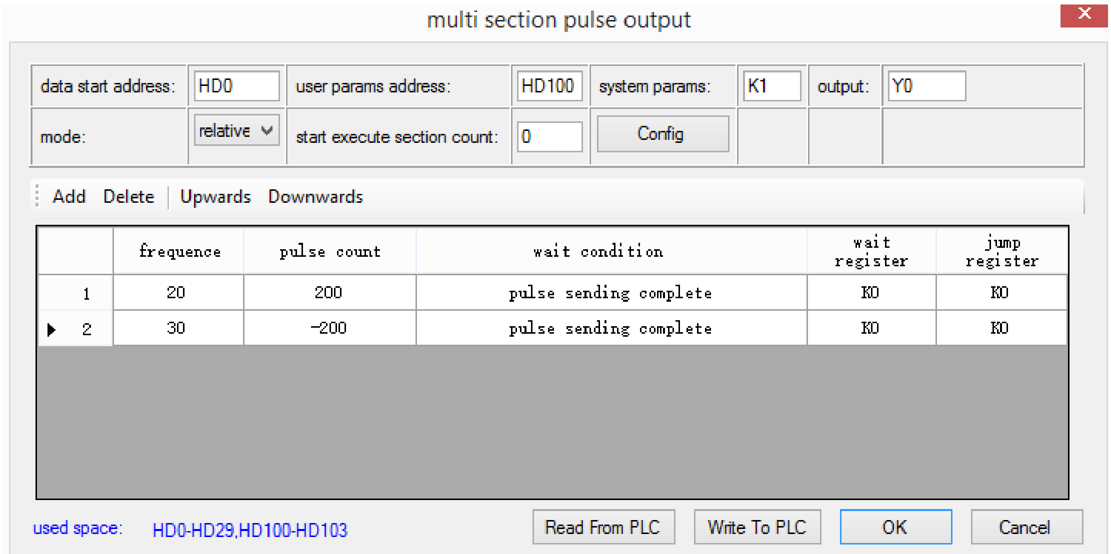
Reverse direction deceleration slope: it changes 120000Hz every 200ms

It needs two groups of parameters as there are two groups of acceleration and deceleration slope.The max frequency is 40KHz/s and 120KHz/s, they are less than 200KHz/s. So they can run normally.

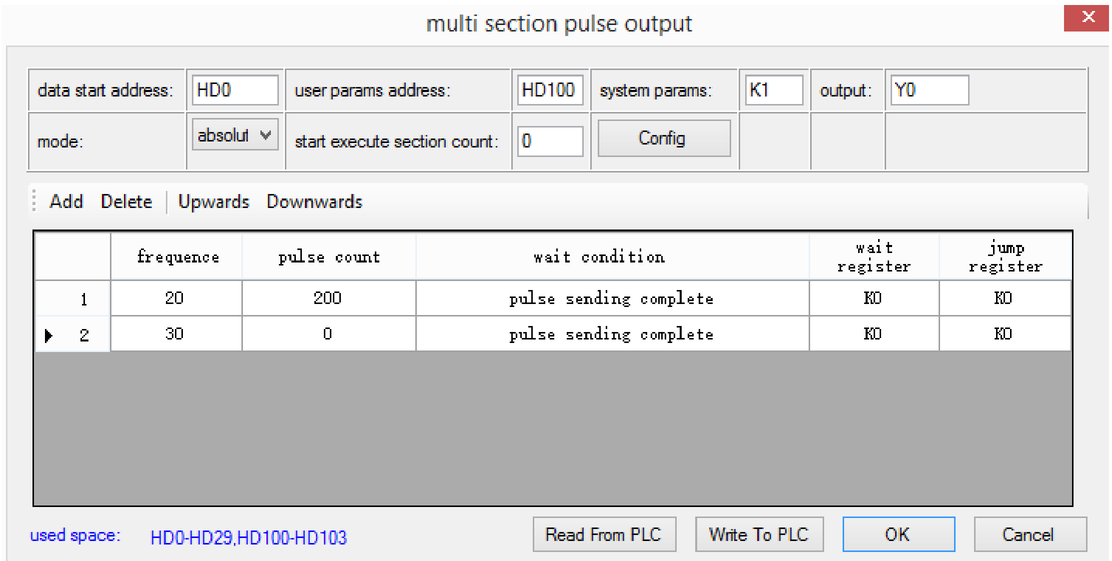
Pulse instruction:



Method 1: set via XDPpro software

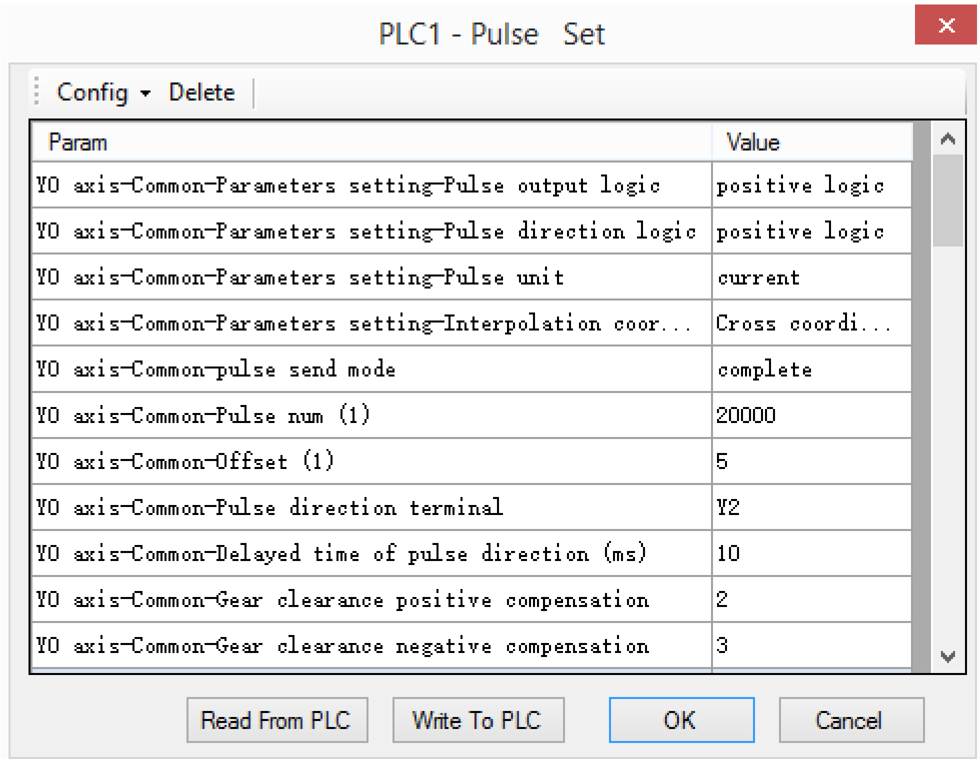


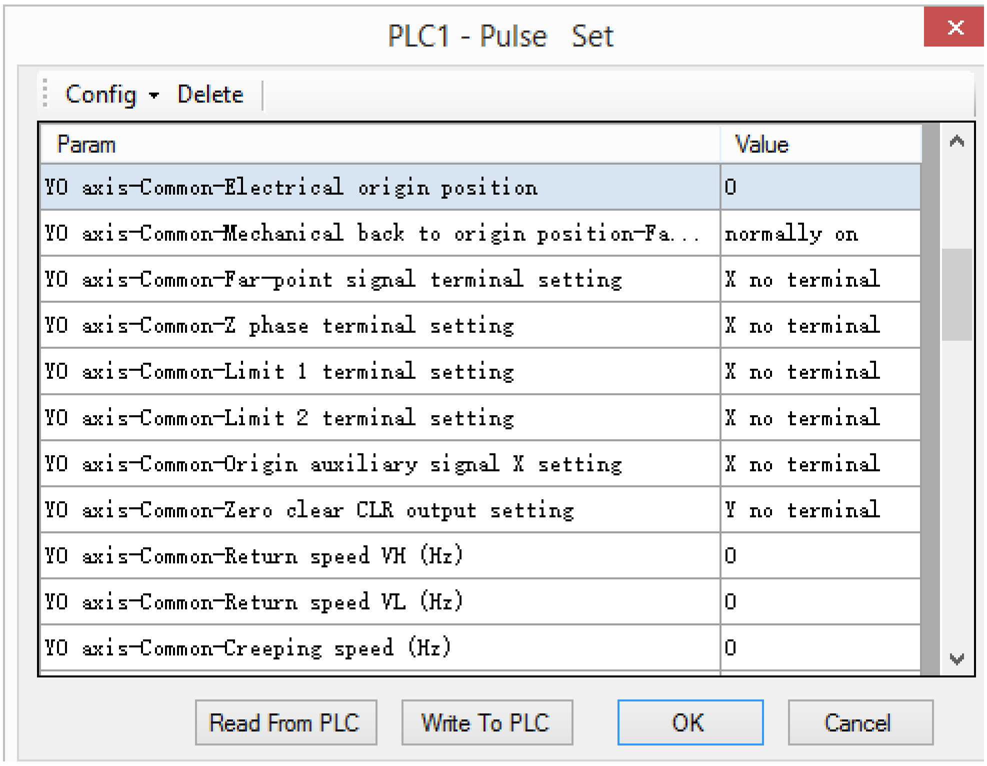
Relative mode

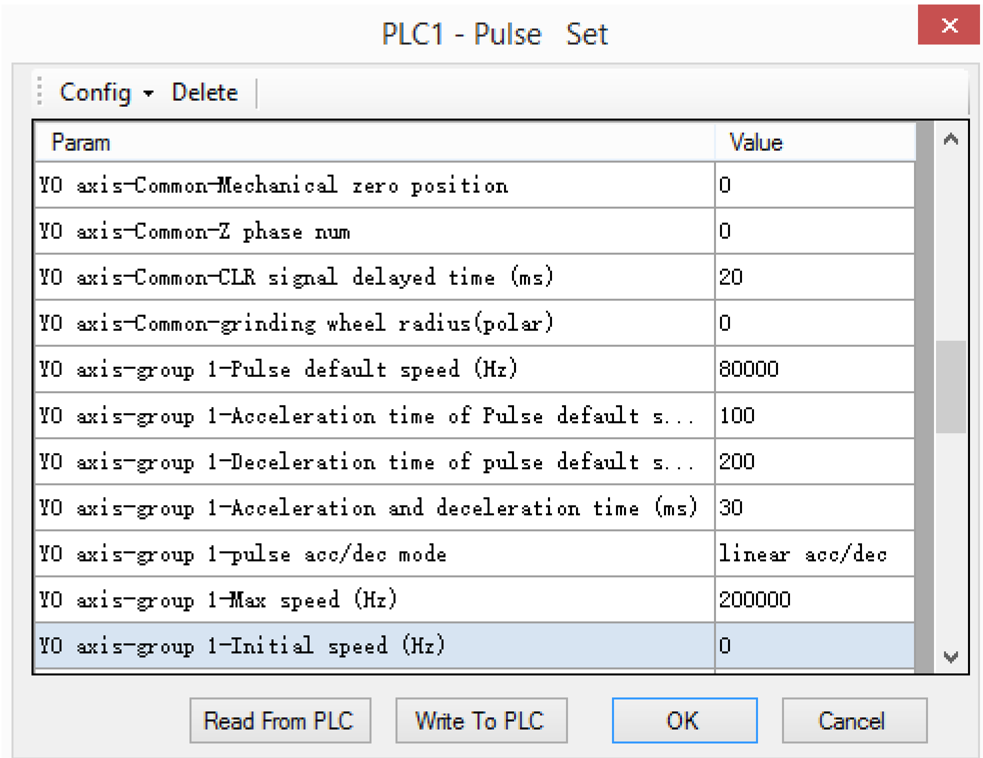


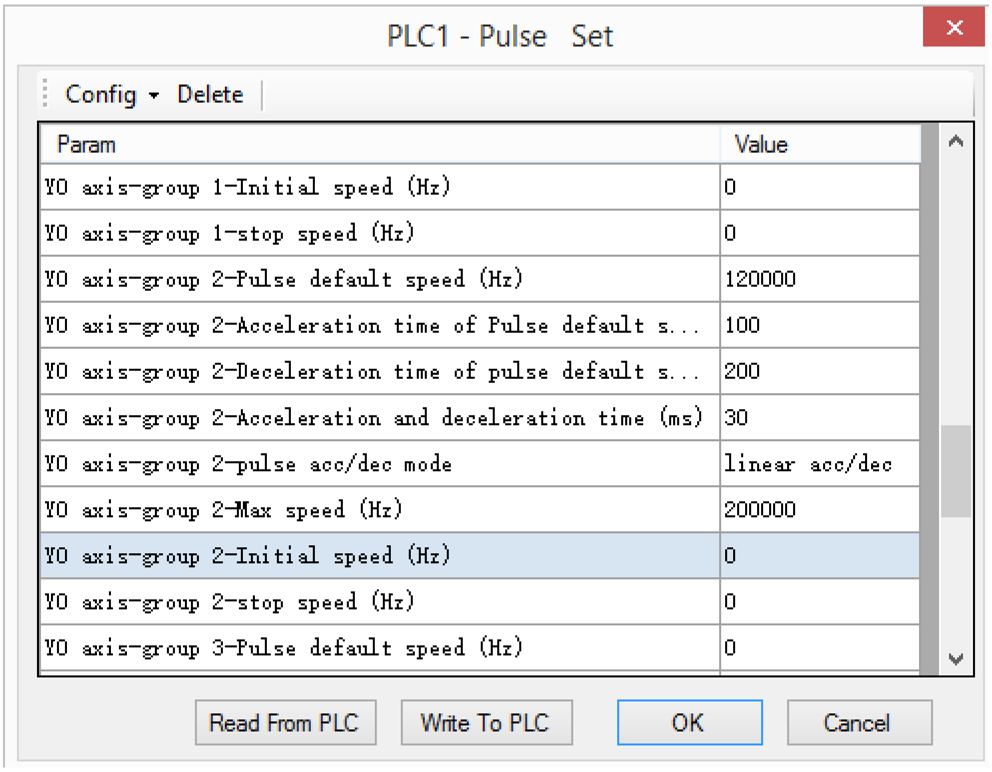
Absolute mode

System parameter (relative mode)









Method 2: set via pulse register

Pulse data address table (relative mode)

Address	Explanation	Value
HD0 (dword)	Pulse segment quantity (1~100)	2
HD2 (8 words)	Reserved (8 words)	0

HD10 (dword)	Pulse frequency (segment 1)	20
HD12 (dword)	Pulse quantity (segment 1)	200
HD14	bit15~bit8: wait condition (segment 1) 00: pulse sending end 01: wait time 02: wait signal 03: ACT time 04: EXT signal 05: EXT signal or pulse sending end bit7~bit0: wait condition register type 00: constant 01: D 02: X 03: M	0
HD15 (dword)	Constant/register(waiting condition) (segment 1)	0
HD17	bit7~bit0: jump register type 00: constant 01: D	0
HD+18 (dword)	Constant/register (jump register) (segment 1)	0
HD+20 (dword)	Pulse frequency (segment 2)	20
HD+22 (dword)	Pulse quantity (segment 2)	-200
HD+24	Wait condition, wait condition register type (segment 2)	0
HD+25 (dword)	Constant or register (waiting condition) (segment 2)	0
HD+27	Jump type, jump register type (waiting condition) (segment 2)	0
HD+28 (dword)	Constant or register (jump register) (segment 2)	0

System parameter address:

SFD900	Pulse parameters	Bit 0: pulse output logic 0: positive logic(default setting) 1: negative logic, Bit 1：pulse direction logic 0: positive logic(default setting) 1: negative logic Bit 8: pulse unit 0: pulse quantity(default setting) 1: equivalent value	256	Public parameter
SFD901	Reserved		0	
SFD902	Pulse quantity/ 1 rotation low 16 bits		20000	
SFD903	Pulse quantity/ 1 rotation high 16 bits		0	
SFD904	Movement amount/1 rotation low 16 bits		10	
SFD905	Movement amount/1 rotation high 16 bits		0	

SFD906	Pulse direction terminal	The number of Y, 0xFF is no terminal	2	
SFD907	Direction delay time	Default value is 20, unit is ms	10	
SFD908	Gear clearance positive compensation		0	
SFD909	Gear clearance negative compensation		0	
SFD910	Electrical origin low 16 bits		0	
SFD911	Electrical origin high 16 bits		0	
SFD912	Machine back to origin parameters	Bit0: promixity switch state 0: normal open 1: normal close	0	
SFD913	Near signal terminal	Bit0~bit7: the X terminal number, 0xFF is no terminal	0xFF	
SFD914	Z phase terminal	Bit0~bit7: the X terminal number, 0xFF is no terminal	0xFF	
SFD915	Limit terminal	Bit7~bit0: limit 1 X terminal number, 0xFF is no terminal Bit15~bit8: limit 2 X terminal number, 0xFF is no terminal	FFFF	
SFD916	Origin auxiliary signal terminal	Bit0~bit7: X terminal number, 0xFF is no terminal	0xFF	
SFD917	CLR signal output terminal	Bit0~bit7: Y terminal number, 0xFF is no terminal	0xFF	
SFD918	Back speed VH low 16 bits		0	
SFD919	Back speed VH high 16 bits		0	
SFD920	Back speed VL low 16 bits		0	
SFD921	Back speed VL high 16 bits		0	
SFD922	Creep speed low 16 bits		0	
SFD923	Creep speed high 16 bits		0	
SFD924	Mechanical origin low 16 bits		0	
SFD925	Mechanical origin high 16 bits		0	
SFD926	Z phase quantity		0	
SFD927	CLR signal delay time	Default value 20, unit: ms	20	
SFD936	G instruction which group of parameter to use	G instruction system parameter group choice	2	
…				
SFD950	Pulse default speed low 16 bits	Send pulse with default speed when speed is 0	40000	Group 1 parameters
SFD951	Pulse default speed high 16 bits			

SFD952	Pulse default speed acceleration time		100	
SFD953	Pulse default speed deceleration time		200	
SFD954	Tween acceleration/deceleration time		0	
SFD955	Reserved		0	
SFD956	Max speed limit low 16 bits		3392	
SFD957	Max speed limit high 16 bits		3	
SFD958	Start speed low 16 bits		0	
SFD959	Start speed high 16 bits		0	
SFD960	End speed low 16 bits		0	
SFD961	End speed high 16 bits		0	
SFD962	Curve acceleration time (ms)		0	
…				

6-2-2． Variable frequency pulse output [PLSF]

Summarization

Variable frequency pulse output instruction

Variable frequency pulse output [PLSF]			
16 bits instruction		32 bits instruction	PLSF
Execution condition	Rising/ falling pulse edge	Suitable model	XD3, XDM
Hardware requirements	-	Software requirements	-

Operand

Operand	Function	Type
S0	Pulse frequency address	
S1	System parameters (1~4)	
D	Pulse output terminal	

Suitable soft component

Word

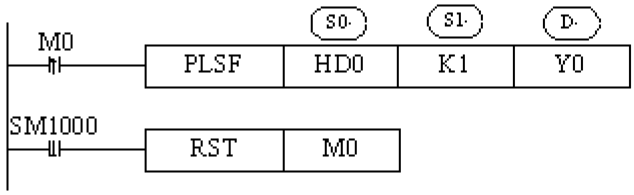
Operand	System								Constant	Module	
	D*	FD	TD*	CD*	DX	DY	DM*	DS*	K/H	ID	QD
S0	●	●	●	●	●	●	●	●			
S1	●	●									

Bit

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

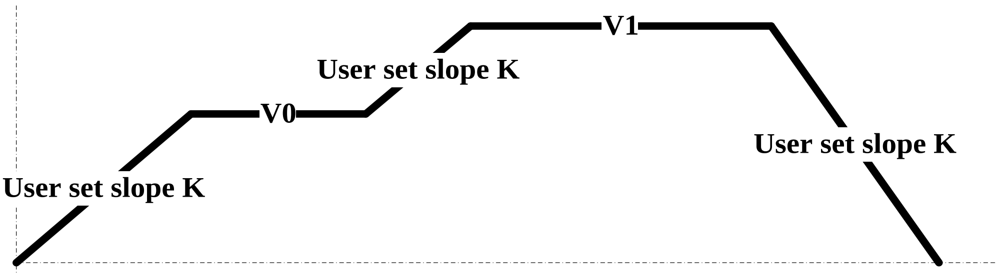
Notes: D includes D, HD; TD includes TD, HTD; CD includes CD, HCD, HSCD, HSD; DM includes DM, DHM; DS includes DS, DHS. M includes M, HM, SM; S includes S and HS; T includes T and HT; C includes C and HC.

《Instruction》



Frequency range: 1Hz ~200KHz
Pulse can output from Y0 or Y1
The frequency output from Y0 or Y1 is changing as the S0 setting frequency
Accumulate the pulse amounts in register HSD0 (dword), accumulate the equivalent in HSD2 (dword).
Dynamic adjust the pulse as the slope when frequency jumped (refer to PLSR)
The system parameters are the same to PLSR instruction

Output mode



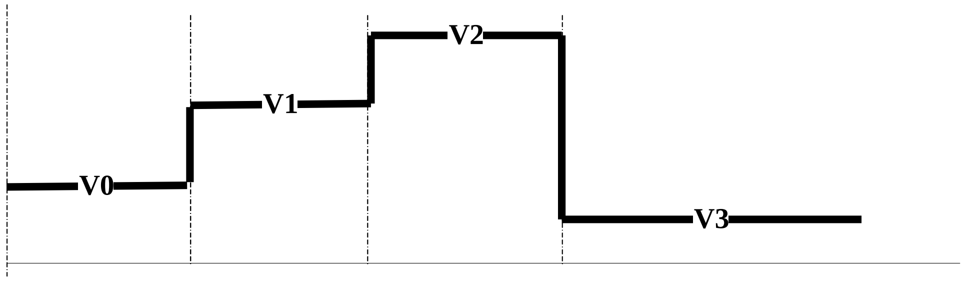
The pulse output direction is set through system parameters (refer to the PLSR pulse direction setting)
The pulse direction is decided by the positive or negative of pulse frequency. Positive pulse frequency means the pulse direction is positive, negative pulse frequency means the pulse direction is negative.
When S0 is 0, PLSF stop pulse output

The instruction will adjust the pulse output as the frequency and slope set by user. If user set the frequency to 0, the current segment frequency will fall to 0 then output as default pulse speed.

Analysis of different modes

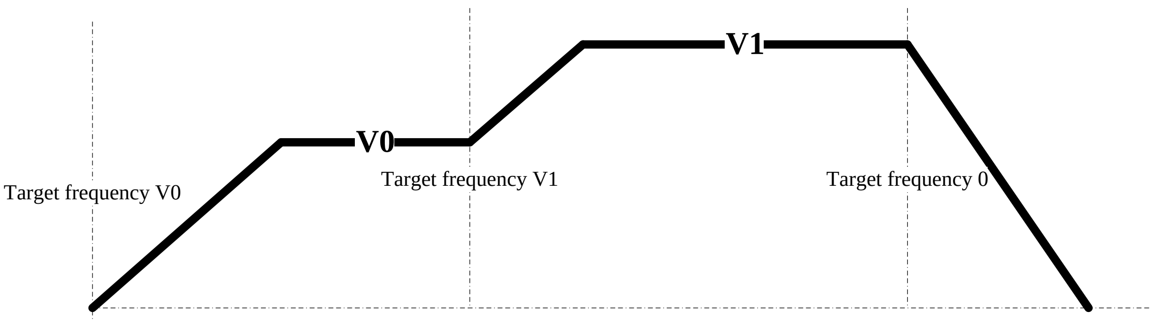
(A) Default pulse speed, acceleration/deceleration time is 0

The pulse frequency will jump as the user set frequency

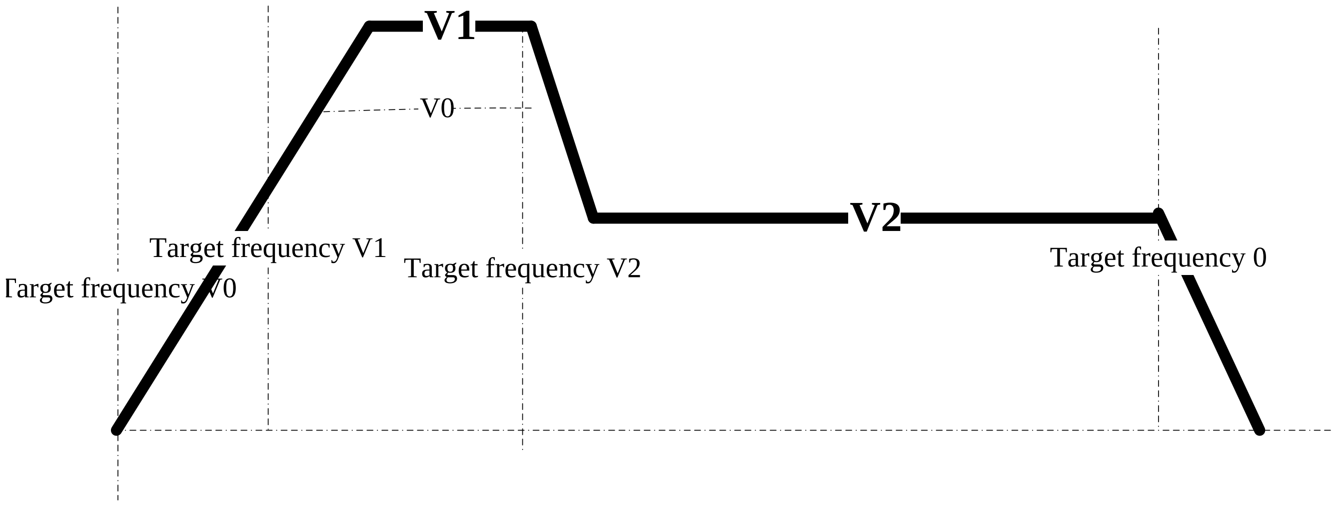


(B) Default pulse speed, acceleration/deceleration time is not 0

(1) The pulse is in stable period when the user set new frequency. The pulse will change to target frequency with set slope.

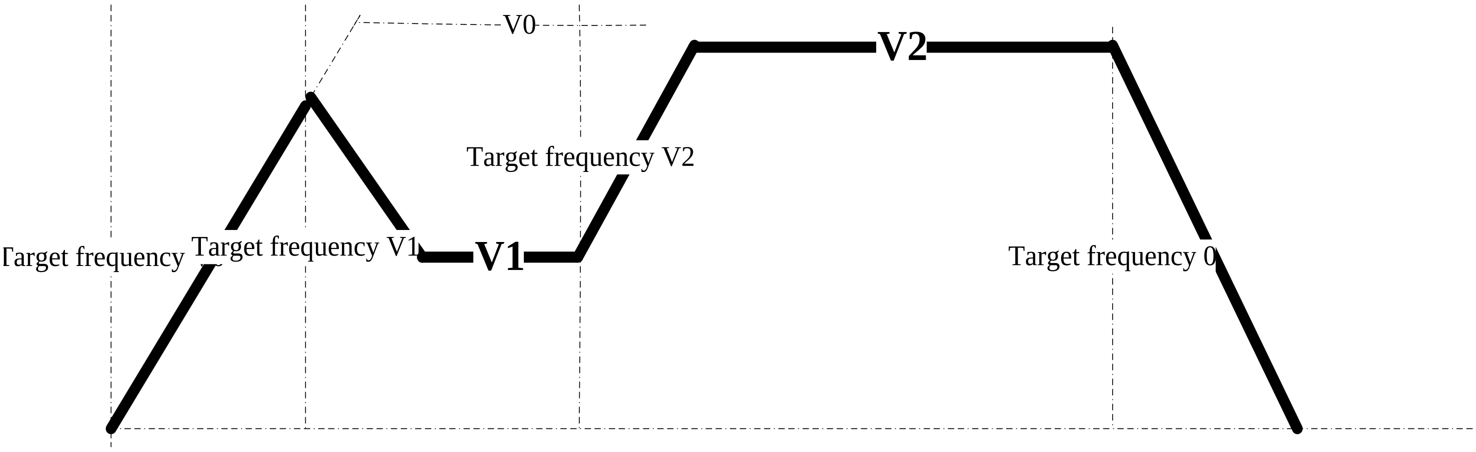


(2) The pulse is not in stable period when the user set new frequency. The pulse will change to target frequency with set slope (the current set frequency will be considered as target if it is



larger than last set frequency. If user set new frequency V1 before the pulse reaches set frequency V0 (V1>V0), the pulse will change to V1 as the set slope.

(3) The pulse is not in stable period when the user set new frequency. The pulse will change to target frequency with set slope (current set frequency < last set frequency, current set frequency < current frequency). The user set new frequency V1 before pulse reaches set frequency V0 (V1<V0, V1 < current frequency), the pulse will change to V1 with set slope.

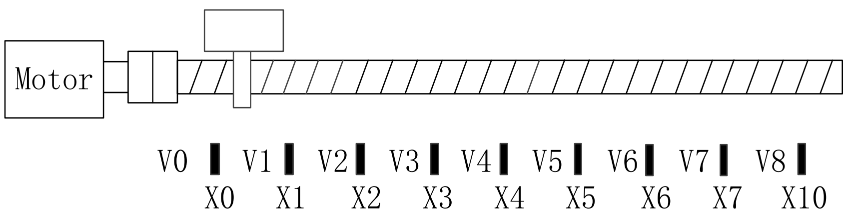


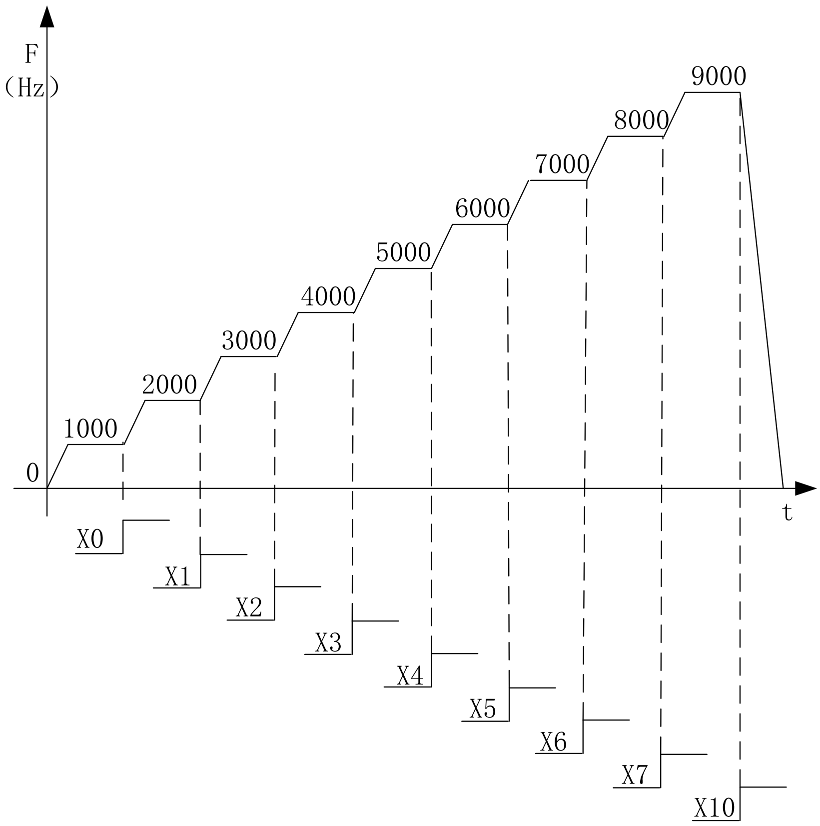
Example 1

The working table needs to move from the left end to position X10. There are proximity in the position from X0 to X10.

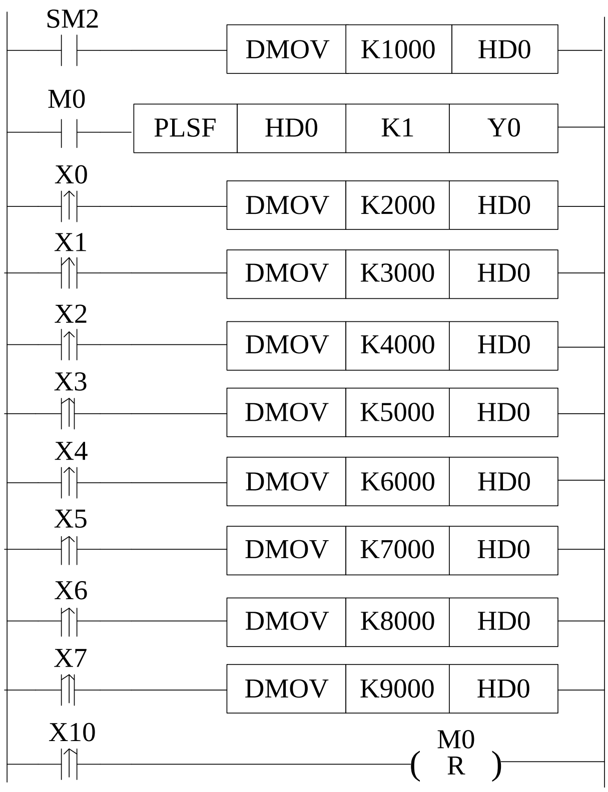
The speed of each segment please see the following table. Acceleration and deceleration slope is 100ms every 1000Hz. Pulse direction terminal is Y2.

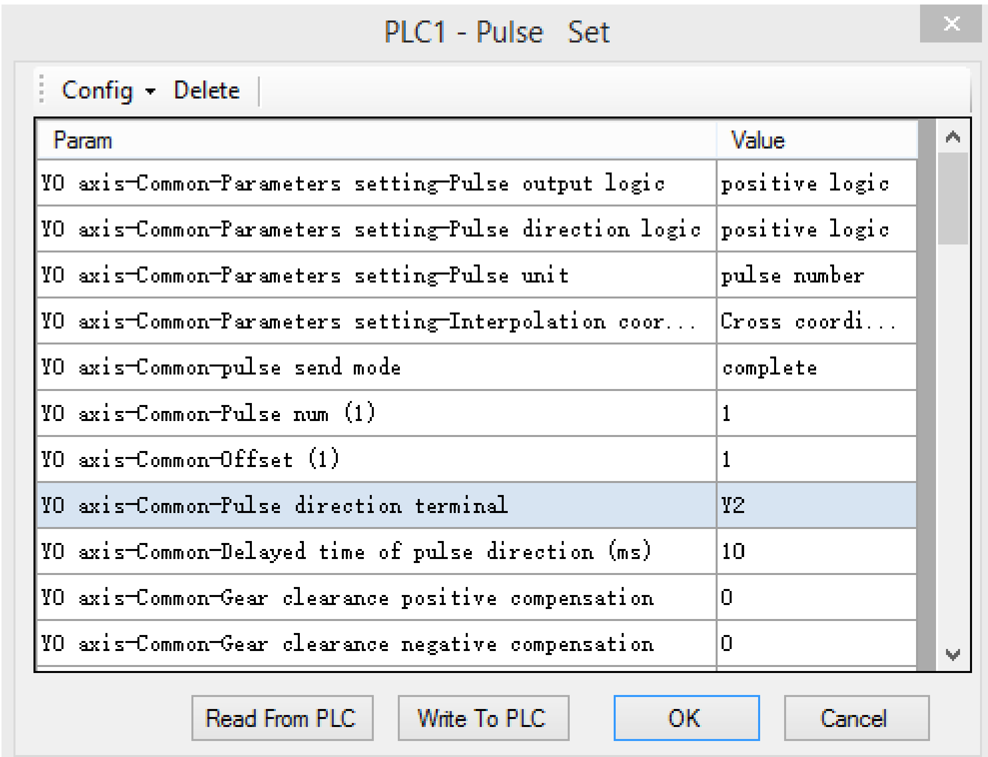
Segment	Speed no.	Speed
Left end → X0	V0	1000Hz
X0 → X1	V1	2000Hz
X1 → X2	V2	3000Hz
X2 → X3	V3	4000Hz
X3 → X4	V4	5000Hz
X4 → X5	V5	6000Hz
X5 → X6	V6	7000Hz
X6 → X7	V7	8000Hz
X7 → X10	V8	9000Hz

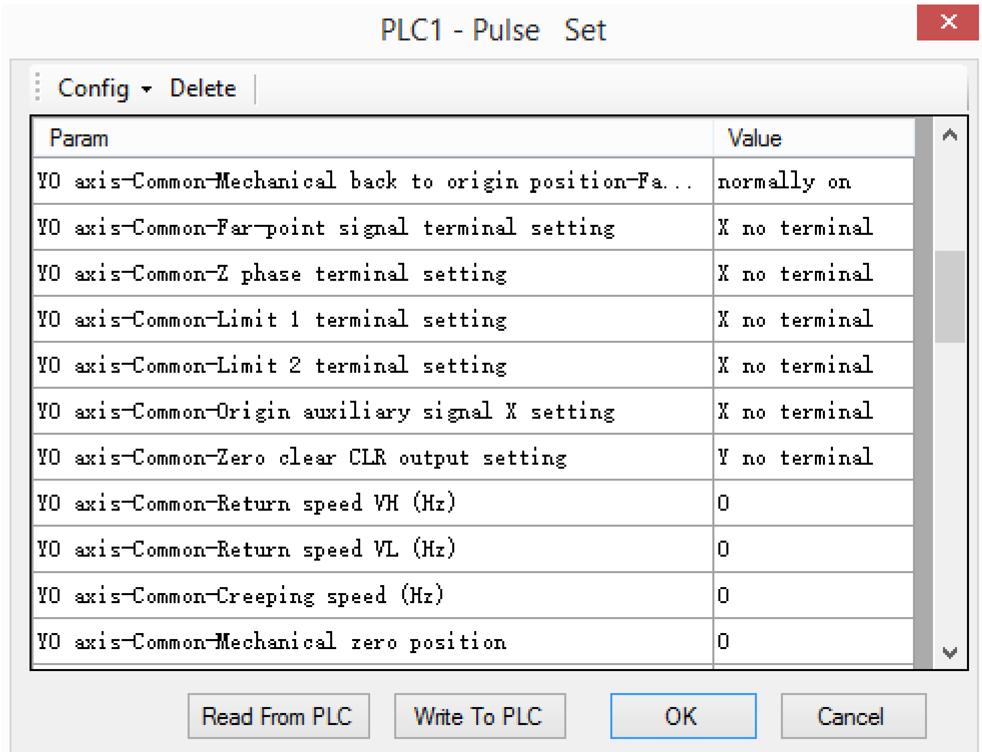


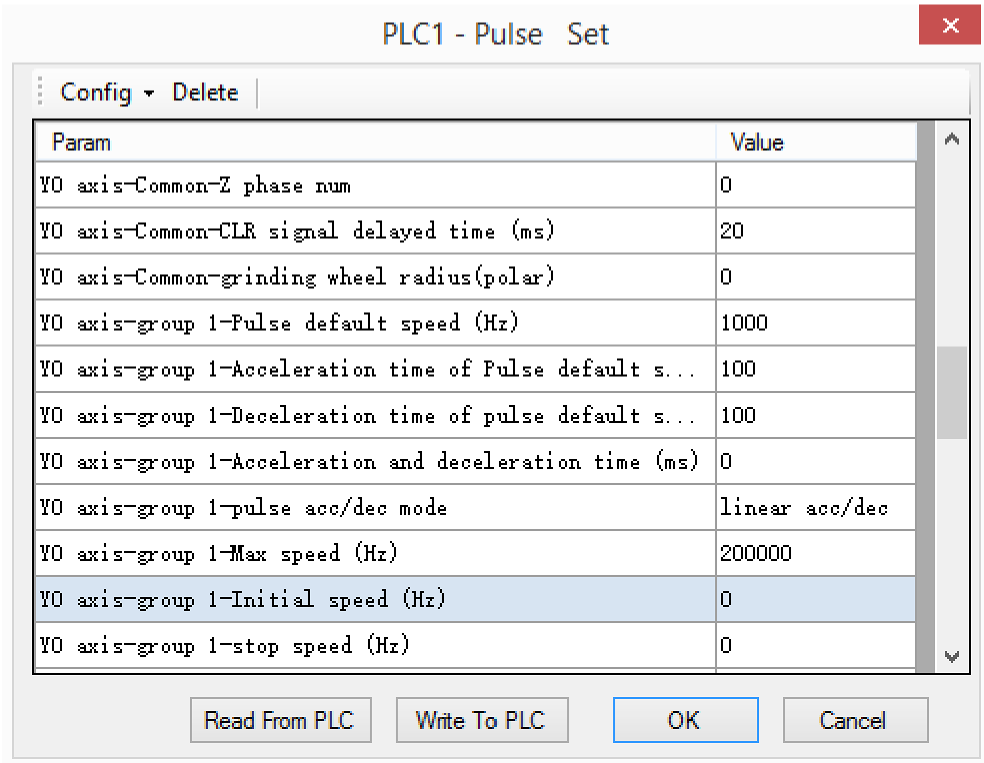


Method 1: set the parameters via XDPpro software




PLC1 - Pulse Set
Config
Delete
Param
Value
Y0 axis-Common-Parameters setting-Pulse output logic
positive logic
Y0 axis-Common-Parameters setting-Pulse direction logic
positive logic
Y0 axis-Common-Parameters setting-Pulse unit
pulse number
Y0 axis-Common-Parameters setting-Interpolation coor...
Cross coordi...
Y0 axis-Common-pulse send mode
complete
Y0 axis-Common-Pulse num (1)
1
Y0 axis-Common-Offset (1)
1
Y0 axis-Common-Pulse direction terminal
Y2
Y0 axis-Common-Delayed time of pulse direction (ms)
10
Y0 axis-Common-Gear clearance positive compensation
0
Y0 axis-Common-Gear clearance negative compensation
0
Read From PLC
Write To PLC
OK
Cancel


PLC1 - Pulse Set
Config
Delete
Param
Value
Y0 axis-Common-Mechanical back to origin position-Fa...
normally on
Y0 axis-Common-Far-point signal terminal setting
X no terminal
Y0 axis-Common-Z phase terminal setting
X no terminal
Y0 axis-Common-Limit 1 terminal setting
X no terminal
Y0 axis-Common-Limit 2 terminal setting
X no terminal
Y0 axis-Common-Origin auxiliary signal X setting
X no terminal
Y0 axis-Common-Zero clear CLR output setting
Y no terminal
Y0 axis-Common-Return speed VH (Hz)
0
Y0 axis-Common-Return speed VL (Hz)
0
Y0 axis-Common-Creeping speed (Hz)
0
Y0 axis-Common-Mechanical zero position
0
Read From PLC
Write To PLC
OK
Cancel



Method 2: set the parameters via data registers

System parameters

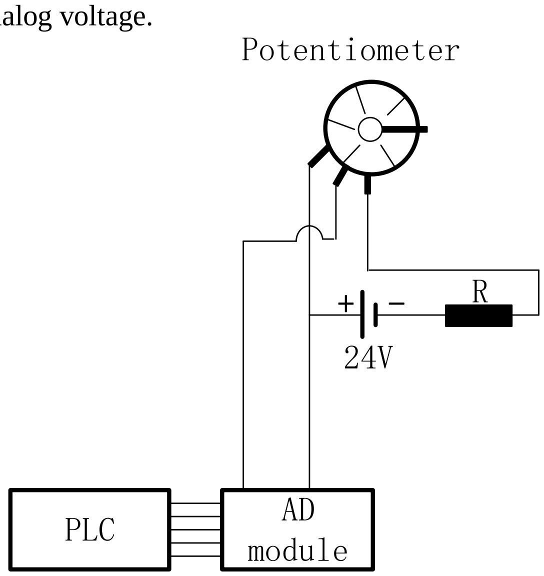
SFD900	Pulse parameters	Bit 0: pulse output logic 0: positive logic(default setting) 1: negative logic, Bit 1: pulse direction logic 0: positive logic(default setting) 1: negative logic Bit 8: pulse unit 0: pulse quantity(default setting) 1: equivalent value	0	Public parameter
SFD901	Reserved		0	
SFD902	Pulse quantity/ 1 rotation low 16 bits		0	
SFD903	Pulse quantity/ 1 rotation high 16 bits		0	
SFD904	Movement amount/1 rotation low 16 bits		0	
SFD905	Movement amount/1 rotation high 16 bits		0	
SFD906	Pulse direction terminal	The number of Y, 0xFF is no terminal	2	
SFD907	Direction delay time	Default value is 20, unit is ms	10	
SFD908	Gear clearance positive compensation		0	
SFD909	Gear clearance negative compensation		0	

SFD910	Electrical origin low 16 bits		0	
SFD911	Electrical origin high 16 bits		0	
SFD912	Machine back to origin parameters	Bit0: promixity switch state 0: normal open 1: normal close	0	
SFD913	Near signal terminal	Bit0~bit7: the X terminal number, 0xFF is no terminal	0xFF	
SFD914	Z phase terminal	Bit0~bit7: the X terminal number, 0xFF is no terminal	0xFF	
SFD915	Limit terminal	Bit7~bit0: limit 1 X terminal number, 0xFF is no terminal Bit15~bit8: limit 2 X terminal number, 0xFF is no terminal	FFFF	
SFD916	Origin auxiliary signal terminal	Bit0~bit7: X terminal number, 0xFF is no terminal	0xFF	
SFD917	CLR signal output terminal	Bit0~bit7: Y terminal number, 0xFF is no terminal	0xFF	
SFD918	Back speed VH low 16 bits		0	
SFD919	Back speed VH high 16 bits		0	
SFD920	Back speed VL low 16 bits		0	
SFD921	Back speed VL high 16 bits		0	
SFD922	Creep speed low 16 bits		0	
SFD923	Creep speed high 16 bits		0	
SFD924	Mechanical origin low 16 bits		0	
SFD925	Mechanical origin high 16 bits		0	
SFD926	Z phase quantity		0	
SFD927	CLR signal delay time	Default value 20, unit: ms	20	
SFD936	G instruction which group of parameter to use	G instruction system parameter group choice	2	
…				
SFD950	Pulse default speed low 16 bits	Send pulse with default speed when speed is 0	1000	Group 1 parameters
SFD951	Pulse default speed high 16 bits			
SFD952	Pulse default speed acceleration time		100	
SFD953	Pulse default speed deceleration time		100	
SFD954	Tween acceleration/deceleration time		0	
SFD955	Reserved		0	

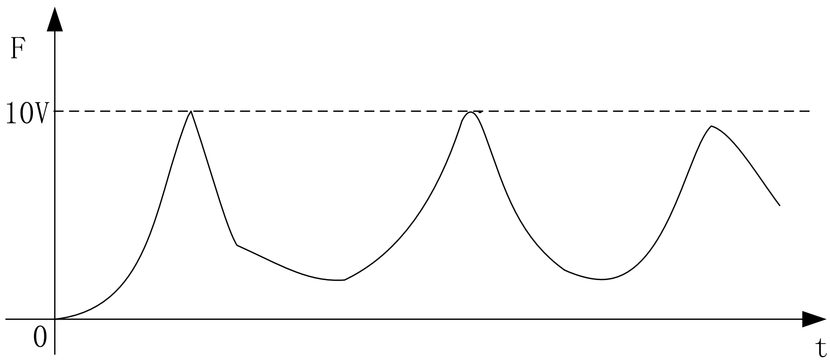
SFD956	Max speed limit low 16 bits		200000	
SFD957	Max speed limit high 16 bits			
SFD958	Start speed low 16 bits		0	
SFD959	Start speed high 16 bits		0	
SFD960	End speed low 16 bits		0	
SFD961	End speed high 16 bits		0	
SFD962	Curve acceleration time (ms)		0	
…				

Example 2

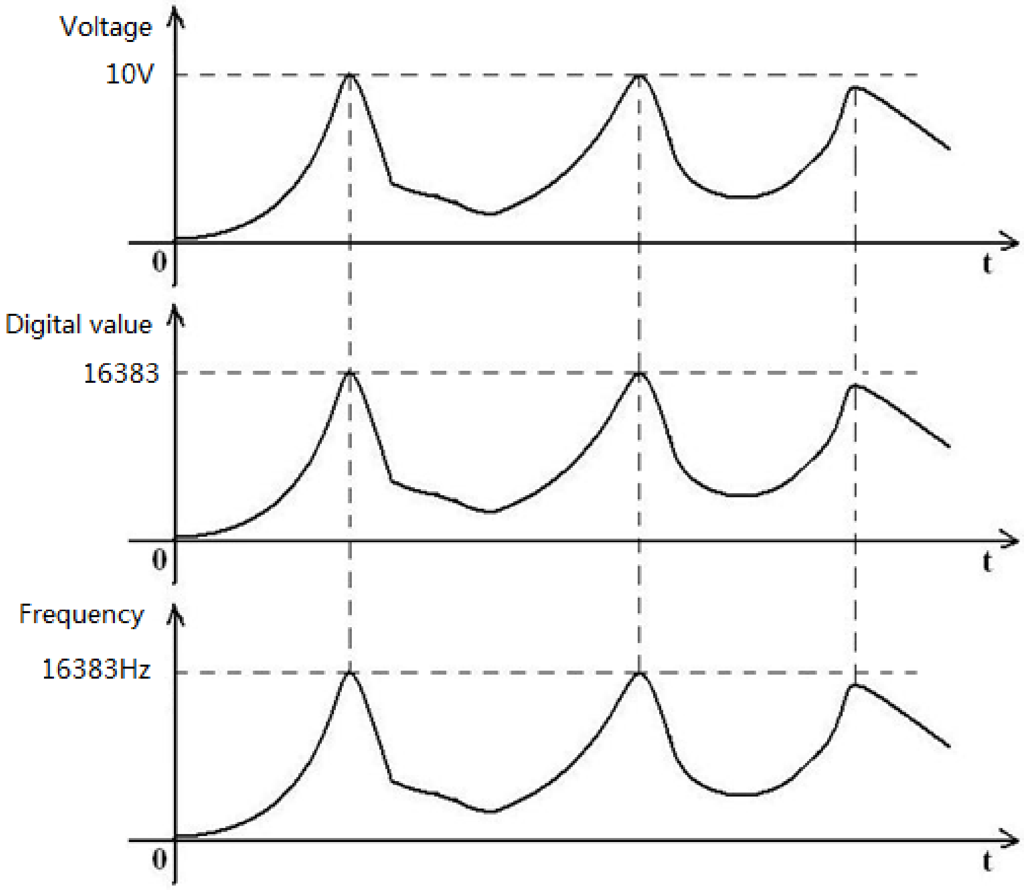
The AD module will collect the 0~10V voltage signal, then transform the analog value to digital value 0~16383 and send to pulse frequency register of PLSF. The PLSF frequency will change following the analog voltage.

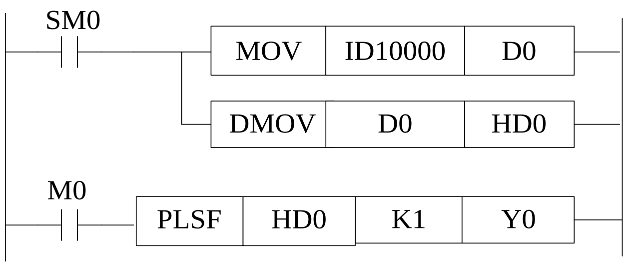


The voltage signal output from the potentiometer:

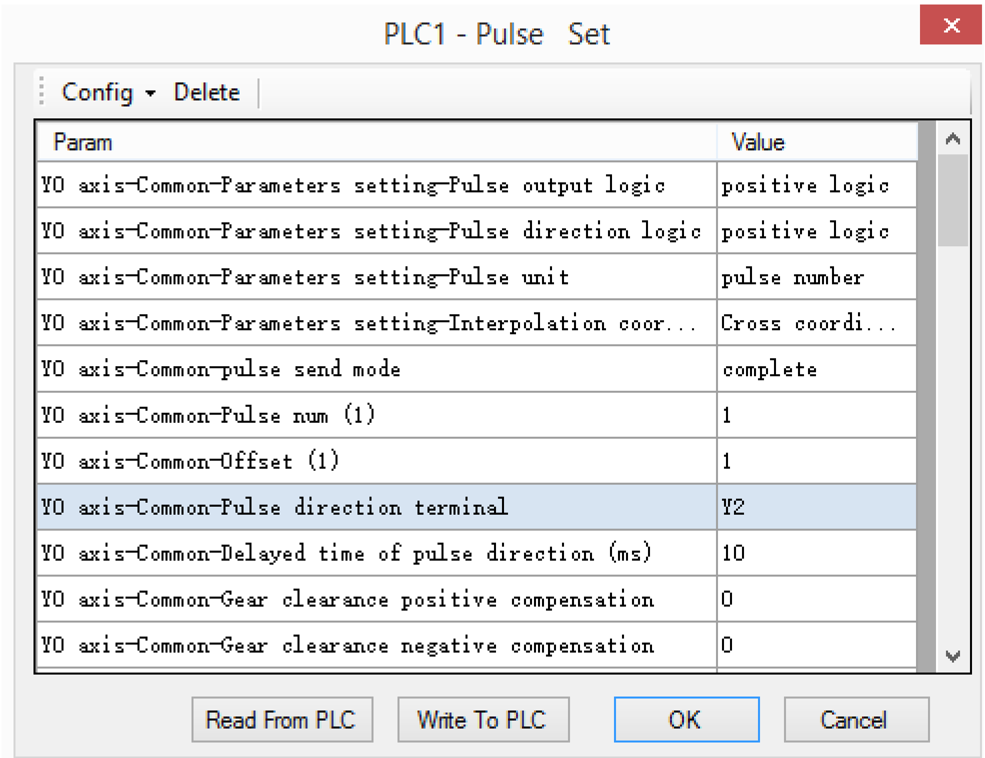


As the 0~10V signal will be transformed to digital value 0~16383, which means the output frequency is 0~16383Hz. As the response problem, PLSF acceleration and deceleration time is 0.

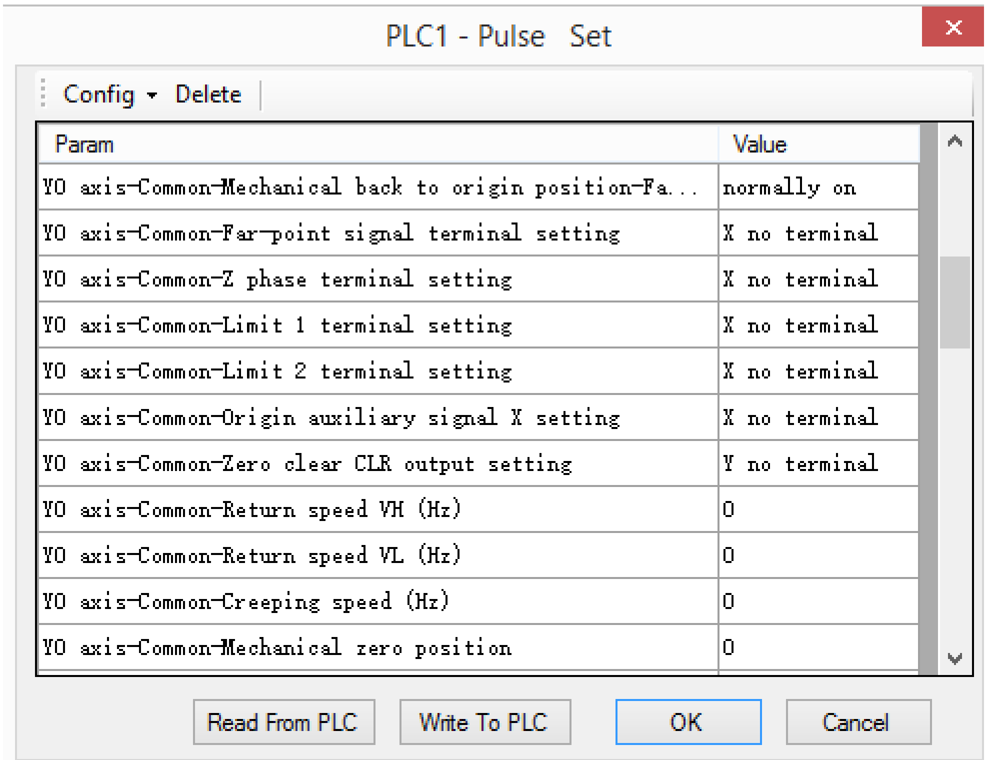




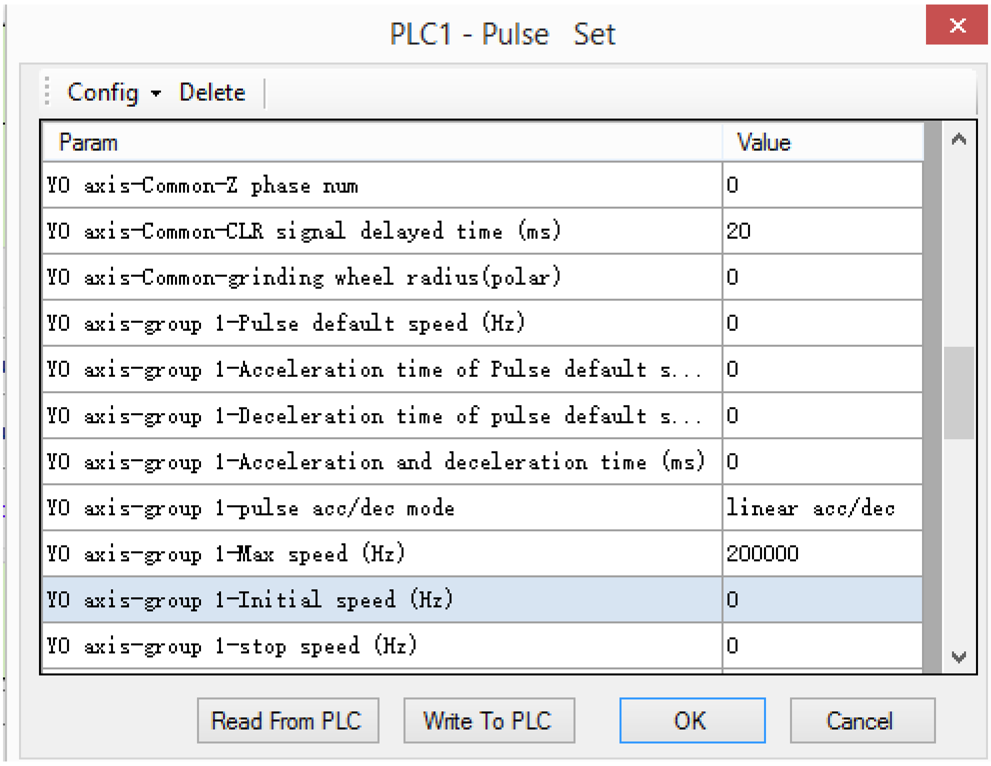
System parameters (relative mode)



Param	Value
Y0 axis-Common-Parameters setting-Pulse output logic	positive logic
Y0 axis-Common-Parameters setting-Pulse direction logic	positive logic
Y0 axis-Common-Parameters setting-Pulse unit	pulse number
Y0 axis-Common-Parameters setting-Interpolation coor...	Cross coordi...
Y0 axis-Common-pulse send mode	complete
Y0 axis-Common-Pulse num (1)	1
Y0 axis-Common-Offset (1)	1
Y0 axis-Common-Pulse direction terminal	Y2
Y0 axis-Common-Delayed time of pulse direction (ms)	10
Y0 axis-Common-Gear clearance positive compensation	0
Y0 axis-Common-Gear clearance negative compensation	0



Param	Value
Y0 axis-Common-Mechanical back to origin position-Fa...	normally on
Y0 axis-Common-Far-point signal terminal setting	X no terminal
Y0 axis-Common-Z phase terminal setting	X no terminal
Y0 axis-Common-Limit 1 terminal setting	X no terminal
Y0 axis-Common-Limit 2 terminal setting	X no terminal
Y0 axis-Common-Origin auxiliary signal X setting	X no terminal
Y0 axis-Common-Zero clear CLR output setting	Y no terminal
Y0 axis-Common-Return speed VH (Hz)	0
Y0 axis-Common-Return speed VL (Hz)	0
Y0 axis-Common-Creeping speed (Hz)	0
Y0 axis-Common-Mechanical zero position	0



6-2-3. Mechanical zero return [ZRN]

Instruction summary

Pulse instruction of mechanical zero return

Mechanical zero return[ZRN]			
16 bits instruction		32 bits instruction	ZRN
Execution condition	Rising/falling edge trigger	Suitable type	XD3, XDM
Hardware requirement	-	Software requirement	-

Operands

operands	Function	type
S	Specify system parameters ID number	
D	Specify pulse output terminal number	

suitable soft component

Word

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

Bit

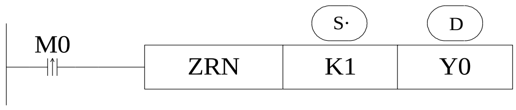
operands	System						
	X	Y	M*	S*	T*	C*	Dn.m
D		●					

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 and HS; T includes T and HT; C includes C and HC.

Function and Action

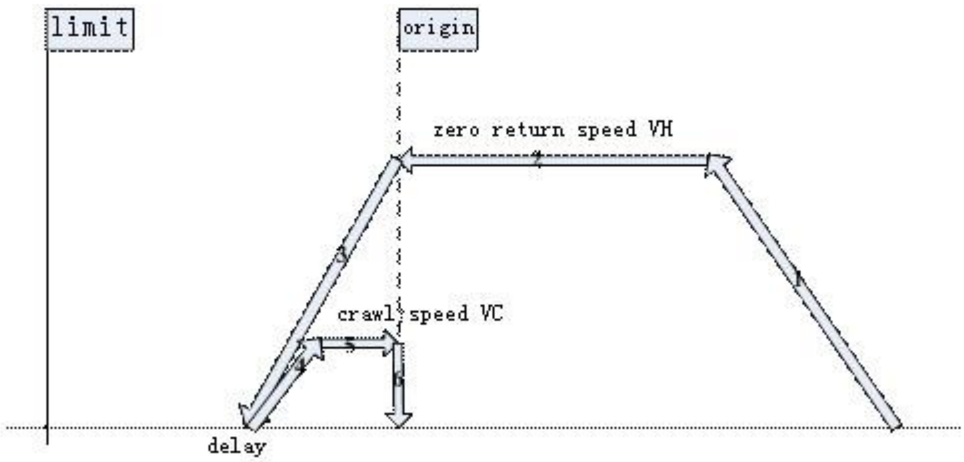
《Instruction form》

Its system parameters block is the same with PLSR, please refer to PLSR system parameters block.



(A) No Z phase signal, no limit signal:

By mechanical zero signal and limit, no origin auxiliary signal; or come near the mechanical zero signal when mechanical zero return begins:



Action description:

(1) Accelerate to speed VH with slope K in origin return direction.

(2) Decelerate to VL with slope K when encountering origin auxiliary signal.

(3) Decelerate to 0 with slope K when touching the origin.

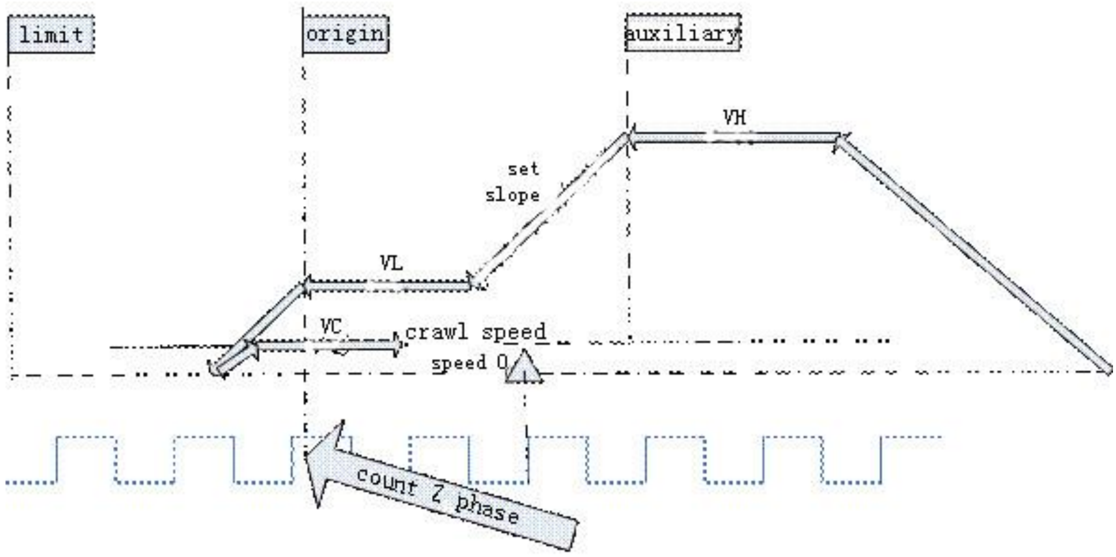
(4) Delay(direction delay in SFD), accelerate to crawl speed with slope K, and stop action once leaving the origin; Output clear signal immediately and delay if ‘zero return CLR signal’ is set.（CLR signal delay in SFD can use ‘zero return CLR signal’ output to clear Error Counter of servo motor）, then copy mechanical origin to the current position, zero return is finished.

Note: (some special occasions)

Decelerate to 0 with slope K immediately if it reaches the origin during the process that ZRN start to accelerate; delay（direction delay in SFD）, then accelerate to VH with slope K, output clear signal immediately and delay（CLR signal delay in SFD can use ‘zero return

CLR signal' output to clear Error Counter of servo motor）if 'zero return CLR signal' is set, then copy mechanical origin to the current position, zero return is finished.
Decelerate with slope K when encountering origin signal, as mechanical origin structure is short, it may haven't slowed down to 0 when passes the origin, it will still decelerate to 0;
After delay（Direction delay in SFD）, accelerate to VH with slope K backwards, the moment it leaves origin (near point sensing signal 1→0), output clear signal immediately and delay if 'zero return CLR signal' is set.（CLR signal delay in SFD can use 'zero return CLR signal' output to clear Error Counter of servo motor）, then copy mechanical origin to current position, zero return is finished.

(B) set Z phase signal, no limit signal:

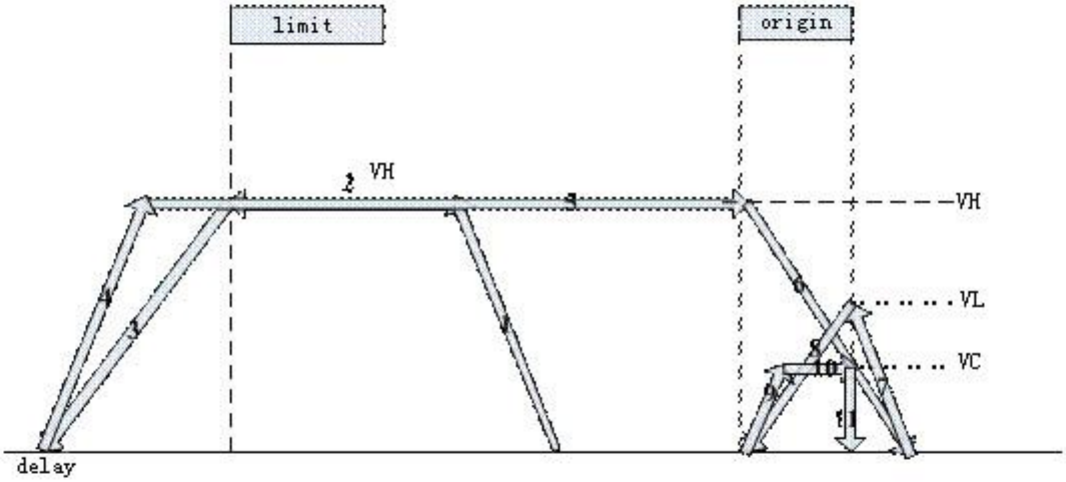


Action description:
(1) Accelerate to speed VH with slope K and in origin return direction.
(2) Decelerate to VL with slope K when encountering origin auxiliary signal.
(3) Decelerate to 0 with slope K when encountering origin signal.
(4) Delay（direction delay in SFD）, and accelerate to crawl speed backwards with acceleration time slope. The moment it leaves origin signal, Z phase input signal starts to count.
(5) Stop action when Z phase signal counter reaches the set value. Output clear signal and delay if 'zero return CLR signal' is set. (CLR signal delay in SFD can use 'zero return CLR signal' output to clear Error Counter of servo motor), then copy mechanical origin to the current position, zero return is finished.
Note: (some special occasions)
Decelerate to VL immediately with set slope if touching mechanical origin during the process that ZRN start to accelerate, and come near to origin signal at speed VL, the following action description is the same with above.
Decelerate to 0 with deceleration slope when touching origin signal during the process that it decelerates with set slope from origin auxiliary signal.
Decelerate with deceleration slope when encountering origin signal, as mechanical origin structure is short, it may haven't slowed down to 0 when passes the origin, it will still

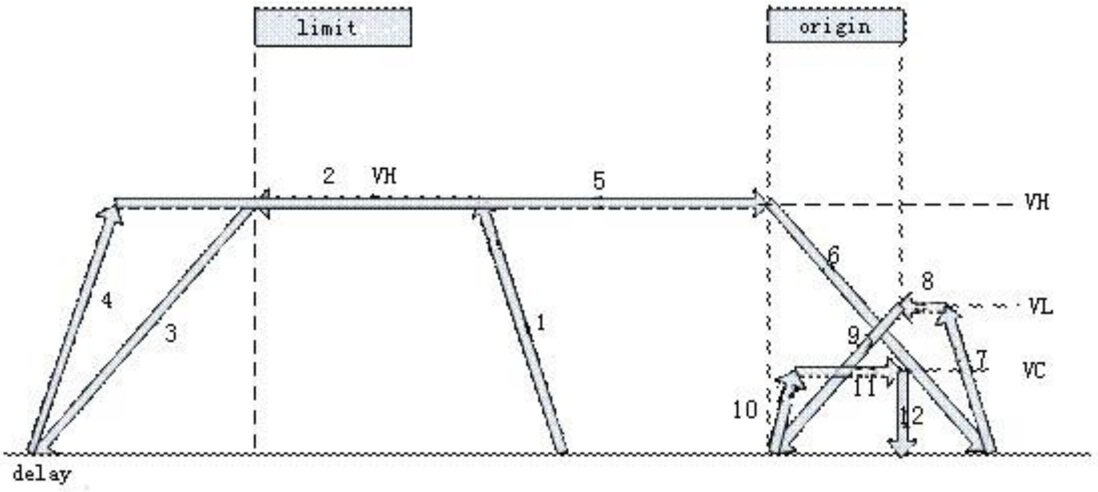
decelerate to 0; After delay (Direction delay in SFD), accelerate to VH with acceleration slope backwards, the moment it leaves origin (near point sensing signal 1→0), output clear signal immediately and delay if ‘zero return CLR signal’ is set. (CLR signal delay in SFD can use ‘zero return CLR signal’ output to clear Error Counter of servo motor), then copy mechanical origin to current position, zero return is finished.

(C) come across the limit at first

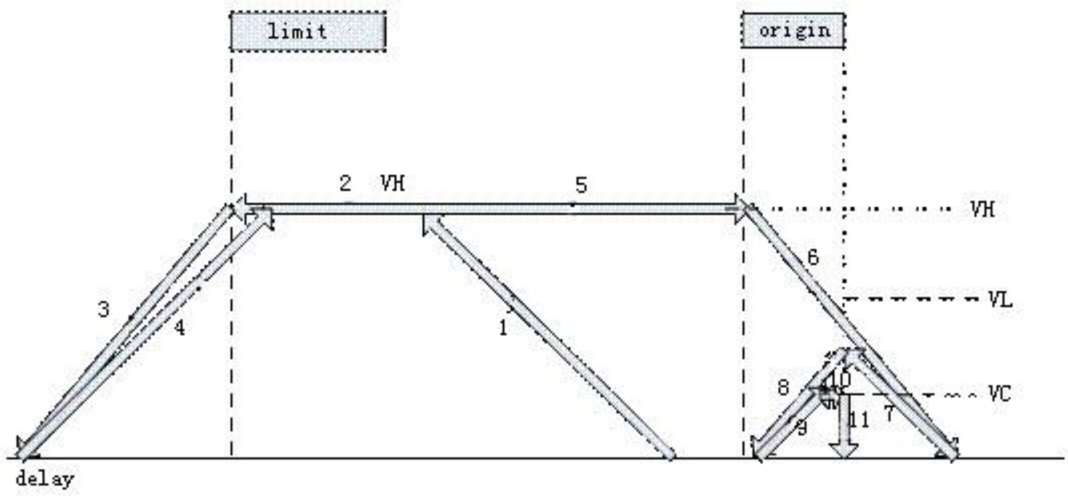
Before back to mechanical origin, the device is between left limit switch and origin switch, decelerate when touching origin, and pass origin switch before speed reaching 0:



Situation 1



Situation 2



Situation 3

Action description:

(1) Accelerate to speed VH with acceleration slope(zero return speed) in origin return direction until touch limit switch.

(2) Decelerate to 0 with deceleration slope when touching limit switch, then accelerate in the direction that is opposite with origin return direction, decelerate to 0 when touching limit switch, then accelerate in the origin return opposite direction until touch origin switch, decelerate to 0 with set deceleration slope.

(3) When accelerate to VL with set slope, three possible situations may occur as the acceleration and deceleration slope is different

Decelerate to 0 with deceleration slope, during the process accelerate to VL the moment touching the origin right signal;

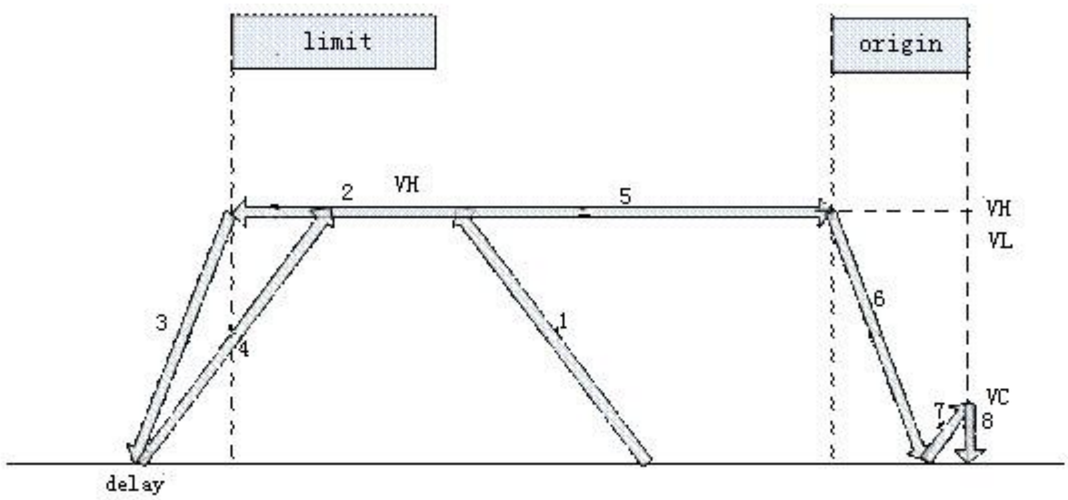
Move on at speed VL until touching origin signal right side, if it still does not touch origin right signal when accelerates to VL with acceleration slope. Decelerate to 0 when touch the origin signal right side;

Decelerate to 0 with set deceleration slope, when accelerate with set acceleration slope and it has touched origin signal right side before reaching speed VL.

(4) Any situation in (3), accelerate to VC (crawl speed) with acceleration slope in the opposite direction after decelerating to 0 and delaying.

(5) Stop action the moment it leaves the right side of origin signal, output clear signal and delay if ‘zero return CLR signal’ is set.(CLR signal time delay in SFD can use ‘zero return CLR signal’ output point to clear Error Counter of servo), then copy mechanical origin position to the current position, zero return is finished.

(D) Before back to mechanical origin, the device is between left limit switch and origin switch, and start to decelerate when touch the origin rising edge, the speed reaches 0 before leaving origin signal right side:



Action description:

(1) Accelerate to VH (zero return speed) with acceleration slope, and move in origin return direction at speed VH until touch the route limit switch.

(2) Decelerate to 0 with deceleration slope when touch the route limit switch, then accelerate in origin return opposite direction until touch the origin switch, decelerate to 0 at set deceleration slope.

(3) Accelerate to VC (crawl speed) with set acceleration slope. Three possible situations may occur as the acceleration and deceleration slope is different：

Stop by pulse at once, when accelerate to VC just the moment touching origin right side;

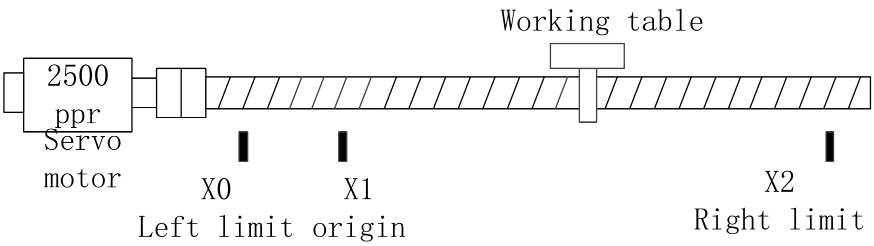
Move on to origin right edge at speed VC until leaves origin signal right edge, stop by pulse at once, if the speed reaches VC before touching the origin right edge;

Stop by pulse at once, if leaves origin right side before reaching speed VC;

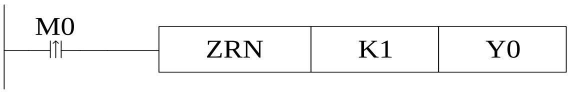
(4) Any situation in (3), stop action after stop by pulse, if 'zero return CLR signal' is set.(CLR signal time delay in SFD can use 'zero return CLR signal' output point to clear Error Counter of servo), then copy mechanical origin position to the current position, zero return is finished.

Example 1

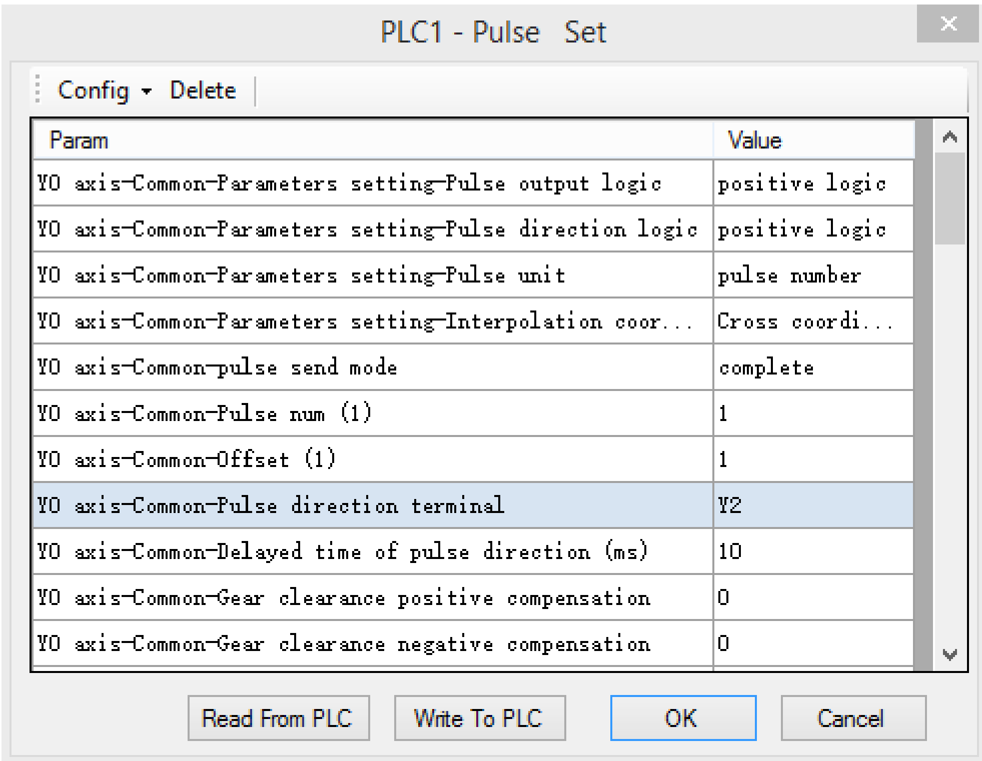
The servo drive (electronic gear ratio is 1:1) controls the servo motor (encoder 2500 ppr), the ball screw pitch is 10mm. the ball screw drives a working table moving left and right. Now the working table must go back to the origin. The left limit switch connects to PLC X0 input, right limit switch connects to PLC X2 input. Origin position switch connects to X1. Origin returning speed VH is 10000Hz, SFD direction delay time is 100ms, creeping speed VC is 100Hz, not count Z phase signal, pulse outputs from Y0, pulse direction terminal is Y2. Mechanical origin position is 0, the acceleration slope is 1000Hz every 100ms. The deceleration slope is 1000Hz every 150ms. Please note there is no near origin signal, so it no needs to set origin returning speed VL.



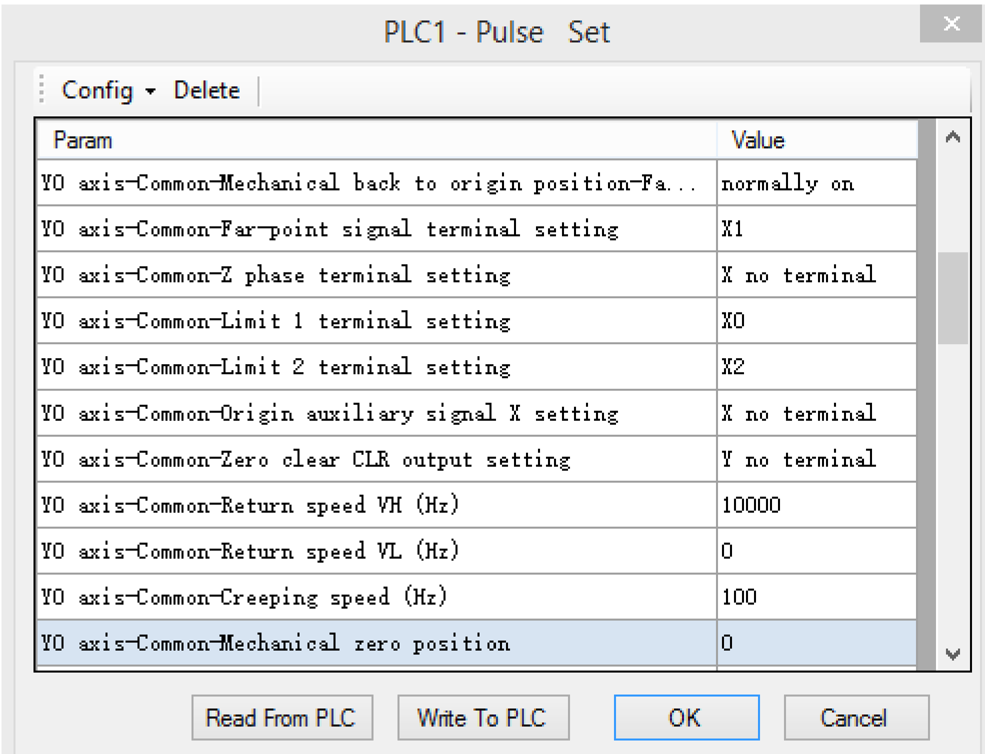
Returning to mechanical origin instruction

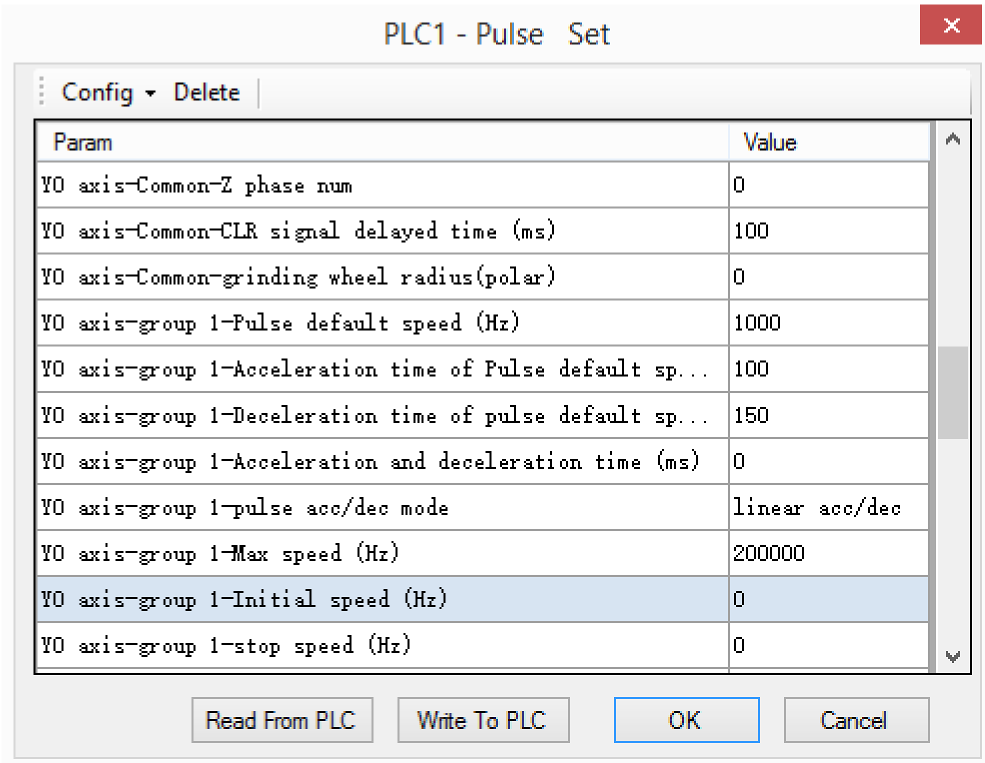


System pararmeters

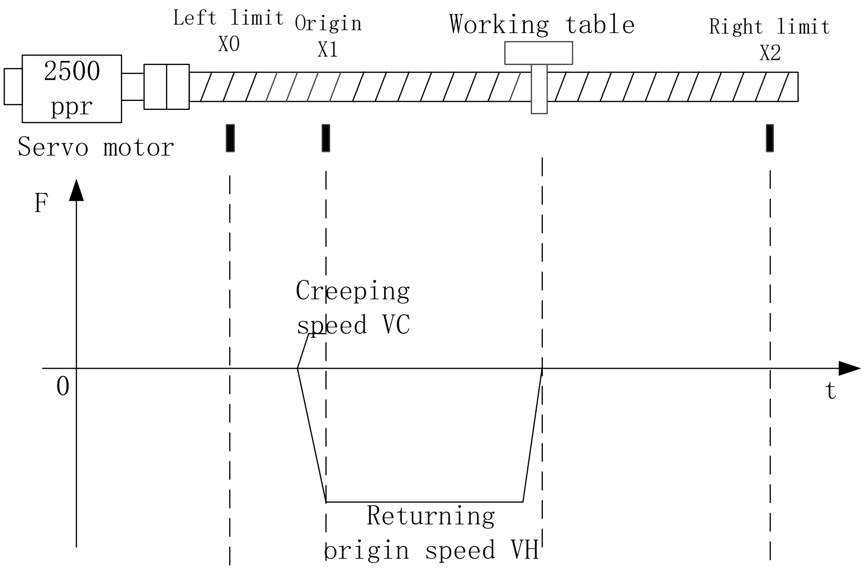


Param	Value
Y0 axis-Common-Parameters setting-Pulse output logic	positive logic
Y0 axis-Common-Parameters setting-Pulse direction logic	positive logic
Y0 axis-Common-Parameters setting-Pulse unit	pulse number
Y0 axis-Common-Parameters setting-Interpolation coor...	Cross coordi...
Y0 axis-Common-pulse send mode	complete
Y0 axis-Common-Pulse num (1)	1
Y0 axis-Common-Offset (1)	1
Y0 axis-Common-Pulse direction terminal	Y2
Y0 axis-Common-Delayed time of pulse direction (ms)	10
Y0 axis-Common-Gear clearance positive compensation	0
Y0 axis-Common-Gear clearance negative compensation	0





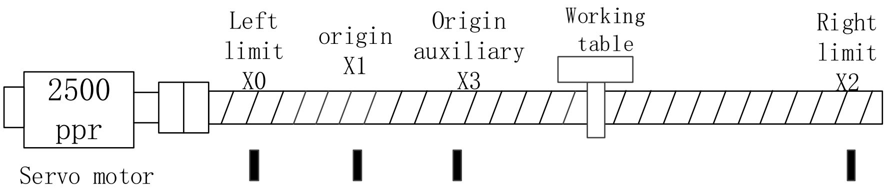
Mechanical returning to origin movement diagram



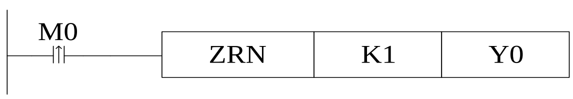
1. At the moment leaving the falling edge of X1 with creeping speed, the mechanical returning origin finished at once.
2. If the setting of origin returning speed, pulse acceleration and deceleration time, left limit origin position is not reasonable, it will cause the working table touching the left limit in the process of touching the origin signal and decelerating. Please avoid the special condition even the software has solution for it.
3. Y2 terminal is OFF when the working table moves from right to left. Y2 terminal is ON when the working table reverse moves with creeping speed and stops.

Example 2

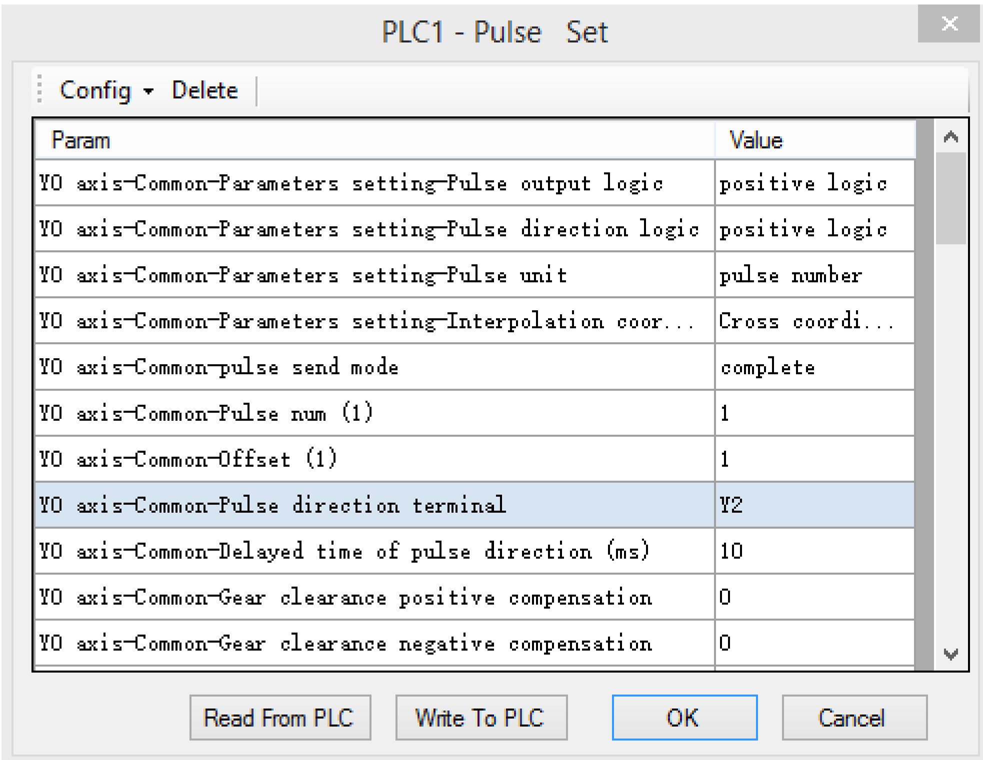
The servo drive (electronic gear ratio is 1:1) controls the servo motor (encoder 2500 ppr), the ball screw pitch is 10mm. the ball screw drives a working table moving left and right. Now the working table must go back to the origin. The left limit switch connects to PLC X0 input, right limit switch connects to PLC X2 input. Origin position switch connects to X1. Origin returning speed VH is 10000Hz, origin returning speed is 1000Hz, SFD direction delay time is 100ms, creeping speed VC is 100Hz, not count Z phase signal, pulse outputs from Y0, pulse direction terminal is Y2. Mechanical origin position is 0, the acceleration slope is 1000Hz every 100ms. The deceleration slope is 1000Hz every 150ms.



Returning to mechanical origin instruction



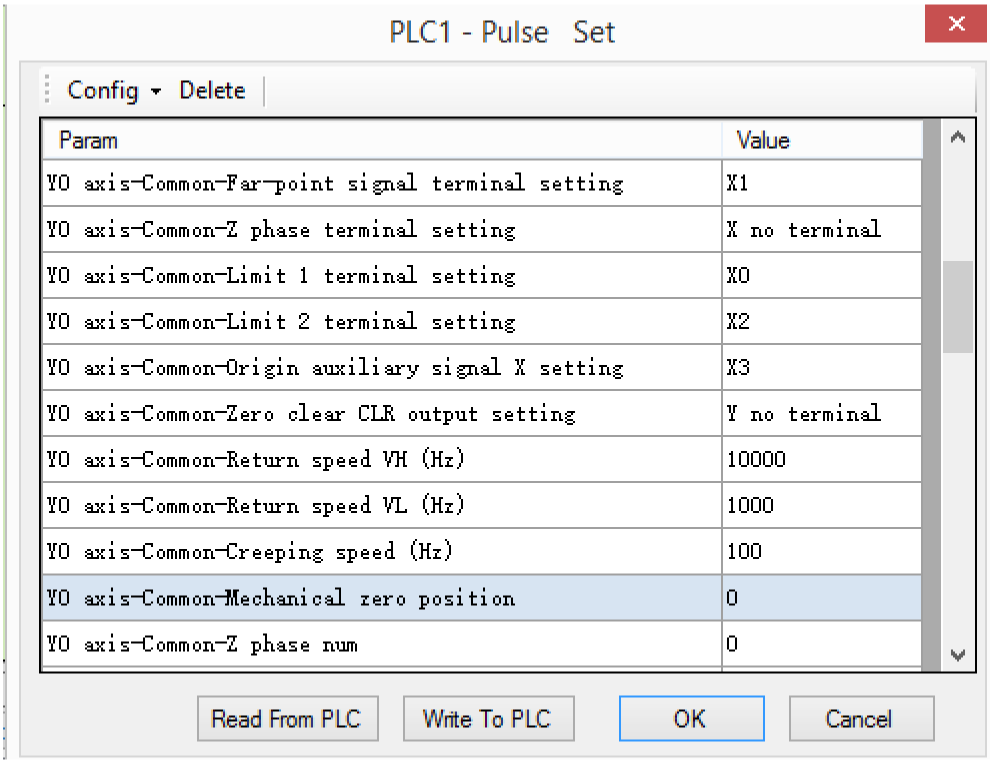
System parameters


PLC1 - Pulse Set

Config Delete

Param	Value
Y0 axis-Common-Parameters setting-Pulse output logic	positive logic
Y0 axis-Common-Parameters setting-Pulse direction logic	positive logic
Y0 axis-Common-Parameters setting-Pulse unit	pulse number
Y0 axis-Common-Parameters setting-Interpolation coor...	Cross coordi...
Y0 axis-Common-pulse send mode	complete
Y0 axis-Common-Pulse num (1)	1
Y0 axis-Common-Offset (1)	1
Y0 axis-Common-Pulse direction terminal	Y2
Y0 axis-Common-Delayed time of pulse direction (ms)	10
Y0 axis-Common-Gear clearance positive compensation	0
Y0 axis-Common-Gear clearance negative compensation	0

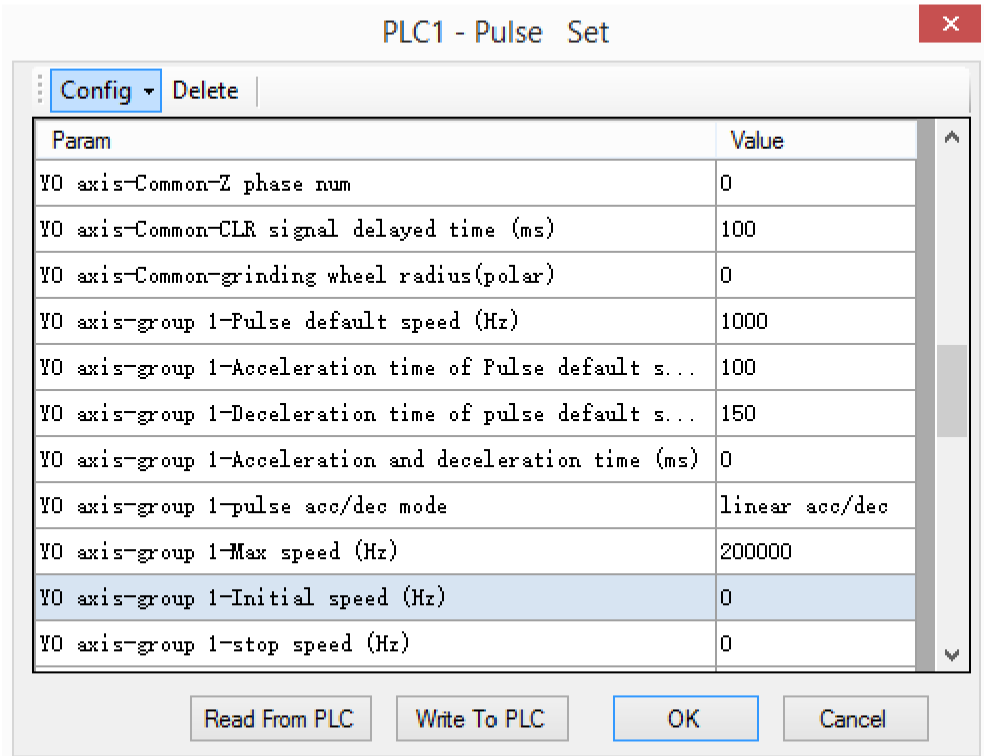
Read From PLC Write To PLC OK Cancel


PLC1 - Pulse Set

Config Delete

Param	Value
Y0 axis-Common-Far-point signal terminal setting	X1
Y0 axis-Common-Z phase terminal setting	X no terminal
Y0 axis-Common-Limit 1 terminal setting	X0
Y0 axis-Common-Limit 2 terminal setting	X2
Y0 axis-Common-Origin auxiliary signal X setting	X3
Y0 axis-Common-Zero clear CLR output setting	Y no terminal
Y0 axis-Common-Return speed VH (Hz)	10000
Y0 axis-Common-Return speed VL (Hz)	1000
Y0 axis-Common-Creeping speed (Hz)	100
Y0 axis-Common-Mechanical zero position	0
Y0 axis-Common-Z phase num	0

Read From PLC Write To PLC OK Cancel

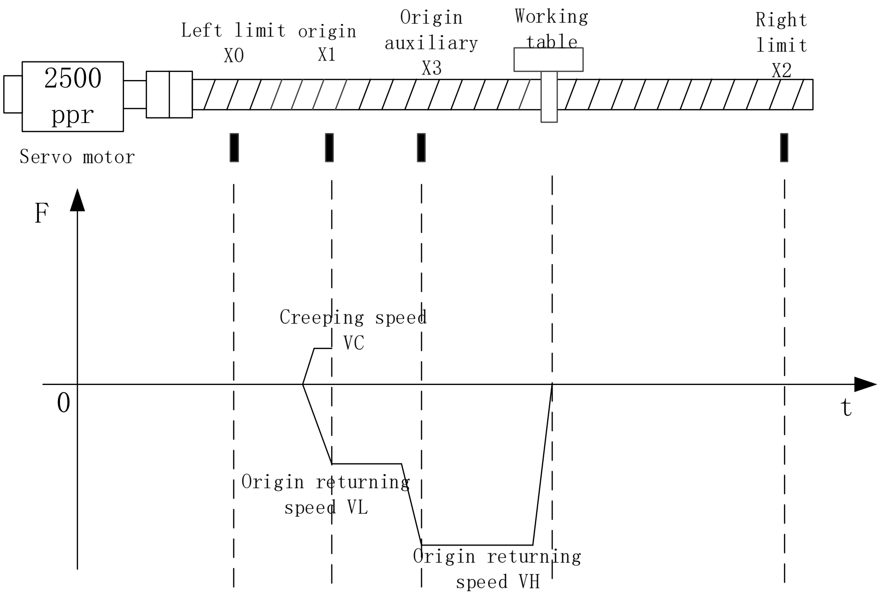

PLC1 - Pulse Set

Config Delete

Param	Value
Y0 axis-Common-Z phase num	0
Y0 axis-Common-CLR signal delayed time (ms)	100
Y0 axis-Common-grinding wheel radius(polar)	0
Y0 axis-group 1-Pulse default speed (Hz)	1000
Y0 axis-group 1-Acceleration time of Pulse default s...	100
Y0 axis-group 1-Deceleration time of pulse default s...	150
Y0 axis-group 1-Acceleration and deceleration time (ms)	0
Y0 axis-group 1-pulse acc/dec mode	linear acc/dec
Y0 axis-group 1-Max speed (Hz)	200000
Y0 axis-group 1-Initial speed (Hz)	0
Y0 axis-group 1-stop speed (Hz)	0

Read From PLC | Write To PLC | OK | Cancel

Mechanical returning origin movement diagram



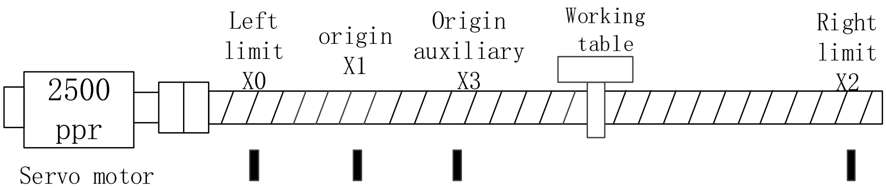
1. At the moment leaving the falling edge of X1 with creeping speed, the mechanical returning origin finished at once.

2. If the setting of origin returning speed, pulse acceleration and deceleration time, left limit origin position is not reasonable, it will cause the working table touching the left limit in the process of touching the origin signal and decelerating. Please avoid the special condition even the software has solution for it.

3. Y2 terminal is OFF when the working table moves from right to left. Y2 terminal is ON when the working table reverse moves with creeping speed and stops.

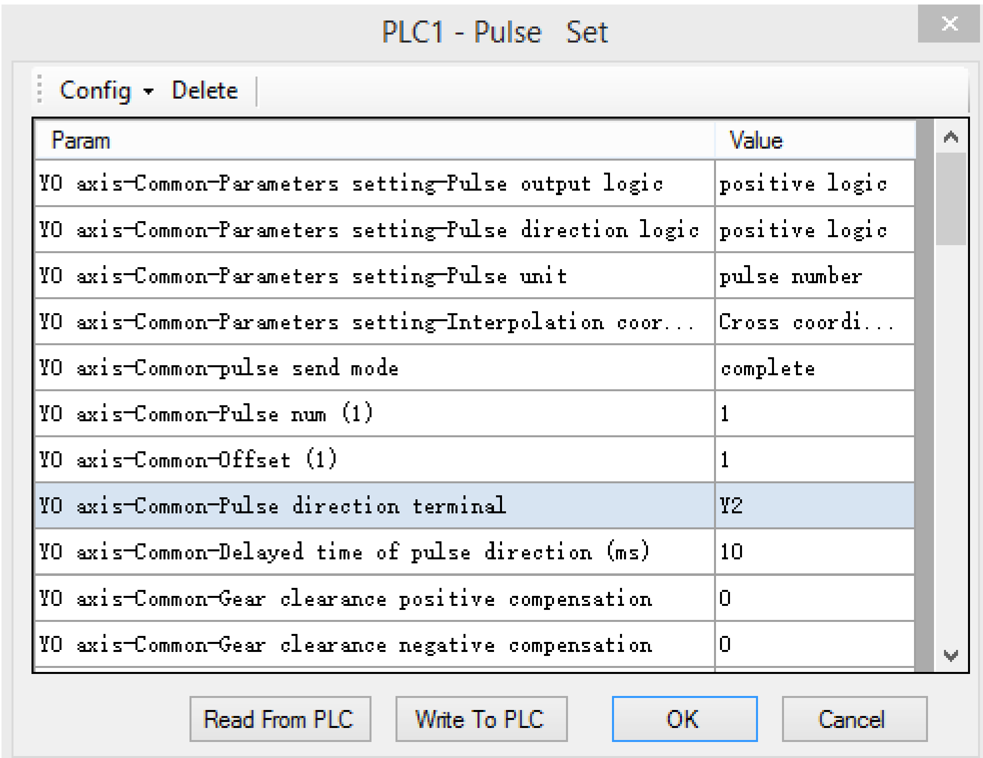
Example 3

The servo drive (electronic gear ratio is 1:1) controls the servo motor (encoder 2500 ppr), the ball screw pitch is 10mm. the ball screw drives a working table moving left and right. Now the working table must go back to the origin. The left limit switch connects to PLC X0 input, right limit switch connects to PLC X2 input. Origin position switch connects to X1. Origin returning speed VH is 10000Hz, origin returning speed is 1000Hz, SFD direction delay time is 100ms, creeping speed VC is 100Hz, count the Z phase number when reverse leaving the origin signal (connect to X4), Z phase number set to 6, pulse outputs from Y0, pulse direction terminal is Y2. Mechanical origin position is 0, the acceleration slope is 1000Hz every 100ms. The deceleration slope is 1000Hz every 150ms.

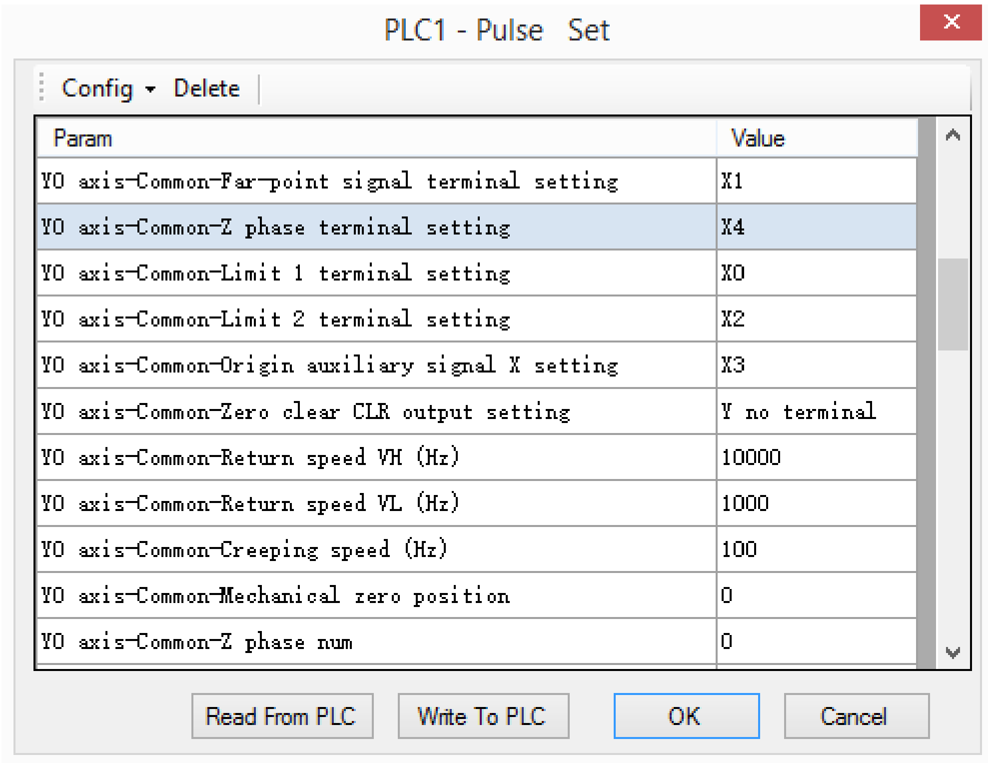


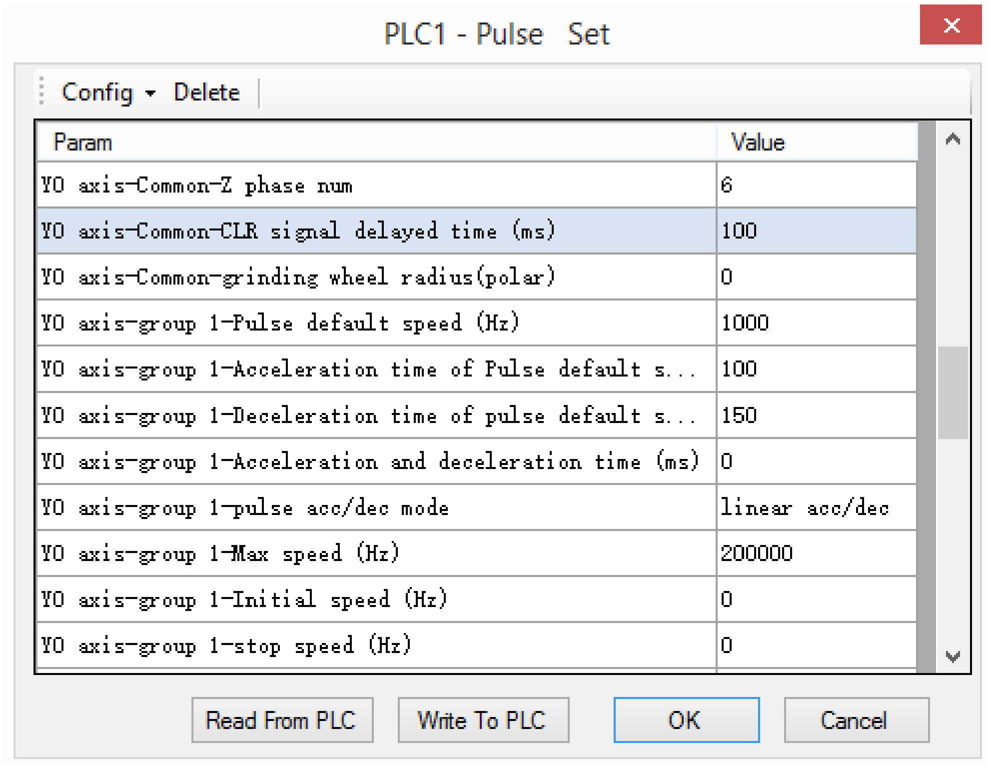
Returning to mechanical origin instruction

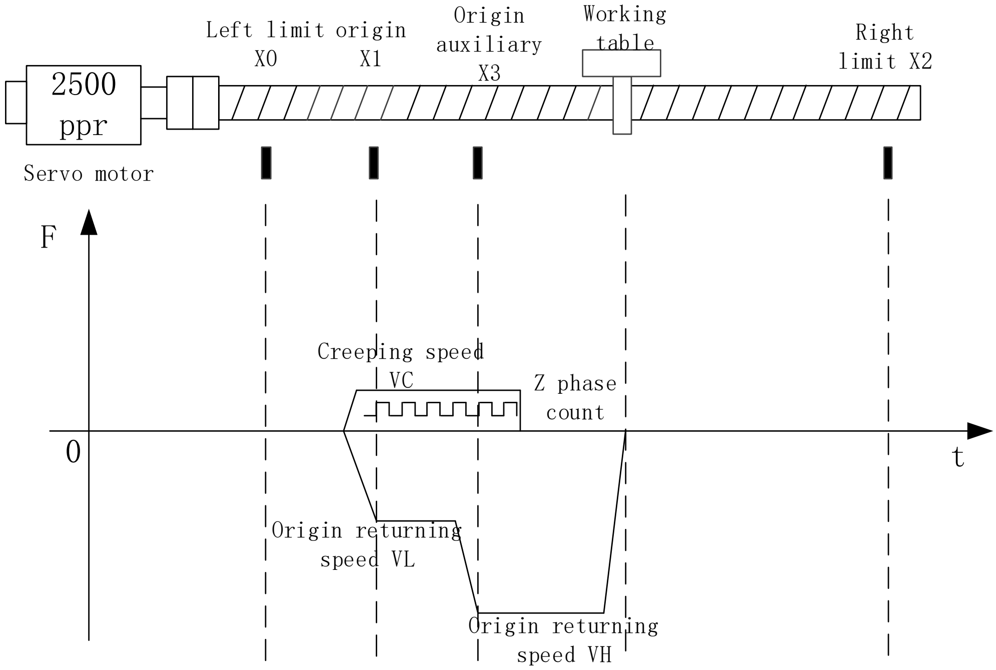
System parameters



Param	Value
Y0 axis-Common-Parameters setting-Pulse output logic	positive logic
Y0 axis-Common-Parameters setting-Pulse direction logic	positive logic
Y0 axis-Common-Parameters setting-Pulse unit	pulse number
Y0 axis-Common-Parameters setting-Interpolation coor...	Cross coordi...
Y0 axis-Common-pulse send mode	complete
Y0 axis-Common-Pulse num (1)	1
Y0 axis-Common-Offset (1)	1
Y0 axis-Common-Pulse direction terminal	Y2
Y0 axis-Common-Delayed time of pulse direction (ms)	10
Y0 axis-Common-Gear clearance positive compensation	0
Y0 axis-Common-Gear clearance negative compensation	0


PLC1 - Pulse Set
Config Delete
Param Value
Y0 axis-Common-Far-point signal terminal setting X1
Y0 axis-Common-Z phase terminal setting X4
Y0 axis-Common-Limit 1 terminal setting X0
Y0 axis-Common-Limit 2 terminal setting X2
Y0 axis-Common-Origin auxiliary signal X setting X3
Y0 axis-Common-Zero clear CLR output setting Y no terminal
Y0 axis-Common-Return speed VH (Hz) 10000
Y0 axis-Common-Return speed VL (Hz) 1000
Y0 axis-Common-Creeping speed (Hz) 100
Y0 axis-Common-Mechanical zero position 0
Y0 axis-Common-Z phase num 0
Read From PLC
Write To PLC
OK
Cancel


PLC1 - Pulse Set
Config Delete
Param Value
Y0 axis-Common-Z phase num 6
Y0 axis-Common-CLR signal delayed time (ms) 100
Y0 axis-Common-grinding wheel radius(polar) 0
Y0 axis-group 1-Pulse default speed (Hz) 1000
Y0 axis-group 1-Acceleration time of Pulse default s... 100
Y0 axis-group 1-Deceleration time of pulse default s... 150
Y0 axis-group 1-Acceleration and deceleration time (ms) 0
Y0 axis-group 1-pulse acc/dec mode linear acc/dec
Y0 axis-group 1-Max speed (Hz) 200000
Y0 axis-group 1-Initial speed (Hz) 0
Y0 axis-group 1-stop speed (Hz) 0
Read From PLC
Write To PLC
OK
Cancel



1. At the moment leaving the falling edge of X1 with creeping speed, it starts to count the Z phase pulse, the pulse will stop when the count value reach the setting value, the mechanical returning origin finished.
2. If the setting of origin returning speed, pulse acceleration and deceleration time, left limit origin position is not reasonable, it will cause the working table touching the left limit in the process of touching the origin signal and decelerating. Please avoid the special condition even the software has solution for it.
3. Y2 terminal is OFF when the working table moves from right to left. Y2 terminal is ON when the working table reverse moves with creeping speed and stops.

6-2-4. Pulse stop [STOP]

Summary

Deceleration stop the pulse.

Pulse number refresh [STOP]			
16 bit instruction	STOP	32 bit instruction	-
Execution condition	Rising / falling edge trigger	Suitable model	XD3, XDM
Hardware requirement	-	Software requirement	-

operands

Operands	Function	Type
S	The stop pulse terminal	bit
D	Pulse stop mode (0: slow stop, 1: urgent stop)	Word 16 bits

Suitable soft component

Bit

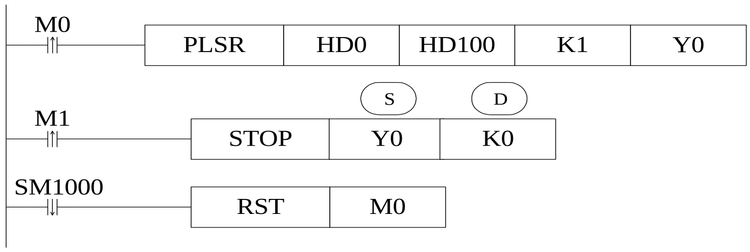
Operand	System								Constant	Module	
	D*	FD	TD*	CD*	DX	DY	DM*	DS*	K/H	ID	QD
D	●	●	●	●	●	●	●	●			

Word

Operand	System						
	X	Y	M*	S*	T*	C*	Dn.m
S		●					

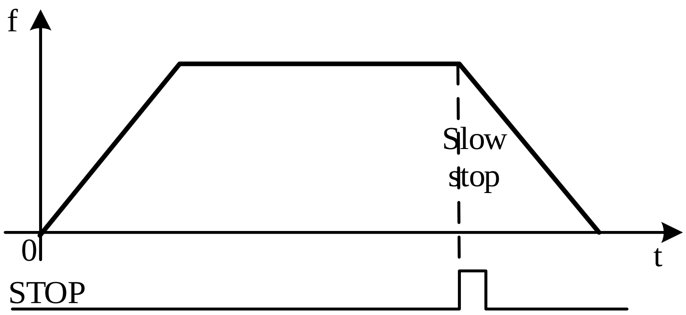
Notes: D includes D, HD; TD includes TD, HTD; CD includes CD, HCD, HSCD, HSD; DM includes DM, DHM; DS includes DS, DHS. M includes M, HM, SM; S includes S and HS; T includes T and HT; C includes C and HC.

Function and Action



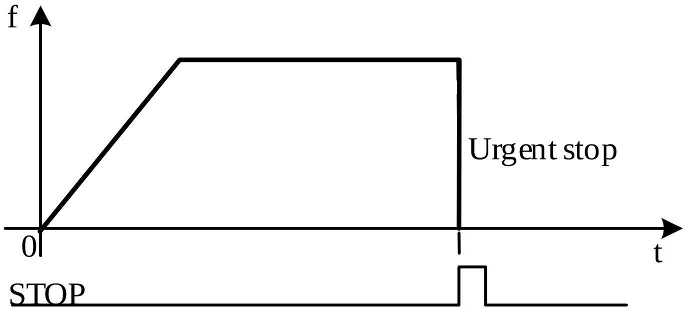
when M0 from OFF to ON, PLSR output pulse from Y0. It stop outputting when the pulse numbers are equal to the setting value. At the rising edge of M1, STOP instruction will stop the pulse output immediately, as D0 is 0, the pulse will slow stop.
Stop all the pulses include PLSR, PLSF, ZRN.

Slow stop (K0)



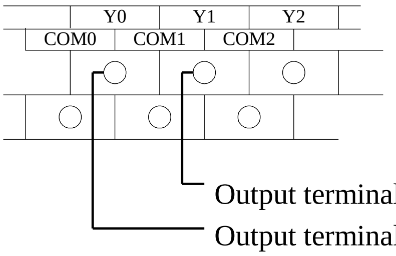
The pulse will stop when pulse present frequency decreases to the pulse end frequency according to the falling slope or the pulse numbers outputting end.

Urgent stop (K1)



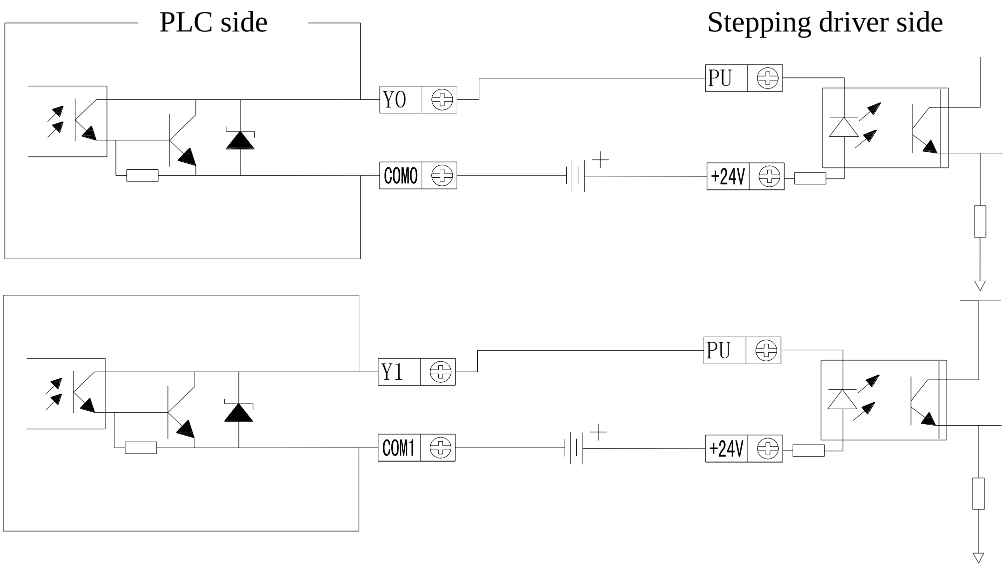
Stop the pulse outputting immediately.

6-3. Output wiring

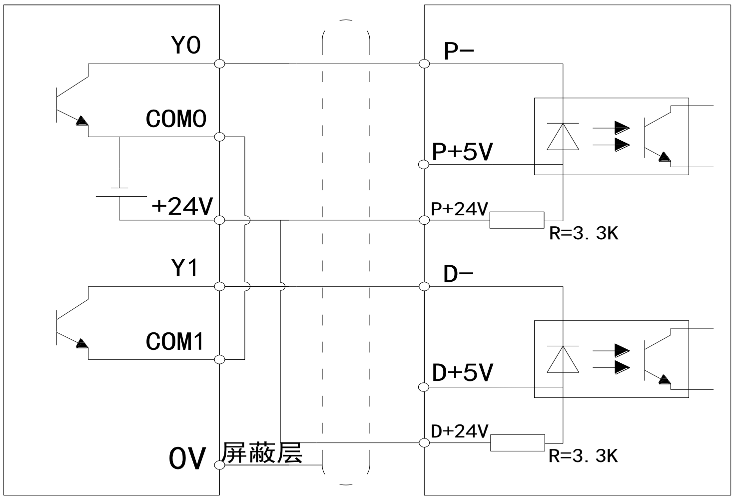


Output terminal Y0: pulse output terminal 0 (single phase)
Output terminal Y1: pulse output terminal 1(single phase)

Below is the graph of the output terminals and step driver wiring:



Note: if the stepper motor pulse and direction terminals are drived by DC5V, please connect 2.2KΩ resistor after pulse and direction terminal.
The wiring diagram of output terminal and servo motor:



Note: please hang the P+5V and D+5V in the air.
Please refer to the XD series hardware manual for the details.

6-4. Relative coils and registers of pulse output

Some flags of pulse output are listed below:

ID	Function	Description	
SM1000	'sending pulse' flag	Being ON when sending the pulse	PULSE_1
SM1001	Direction flag	1 is positive direction, the corresponding direction port is ON	
SM1002	Overflow flag of accumulated pulse number	ON when overflow	
SM1003	Overflow flag of accumulated pulse equivalent	ON when overflow	
SM1010	Pulse error flag	ON when pulse error	
SM1020	'sending pulse' flag	Being ON when sending the pulse	PULSE_2
SM1021	Direction flag	1 is positive direction, the corresponding direction port is ON	
SM1022	Overflow flag of accumulated pulse number	ON when overflow	
SM1023	Overflow flag of accumulated pulse equivalent	ON when overflow	
SM1030	Pulse error flag	ON when pulse error	
SM1040	'sending pulse' flag	Being ON when sending the pulse	PULSE_3

SM1041	Direction flag	1 is positive direction, the corresponding direction port is ON	
SM1042	Overflow flag of accumulated pulse number	ON when overflow	
SM1043	Overflow flag of accumulated pulse equivalent	ON when overflow	
SM1050	Pulse error flag	ON when pulse error	
SM1060	‘sending pulse’ flag	Being ON when sending the pulse	PULSE_4
SM1061	Direction flag	1 is positive direction, the corresponding direction port is ON	
SM1062	Overflow flag of accumulated pulse number	ON when overflow	
SM1063	Overflow flag of accumulated pulse equivalent	ON when overflow	
SM1070	Pulse error flag	ON when pulse error	
SM1080	‘sending pulse’ flag	Being ON when sending the pulse	PULSE_5
SM1081	Direction flag	1 is positive direction, the corresponding direction port is ON	
SM1082	Overflow flag of accumulated pulse number	ON when overflow	
SM1083	Overflow flag of accumulated pulse equivalent	ON when overflow	
SM1090	Pulse error flag	ON when pulse error	
SM1100	‘sending pulse’ flag	Being ON when sending the pulse	PULSE_6
SM1101	Direction flag	1 is positive direction, the corresponding direction port is ON	
SM1102	Overflow flag of accumulated pulse number	ON when overflow	
SM1103	Overflow flag of accumulated pulse equivalent	ON when overflow	
SM1110	Pulse error flag	ON when pulse error	
SM1120	‘sending pulse’ flag	Being ON when sending the pulse	PULSE_7
SM1121	Direction flag	1 is positive direction, the corresponding direction port is ON	
SM1122	Overflow flag of accumulated pulse number	ON when overflow	
SM1123	Overflow flag of accumulated pulse equivalent	ON when overflow	

SM1130	Pulse error flag	ON when pulse error	
SM1140	'sending pulse' flag	Being ON when sending the pulse	PULSE_8
SM1141	Direction flag	1 is positive direction, the corresponding direction port is ON	
SM1142	Overflow flag of accumulated pulse number	ON when overflow	
SM1143	Overflow flag of accumulated pulse equivalent	ON when overflow	
SM1150	Pulse error flag	ON when pulse error	
SM1160	'sending pulse' flag	Being ON when sending the pulse	PULSE_9
SM1161	Direction flag	1 is positive direction, the corresponding direction port is ON	
SM1162	Overflow flag of accumulated pulse number	ON when overflow	
SM1163	Overflow flag of accumulated pulse equivalent	ON when overflow	
SM1170	Pulse error flag	ON when pulse error	
SM1180	'sending pulse' flag	Being ON when sending the pulse	PULSE_10
SM1181	Direction flag	1 is positive direction, the corresponding direction port is ON	
SM1182	Overflow flag of accumulated pulse number	ON when overflow	
SM1183	Overflow flag of accumulated pulse equivalent	ON when overflow	
SM1190	Pulse error flag	ON when pulse error	

Some special registers of pulse output are listed below:

ID	Function	Description	
SD1000	Current segment(No. n)		PULSE_1
SD1001			
SD1002	Low 16 bit of current pulse (unit is pulse number)		
SD1003	High 16 bit of current pulse (unit is pulse number)		
SD1004	Low 16 bit of current pulse (unit is pulse equivalent)		
SD1005	High 16 bit of current pulse (unit is pulse equivalent)		
SD1006	Low 16 bit of current output frequency (unit is pulse number)		

SD1007	High 16 bit of current output frequency (unit is pulse number)		
SD1008	Low 16 bit of current output frequency (unit is pulse equivalent)		
SD1009	High 16 bit of current output frequency (unit is pulse equivalent)		
SD1010	Pulse error message	1：pulse data block error 2：equivalent mode：pulse number/ turn, shift amount/turn is 0 3：system parameter block number error 4：pulse data block exceed max limit 10：origin return do not set near point signal 11：origin return speed is 0 12：origin return crawling speed is 0 13：origin return speed and origin auxiliary speed direction is different	
SD1011	Error pulse data block number		
SD1020	Current segment(No. n)		PULSE_2
SD1021			
SD1022	Low 16 bit of current pulse (unit is pulse number)		
SD1023	High 16 bit of current pulse (unit is pulse number)		
SD1024	Low 16 bit of current pulse (unit is pulse equivalent)		
SD1025	High 16 bit of current pulse (unit is pulse equivalent)		
SD1026	Low 16 bit of current output frequency (unit is pulse number)		
SD1027	High 16 bit of current output frequency (unit is pulse number)		
SD1028	Low 16 bit of current output frequency (unit is pulse equivalent)		
SD1029	High 16 bit of current output frequency (unit is pulse equivalent)		
SD1030	Pulse error message	1：pulse data block error	

		2：equivalent mode：pulse number/ turn, shift amount/turn is 0 3：system parameter block number error 4：pulse data block exceed max limit 10：origin return do not set near point signal 11：origin return speed is 0 12：origin return crawling speed is 0 13：origin return speed and origin auxiliary speed direction is different	
SD1031	Error pulse data block number		
SD1040	Current segment(No. n)		PULSE_3
SD1041			
SD1042	Low 16 bit of current pulse (unit is pulse number)		
SD1043	High 16 bit of current pulse (unit is pulse number)		
SD1044	Low 16 bit of current pulse (unit is pulse equivalent)		
SD1045	High 16 bit of current pulse (unit is pulse equivalent)		
SD1046	Low 16 bit of current output frequency (unit is pulse number)		
SD1047	High 16 bit of current output frequency (unit is pulse number)		
SD1048	Low 16 bit of current output frequency (unit is pulse equivalent)		
SD1049	High 16 bit of current output frequency (unit is pulse equivalent)		
SD1050	Pulse error message	1：pulse data block error 2：equivalent mode：pulse number/ turn, shift amount/turn is 0 3：system parameter block number error 4：pulse data block exceed max limit 10：origin return do not set near point signal 11：origin return speed is 0	

		12：origin return crawling speed is 0 13：origin return speed and origin auxiliary speed direction is different	
SD1051	Error pulse data block number		
SD1060	Current segment(No. n)		PULSE_4
SD1061			
SD1062	Low 16 bit of current pulse (unit is pulse number)		
SD1063	High 16 bit of current pulse (unit is pulse number)		
SD1064	Low 16 bit of current pulse (unit is pulse equivalent)		
SD1065	High 16 bit of current pulse (unit is pulse equivalent)		
SD1066	Low 16 bit of current output frequency (unit is pulse number)		
SD1067	High 16 bit of current output frequency (unit is pulse number)		
SD1068	Low 16 bit of current output frequency (unit is pulse equivalent)		
SD1069	High 16 bit of current output frequency (unit is pulse equivalent)		
SD1070	Pulse error message	1：pulse data block error 2：equivalent mode：pulse number/ turn, shift amount/turn is 0 3：system parameter block number error 4：pulse data block exceed max limit 10：origin return do not set near point signal 11：origin return speed is 0 12：origin return crawling speed is 0 13：origin return speed and origin auxiliary speed direction is different	
SD1071	Error pulse data block number		
SD1080	Current segment(No. n)		PULSE_5

SD1081			
SD1082	Low 16 bit of current pulse (unit is pulse number)		
SD1083	High 16 bit of current pulse (unit is pulse number)		
SD1084	Low 16 bit of current pulse (unit is pulse equivalent)		
SD1085	High 16 bit of current pulse (unit is pulse equivalent)		
SD1086	Low 16 bit of current output frequency (unit is pulse number)		
SD1087	High 16 bit of current output frequency (unit is pulse number)		
SD1088	Low 16 bit of current output frequency (unit is pulse equivalent)		
SD1089	High 16 bit of current output frequency (unit is pulse equivalent)		
SD1090	Pulse error message	1： pulse data block error 2： equivalent mode： pulse number/ turn, shift amount/turn is 0 3： system parameter block number error 4： pulse data block exceed max limit 10： origin return do not set near point signal 11： origin return speed is 0 12： origin return crawling speed is 0 13： origin return speed and origin auxiliary speed direction is different	
SD1091	Error pulse data block number		
SD1100	Current segment(No. n)		PULSE_6
SD1101			
SD1102	Low 16 bit of current pulse (unit is pulse number)		
SD1103	High 16 bit of current pulse (unit is pulse number)		
SD1104	Low 16 bit of current pulse (unit is pulse equivalent)		
SD1105	High 16 bit of current pulse (unit is pulse equivalent)		

SD1106	Low 16 bit of current output frequency (unit is pulse number)		
SD1107	High 16 bit of current output frequency (unit is pulse number)		
SD1108	Low 16 bit of current output frequency (unit is pulse equivalent)		
SD1109	High 16 bit of current output frequency (unit is pulse equivalent)		
SD1110	Pulse error message	1：pulse data block error 2：equivalent mode：pulse number/ turn, shift amount/turn is 0 3：system parameter block number error 4：pulse data block exceed max limit 10：origin return do not set near point signal 11：origin return speed is 0 12：origin return crawling speed is 0 13：origin return speed and origin auxiliary speed direction is different	
SD1111	Error pulse data block number		
SD1120	Current segment(No. n)		PULSE_7
SD1121			
SD1122	Low 16 bit of current pulse (unit is pulse number)		
SD1123	High 16 bit of current pulse (unit is pulse number)		
SD1124	Low 16 bit of current pulse (unit is pulse equivalent)		
SD1125	High 16 bit of current pulse (unit is pulse equivalent)		
SD1126	Low 16 bit of current output frequency (unit is pulse number)		
SD1127	High 16 bit of current output frequency (unit is pulse number)		
SD1128	Low 16 bit of current output frequency (unit is pulse equivalent)		
SD1129	High 16 bit of current output frequency (unit is pulse equivalent)		

SD1130	Pulse error message	1：pulse data block error 2：equivalent mode：pulse number/ turn, shift amount/turn is 0 3：system parameter block number error 4：pulse data block exceed max limit 10：origin return do not set near point signal 11：origin return speed is 0 12：origin return crawling speed is 0 13：origin return speed and origin auxiliary speed direction is different	
SD1131	Error pulse data block number		
SD1140	Current segment(No. n)		
SD1141			
SD1142	Low 16 bit of current pulse (unit is pulse number)		
SD1143	High 16 bit of current pulse (unit is pulse number)		
SD1144	Low 16 bit of current pulse (unit is pulse equivalent)		
SD1145	High 16 bit of current pulse (unit is pulse equivalent)		
SD1146	Low 16 bit of current output frequency (unit is pulse number)		
SD1147	High 16 bit of current output frequency (unit is pulse number)		
SD1148	Low 16 bit of current output frequency (unit is pulse equivalent)		
SD1149	High 16 bit of current output frequency (unit is pulse equivalent)		
SD1150	Pulse error message	1：pulse data block error 2：equivalent mode：pulse number/ turn, shift amount/turn is 0 3：system parameter block number error 4：pulse data block exceed max limit 10：origin return do not set near point signal 11：origin return speed is 0	PULSE_8

		12：origin return crawling speed is 0 13：origin return speed and origin auxiliary speed direction is different	
SD1151	Error pulse data block number		
SD1160	Current segment(No. n)		PULSE_9
SD1161			
SD1162	Low 16 bit of current pulse (unit is pulse number)		
SD1163	High 16 bit of current pulse (unit is pulse number)		
SD1164	Low 16 bit of current pulse (unit is pulse equivalent)		
SD1165	High 16 bit of current pulse (unit is pulse equivalent)		
SD1166	Low 16 bit of current output frequency (unit is pulse number)		
SD1167	High 16 bit of current output frequency (unit is pulse number)		
SD1168	Low 16 bit of current output frequency (unit is pulse equivalent)		
SD1169	High 16 bit of current output frequency (unit is pulse equivalent)		
SD1170	Pulse error message	1：pulse data block error 2：equivalent mode：pulse number/ turn, shift amount/turn is 0 3：system parameter block number error 4：pulse data block exceed max limit 10：origin return do not set near point signal 11：origin return speed is 0 12：origin return crawling speed is 0 13：origin return speed and origin auxiliary speed direction is different	
SD1171	Error pulse data block number		
SD1180	Current segment(No. n)		
SD1181			

SD1182	Low 16 bit of current pulse (unit is pulse number)		PULSE-_10
SD1183	High 16 bit of current pulse (unit is pulse number)		
SD1184	Low 16 bit of current pulse (unit is pulse equivalent)		
SD1185	High 16 bit of current pulse (unit is pulse equivalent)		
SD1186	Low 16 bit of current output frequency (unit is pulse number)		
SD1187	High 16 bit of current output frequency (unit is pulse number)		
SD1188	Low 16 bit of current output frequency (unit is pulse equivalent)		
SD1189	High 16 bit of current output frequency (unit is pulse equivalent)		
SD1190	Pulse error message	1： pulse data block error 2： equivalent mode： pulse number/ turn, shift amount/turn is 0 3： system parameter block number error 4： pulse data block exceed max limit 10： origin return do not set near point signal 11： origin return speed is 0 12： origin return crawling speed is 0 13： origin return speed and origin auxiliary speed direction is different	
SD1191	Error pulse data block number		

High speed special data register HSD (power-loss memory)

Code	**Function**	**Description**	
HSD0	Low 16 bit of accumulated pulse number (unit is pulse number)		PULSE _1
HSD1	High 16 bit of accumulated pulse number (unit is pulse number)		
HSD2	Low 16 bit of accumulated pulse number (unit is pulse equivalent)		
HSD3	High 16 bit of accumulated pulse number (unit is pulse equivalent)		
HSD4	Low 16 bit of accumulated pulse number (unit is pulse number)		PULSE _2
HSD5	High 16 bit of accumulated pulse number (unit is pulse number)		

HSD6	Low 16 bit of accumulated pulse number (unit is pulse equivalent)		
HSD7	High 16 bit of accumulated pulse number (unit is pulse equivalent)		
HSD8	Low 16 bit of accumulated pulse number (unit is pulse number)		PULSE _3
HSD9	High 16 bit of accumulated pulse number (unit is pulse number)		
HSD10	Low 16 bit of accumulated pulse number (unit is pulse equivalent)		
HSD11	High 16 bit of accumulated pulse number (unit is pulse equivalent)		
HSD12	Low 16 bit of accumulated pulse number (unit is pulse number)		PULSE _4
HSD13	High 16 bit of accumulated pulse number (unit is pulse number)		
HSD14	Low 16 bit of accumulated pulse number (unit is pulse equivalent)		
HSD15	High 16 bit of accumulated pulse number (unit is pulse equivalent)		
HSD16	Low 16 bit of accumulated pulse number (unit is pulse number)		PULSE _5
HSD17	High 16 bit of accumulated pulse number (unit is pulse number)		
HSD18	Low 16 bit of accumulated pulse number (unit is pulse equivalent)		
HSD19	High 16 bit of accumulated pulse number (unit is pulse equivalent)		
HSD20	Low 16 bit of accumulated pulse number (unit is pulse number)		PULSE _6
HSD21	High 16 bit of accumulated pulse number (unit is pulse number)		
HSD22	Low 16 bit of accumulated pulse number (unit is pulse equivalent)		
HSD23	High 16 bit of accumulated pulse number (unit is pulse equivalent)		
HSD24	Low 16 bit of accumulated pulse number (unit is pulse number)		PULSE _7
HSD25	High 16 bit of accumulated pulse number (unit is pulse number)		
HSD26	Low 16 bit of accumulated pulse number (unit is pulse equivalent)		
HSD27	High 16 bit of accumulated pulse number (unit is pulse equivalent)		
HSD28	Low 16 bit of accumulated pulse number (unit is pulse number)		PULSE _8
HSD29	High 16 bit of accumulated pulse number (unit is pulse number)		
HSD30	Low 16 bit of accumulated pulse number (unit is pulse equivalent)		
HSD31	High 16 bit of accumulated pulse number (unit is pulse equivalent)		
HSD32	Low 16 bit of accumulated pulse number (unit is pulse number)		PULSE _9
HSD33	High 16 bit of accumulated pulse number (unit is pulse number)		
HSD34	Low 16 bit of accumulated pulse number (unit is pulse equivalent)		
HSD35	High 16 bit of accumulated pulse number (unit is pulse equivalent)		
HSD36	Low 16 bit of accumulated pulse number (unit is pulse number)		

HSD37	High 16 bit of accumulated pulse number (unit is pulse number)		PULSE_10
HSD38	Low 16 bit of accumulated pulse number (unit is pulse equivalent)		
HSD39	High 16 bit of accumulated pulse number (unit is pulse equivalent)		

7 Communication Function

This chapter mainly includes: basic concept of communication, Modbus communication and field bus X-NET.

Relative Instruction

Mnemonic	Function	Circuit and soft components	Chapter
MODBUS Communication			
COLR	Coil Read	COLR S1 S2 S3 D1 D2	7-2-3
INPR	Input coil read	INPR S1 S2 S3 D1 D2	7-2-3
COLW	Single coil write	COLW D1 D2 S1 S2	7-2-3
MCLW	Multi-coil write	MCLW D1 D2 D3 S1 S2	7-2-3
REGR	Register read	REGR S1 S2 S3 D1 D2	7-2-3
INRR	Input register read	INRR S1 S2 S3 D1 D2	7-2-3
REGW	Single register write	REGW D1 D2 S1 S2	7-2-3
MRGW	Multi-register write	MRGW D1 D2 D3 S1 S2	7-2-3
X-NET communication			
BIT_READ	Read bit	BIT_READ K1 K2 K3 K0 K8 Y0	7-3-5
BIT_WRITE	Write bit	BIT_WRITE K1 K3 K2 K0 K8 M0	7-3-5
REG_READ	Read register	REG_READ K1 K2 K128 K0 K10 D0	7-3-5
REG_WRITE	Write register	REG_WRITE K2 K5 K128 K0 K10 D0	7-3-5

7-1． Summary

XD3, XDM series PLC main units can fulfill your requirement on communication and network. They not only support Modbus RTU, but also support Modbus ASCII and field bus X-NET. XD3, XDM series PLC offer multiple communication methods, with which you can communicate with the devices (such as printer, instruments etc.) that have Modbus communication protocol.

7-1-1． COM port

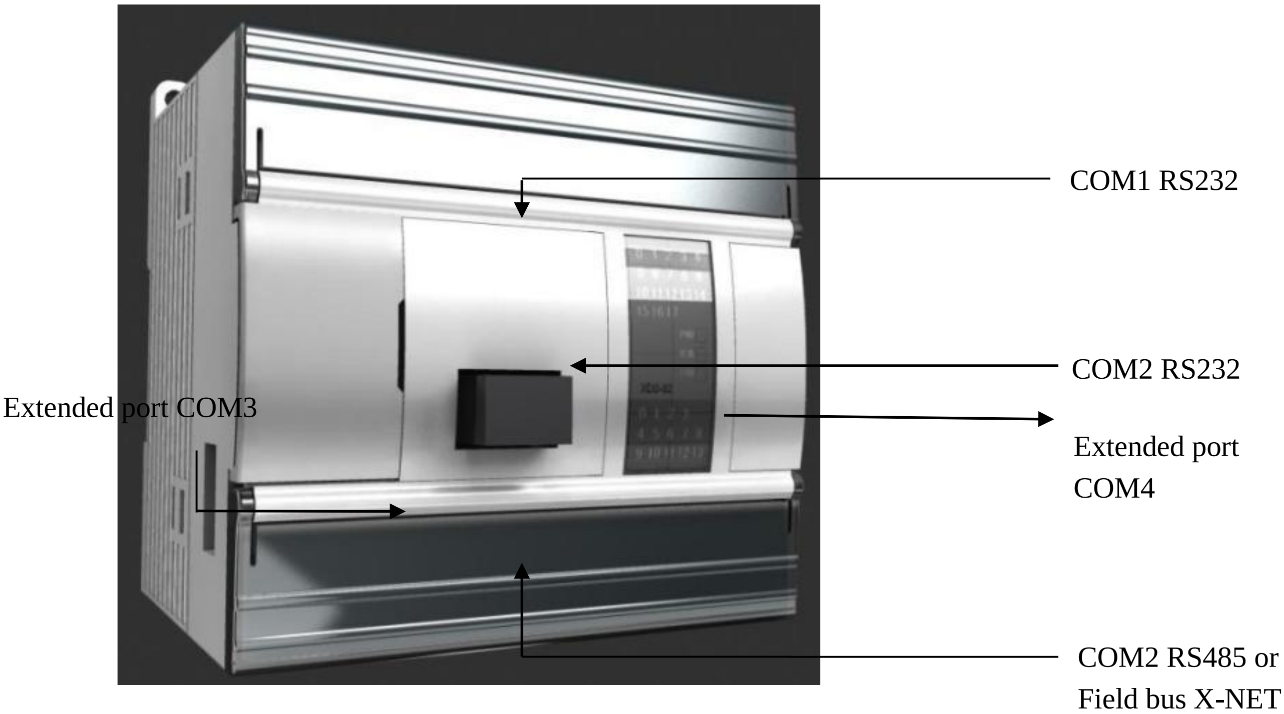
COM Port

XD3, XDM series PLC have 2 COM ports (Port1, Port2)

COM 1 (Port1) is programming port, support RS232, and can be used to download the program and connect with the other devices. The parameters (baud rate, data bit etc) of this COM port can be set by software.

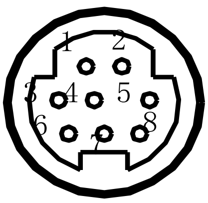
Note: If COM1 can't connect to PC successfully after parameters are changed, users can stop the PLC when start, and then initialize the PLC.

COM 2 (Port2) is communication port. It can be used to download program and connect with other devices. The parameters (baud rate, data bit etc) of this COM port can be changed by software. Port2 supports both RS232 and RS485 (RS485 is at output terminal, terminal A is 485+, terminal B is 485-) and X-NET. But these two ports cannot be used at the same time.



RS232 port

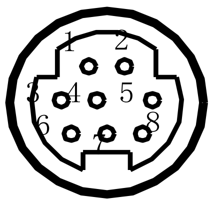
- **COM1 (Port1) Pin definiti**



2：PRG
4：RxD
5：TxD
6：VCC
8：GND

Mini Din 8 pin female port

COM2 (Port2) Pin definition:



4：RxD
5：TxD
8：GND

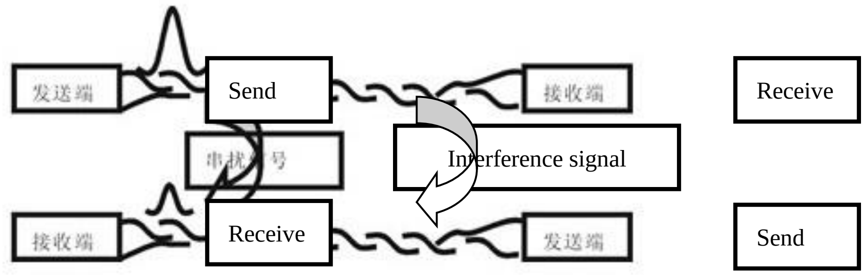
Mini Din 8 pin female port

Note:

1. COM1 supports RS232 and USB port, but they cannot be used at the same time.
2. COM2 supports X-NET and RS485, but X-NET and RS485 cannot be used at the same time.

2. RS485 port

About RS485 port, A is “+” signal、B is “-” signal.
Please use twisted pair cable for RS485. (See below diagram). But shielded twisted pair cable is better and the single-ended connect to the ground.



7-1-2． Communication parameters

Communication Parameters

Station	Modbus station number: 1~254
Baud Rate	300bps~115.2Kbps
Data Bit	8 data bits, 7 data bits
Stop Bit	2 stop bits, 1 stop bit
Parity	Even, Odd, No check

The default parameters of COM1: Station number is 1, baud rate is 19200bps, 8 data bits, 1 stop bit, even parity.
Note: Do not modify COM1 parameters, otherwise connection between PLC and PC may fail!

XD3, XDM series PLC can set the parameters by the COM ports.

COM1	Function	Description	Note
SFD600*	Communication mode		See value of corresponding bit
SFD601*	Communication format	Baud rate, data bit, stop bit, parity	See value of corresponding bit
SFD602*	Frame timeout judgment time	Unit: character	High 8 bits invalid
SFD603*	Reply timeout judgment time		High 8 bits invalid
SFD604	Delay time before sending		Unit: ms
COM2			
SFD610*	Communication mode		See value of corresponding bit
SFD611*	Communication format	Baud rate, data bit, stop bit, parity	See value of corresponding bit
SFD612*	Frame timeout judgment time		Unit: ms

SFD613*	Reply timeout judgment time		Unit: ms, if set to be 0, it means no timeout wait
SFD614	delay time before sending		Unit: ms

Timeout:

Frame timeout judgment set to be 0: then wait for time of a character, and it means over if timeout; 8bit unsigned number.

Reply timeout judgment set to be 0: no timeout; 16bits unsigned number.

Delay time before sending set to be 0: no delay; 16bits unsigned number.

Value of corresponding bit in SFD600, SFD610

Bit	Value
0~7：Modbus station No	Modbus station No
8~15：communication mode	0：modbus RTU mode (default) 1：modbus ASCII mode

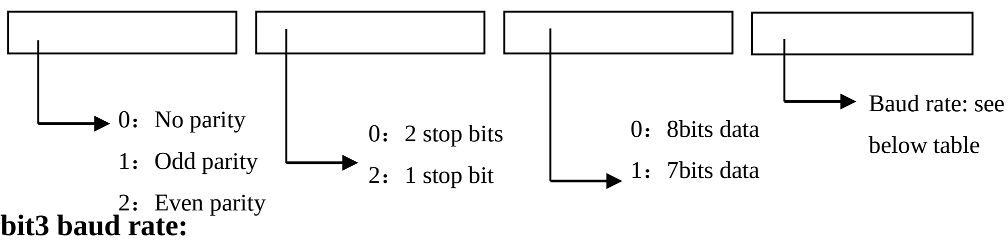
Value of corresponding bit in SFD601, SFD611

Bit	Value		
0~1：Buffer bits	0x0, 8bits	0x1, 16bits	
2：Start symbol	0x0, no start symbol	0x1, start symbol	
3：End symbol	0x0, no end symbol	0x1, end symbol	
4~7：extra communication parity	0x0, no extra parity	0x1, and parity	0x2, CRC parity
8~15：Reserved			

XD3 series communication parameters

SFD601（**COM1**）**/** SFD611（**COM2**）**:**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0



bit0~bit3 baud rate:

Bit	Value		
0~3: baud rate	0x0, BaudRate600	0x1, BaudRate1200	0x2, BaudRate2400
	0x3, BaudRate4800	0x4, BaudRate9600	0x5, BaudRate19200
	0x6, BaudRate38400	0x7, BaudRate57600	0x8, BaudRate115200
	0x9,BaudRate192000	0xA,BaudRate256000	0xB,BaudRate288000
	0xC,BaudRate384000	0xD,BaudRate512000	0xE,BaudRate576000

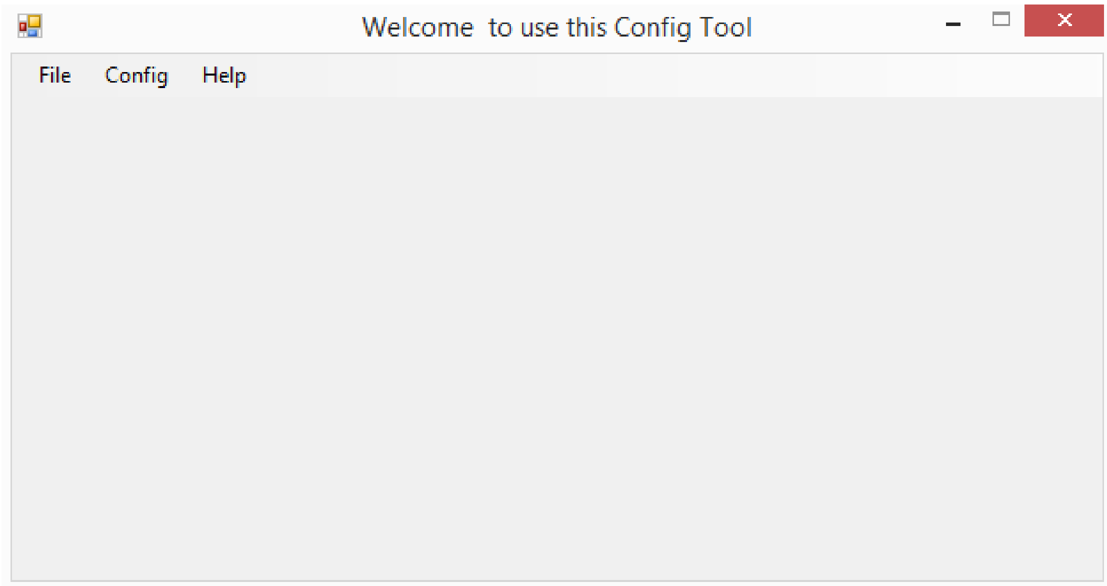
	0xF, BaudRate768000		
4~7: data bit	0x0, 8bits	0x1, 7bits	
8~11: stop bit	0x0, 2bits		0x2, 1bit
12~15: odd and even parity	0x0, none	0x1, odd parity	0x2, even parity

Note: Users don’t have to modify each communication parameter by SFD separately. It is much more convenient to modify parameters by XINJEconfig software.

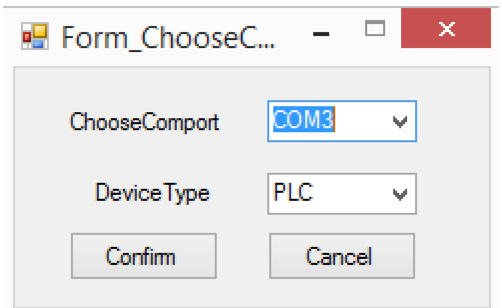
For example, two XD3-32T-E will X-Net communicate with each other by RS485.

Note: Please connect PLC with PC through USB cable before configuration. The USB cable is the same to HMI download cable. Please install the USB driver before using it.

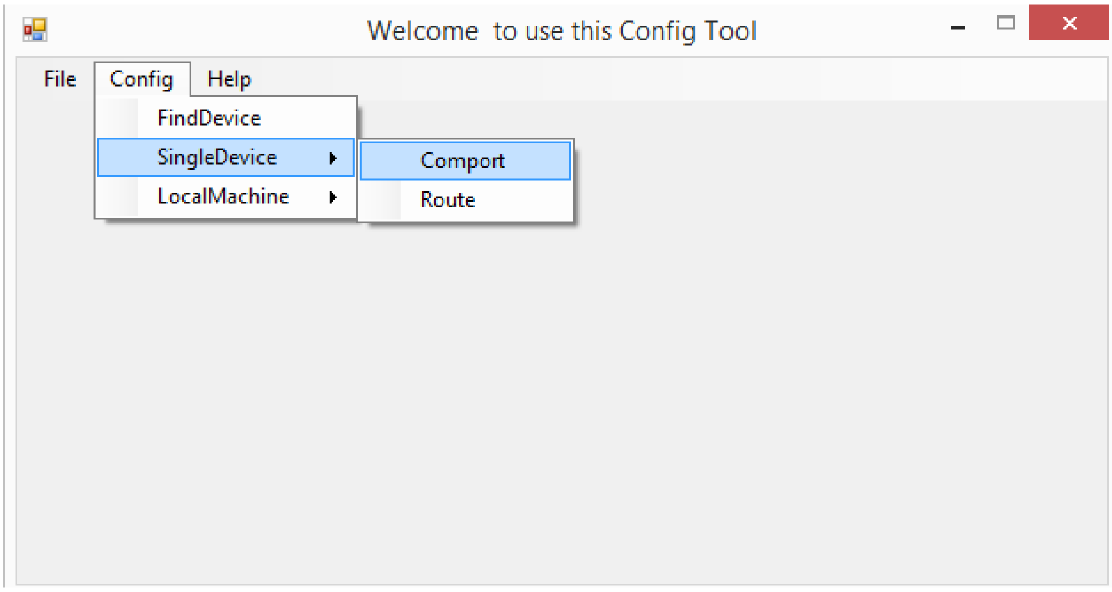
1. Open the XINJEconfig software



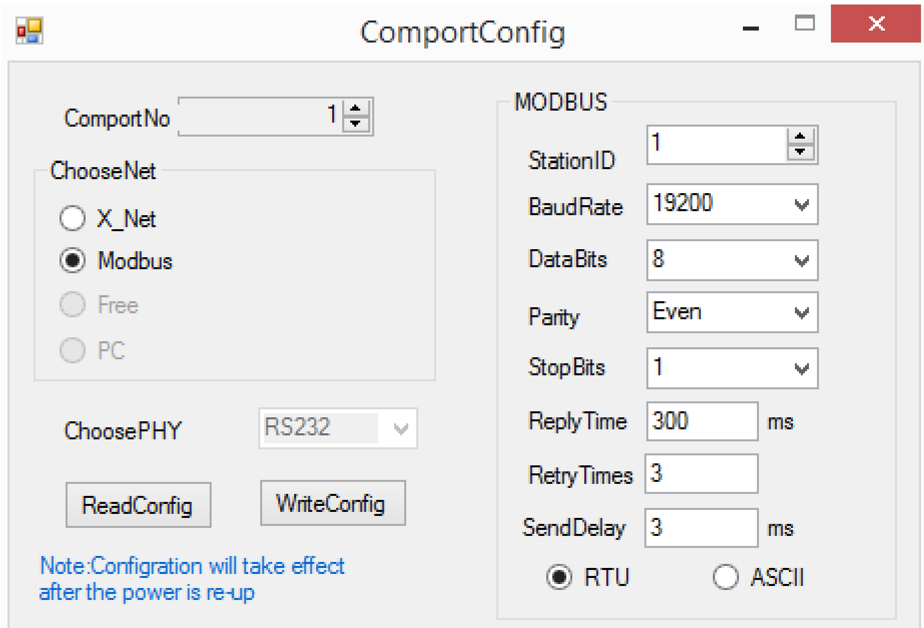
2. click config/find device to show below window, then choose com port and deivce type. Click confirm.



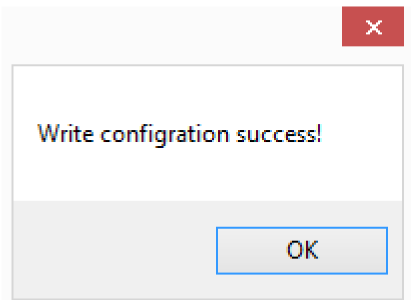
3. Click config/single device/comport



4. set the serial port parameters in below window



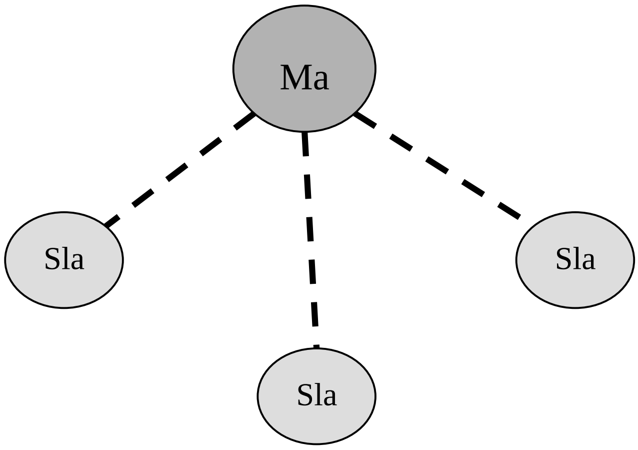
5. After setting, click writeconfig. It will show write success window.



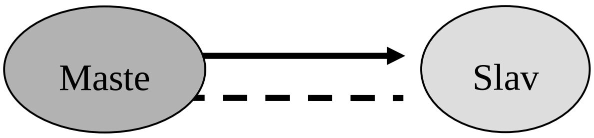
7-2．MODBUS communication

7-2-1．Function

XD3, XDM 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.



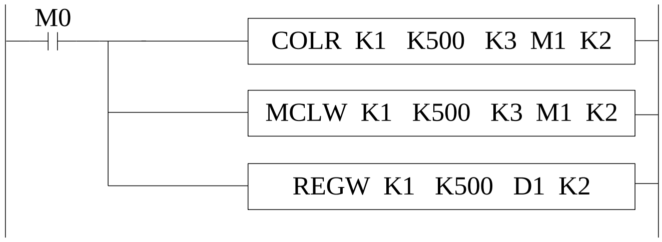
There is dotted line in the diagram. It means any PLC can be master station when all PLC in the network don't send data. As the PLC do not have unified clock standard, communication will fail when more than one PLC send data at one time. It is not recommended to use.

Note:

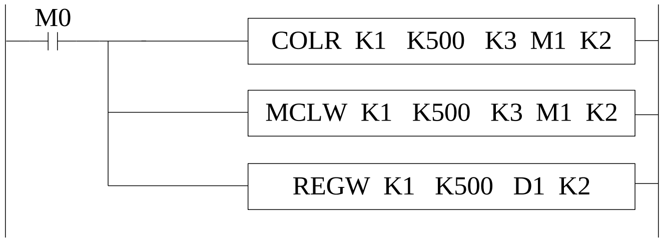
1. For XD3 series PLC, RS232 and RS485 only support half-duplex.
2. For XC series PLC, if master PLC send one data to slave PLC, and master PLC send data again before slave PLC receiving the last one completely, slave PLC end data error may occur; For XD3 series PLC, we solve this problem by adding waiting time before communication (COM1/COM2: SFD1337/SFD1347), which means the slave PLC will receive the next data only after some time the last data finished.

7-2-2． Changing of Modbus instruction

Modbus instruction handling mode has changed, 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: XD3 series PLC sequence block has cancelled Modbus communication instructions, which is replaced by the current Modbus instruction handling mode.

7-2-3． Communication address

The soft component's code in PLC corresponds with Modbus ID number, please see the following table:

XD3 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	64	5200~523F	20992~21055

		(#5 module)			
		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 (#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
		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~FD6143	6144	C4C0~DCBF	50368~56511
	SFD[※2]	SFD0~SFD1999	2000	E4C0~EC8F	58560~60559
	FS[※2]	FS0~FS47	48	F4C0~F4EF	62656~62703

XDM 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	0~4FFFF	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	64	6180~61BF	24960~25023

		(#3 module)			
		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	100	5290~52F3	21136~21235

		(#5 module)			
		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
	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	100	6614~6677	26132~26231

		(#14 module)			
		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~SFD4095	4096	E4C0~F4BF	58560~62655
	FS※2	FS0~FS255	256	F4C0~F5BF	62656~62911

Note:

1. *1 is power-off retentive range, *2 is flash range.
2. The address is usually for Modbus-RTU and Modbus-ASCII communication when PLC works as lower computer, and upper computer: configuration/screen/PLC......
3. If upper computer is PLC, then we write program according to Modbus-RTU or Modbus-ASCII protocol; if upper computer is configuration or touch screen, there will be two situations: 1. with xinje driver. E.g.: xinje touch screen/ Real bridge configuration can use PLC soft components directly (Y0/M0). 2. without xinje driver. Then users have to use below address to define variables after select Modbus-RTU or Modbus-ASCII protocol.
4. For Octonary I/O, calculate corresponding octonary I/O Modbus address.

7-2-4 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

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 high	CRC low

Format:

START	No input signal ≧ 10ms
Address（station）	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	INRR
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

Take 06H function code as example (single register write）, and introduce data format (other function code is similar to this):

E.g.: upper computer write data to PLC H0002 (D2).

RTU mode:

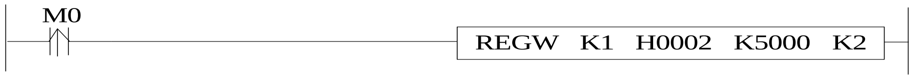
Asking format		Response format	
ID	01H	ID	01H
Function code	06H	Function code	06H
Register ID	00H	Register ID	00H
	02H		02H
Data content	13H	Data contents	13H
	88H		88H
CRC CHECK High	25H	CRC CHECK High	25H
CRC CHECK Low	5CH	CRC CHECK Low	5CH

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 7-2-2.
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 each other, 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

Modbus-ASCII data structure
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:

<table>
<tr><td>Start mark</td><td>Modbus no.</td><td>Function code</td><td>data</td><td>LRC</td><td colspan="2">End mark</td></tr>
<tr><td>1 character</td><td rowspan="2">2 characters</td><td rowspan="2">2 characters</td><td rowspan="2">0~252*2 characters</td><td rowspan="2">2 characters</td><td colspan="2">2 characters</td></tr>
<tr><td>0x3A</td><td>0x0D</td><td>0x0A</td></tr>
</table>

Format：

<table>
<tr><td>STX (3AH)</td><td>Start mark=3AH</td></tr>
<tr><td>Address code high bit</td><td rowspan="2">Communication position（no）：</td></tr>
<tr><td>Address code low bit</td></tr>
</table>

	Consist of 2 ASCII codes
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： Consist of 4 ASCII codes
Data length	
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 the similar to this）：

E.g.：upper computer write data to PLC H0002 (D2).

ASCII mode：

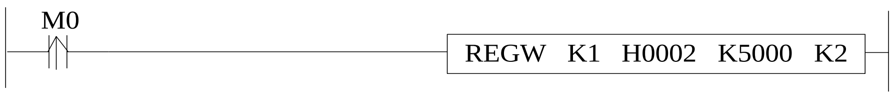
Start mark	3AH
ID	30H
	31H
Function code	30H
	36H
Register ID high byte	30H
	30H
Register ID low byte	30H

	32H
Data content high byte	31H
	33H
Data content low byte	38H
	38H
LRC	35H
	43H
End mark	0DH
	0AH

Description：

1. ID 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

7-2-5. Communication Instructions

Modbus instructions include coil read/write, register read/write; below, we describe these

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

Coil Read [COLR]

Instruction Summary

Read the specified station's specified coil status to the local PLC;

Coil read [COLR]			
16 bits instruction	COLR	32 bits instruction	-
Execution condition	Normally ON/OFF coil	Suitable models	XD3, XDM
Hardware requirement	-	Software Requirement	-

Operands

Operands	Function	Type
S1	Specify the remote communication station	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

Suitable soft components

Word

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

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

Bit

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

	S1·	S2·	S3·	D1·	D2·
X0 ↑ COLR	K1	K500	K3	M1	K2

Instruction to read coil, Modbus function code 01H.
Serial port: K1~K3.
Operands S3：K1~K984, the max coil quantity is 984.

Input coil read [INPR]

Summary

Write input coils status in specified station to the local station.

Input coil read[INPR]			
16 bits instruction	INPR	32 bits instruction	-
Execution condition	Normally ON/OFF, rising edge	Suitable models	XD3, XDM
Hardware requirement	-	Software requirement	-

Operands

Operands	Function	Type
S1	Specify remote communication 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

Suitable soft components

Word

Operands	System								Constant	Module	
	D*	FD	TD*	CD*	DX	DY	DM*	DS*	K/H	ID	QD
S1	●	●		●	●				●		
S2	●	●		●	●				●		
S3	●	●		●	●				●		
D2									K		

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

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

X0		S1·	S2·	S3·	D1·	D2·
↑	INPR	K1	K500	K3	M1	K2

Instruction to read input coil, Modbus function code is 02H.
Serial port：K1~K3.
Operand S3：K1~K984, max input coil number is 984.
When X0 is ON, then execute COLR or INPR instruction, set communication end flag after executing the instruction; When X0 is OFF, no operation. If communication errors, it will resend automatically. If the third time communication fails, then error flag forms. Users can check the relative registers to find the reason.

Coil Write [COLW]

Summary

Write input coils status in specified station to the local station.

Coil write [COLW]			
16 bits instruction	COLW	32 bits instruction	-
Execution Condition	Normally ON/OFF、rising edge	Suitable Models	XD3, XDM
Hardware Requirement	-	Software Requirement	-

Operands

Operands	Function	Type
D1	Specify remote communication number	16 bits，BIN
D2	Specify remote coil start address number	16 bits，BIN
S1	Specify start address number of local send coil	bit
S2	Specify serial port number	16 bits，BIN

Suitable soft components

Word

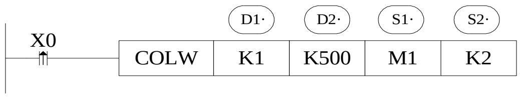
Operands	System								Constant	Module	
	D*	FD	TD*	CD*	DX	DY	DM*	DS*	K/H	ID	QD
D1	●	●		●	●				●		
D2	●	●		●	●				●		
S2									K		

Bit

Operand	System						
	X	Y	M*	S*	T*	C*	Dn.m
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 and HS; T includes T and HT; C includes C and HC.

Function



Instruction to write coil, Modbus function code is 05H.
Serial port: K1~K3.

Multi-coils write [MCLW]

Summary

Write input coils status in the local station to the specified station.

Multi-coils write [MCLW]			
16 bits instruction	MCLW	32 bits instruction	-
Execution Condition	Normally ON/OFF, rising edge	Suitable models	XD3, XDM
Hardware Requirement	-	Software Requirement	-

Operands

Operands	Function	Type
D1	Specify remote communication number	16 bits，BIN
D2	Specify remote coil start address number	16 bits，BIN
D3	Specify coil number	16 bits，BIN

S1	Specify start address number of local send coils	bit
S2	Specify serial port number	16 bits，BIN

Suitable soft components

Word

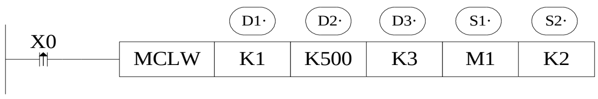
Operands	System								Constant	Module	
	D*	FD	TD*	CD*	DX	DY	DM*	DS*	K/H	ID	QD
D1	●	●	●	●					●		
D2	●	●	●	●					●		
D3	●	●	●	●					●		
S2									K		

Bit

Operands	System						
	X	Y	M*	S*	T*	C*	Dn.m
S1	●	●	●	●	●	●	

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



Instruction to write multi-coils, Modbus function code is 0FH.
Serial port: K1~K3.
Operand D3, the max coils max quantity is 952.
When X0 is ON, execute COLW or MCLW instruction, set communication end flag after finishing the instruction; When X0 is OFF, no operation. If communication errors, it will resend automatically. The third time communication fails, then error flag forms. Users can check the relative registers to find the error reason.

Register read [REGR]

Summary

Write registers content in the specified station to the local station.

Register read[REGR]			
16 bits instruction	REGR	32 bits instruction	-
Execution Condition	Normally ON/OFF, rising edge	Suitable models	XD3, XDM
Hardware Requirement	-	Software Requirement	-

Operands

Operands	Function	Type
S1	Specify remote communication number	16 bits，BIN
S2	Specify remote register start address number	16 bits，BIN
S3	Specify register number	16 bits，BIN
D1	Specify start address number of local receipt register	16 bits，BIN
D2	Specify serial port number	16 bits，BIN

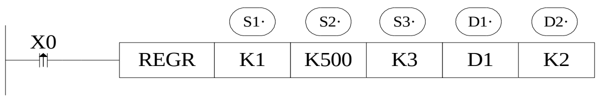
Suitable soft components

Word

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

Function



Instruction to read register, Modbus function code is 03H.

Serial port : K1~K3

Operand S3 and max register number is 61.

Input register read [INRR]

Summary

Write input register content in specified number to the local register.

Input register read [INRR]			
16 bits instruction	INRR	32 bits instruction	-
Execution Condition	Normally ON/OFF, rising edge	Suitable models	XD3, XDM
Hardware Requirement	-	Software Requirement	-

Operands

Operands	Function	Type
S1	Specify remote communication number	16 bits，BIN
S2	Specify remote register start address number	16 bits，BIN
S3	Specify coil number	16 bits，BIN

D1	Specify start address number of local receipt register	16 bits， BIN
D2	Specify serial port number	16 bits， BIN

suitable soft components

Word

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

Function

X0		S1·	S2·	S3·	D1·	D2·
⤊	INRR	K1	K500	K3	D1	K2

Instruction to read input register, Modbus function code is 04H.
Serial port： K1~K3.
Operand S3, the max input register number is 61.
When X0 is ON, execute REGR or INRR instruction, set communication end flag after executing the instruction; When X0 is OFF, no operation. If communication errors, it will resend automatically. If the forth communication fails, then communication error flag forms. Users can check relative registers to find the reasons.

Register write [REGW]

summary

Write register content in local station to the specified station.

Register write[REGW]			
16 bits instruction	REGW	32 bits instruction	-
Execution Condition	Normally ON/OFF, rising edge	Suitable models	XD3, XDM
Hardware Requirement	-	Software Requirement	-

Operands

Operands	Function	Type
D1	Specify remote communication number	16 bits， BIN
D2	Specify remote register start address number	16 bits， BIN
S1	Specify start address number of local send register	16 bits， BIN

S2	Specify serial port number	16 bits，BIN

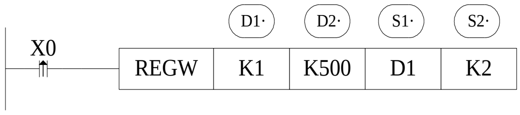
suitable soft components

Word

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

Function



Instruction to write register, Modbus function code is 06H.
Serial port：K1~K3.

Multi-register write [MRGW]

Summary

Write register content in local station to the specified register.

Multi-register write [MRGW]			
16 bits instruction	MRGW	32 bits instruction	-
Execution Condition	Normally ON/OFF, rising edge	Suitable models	XD3, XDM
Hardware Requirement	-	Software Requirement	-

Operands

Operands	Function	Type
D1	Specify remote communication number	16 bits，BIN
D2	Specify remote register start address number	16 bits，BIN
D3	Specify register number	16 bits，BIN
S1	Specify start address number of local send registers	16 bits，BIN
S2	Specify serial port number	16 bits，BIN

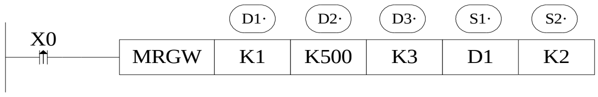
suitable soft components

Word

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

Function



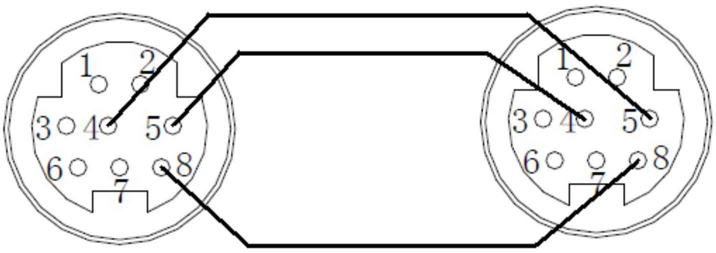
Instruction to write multi-registers, Modbus function code is 10H.
Serial port：K1~K3.
Operand D3, the max register quantity is 59.
When X0 is ON, execute REGW or MRGW instruction, set communication end flag after executing the instruction; When X0 is OFF, no operation. If communication fails, it will resend automatically. If the third communication fails, communication error flag forms. Users can check relative registers to find the error reason.

7-2-6．Communication application

Wiring method
There are two wiring methods:
232 wiring methods

COM2*1 diagram

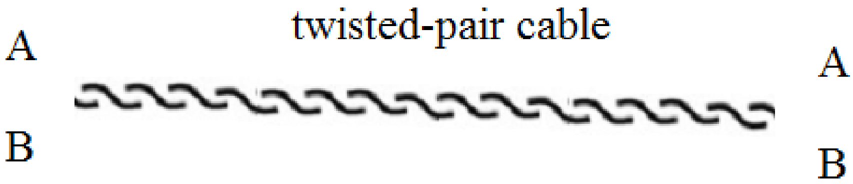
4：RxD
5：TxD
8：GND



Mini Din 8 Pins port

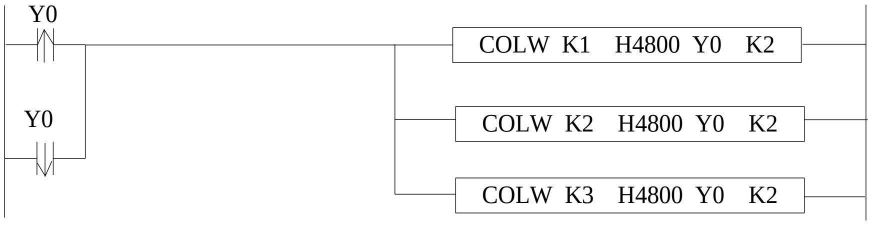
Note:
1. COM2 with *1 only show the RS232 pins. The RS485 pins are external terminal（A、B）.
2. XD3 series PLC，RS232 do not support full-duplex, so it can only communicate in single direction.
3. RS232 communication distance is short (about 13m); RS485 is suitable for longer distance.

485 wiring methods



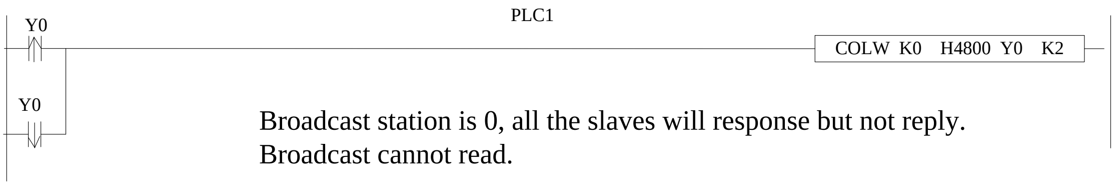
Connect all A terminals, connect all B terminals. A is RS485+, B is RS485-.
Application：One xinje XD3 series PLC control 3 PLCs, slave PLCs follow the master's action. (Master PLC Y0 ON, then slave PLC Y0 ON; Master PLC Y0 OFF, then slave PLC Y0 OFF) Precondition: on-off of Y0 makes communication have 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.

7-3. X-NET communication

7-3-1. X-NET introduction

X-NET protocol stack which is developed by XINJE Company committed to the automation production network. X-NET supports different physical media and networks. X_NET refers to OSI seven layers network protocol, it defines the service and function of each layer and make the network more efficient and simple.

The physical protocol supported by XD3 and XDM including RS485, RS232, OC (optical circle), USB_PHY. The network protocol includes TBN (Token Bus Net), OMMS (one master multi slave net). TBN is fit for embedded equipment network, the advantage includes real-time and multi-master. OMMS is fit for master-slave network, the advantage is wide applicability, but the response time is longer than TBN.

	TBN	OMMS
RS485	√	√
RS232		√
OC	√	√
USB_PHY		√

TBN belongs to industry fieldbus. It is token structure, any node in the network has access rights (token), any node who got the token can send message to other node. It doesn't have master and slave. It makes the data access efficient and flexible.

OMMS focus on the connection between upper device and embedded equipment. The connection between equipments is permitted. The PLC connection with programming software and network configuration software uses OMMS protocol.

The MCP (motion control protocol) based on OMMS makes the multi-axis controlling simple. 8-axis 2ms control period can meet most of motion control needs. It needs one PLC and 8 servo drives to make the project, no need special communication module.

7-3-2. X-NET wiring

Please use serial connection for TBN protocol running on RS485. As the star connection has low driving ability and short communication distance, strong signal reflection, higher communication error rate. It is not recommended to use star connection.

TBN_RS485 baud rate range is 600bit~3Mbit. The communication speed and medium are related to the local environment. As the TBN_RS485 using electrical signal to transfer data, there are some requirements for the communication distance. Please refer to the following table.

Please use copper shielded wire up 0.3mm^2 for TBN_RS485 fieldbus.

The relationship between TBN_RS485 filedbus cable length and baud rate:

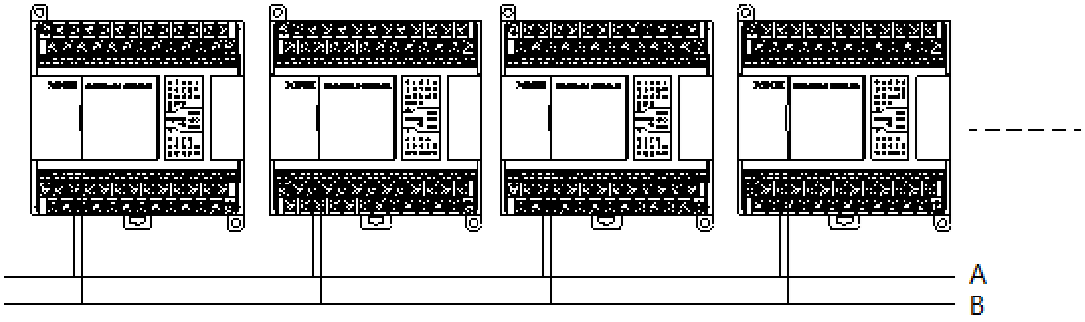
Baud rate	Wire length
9.6-187.5Kbit/s	1000m
500Kbit/s	400m
1.5Mbit/s	200m
3Mbit/s	100m

There are three kinds of connection methods for PLC.

1. PLC RS485 port on the main unit
2. PLC expansion BD board XD-RS485-BD
3. PLC expansion ED board

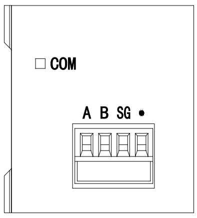
Connection method 1:

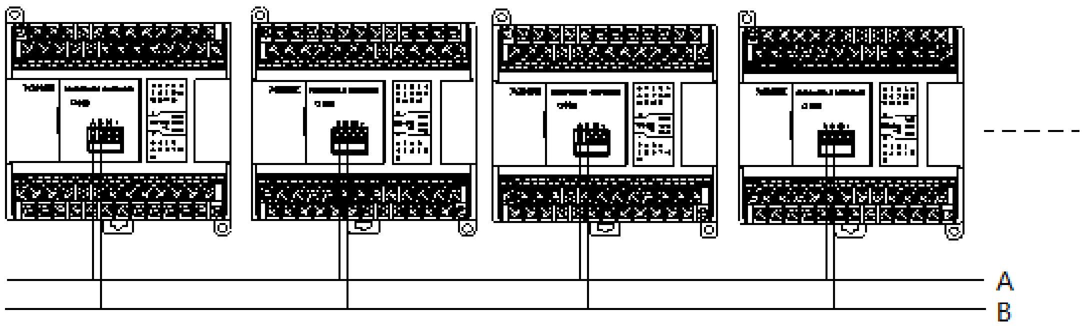
Terminal A is RS485+, terminal B is RS485-. Connnect A to A, B to B.



Note: RS485 port can communication through Modbus_RTU or X-NET. Please choose the mode in XINJEConfig software.

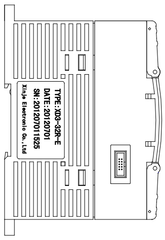
Connection method 2:
There are four terminals including A, B, SG(signal ground), FG(shielded ground) on expansion BD board XD-RS485-BD. Normally, X-NET needs to connect terminal A and B, the shielded layer connects to FG. If there is requirements for anti-interference, such as motion control fieldbus, it needs to connect terminal A, B, SG.
XD-RS485-BD has switch to set terminal resistor. The default setting is switch OFF(left) without terminal resistor. If XD-RS485-BD is at the head or end of the filedbus, it needs to add 120ohm terminal resistor at both end, the switch must be ON (right).





Note: RS485 port of XD-RS485-BD can communication through Modbus_RTU or X-NET. Please choose the mode in XINJEConfig software.

Connection method 3:
Connect through the expansion ED board on the left side of PLC.



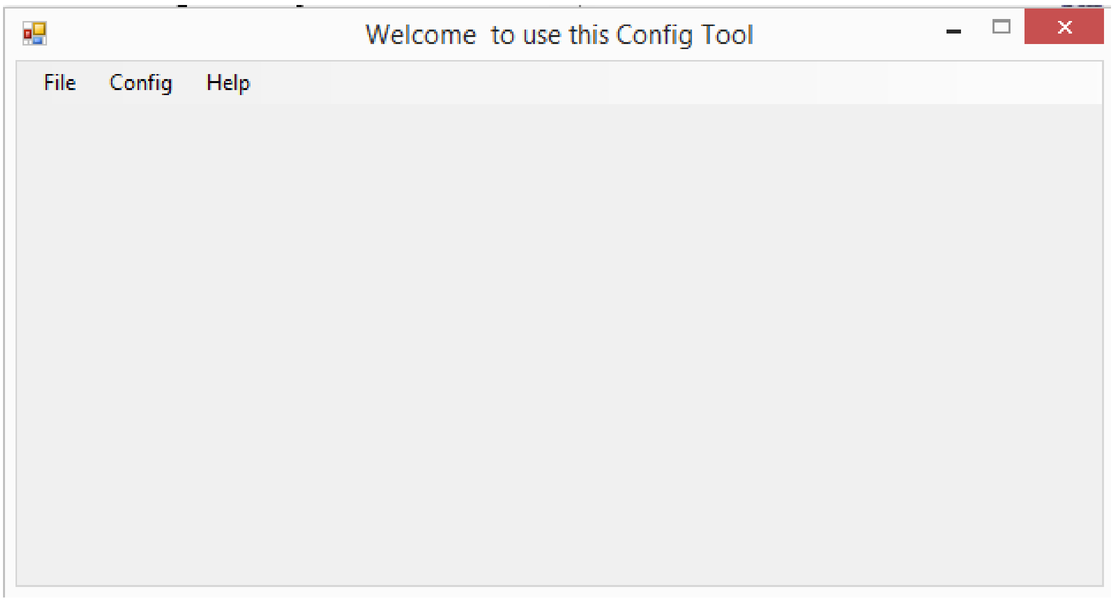
Note: the ED board is in developing.

When TBN runs on OC, the station connection wire is optical fiber. The last station sending terminal connects to the receiving terminal of the first station to make the loop network. The optical signal has strong anti-interference ability. But the optical fiber is easy to broken and complicated to make the pigtail, this limits the developing of optical fiber. TBN_OC supports 600bit~3Mbit baud rate.
As the RS485 device driving ability and conversion speed of photoelectric module, the node quantity in the TBN network must be less than 32.

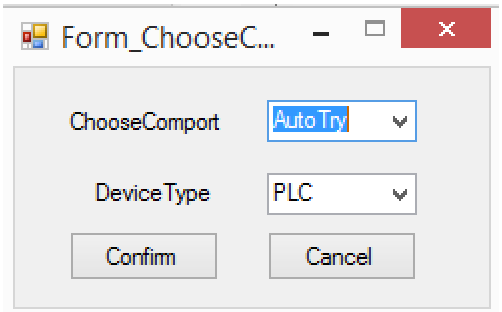
7-3-3. XINJEConfig software

The using steps of XINJEConfig:
For example, two XD3-32T-E communicate with each other through RS485 port based on X-NET protocol.
Note: Please connect PLC with PC through USB cable before matching the PLC by software. Make sure to install the USB driver before using.

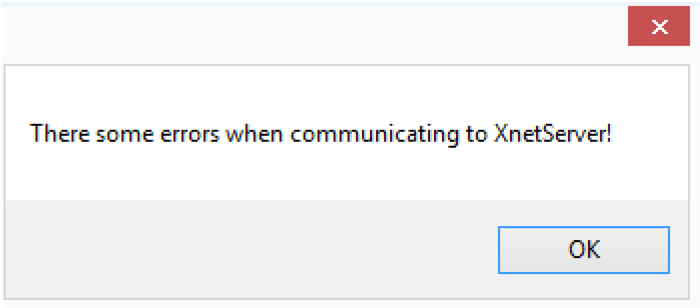
Open the XINJEConfig software, click config/find device.



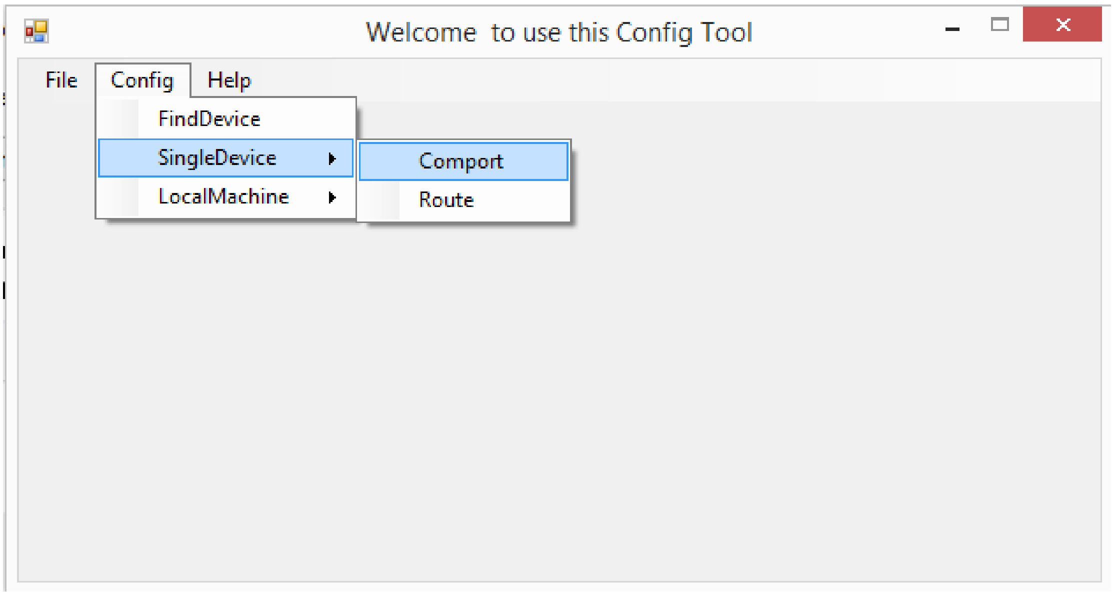
Choose the com port connected PLC, device type is PLC.



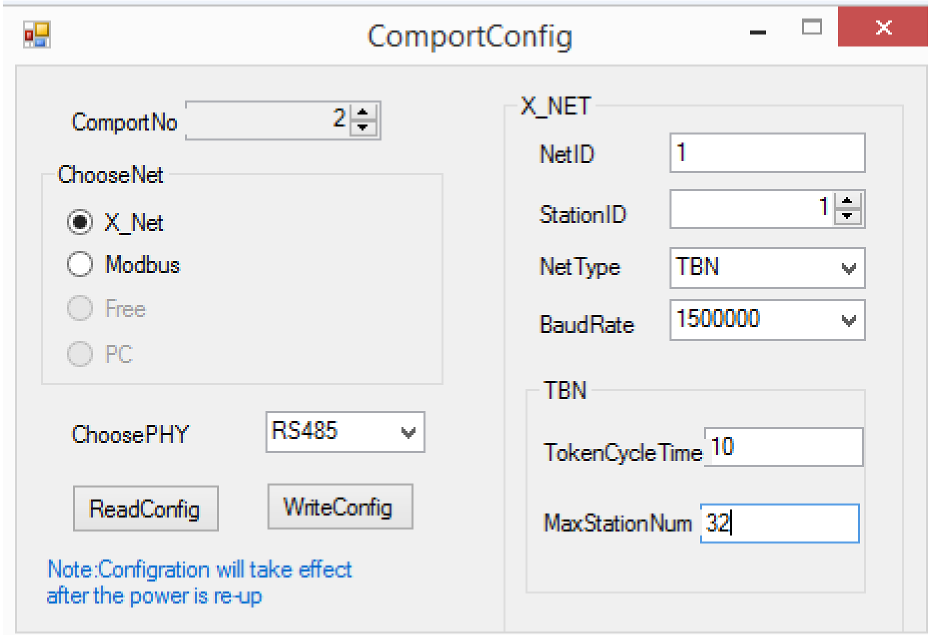
If the communication between PLC and PC is error, there will show below error. Please restart the software and configure again.



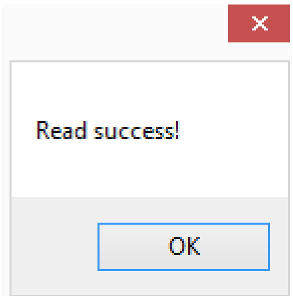
Click ok to back to main window. Then click config/single deivce/com port.



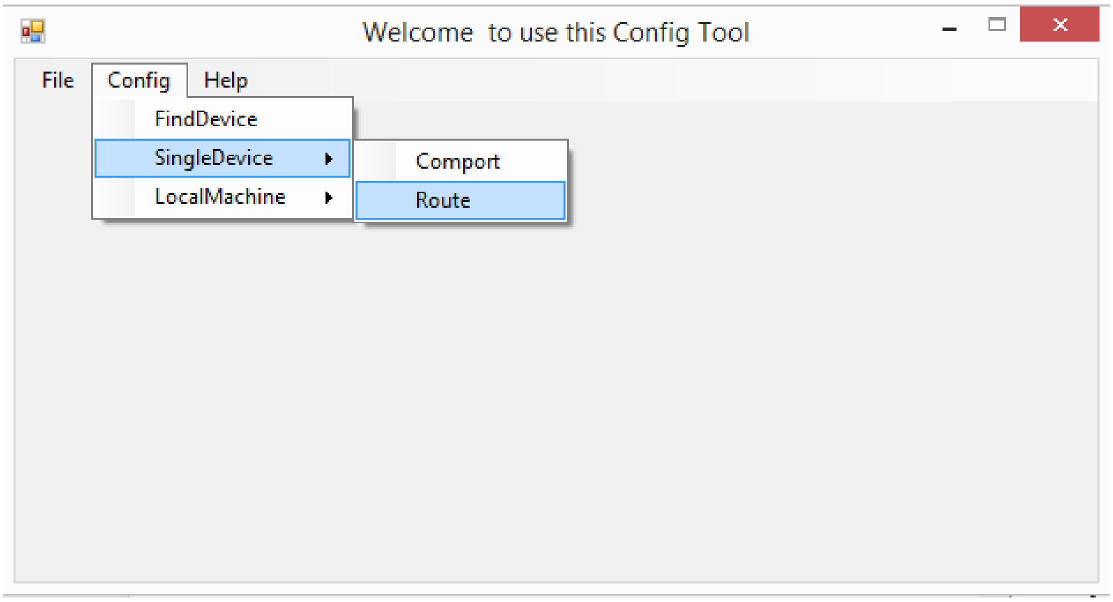
As the PLC RS485 port is serial port2, here we choose no.2 for comportNo. The protocol is X-NET. The physical layer is RS485.



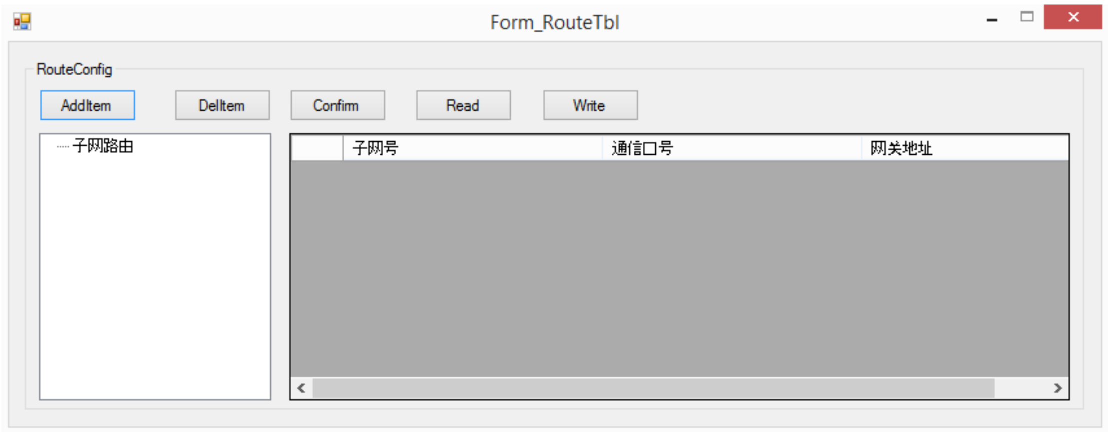
NetID: the network number of the two PLC. The device net ID in the same network must be the same. Here we set it to 1.
StationID: each PLC station number in the network. The two PLC station number is 1 and 2.
Net type:
TBN--- PLC communicate with PLC
TBN or HDN--- PLC communicate with HMI
HDN--- PLC communicate with servo.
Baud rate: here we set it to 1.5M
TokenCycle time: the time of each station in the network cycle once, the unit is ms. Here we have two PLC, so the time set to 10ms.
Max station number: the max device numbers in the network.
Click “write config”, it will show read success window.



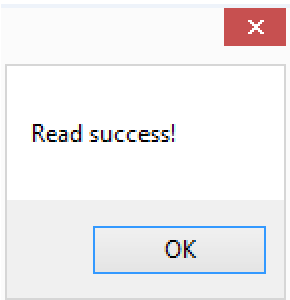
Click ok to back to the main window. Cut off the PLC power and power on again to make the setting effective.
Then click config/single device/route.



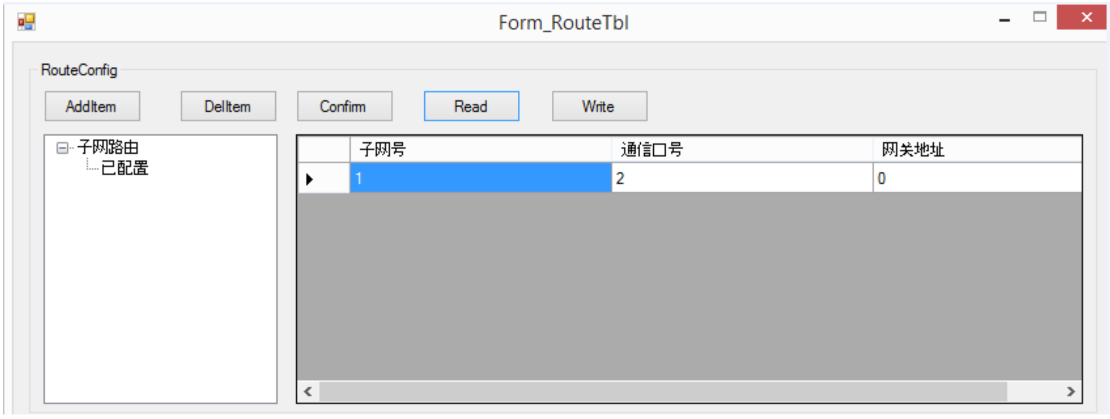
It shows the Form_RouteTb1 window.



Click “read”, it shows read success.



Click ok, the Form_RouteTb1 window shows the subnet routing.



The subnet ID is the same to the netID. The communication port is the PLC physical terminal, RS485 port number is 2. Gateway address: the default setting is 0.
Click “write”, it will show write success. Then click ok. Please cut off the PLC power and power on again to make the setting effective.
Note:
1. After changing the device, it needs to configure again.
2. If it shows the error “there are some errors when communicating to XnetServer”, please restart the software and configure again.

7-3-4. Communication address

The X-NET address of PLC soft components:
XD3 soft component address:

Soft component		Type code (decimal)	Numbers (decimal)
X	X0	1	64
	X10000 (octal)		640
	X20000 (octal)		64
Y	Y0	2	64
	Y10000 (octal)		640
	Y20000 (octal)		64
M		3	8000
S		4	1024
T		5	576
C		6	576
ET		7	32
HM		8	960
HS		9	128
HT		10	96
HC		11	96
HSC		12	32
SM		13	2048
SEM		18	32
D		128	8000
TD		129	576

CD		130	576
SD		131	2048
ETD		133	32
ID	ID0	134	100
	ID10000		1000
	ID20000		100
QD	QD0	135	100
	QD10000		1000
	QD20000		100
HD		136	1000
HTD		137	96
HCD		138	96
HSCD		139	32
HSD		140	500
FD		141	6144
SFD		142	2000

XDM soft component address

Soft component		**Type code (decimal)**	**Numbers (decimal)**
X	X0	1	64
	X10000 (octal)		1024
	X20000 (octal)		64
Y	Y0	2	64
	Y10000 (octal)		1024
	Y20000 (octal)		64
M		3	20480
S		4	8000
T		5	4096
C		6	4096
ET		7	40
HM		8	6144
HS		9	1000
HT		10	1024
HC		11	1024
HSC		12	40
SM		13	4096
SEM		18	158
D		128	20480
TD		129	4096
CD		130	4096
SD		131	4096
ETD		133	40
ID	ID0	134	100
	ID10000		1600
	ID20000		500
QD	QD0	135	100

	QD10000		1600
	QD20000		500
HD		136	6144
HTD		137	1024
HCD		138	1024
HSCD		139	40
HSD		140	1024
FD		141	8192
SFD		142	4096

Note:

1. The input and output is octal, please calculate the input and output X-NET address as octal.
2. make sure there is no cross-border access for the target object.

7-3-5. Communication instruction

X-NET instructions include bit read and write, register read and write.

The communication instruction operand explanation:

1. Target net ID: all the devices construct the communication network. The ID of the network is the target net ID.
For example: 5 PLCs will communicate with each other in the network through X-NET protocol. User can define the net ID for this network.

2. Target station number: the station number of target device in the network.
For example: PLC connects to 3 frequency inverters, the frequency inverter station number is 1, 2, 3, and PLC station number is 4. The PLC will read the parameters of three inverters. Station number 1, 2, 3 are the target station number of PLC.

3. Target object type: the target device is coil or register.
For example:
The target device is coil X, the target object type is K1
The target device is coil Y, the target object type is K2
The target device is coil M, the target object type is K3
The target device is coil HM, the target object type is K8
The target device is register D, the target object type is K128
The target device is register HD, the target object type is K136

4. Target object number: the target device coil or register address in the network
For example: write register D0, write the D0 value to target address

5. Access object numbers: the target station numbers need to access
For example: PLC needs to read the frequency inverter output frequency, output current and bu voltage, the access object numbers are 3.

6. Local object address: the local coil or register address
For example: PLC register D0 value transfers to frequency inverter address H2103. So the local object address is D0, the target object address is H2103.

Read bit [BIT_READ]

1. Instruction explanation
Read the target coil to local coil.

Read bit [BIT_READ]			
Execution conditoin	Edge trigger	Suitable model	XD3, XDM
Hardware	V3.2 and up	Software	V3.2 and up

2. Operand

Operand	Function	Type
S1	Target net ID	16 bits constant or single word register
S2	Target station number	16 bits constant or single word register
S3	Target object type (refer to chapter 7-3-4)	16 bits constant or single word register
S4	Target object address (refer to chapter 7-3-4)	32 bits constant or double words register
S5	Access object numbers	16 bits constant or single word register
D1	Local object address	Local coil

3. Suitable soft component

Word

Operand	System								Constant	Module	
	D*	FD	TD*	CD*	DX	DY	DM*	DS*	K/H	ID	QD
S1	●	●		●	●				●		
S2	●	●		●	●				●		
S3	●	●		●	●				●		
S4	●	●		●	●				●		
S5	●	●		●	●				●		

Bit

Operand	System						
	X	Y	M*	S*	T*	C*	Dn.m
D1	●	●	●	●	●	●	

*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; S includes S, HS; T includes T, HT; C includes C, HC.

Intruction

X0		S1·	S2·	S3·	S4·	S5·	D1.
⊣↑⊢	BIT_READ	K1	K2	K3	K0	K8	Y0

Function: read the station no.2 coil M0~M7 to local address Y0~Y7 in ID1 network.

Write bit [BIT_WRITE]

1. Intruction explanation

Write the local coil value to target coil.

Write bit [BIT_WRITE]			
Execution conditoin	Edge trigger	Suitable model	XD3, XDM
Hardware	V3.2 and up	Software	V3.2 and up

2. Operand

Operand	Function	Type
S1	Target net ID	16 bits constant or single word register
S2	Target station number	16 bits constant or single word register
S3	Target object type (refer to chapter 7-3-4)	16 bits constant or single word register
S4	Target object address (refer to chapter 7-3-4)	32 bits constant or double words register
S5	Access object numbers	16 bits constant or single word register
S6	Local object address	Local coil

3. Suitable soft component

Word

Operand	System								Constant	Module	
	D*	FD	TD*	CD*	DX	DY	DM*	DS*	K/H	ID	QD
S1	●	●		●	●				●		
S2	●	●		●	●				●		
S3	●	●		●	●				●		
S4	●	●		●	●				●		
S5	●	●		●	●				●		

Bit

Operand	System						
	X	Y	M*	S*	T*	C*	Dn.m
S6	●	●	●	●	●	●	

*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; S includes S, HS; T includes T, HT; C includes C, HC.

Intruction

X0		S1·	S2·	S3·	S4·	S5·	S6.
↑	BIT_WRITE	K1	K3	K2	K0	K8	M0

Function: write the local coil M0~M7 to station no.3 Y0~Y7 in ID1 network.

Read register [REG_READ]

1. Instruction explanation

Read the target register to local register.

Read register [REG_READ]			
Execution conditoin	Edge trigger	Suitable model	XD3, XDM
Hardware	V3.2 and up	Software	V3.2 and up

2. Operand

Operand	Function	Type
S1	Target net ID	16 bits constant or single word register
S2	Target station number	16 bits constant or single word register
S3	Target object type (refer to chapter 7-3-4)	16 bits constant or single word register
S4	Target object address (refer to chapter 7-3-4)	32 bits constant or double words register
S5	Access object numbers	16 bits constant or single word register
D1	Local object address	Local register

3. Suitable soft component

Word

Operand	System								Constant	Module	
	D*	FD	TD*	CD*	DX	DY	DM*	DS*	K/H	ID	QD
S1	●	●		●	●				●		
S2	●	●		●	●				●		
S3	●	●		●	●				●		
S4	●	●		●	●				●		
S5	●	●		●	●				●		
D1	●	●		●	●						

*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; S includes S, HS; T includes T, HT; C includes C, HC.

Intruction

X0		S1·	S2·	S3·	S4·	S5·	D1.
	REG_READ	K1	K2	K128	K0	K10	D0

Function: read the station no.2 register D0~D9 to local address D0~D9 in ID1 network.

Write register [REG_WRITE]

1. Instruction explanation

Write the local register to target register.

Write register [REG_WRITE]			
Execution conditoin	Edge trigger	Suitable model	XD3, XDM
Hardware	V3.2 and up	Software	V3.2 and up

2. Operand

Operand	Function	Type
S1	Target net ID	16 bits constant or single word register
S2	Target station number	16 bits constant or single word register
S3	Target object type (refer to chapter 7-3-4)	16 bits constant or single word register
S4	Target object address (refer to chapter 7-3-4)	32 bits constant or double words register
S5	Access object numbers	16 bits constant or single word register
S6	Local object address	Local register

3. Suitable soft component

Word

Operand	System								Constant	Module	
	D*	FD	TD*	CD*	DX	DY	DM*	DS*	K/H	ID	QD
S1	●	●		●	●				●		
S2	●	●		●	●				●		
S3	●	●		●	●				●		
S4	●	●		●	●				●		
S5	●	●		●	●				●		
S6	●	●		●	●						

*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; S includes S, HS; T includes T, HT; C includes C, HC.

Intruction

		S1·	S2·	S3·	S4·	S5·	S6.
X0	REG_WRITE	K2	K5	K128	K0	K10	D0

Function: write the local register D0~D9 to station no.5 D0~D9 in ID2 network.

7-3-6. X-NET Communication application

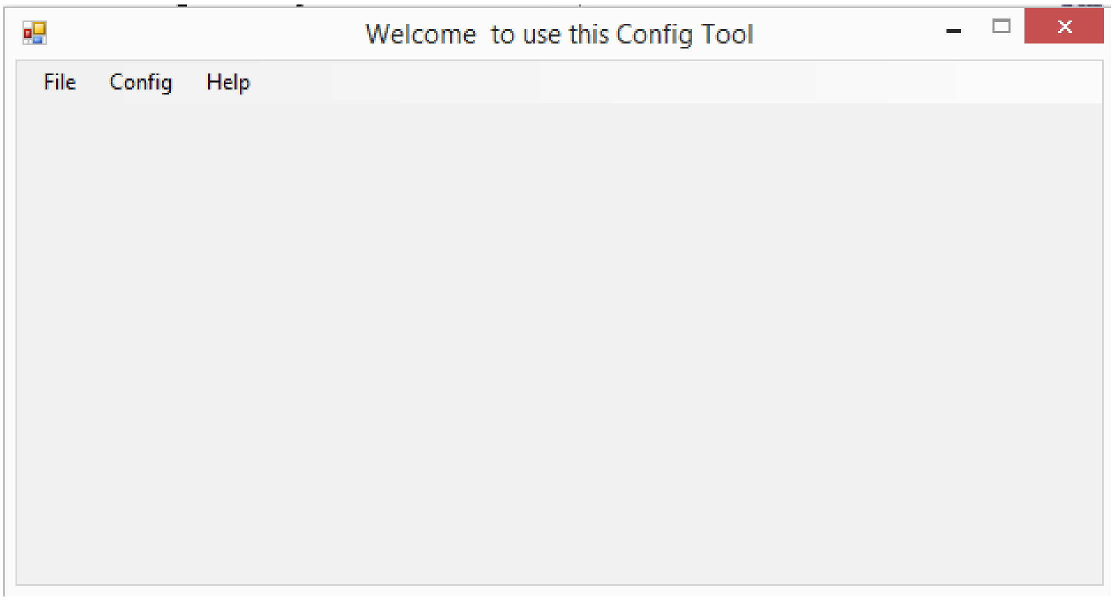
There are two XD3 sereis PLC. They communicate with each other through XD-RS485-BD based on X-NET protocol.

The purposes:

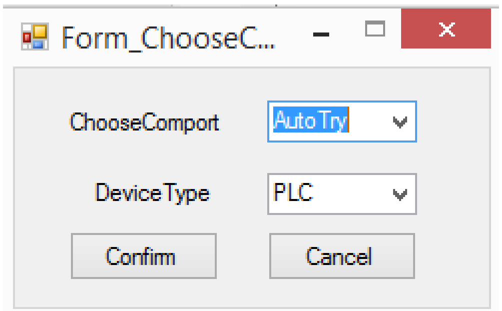
1. Write PLC A register D0~D9 to PLC B D0~D9
2. Write PLC A coil M0~M9 to PLC B M0~M9
3. Read PLC B register D0~D9 to PLC A D10~D19
4. Read PLC B coil M0~M9 to PLC A M10~M19

Step 1: wiring
Connect terminal A to A, terminal B to B of the XD-RS485-BD.

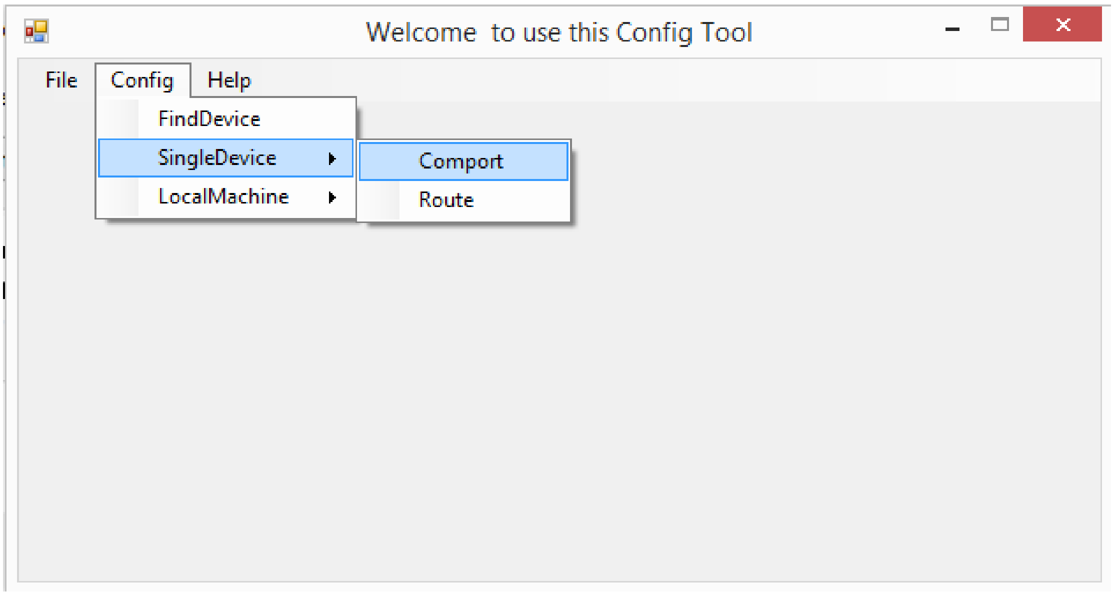
Step 2: RS485 port setting
1. Connect PLC A with PC through USB cable. Open the XINJEConfig software.



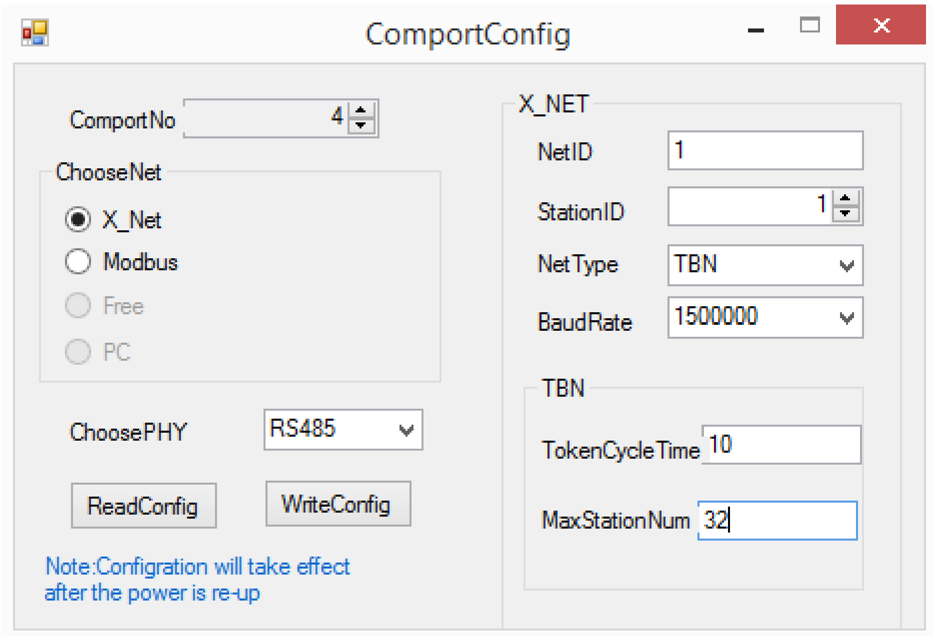
Choose the com port connected PLC, device type is PLC.



Click ok to back to main window. Then click config/single deivce/com port.



As the PLC expansion XD-RS485-BD port is serial port 4, here we choose no.4 for comportNo. The protocol is X-NET. The physical layer is RS485.



NetID: the network number of the two PLC. The device net ID in the same network must be the same. Here we set it to 1.

StationID: each PLC station number in the network. The two PLC station number is 1 and 2.

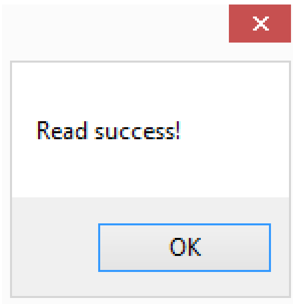
Net type: it is PLC communication, please choose TBN.

Baud rate: here we set it to 1.5M

TokenCycle time: the time of each station in the network cycle once, the unit is ms. Here we have two PLC, so the time set to 10ms.

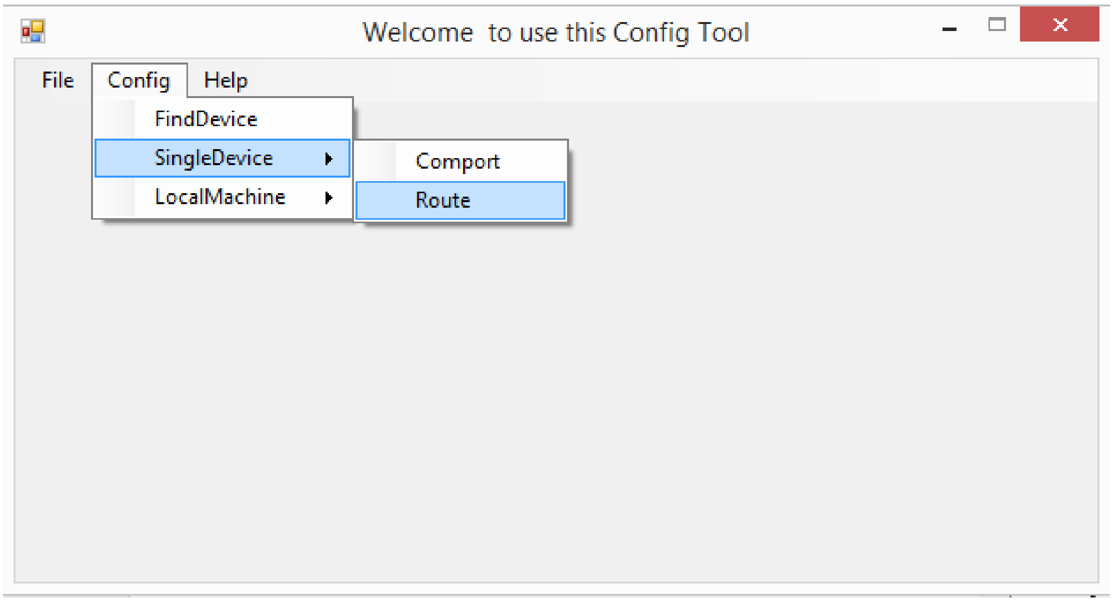
Max station number: the max device numbers in the network. As the X-NET communication max station no. is 32, so it is set to 32.

Click “write config”, it will show read success window.

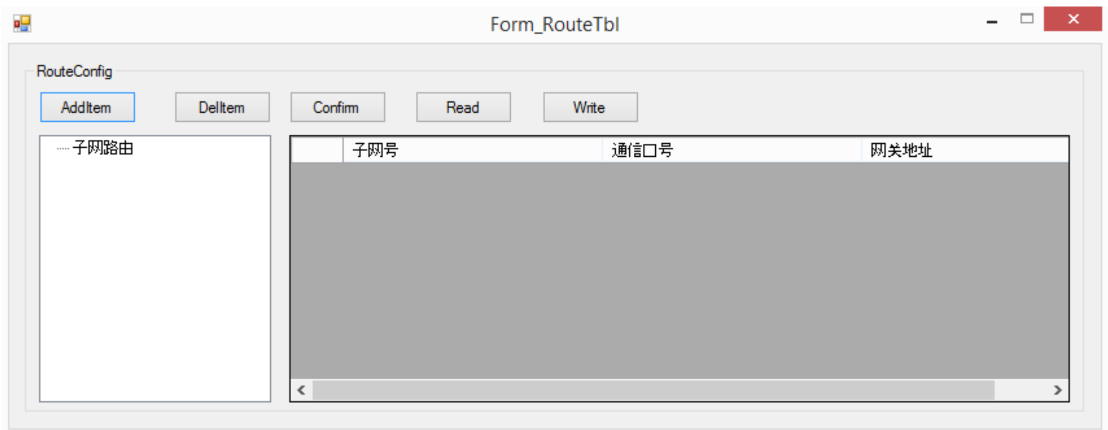


Click ok to back to the main window. Cut off the PLC power and power on again to make the setting effective.

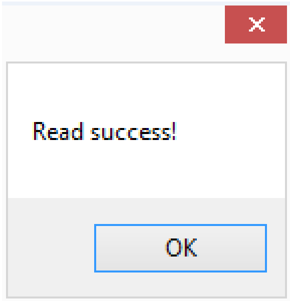
Then click config/single device/route.



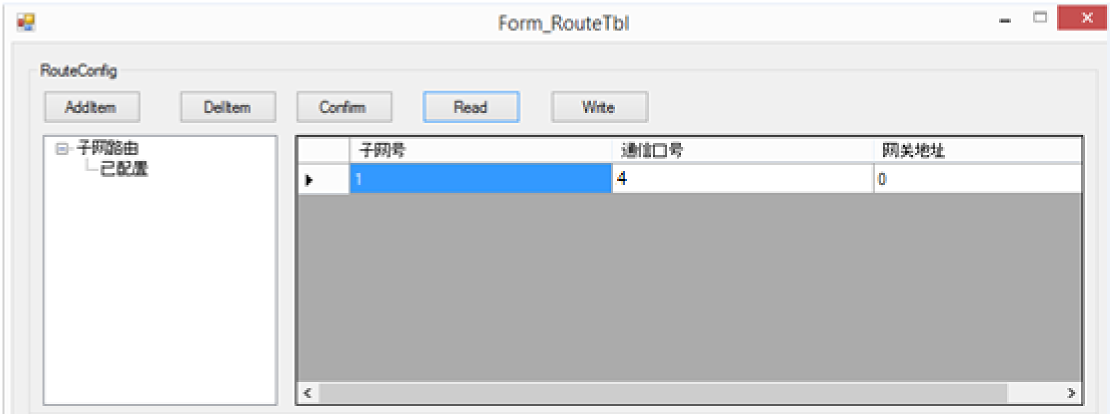
It shows the Form_RouteTb1 window.



Click “read”, it shows read success.



Click ok, the Form_RouteTb1 window shows the subnet routing.

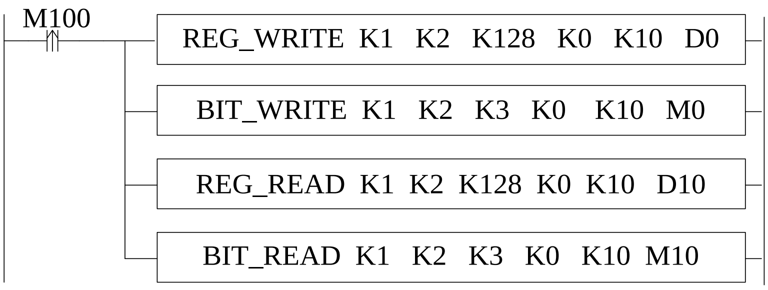


The subnet ID is the same to the netID. The communication port is the PLC physical terminal, RS485 port number is 4. Gateway address: the default setting is 0.

Click "write", it will show write success. Then click ok. Please cut off the PLC power and power on again to make the setting effective.

PLC B RS485 configuration is the same to PLC A.

Step 3: the program



8 PID Control Function

In this chapter, we mainly introduce the applications of PID instructions for XD, XDM series, including: call the instructions, set the parameters, items to notice, sample programs etc.

8-1. PID Introduction

PID instruction and auto tune function are added into XD 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.
XD3 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.

8-2. Instruction Form

Brief Introduction of the Instructions
Execute PID control instructions with the data in specified registers.

PID control [PID]			
16 bits instruction	PID	32 bits instruction	-
Executing condition	Normally ON/normally closed coil trigger	Suitable models	XD3, XDM
Hardware requirement		Software requirement	V3.2

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 bits, BIN; bit

Suitable soft components

Word

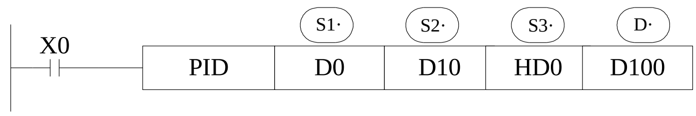
Operands	System								Constant	Module	
	D*	FD	TD*	CD*	DX	DY	DM*	DS*	K/H	ID	QD
S1	●	●							●		
S2	●	●									
S3	●	●									
D	●	●									

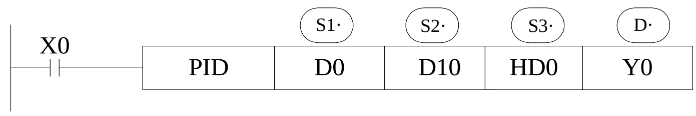
Bit

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

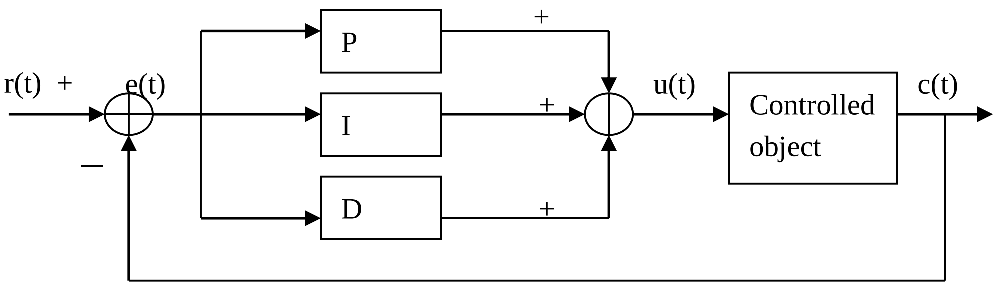
*Note: D includes D, HD; TD includes TD, HTD; CD includes CD, HCD, HSCD, HSD; DM includes DM, DHM; DS includes DS, DHS.
M includes M, HM, SM; S includes S and HS; T includes T and HT; C includes C and HC.

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 ∫ 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.

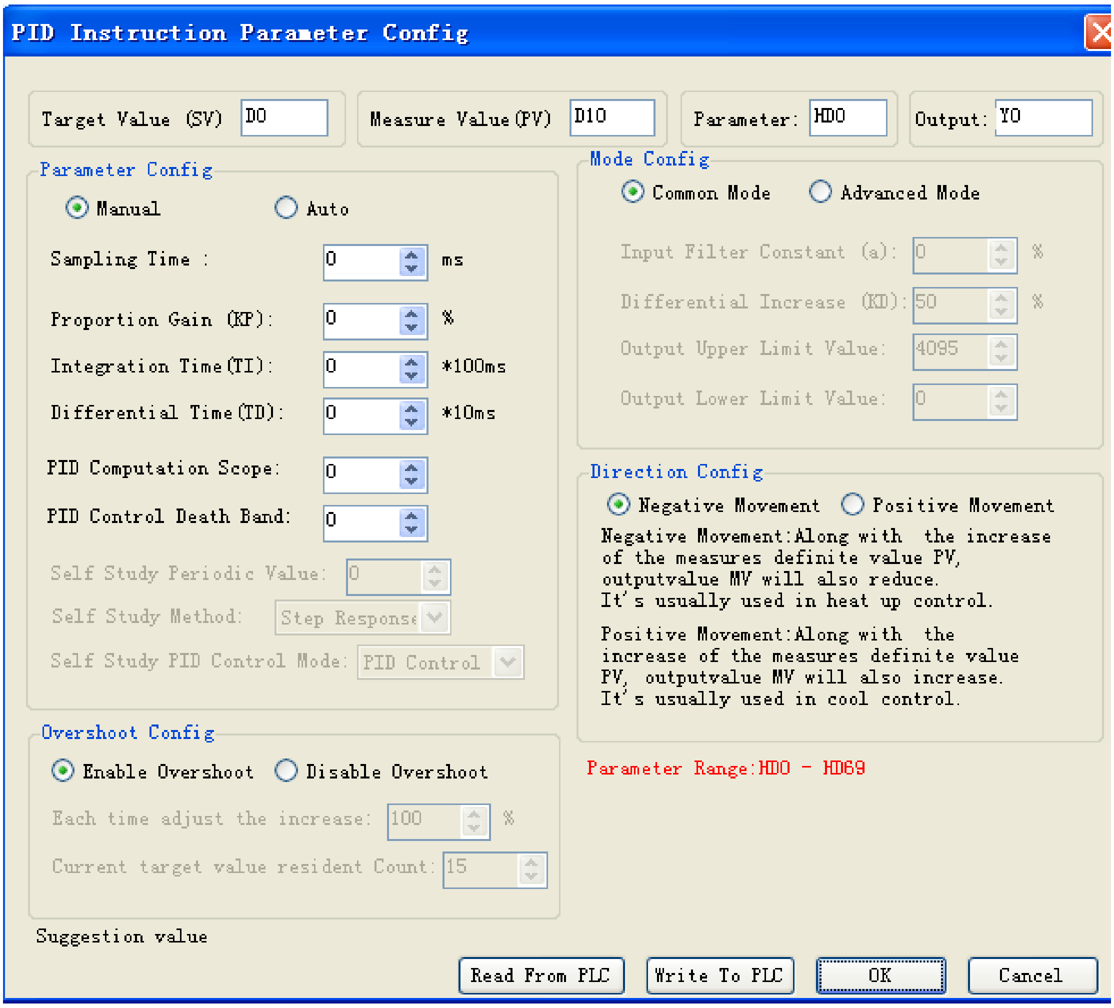
The result of the operation:

1. Analog output: digital form of MV = u(t), the default range is 0~4095.

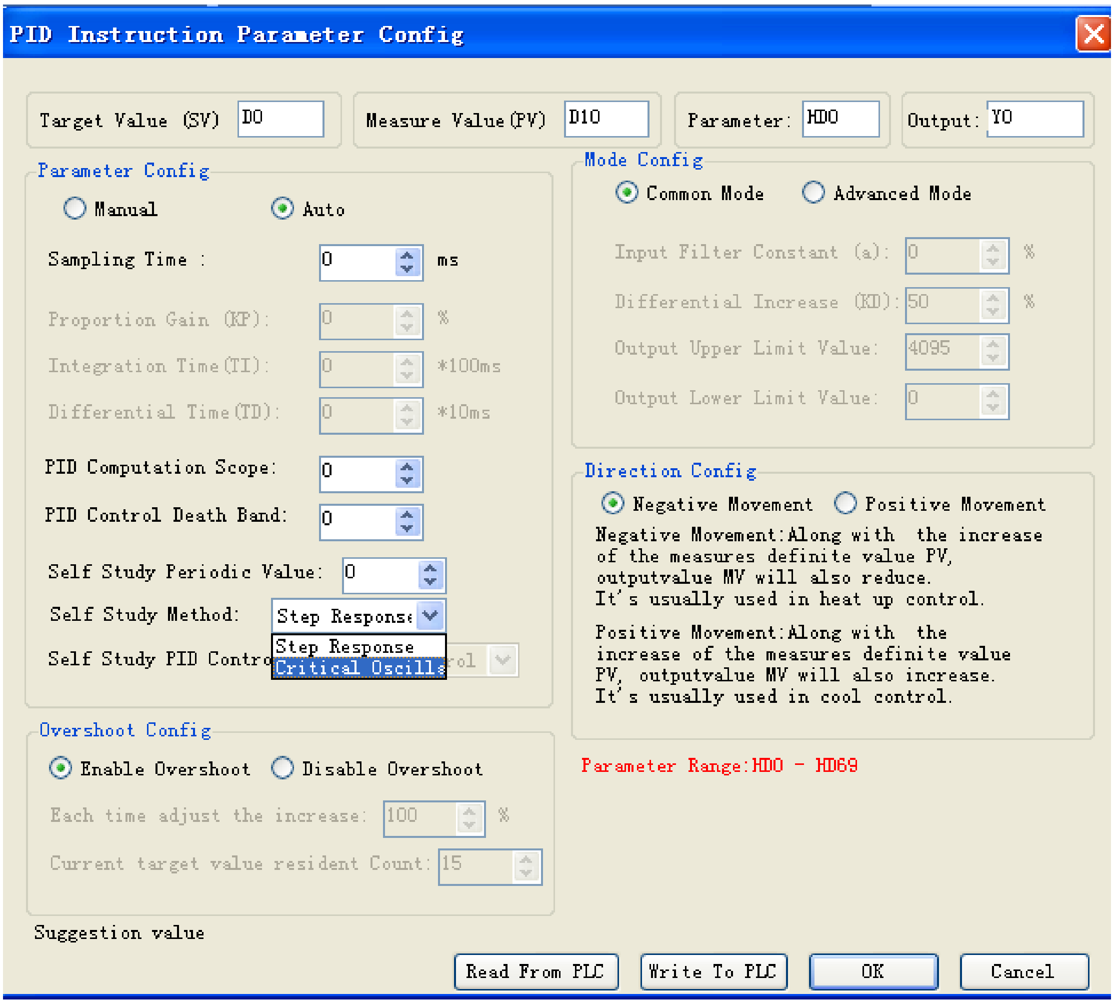
2. Digital output: Y = T * [MV / PID output upper limit]. Y is the outputs activate time within the control cycle. T is the control cycle, equals to the sampling time. PID output upper limit default value is 4095.

8-3. Parameters setting

Users can call PID in XCP Pro software directly and set the parameters in the window (see graph below), for the details please refer to XCP Pro user manual. Users can also write the parameters into the specified registers by MOV instructions before PID operation.



Auto tune mode:



V3.2 and higher version software can choose auto tune mode: step response or critical oscillation.

8-3-1． Register and their functions

PID control instruction's relative parameters ID, please refer to the below table:

ID	Function	Description	Memo
S3	Sampling time	32bits without sign	Unit ms
S3+1	Sampling time	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)	Range：1～32767[%]	
S3+4	Integration time (TI)	0～32767[*100ms]	0 is taken as no integral.
S3+5	Differential time (TD)	0～32767[*10ms]	0 is taken as no differential.
S3+6	PID operation zone	0～32767	PID adjustment band width value
S3+7	Control death zone	0～32767	PID value keeps constant in death zone
S3+8	Input filtering constant (a)	0～99[%]	0:No input filtering
S3+9	Differential gain(KD)	0～100[%]	0:No differential gain
S3+10	Upper limit value of output	-32767～32767	
S3+11	Lower limit value of output	-32767～32767	
S3+12	Change of PID auto tune cycle	full scale AD value *（0.3~1%）	threshold
S3+13	PID auto tune overshoot allowing	0：enable overshoot 1： not overshoot（try to reduce the overshoot）	（valid when using step response method）
S3+14	current target value adjustment percent in auto tune finishing transition stage		
S3+15	current target value stop count in auto tune finishing transition stage		
S3+13~S3+15do not use ID			
S3+16 ~ S3+69		occupied by PID operation's internal process	

8-3-2．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+12 need to be set by users;

S3+16～S3+69 are occupied by system, users can’t use them.

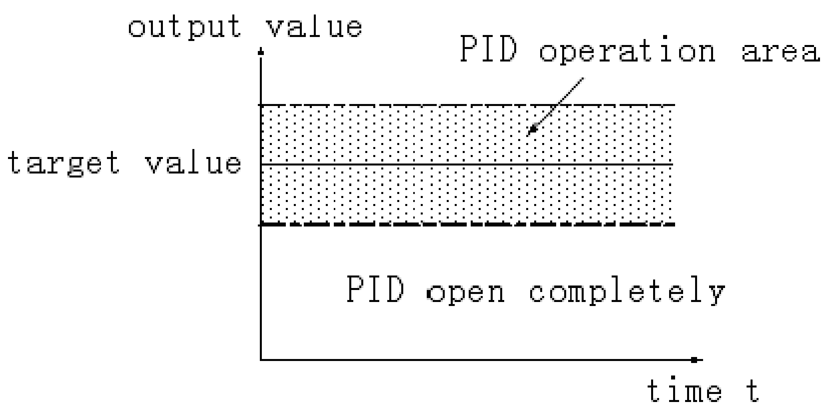
Sample time[S3]

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.

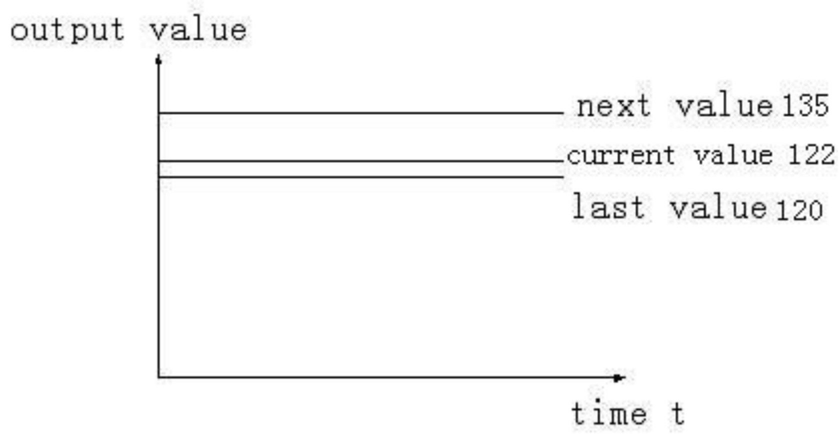
See graph below:



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]

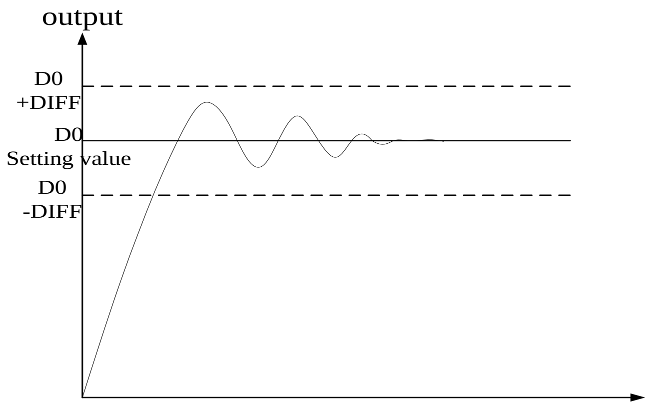
If the measured value changed slightly for a long time, and PID control is still in working mode, then it belongs to meaningless control. Via setting the control death region, we can overcome this situation. See graph below:



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.

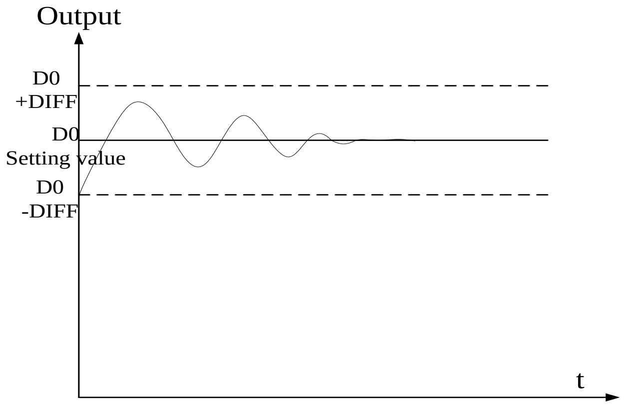
8-4. Auto Tune Mode

If users do not know how to set the PID parameters, they can choose auto tune mode which can find the 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.
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:
Step response method
Make sure current temperature is equal to ambient temperature



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 at auto tune process finishing transition stage **[S3+14]**

This parameter is effective only when **[S3+13]** is 1.

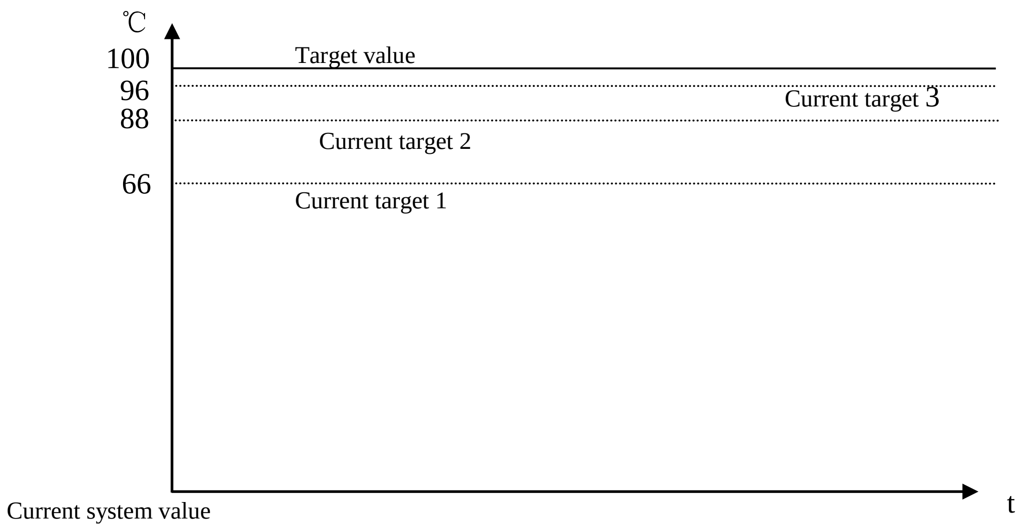
If doing PID control after auto tune, small range of overshooting may be occurred. It is better to decrease this parameter to control the overshooting. But response delay may occur if this value is too small. The defaulted value is 100% which means the parameter is not effective. The recommended range 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.



The stay times of the current target value in auto tune process finishing transition stage **[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 response lag if this value is too large. The default value is 15 times. The recommended range is from 5 to 20.

8-5. Advanced Mode

Users can set some parameters in advanced mode in order to get better PID control effect. Enter into the advanced mode, please set **[S3+2]** bit 15 to be 1, or set it in the XCP Pro software.

Input Filter constant

It will smooth the sampling value. The default value is 0%, which means no filter.

Differential Gain

The low pass filtering process will relax the sharp change of the output value. The default value is 50%; the relaxing effect will be more obviously if increasing this value. Users do not need to change it.

Upper-limit and lower-limit value

Users can choose the analog output range via setting this value.

Default value: lower-limit output =0

Upper-limit =4095

8-6. Application outlines

Under 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 self study.

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.

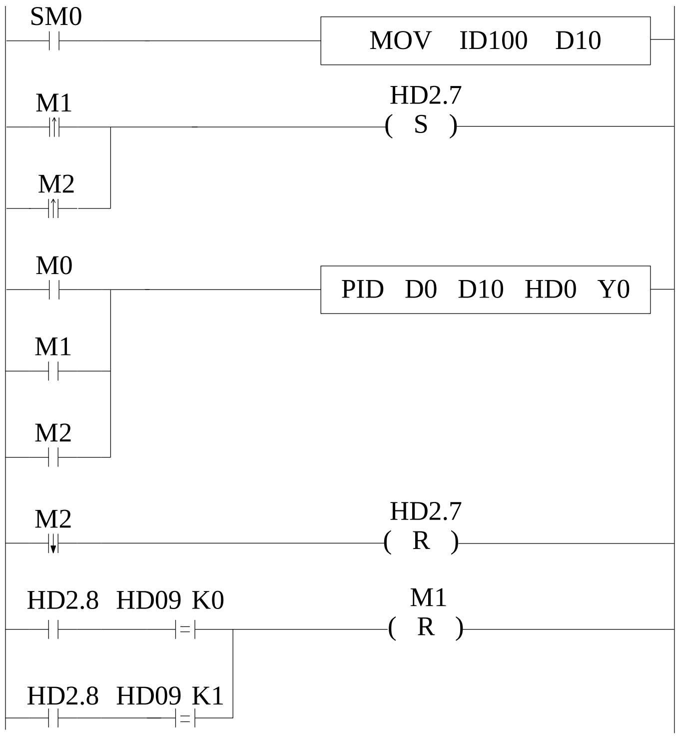
If the auto tune is not available, users can set the PID parameters according to practical experience. Users need to modify the parameters when debugging. Below are some experience values of the control system for your reference:

- 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

8-7. 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.

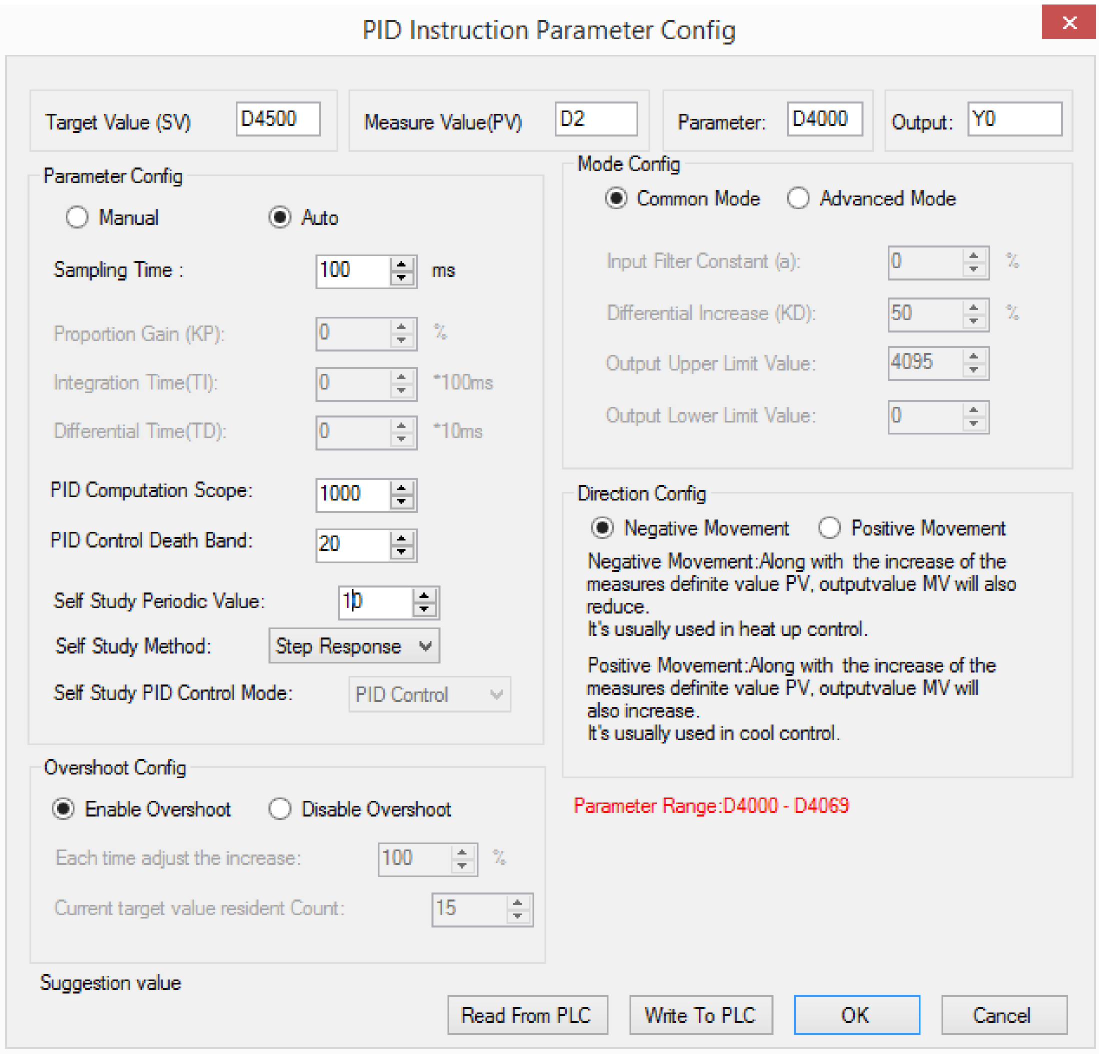
Example 2:

To control the target temperature 60° C in step response mode.

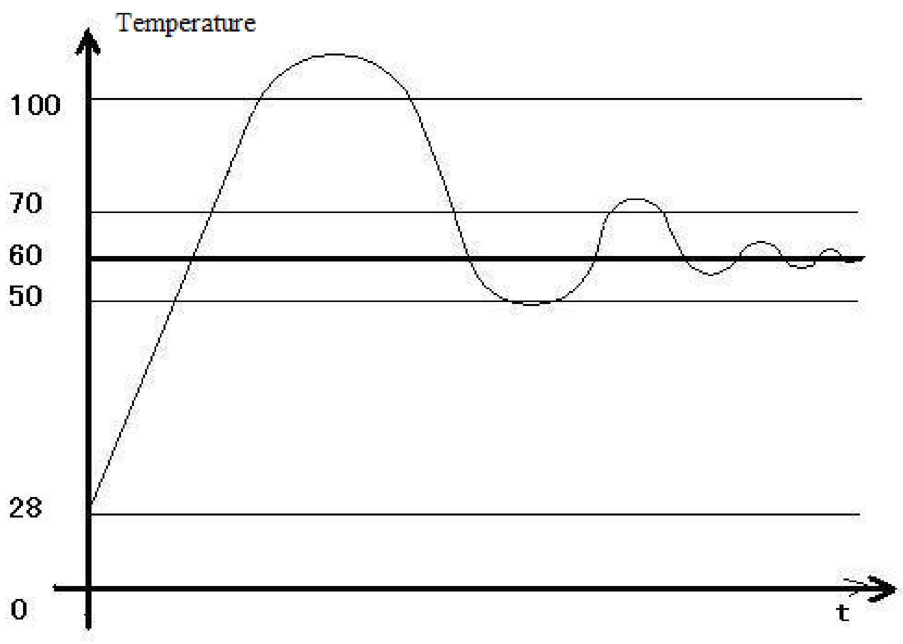
Overshoot is permitted:

1. The target temperature 60° C (6000)

2. Parameters setting



3. The result curve



Explanation:

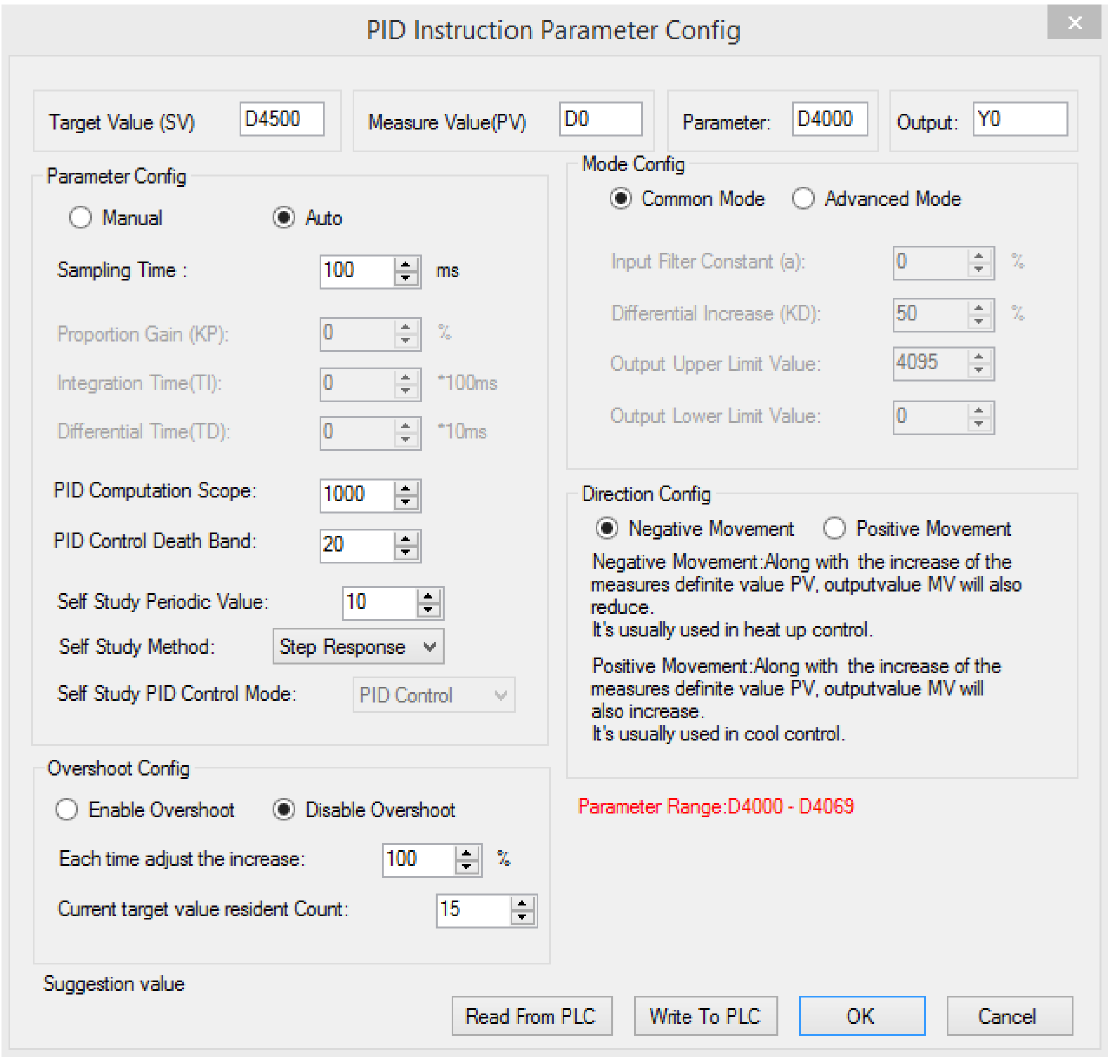
The target temperature is 60 degree, PID calculation range is 10 degree, PID control dead area is 0.2 degree, auto tune period changing value is 10. When the PID control works in normal atmospheric temperature, the PID output terminal will heat the temperature from 28 to 100 degree, then the output stops, 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:

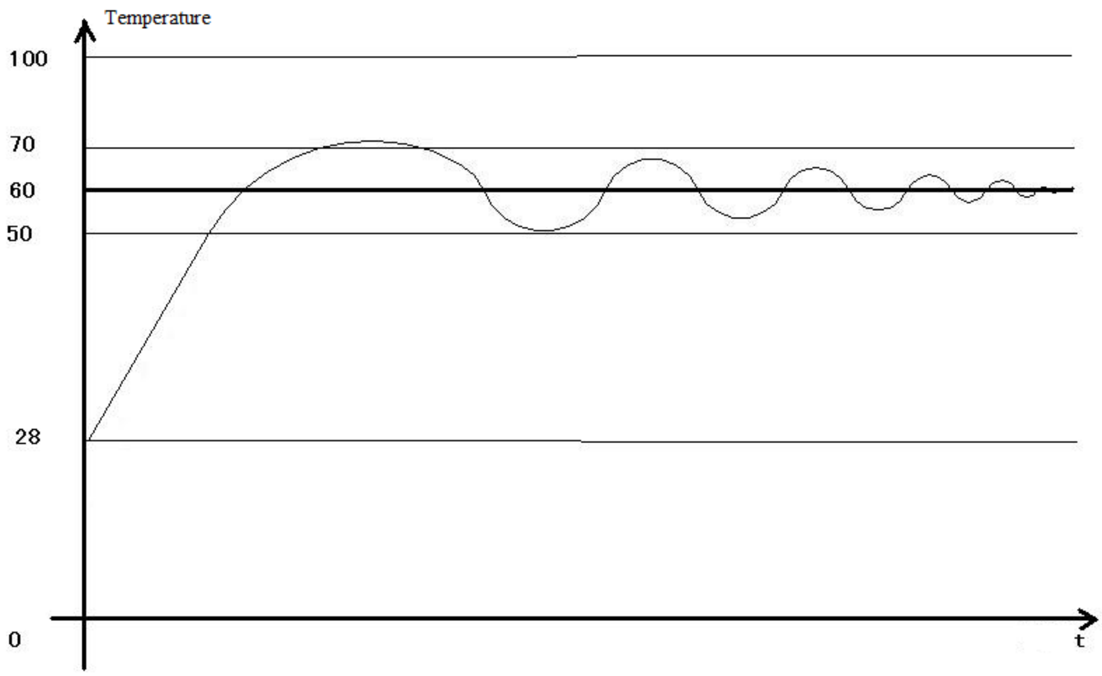
1. When the temperature reaches 100 degree and stops heating, the PID start bit D4002.7 will not reset at once, it has delay before reset.
2. When the temperature reaches 100 degree and stops heating, the PID auto tune success bit D4002.8 will be ON at once.
3. When it starts PID calculation, the PLC 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 (6000)
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.

9 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.

9-1. Summary

XD3, XDM supports almost all C language function in XDPPro software (also supports global variable). Users can call the function at many places and call different functions, which greatly increase program security and programmer's efficiency.

9-2. Instruction Format

1. Instruction Summary

Call the C language Function Block at the specified place.

Call the C language function block [NAME_C]			
16 bits instruction	NAME_C	32 bits Instruction	-
Execution condition	Normally ON/OFF, Rising/Falling Edge activation	Suitable Models	XD3, XDM
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 word B in C language function	bit, BIN

3. Suitable Soft Components

Word

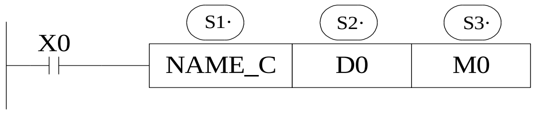
Operands	System								Constant	Module	
	D*	FD	TD*	CD*	DX	DY	DM*	DS*	K/H	ID	QD
S2	●	●	●	●	●	●	●	●			

Bit

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



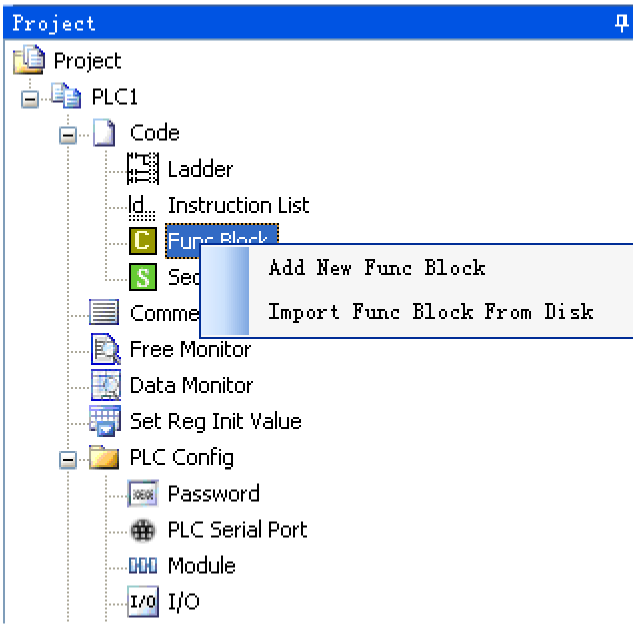
S1 is the function name. It consists of numbers, letters and underlines. The first character can't be number, and the name length should be < =9 ASC.

The name can be the same with PLC's self instructions like LD, ADD, SUB, PLSR etc.

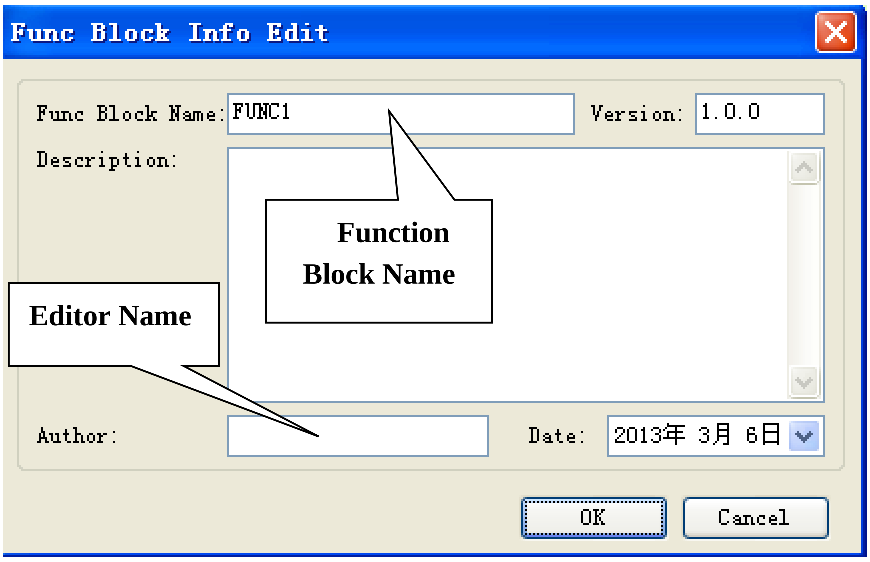
The name can't be the same with the function blocks existing in current PLC;

9-3. Operation Steps

1. Open PLC edit tool, in the left "Project" toolbar, choose "Func Block", right click it and choose "Add New Func Block".

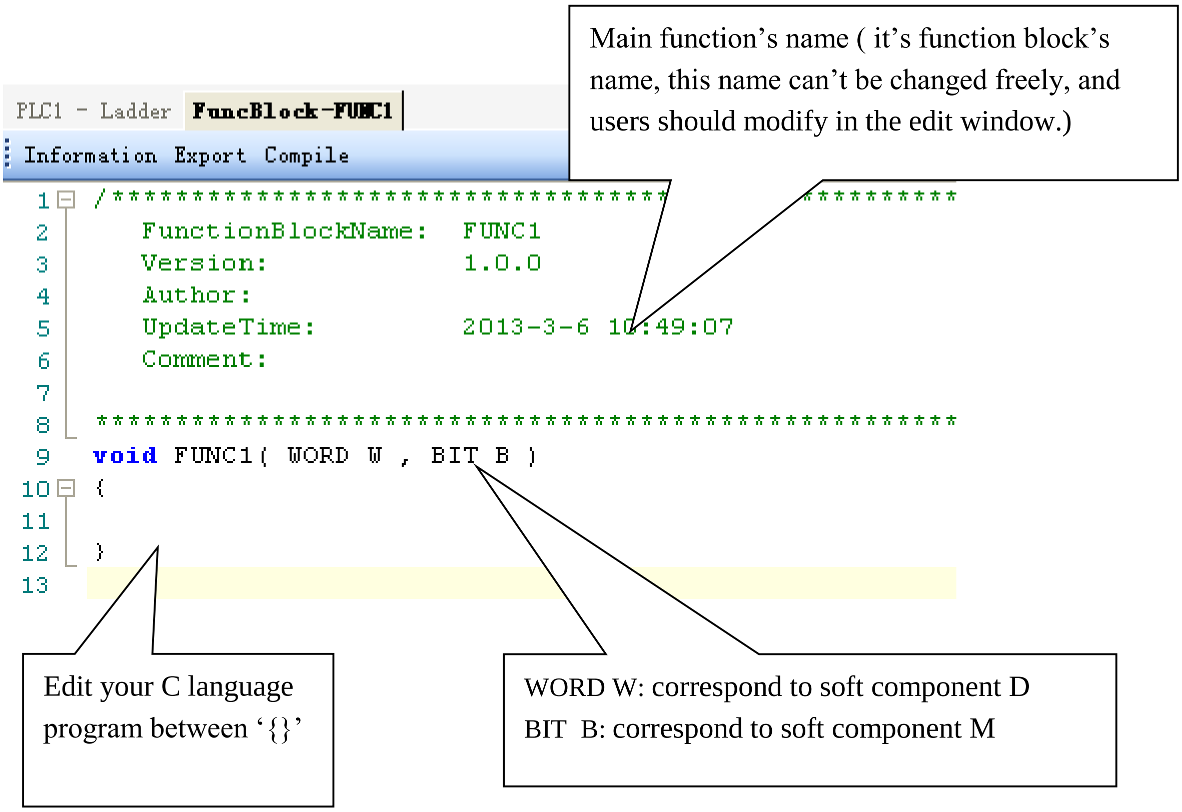


2. See graph below, fill in the information of your function;



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 and M is the start ID of W and B. Take the above graph as the example, start with D0 and M0, then W[0] is D0, W[10] is D10, B［0］is M0, B［10］is M10; if the used parameters in the ladder are D100, M100, then W[0] is D100, B［0］is M100; 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 coil and data type in one C language should be the same. All the coils in C language are power loss retentive, or not power loss retentive; so is the same with data register.

- 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:

In C language of PLC, if you want to use input(X) and output(Y), then macro definition '#define SysReg Addr_X_Y' is needed; E.g: send the state of input X0 to given coil M0, then B[0]=X[0]; send the state of Y0 to given coil M10, then: B[10]=Y[0];（Note: corresponding X Y in C language is decimal, not Octonary number）.

In a similar way, if the not-power-loss-retentive flow S, Counter C, timer T, counter register TD is in the C language, macro definition '#define SysRegAddr_S_C_T_CD_TD' is also needed; if the power-loss-retentive flow HS, counter HC, timer HT, counter register HCD, timing register HTD etc, macro definition '#define SysRegAddr_HS_HC_HT_HCD_HTD' is needed.

E.g: W[0]=CD[0];W[1]=TD[0];B[1]=C[0];B[2]=T[0];

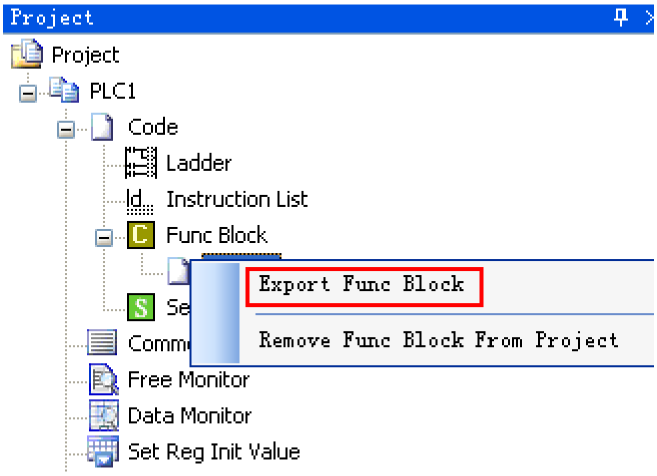
- Function Library: In **Function Block**, users can use the Functions and Constants in function library directly. For the Functions and Constants in function library, see 9-8.
- The other data type supported:

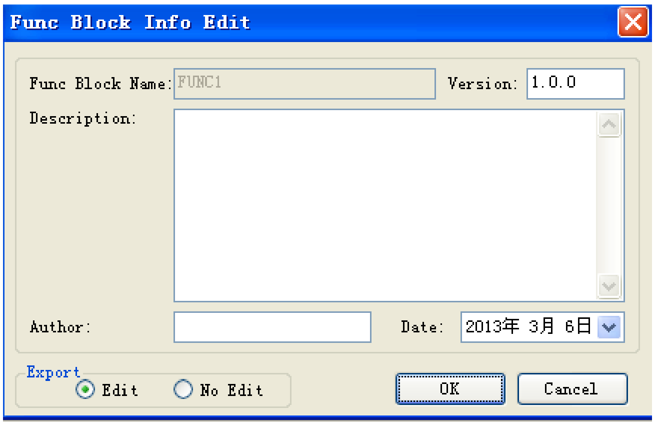
```
      BOOL;              //BOOL Quantity
INT8U;        //8 bits unsigned integer
INT8S;        //8 bits signed integer
INT16U                //16 bits unsigned integer
INT16S        //16 bits signed integer
INT32U                //32 bits unsigned integer
INT32S        //32 bits signed integer
FP32;    // single precision floating
FP64;         //double precision floating
Predefined Marco:    #define    true     1
                          #define     false    0
                          #define     TRUE     1
                          #define     FALSE   0
```

9-4. Import and Export the Functions

1. Export

(1) Function: Export the function as the file, then other PLC program can import to use;





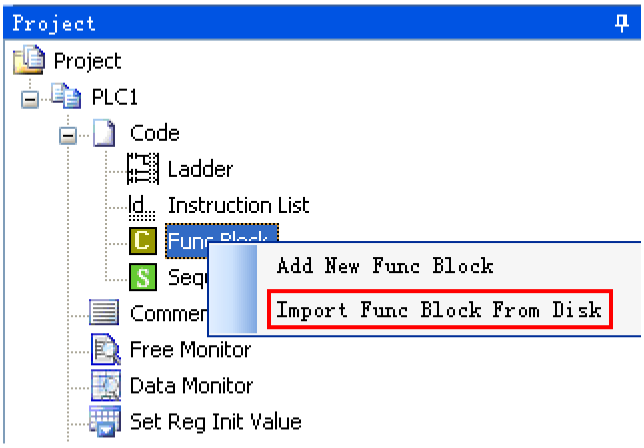
(2) Export Format

a) Editable: Export the source codes out and save as a file. If import again, the file is editable;

b) Not editable: Don't export the source code, if import the file, it's not editable;

2. Import

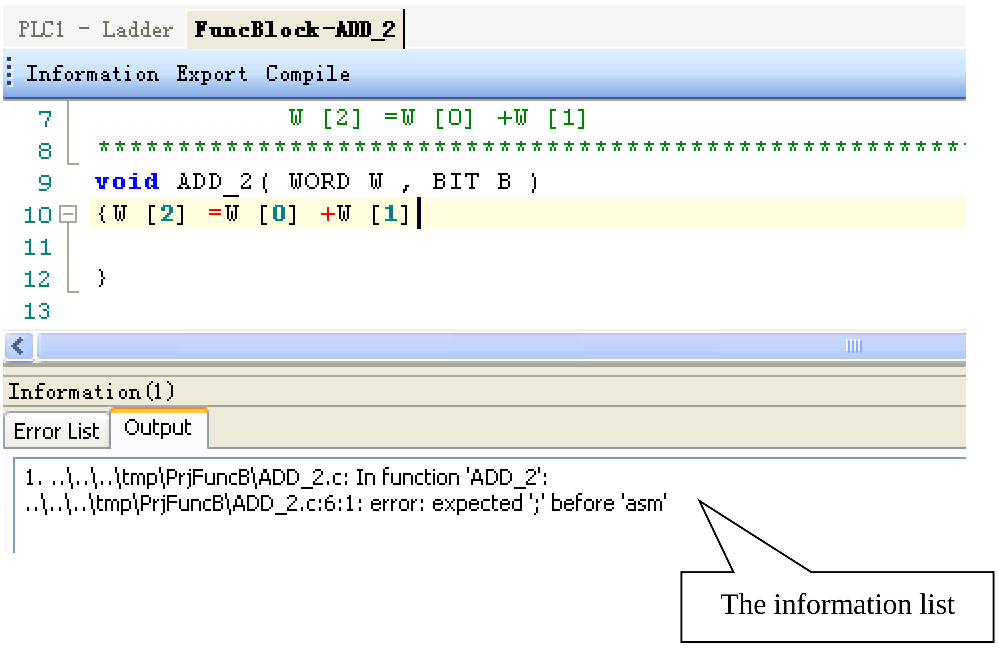
Function: Import the existing **Func Block** file, to use in the PLC program.



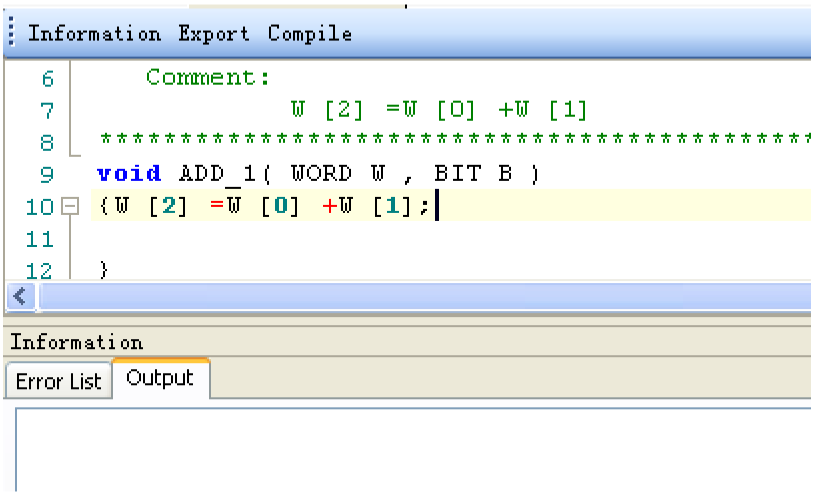
Choose the **Func Block**, right click ‘Import Func Block from Disk’, choose the correct file, and then click OK.

9-5. Edit the Func Blocks

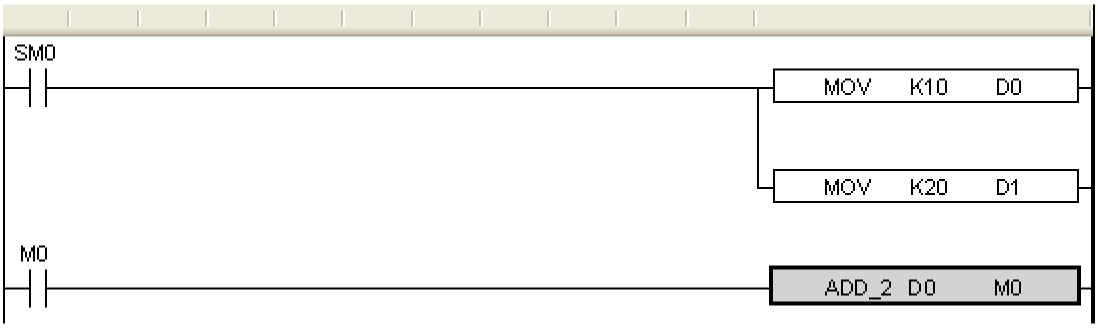
Example: Add D0 and D1 in PLC’s registers, and then assign the value to D2;
(1) In ‘Project’ toolbar, new create a **Func Block**, here we name the **Func Block** as **ADD_2**, then edit C language program;
(2) Click ‘compile' after edition.



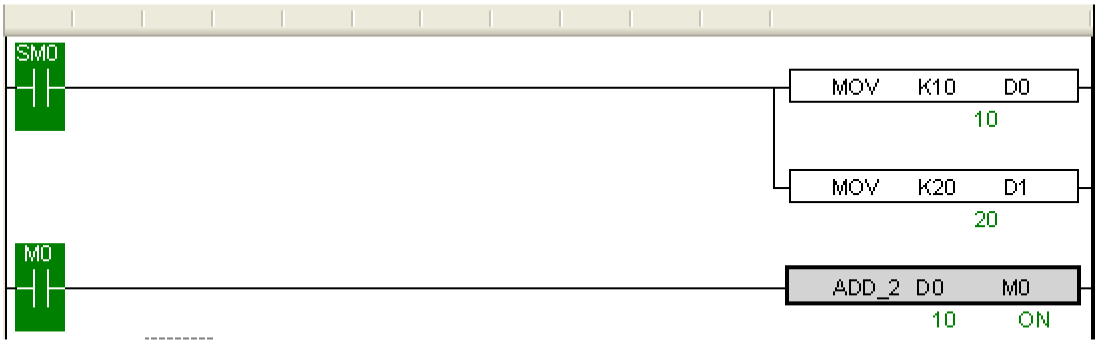
According to the information shown in the output blank, we can search and modify the grammar error in C language program. Here we can see that in the program there is no ‘;’ sign behind W [2] =W [0] + W [1].
Compile the program again after modifying the program. In the information list, we can confirm that there is no grammar error in the program.



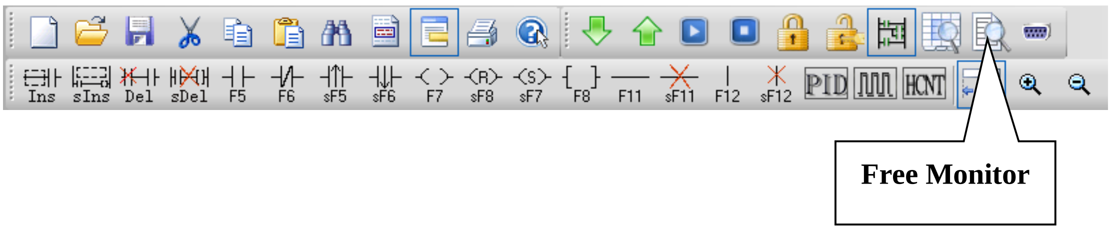
(3) Write PLC program, assign value 10 and 20 into registers D0, D1 separately, then call Func Block ADD_2, see graph below:



(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;



9-6. 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

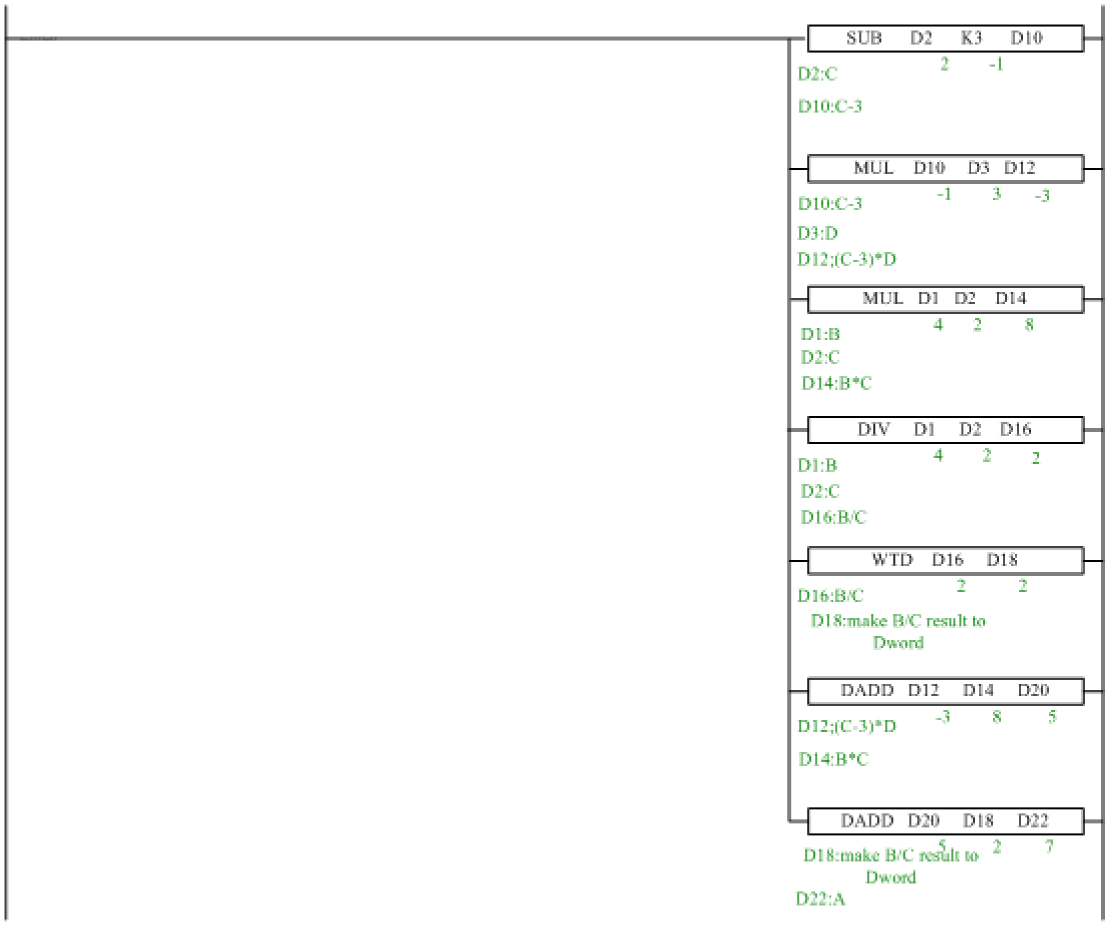
Method 1: use ladder chart:

Get the result of c-3

Get the result of three multiplication equations

Get the sum

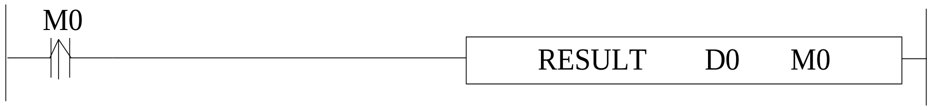
Ladder chart only support two original operands, it needs many steps to get the result.



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.

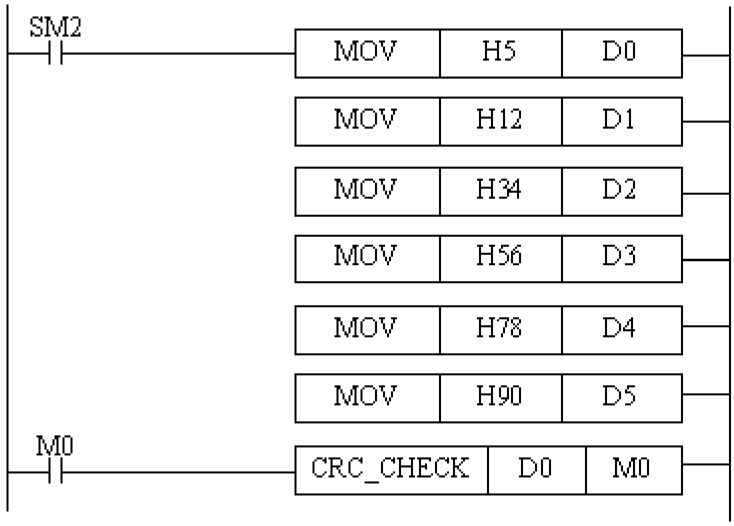
Edit C language Function Block program, see graph below:

```
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;

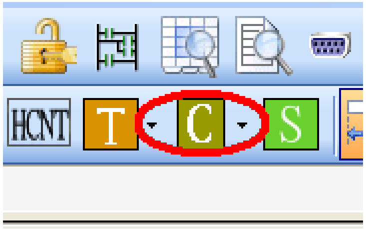
9-7. Application

In one Func Block file, you can write many functions, and they can be called by each other. Each Func Block file is independent, they can’t call block in each other;

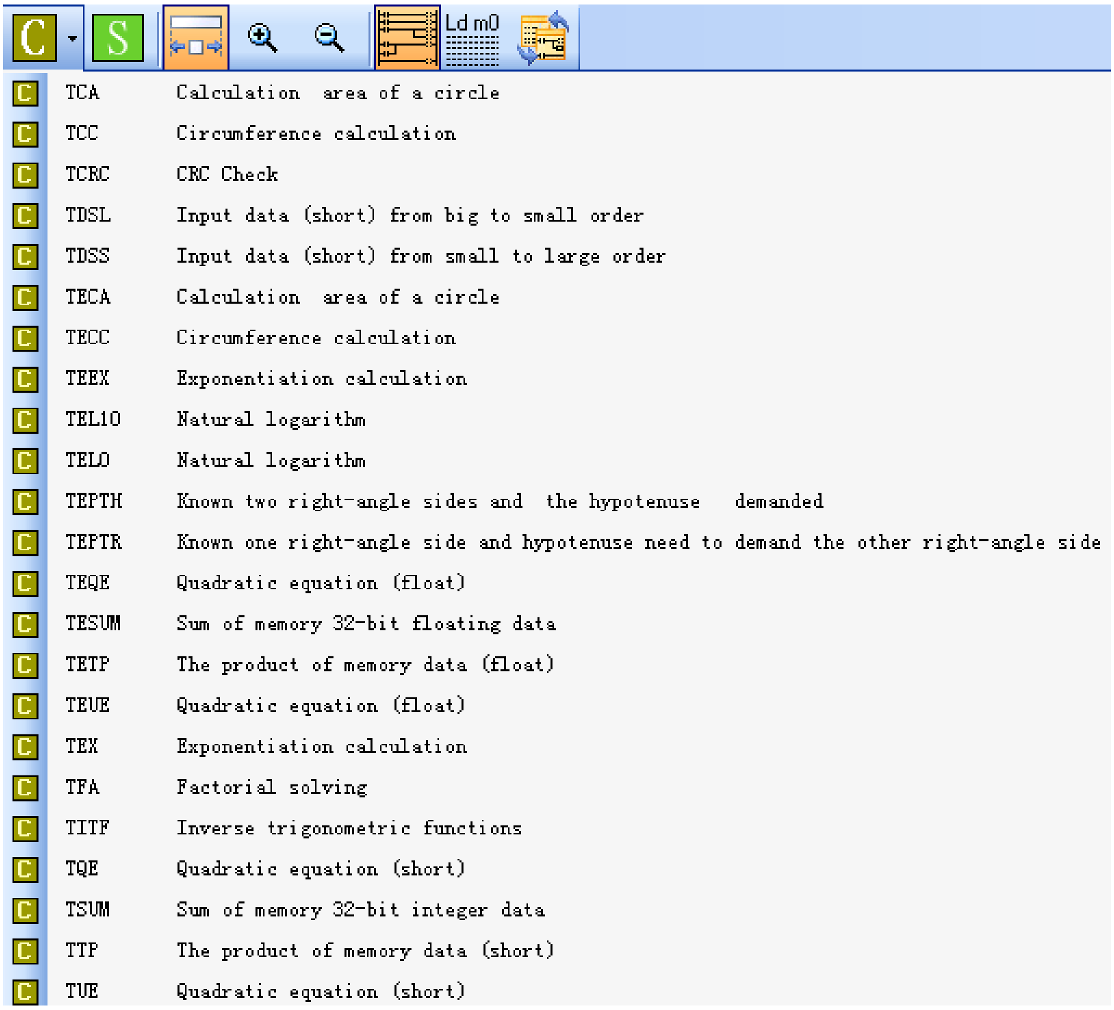
Func Block files can call C language library function in form of floating, arithmetic like sin, cos, tan.

XC series PLC only support local variable, while XD3 series PLC support both local and global variable. This makes C language Block more flexible and convenient.

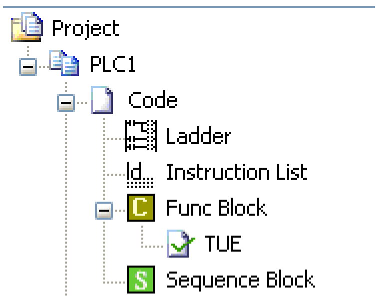
XDPPro software v3.3 and later version keep C function library:



In this function block, user can call the C function directly:



For example: click TEL10, the function name will show on the project bar:



User can call it in the ladder chart editing window at any time.

9-8. 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:

```
9  void ZHENGXIAN( WORD W , BIT B )
10 {
11 int a;
12 float x,y,z;
13 x=FW[0];
14 y=asinf(x);
15 z=180*y/3.14159;
16 a=(int)z;
17 W[2]=a;
18 }
```

10 Sequence BLOCK

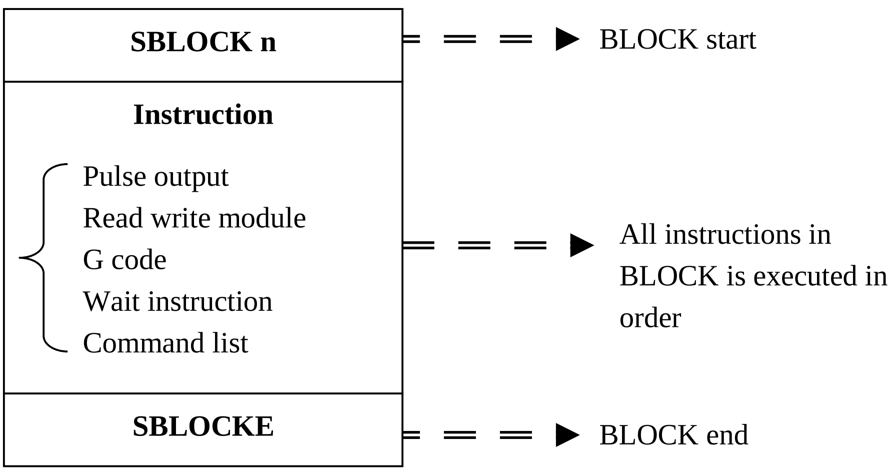
This chapter mainly introduces sequence block instruction and the application.

Sequence Block instruction:

Mnemonic	Function	Ladder chart	Chapter
Sequence Block			
SBSTOP	Pause BLOCK	SBSTOP S1 S2	10-6-1
SBGOON	Go to execute BLOCK	SBGOON S1 S2	10-6-1

10-1. 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:



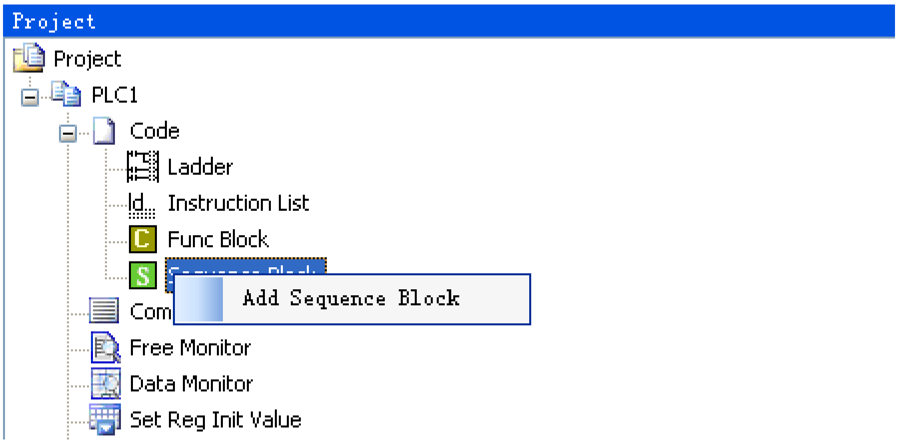
※1: XD3, XDM series PLC can support up to 8 BLOCK. When the BLOCK trigger condition is normal ON coil, the BLOCK will execute from the top to down and return to the top to execute until the trigger condition is OFF; When the BLOCK trigger condition is rising edge, the sequence BLOCK will execute from the top to bottom only one time.

10-2. Call the BLOCK

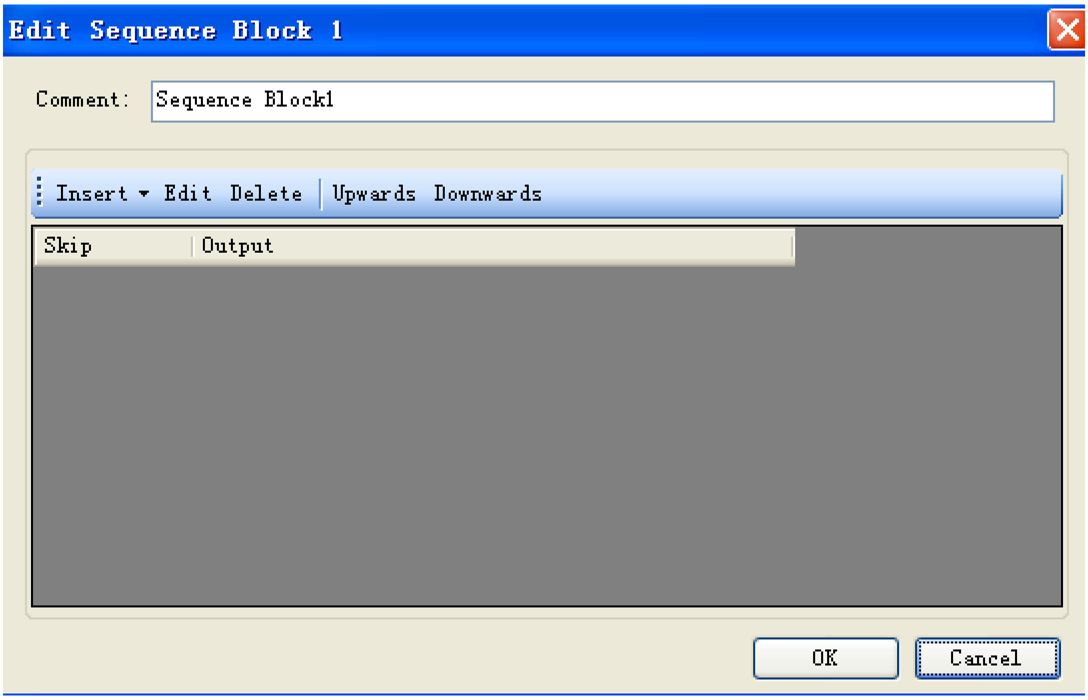
In one program file, it can call many BLOCK; the following is the method to add BLOCK in the program.

10-2-1. Add the BLOCK

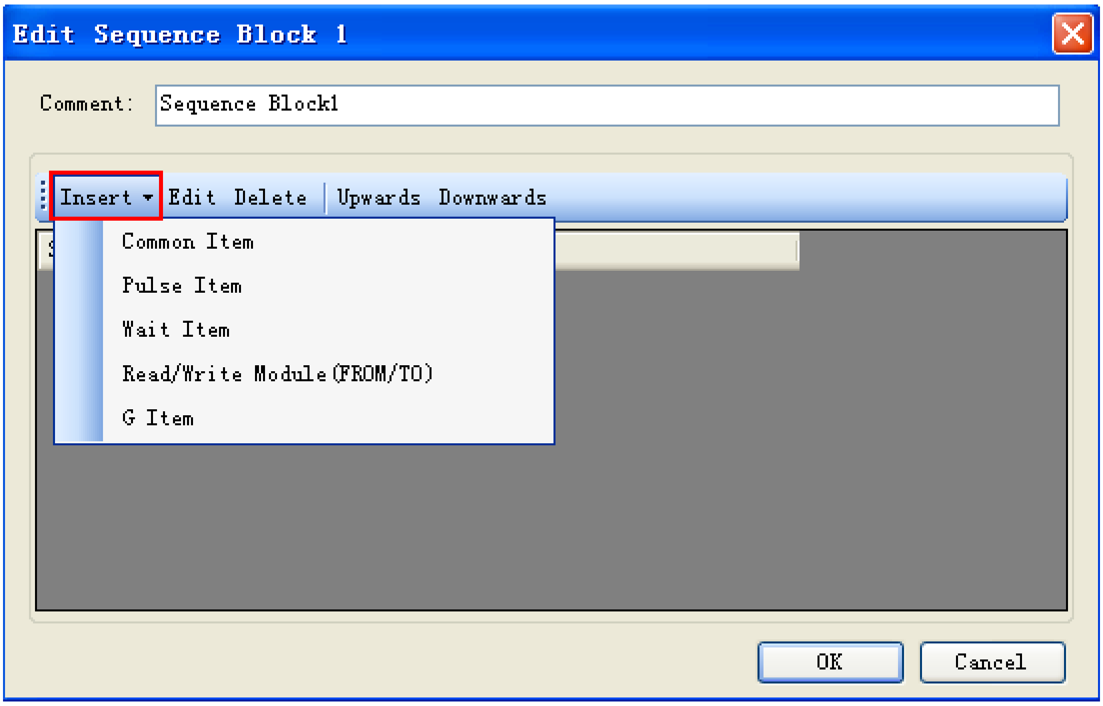
Open XDPPro software, right click the sequence block in the project bar:



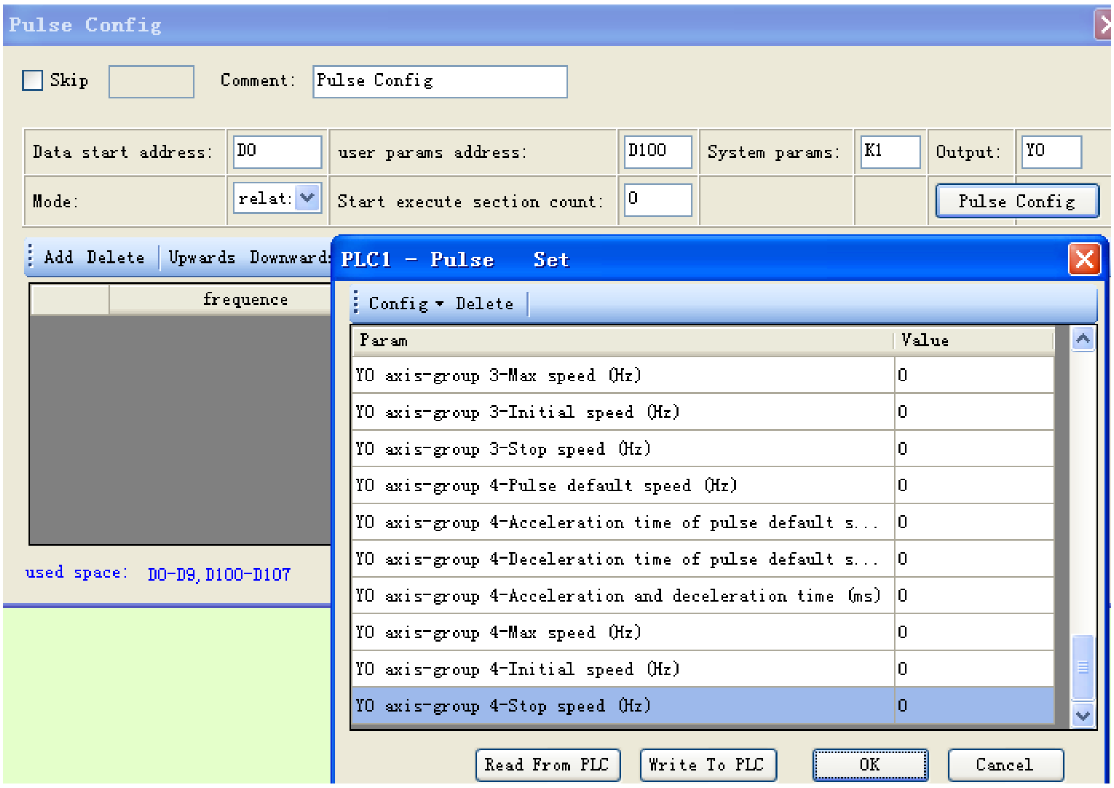
Click the command 'add sequence block', the following window will jump out:



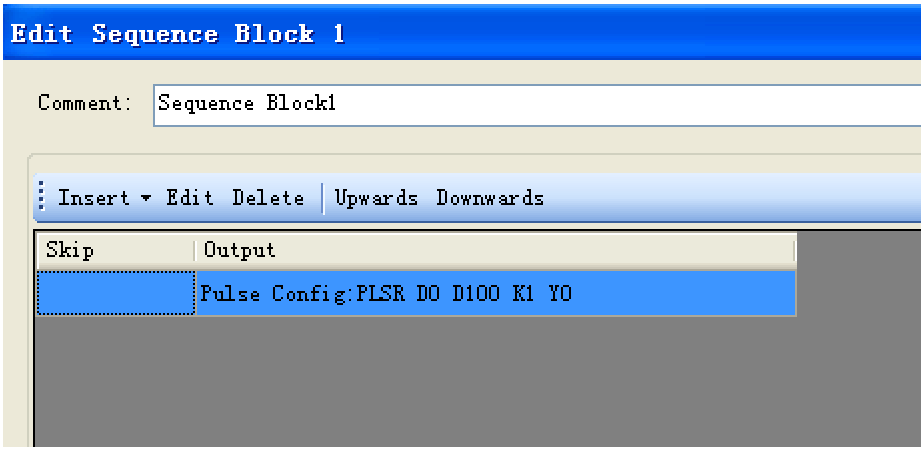
You can edit the BLOCK in the window, Upwards/Downwards are used to change the position of instructions in the block.
Click 'insert' button, some instructions list under the menu:



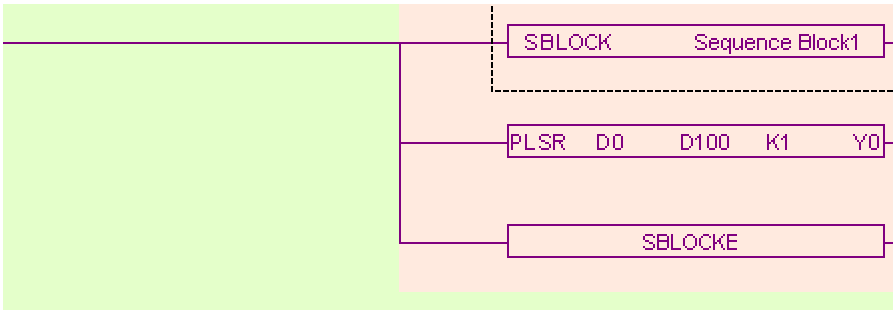
Take 'Pulse Item' for example:



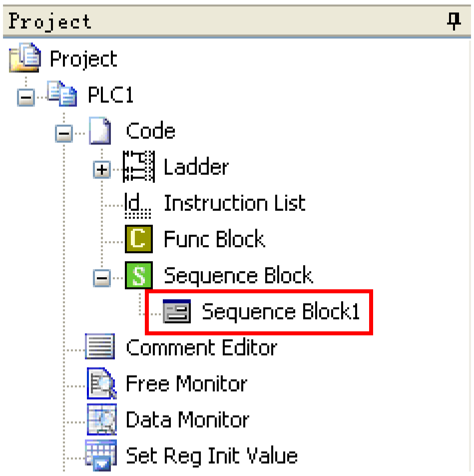
After click ‘OK’, you will find information in the configuration:



Click ‘OK’, the following instructions are added in the ladder:

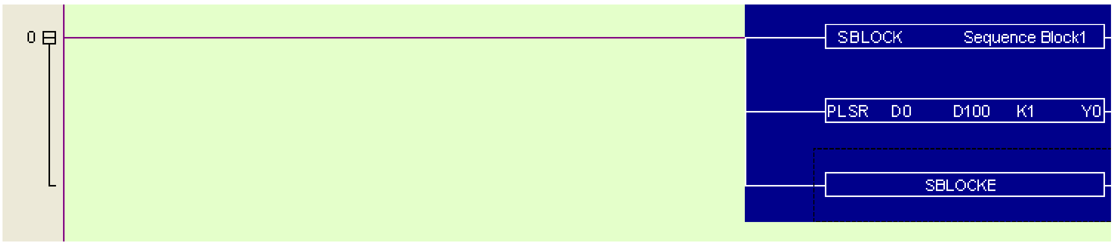


Meantime, a new sequence block is added in the right of the project bar:

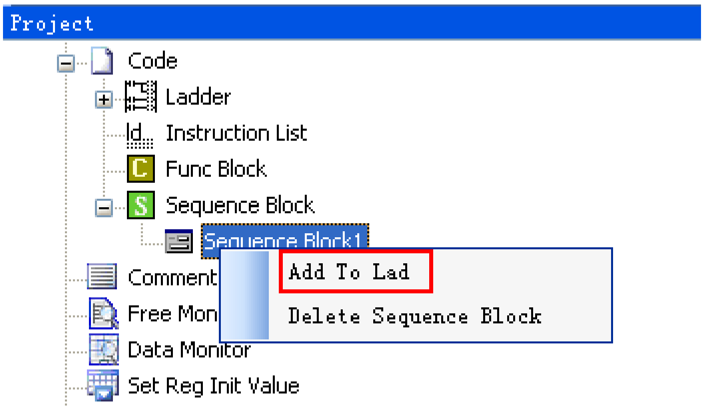


10-2-2． Move the BLOCK

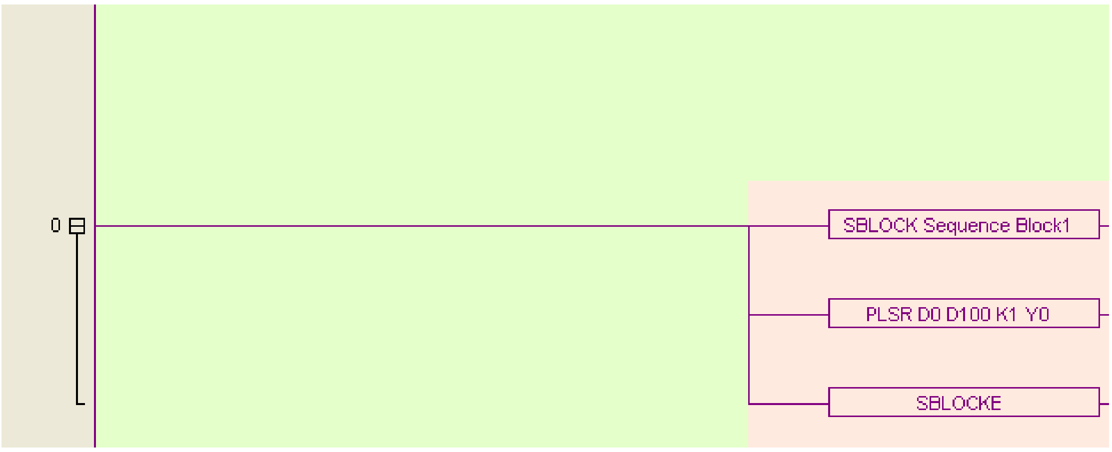
If you want to move the BLOCK to other place, you have to select the original BLOCK and delete it (select all, then delete):



Move the cursor to the new place, and then right click the BLOCK and select 'add to lad':

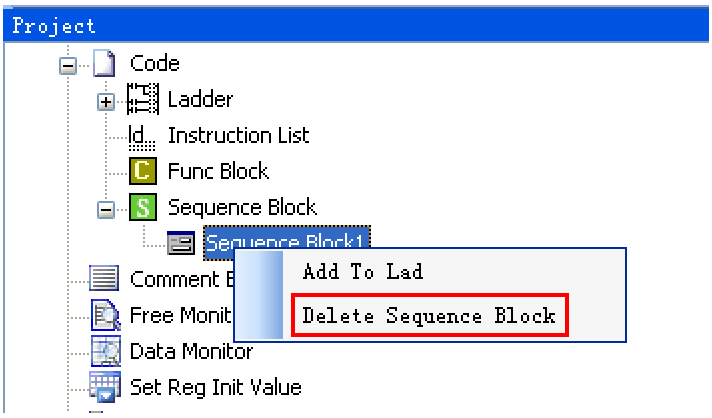


Now the BLOCK is moved to the new place:



10-2-3． Delete the BLOCK

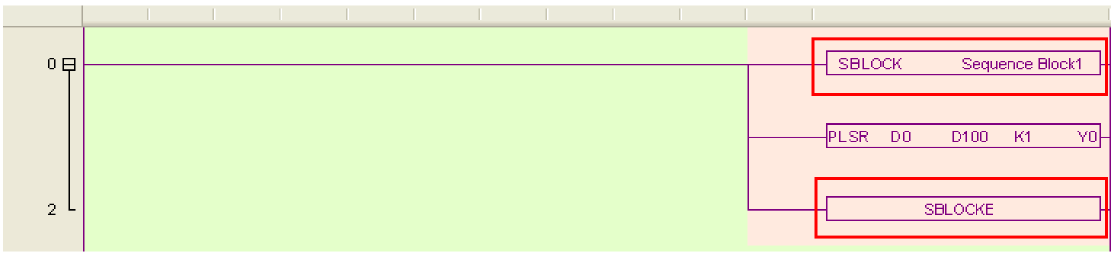
You can select the called BLOCK and delete it. If you want to completely delete the BLOCK, right click the function block and select ‘delete sequence block’. After this operation, you can’t call this BLOCK any more:

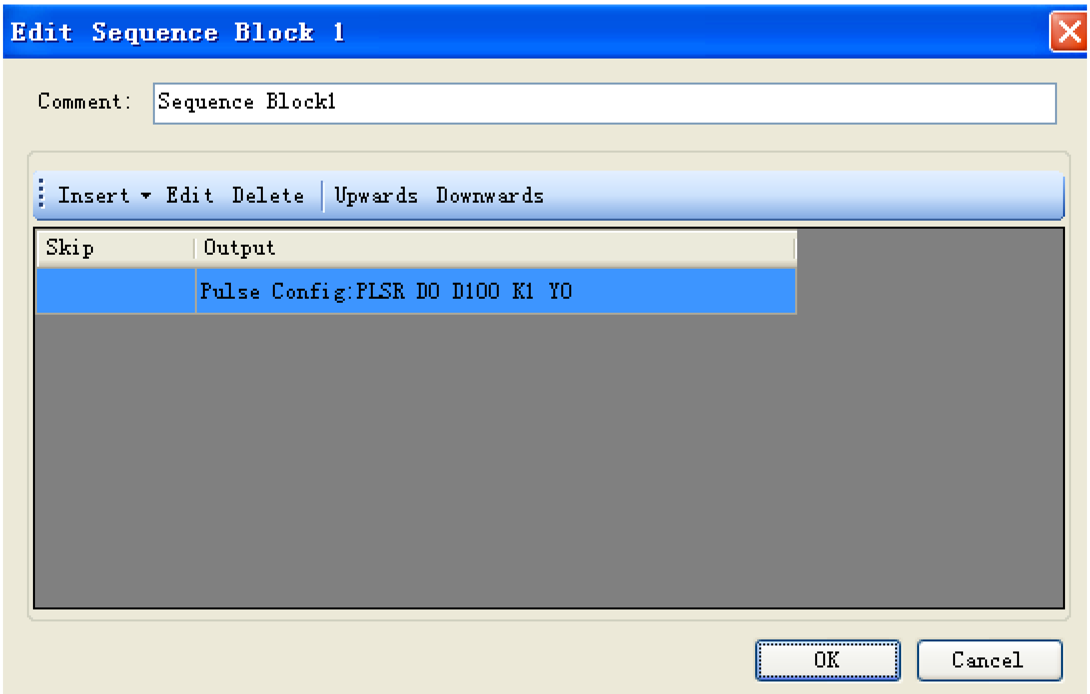


10-2-4． Modify the BLOCK

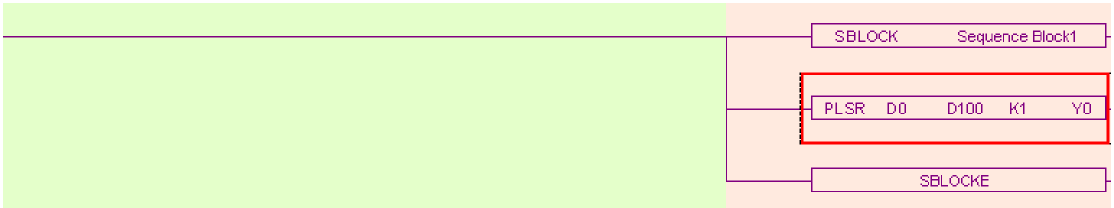
There are two methods to modify the BLOCK.

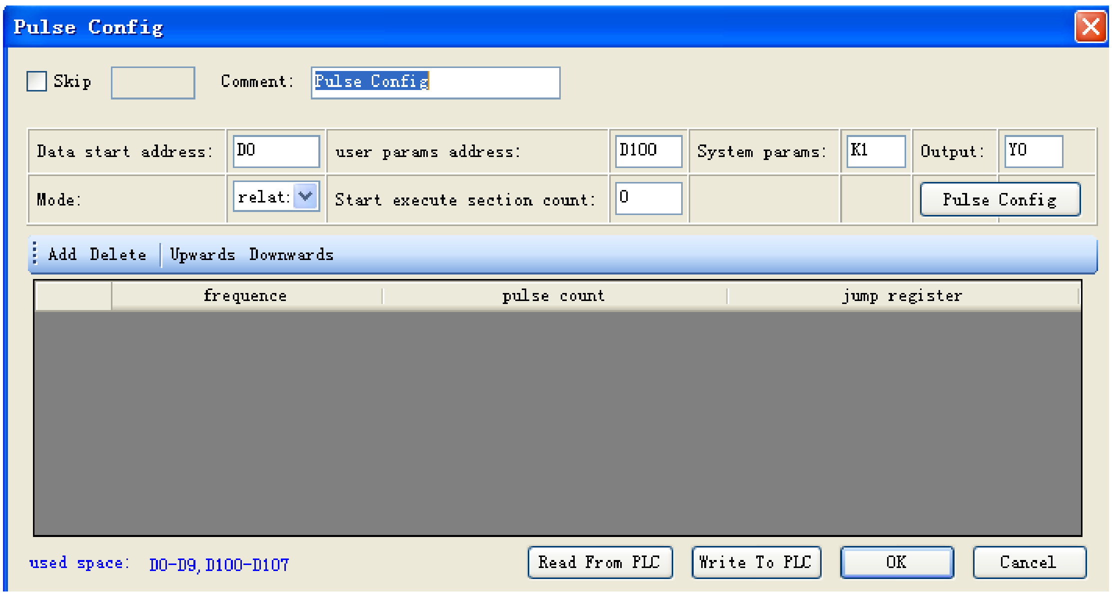
(A) Double click the start/end segment to modify the BLOCK in general:





(B) Double click the middle part to modify :

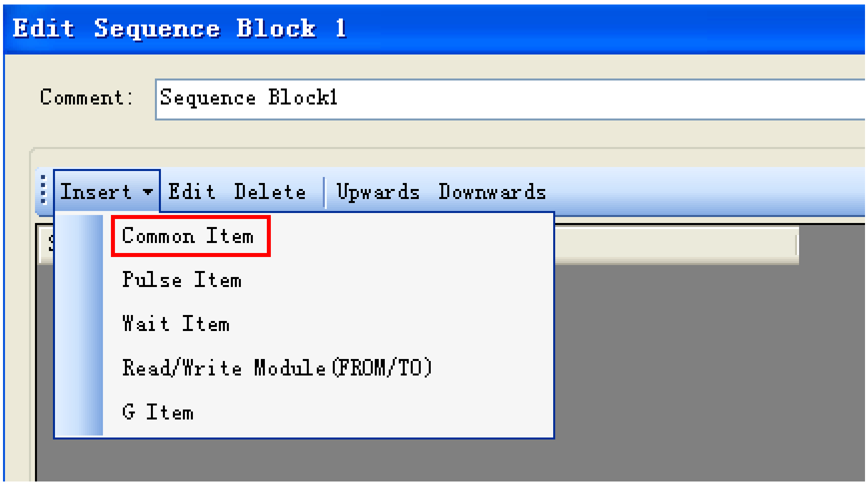




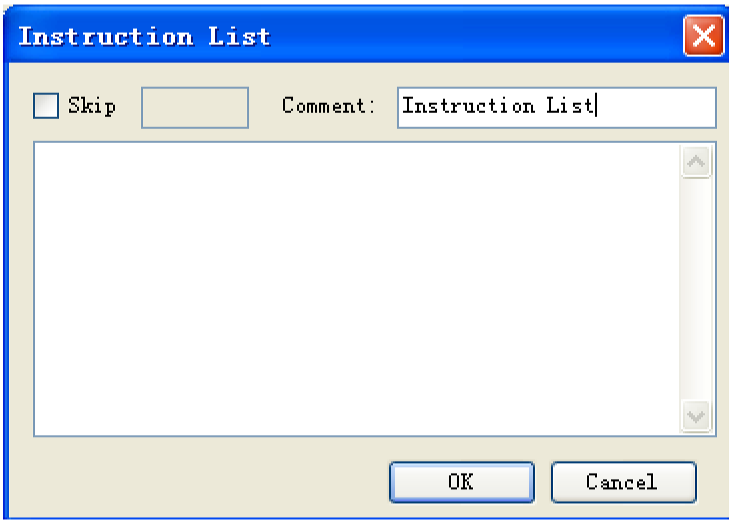
10-3. Edit the instruction of the BLOCK

10-3-1. Command item

Use ‘command item’ to edit the program:



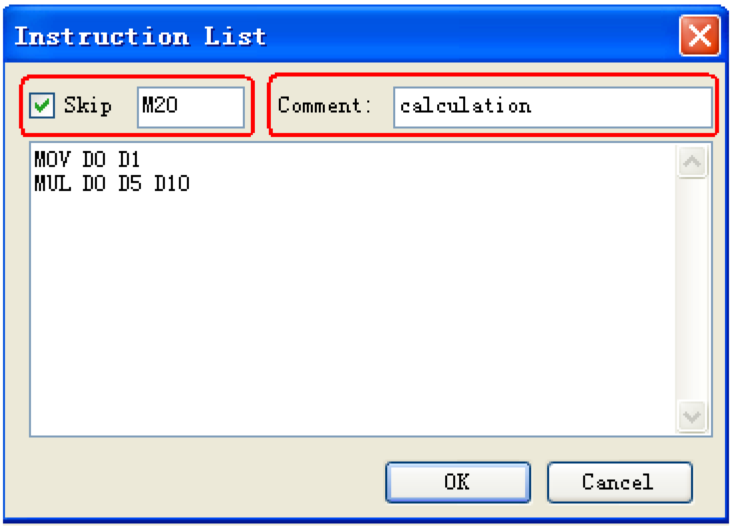
An ‘instruction list’ will jump out after click the ‘command item’:



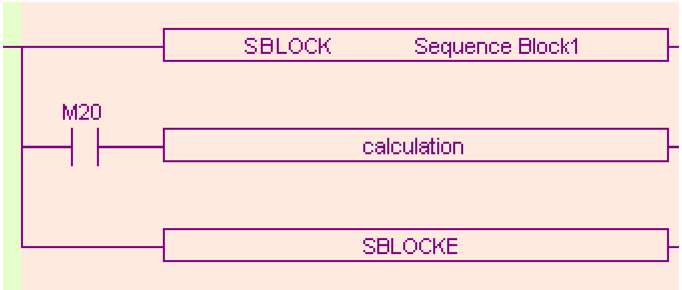
Users can add instructions in the frame.

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.



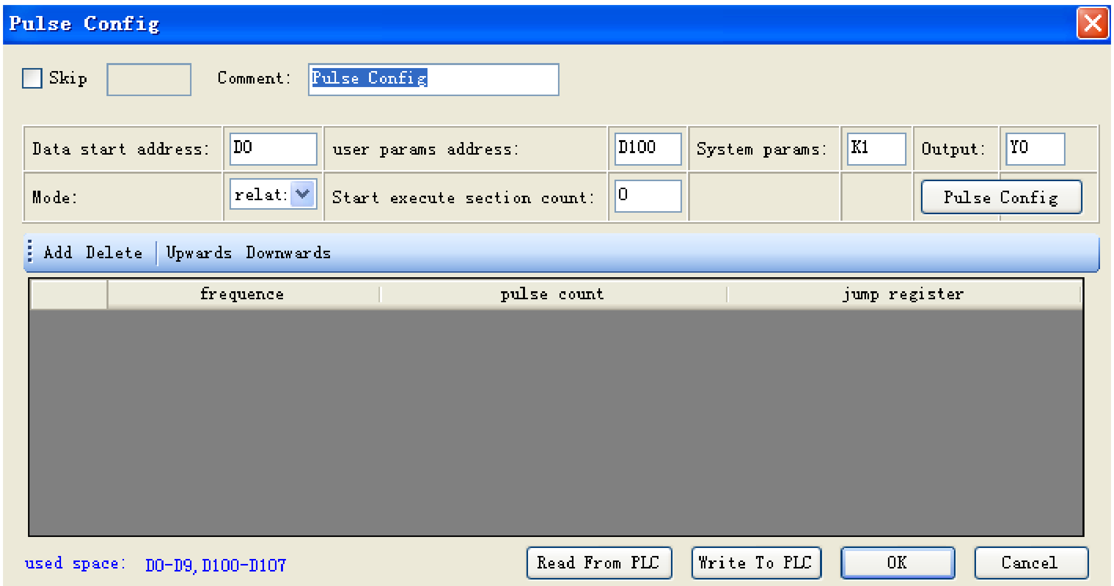
Click ‘OK’, the ladder program will change as the following:



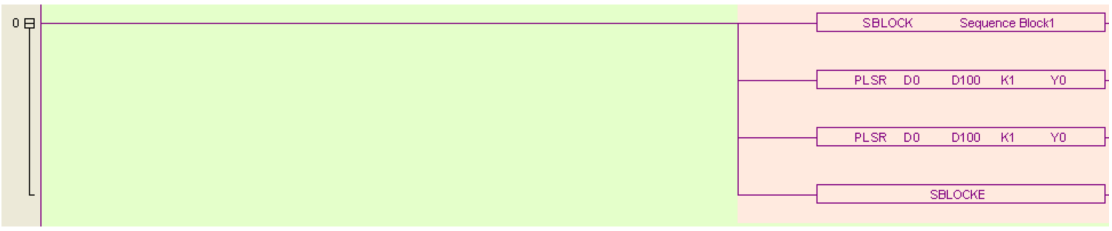
Note: We can add multiply instructions in one BLOCK and use ‘Skip’ as every instruction’s execution condition.

10-3-2. Pulse Item

Open the ‘pulse item’ in the same way:



In the following BLOCK, we add two impulse instructions:

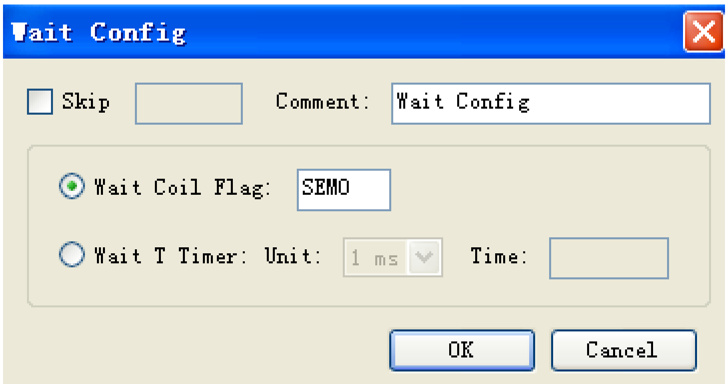


10-3-3. Wait Item

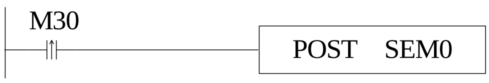
'Wait Item': to wait coil flag or timer bit.

Open 'Wait Item' in the same way. There are two waiting modes: flag bit and timer wait.

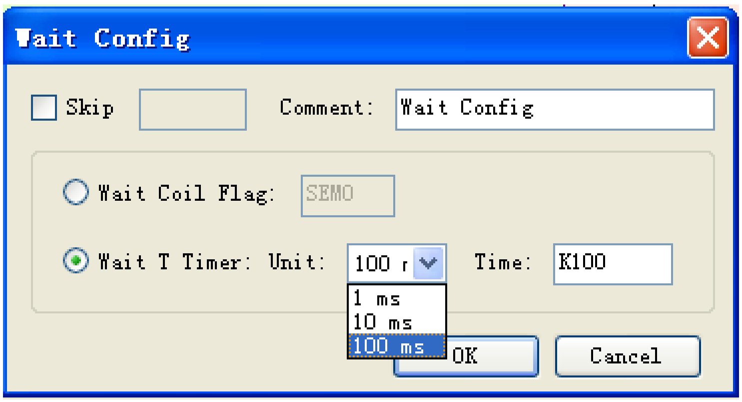
(A) Flag bit



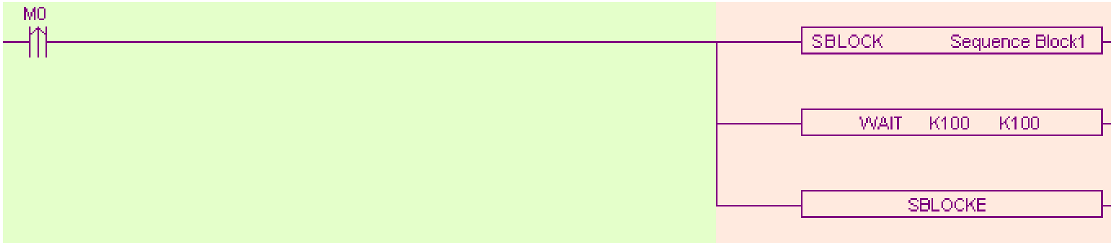
SEM corresponding ladder diagram is as below:



(B) Timer wait



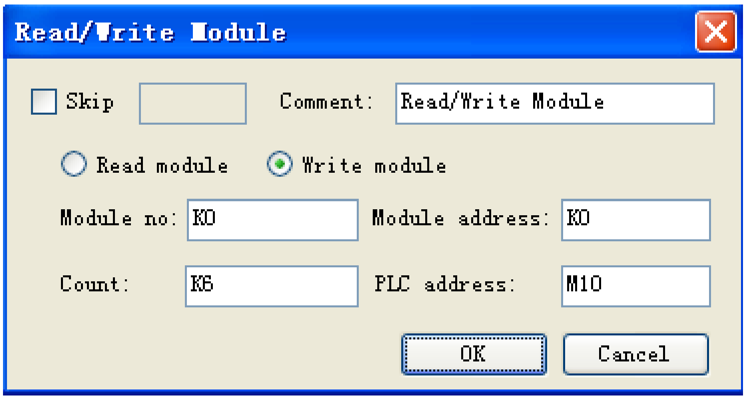
(C) Corresponding ladder diagram:



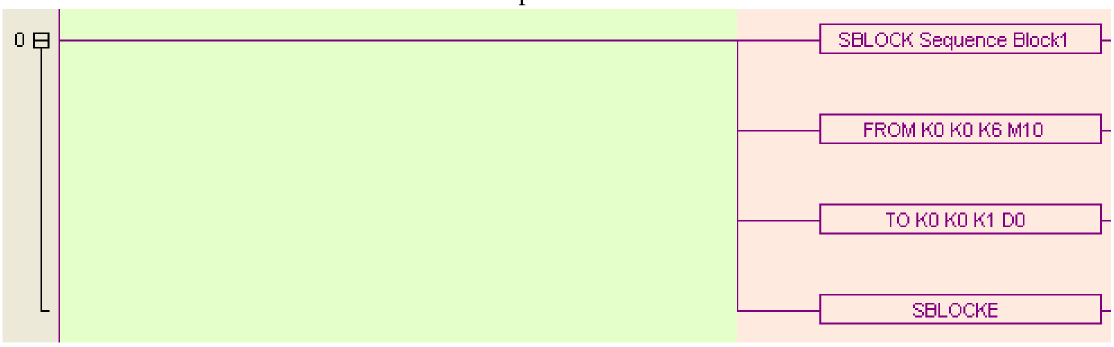
Note: Do not add normal coil after WAIT instruction in XD3 series PLC sequence BLOCK, and add XD3, XDM series PLC special signal SEM bit(SEM0~SEM31); SEM cannot be controlled by set or reset. It can only be set by POST instruction and reset by WAIT SEM instruction.

10-3-4. 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

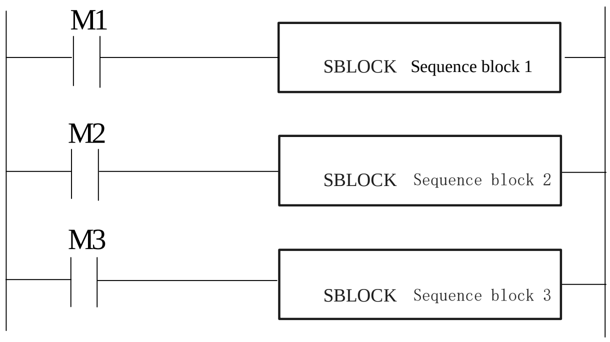


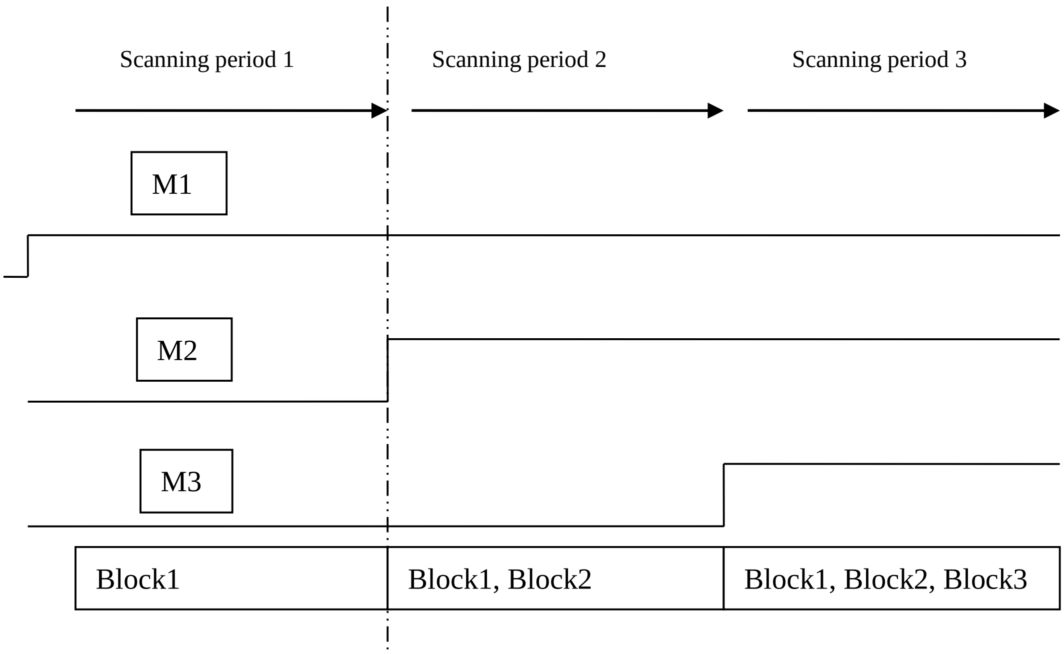
FROM\TO instruction can be selected from pull-down list:



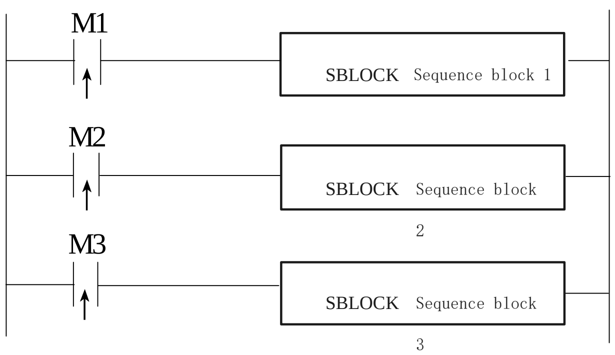
10-4. Running form of the BLOCK

1. If there are many blocks, they run as the normal program. The block is running when the condition is ON.
(A) The condition is normal ON, normal OFF coil





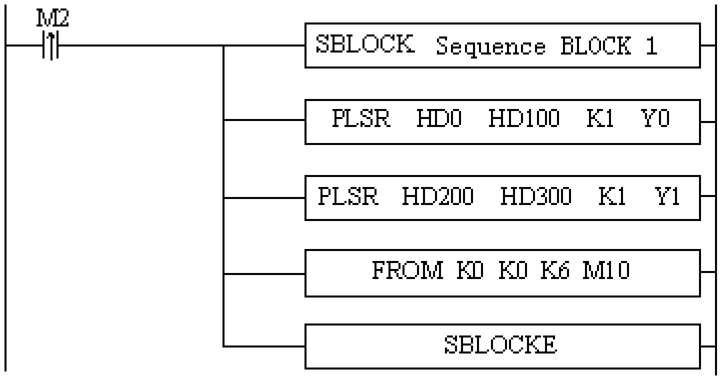
(B) The condition is rising or falling edge of pulse



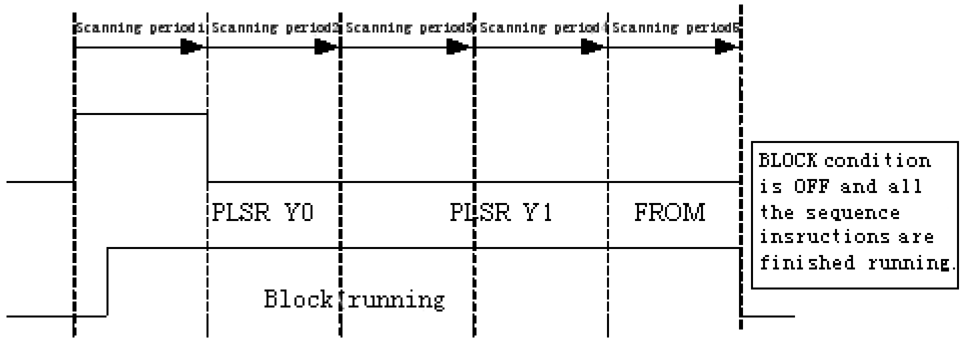
When M1, M2, M3 is from OFF to ON, all these blocks will run once.

2. The instructions in the block run in sequence according to the scanning time. They run one after another when the condition is ON.

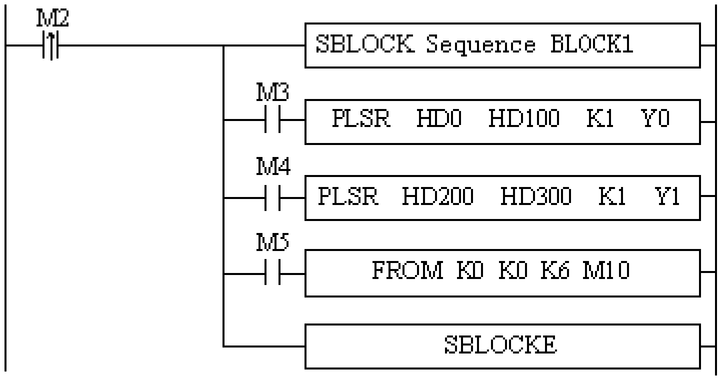
(A) Without SKIP condition



The instructions running sequence in block 1 is shown as below:



(B) With SKIP condition



Explanation:

A) When M2 is ON, block 1 is running.

B) All the instructions run in sequence in the block.

C) M3, M4, M5 are the sign of SKIP, when they are ON, this instruction will not run.

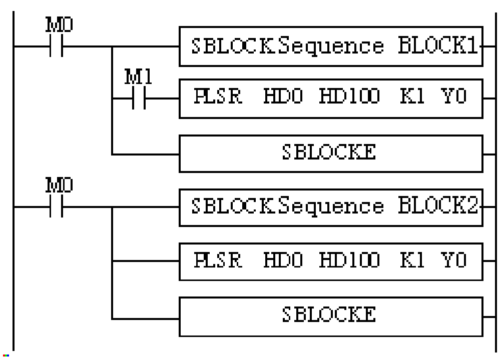
D) When M3 is OFF, if no other instructions use this Y0 pulse , DPLSR D0 D2 D4 Y0 will run; if not, the DPLSR D0 D2 D4 Y0 will run after it is released by other instructions.

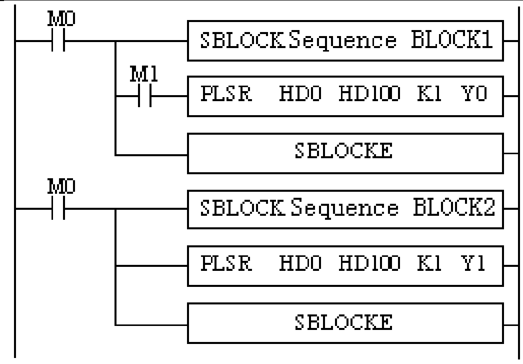
E) After “DPLSR D0 D2 D4 Y0” is over, check M4. If M4 is OFF, check “DPLSR D0 D2 D4 Y1”, if M4 is ON, check M5. If M5 is OFF, “inverter config” will run.

10-5. BLOCK instruction editing rules

In the BLOCK, the instruction editing should accord with some standards.

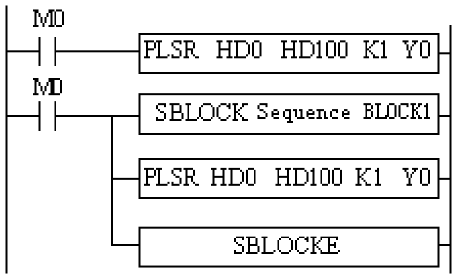
Do not use the same pulse output terminal in different BLOCK.

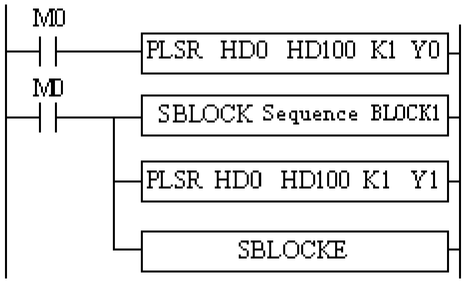
<table>
<tr><th>NO (×)</th><th>YES (√)</th></tr>
<tr><td>


</td><td>


</td></tr>
</table>

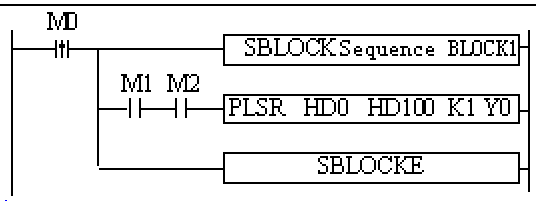
Do not use the same pulse output terminal in BLOCK and main program.

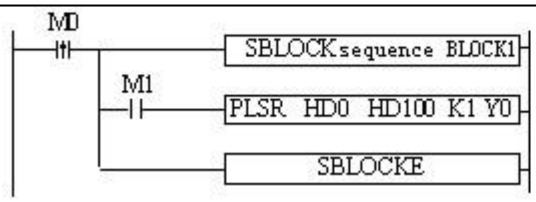
<table>
<tr><th>NO (×)</th><th>YES (√)</th></tr>
<tr><td>


</td><td>


</td></tr>
</table>

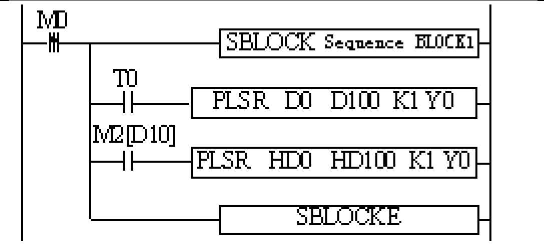
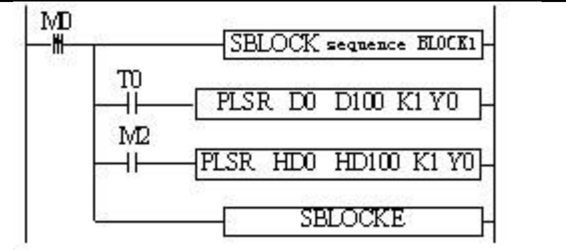
There only can be one SKIP condition for one BLOCK instruction.

<table>
<tr><th>NO (×)</th><th>YES (√)</th></tr>
<tr><td>


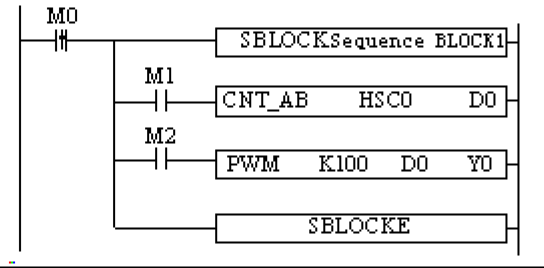
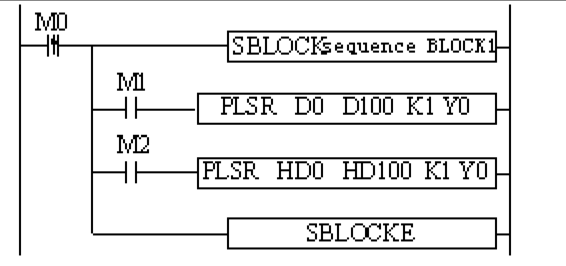
</td><td>


</td></tr>
</table>

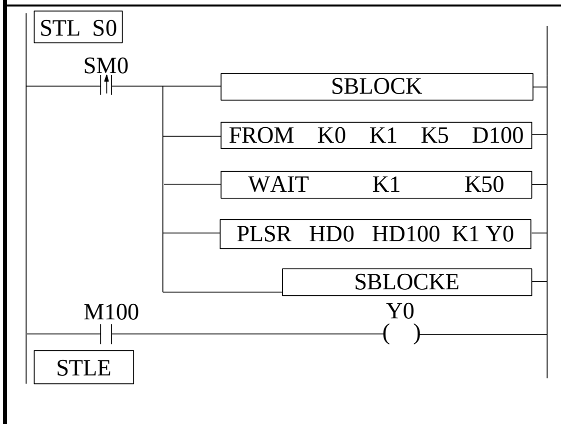
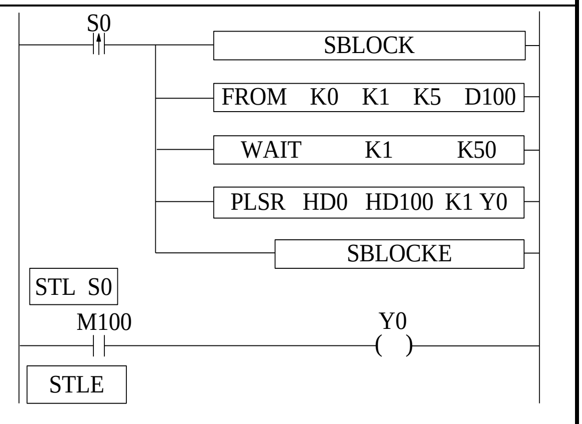
The SKIP condition only can use M, X, can not use other coil or register.

NO（×）	**YES**（√）
	

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.

10-6. BLOCK related instructions

10-6-1. Instruction explanation

stop running the BLOCK [SBSTOP]

Summarization

Stop the instructions running in the block

[SBSTOP]			
16 bits	SBSTOP	32 bits	-
Condition	NO,NC coil and pulse edge	Suitable types	XD3, XDM
Hardware		Software	V3.2

Operand

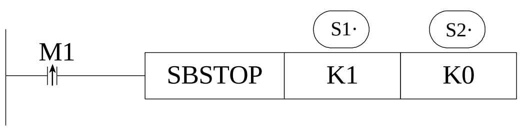
Operand	Function	Type
S1	The number of the BLOCK	16bits, BIN
S2	The mode to stop the BLOCK	16bits, BIN

Suitable component

	Operand	Register								Constant	Module	
		D*	FD	TD*	CD*	DX	DY	DM*	DS*	K/H	ID	QD
Word	S1	●								●		
	S2									●		

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

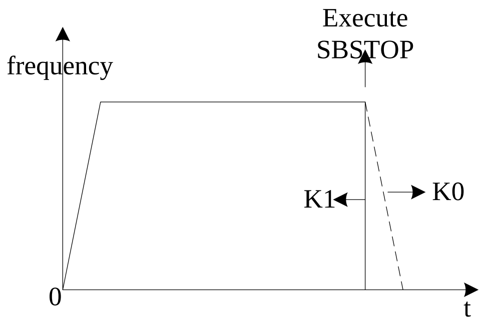
Function



S2 is the mode for BLOCK stop, operand: K0, K1

K0: stop the BLOCK slowly, if the pulse is outputting, the BLOCK will stop after the pulse outputting is finished.

K1: stop the BLOCK immediately; stop all the instructions running in the BLOCK.



Continue running the BLOCK[SBGOON]

Summarization

This instruction is opposite to SBSTOP. To continue running the BLOCK.

[SBGOON]			
16 bits	SBGOON	32 bits	-
Condition	Pulse edge	Suitable types	XD3, XDM
Hardware	-	Software	V3.2

Operand

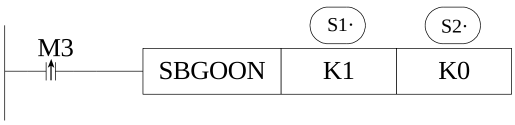
Operand	Function	Type
S1	The number of the BLOCK	16 bits, BIN
S2	The mode to continue running the BLOCK	16 bits, BIN

Suitable component

Word	Operand	Register								Constant	Module	
		D*	FD	TD*	CD*	DX	DY	DM*	DS*	K/H	ID	QD
	S1	●								●		
	S2									●		

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

Function



S2 is the mode to continue running the BLOCK. Operand: K0, K1.

K0: continue running the instructions in the BLOCK.

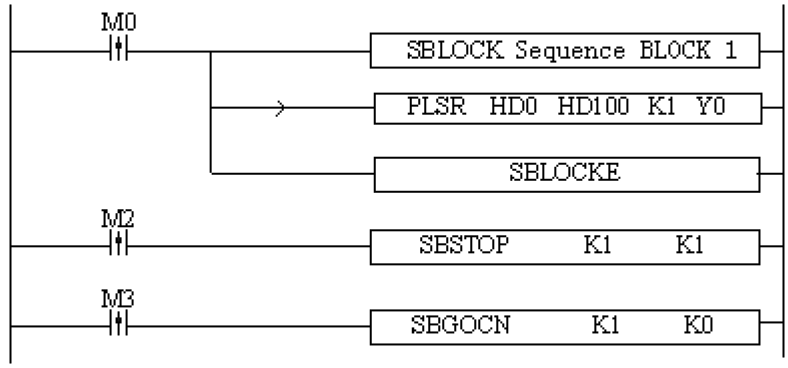
For example, if pulse outputting stopped last time, SBGOON will continue outputting the rest pulse;

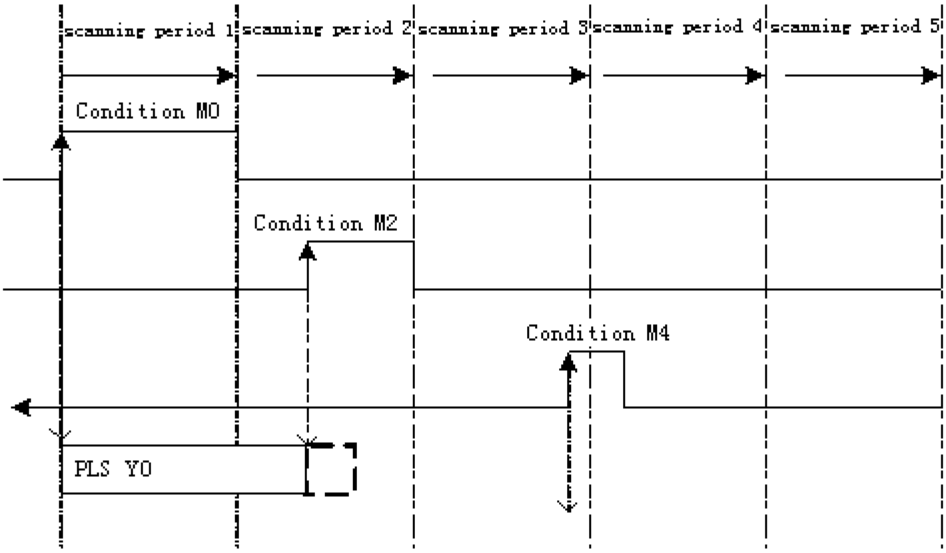
K1: continue running the BLOCK, but abandon the instructions have not finished last time.

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.

10-6-2. The timing sequence of the instructions

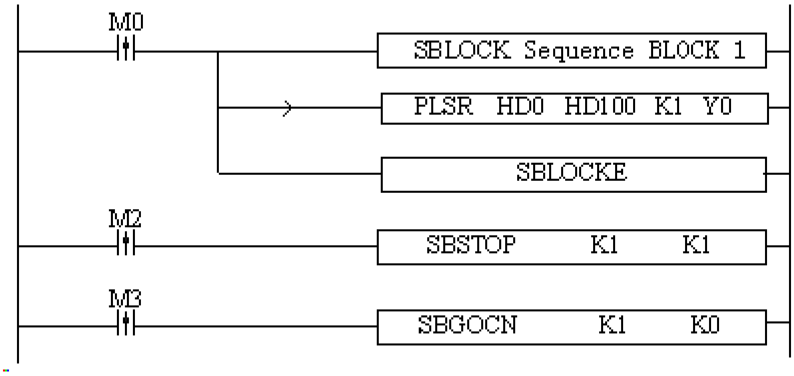
SBSTOP (K1 K1) + SBGOON (K1 K1)

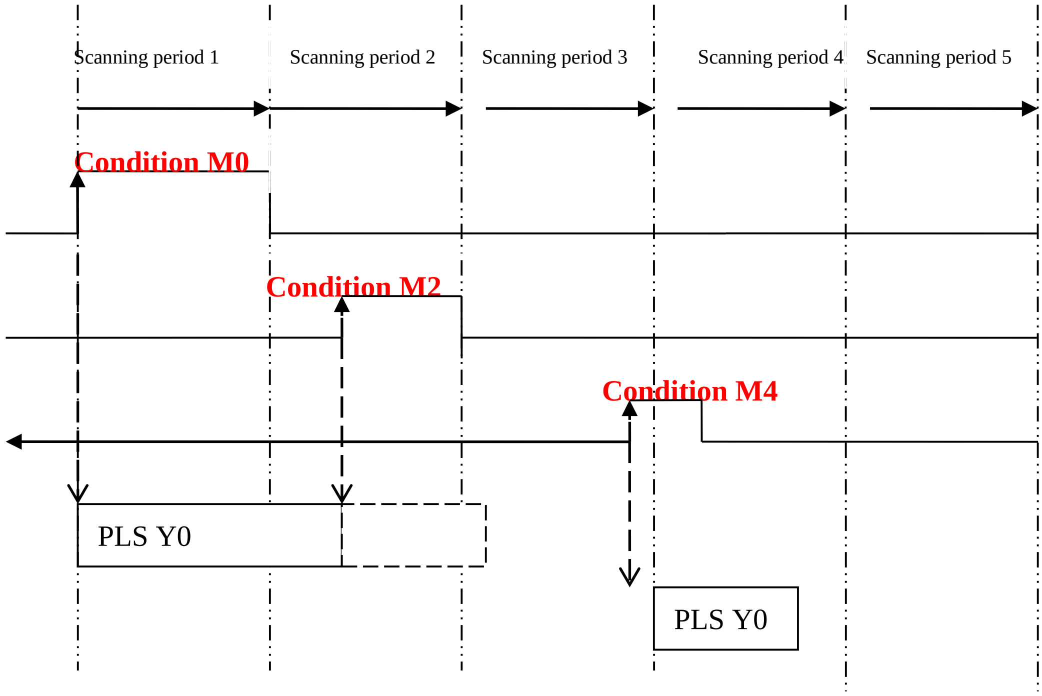




When M0 is from OFF→ON, run “PLSR HD0 HD100 K1 Y0” in the BLOCK to output the pulse;
When M2 is from OFF→ON, the BLOCK stops running at once;
When M4 is from OFF→ON, abandon the rest pulse.

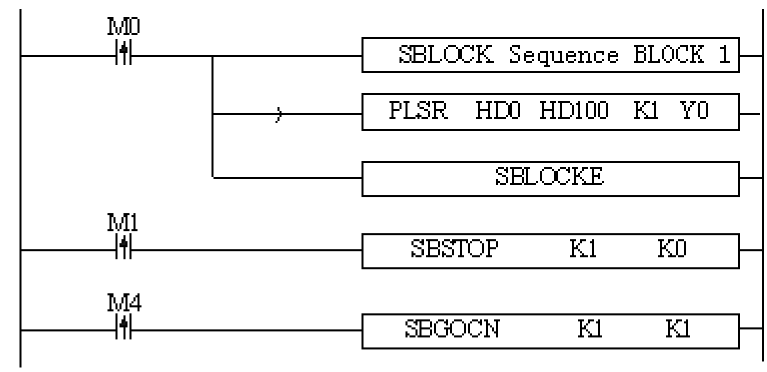
SBSTOP（K1 K1）+SBGOON（K1 K0）

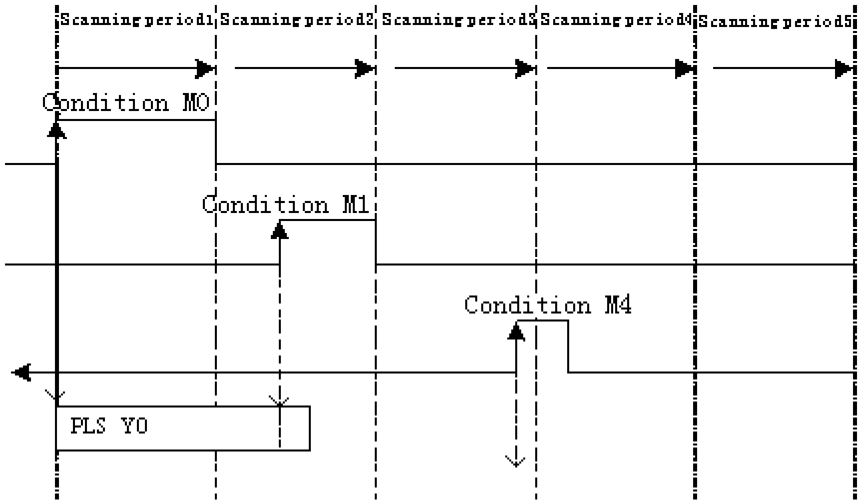




When M0 is OFF→ON, run ‘PLSR HD0 HD100 K1 Y0’ in the BLOCK to output the pulse;
When M2 is OFF→ON, the BLOCK stops running, the pulse output stops at once;
When M4 is OFF→ON, output the rest pulses.

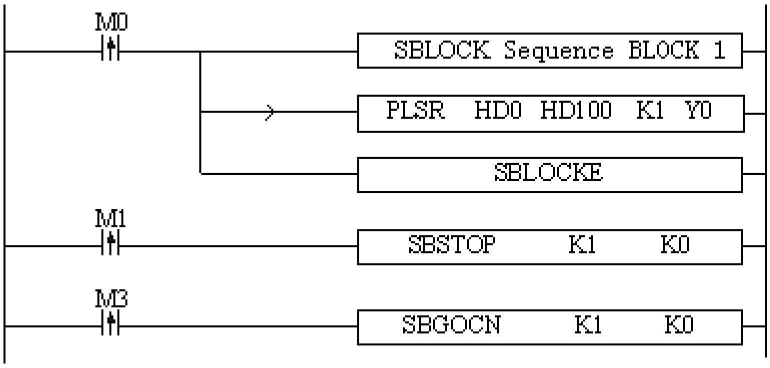
3. SBSTOP(K1 K0)+SBGOON(K1 K1)

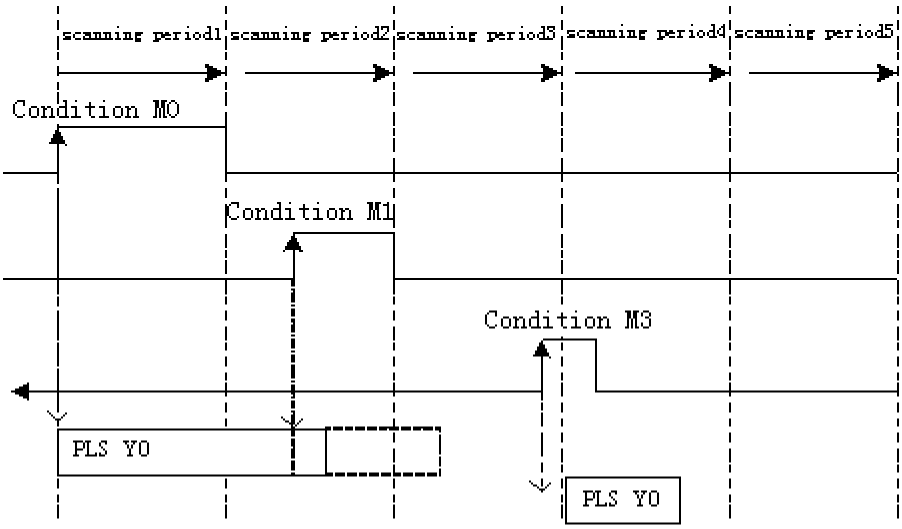




When M0 is from OFF→ON, run ‘PLSR HD0 HD100 K1 Y0’ in the BLOCK to output the pulse;
When M1 is from OFF→ON, stop running the BLOCK, the pulse will stop slowly with slope;
When M4 is from OFF→ON, abandon the rest pulses.

4. 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 again, it will output the rest of the pulses.

10-7. 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	

11 Special Function Instructions

This chapter mainly introduces PWM (pulse width modulation), 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	11-1
FRQM	Fixed pulses frequency measurement	FRQM S1 D S2 S3	11-2
Time			
STR	Precise Time	STR D1 D2	11-3
Interruption			
EI	Enable Interruption	EI	11-4-1
DI	Disable Interruption	DI	11-4-1
IRET	Interruption Return	IRET	11-4-1

11-1. Pulse Width Modulation [PWM]

1. Instruction's Summary

Instruction to realize PWM pulse width modulation

PWM pulse width modulation [PWM]			
16 bits instruction	PWM	32 bits instruction	-
execution condition	normally ON/OFF coil	suitable models	XD3, XDM
hardware requirement	-	software requirement	-

2. Operands

Operands	Function	Type
S1	specify the duty cycle value or soft component's ID number	16 bits, BIN

S2	specify the output frequency or soft component's ID number	16 bits BIN
D	specify the pulse output port	bit

3. Suitable Soft Components

Word

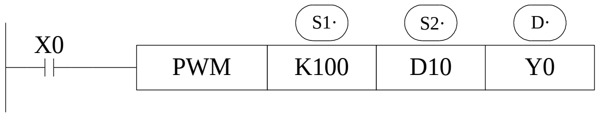
Operands	System									Constant	Module	
	D*	FD	ED	TD*	CD*	DX	DY	DM*	DS*	K/H	ID	QD
S1	●	●		●	●					●		
S2	●	●		●	●					●		

Bit

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

*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 HS ; T includes T HT ; C includes C HC

Function and



Duty cycle **n**: 1~32767

Output pulse **f**: 1~200KHz

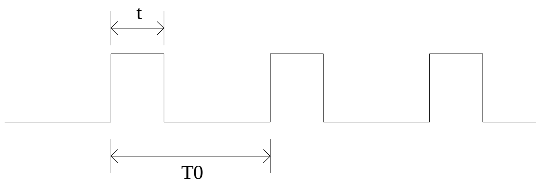
XD, XDM series PLC PWM output need transistor type terminal:

PLC model	PWM terminal
XD3-16T/RT -24T/RT -32T/RT -48T/RT -60T/RT	Y0, Y1
XDM-24T-32T-60T4-60T10	Y0, Y1, Y2, Y3

Duty cycle of **PWM** output =n /32768×100%

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.



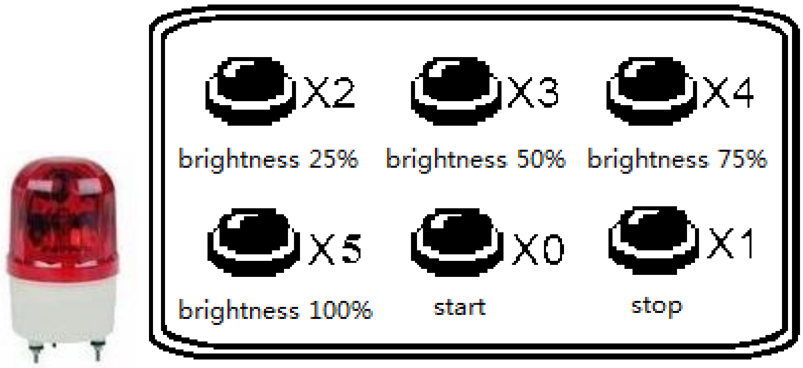
In the left graph:

T0=1/f

t/T0=n/32768

Note: it needs to connect 1K ohm amplification resistor between output terminal and common terminal when using PWM instruction.

Example



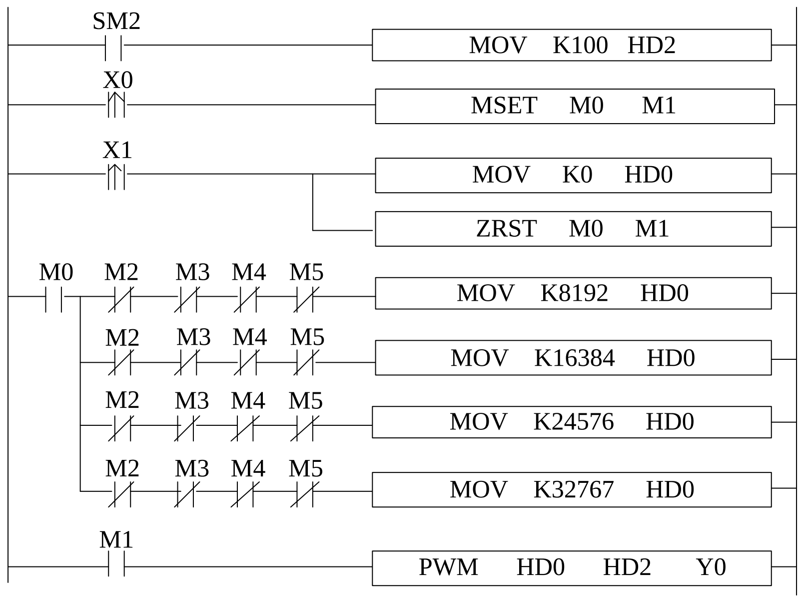
There is a LED drived by DC24V. It needs to control the brightness of the LED. In order to decrease the power loss of wave collector, turn ON the switch at the moment it is OFF, then turn it OFF. This process will cycle. Connet a transistor between the power supply and LED. The pulse signal will input 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*HD0/32767, 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=8192, HD0/32768=0.25, the LED brightness is 25%.
4. X3 is ON, HD0=16384, HD0/32768=0.5, the LED brightness is 50%.
5. X4 is ON, HD0=24576, HD0/32768=0.75, the LED brightness is 75%.
6. X5 is ON, HD0=32768, 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.

11-2. Frequency measurement [FRQM]

1. Instruction list

Measure the frequency.

Frequency measurement [FRQM]			
16 bits instruction	-	32 bits instruction	FRQM
execution condition	Normally ON OFF coil	suitable models	XD3, XDM
hardware requirement	-	software requirements	-

2. Operand

Operands	Function	Type
S1	Sampling pulse numbers	32 bits, BIN
S2	Frequency division option	32 bits, BIN
D	Measurement result	32 bits, BIN
S3	Pulse input terminal	bit

3. Suitable component

Word

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

Bit

Operand	System						
	X	Y	M*	S*	T*	C*	Dn.m
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, HS; T includes T, HT; C includes C, HC.

Function and Action

M0	FRQM	K20 (S1·)	D100 (D·)	X0 (S3.)	K1 (S2.)

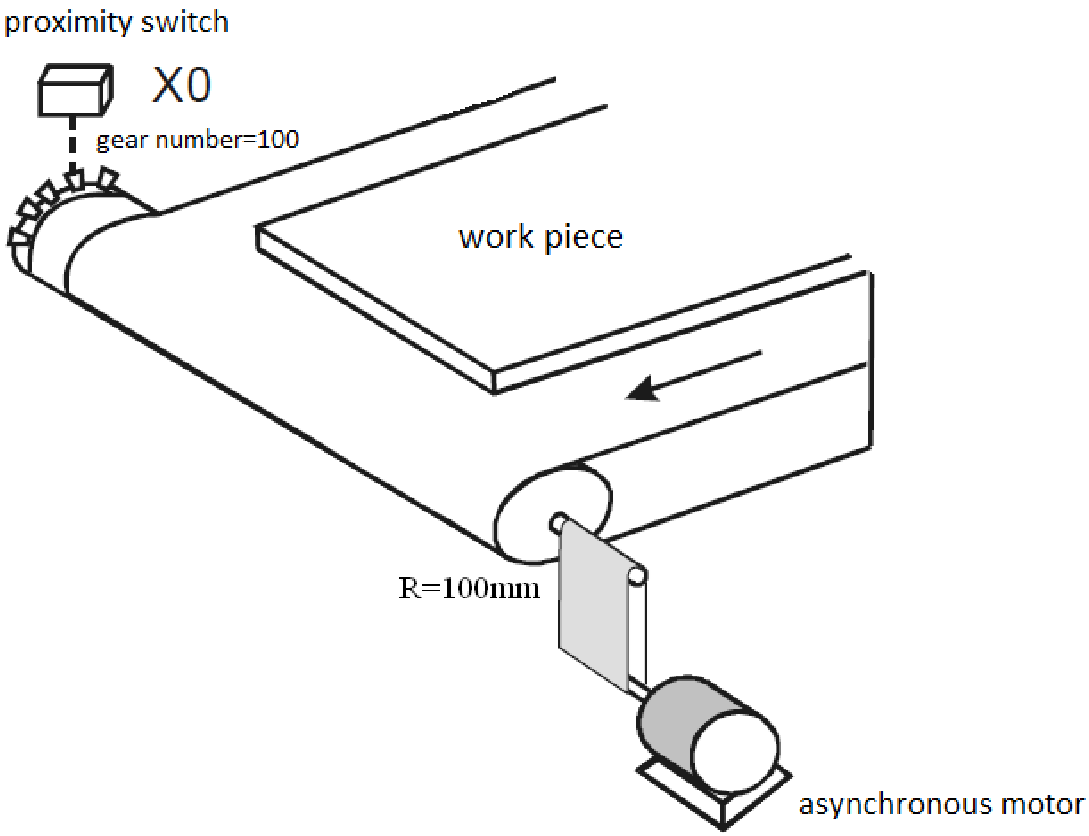
- The sampling pulse numbers can be adjusted according to the frequency, the higher the frequency, the bigger the sampling pulse numbers
- Measurement result, the unit is Hz
- Frequency division option, the range is 1 or 2. Whatever 1 or 2, the frequency range is 1Hz to 80KHz
- When M0 is ON, FRQM collects 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:

Model		X terminal	Max frequency
XD3	16	X0	80KHz
		X3	10KHz
	24/32	X0	80KHz
		X3	10KHz
		X6	10KHz
	48/60	X0	80KHz
		X3	10KHz
		X6	10KHz
XDM	24/32/60T4/60T10	X0	80KHz
		X3	80KHz
		X6	80KHz
		X11	80KHz

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:

```
M0
--| |--+--[FRQM  K20   D0    X0   K1]
       +--[DFLT  D0    D2]
       +--[EDIV  D2    K100  D4]
       +--[EMUL  K0.1  K3.14 D6]
       +--[EMUL  D4    D6    D10]
       +--[EMUL  D10   K60   D16]
```

Program explanation:

1. Set ON the start signal M0, to run the frequency measurement 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).

11-3. Precise Timing [STR]

1. Instruction List

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	XD3, XDM
hardware requirement	-	software requirements	-

2. Operands

Operands	Function	Type
D1	Timer Number	bit
D2	specify timer’s value or soft component’s ID number	16 bits, BIN

3. Suitable Soft Components

Word

Operands	system									constant	module	
	D*	FD	ED	TD*	CD*	DX	DY	DM*	DS*	K/H	ID	QD
D2	●	●		●	●					●		

Bit

Operands	system						
	X	Y	M*	S*	T*	C*	Dn.m
D					●		
D1					●		

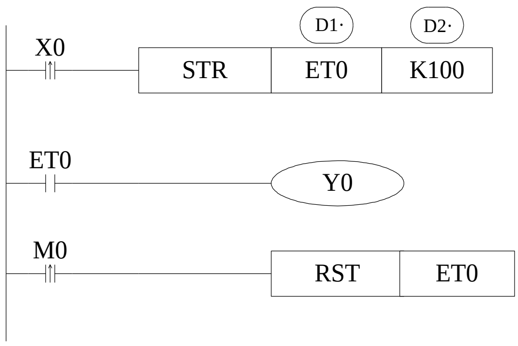
*Note: D includes D HD; TD includes TD HTD; CD includes CD HCD HSCD HSD; DM includes DM DHM;

DS includes DS DHS.

M includes M HM SM； S includes S HS ； T includes T HT ； C includes C HC.

Function and Action

<Precise timing>, <Precise timing reset>



D1· Timer’s number. Range: ET0~ET30 (ET0, ET2, ET4……all number should be even)

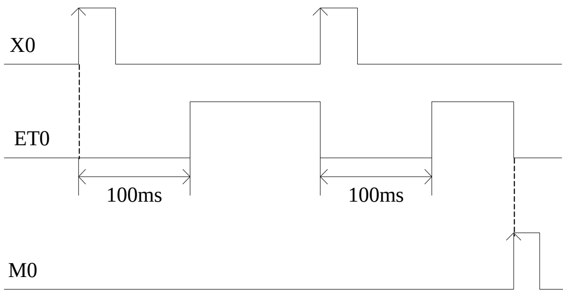
D2· Timing value

Precise timer works in unit of 1ms.

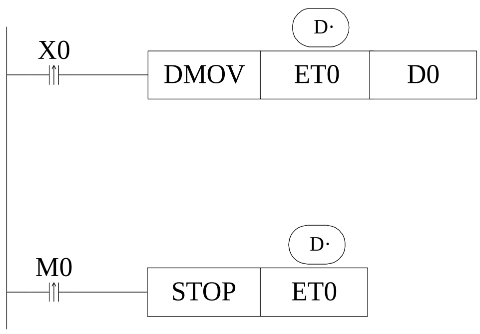
Precise timer 32 bits, the counting range is 0~+2,147,483,647.

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:



<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;

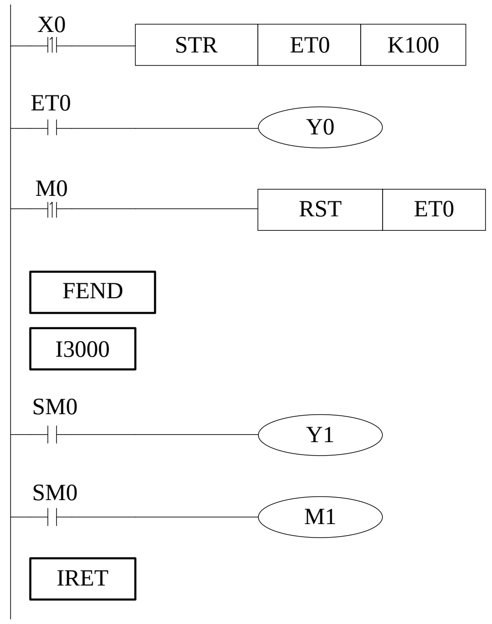
Precise Timing Interruption

- 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

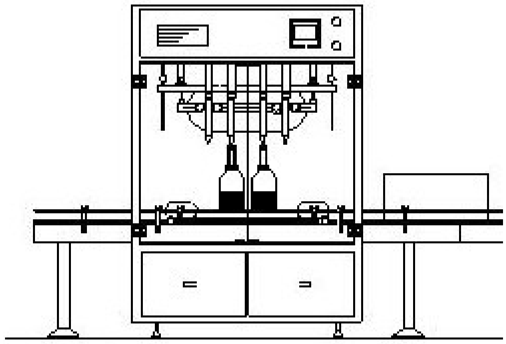
Note: XD3-16, XD3-24, XD3-32, XD3-48, XD3-60, XDM-24, XDM-32, XDM-60T10 timer number are ET0, ET2, ET4, ET6……ET24; XDM-60T4 timer number is ET0, ET2, ET4, ET6……ET22.



When X0 changes from OFF to ON, ET0 will start timing. And ET0 reset when accumulation time is up to 100ms; meantime generate 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.

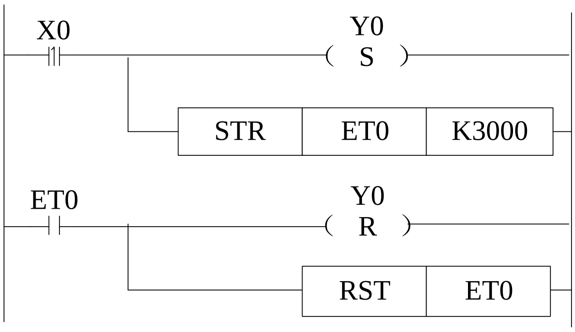


Filling machine

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:

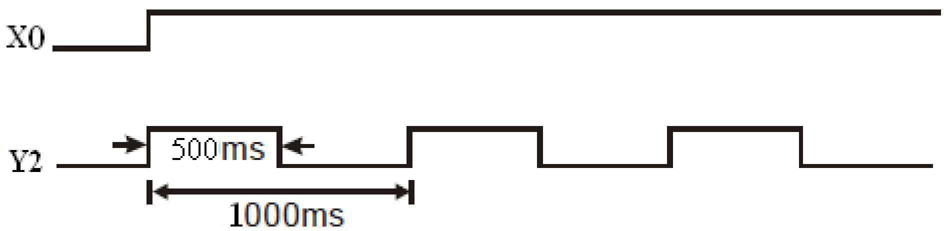


Program explanation:

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

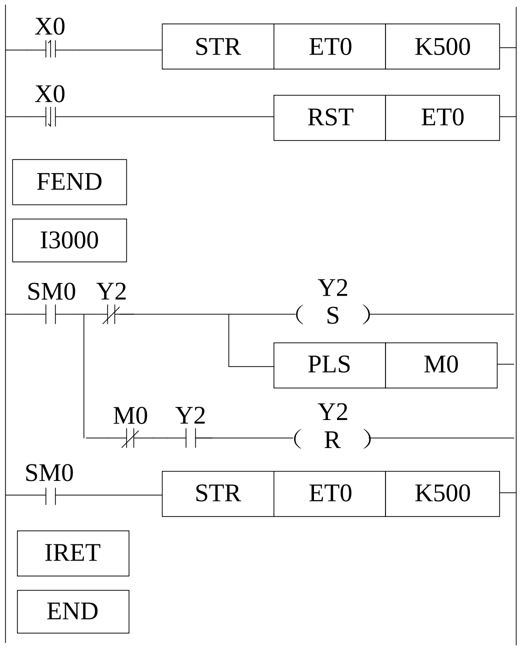
The precise timer interruption can produce the following pulse wave. The Y2 ON time is 500ms, the pulse period is 1000ms.



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:

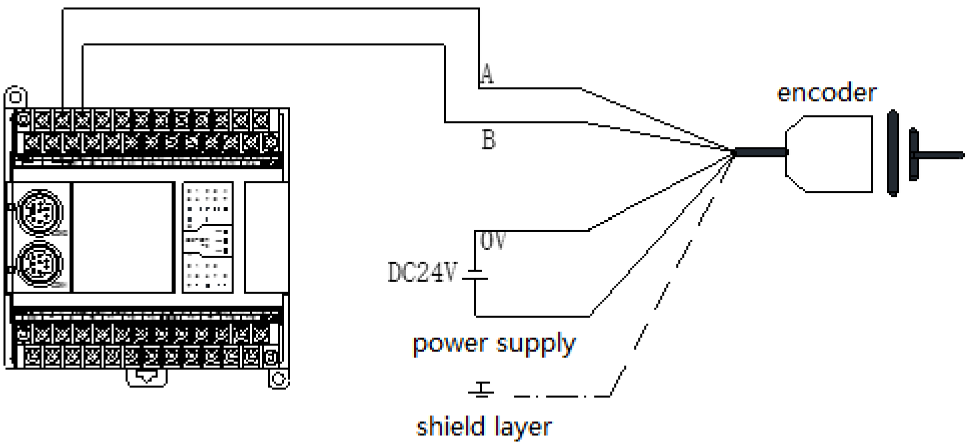


Program explanation:

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

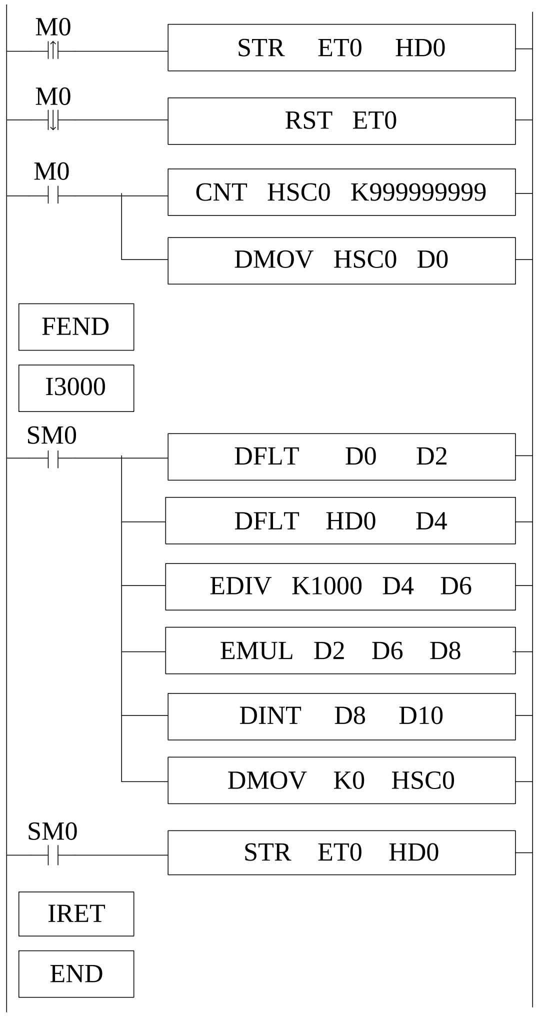
As the FRQM calculating the time for fixed pulse numbers, we will change the way to calculate the pulse numbers in fixed time.



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:

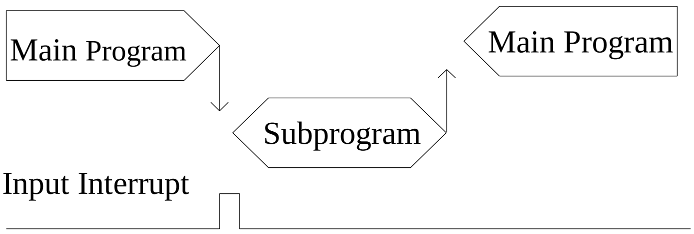
1. Set the high speed counter sampling period register HD0, the unit is ms.
2. Set ON M0 to start the precise timer interruption and high speed counter, calcuate the frequency
3. The frequency range is 0-80KHz, the precision is 0.005%.

11-4. Interruption [EI], [DI], [IRET]

XD3 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.

11-4-1. External Interruption

The input terminals X can be used to input external interruption. Each input terminal corresponds with one external interruption. The input's rising/falling edge can activate the interruption. The interruption subroutine is written behind the main program (behind FEND). After interruption generates, the main program stops running immediately, turn to run the correspond subroutine. After subroutine running ends, continue to execute the main program.



Note: The external interruption of XC series PLC cannot be activated by rising edge and falling edge at the same time; but XD3 series PLC supports rising edge and falling edge activation meantime.

External Interruption's Port Definition

XD3-16 points

Input terminal	Pointer No.		Disable the interruption instruction
	Rising Interruption	Falling interruption	
X2	I0000	I0001	SM050
X3	I0100	I0101	SM051
X4	I0200	I0201	SM052
X5	I0300	I0301	SM053
X6	I0400	I0401	SM054
X7	I0500	I0501	SM055

XD3-24/32/48/60 and XDM-24/32/60T4/60T10

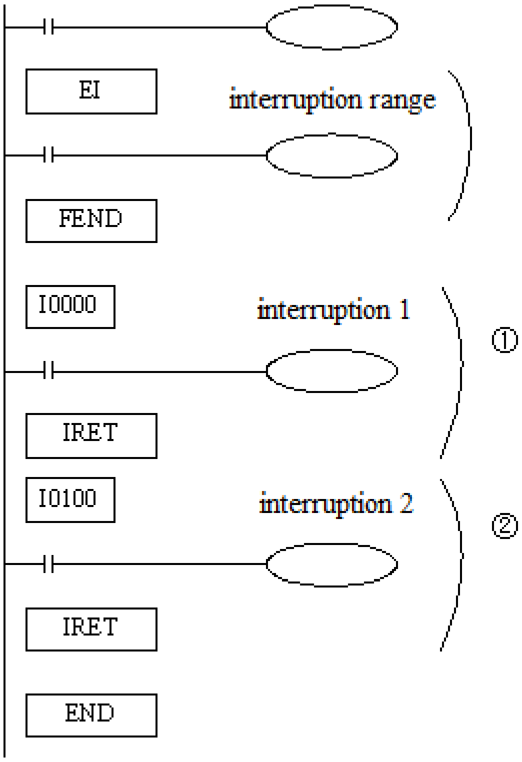
Input terminal	Pointer No.		Disable the interruption instruction
	Rising Interruption	Falling interruption	
X2	I0000	I0001	SM050
X3	I0100	I0101	SM051
X4	I0200	I0201	SM052
X5	I0300	I0301	SM053
X6	I0400	I0401	SM054

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.

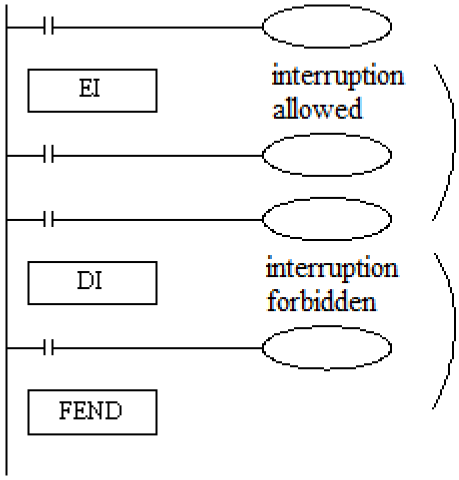
Interruption Instruction

Enable Interruption [EI], Disable Interruption [DI], Interruption Return [IRET]



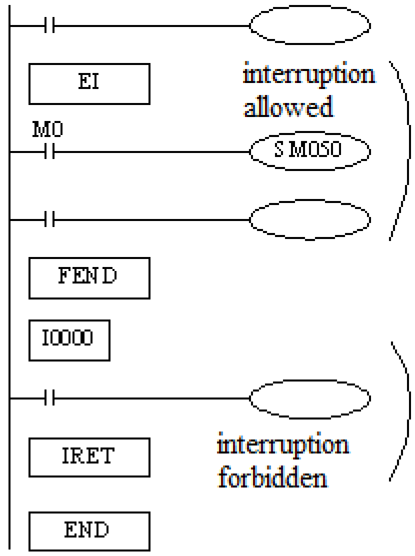
- If use EI instruction to allow interruption, then when scanning the program, if interruption input changes from OFF to ON, then execute subroutine ①、②. Return to the original main program.
- Interruption pointer (I****) should be behind FEND instruction;
- PLC is usually on the status that allows interruption.

Interruption's Range Limitation



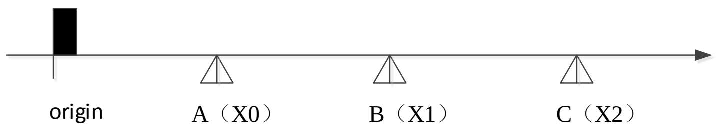
- By programming DI instruction, can set interruption disabled area;
- Allow interruption input between EI~DI
- If interruption forbidden is not required, please program only with EI, and program with DI is not required.

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.

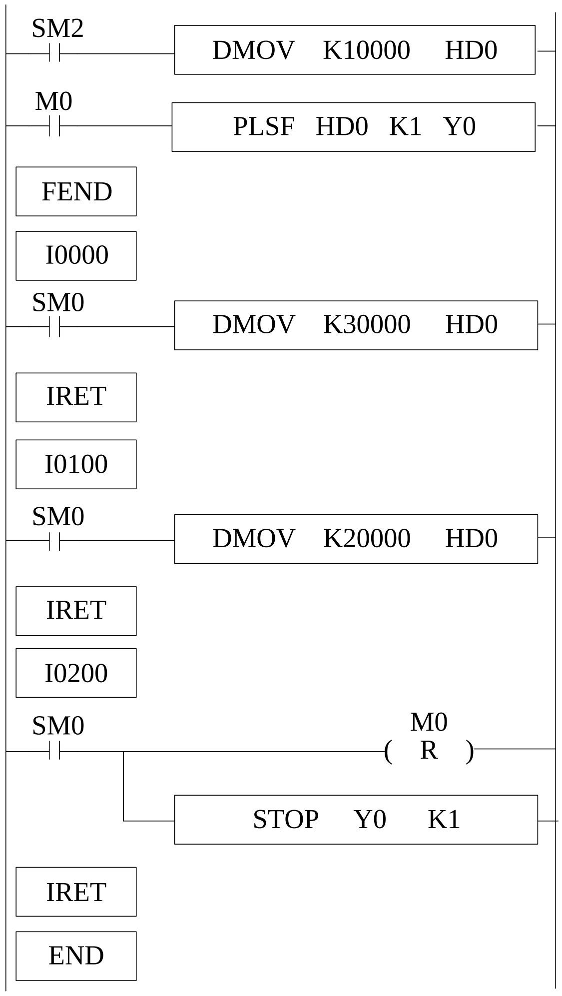
Segment	Frequency setting value (Hz)	Pulse numbers
Origin ---- A	10000	999999999
A---- B	30000	999999999
B ----- C	20000	999999999
Acceleration and deceleratoin time	0	

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

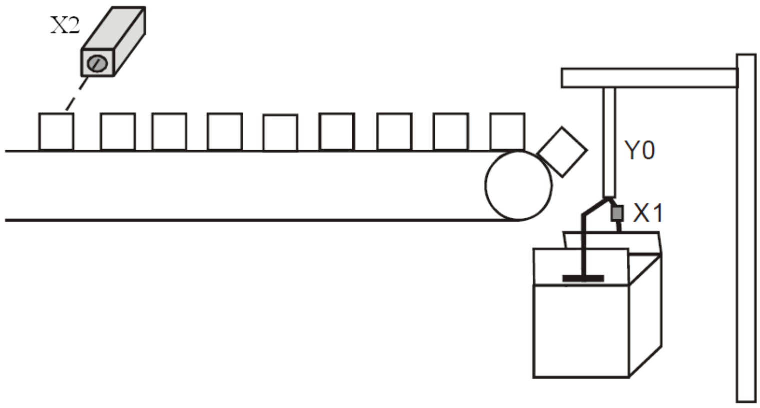


Program explanation

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

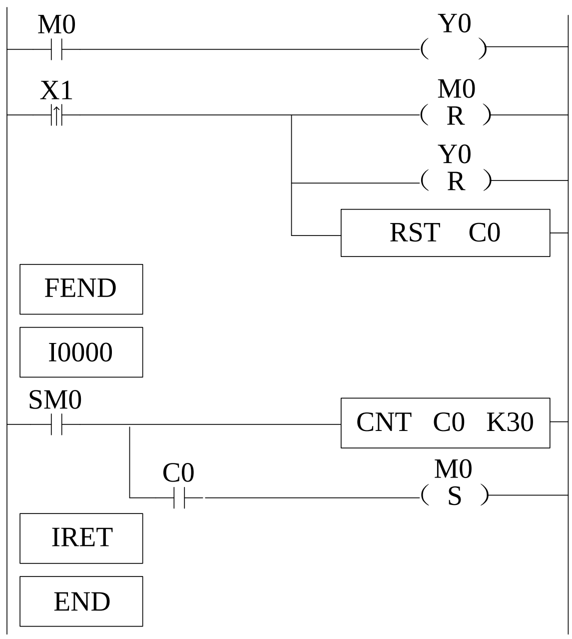
The diagram is the product packing machine. The robot will pack the product when 30 products are detected, the robot and counter will be reset after packing completed. To improve the working efficiency, the product sending speed is very fast, the sensor X2 detects the product time is 8ms, PLC input terminal 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:



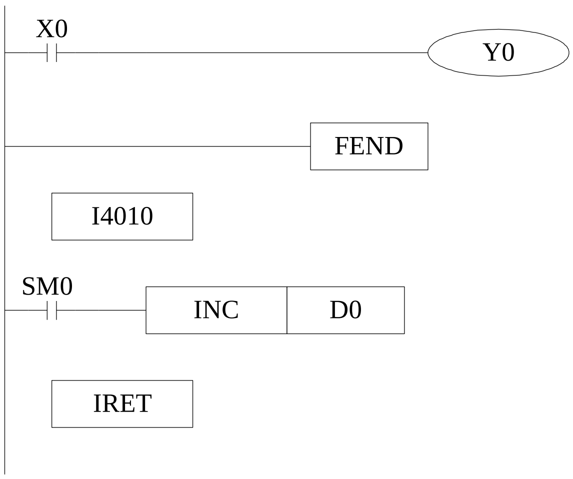
Program explanation:

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.

11-4-2. 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 20CH 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

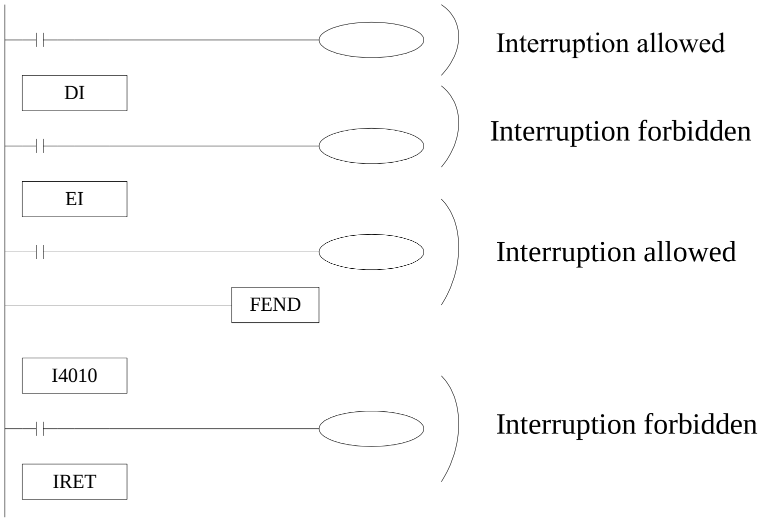
XD3 and XDM series timing interruption:

Interruption number	Interruption ban instruction	Interruption number	Interruption ban instruction	Explanation

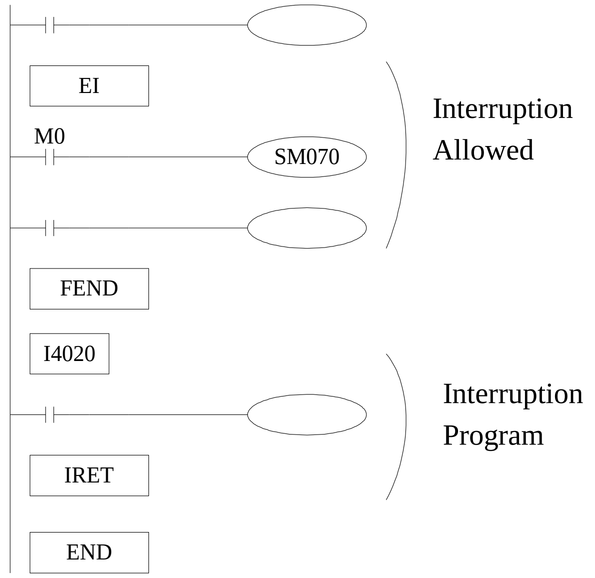
I40**	SM070	I50**	SM080	** means the timing interruption time, the range is 1~99, the unit is ms.
I41**	SM071	I51**	SM081	
I42**	SM072	I52**	SM082	
I43**	SM073	I53**	SM083	
I44**	SM074	I54**	SM084	
I45**	SM075	I55**	SM085	
I46**	SM076	I56**	SM086	
I47**	SM077	I57**	SM087	
I48**	SM078	I58**	SM088	
I49**	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.

12 Application Example

This chapter mainly introduces main instructions, such as pulse output and Modbus communication instructions, and gives some examples.

12-1. Pulse output application

There are 3 segments of pulse. Y0 is pulse output terminal, Y2 is direction output terminal.

Segment	Frequency (Hz)	Pulse numbers
1	3000	1000
2	800	2000
3	6000	8000
Acceleration and deceleration	The frequency changes 1000Hz every 100ms	

Pulse parameters

Address	Explanation	Value
HD0 (dword)	Pulse segment quantity (1~100)	3
HD2 (8 words)	Reserved (8 words)	0
HD10 (dword)	Pulse frequency (segment 1)	3000
HD12 (dword)	Pulse quantity (segment 1)	1000
HD14	bit15~bit8: wait condition (segment 1) 00: pulse sending end 01: wait time 02: wait signal 03: ACT time 04: EXT signal 05: EXT signal or pulse sending end bit7~bit0: wait condition register type 00: constant 01: D 02: X 03: M	0
HD15 (dword)	Constant/register(waiting condition) (segment 1)	0
HD17	bit7~bit0: jump register type 00: constant 01: D	0
HD+18 (dword)	Constant/register (jump register) (segment 1)	0
HD+20 (dword)	Pulse frequency (segment 2)	800
HD+22 (dword)	Pulse quantity (segment 2)	2000
HD+24	Wait condition, wait condition register type (segment 2)	0

HD+25 (dword)	Constant or register (waiting condition) (segment 2)	0
HD+27	Jump type, jump register type (waiting condition) (segment 2)	0
HD+28 (dword)	Constant or register (jump register) (segment 2)	0
HD+30 (dword)	Pulse frequency (segment 3)	6000
HD+32 (dword)	Pulse quantity (segment 3)	8000
HD+34	Wait condition, wait condition register type (segment 3)	0
HD+35 (dword)	Constant or register (waiting condition) (segment 3)	0
HD+37	Jump type, jump register type (waiting condition) (segment 3)	0
HD+38 (dword)	Constant or register (jump register) (segment 3)	0

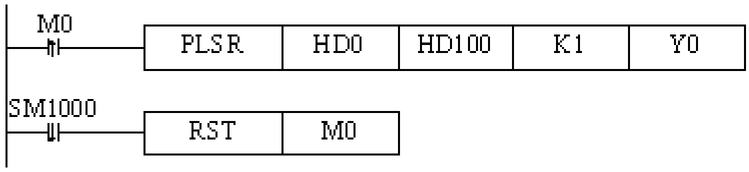
System parameters:

SFD900	Pulse parameters	Bit 0: pulse output logic 0: positive logic(default setting) 1: negative logic, Bit 1：pulse direction logic 0: positive logic(default setting) 1: negative logic Bit 8: pulse unit 0: pulse quantity(default setting) 1: equivalent value	0	Public parameter
SFD901	Reserved		0	
SFD902	Pulse quantity/ 1 rotation low 16 bits		0	
SFD903	Pulse quantity/ 1 rotation high 16 bits		0	
SFD904	Movement amount/1 rotation low 16 bits		0	
SFD905	Movement amount/1 rotation high 16 bits		0	
SFD906	Pulse direction terminal	The number of Y, 0xFF is no terminal	2	
SFD907	Direction delay time	Default value is 20, unit is ms	20	
SFD908	Gear clearance positive compensation		0	
SFD909	Gear clearance negative compensation		0	
SFD910	Electrical origin low 16 bits		0	
SFD911	Electrical origin high 16 bits		0	
SFD912	Machine back to origin parameters	Bit0: promixity switch state 0: normal open 1: normal close	0	

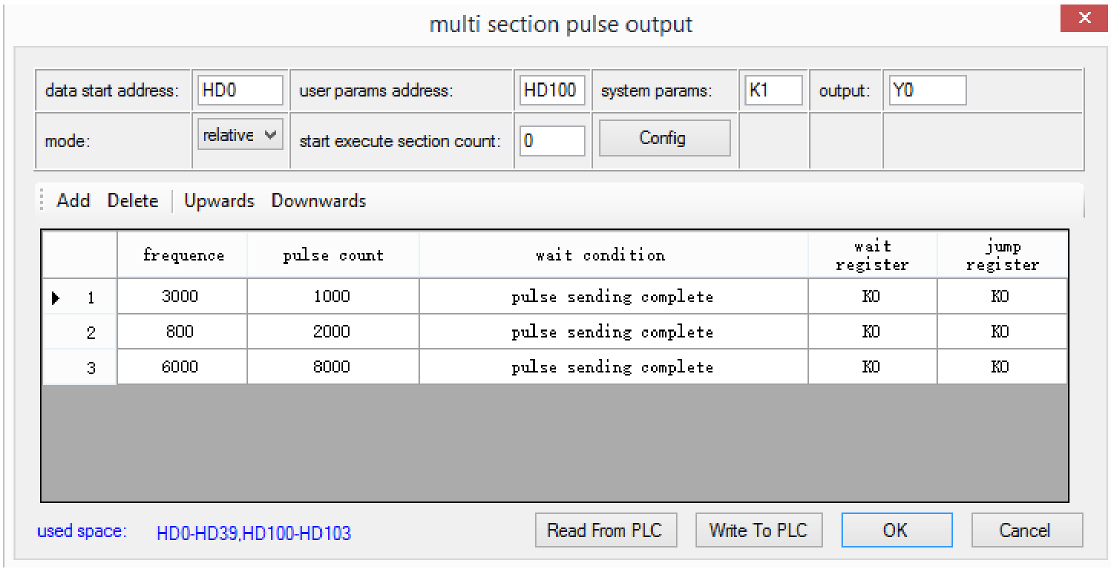
SFD913	Near signal terminal	Bit0~bit7: the X terminal number, 0xFF is no terminal	0xFF	
SFD914	Z phase terminal	Bit0~bit7: the X terminal number, 0xFF is no terminal	0xFF	
SFD915	Limit terminal	Bit7~bit0: limit 1 X terminal number, 0xFF is no terminal Bit15~bit8: limit 2 X terminal number, 0xFF is no terminal	FFFF	
SFD916	Origin auxiliary signal terminal	Bit0~bit7: X terminal number, 0xFF is no terminal	0xFF	
SFD917	CLR signal output terminal	Bit0~bit7: Y terminal number, 0xFF is no terminal	0xFF	
SFD918	Back speed VH low 16 bits		0	
SFD919	Back speed VH high 16 bits		0	
SFD920	Back speed VL low 16 bits		0	
SFD921	Back speed VL high 16 bits		0	
SFD922	Creep speed low 16 bits		0	
SFD923	Creep speed high 16 bits		0	
SFD924	Mechanical origin low 16 bits		0	
SFD925	Mechanical origin high 16 bits		0	
SFD926	Z phase quantity		0	
SFD927	CLR signal delay time	Default value 20, unit: ms	20	
SFD936	G instruction which group of parameter to use	G instruction system parameter group choice	2	
…				
SFD950	Pulse default speed low 16 bits	Send pulse with default speed when speed is 0	1000	Group 1 parameters
SFD951	Pulse default speed high 16 bits		0	
SFD952	Pulse default speed acceleration time		100	
SFD953	Pulse default speed deceleration time		100	
SFD954	Tween acceleration/deceleration time		0	
SFD955	Reserved		0	
SFD956	Max speed limit low 16 bits		3392	
SFD957	Max speed limit high 16 bits		3	
SFD958	Start speed low 16 bits		0	

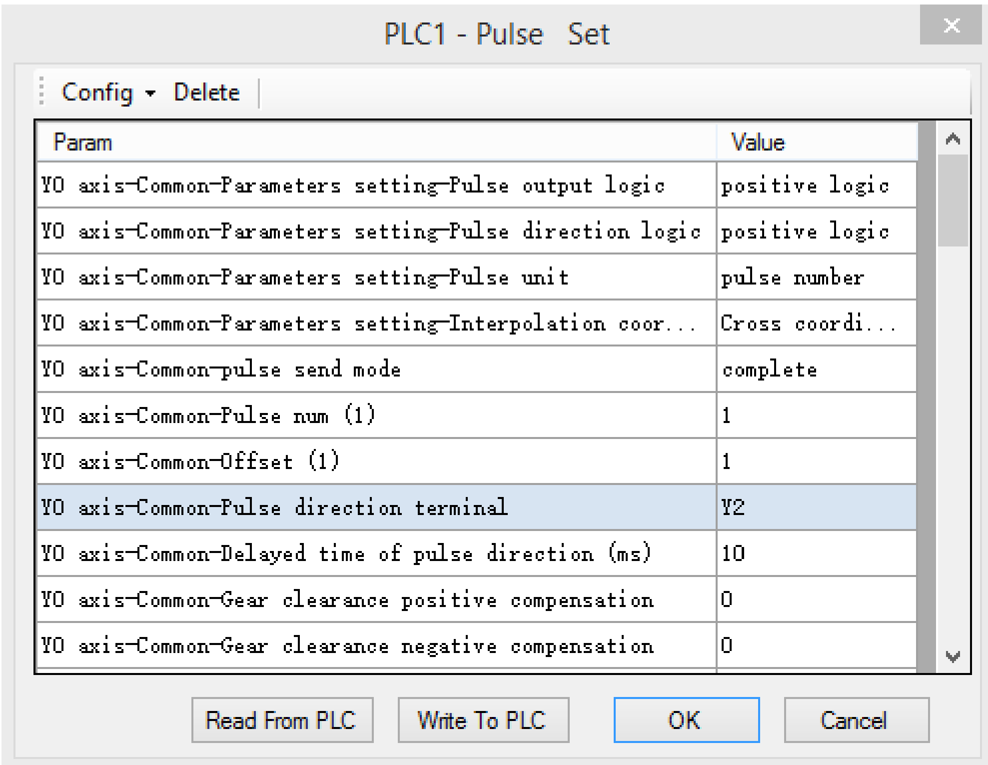
SFD959	Start speed high 16 bits		0	
SFD960	End speed low 16 bits		0	
SFD961	End speed high 16 bits		0	
SFD962	Curve acceleration time (ms)		2	
...				

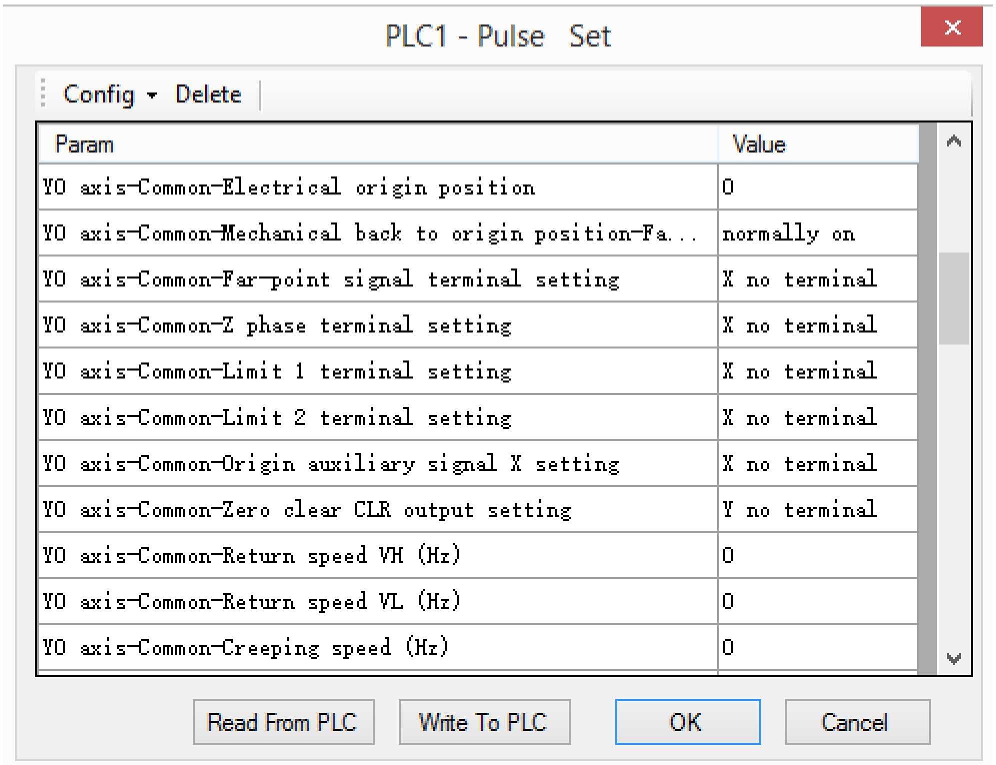
Program

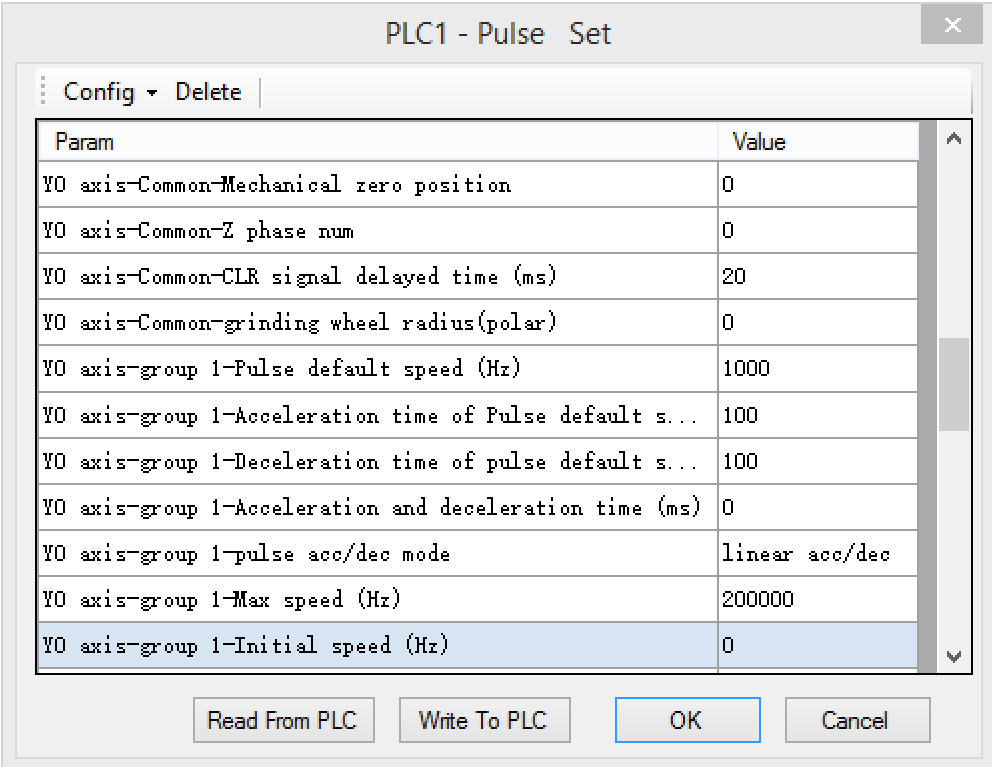


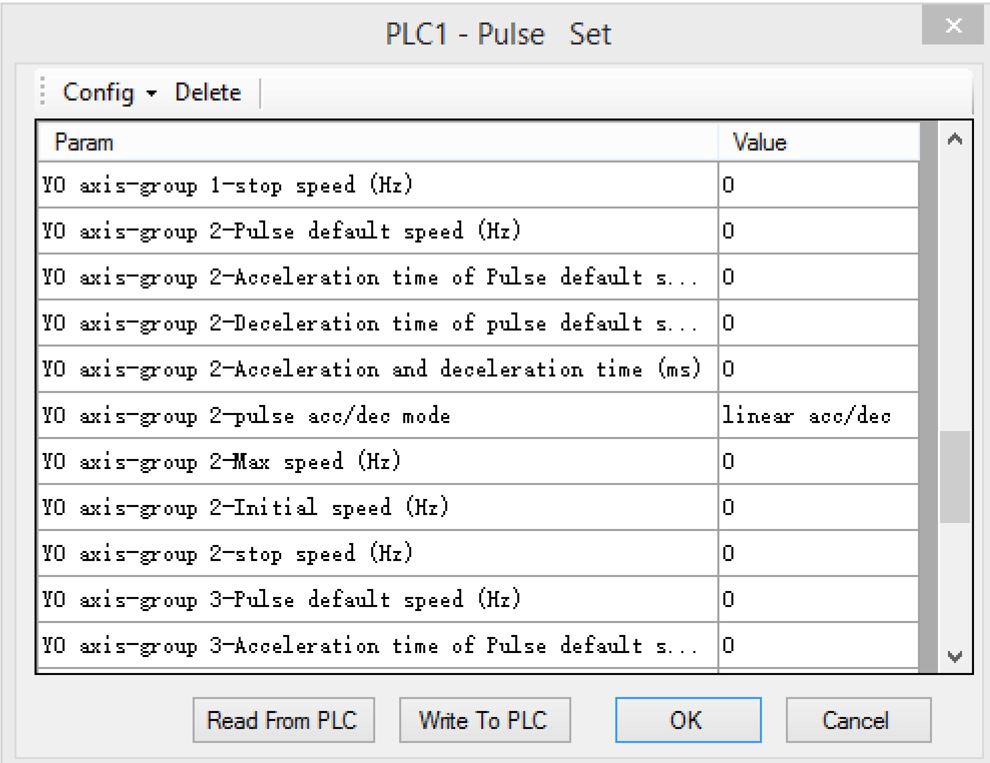
Software setting



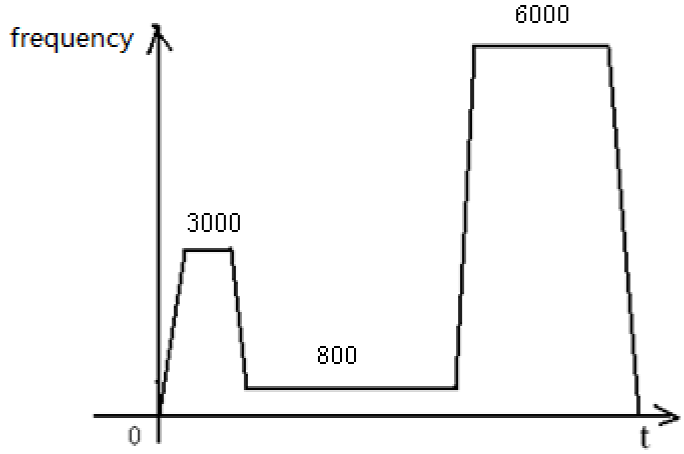

PLC1 - Pulse Set
Config
Delete
Param
Value
Y0 axis-Common-Parameters setting-Pulse output logic
positive logic
Y0 axis-Common-Parameters setting-Pulse direction logic
positive logic
Y0 axis-Common-Parameters setting-Pulse unit
pulse number
Y0 axis-Common-Parameters setting-Interpolation coor...
Cross coordi...
Y0 axis-Common-pulse send mode
complete
Y0 axis-Common-Pulse num (1)
1
Y0 axis-Common-Offset (1)
1
Y0 axis-Common-Pulse direction terminal
Y2
Y0 axis-Common-Delayed time of pulse direction (ms)
10
Y0 axis-Common-Gear clearance positive compensation
0
Y0 axis-Common-Gear clearance negative compensation
0
Read From PLC
Write To PLC
OK
Cancel


PLC1 - Pulse Set
Config
Delete
Param
Value
Y0 axis-Common-Electrical origin position
0
Y0 axis-Common-Mechanical back to origin position-Fa...
normally on
Y0 axis-Common-Far-point signal terminal setting
X no terminal
Y0 axis-Common-Z phase terminal setting
X no terminal
Y0 axis-Common-Limit 1 terminal setting
X no terminal
Y0 axis-Common-Limit 2 terminal setting
X no terminal
Y0 axis-Common-Origin auxiliary signal X setting
X no terminal
Y0 axis-Common-Zero clear CLR output setting
Y no terminal
Y0 axis-Common-Return speed VH (Hz)
0
Y0 axis-Common-Return speed VL (Hz)
0
Y0 axis-Common-Creeping speed (Hz)
0
Read From PLC
Write To PLC
OK
Cancel


PLC1 - Pulse Set
Config Delete
Param
Value
Y0 axis-Common-Mechanical zero position
0
Y0 axis-Common-Z phase num
0
Y0 axis-Common-CLR signal delayed time (ms)
20
Y0 axis-Common-grinding wheel radius(polar)
0
Y0 axis-group 1-Pulse default speed (Hz)
1000
Y0 axis-group 1-Acceleration time of Pulse default s...
100
Y0 axis-group 1-Deceleration time of pulse default s...
100
Y0 axis-group 1-Acceleration and deceleration time (ms)
0
Y0 axis-group 1-pulse acc/dec mode
linear acc/dec
Y0 axis-group 1-Max speed (Hz)
200000
Y0 axis-group 1-Initial speed (Hz)
0
Read From PLC
Write To PLC
OK
Cancel


PLC1 - Pulse Set
Config Delete
Param
Value
Y0 axis-group 1-stop speed (Hz)
0
Y0 axis-group 2-Pulse default speed (Hz)
0
Y0 axis-group 2-Acceleration time of Pulse default s...
0
Y0 axis-group 2-Deceleration time of pulse default s...
0
Y0 axis-group 2-Acceleration and deceleration time (ms)
0
Y0 axis-group 2-pulse acc/dec mode
linear acc/dec
Y0 axis-group 2-Max speed (Hz)
0
Y0 axis-group 2-Initial speed (Hz)
0
Y0 axis-group 2-stop speed (Hz)
0
Y0 axis-group 3-Pulse default speed (Hz)
0
Y0 axis-group 3-Acceleration time of Pulse default s...
0
Read From PLC
Write To PLC
OK
Cancel

Pulse wave:



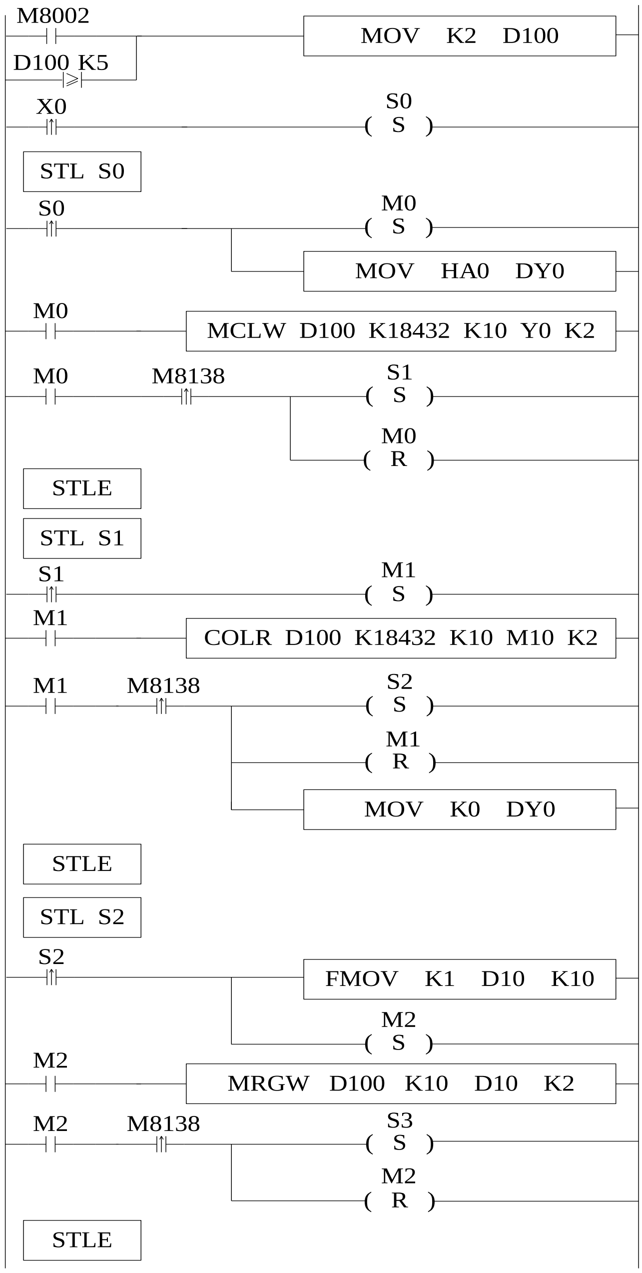
12-2. MODBUS Communication Application

The following program is write and read of Modbus communication between 1 master station and 3 slave stations.

Program operation

（1）Write Y0~~Y11 status of host station to 2# slave station Y0~~Y11;
（2）Read 2#slave station Y0~~Y10 to host station M10~~M19;
（3）Write D10~~D19 to 2#slave station D10~~D19；
（4）Read 2#slave station D10~~D19 to host station D20~~D29；
（5）So are the 3#、4# slave station

Here is the program contrast of XC and XD series modbus RTU communication.
XC series communication program:



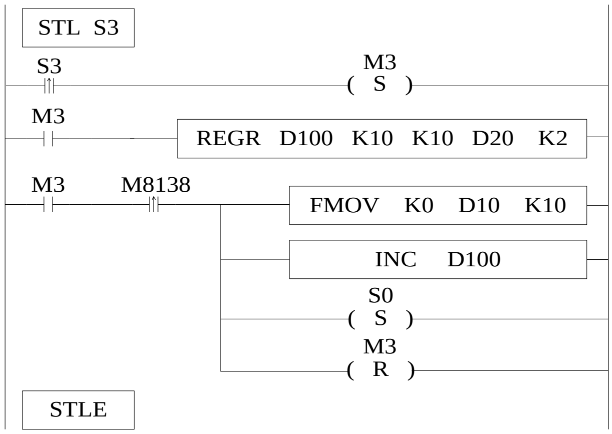
Send station#2 to D100

Start Flow S0

Set Y0~Y11 of host station ON; write the host state to 2#、3#、4# slave station Y0~Y11 in turn; and enter Flow S1 after communicating successfully.

M10~M19 in host station read 2#、3#、4#slave station Y0~~Y11 state in turn; Reset Y0~Y11 of host station and enter Flow S2.

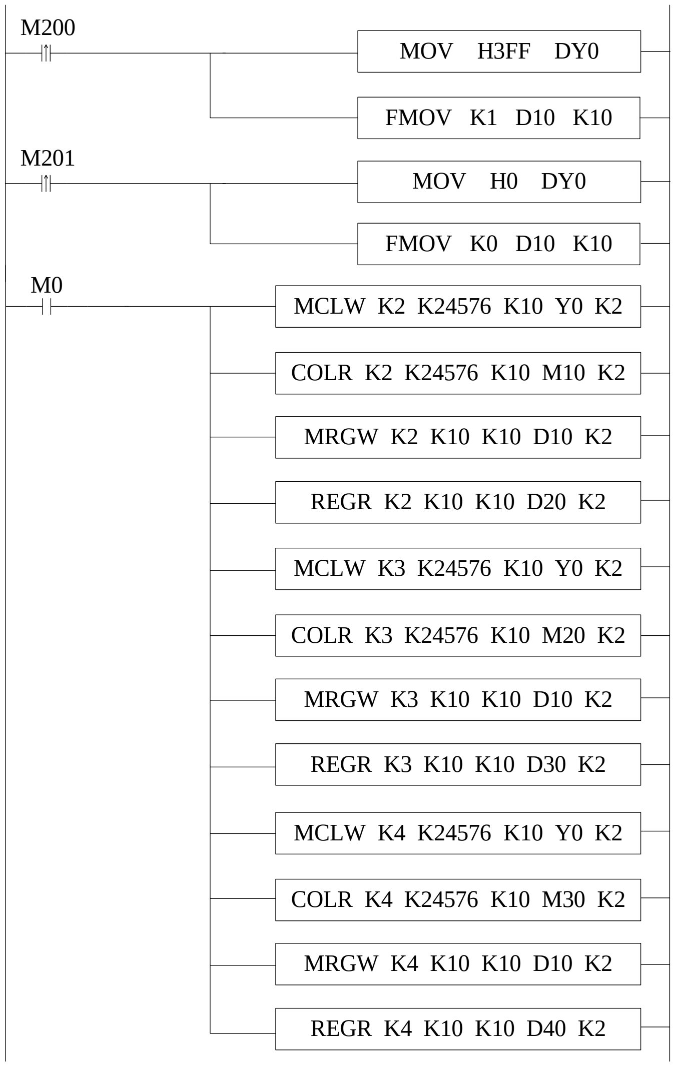
Write 1 to D10~D19 in host station; Write 2#、3#、4#slave station D10~D19 to host station register; Enter Flow S3 after communicating successfully.



Host station register D20~D29 read D10~D19 in 2#、3#、4#slave station; Reset host station D10~D19 after communicating successfully. Then station number increase 1, execute flow S0, and repeat above steps.

Modbus RTU instruction can be written in program directly, and the protocol station will queue Modbus communication request. Communication is another task which means users can write multiply Modbus RTU communication instructions together in the main program, and the instructions can be activated by one trigger condition. Then PLC will handle with this Modbus RTU communication instructions in turn, while XC series PLC errors if multiply communication instructions execute at the same time.

XD series program:



Set Y0~Y11 of host station and write 1 to D10~D19 if M200 is rising edge; RST Y0~Y11 of host station and write 0 to D10~D19 if M201 is rising edge.

write Y0~Y11 of host station to Y0~Y11 of 2#slave station; read Y0~Y11 of 2#slave station to M10~M19; write D10~D19 to D10~D19 of 2#slave station; read D20~D29 of 2#slave station to D20~D29 of host station, and so on.

Command language:

LDP M200 //set M200
MOV H3FF DY0 //send Hexadecimal 3FF to Y0~Y15
FMOV K1 D10 K10 //send decimal 1 to register D10~D19
LDP M201 //set M201
MOV H0 DY0 //send decimal 0 to Y0~Y15
FMOV K0 D10 K10 //send decimal 0 to register D10~D19
LD M0 //set M0
MCLW K2 K24576 K10 Y0 K2 //write Y0~Y11 status of host station to 2#slave station Y0~Y11

COLR K2 K24576 K10 M10 K2 //read status of 2#slave station Y0~Y11 to host station M10~M19
MRGW K2 K10 K10 D10 K2 //write host station D10~D19 to 2#slave station D10~D19
REGR K2 K10 K10 D20 K2 //read 2#slave station D10~D19 to host station D20~D29
MCLW K3 K24576 K10 Y0 K2 //write status of host station Y0~Y11 to 3#slave stationY0~Y11
COLR K3 K24576 K10 M20 K2 //read status of 3#slave station Y0~Y11 to host station
M20~M29
MRGW K3 K10 K10 D10 K2 //write 3 to D10~D19 of host station
REGR K3 K10 K10 D30 K2 //read D10~D19 status of 3#slave station to D30~D39 of host station
MCLW K4 K24576 K10 Y0 K2 //write the status of host station Y0~Y11 to Y0~Y11 of 4#slave station
COLR K4 K24576 K10 M30 K2 //read Y0~Y11 status of 4#slave station to M30~M39 of host station
MRGW K4 K10 K10 D10 K2 //write D10~D19 of host station to D10~D19 of 4#slave station
REGR K4 K10 K10 D40 K2 //read D10~D19 of 4#slave station to D40~D49 of host station

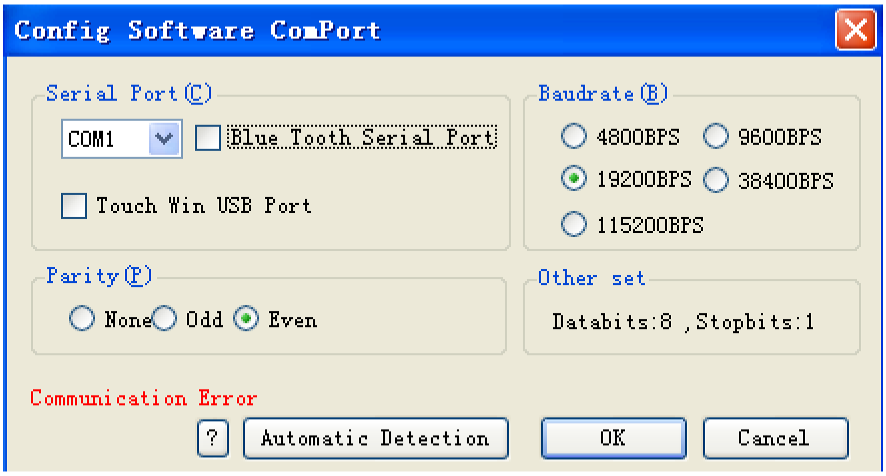
13 Common Questions and Answers

This chapter mainly introduces XD3 series PLC common questions and answers.

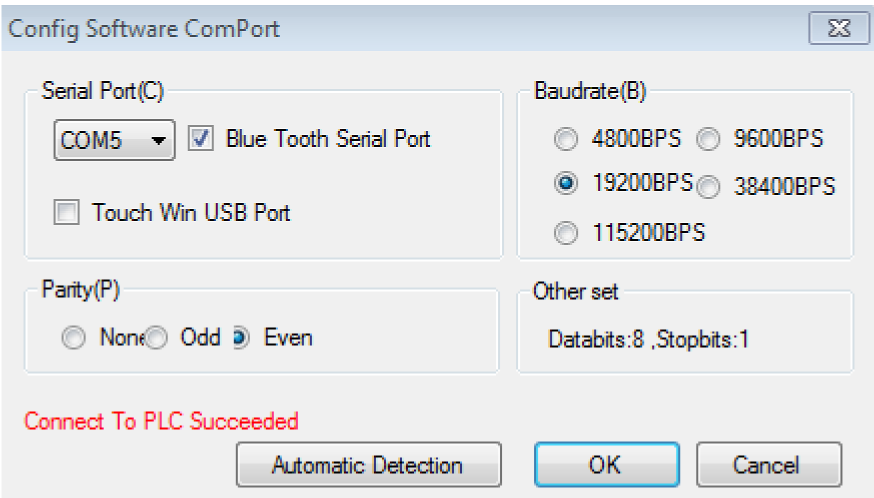
Q1: How to connect PLC with PC?

A1:

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 PLC successfully’ 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)!

Q2： Current PC is offline, unable connect with PLC?

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 DVP download communication cable brand is not Xinje.

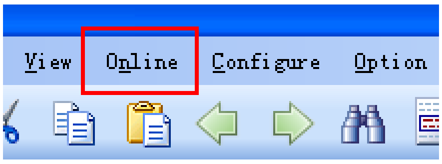
Solutions:

At first, change the DVP or XVP cable used to connect PC and PLC to Xinje special cable if it is not;

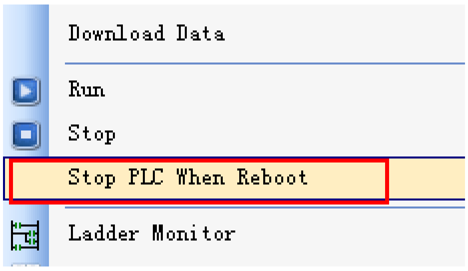
After confirming the connection cable is the Xinje special DVP cable and USB converter has been used, you can use it to try to connect desktop PC with 9-needle serial port to PLC. If the desktop PC can be connected correctly, please change the USB converter cable with higher performance or install the 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;

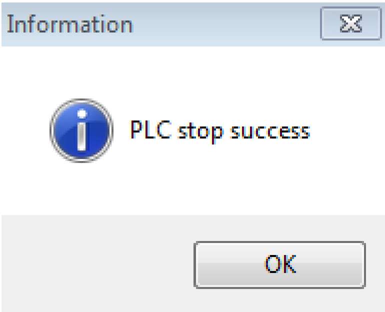


Click ‘Stop when PLC reboot’ from the drop-down menu;

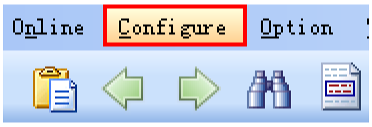


Following window will jump out;

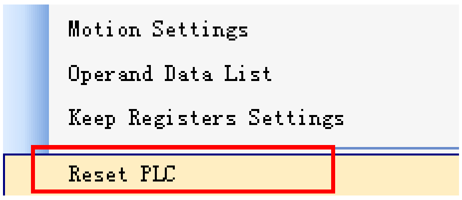
By this time, cut off PLC power for 2-3s and power on again, then a ‘PLC has been stopped successfully’ window will normally jump out; if the window do not jump out after power on, try again a few times until the information window of successful stop jump out.



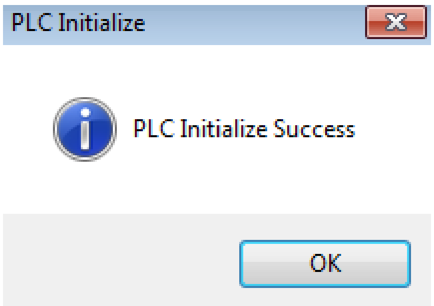
Then click ‘configure’ button ;



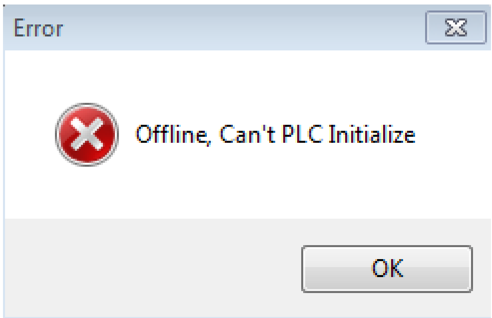
Click ‘Reset PLC’ in the drop-down menu;



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 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！).

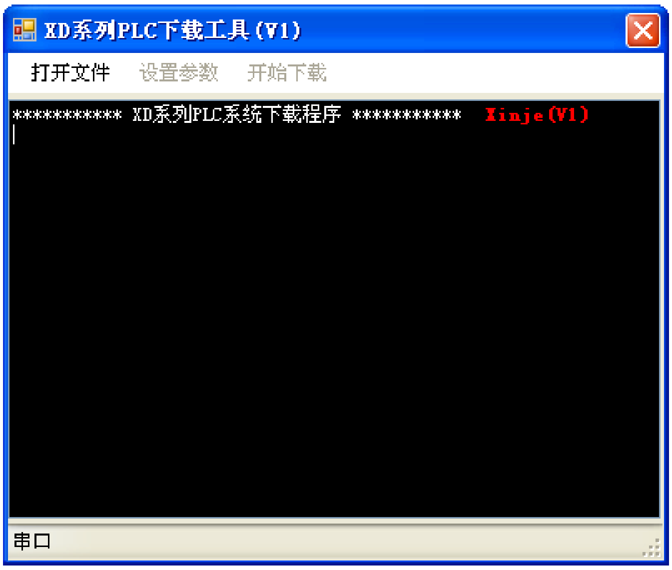
How to update XD series PLC？

PLC update tool:

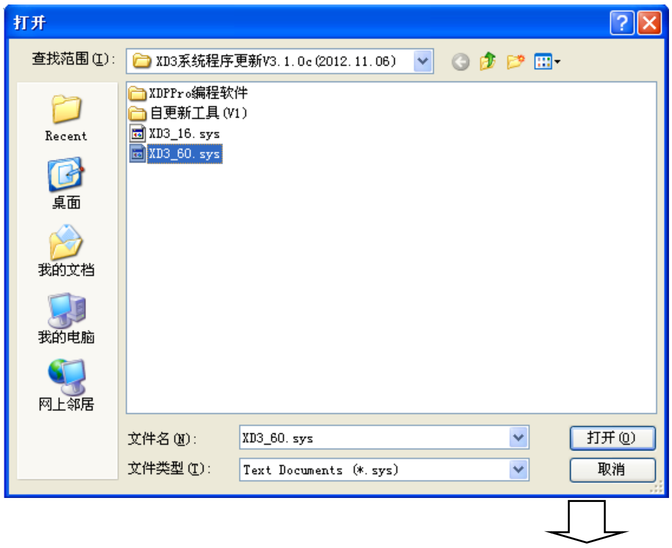
‘XD series PLC download program tool’ and ‘system file’ (*.sys file)

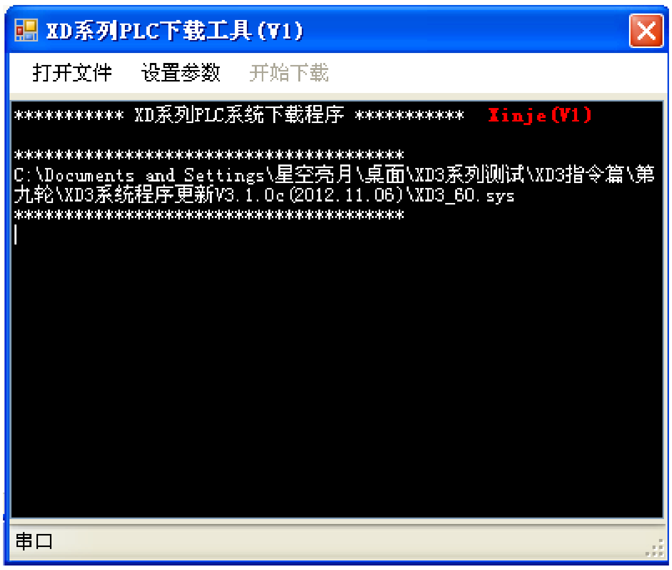
Close all the programs which may occupy the serial port

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):**



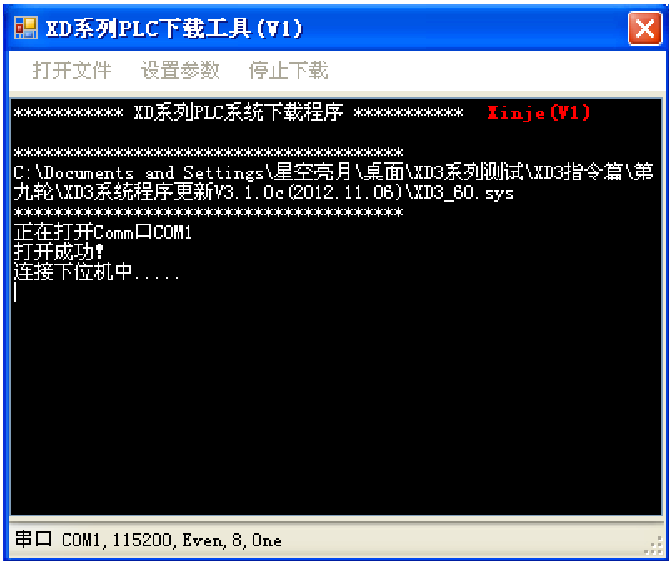


Set the parameters:

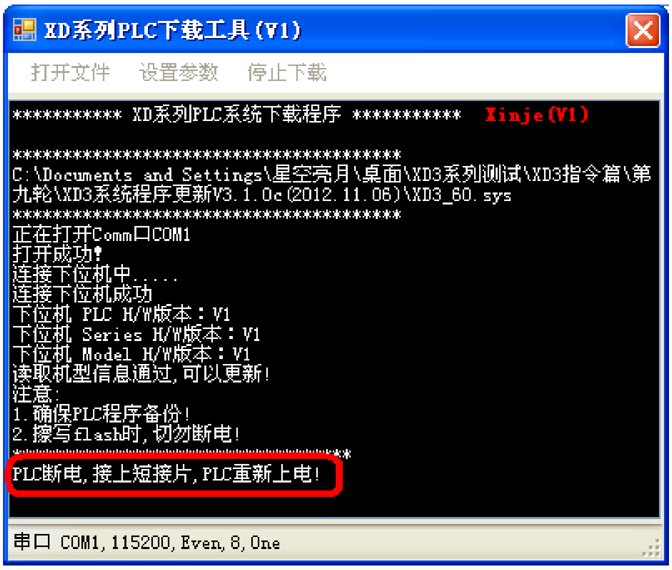
Click “set parameter”, it will show the parameter window:

Note: set the com port, the baud rate is default setting, no need to change.

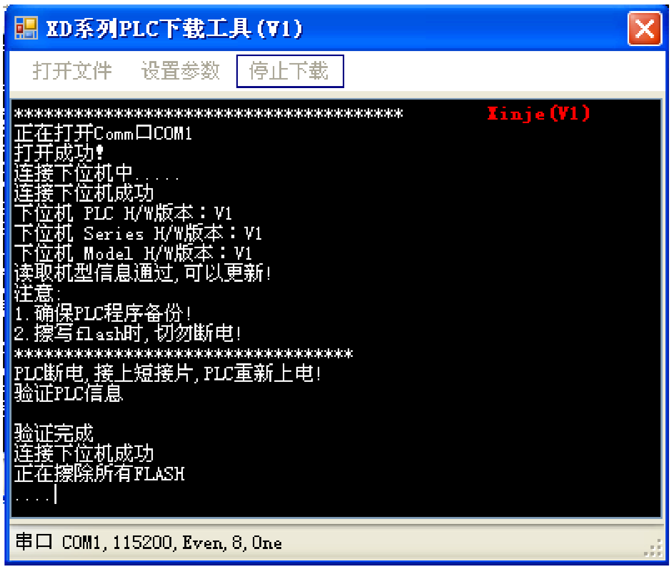
Click “download”, the window will show below words:



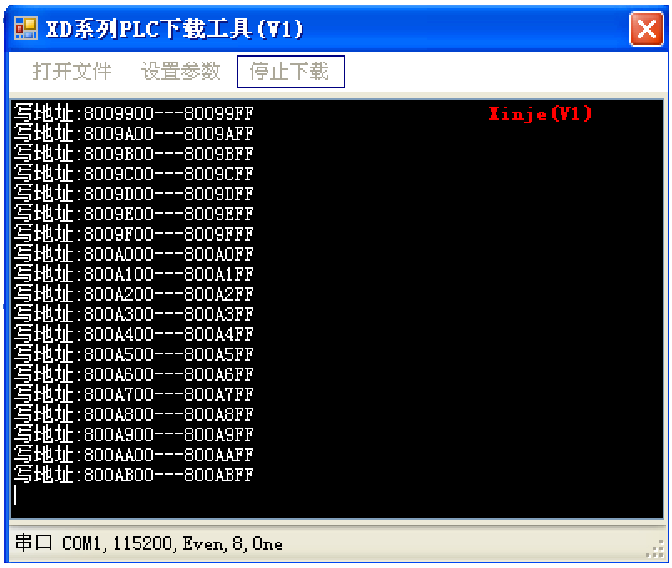
Power on the PLC, the update tool will show below words:



Cut off the power of PLC, connect the short jumper, then power on the PLC again.



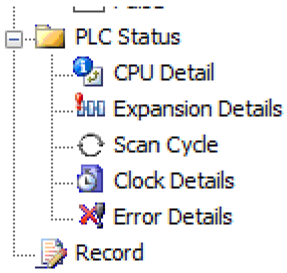
PLC start to update, the updating will take few minutes.

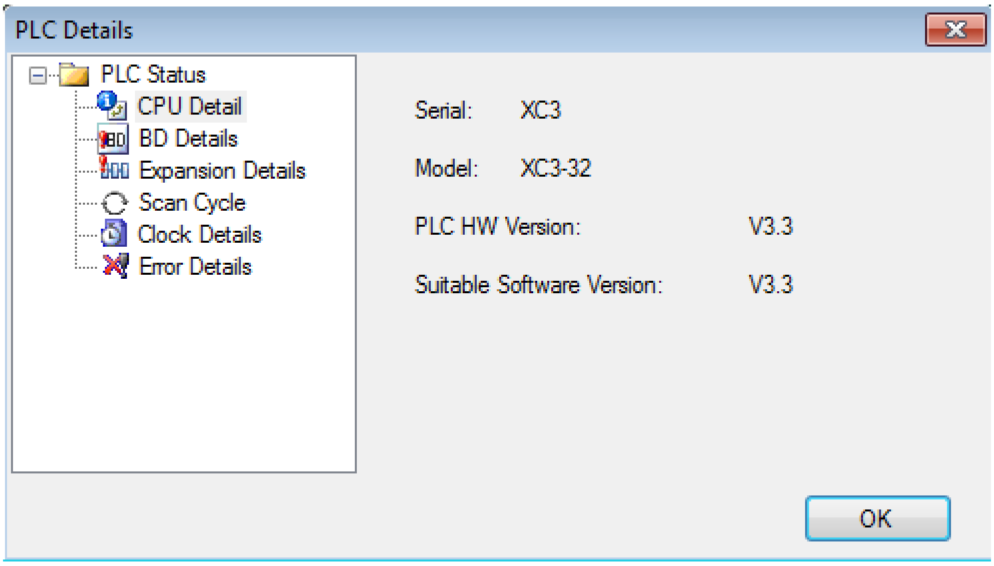


After finishing the update, cut off the PLC power, take off the short jumper, then power on the PLC again.

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, XDM 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

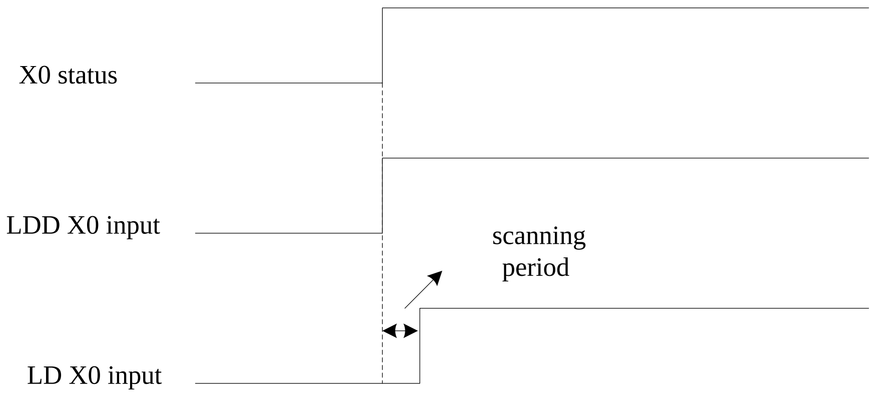
The other mode is bit operation of fixed register. E.g: D0.0 is the first bit of 16 bits in register D0. Similarly, D0.1 is the second bit and so on, as shown below:

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.

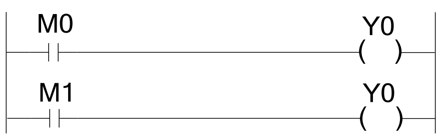
Q5: What's the use of execution instruction LDD/OUTD etc?
A5:
When PLC executes program, state of input point state will map to image register. From then on, PLC will refresh input state at the beginning of every scan cycle; if we use LDD instruction, then the state of input point will not need map to 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.

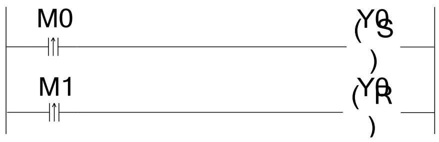
Q7: Why the M and Y cannot output sometime?
A7:
Output mainly has two ways: 1. OUT instruction; 2. SET instruction. The coil will keep outputting if there is no RST instruction.
Usually in the program, one coil M or Y should use the same output way. Otherwise, the coil cannot output.
For example:



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.

Q9: Communicate with SCADA software
A9:
If there is no choice for XD series PLC in SCADA software, please choose Modbus-RTU protocol and communicate through RS485 port. Please refer to XD series PLC instruction manual chapter 7.

Q10: MODBUS Communication
A10:
Make sure the RS485 connection is correct. (Terminal A and B on the PLC). Please modify the port 2 parameters through SFD610 to SFD614.
Method 1: set the parameters through SFD register
Connect PLC and XDPpro software, set the SFD610 to SFD614 through free monitor function. Then restart the PLC again.
Please set the parameters according to different device. Make sure the modbus address and function code. Some device will show setting frequency after sending running signal.
Set the parameters through control panel
Please refer to XD series PLC instruction manual chapter 7.

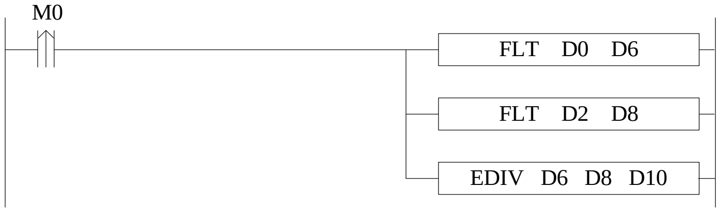
Q11: The LED light of XD series PLC (PWR/RUN/ERR)
A11:

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.

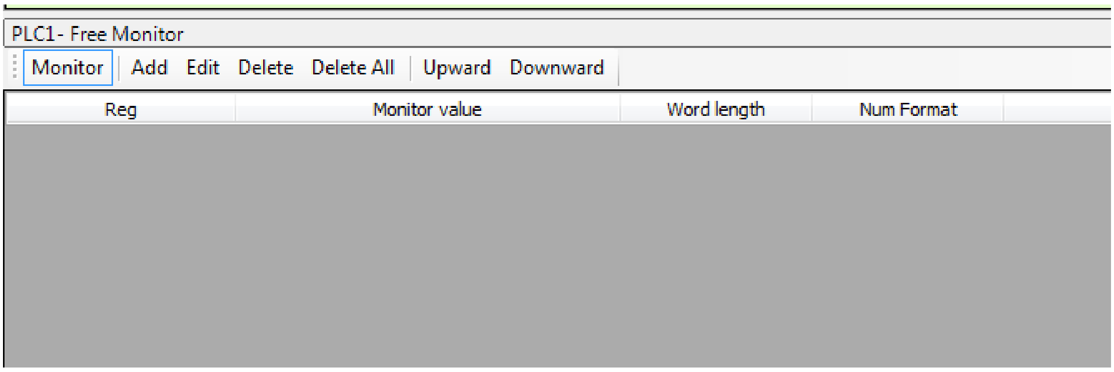


Q13: Why the floating numbers become messy code in online ladder monitor window?

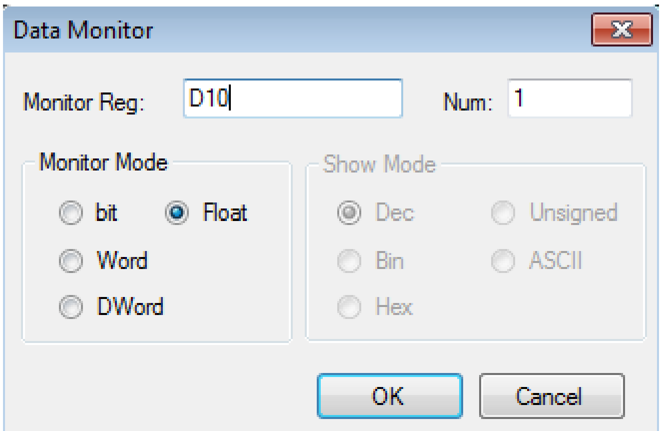
A13:

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 bit*32 bit=64 bit 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 bit (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 pulse do not output when we get through the conduction condition？
A17:
First, make sure your PLC has pulse control function and output is transistor type.
Second, check if pulse instruction（absolute or relative instruction）and parameters are OK.
Third, check if there is double coils output of one pulse output terminal in program. Please refer to Section 6-4 ‘notice’ in Chapter 6.

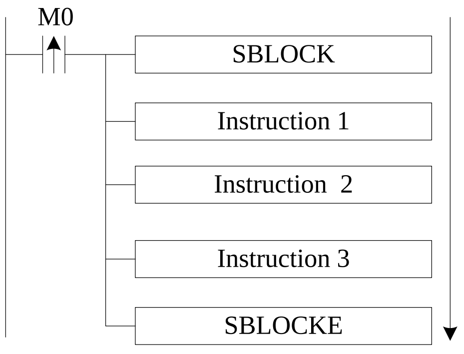
Q18: Why the corresponding temporary register still not count when the PLC input terminal of HSC has been connected correctly?
A18:
To realize HSC function, we not only need connect the high speed pulse to HSC terminal, but also need write corresponding HSC program according to function instructions; For more details please refer to Chapter 5 ‘high speed counter’ in this manual.

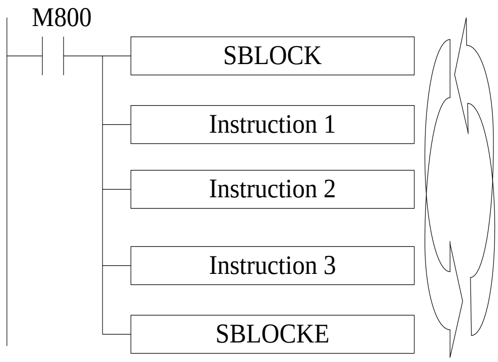
Q19: What’s PLC output terminal A, B？
A19:
PLC output terminal A、B are RS485 terminals of PORT2 on PLC. It’s the same communication port with the round port of PORT2 which is RS232 port.
Note：PORT2 communication port RS232 and RS485 can not be used simultaneously.

Q20: What’s the advantage of C language compared with ladder diagram？
A20:
(1) XD series PLC supports all C language function;
(2) Under any download mode, C language function parts can not be uploaded;
(3) C language function block can be called between different files.

Q21: What’s the difference of sequence function BLOCK trigger condition: rising edge triggered and normally closed conduction?
A21:

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

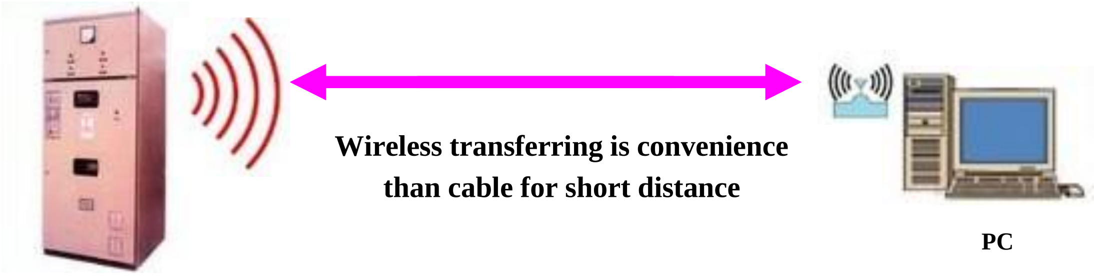
from up to down, cyclic run the instruction

Q22: what's the advantage that XD series PLC replaces DVP download cable with Bluetooth?

A22:

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.



Control cabinet installed XD series PLCand COM-Bluetooth

Q23: XD series PLC program several download modes and each mode's feature？

A23:

XD series PLC have three program download modes:

General download mode： Under this mode, users can download program from computer to PLC or upload program from PLC to computer very conveniently, so we usually use this mode when we debug devices.

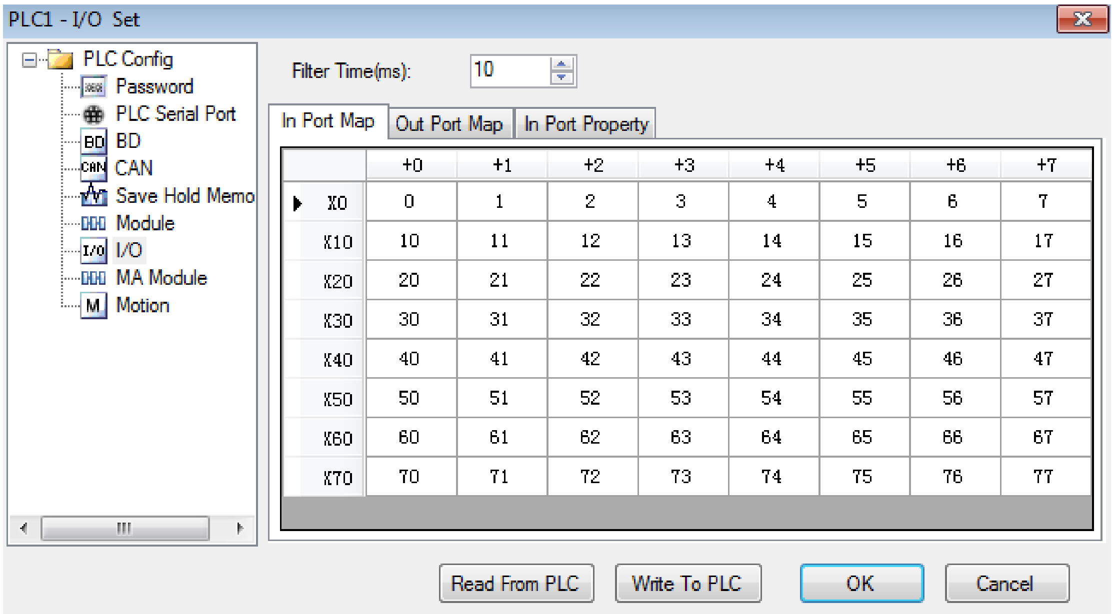
Password download mode： If you set a password for PLC, you need input correct password when you upload program in PLC to computer. In the password advanced options, you still can check 'decrypt before program download' function（Note: this operation is dangerous. If you forget password, your PLC will be locked!）. If you want to protect your program and still want to upload the program to use by yourself, you can use this mode.

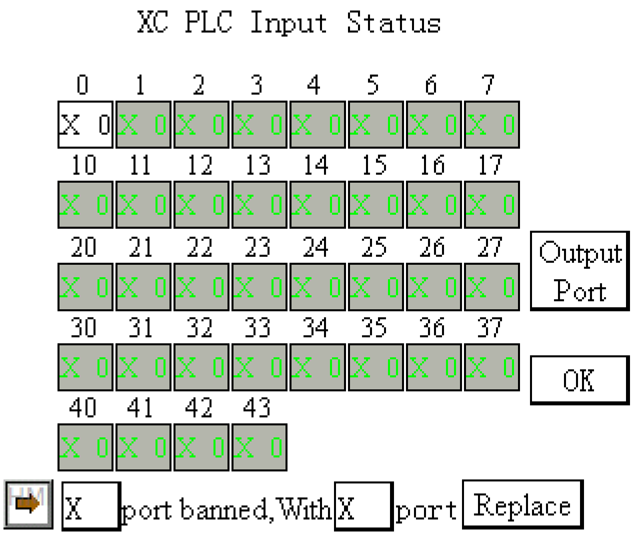
Confidential download mode: Under this mode, once users download the program to PLC, the program will not be uploaded anyway. This mode can save more PLC internal resources, increase PLC capacity and improve download speed.

Q24: PLC I/O terminal exchanging

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.





Touchwin HMI I/O terminal exchanging screen

Q25: What's the function of XD 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 series PLC connect to the network?

A26:

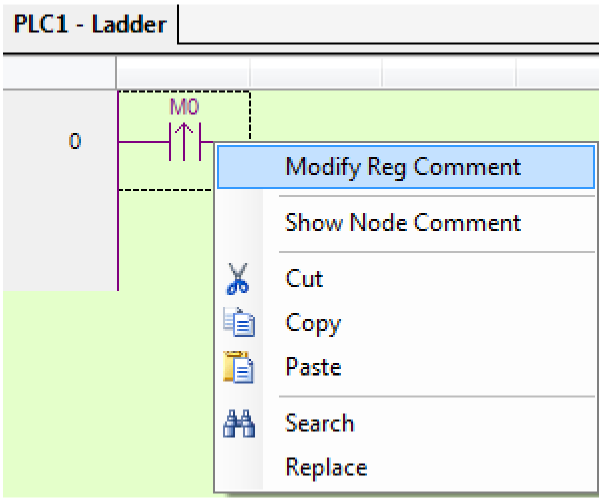
XD series PLC can connect to network by Xinje T-BOX, G-BOX, Z-BOX expansion modules or expansion BD boards which have their own communication characteristics. Details please refer to the user manual of communication module or BD board.

Q27: how to add soft element and line note in XDppro software?

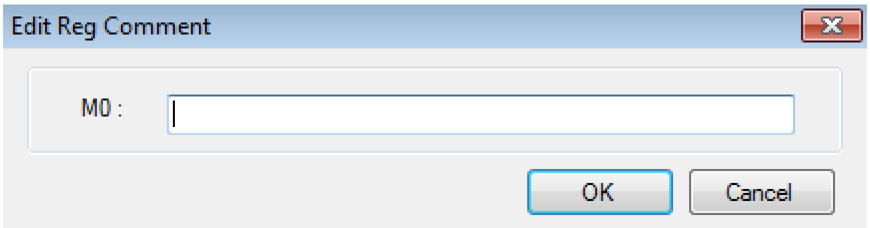
A27:

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:

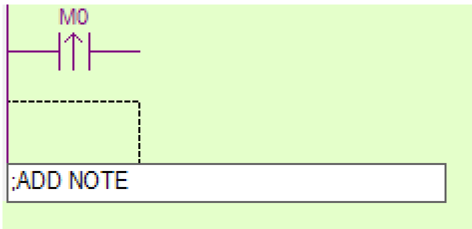


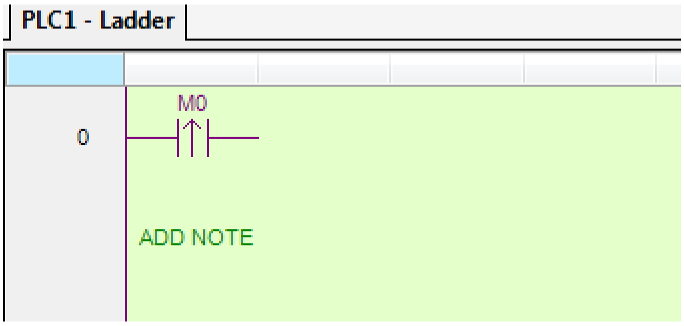
Click "Modify reg comment" to add element notes in below window:



Line note

Line note starts from ";". Double click the line, then input semicolon and the contents.





Q28: do not have clock function?

A28:

XD 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.

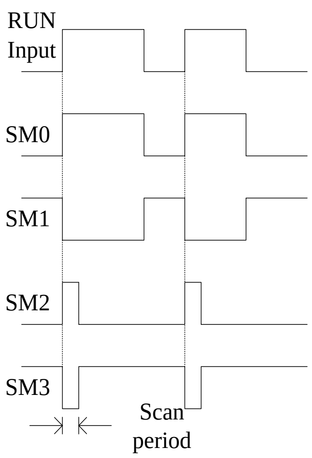
If the PLC has clock function, then please check whether the value in register SD13-SD19 is decimal, if not, transform it to decimal by instruction BIN or TRD.

Appendix Special soft components

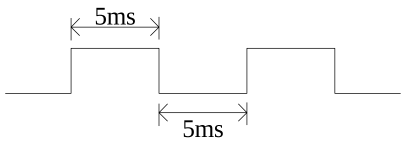
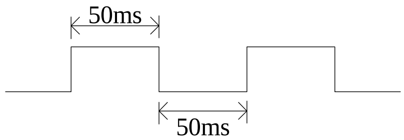
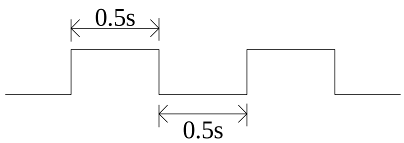
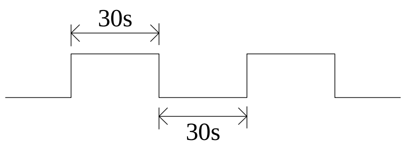
Appendix mainly introduces the functions of XD3 series PLC special soft element, data register, FlashROM and the address distribution of expansions for users to search.

Appendix 1． Special Auxiliary Relay

Initial Status（SM0-SM3）

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

Clock（SM11-SM14）

ID	Function	Description
SM011	10ms frequency cycle	
SM012	100ms frequency cycle	
SM013	1s frequency cycle	
SM014	1min frequency cycle	

Mark（SM20-SM29）

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（SM32-SM34）

ID	Function	Description
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		

Interruption（SM50-SM80）

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 Pulse（SM140-SM199）

ID	Function	Description	
SM1000	‘Sending pulse’ flag	SM1000 will be ON when sending the pulse	PULSE_1
SM1001	Direction flag	SM1001 value being 1 stands for positive direction and corresponding port is ON	
SM1002	Overflow flag of accumulated pulse number	SM1002 value will be 1 when accumulated pulse number overflows.	
SM1003	Overflow flag of pulse equivalent	SM1003 value will be 1 when pulse equivalent overflows	
SM1004			
SM1005			
SM1006			
SM1007			
SM1008			
SM1009			
SM1010	Pulse error flag	SM1010 will be ON when pulse errors	
SM1020	‘Sending pulse’ flag	SM1020 will be ON when sending the pulse	PULSE_2
SM1021	Direction flag	SM1021 value being 1 stands for positive direction and corresponding port is ON	
SM1022	Overflow flag of accumulated pulse number	SM1022 value will be 1 when accumulated pulse number overflows.	
SM1023	Overflow flag of pulse equivalent	SM1023 value will be 1 when pulse equivalent overflows	
SM1024			
SM1025			
SM1026			
SM1027			
SM1028			
SM1029			
SM1030	Pulse error flag	SM1030 will be ON when pulse errors	
SM1040	‘Sending pulse’ flag	SM1040 will be ON when sending the pulse	PULSE_3
SM1041	Direction flag	SM1041 value being 1 stands for positive direction and corresponding port is ON	
SM1042	Overflow flag of accumulated pulse number	SM1042 value will be 1 when accumulated pulse number overflows.	
SM1043	Overflow flag of pulse equivalent	SM1043 value will be 1 when pulse equivalent overflows	
SM1044			
SM1045			
SM1046			
SM1047			
SM1048			
SM1049			
SM1050	Pulse error flag	SM1050 will be ON when pulse errors	

SM1060	'Sending pulse' flag	SM1060 will be ON when sending the pulse	PULSE_4
SM1061	Direction flag	SM1061 value being 1 stands for positive direction and corresponding port is ON	
SM1062	Overflow flag of accumulated pulse number	SM1062 value will be 1 when accumulated pulse number overflows.	
SM1063	Overflow flag of pulse equivalent	SM1063 value will be 1 when pulse equivalent overflows	
SM1064			
SM1065			
SM1066			
SM1067			
SM1068			
SM1069			
SM1070	Pulse error flag	SM1070 will be ON when pulse errors	
SM1080	'Sending pulse' flag	SM1080 will be ON when sending the pulse	PULSE_5
SM1081	Direction flag	SM1081 value being 1 stands for positive direction and corresponding port is ON	
SM1082	Overflow flag of accumulated pulse number	SM1082 value will be 1 when accumulated pulse number overflows.	
SM1083	Overflow flag of pulse equivalent	SM1083 value will be 1 when pulse equivalent overflows	
SM1084			
SM1085			
SM1086			
SM1087			
SM1088			
SM1089			
SM1090	Pulse error flag	SM1090 will be ON when pulse errors	
SM1100	'Sending pulse' flag	SM1100 will be ON when sending the pulse	PULSE_6
SM1101	Direction flag	SM1101 value being 1 stands for positive direction and corresponding port is ON	
SM1102	Overflow flag of accumulated pulse number	SM1102 value will be 1 when accumulated pulse number overflows.	
SM1103	Overflow flag of pulse equivalent	SM1103 value will be 1 when pulse equivalent overflows	
SM1104			
SM1105			
SM1106			
SM1107			
SM1108			
SM1109			
M1110	Pulse error flag	SM1110 will be ON when pulse errors	
SM1120	'Sending pulse' flag	SM1120 will be ON when sending the pulse	PULSE_7
SM1121	Direction flag	SM1121 value being 1 stands for positive direction and corresponding port is ON	

SM1122	Overflow flag of accumulated pulse number	SM1122 value will be 1 when accumulated pulse number overflows.	
SM1123	Overflow flag of pulse equivalent	SM1123 value will be 1 when pulse equivalent overflows	
SM1124			
SM1125			
SM1126			
SM1127			
SM1128			
SM1129			
SM1130	Pulse error flag	SM1130 will be ON when pulse errors	
SM1140	'Sending pulse' flag	SM1140 will be ON when sending the pulse	PULSE_8
SM1141	Direction flag	SM1141 value being 1 stands for positive direction and corresponding port is ON	
SM1142	Overflow flag of accumulated pulse number	SM1142 value will be 1 when accumulated pulse number overflows.	
SM1143	Overflow flag of pulse equivalent	SM1143 value will be 1 when pulse equivalent overflows	
SM1144			
SM1145			
SM1146			
SM1147			
SM1148			
SM1149			
SM1150	Pulse error flag	SM1150 will be ON when pulse errors	
SM1160	'Sending pulse' flag	SM1160 will be ON when sending the pulse	PULSE_9
SM1161	Direction flag	SM1161 value being 1 stands for positive direction and corresponding port is ON	
SM1162	Overflow flag of accumulated pulse number	SM1162 value will be 1 when accumulated pulse number overflows.	
SM1163	Overflow flag of pulse equivalent	SM1163 value will be 1 when pulse equivalent overflows	
SM1164			
SM1165			
SM1166			
SM1167			
SM1168			
SM1169			
SM1170	Pulse error flag	SM1170 will be ON when pulse errors	
SM1180	'Sending pulse' flag	SM1180 will be ON when sending the pulse	PULSE-_10
SM1181	Direction flag	SM1181 value being 1 stands for positive direction and corresponding port is ON	
SM1182	Overflow flag of accumulated pulse number	SM1182 value will be 1 when accumulated pulse number overflows.	
SM1183	Overflow flag of pulse equivalent	SM1183 value will be 1 when pulse equivalent overflows	

SM1184			
SM1185			
SM1186			
SM1187			
SM1188			
SM1189			
SM1190	Pulse error flag	SM1190 will be ON when pulse errors	

Sequence Function BLOCK（SM240-SM339）

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-SM413）

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		

Error Message（SM450-SM452）

ID	Function	Description
SM450	System error check	
SM451		
SM452		

Expansion Modules, BD Status（SM500）

ID	Function	Description
SM500	Module status read is finished	

Communication（SM130-SM1319）

	ID	Function	Description
COM1	SM130	Accurate receipt flag	
	SM131	Error receipt flag	
	SM132		
	SM133		
	SM134		
	SM135		
	SM136		
	SM137		
	SM138		
	SM139		
COM2	SM140	Accurate receipt flag	
	SM141	Error receipt flag	
	SM142		
	SM143		
	SM144		
	SM145		
	SM146		
	SM147		
	SM148		
	SM149		

Appendix 2. Special Data Register

Clock（SD010-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)	0~31 (BCD code)
SD017	Month (clock)	0~12 (BCD code)
SD018	Year (clock)	2000~2099 (BCD code)
SD019	Week (clock)	0 (Sunday) ~6 (Saturday) (BCD code)

Flag （SD020-SD031）

ID	Function	Description
SD020	Information of type	
SD021	Information of type	
:		
SD030	Information of type	
SD031	Information of type	

Step ladder（SD040）

ID	Function	Description
SD40	Flag of the executing process S	

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

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）

SD401	Number of communication error expansion module	
SD402	Number of communication error BD	
……		
SD405		
SD406		
SD407		
SD408		
SD409	Operation error code number	1：Divided by zero error 2：Former operand’s address less that the latter one’s of MRST,MSET 3：ENCO,DECO encoding, decoding instruction data bit overruns. 4：BDC code error 7：Square root error
SD410	Numbers of shift register D when migration overruns	
SD411		
SD412		
SD413		

High Speed Pulse（SD1000-SD1099）

ID	**Function**	**Description**	
SD1000	Current segment（No. n segment）		
SD1001			
SD1002	Low 16 bits of accumulated pulse number (the unit is the pulse number)		
SD1003	High 16 bits of accumulated pulse number		
SD1004	The low 16 bits of accumulated pulse number		
SD1005	High 16 bits of accumulated pulse number		
SD1006	Low 16 bits of current output frequency		
SD1007	high 16 bits of current output frequency		
SD1008	Low 16 bits of current output frequency(The unit is pulse equivalent)		
SD1009	High 16 bits of current output frequency		PULSE_1
SD1010	Wrong Pulse message	1：Pulse data block error 2：Equivalent mode：pulse amount/turn, amount/ turn of movement is 0 3:Code of system parameters block error 4:Pulse data block exceeds max limit 10:Zero return do not set near point signal 11：Speed of zero return is 0 12：Crawling speed of zero return is 0 13：Directions of zero return speed and zero auxiliary speed differ	
SD1011	Pulse data block error		
SD1020	Current segment(No. n segment)		
SD1021			
SD1022	Low 16 bits of accumulated pulse number (the unit is pulse number)		
SD1023	High 16 bits of accumulated pulse number		PULSE_2
SD1024	Low 16 bits of accumulated pulse number		
SD1025	High 16 bits of accumulated pulse number		
SD1026	Low 16 bits of current output frequency(the unit is pulse number)		

SD1027	High 16 bits of current output frequency(the unit is pulse number)		
SD1028	Low 16 bits of current output frequency(the unit is pulse equivalent)		
SD1029	High 16 bits of current output frequency(the unit is pulse equivalent)		
SD1030	Wrong Pulse message	1： Pulse data block error 2： Equivalent mode： pulse amount/turn, amount/ turn of movement is 0 3:Code of system parameters block error 4: Pulse data block exceeds max limit 10: Zero return do not set near point signal 11： Speed of zero return is 0 12： Crawling speed of zero return is 0 13 Direction of zero return speed and zero auxiliary speed	
SD1031	Code of error pulse block		
SD1040	Current segment(No. n segment)		PULSE_3
SD1041			
SD1042	Low 16 bits of accumulated pulse number (the unit is pulse number)		
SD1043	High 16 bits of accumulated pulse number (the unit is pulse number)		
SD1044	Low 16 bits of accumulated pulse number(the unit is pulse equivalent)		
SD1045	High 16 bits of accumulated pulse number(the unit is pulse equivalent)		
SD1046	Low 16 bits of current output frequency(the unit is pulse number)		
SD1047	High 16 bits of current output frequency(the unit is pulse number)		
SD1048	Low 16 bits of current output frequency(the unit is pulse equivalent)		
SD1049	High 16 bits of current output frequency(the unit is pulse equivalent)		
SD1050	Wrong Pulse message	1： Pulse data block error 2： Equivalent mode： pulse amount/turn、 amount/ turn of movement is 0 3:Code of system parameters block error	

		4：Pulse data block exceeds max limit 10：Zero return do not set near point signal 11：Speed of zero return is 0 12：Crawling speed of zero return is 0 13 Direction of zero return speed and zero auxiliary speed	
SD1051	Code of error pulse block		
SD1060	Current segment(No. n segment)		PULSE_4
SD1061			
SD1062	Low 16 bits of accumulated pulse number (the unit is pulse number)		
SD1063	High 16 bits of accumulated pulse number (the unit is pulse number)		
SD1064	Low 16 bits of accumulated pulse number(the unit is pulse equivalent)		
SD1065	High 16 bits of accumulated pulse number(the unit is pulse equivalent)		
SD1066	Low 16 bits of current output frequency(the unit is pulse number)		
SD1067	High 16 bits of current output frequency(the unit is pulse number)		
SD1068	Low 16 bits of current output frequency(the unit is pulse equivalent)		
SD1069	High 16 bits of current output frequency(the unit is pulse equivalent)		
SD1070	Wrong Pulse message	1： Pulse data block error 2：Equivalent mode：pulse amount/turn、 amount/ turn of movement is 0 3:Code of system parameters block error 4：Pulse data block exceeds max limit 10：Zero return do not set near point signal 11：Speed of zero return is 0 12：Crawling speed of zero return is 0 13 Direction of zero return speed and zero auxiliary speed	
SD1071	Code of error pulse block		
SD1080	Current segment(No. n segment)		PULSE_5

SD1082	Low 16 bits of accumulated pulse number (the unit is pulse number)		
SD1083	High 16 bits of accumulated pulse number (the unit is pulse number)		
SD1084	Low 16 bits of accumulated pulse number(the unit is pulse equivalent)		
SD1085	High 16 bits of accumulated pulse number(the unit is pulse equivalent)		
SD1086	Low 16 bits of current output frequency(the unit is pulse number)		
SD1087	High 16 bits of current output frequency(the unit is pulse number)		
SD1088	Low 16 bits of current output frequency(the unit is pulse equivalent)		
SD1089	High 16 bits of current output frequency(the unit is pulse equivalent)		
SD1090	Wrong Pulse message	1： Pulse data block error 2：Equivalent mode：pulse amount/turn、 amount/ turn of movement is 0 3:Code of system parameters block error 4：Pulse data block exceeds max limit 10：Zero return do not set near point signal 11：Speed of zero return is 0 12：Crawling speed of zero return is 0 13 Direction of zero return speed and zero auxiliary speed	
SD1091	Code of error pulse block		
SD1100	Current segment(No. n segment)		PULSE_6
SD1102	Low 16 bits of accumulated pulse number (the unit is pulse number)		
SD1103	High 16 bits of accumulated pulse number (the unit is pulse number)		
SD1104	Low 16 bits of accumulated pulse number(the unit is pulse equivalent)		
SD1105	High 16 bits of accumulated pulse number(the unit is pulse equivalent)		
SD1106	Low 16 bits of current output frequency(the unit is pulse number)		
SD1107	High 16 bits of current output frequency(the unit is pulse number)		

SD1108	Low 16 bits of current output frequency(the unit is pulse equivalent)		
SD1109	High 16 bits of current output frequency(the unit is pulse equivalent)		
SD1110	Wrong Pulse message	1: Pulse data block error 2:Equivalent mode: pulse amount/turn, amount/ turn of movement is 0 3:Code of system parameters block error 4:Pulse data block exceeds max limit 10:Zero return do not set near point signal 11:Speed of zero return is 0 12:Crawling speed of zero return is 0 13 Direction of zero return speed and zero auxiliary speed	
SD1111	Code of error pulse block		
SD1120	Current segment(No. n segment)		PULSE_7
SD1122	Low 16 bits of accumulated pulse number (the unit is pulse number)		
SD1123	High 16 bits of accumulated pulse number (the unit is pulse number)		
SD1124	Low 16 bits of accumulated pulse number(the unit is pulse equivalent)		
SD1125	High 16 bits of accumulated pulse number(the unit is pulse equivalent)		
SD1126	Low 16 bits of current output frequency(the unit is pulse number)		
SD1127	High 16 bits of current output frequency(the unit is pulse number)		
SD1128	Low 16 bits of current output frequency(the unit is pulse equivalent)		
SD1129	High 16 bits of current output frequency(the unit is pulse equivalent)		
SD1130	Wrong Pulse message	1: Pulse data block error 2:Equivalent mode: pulse amount/turn, amount/ turn of movement is 0 3:Code of system parameters block error 4:Pulse data block exceeds max limit	

		10:Zero return do not set near point signal 11:Speed of zero return is 0 12:Crawling speed of zero return is 0 13 Direction of zero return speed and zero auxiliary speed	
SD1131	Code of error pulse block		
SD1140	Current segment(No. n segment)		PULSE_8
SD1142	Low 16 bits of accumulated pulse number (the unit is pulse number)		
SD1143	High 16 bits of accumulated pulse number (the unit is pulse number)		
SD1144	Low 16 bits of accumulated pulse number(the unit is pulse equivalent)		
SD1145	High 16 bits of accumulated pulse number(the unit is pulse equivalent)		
SD1146	Low 16 bits of current output frequency(the unit is pulse number)		
SD1147	High 16 bits of current output frequency(the unit is pulse number)		
SD1148	Low 16 bits of current output frequency(the unit is pulse equivalent)		
SD1149	High 16 bits of current output frequency(the unit is pulse equivalent)		
SD1150	Wrong Pulse message	1: Pulse data block error 2:Equivalent mode: pulse amount/turn, amount/ turn of movement is 0 3:Code of system parameters block error 4:Pulse data block exceeds max limit 10:Zero return do not set near point signal 11:Speed of zero return is 0 12:Crawling speed of zero return is 0 13 Direction of zero return speed and zero auxiliary speed	
SD1151	Code of error pulse block		
SD1160	Current segment(No. n segment)		PULSE_9
SD1162	Low 16 bits of accumulated pulse number (the unit is pulse number)		

SD1163	High 16 bits of accumulated pulse number (the unit is pulse number)		
SD1164	Low 16 bits of accumulated pulse number(the unit is pulse equivalent)		
SD1165	High 16 bits of accumulated pulse number(the unit is pulse equivalent)		
SD1166	Low 16 bits of current output frequency(the unit is pulse number)		
SD1167	High 16 bits of current output frequency(the unit is pulse number)		
SD1168	Low 16 bits of current output frequency(the unit is pulse equivalent)		
SD1169	High 16 bits of current output frequency(the unit is pulse equivalent)		
SD1170	Wrong Pulse message	1: Pulse data block error 2:Equivalent mode: pulse amount/turn, amount/ turn of movement is 0 3:Code of system parameters block error 4:Pulse data block exceeds max limit 10:Zero return do not set near point signal 11:Speed of zero return is 0 12:Crawling speed of zero return is 0 13 Direction of zero return speed and zero auxiliary speed	
SD1171	Code of error pulse block		
SD1180	Current segment(No. n segment)		PULSE_10
SD1182	Low 16 bits of accumulated pulse number (the unit is pulse number)		
SD1183	High 16 bits of accumulated pulse number (the unit is pulse number)		
SD1184	Low 16 bits of accumulated pulse number(the unit is pulse equivalent)		
SD1185	High 16 bits of accumulated pulse number(the unit is pulse equivalent)		
SD1186	Low 16 bits of current output frequency(the unit is pulse number)		
SD1187	High 16 bits of current output frequency(the unit is pulse number)		
SD1188	Low 16 bits of current output frequency(the unit is pulse equivalent)		

SD1189	High 16 bits of current output frequency(the unit is pulse equivalent)		
SD1190	Wrong Pulse message	1: Pulse data block error 2:Equivalent mode: pulse amount/turn, amount/ turn of movement is 0 3:Code of system parameters block error 4:Pulse data block exceeds max limit 10:Zero return do not set near point signal 11:Speed of zero return is 0 12:Crawling speed of zero return is 0 13 Direction of zero return speed and zero auxiliary speed	
SD1191	Code of error pulse block		

Error Check（SD450-SD452）

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	

Expansion Modules, BD Status（SD500-SD516）

ID	Function	Description	
SD500	Module number Expansion modules：#1～16 BD：#10001～10005		
SD501~516	Expansion module、BD status		16 registers

Modules Information（SD520-SD855）

ID	Function	Description	
SD520		Expansion module 1	Each expansion module occupies 16 registers
……			
SD535			
……	……	……	
SD760		Expansion module 16	
……			
SD775			
SD776		BD module 1	Each BD module occupies 16 registers
……			
SD791			
……	……	……	
SD840		BD module 5	
……			
SD855			

Expansion Module Error Information

ID	Function	Description	
SD860	Error times of module read		Expansion module 1
SD861	Error types of module read	Expansion's CRC parity error Expansion's address error Expansion accepted data length error Expansion's accept buffer zone overflows Expansion timeout error CRC parity error when PLC is accepting data Unknown 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	Expansion's CRC parity error Expansion's address error Expansion accepted data length error Expansion's accept buffer zone overflows Expansion timeout error CRC parity error when PLC is accepting data Unknown 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	Expansion's CRC parity error Expansion's address error	

		Expansion accepted data length error Expansion's accept buffer zone overflows Expansion timeout error CRC parity error when PLC is accepting data Unknown error	
SD922	Error times of module write		
SD923	Error types of module write		
SD924	Error times of module read		BD module 1
SD925	Error types of module read		
SD926	Error times of module write		
SD927	Error types of module write		
……			
SD940			BD module 5
SD941			
SD942			
SD943			

Communication

	ID	Function	Description
COM 1	SD130		
	SD131	Serial port communication error code	0：Correct Serial port communication error code : 13：No initial character 14：No ending character 100：Hardware error 101：Timeout error 108：CRC parity error 110：Station number error Modbus communication error code: 211：Function number do not support 212：Address error（overrun） 213：Data length error 214：Data error 215：Slave station busy 216：Data storage error（Erase FLASH）
	SD132		
	SD133		
	SD134		
	SD135		
	SD136		
	SD137		
	SD138		
	SD139		

COM 2	SD140		
	SD141	Serial port communication error code	0：Correct Serail port communication error code : 13：No initial character 14：No ending character 100：Hardware error 101：Timeout error 108：CRC parity error 110：Station number error Modbus communication error code: 211：Function number do not support 212：Address error（overrun） 213：Data length error 214：Data error 215：Slave station busy 216：Data storage error（Erase FLASH）
	SD142		
	SD143		
	SD144		
	SD145		
	SD146		
	SD147		
	SD148		
	SD149		

Special Data Register HSD（Power off retentive）

ID	Function	Description
HSD0	Low 16 bits of accumulated pulse number (the unit is pulse number)	PULSE_1
HSD1	High 16 bits of accumulated pulse number (the unit is pulse number)	
HSD2	Low 16 bits of accumulated pulse number(the unit is pulse equivalent)	
HSD3	High 16 bits of accumulated pulse number(the unit is pulse equivalent)	
HSD4	Low 16 bits of accumulated pulse number (the unit is pulse number)	PULSE_2
HSD5	High 16 bits of accumulated pulse number (the unit is pulse number)	
HSD6	Low 16 bits of accumulated pulse number(the unit is pulse equivalent)	
HSD7	High 16 bits of accumulated pulse number(the unit is pulse equivalent)	
HSD8	Low 16 bits of accumulated pulse number (the unit is pulse number)	PULSE_3
HSD9	High 16 bits of accumulated pulse number (the unit is pulse number)	
HSD10	Low 16 bits of accumulated pulse number(the unit is pulse equivalent)	
HSD11	High 16 bits of accumulated pulse number(the unit is pulse equivalent)	

HSD12	Low 16 bits of accumulated pulse number (the unit is pulse number)	
HSD13	High 16 bits of accumulated pulse number (the unit is pulse number)	
HSD14	Low 16 bits of accumulated pulse number(the unit is pulse equivalent)	
HSD15	High 16 bits of accumulated pulse number(the unit is pulse equivalent)	PULSE_4
HSD16	Low 16 bits of accumulated pulse number (the unit is pulse number)	
HSD17	High 16 bits of accumulated pulse number (the unit is pulse number)	
HSD18	Low 16 bits of accumulated pulse number(the unit is pulse equivalent)	
HSD19	High 16 bits of accumulated pulse number(the unit is pulse equivalent)	PULSE_5
HSD20	Low 16 bits of accumulated pulse number (the unit is pulse number)	
HSD21	High 16 bits of accumulated pulse number (the unit is pulse number)	
HSD22	Low 16 bits of accumulated pulse number(the unit is pulse equivalent)	
HSD23	High 16 bits of accumulated pulse number(the unit is pulse equivalent)	PULSE_6
HSD24	Low 16 bits of accumulated pulse number (the unit is pulse number)	
HSD25	High 16 bits of accumulated pulse number (the unit is pulse number)	
HSD26	Low 16 bits of accumulated pulse number(the unit is pulse equivalent)	
HSD27	High 16 bits of accumulated pulse number(the unit is pulse equivalent)	PULSE_7
HSD28	Low 16 bits of accumulated pulse number (the unit is pulse number)	
HSD29	High 16 bits of accumulated pulse number (the unit is pulse number)	
HSD30	Low 16 bits of accumulated pulse number(the unit is pulse equivalent)	
HSD31	High 16 bits of accumulated pulse number(the unit is pulse equivalent)	PULSE_8
HSD32	Low 16 bits of accumulated pulse number (the unit is pulse number)	
HSD33	High 16 bits of accumulated pulse number (the unit is pulse number)	
HSD34	Low 16 bits of accumulated pulse number(the unit is pulse equivalent)	
HSD35	High 16 bits of accumulated pulse number(the unit is pulse equivalent)	PULSE_9
HSD36	Low 16 bits of accumulated pulse number (the unit is pulse number)	
HSD37	High 16 bits of accumulated pulse number (the unit is pulse number)	
HSD38	Low 16 bits of accumulated pulse number(the unit is pulse equivalent)	
HSD39	High 16 bits of accumulated pulse number(the unit is pulse equivalent)	PULSE_10

Appendix 3. Special Flash Register

Special FLASH data register SFD

*** means it works only after repowering**

I filtering

ID	Function	Description
SFD0*	Input filter time	
SFD2*	Watchdog run-up time, default value is 200ms	

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	0xFF means terminal bad，0xFE means terminal idle
		Default value is 0	
……	……		
SFD134 *	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

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	Description	
SFD350			Configuration of the first expansion module
:			
SFD359			
SFD360			Configuration of the second expansion module
:			
SFD369			
:	:	:	
SFD500			Configuration of the 16th expansion module
:			
SFD509			
SFD510			Configuration 1 of BD module
:			
SFD519			
:	:	:	
SFD550			Configuration 5 of BD module
:			
SFD559			

Communication

ID	Function	Description	Note
COM 1			
SFD600*	Communication mode		Refer to the value meaning of corresponding bit
SFD601*	Communication format	Baud rate, data bit, stop bit, parity	Refer to the value meaning of corresponding bit
SFD602*	Judgment time of frame timeout	In characters	High 8 bits invalid
SFD603*	Judgment time of reply timeout		High 8 bits invalid
SFD604	Waiting time before sending		Unit ms
COM 2			
SFD610*	Communication mode		Refer to the value meaning of corresponding bit
SFD611*	Communication format	Baud rate，data bit，stop bit，parity	Refer to the value meaning of corresponding bit
SFD612*	Judgment time of frame timeout		Unit: ms
SFD613*	Judgment time of reply timeout		Unit: ms，if value is set 0, it means no timeout waiting
SFD614	Waiting time before sending		Unit: ms

Timeout:

If ‘judgment time of frame timeout’ is set 0, then it will finish after accepting one character;8bit unsigned number.

If ‘judgment time of reply timeout is set’ 0, it means no timeout waiting；16bits unsigned number.

If ‘waiting time before sending’ is set 0, it means no time-lapse; 16 bit unsigned number.

Value meaning of SFD600, SFD610 corresponding bits

Corresponding bit	Value meaning
0~7：Modbus station number	Modbus station number
8~15：Communication mode	0：modbus RTU mode (default value) 1：modbus ASCII mode 2：free-format

Value meaning of SFD601, SFD611 corresponding bits

Corresponding bit	Value meaning			
0~3：Baud rate	0x0, BaudRate600	0x1, BaudRate1200	0x2, BaudRate2400	0x3, BaudRate4800
	0x4, BaudRate9600	0x5, BaudRate19200	0x6, BaudRate38400	0x7, BaudRate57600
	0x8, BaudRate115200	0x9, BaudRate192000	0xA, BaudRate256000	0xB, BaudRate288000

	0xC, BaudRate384000	0xD, BaudRate512000	0xE, BaudRate576000	0xF, BaudRate768000
4~7：Data bit	0x0，8 bits	0x1，7 bits		
8~11：Stop bit	0x0，2 bits		0x2，1bit	
12~15：Odd-even parity	0x0，none	0x1，odd parity	0x2，even parity	

Reserved Motion Control Usage

<table>
<tr><th>ID</th><th>Function</th><th>Description</th><th colspan="2"></th></tr>
<tr><td>SFD900</td><td>Pulse parameters setting</td><td>Bit 0：logic of pulse output
0：positive logic；
1：negative logic, default value is 0
Bit 1：logic of pulse direction
0：positive logic；
1：negative logic，default value is 0
Bit 8：unit of pulse
0：pulse number；
1：pulse equivalent，default value is 0</td><td rowspan="14">Common parameters</td><td rowspan="14">pulse _1</td></tr>
<tr><td>SFD901</td><td>Reserved</td><td></td></tr>
<tr><td>SFD902</td><td>Pulse number/1turn of low 16 bits</td><td rowspan="2"></td></tr>
<tr><td>SFD903</td><td>Pulse number/1turn of high 16 bits</td></tr>
<tr><td>SFD904</td><td>Amount of movement/1turn of low 16 bits</td><td rowspan="2"></td></tr>
<tr><td>SFD905</td><td>Amount of movement/1turn of high 16 bits</td></tr>
<tr><td>SFD906</td><td>Pulse direction terminal</td><td>Set number of terminal Y，0xFF means no terminal</td></tr>
<tr><td>SFD907</td><td>Direction delay time</td><td>Default value is 20，unit：ms</td></tr>
<tr><td>SFD908</td><td>Positive compensation of gear clearance</td><td></td></tr>
<tr><td>SFD909</td><td>Negative compensation of gear clearance</td><td></td></tr>
<tr><td>SFD910</td><td>Low 16 bits of Electrical origin position</td><td rowspan="2"></td></tr>
<tr><td>SFD911</td><td>High 16 bits of Electrical origin position</td></tr>
<tr><td>SFD912</td><td>Mechanical back to origin parameter setting</td><td>Bit0：Switch state setting of near point, 0：Normally ON；1Normally OFF</td></tr>
<tr><td>SFD913</td><td>Terminal setting of near point signal</td><td>Bit0~bit7：Assign the number of terminal X, 0Xff for not terminal</td></tr>
</table>

SFD914	Z phase terminal setting	Bit0~bit7：Assign the number of terminal X, 0Xff for not terminal		
SFD915	Limit terminal setting	Bit7~bit0：Assign limit 1 number of terminal X, 0Xff for not terminal Bit15~bit8：Assign limit 2 number of terminal X, 0Xff for not terminal		
SFD916	Terminal setting of origin auxiliary signal	Bit0~bit7：Assign the number of terminal X, 0Xff for not terminal		
SFD917	Terminal setting of zero clear CLR signal output terminal	Bit0~bit7：Assign the number of terminal Y, 0Xff for not terminal		
SFD918	Low 16 bits of return speed VH			
SFD919	High 16 bits of return speed VH			
SFD920	Low 16 bits of return speed VL			
SFD921	High 16 bits of return speed VL			
SFD922	Low 16 bits of crawling speed			
SFD923	High 16 bits of crawling speed			
SFD924	Low 16 bits of mechanical origin			
SFD925	High 16 bits of mechanical origin			
SFD926	Z phase number			
SFD927	CLR signal delay time	Default value is 20, unit：ms		
SFD936	G instruction parameter group using choice	Choose the G instruction parameter group		
…				
SFD950	Low 16 bits of pulse default speed	Only when speed= 0, default speed is used to transmit pulse.	The first set of parameters	
SFD951	High 16 bits of pulse default speed			
SFD952	Accelerating time of pulse default speed			
SFD953	Decelerating time of pulse default speed			
SFD954	Acc and dec time of tween			
SFD955	Reserved			
SFD956	Low 16 bits of max speed limiting			
SFD957	High 16 bits of max speed limiting			
SFD958	Low 16 bits of starting speed			
SFD959	High 16 bits of starting speed			
SFD960	Low 16 bits of ending speed			
SFD961	High 16 bits of ending speed			

SFD962	Curve acceleration time (ms)			
…				
SFD970	Low 16 bits of pulse default speed	Only when speed=0, default speed is used to transmit pulse.	Second set of parameters	
SFD971	High 16 bits of pulse default speed			
SFD972	Accelerating time of pulse default speed			
SFD973	Decelerating time of pulse default speed			
SFD974	Acc and Dec time of tween			
SFD975	Reserved			
SFD976	Low 16 bits of max speed limiting			
SFD977	High 16 bits of max speed limiting			
SFD978	Low 16 bits of starting speed			
SFD979	High 16 bits of starting speed			
SFD980	Low 16 bits of ending speed			
SFD981	High 16 bits of ending speed			
SFD982	Curve acceleration time (ms)			
…				
SFD990	Low 16 bits of pulse default speed	Only when speed=0, default speed is used to transmit pulse.	Third set of parameters	
SFD991	High 16 bits of pulse default speed			
SFD992	Accelerating time of pulse default speed			
SFD993	Decelerating time of pulse default speed			
SFD994	Acc and Dec time of tween			
SFD995	Reserved			
SFD996	Low 16 bits of max speed limiting			
SFD997	High 16 bits of max speed limiting			
SFD998	Low 16 bits of starting speed			
SFD999	High 16 bits of starting speed			
SFD1000	Low 16 bits of ending speed			
SFD1001	High 16 bits of ending speed			
SFD1002	Curve acceleration time (ms)			
…				
SFD1010	Low 16 bits of pulse default speed	Only when speed=0, default speed is used to transmit pulse.	Forth set of parameters	
SFD1011	High 16 bits of pulse default speed			
SFD1012	Accelerating time of pulse default speed			
SFD1013	Decelerating time of pulse default speed			
SFD1014	Acc and Dec time of tween			
SFD1015	Reserved			

SFD1016	Low 16 bits of max speed limiting			
SFD1017	High 16 bits of max speed limiting			
SFD1018	Low 16 bits of starting speed			
SFD1019	High 16 bits of starting speed			
SFD1020	Low 16 bits of ending speed			
SFD1021	High 16 bits of ending speed			
SFD1022	Curve acceleration time (ms)			
…				
SFD1030	Pulse parameters setting	Bit 0：logic of pulse output 0：positive logic; 1：negative logic , default is 0 Bit 1：logic of pulse direction 0：positive logic; 1：negative logic , default is 0 Bit 8：pulse unit 0：pulse number; 1：pulse equivalent, default is 0	Public parameters	PULSE_2
SFD1031				
SFD1032	Low 16 bits of pulse number per circle			
SFD1033	High 16 bits of pulse number per circle			
SFD1034	Low 16 bits of pulse equivalent per circle			
SFD1035	High 16 bits of pulse equivalent per circle			
SFD1036	Pulse direction terminal	Assign the number of terminal Y, 0xFF for no terminal		
SFD1037	Direction delay time	Default 20，unit：ms		
SFD1038	Positive compensation of gear gap	Negative compensation will also use this data when gear gap negative compensation =0		
SFD1039	Negative compensation of gear gap			
SFD1040	Low 16 bits of Electrical origin position			
SFD1041	High 16 bits of Electrical origin position			
SFD1042	Mechanical back to origin parameter setting	Bit0：Switch state setting of near point, 0：Normally ON；1Normally OFF		
SFD1043	Terminal setting of near point signal	Bit0~bit7：Assign the number of terminal X, 0Xff for not terminal		
SFD1044	Z phase terminal setting	Bit0~bit7：Assign the number of terminal X, 0Xff for not terminal		

SFD1045	Limit terminal setting	Bit7~bit0：Assign limit 1 number of terminal X, 0Xff for not terminal Bit15~bit8：Assign limit 2 number of terminal X, 0Xff for not terminal		
SFD1046	Terminal setting of origin auxiliary signal	Bit0~bit7：Assign the number of terminal X, 0Xff for not terminal		
SFD1047	Terminal setting of zero clear CLR signal output terminal	Bit0~bit7：Assign the number of terminal Y, 0Xff for not terminal		
SFD1048	Low 16 bits of return speed VH			
SFD1049	High 16 bits of return speed VH			
SFD1050	Low 16 bits of return speed VL			
SFD1051	High 16 bits of return speed VL			
SFD1052	Low 16 bits of crawling speed			
SFD1053	High 16 bits of crawling speed			
SFD1054	Low 16 bits of mechanical origin			
SFD1055	High 16 bits of mechanical origin			
SFD1056	Z phase number			
SFD1057	CLR signal delay time	Default 20, unit：ms		
SFD936	G instruction parameter group using choice	Choose the G instruction parameter group		
…				
SFD1080	Low 16 bits of pulse default speed	Only when speed=0, default speed is used to transmit pulse.	First set of parameters	
SFD1081	High 16 bits of pulse default speed			
SFD1082	Accelerating time of pulse default speed			
SFD1083	Decelerating time of pulse default speed			
SFD1084	Acc and Dec time of tween			
SFD1085	Reserved			
SFD1086	Low 16 bits of max speed limiting			
SFD1087	High 16 bits of max speed limiting			
SFD1088	Low 16 bits of starting speed			
SFD1089	High 16 bits of starting speed			
SFD1090	Low 16 bits of ending speed			
SFD1091	High 16 bits of ending speed			
SFD1092	Curve acceleration time (ms)			
…				

SFD1100	Low 16 bits of pulse default speed	Only when speed=0, default speed is used to transmit pulse.	Second set of parameters	
SFD1101	High 16 bits of pulse default speed			
SFD1102	Accelerating time of pulse default speed			
SFD1103	Decelerating time of pulse default speed			
SFD1104	Acc and Dec time of tween			
SFD1105	Reserved			
SFD1106	Low 16 bits of max speed limiting			
SFD1107	High 16 bits of max speed limiting			
SFD1108	Low 16 bits of starting speed			
SFD1109	High 16 bits of starting speed			
SFD1110	Low 16 bits of ending speed			
SFD1111	High 16 bits of ending speed			
SFD1112	Curve acceleration time (ms)			
…				
SFD1120	Low 16 bits of pulse default speed	Only when speed=0, default speed is used to transmit pulse.	Third set of parameters	
SFD1121	High 16 bits of pulse default speed			
SFD1122	Accelerating time of pulse default speed			
SFD1123	Decelerating time of pulse default speed			
SFD1124	Acc and Dec time of tween			
SFD1125	Reserved			
SFD1126	Low 16 bits of max speed limiting			
SFD1127	High 16 bits of max speed limiting			
SFD1128	Low 16 bits of starting speed			
SFD1129	High 16 bits of starting speed			
SFD1130	Low 16 bits of ending speed			
SFD1131	High 16 bits of ending speed			
SFD1132	Curve acceleration time (ms)			
…				
SFD1140	Low 16 bits of pulse default speed	Only when speed=0, default speed is used to transmit pulse.	Forth set of parameters	
SFD1141	High 16 bits of pulse default speed			
SFD1142	Accelerating time of pulse default speed			
SFD1143	Decelerating time of pulse default speed			
SFD1144	Acc and Dec time of tween			
SFD1145	Reserved			
SFD1146	Low 16 bits of max speed limiting			

SFD1147	High 16 bits of max speed limiting			
SFD1148	Low 16 bits of starting speed			
SFD1149	High 16 bits of starting speed			
SFD1150	Low 16 bits of ending speed			
SFD1151	High 16 bits of ending speed			
SFD1152	Curve acceleration time (ms)			
…				
SFD1160	Pulse parameters setting	Bit 0：logic of pulse output 0：positive logic; 1：negative logic , default is 0 Bit 1：logic of pulse direction 0：positive logic; 1：negative logic , default is 0 Bit 8：pulse unit 0：pulse number; 1：pulse equivalent, default is 0	Public parameters	PUL SE_3
SFD1161				
SFD1162	Low 16 bits of pulse number per circle			
SFD1163	High 16 bits of pulse number per circle			
SFD1164	Low 16 bits of pulse equivalent per circle			
SFD1165	High 16 bits of pulse equivalent per circle			
SFD1166	Pulse direction terminal	Assign the number of termimal Y, 0xFF for no terminal		
SFD1167	Direction delay time	Default 20，unit：ms		
SFD1168	Positive compensation of gear gap	Negative compensation will also use this data when gear gap negative compensation =0		
SFD1169	Negative compensation of gear gap			
SFD1170	Low 16 bits of Electrical origin position			
SFD1171	High 16 bits of Electrical origin position			
SFD1172	Mechanical back to origin parameter setting	Bit0：Switch state setting of near point, 0：Normally ON；1Normally OFF		
SFD1173	Terminal setting of near point signal	Bit0~bit7：Assign the number of terminal X, 0Xff for not terminal		
SFD1174	Z phase terminal setting	Bit0~bit7：Assign the number of terminal X, 0Xff for not terminal		

SFD1175	Limit terminal setting	Bit7~bit0：Assign limit 1 number of terminal X, 0Xff for not terminal Bit15~bit8：Assign limit 2 number of terminal X, 0Xff for not terminal		
SFD1176	Terminal setting of origin auxiliary signal	Bit0~bit7：Assign the number of terminal X, 0Xff for not terminal		
SFD1177	Terminal setting of zero clear CLR signal output terminal	Bit0~bit7：Assign the number of terminal Y, 0Xff for not terminal		
SFD1178	Low 16 bits of return speed VH			
SFD1179	High 16 bits of return speed VH			
SFD1180	Low 16 bits of return speed VL			
SFD1181	High 16 bits of return speed VL			
SFD1182	Low 16 bits of crawling speed			
SFD1183	High 16 bits of crawling speed			
SFD1184	Low 16 bits of mechanical origin			
SFD1185	High 16 bits of mechanical origin			
SFD1186	Z phase number			
SFD1187	CLR signal delay time	Default 20, unit：ms		
SFD936	G instruction parameter group using choice	Choose the G instruction parameter group		
…				
SFD1210	Low 16 bits of pulse default speed	Only when speed=0, default speed is used to transmit pulse.	First set of parameters	
SFD1211	High 16 bits of pulse default speed			
SFD1212	Accelerating time of pulse default speed			
SFD1213	Decelerating time of pulse default speed			
SFD1214	Acc and Dec time of tween			
SFD1215	Reserved			
SFD1216	Low 16 bits of max speed limiting			
SFD1217	High 16 bits of max speed limiting			
SFD1218	Low 16 bits of starting speed			
SFD1219	High 16 bits of starting speed			
SFD1220	Low 16 bits of ending speed			
SFD1221	High 16 bits of ending speed			
SFD1222	Curve acceleration time (ms)			
…				

SFD1230	Low 16 bits of pulse default speed	Only when speed=0, default speed is used to transmit pulse.	Second set of parameters	
SFD1231	High 16 bits of pulse default speed			
SFD1232	Accelerating time of pulse default speed			
SFD1233	Decelerating time of pulse default speed			
SFD1234	Acc and Dec time of tween			
SFD1235	Reserved			
SFD1236	Low 16 bits of max speed limiting			
SFD1237	High 16 bits of max speed limiting			
SFD1238	Low 16 bits of starting speed			
SFD1239	High 16 bits of starting speed			
SFD1240	Low 16 bits of ending speed			
SFD1241	High 16 bits of ending speed			
SFD1242	Curve acceleration time (ms)			
…				
SFD1250	Low 16 bits of pulse default speed	Only when speed=0, default speed is used to transmit pulse.	Third set of parameters	
SFD1251	High 16 bits of pulse default speed			
SFD1252	Accelerating time of pulse default speed			
SFD1253	Decelerating time of pulse default speed			
SFD1254	Acc and Dec time of tween			
SFD1255	Reserved			
SFD1256	Low 16 bits of max speed limiting			
SFD1257	High 16 bits of max speed limiting			
SFD1258	Low 16 bits of starting speed			
SFD1259	High 16 bits of starting speed			
SFD1260	Low 16 bits of ending speed			
SFD1261	High 16 bits of ending speed			
SFD1262	Curve acceleration time (ms)			
…				
SFD1270	Low 16 bits of pulse default speed	Only when speed=0, default speed is used to transmit pulse.	Forth set of parameters	
SFD1271	High 16 bits of pulse default speed			
SFD1272	Accelerating time of pulse default speed			
SFD1273	Decelerating time of pulse default speed			
SFD1274	Acc and Dec time of tween			
SFD1275	Reserved			
SFD1276	Low 16 bits of max speed limiting			

SFD1277	High 16 bits of max speed limiting			
SFD1278	Low 16 bits of starting speed			
SFD1279	High 16 bits of starting speed			
SFD1280	Low 16 bits of ending speed			
SFD1281	High 16 bits of ending speed			
SFD1282	Curve acceleration time (ms)			
…				
SFD1290	Pulse parameters setting	Bit 0：logic of pulse output 0：positive logic; 1：negative logic , default is 0 Bit 1：logic of pulse direction 0：positive logic; 1：negative logic , default is 0 Bit 8：pulse unit 0：pulse number; 1：pulse equivalent, default is 0	Public parameters	PUL SE_4
SFD1291				
SFD1292	Low 16 bits of pulse number per circle			
SFD1293	High 16 bits of pulse number per circle			
SFD1294	Low 16 bits of pulse equivalent per circle			
SFD1295	High 16 bits of pulse equivalent per circle			
SFD1296	Pulse direction terminal	Assign the number of terminal Y, 0xFF for no terminal		
SFD1297	Direction delay time	Default 20，unit：ms		
SFD1298	Positive compensation of gear gap	Negative compensation will also use this data when gear gap negative compensation =0		
SFD1299	Negative compensation of gear gap			
SFD1300	Low 16 bits of Electrical origin position			
SFD1301	High 16 bits of Electrical origin position			
SFD1302	Mechanical back to origin parameter setting	Bit0：Switch state setting of near point, 0：Normally ON；1Normally OFF		
SFD1303	Terminal setting of near point signal	Bit0~bit7：Assign the number of terminal X, 0Xff for not terminal		
SFD1304	Z phase terminal setting	Bit0~bit7：Assign the number of terminal X, 0Xff for not terminal		

SFD1305	Limit terminal setting	Bit7~bit0：Assign limit 1 number of terminal X, 0Xff for not terminal Bit15~bit8：Assign limit 2 number of terminal X, 0Xff for not terminal		
SFD1306	Terminal setting of origin auxiliary signal	Bit0~bit7：Assign the number of terminal X, 0Xff for not terminal		
SFD1307	Terminal setting of zero clear CLR signal output terminal	Bit0~bit7：Assign the number of terminal Y, 0Xff for not terminal		
SFD1308	Low 16 bits of return speed VH			
SFD1309	High 16 bits of return speed VH			
SFD1310	Low 16 bits of return speed VL			
SFD1311	High 16 bits of return speed VL			
SFD1312	Low 16 bits of crawling speed			
SFD1313	High 16 bits of crawling speed			
SFD1314	Low 16 bits of mechanical origin			
SFD1315	High 16 bits of mechanical origin			
SFD1316	Z phase number			
SFD1317	CLR signal delay time	Default 20, unit：ms		
SFD936	G instruction parameter group using choice	Choose the G instruction parameter group		
…				
SFD1340	Low 16 bits of pulse default speed	Only when speed=0, default speed is used to transmit pulse.	First set of parameters	
SFD1341	High 16 bits of pulse default speed			
SFD1342	Accelerating time of pulse default speed			
SFD1343	Decelerating time of pulse default speed			
SFD1344	Acc and Dec time of tween			
SFD1345	Reserved			
SFD1346	Low 16 bits of max speed limiting			
SFD1347	High 16 bits of max speed limiting			
SFD1348	Low 16 bits of starting speed			
SFD1349	High 16 bits of starting speed			
SFD1350	Low 16 bits of ending speed			
SFD1351	High 16 bits of ending speed			
SFD1352	Curve acceleration time (ms)			
…				

SFD1360	Low 16 bits of pulse default speed	Only when speed=0, default speed is used to transmit pulse.	Second set of parameters	
SFD1361	High 16 bits of pulse default speed			
SFD1362	Accelerating time of pulse default speed			
SFD1363	Decelerating time of pulse default speed			
SFD1364	Acc and Dec time of tween			
SFD1365	Reserved			
SFD1366	Low 16 bits of max speed limiting			
SFD1367	High 16 bits of max speed limiting			
SFD1368	Low 16 bits of starting speed			
SFD1369	High 16 bits of starting speed			
SFD1370	Low 16 bits of ending speed			
SFD1371	High 16 bits of ending speed			
SFD1372	Curve acceleration time (ms)			
…				
SFD1380	Low 16 bits of pulse default speed	Only when speed=0, default speed is used to transmit pulse.	Third set of parameters	
SFD1381	High 16 bits of pulse default speed			
SFD1382	Accelerating time of pulse default speed			
SFD1383	Decelerating time of pulse default speed			
SFD1384	Acc and Dec time of tween			
SFD1385	Reserved			
SFD1386	Low 16 bits of max speed limiting			
SFD1387	High 16 bits of max speed limiting			
SFD1388	Low 16 bits of starting speed			
SFD1389	High 16 bits of starting speed			
SFD1390	Low 16 bits of ending speed			
SFD1391	High 16 bits of ending speed			
SFD1392	Curve acceleration time (ms)			
…				
SFD1400	Low 16 bits of pulse default speed	Only when speed=0, default speed is used to transmit pulse.	Forth set of parameters	
SFD1401	High 16 bits of pulse default speed			
SFD1402	Accelerating time of pulse default speed			
SFD1403	Decelerating time of pulse default speed			
SFD1404	Acc and Dec time of tween			
SFD1405	Reserved			
SFD1406	Low 16 bits of max speed limiting			

SFD1407	High 16 bits of max speed limiting			
SFD1408	Low 16 bits of starting speed			
SFD1409	High 16 bits of starting speed			
SFD1410	Low 16 bits of ending speed			
SFD1411	High 16 bits of ending speed			
SFD1412	Curve acceleration time (ms)			
…				
SFD1420	Pulse parameters setting	Bit 0：logic of pulse output 0：positive logic; 1：negative logic , default is 0 Bit 1：logic of pulse direction 0：positive logic; 1：negative logic , default is 0 Bit 8：pulse unit 0：pulse number; 1：pulse equivalent, default is 0		
SFD1421				
SFD1422	Low 16 bits of pulse number per circle			
SFD1423	High 16 bits of pulse number per circle			
SFD1424	Low 16 bits of pulse equivalent per circle			
SFD1425	High 16 bits of pulse equivalent per circle			
SFD1426	Pulse direction terminal	Assign the number of terminal Y, 0xFF for no terminal	Public parameters	PUL SE_5
SFD1427	Direction delay time	Default 20，unit：ms		
SFD1428	Positive compensation of gear gap	Negative compensation will also use this data when gear gap negative compensation =0		
SFD1429	Negative compensation of gear gap			
SFD1430	Low 16 bits of Electrical origin position			
SFD1431	High 16 bits of Electrical origin position			
SFD1432	Mechanical back to origin parameter setting	Bit0：Switch state setting of near point, 0：Normally ON；1Normally OFF		
SFD1433	Terminal setting of near point signal	Bit0~bit7：Assign the number of terminal X, 0Xff for not terminal		
SFD1434	Z phase terminal setting	Bit0~bit7：Assign the number of terminal X, 0Xff for not terminal		

SFD1435	Limit terminal setting	Bit7~bit0：Assign limit 1 number of terminal X, 0Xff for not terminal Bit15~bit8：Assign limit 2 number of terminal X, 0Xff for not terminal		
SFD1436	Terminal setting of origin auxiliary signal	Bit0~bit7：Assign the number of terminal X, 0Xff for not terminal		
SFD1437	Terminal setting of zero clear CLR signal output terminal	Bit0~bit7：Assign the number of terminal Y, 0Xff for not terminal		
SFD1438	Low 16 bits of return speed VH			
SFD1439	High 16 bits of return speed VH			
SFD1440	Low 16 bits of return speed VL			
SFD1441	High 16 bits of return speed VL			
SFD1442	Low 16 bits of crawling speed			
SFD1443	High 16 bits of crawling speed			
SFD1444	Low 16 bits of mechanical origin			
SFD1445	High 16 bits of mechanical origin			
SFD1446	Z phase number			
SFD1447	CLR signal delay time	Default 20, unit：ms		
SFD936	G instruction parameter group using choice	Choose the G instruction parameter group		
…				
SFD1470	Low 16 bits of pulse default speed	Only when speed=0, default speed is used to transmit pulse.	First set of parameters	
SFD1471	High 16 bits of pulse default speed			
SFD1472	Accelerating time of pulse default speed			
SFD1473	Decelerating time of pulse default speed			
SFD1474	Acc and Dec time of tween			
SFD1475	Reserved			
SFD1476	Low 16 bits of max speed limiting			
SFD1477	High 16 bits of max speed limiting			
SFD1478	Low 16 bits of starting speed			
SFD1479	High 16 bits of starting speed			
SFD1480	Low 16 bits of ending speed			
SFD1481	High 16 bits of ending speed			
SFD1482	Curve acceleration time (ms)			
…				

SFD1490	Low 16 bits of pulse default speed	Only when speed=0, default speed is used to transmit pulse.	Second set of parameters	
SFD1491	High 16 bits of pulse default speed			
SFD1492	Accelerating time of pulse default speed			
SFD1493	Decelerating time of pulse default speed			
SFD1494	Acc and Dec time of tween			
SFD1495	Reserved			
SFD1496	Low 16 bits of max speed limiting			
SFD1497	High 16 bits of max speed limiting			
SFD1498	Low 16 bits of starting speed			
SFD1499	High 16 bits of starting speed			
SFD1500	Low 16 bits of ending speed			
SFD1501	High 16 bits of ending speed			
SFD1502	Curve acceleration time (ms)			
…				
SFD1510	Low 16 bits of pulse default speed	Only when speed=0, default speed is used to transmit pulse.	Third set of parameters	
SFD1511	High 16 bits of pulse default speed			
SFD1512	Accelerating time of pulse default speed			
SFD1513	Decelerating time of pulse default speed			
SFD1514	Acc and Dec time of tween			
SFD1515	Reserved			
SFD1516	Low 16 bits of max speed limiting			
SFD1517	High 16 bits of max speed limiting			
SFD1518	Low 16 bits of starting speed			
SFD1519	High 16 bits of starting speed			
SFD1520	Low 16 bits of ending speed			
SFD1521	High 16 bits of ending speed			
SFD1522	Curve acceleration time (ms)			
…				
SFD1530	Low 16 bits of pulse default speed	Only when speed=0, default speed is used to transmit pulse.	Forth set of parameters	
SFD1531	High 16 bits of pulse default speed			
SFD1532	Accelerating time of pulse default speed			
SFD1533	Decelerating time of pulse default speed			
SFD1534	Acc and Dec time of tween			
SFD1535	Reserved			
SFD1536	Low 16 bits of max speed limiting			

SFD1537	High 16 bits of max speed limiting			
SFD1538	Low 16 bits of starting speed			
SFD1539	High 16 bits of starting speed			
SFD1540	Low 16 bits of ending speed			
SFD1541	High 16 bits of ending speed			
SFD1542	Curve acceleration time (ms)			
…				
SFD1550	Pulse parameters setting	Bit 0：logic of pulse output 0：positive logic; 1：negative logic , default is 0 Bit 1：logic of pulse direction 0：positive logic; 1：negative logic , default is 0 Bit 8：pulse unit 0：pulse number; 1：pulse equivalent, default is 0		
SFD1551				
SFD1552	Low 16 bits of pulse number per circle			
SFD1553	High 16 bits of pulse number per circle			
SFD1554	Low 16 bits of pulse equivalent per circle			
SFD1555	High 16 bits of pulse equivalent per circle			
SFD1556	Pulse direction terminal	Assign the number of terminal Y, 0xFF for no terminal	Public parameters	PUL SE_6
SFD1557	Direction delay time	Default 20，unit：ms		
SFD1558	Positive compensation of gear gap	Negative compensation will also use this data when gear gap negative compensation =0		
SFD1559	Negative compensation of gear gap			
SFD1560	Low 16 bits of Electrical origin position			
SFD1561	High 16 bits of Electrical origin position			
SFD1562	Mechanical back to origin parameter setting	Bit0：Switch state setting of near point, 0：Normally ON；1Normally OFF		
SFD1563	Terminal setting of near point signal	Bit0~bit7：Assign the number of terminal X, 0Xff for not terminal		
SFD1564	Z phase terminal setting	Bit0~bit7：Assign the number of terminal X, 0Xff for not terminal		

SFD1565	Limit terminal setting	Bit7~bit0：Assign limit 1 number of terminal X, 0Xff for not terminal Bit15~bit8：Assign limit 2 number of terminal X, 0Xff for not terminal		
SFD1566	Terminal setting of origin auxiliary signal	Bit0~bit7：Assign the number of terminal X, 0Xff for not terminal		
SFD1567	Terminal setting of zero clear CLR signal output terminal	Bit0~bit7：Assign the number of terminal Y, 0Xff for not terminal		
SFD1568	Low 16 bits of return speed VH			
SFD1569	High 16 bits of return speed VH			
SFD1570	Low 16 bits of return speed VL			
SFD1571	High 16 bits of return speed VL			
SFD1572	Low 16 bits of crawling speed			
SFD1573	High 16 bits of crawling speed			
SFD1574	Low 16 bits of mechanical origin			
SFD1575	High 16 bits of mechanical origin			
SFD1576	Z phase number			
SFD1577	CLR signal delay time	Default 20, unit：ms		
SFD936	G instruction parameter group using choice	Choose the G instruction parameter group		
…				
SFD1600	Low 16 bits of pulse default speed	Only when speed=0, default speed is used to transmit pulse.	First set of parameters	
SFD1601	High 16 bits of pulse default speed			
SFD1602	Accelerating time of pulse default speed			
SFD1603	Decelerating time of pulse default speed			
SFD1604	Acc and Dec time of tween			
SFD1605	Reserved			
SFD1606	Low 16 bits of max speed limiting			
SFD1607	High 16 bits of max speed limiting			
SFD1608	Low 16 bits of starting speed			
SFD1609	High 16 bits of starting speed			
SFD1610	Low 16 bits of ending speed			
SFD1611	High 16 bits of ending speed			
SFD1612	Curve acceleration time (ms)			
…				

SFD1620	Low 16 bits of pulse default speed	Only when speed=0, default speed is used to transmit pulse.	Second set of parameters	
SFD1621	High 16 bits of pulse default speed			
SFD1622	Accelerating time of pulse default speed			
SFD1623	Decelerating time of pulse default speed			
SFD1624	Acc and Dec time of tween			
SFD1625	Reserved			
SFD1626	Low 16 bits of max speed limiting			
SFD1627	High 16 bits of max speed limiting			
SFD1628	Low 16 bits of starting speed			
SFD1629	High 16 bits of starting speed			
SFD1630	Low 16 bits of ending speed			
SFD1631	High 16 bits of ending speed			
SFD1632	Curve acceleration time (ms)			
…				
SFD1640	Low 16 bits of pulse default speed	Only when speed=0, default speed is used to transmit pulse.	Third set of parameters	
SFD1641	High 16 bits of pulse default speed			
SFD1642	Accelerating time of pulse default speed			
SFD1643	Decelerating time of pulse default speed			
SFD1644	Acc and Dec time of tween			
SFD1645	Reserved			
SFD1646	Low 16 bits of max speed limiting			
SFD1647	High 16 bits of max speed limiting			
SFD1648	Low 16 bits of starting speed			
SFD1649	High 16 bits of starting speed			
SFD1650	Low 16 bits of ending speed			
SFD1651	High 16 bits of ending speed			
SFD1652	Curve acceleration time (ms)			
…				
SFD1660	Low 16 bits of pulse default speed	Only when speed=0, default speed is used to transmit pulse.	Forth set of parameters	
SFD1661	High 16 bits of pulse default speed			
SFD1662	Accelerating time of pulse default speed			
SFD1663	Decelerating time of pulse default speed			
SFD1664	Acc and Dec time of tween			
SFD1665	Reserved			
SFD1666	Low 16 bits of max speed limiting			

SFD1667	High 16 bits of max speed limiting			
SFD1668	Low 16 bits of starting speed			
SFD1669	High 16 bits of starting speed			
SFD1670	Low 16 bits of ending speed			
SFD1671	High 16 bits of ending speed			
SFD1542	Curve acceleration time (ms)			
…				
SFD1680	Pulse parameters setting	Bit 0：logic of pulse output 0：positive logic; 1：negative logic , default is 0 Bit 1：logic of pulse direction 0：positive logic; 1：negative logic , default is 0 Bit 8：pulse unit 0：pulse number; 1：pulse equivalent, default is 0	Public parameters	PULSE_7
SFD1681				
SFD1682	Low 16 bits of pulse number per circle			
SFD1683	High 16 bits of pulse number per circle			
SFD1684	Low 16 bits of pulse equivalent per circle			
SFD1685	High 16 bits of pulse equivalent per circle			
SFD1686	Pulse direction terminal	Assign the number of terminal Y, 0xFF for no terminal		
SFD1687	Direction delay time	Default 20，unit：ms		
SFD1688	Positive compensation of gear gap	Negative compensation will also use this data when gear gap negative compensation =0		
SFD1689	Negative compensation of gear gap			
SFD1690	Low 16 bits of Electrical origin position			
SFD1691	High 16 bits of Electrical origin position			
SFD1692	Mechanical back to origin parameter setting	Bit0：Switch state setting of near point, 0：Normally ON；1Normally OFF		
SFD1693	Terminal setting of near point signal	Bit0~bit7：Assign the number of terminal X, 0Xff for not terminal		
SFD1694	Z phase terminal setting	Bit0~bit7：Assign the number of terminal X, 0Xff for not terminal		

SFD1695	Limit terminal setting	Bit7~bit0：Assign limit 1 number of terminal X, 0Xff for not terminal Bit15~bit8：Assign limit 2 number of terminal X, 0Xff for not terminal		
SFD1696	Terminal setting of origin auxiliary signal	Bit0~bit7：Assign the number of terminal X, 0Xff for not terminal		
SFD1697	Terminal setting of zero clear CLR signal output terminal	Bit0~bit7：Assign the number of terminal Y, 0Xff for not terminal		
SFD1698	Low 16 bits of return speed VH			
SFD1699	High 16 bits of return speed VH			
SFD1700	Low 16 bits of return speed VL			
SFD1701	High 16 bits of return speed VL			
SFD1702	Low 16 bits of crawling speed			
SFD1703	High 16 bits of crawling speed			
SFD1704	Low 16 bits of mechanical origin			
SFD1705	High 16 bits of mechanical origin			
SFD1706	Z phase number			
SFD1707	CLR signal delay time	Default 20, unit：ms		
SFD936	G instruction parameter group using choice	Choose the G instruction parameter group		
…				
SFD1730	Low 16 bits of pulse default speed	Only when speed is 0, default speed is used to transmit pulse.	First set of parameters	
SFD1731	High 16 bits of pulse default speed			
SFD1732	Accelerating time of pulse default speed			
SFD1733	Decelerating time of pulse default speed			
SFD1734	Acc and Dec time of tween			
SFD1735	Reserved			
SFD1736	Low 16 bits of max speed limiting			
SFD1737	High 16 bits of max speed limiting			
SFD1738	Low 16 bits of starting speed			
SFD1739	High 16 bits of starting speed			
SFD1740	Low 16 bits of ending speed			
SFD1741	High 16 bits of ending speed			
SFD1742	Curve acceleration time (ms)			
…				

SFD1750	Low 16 bits of pulse default speed	Only when speed is 0, default speed is used to transmit pulse.	Second set of parameters	
SFD1751	High 16 bits of pulse default speed			
SFD1752	Accelerating time of pulse default speed			
SFD1753	Decelerating time of pulse default speed			
SFD1754	Acc and Dec time of tween			
SFD1755	Reserved			
SFD1756	Low 16 bits of max speed limiting			
SFD1757	High 16 bits of max speed limiting			
SFD1758	Low 16 bits of starting speed			
SFD1759	High 16 bits of starting speed			
SFD1760	Low 16 bits of ending speed			
SFD1761	High 16 bits of ending speed			
SFD1762	Curve acceleration time (ms)			
…				
SFD1770	Low 16 bits of pulse default speed	Only when speed=0, default speed is used to transmit pulse.	Third set of parameters	
SFD1771	High 16 bits of pulse default speed			
SFD1772	Accelerating time of pulse default speed			
SFD1773	Decelerating time of pulse default speed			
SFD1774	Acc and Dec time of tween			
SFD1775	Reserved			
SFD1776	Low 16 bits of max speed limiting			
SFD1777	High 16 bits of max speed limiting			
SFD1778	Low 16 bits of starting speed			
SFD1779	High 16 bits of starting speed			
SFD1780	Low 16 bits of ending speed			
SFD1781	High 16 bits of ending speed			
SFD1782	Curve acceleration time (ms)			
…				
SFD1790	Low 16 bits of pulse default speed	Only when speed=0, default speed is used to transmit pulse.	Forth set of parameters	
SFD1791	High 16 bits of pulse default speed			
SFD1792	Accelerating time of pulse default speed			
SFD1793	Decelerating time of pulse default speed			
SFD1794	Acc and Dec time of tween			
SFD1795	Reserved			
SFD1796	Low 16 bits of max speed limiting			

SFD1797	High 16 bits of max speed limiting			
SFD1798	Low 16 bits of starting speed			
SFD1799	High 16 bits of starting speed			
SFD1800	Low 16 bits of ending speed			
SFD1801	High 16 bits of ending speed			
SFD1802	Curve acceleration time (ms)			
…				
SFD1810	Pulse parameters setting	Bit 0：logic of pulse output 0：positive logic; 1：negative logic , default is 0 Bit 1：logic of pulse direction 0：positive logic; 1：negative logic , default is 0 Bit 8：pulse unit 0：pulse number; 1：pulse equivalent, default is 0	Public parameters	PULSE_8
SFD1811				
SFD1812	Low 16 bits of pulse number per circle			
SFD1813	High 16 bits of pulse number per circle			
SFD1814	Low 16 bits of pulse equivalent per circle			
SFD1815	High 16 bits of pulse equivalent per circle			
SFD1816	Pulse direction terminal	Assign the number of terminal Y, 0xFF for no terminal		
SFD1817	Direction delay time	Default 20，unit：ms		
SFD1818	Positive compensation of gear gap	Negative compensation will also use this data when gear gap negative compensation =0		
SFD1819	Negative compensation of gear gap			
SFD1820	Low 16 bits of Electrical origin position			
SFD1821	High 16 bits of Electrical origin position			
SFD1822	Mechanical back to origin parameter setting	Bit0：Switch state setting of near point, 0：Normally ON; 1Normally OFF		
SFD1823	Terminal setting of near point signal	Bit0~bit7：Assign the number of terminal X, 0Xff for not terminal		
SFD1824	Z phase terminal setting	Bit0~bit7：Assign the number of terminal X, 0Xff for not terminal		

SFD1825	Limit terminal setting	Bit7~bit0：Assign limit 1 number of terminal X, 0Xff for not terminal Bit15~bit8：Assign limit 2 number of terminal X, 0Xff for not terminal		
SFD1826	Terminal setting of origin auxiliary signal	Bit0~bit7：Assign the number of terminal X, 0Xff for not terminal		
SFD1827	Terminal setting of zero clear CLR signal output terminal	Bit0~bit7：Assign the number of terminal Y, 0Xff for not terminal		
SFD1828	Low 16 bits of return speed VH			
SFD1829	High 16 bits of return speed VH			
SFD1830	Low 16 bits of return speed VL			
SFD1831	High 16 bits of return speed VL			
SFD1832	Low 16 bits of crawling speed			
SFD1833	High 16 bits of crawling speed			
SFD1834	Low 16 bits of mechanical origin			
SFD1835	High 16 bits of mechanical origin			
SFD1836	Z phase number			
SFD1837	CLR signal delay time	Default 20, unit：ms		
SFD936	G instruction parameter group using choice	Choose the G instruction parameter group		
…				
SFD1860	Low 16 bits of pulse default speed	Only when speed=0, default speed is used to transmit pulse.	First set of parameters	
SFD1861	High 16 bits of pulse default speed			
SFD1862	Accelerating time of pulse default speed			
SFD1863	Decelerating time of pulse default speed			
SFD1864	Acc and Dec time of tween			
SFD1865	Reserved			
SFD1866	Low 16 bits of max speed limiting			
SFD1867	High 16 bits of max speed limiting			
SFD1868	Low 16 bits of starting speed			
SFD1869	High 16 bits of starting speed			
SFD1870	Low 16 bits of ending speed			
SFD1871	High 16 bits of ending speed			
SFD1872	Curve acceleration time (ms)			
…				

SFD1880	Low 16 bits of pulse default speed	Only when speed=0, default speed is used to transmit pulse.	Second set of parameters	
SFD1881	High 16 bits of pulse default speed			
SFD1882	Accelerating time of pulse default speed			
SFD1883	Decelerating time of pulse default speed			
SFD1884	Acc and Dec time of tween			
SFD1885	Reserved			
SFD1886	Low 16 bits of max speed limiting			
SFD1887	High 16 bits of max speed limiting			
SFD1888	Low 16 bits of starting speed			
SFD1889	High 16 bits of starting speed			
SFD1890	Low 16 bits of ending speed			
SFD1891	High 16 bits of ending speed			
SFD1892	Curve acceleration time (ms)			
…				
SFD1900	Low 16 bits of pulse default speed	Only when speed=0, default speed is used to transmit pulse.	Third set of parameters	
SFD1901	High 16 bits of pulse default speed			
SFD1902	Accelerating time of pulse default speed			
SFD1903	Decelerating time of pulse default speed			
SFD1904	Acc and Dec time of tween			
SFD1905	Reserved			
SFD1906	Low 16 bits of max speed limiting			
SFD1907	High 16 bits of max speed limiting			
SFD1908	Low 16 bits of starting speed			
SFD1909	High 16 bits of starting speed			
SFD1910	Low 16 bits of ending speed			
SFD1911	High 16 bits of ending speed			
SFD1912	Curve acceleration time (ms)			
…				
SFD1920	Low 16 bits of pulse default speed	Only when speed=0, default speed is used to transmit pulse.	Forth set of parameters	
SFD1921	High 16 bits of pulse default speed			
SFD1922	Accelerating time of pulse default speed			
SFD1923	Decelerating time of pulse default speed			
SFD1924	Acc and Dec time of tween			
SFD1925	Reserved			
SFD1926	Low 16 bits of max speed limiting			

SFD1927	High 16 bits of max speed limiting			
SFD1928	Low 16 bits of starting speed			
SFD1929	High 16 bits of starting speed			
SFD1930	Low 16 bits of ending speed			
SFD1931	High 16 bits of ending speed			
SFD1932	Curve acceleration time (ms)			
…				
SFD1940	Pulse parameters setting	Bit 0：logic of pulse output 0：positive logic; 1：negative logic , default is 0 Bit 1：logic of pulse direction 0：positive logic; 1：negative logic , default is 0 Bit 8：pulse unit 0：pulse number; 1：pulse equivalent, default is 0	Public parameters	PULSE_9
SFD1941				
SFD1942	Low 16 bits of pulse number per circle			
SFD1943	High 16 bits of pulse number per circle			
SFD1944	Low 16 bits of pulse equivalent per circle			
SFD1945	High 16 bits of pulse equivalent per circle			
SFD1946	Pulse direction terminal	Assign the number of terminal Y, 0xFF for no terminal		
SFD1947	Direction delay time	Default 20，unit：ms		
SFD1948	Positive compensation of gear gap	Negative compensation will also use this data when gear gap negative compensation =0		
SFD1949	Negative compensation of gear gap			
SFD1950	Low 16 bits of Electrical origin position			
SFD1951	High 16 bits of Electrical origin position			
SFD1952	Mechanical back to origin parameter setting	Bit0：Switch state setting of near point, 0：Normally ON；1Normally OFF		
SFD1953	Terminal setting of near point signal	Bit0~bit7：Assign the number of terminal X, 0Xff for not terminal		
SFD1954	Z phase terminal setting	Bit0~bit7：Assign the number of terminal X, 0Xff for not terminal		

SFD1955	Limit terminal setting	Bit7~bit0：Assign limit 1 number of terminal X, 0Xff for not terminal Bit15~bit8：Assign limit 2 number of terminal X, 0Xff for not terminal		
SFD1956	Terminal setting of origin auxiliary signal	Bit0~bit7：Assign the number of terminal X, 0Xff for not terminal		
SFD1957	Terminal setting of zero clear CLR signal output terminal	Bit0~bit7：Assign the number of terminal Y, 0Xff for not terminal		
SFD1958	Low 16 bits of return speed VH			
SFD1959	High 16 bits of return speed VH			
SFD1960	Low 16 bits of return speed VL			
SFD1961	High 16 bits of return speed VL			
SFD1962	Low 16 bits of crawling speed			
SFD1963	High 16 bits of crawling speed			
SFD1964	Low 16 bits of mechanical origin			
SFD1965	High 16 bits of mechanical origin			
SFD1966	Z phase number			
SFD1967	CLR signal delay time	Default 20, unit: ms		
SFD936	G instruction parameter group using choice	Choose the G instruction parameter group		
…				
SFD1990	Low 16 bits of pulse default speed	Only when speed=0, default speed is used to transmit pulse.	First set of parameters	
SFD1991	High 16 bits of pulse default speed			
SFD1992	Accelerating time of pulse default speed			
SFD1993	Decelerating time of pulse default speed			
SFD1994	Acc and Dec time of tween			
SFD1995	Reserved			
SFD1996	Low 16 bits of max speed limiting			
SFD1997	High 16 bits of max speed limiting			
SFD1998	Low 16 bits of starting speed			
SFD1999	High 16 bits of starting speed			
SFD2000	Low 16 bits of ending speed			
SFD2001	High 16 bits of ending speed			
SFD2002	Curve acceleration time (ms)			
…				

SFD2010	Low 16 bits of pulse default speed	Only when speed=0, default speed is used to transmit pulse.	Second set of parameters	
SFD2011	High 16 bits of pulse default speed			
SFD2012	Accelerating time of pulse default speed			
SFD2013	Decelerating time of pulse default speed			
SFD2014	Acc and Dec time of tween			
SFD2015	Reserved			
SFD2016	Low 16 bits of max speed limiting			
SFD2017	High 16 bits of max speed limiting			
SFD2018	Low 16 bits of starting speed			
SFD2019	High 16 bits of starting speed			
SFD2020	Low 16 bits of ending speed			
SFD2021	High 16 bits of ending speed			
SFD2022	Curve acceleration time (ms)			
…				
SFD2030	Low 16 bits of pulse default speed	Only when speed=0, default speed is used to transmit pulse.	Third set of parameters	
SFD2031	High 16 bits of pulse default speed			
SFD2032	Accelerating time of pulse default speed			
SFD2033	Decelerating time of pulse default speed			
SFD2034	Acc and Dec time of tween			
SFD2035	Reserved			
SFD2036	Low 16 bits of max speed limiting			
SFD2037	High 16 bits of max speed limiting			
SFD2038	Low 16 bits of starting speed			
SFD2039	High 16 bits of starting speed			
SFD2040	Low 16 bits of ending speed			
SFD2041	High 16 bits of ending speed			
SFD2042	Curve acceleration time (ms)			
…				
SFD2050	Low 16 bits of pulse default speed	Only when speed=0, default speed is used to transmit pulse.	Forth set of parameters	
SFD2051	High 16 bits of pulse default speed			
SFD2052	Accelerating time of pulse default speed			
SFD2053	Decelerating time of pulse default speed			
SFD2054	Acc and Dec time of tween			
SFD2055	Reserved			
SFD2056	Low 16 bits of max speed limiting			

SFD2057	High 16 bits of max speed limiting			
SFD2058	Low 16 bits of starting speed			
SFD2059	High 16 bits of starting speed			
SFD2060	Low 16 bits of ending speed			
SFD2061	High 16 bits of ending speed			
SFD2062	Curve acceleration time (ms)			
…				
SFD2070	Pulse parameters setting	Bit 0：logic of pulse output 0：positive logic; 1：negative logic , default is 0 Bit 1：logic of pulse direction 0：positive logic; 1：negative logic , default is 0 Bit 8：pulse unit 0：pulse number; 1：pulse equivalent, default is 0	Public parameters	PULSE_10
SFD2071				
SFD2072	Low 16 bits of pulse number per circle			
SFD2073	High 16 bits of pulse number per circle			
SFD2074	Low 16 bits of pulse equivalent per circle			
SFD2075	High 16 bits of pulse equivalent per circle			
SFD2076	Pulse direction terminal	Assign the number of terminal Y, 0xFF for no terminal		
SFD2077	Direction delay time	Default 20，unit：ms		
SFD2078	Positive compensation of gear gap	Negative compensation will also use this data when gear gap negative compensation =0		
SFD2079	Negative compensation of gear gap			
SFD2080	Low 16 bits of Electrical origin position			
SFD2081	High 16 bits of Electrical origin position			
SFD2082	Mechanical back to origin parameter setting	Bit0：Switch state setting of near point, 0：Normally ON；1Normally OFF		
SFD2083	Terminal setting of near point signal	Bit0~bit7：Assign the number of terminal X, 0Xff for not terminal		
SFD2084	Z phase terminal setting	Bit0~bit7：Assign the number of terminal X, 0Xff for not terminal		

SFD2085	Limit terminal setting	Bit7~bit0：Assign limit 1 number of terminal X, 0Xff for not terminal Bit15~bit8：Assign limit 2 number of terminal X, 0Xff for not terminal		
SFD2086	Terminal setting of origin auxiliary signal	Bit0~bit7：Assign the number of terminal X, 0Xff for not terminal		
SFD2087	Terminal setting of zero clear CLR signal output terminal	Bit0~bit7：Assign the number of terminal Y, 0Xff for not terminal		
SFD2088	Low 16 bits of return speed VH			
SFD2089	High 16 bits of return speed VH			
SFD2090	Low 16 bits of return speed VL			
SFD2091	High 16 bits of return speed VL			
SFD2092	Low 16 bits of crawling speed			
SFD2093	High 16 bits of crawling speed			
SFD2094	Low 16 bits of mechanical origin			
SFD2095	High 16 bits of mechanical origin			
SFD2096	Z phase number			
SFD2097	CLR signal delay time	Default 20, unit: ms		
SFD936	G instruction parameter group using choice	Choose the G instruction parameter group		
…				
SFD2120	Low 16 bits of pulse default speed	Only when speed is 0, default speed is used to transmit pulse.	First set of parameters	
SFD2121	High 16 bits of pulse default speed			
SFD2122	Accelerating time of pulse default speed			
SFD2123	Decelerating time of pulse default speed			
SFD2124	Acc and Dec time of tween			
SFD2125	Reserved			
SFD2126	Low 16 bits of max speed limiting			
SFD2127	High 16 bits of max speed limiting			
SFD2128	Low 16 bits of starting speed			
SFD2129	High 16 bits of starting speed			
SFD2130	Low 16 bits of ending speed			
SFD2131	High 16 bits of ending speed			
SFD2132	Curve acceleration time (ms)			
…				

SFD2140	Low 16 bits of pulse default speed	Only when speed=0, default speed is used to transmit pulse.	Second set of parameters	
SFD2141	High 16 bits of pulse default speed			
SFD2142	Accelerating time of pulse default speed			
SFD2143	Decelerating time of pulse default speed			
SFD2144	Acc and Dec time of tween			
SFD2145	Reserved			
SFD2146	Low 16 bits of max speed limiting			
SFD2147	High 16 bits of max speed limiting			
SFD2148	Low 16 bits of starting speed			
SFD2149	High 16 bits of starting speed			
SFD2150	Low 16 bits of ending speed			
SFD2151	High 16 bits of ending speed			
SFD2152	Curve acceleration time (ms)			
…				
SFD2160	Low 16 bits of pulse default speed	Only when speed=0, default speed is used to transmit pulse.	Third set of parameters	
SFD2161	High 16 bits of pulse default speed			
SFD2162	Accelerating time of pulse default speed			
SFD2163	Decelerating time of pulse default speed			
SFD2164	Acc and Dec time of tween			
SFD2165	Reserved			
SFD2166	Low 16 bits of max speed limiting			
SFD2167	High 16 bits of max speed limiting			
SFD2168	Low 16 bits of starting speed			
SFD2169	High 16 bits of starting speed			
SFD2170	Low 16 bits of ending speed			
SFD2171	High 16 bits of ending speed			
SFD2172	Curve acceleration time (ms)			
…				
SFD2180	Low 16 bits of pulse default speed	Only when speed=0, default speed is used to transmit pulse.	Forth set of parameters	
SFD2181	High 16 bits of pulse default speed			
SFD2182	Accelerating time of pulse default speed			
SFD2183	Decelerating time of pulse default speed			
SFD2184	Acc and Dec time of tween			
SFD2185	Reserved			
SFD2186	Low 16 bits of max speed limiting			

SFD2187	High 16 bits of max speed limiting			
SFD2188	Low 16 bits of starting speed			
SFD2189	High 16 bits of starting speed			
SFD2190	Low 16 bits of ending speed			
SFD2191	High 16 bits of ending speed			
SFD2192	Curve acceleration time (ms)			
…				

XINJE

WUXI XINJE ELECTRIC CO., LTD.
4th Floor Building 7,No. 100 Dicui Road,Wuxi, China
214072
Tel: (510) 85134139
Fax: (510) 85111290
Email: fiona.xinje@vip.163.com
Web: www.xinje.com