Preface

This manual comprehensively introduces the debugging, programming and application methods of HNC-8 CNC controller, and is the basic manual for users to quickly learn and use this system. The update and upgrade of this manual are authorized and organized by Wuhan Huazhong Numerical Control Co., Ltd (HCNC). Without the authorization or written permission of the company, no unit or individual has the right to modify the contents of this manual, and the company shall not be responsible for the losses caused by that.

In the manual of HNC-8 series CNC controller, we try our best to describe various events related to the application of this controller. Due to space limitations and product development positioning, it is impossible to describe all the unnecessary or impossible events about the controller. Therefore, events that are not specifically described in this manual can be regarded as "impossible" or "not allowed" events.

Limited to the level of editors, there must be shortcomings and inadequacies in the manual, and we hope that users will not hesitate to enlighten us.

Caution

- Regarding "restrictions" and "available functions", the manual provided by the machine tool manufacturer takes precedence over this manual. Before actual machining, please perform dry run, and confirm machining program, tool compensation amount, workpiece offset, etc.
- Matters which are not especially described as possible in this manual should be regarded as" impossible"
- At the time of writing this manual, it is assumed that all optional functions have been equipped. Please check the specifications provided by the machine tool manufacturer when using it.
- For related instructions of machine tool, please refer to the manual provided by the machine tool manufacturer.
- The available screens and functions vary with NC systems (or versions). Be sure to confirm the specifications before use.

HNC-8 Numerical Control System Software

PLC Programming Manual

V2.4

Wuhan Huazhong Numerical Control Co., Ltd.

Content

PREFACE **1**

1 OVERVIEW **8**

1.1 SPECIFICATIONS OF PLC 9
1.2 SEQUENTIAL PROGRAM NOTION 10
1.3 ALLOCATION OF INTERFACE 11
1.4 SEQUENTIAL PROGRAM 12
1.5 SEQUENTIAL PROGRAM COMPOSITION 15
1.6 ADDRESS 17

2 BASIC INSTRUCTION **18**

2.1 LD 21
2.2 LDI 23
2.3 OUT 25
2.4 OOUT 26
2.5 SET 27
2.6 RST 28
2.7 AND 29
2.8 ANI 30
2.9 OR 31
2.10 ORI 32
2.11 LDP 33
2.12 LDF 35
2.13 ANDP 36
2.14 ANDF 37
2.15 ORP 38
2.16 ORF 39
2.17 ORB 40
2.18 ANB 42
2.19 MPS, MRD, MPP 42

3 BASIC ELEMENT **44**

3.1 NORMALLY-OPEN CONTACT 45
3.2 NORMALLY-CLOSED CONTACT 46
3.3 TRUE CONTACT 47
3.4 RISING EDGE OF CONTACT 48
3.5 FALLING EDGE OF CONTACT 49
3.6 LOGIC OUTPUT 50
3.7 INVERTED LOGIC OUTPUT 51
3.8 SETTING OUTPUT 52
3.9 RESET OUTPUT 53

4 BASIC FUNCTION MODULE **54**

4.1 CONTROL INSTRUCTION 55
4.1.1 Instruction M Get MGET *55*

4.1.2 M Instruction Response MACK *56*
4.1.3 T Instruction Get TGET *57*
4.1.4 T Instruction Response TACK *58*
4.1.5 Handwheel Control RTOMPG *59*
4.1.6 Thermal Error Compensation Module TEMPSEN *60*
4.2 MATHEMATICAL OPERATION 62
4.2.1 Addition ADD *62*
4.2.2 Subtraction SUB *64*
4.2.3 Multiplication MUL *65*
4.2.4 Division DIV *66*
4.2.5 Increase One INC *68*
4.2.6 Decrease One DEC *69*
4.2.7 Logic AND WAND *70*
4.2.8 Logic OR WOR *71*
4.2.9 Logic XOR WXOR *72*
4.2.10 Complement NEG *73*
4.3 COUNTER 74
4.3.1 Up/Down Counter CTR *74*
4.3.2 Counter CTRC *76*
4.3.3 Custom Up/down Counter CTUD *77*
4.4 TIMER 79
4.4.1 On-delay Timer TMRB *79*
4.4.2 Off-delay Timer STMR *81*
4.5 PROCESS CONTROL 83
4.5.1 Initialization Module End IEND *83*
4.5.2 PLC1 Module End 1END *84*
4.5.3 PLC2 Module End 2END *85*
4.5.4 Jump JMP *86*
4.5.5 Label LBL *87*
4.5.6 Call Subprogram CALL *88*
4.5.7 Subprogram Start SP *89*
4.5.8 Subprogram End SPE *90*
4.5.9 Subprogram Return RETN *91*
4.5.10 Loop LOOP *92*
4.5.11 Next Loop NEXT *93*
4.6 COMPARISON 94
4.6.1 Comparison CMP *94*
4.6.2 Lower Than LT *95*
4.6.3 Area Comparison ACMP *96*
4.6.4 Consistency Comparison COIN *98*
4.7 DATA MANIPULATION 99
4.7.1 Moving Data MOV *99*
4.7.2 Relative Moving Data XMOV *100*
4.7.3 Batch Moving BMOV *102*
4.7.4 Multiple Moves FMOV *110*
4.7.5 Data Exchange XCH *111*
4.7.6 Data Reset ZRST *113*
4.7.7 Encoding ENCO *114*
4.7.8 Decoding DECO *115*
4.7.9 Transformation COD *117*

4.7.10 Data Search SER *121*
4.7.11 Register Merging ASSEM *122*
4.7.12 Register Decomposition DISAS *123*
4.7.13 Area Conversion ACVT *124*
4.7.14 Alternate Output ALT *125*
4.7.15 Fetch Rising Edge PLS *126*
4.7.16 Fetch Falling edge PLF *127*
4.7.17 Points Transformation PTN *128*
4.7.18 Number Conversion NTP *130*
4.7.19 Parts Count PARTCNT *132*
4.7.20 Parts-counting Clear PARTCLR *133*
4.7.21 Temperature Collection Module HEADSEN *134*
4.7.22 Variable Reading Module VARGET *135*
4.7.23 Variable Writing Module VARSET *137*

5 STATUS WORD AND CONTROL WORD PROGRAMMING 139

5.1 INTRODUCTION ON STATUS WORD AND CONTROL WORD 140
5.1.1 Axis Status Word *144*
5.1.2 Axis Control Word *148*
5.1.3 Channel Status Word *152*
5.1.4 Channel Control Word *155*
5.2 EXAMPLE OF STATUS WORD AND CONTROL WORD PROGRAMMING 161
5.2.1 Working Mode Setting *161*
5.2.2 Working Mode Obtaining *162*
5.2.3 Control of Feed Axis and Spindle *163*
5.2.4 Home *164*
5.2.5 Incremental Magnification Override *165*
5.2.6 Cycle Start and Feed Hold *166*
5.2.7 Program Name Specified by Loaded Variable *167*

6 EXTENSION FUNCTION MODULE 168

6.1 NC FUNCTION 169
6.1.1 Channel Mode Setting MDST *169*
6.1.2 Channel Mode Getting MDGT *170*
6.1.3 Mode MDI *171*
6.1.4 Locking Channel MST *172*
6.1.5 Cycle Start CYCLE *173*
6.1.6 Emergency Stop STOP *174*
6.1.7 RESET *175*
6.1.8 Channel Exchange CHANSW *176*
6.1.9 Feed Hold Start HOLD *177*
6.1.10 Cycle Start LED CYCLED *178*
6.1.11 Feed Hold LED HOLDLED *179*
6.1.12 Block Skip（G31）ESCBLK *180*
6.1.13 Rapid Traverse Override RPOVRD *181*
6.1.14 Feedrate Override FEEDOVRD *182*
6.1.15 Spindle Override SPDLOVRD *183*
6.1.16 Incremental (Stepping) Magnification STEPMUL *184*
6.1.17 Dryrun DRYRUN *185*
6.1.18 Block Skip SKIP *190*

6.1.19 User Input USERIN *191*
6.1.20 User Output USEROUT *192*
6.1.21 Optional Stop SELSTOP *193*
6.1.22 Vector Tool Direction Setting TOOLSET *194*
6.1.23 Vector Tool Direction Clear TOOLCLR *195*
6.1.24 8-bit Nixie Tube NIXIE *196*
6.1.25 Tool Display TOOLUSE *197*
6.1.26 Tool Life TOOLLIFE *198*
6.1.27 Tool Selection Module TOOL *199*
6.2 FUNCTIONAL UNIT OF AXIS 200
6.2.1 Spindle JOG SPDLJOG *200*
6.2.2 Spindle Control【Servo Spindle】SPDLBUS *201*
6.2.3 Spindle Control with Gear【Servo Spindle】SPDLBUS1 ..... *202*
6.2.4 Spindle Orientation Enable SPDLORI *204*
6.2.5 Spindle Orientation Completion SPDLOROK *205*
6.2.6 Spindle Control【DA】SPDA *206*
6.2.7 Zero Speed Detection for Spindle SPDLZERO *208*
6.2.8 Spindle Speed Arrival SPDLRCH *209*
6.2.9 Slave Axis Home SUBAXEN *210*
6.2.10 Release Slave Axis DESYN *211*
6.2.11 Axis Jog JOGSW *212*
6.2.12 Axis Stepping STEPAXIS *213*
6.2.13 Jog Velocity JOGVEL *214*
6.2.14 Home Start HOMRUN *215*
6.2.15 Home Start 1 HOMERUN1 ..... *216*
6.2.16 Home Approaching Switch HOMESW *217*
6.2.17 Homing Completion HOMLED *218*
6.2.18 Axis Enable AXEN *219*
6.2.19 Axis Ready (Bus) AXRDY *220*
6.2.20 Axis Lock AXISLOCK *221*
6.2.21 Relative PMC Axis Traverse AXISMOVE *222*
6.2.22 Absolute PMC Axis Movement AXISMVTO *223*
6.2.23 The Second Soft Limit of Axis AXLMF2 ..... *224*
6.2.24 Block Switch in Positive Limit Direction AXISPLMT *225*
6.2.25 Block Switch in Negative Limit Direction AXISNLMT *226*
6.2.26 Handwheel MPGSET *227*
6.2.27 Servo Enable (Bus) SVSW *228*
6.2.28 Axis Working Mode AXISMODE *229*
6.2.29 Axis Reference REFPT *230*
6.2.30 During Axis Home AXISHOM2 ..... *231*
6.2.31 During Axis Moving AXMOVING *232*
6.3 SYSTEM FUNCTION 233
6.3.1 Rotation ROT *233*
6.3.2 Alarm ALARM *235*
6.3.3 Event EVENT *236*
6.3.4 Save Data SAVEDATA *237*
6.3.5 Reset Setting Output RSTCHK *238*
6.3.6 Reset Clear RSTCLR *239*

7 OPERATIONAL MONITORING AND ONLINE MODIFICATION FOR LADDER DIAGRAM 240

7.1 LADDER MONITORING 242
7.2 FIND 244
7.3 EDIT 246
7.3.1 Insert Straight Line 248
7.3.2 Insert Vertical Line 248
7.3.3 Delete Vertical Line 249
7.3.4 Delete Component 249
7.3.5 Normally-open 250
7.3.6 Normally-closed 251
7.3.7 Logical Output 251
7.3.8 Inverted Output 252
7.3.9 Functional Module 253
7.3.10 Return 254
7.4 EDIT NETWORK 255
7.4.1 Select Network 256
7.4.2 Delete Network 256
7.4.3 Move 258
7.4.4 Copy 260
7.4.5 Paste Network 261
7.4.6 Insert Line 261
7.4.7 Insert Column 263
7.4.8 Return 264
7.4.9 Update Modification 265
7.4.10 Undo modification 266
7.5 RETURN 267

8 INSTRUCTION ON PLC DEVELOPMENT ENVIRONMENT 268

8.1 OVERVIEW 269
8.2 INSTALLATION OF DEVELOPMENT ENVIRONMENT 270
8.3 DEVELOPMENT ENVIRONMENT INTERFACE 272
8.4 DEVELOPMENT ENVIRONMENT OPERATION 278

APPENDIX A 291

1 Overview

This chapter includes：

1.1 Specifications of PLC

1.2 Sequential Program Notion

1.3 Allocation of Interface

1.4 Sequential Program

1.5 Sequential Program Composition

1.6 Address

1.1 Specifications of PLC

Specifications Different types of PLC have different program capacities, number of function instructions, and usage range of register.

Specification	HNC8
Programming language	Ladder, STL
Execution cycle of the first level program	1ms
Program capacity Ladder program Statement list Symbol name	 5000 lines 10000 lines 1000
Instruction Basic instruction, Function instruction	
Internal relay of single byte（R） Internal relay of double-byte（W） Internal relay of four-byte（D） Timer（T） Counter（C） Subprogram （S） Label （L） User-defined parameter（P） Holding storage area Timer（T） Counter（C） Four-byte register（B）	400 bytes (R0~~R399) 400 bytes (W0~~W199) 400 bytes (D0~~D99) 128（T0~~T127） 128（C0~~C127） —— —— 200（P0~~P199） 128（T300~~T427） 128（C300~~C427） 200 bytes（B0~~B49）
I/O module (X) (Y)	X0~~X512 Y0~~Y512

1.2 Sequential Program Notion

Notion

A brief description on sequential program is provided before on programming. Sequential program indicates that, the program which logically controls the machine and its relevant devices. For the personnel of electrical automation control engineering, the widely-used control flow is based on sequential control, from which sequential program, a programming method for PLC control, is generated.

Numerical control system firstly converts the program to a format, then CPU decodes and operates it. CPU rapidly reads each instruction in the storage, and operates program via arithmetic operation. Sequential program starts from the ladder diagram and other standard PLC languages. The ladder diagram can be understood as the execution order of CPU arithmetic operation.

The above is performed by PLC programming software, the role of which is to document sequential program.

1.3 Allocation of Interface

Interface

PLC interacts with external devices by external I/O. After control object is identified and relevant input/output signals are calculated, corresponding interfaces can be allocated to devices.

For easier debugging to PLC of numerical control system, input/output points of panel interfaces for series HNC8 have been fixed, and for that of other devices, refer to the electrical principle drawing.

Allocation

Panel points have been configured in the standard PLC programs of Series HNC 8 system, user doesn't need to change their definitions. At programming time, user uses other intermediate registers instead of input/output registers to program. The system interfaces bas been described in Appendix A to make a better understand of panel point distribution of series HNC8. Y487 and Y488 are output addresses of digital tubes on panel, X480 to X491 are panel input signals, and Y480 to Y486 are panel output signals.

1.4 Sequential Program

PLC sequential control is achieved by software, and the principle of it is different from that of common relay circuit. Thus, the principle of sequential control should be fully understood at the time of designing PLC sequential program.

Execution of sequential program

In general relay control circuit, each relay may operate at the same time. In the figure below, when relay A acts, relay D and E can act at the same time (in the event of contact B and C being closed). In PLC sequential control, relays act in sequence. When relay A operates, relay D operates first, and then relay E operates (see figure 2.1(a)), that is, each relay operates in the order described in the ladder diagram.

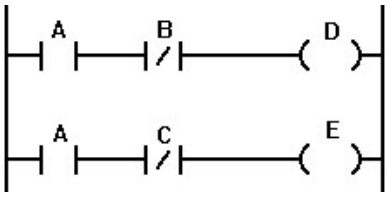


Figure 2.1 (a)

(A) and (B) show the movement difference between relay circuit and PLC program.

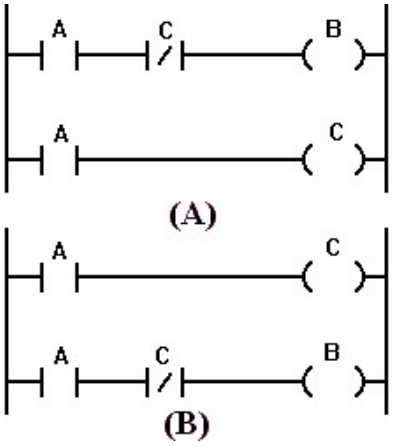


Figure2.1 (b)

Relay circuit

In figure 2.1(b) the actions of (A) and (B) are the same. After A (button switch) is turned on, coil B and C are on, with current flowing through coil B and C. B is cut off after C is switched on.

PLC program

Similar with relay circuit, in figure 2.1b(A), after A (button switch) is turned on, B and C are on, and B is switched off after a PLC program cycle. However, in figure 2.1b(B), after A (button switch) is turned on, C is switched on, but B is not.

Loop execution

Sequential program executes from the beginning of the ladder diagram until the end, after that, it goes back to execute the begging of the ladder diagram again, which is called loop execution.

That execution time from the begging to the end is called loop period. The loop period of PLC2 depends on the controlled steps. The shorter the loop period, the rapider the response of signal.

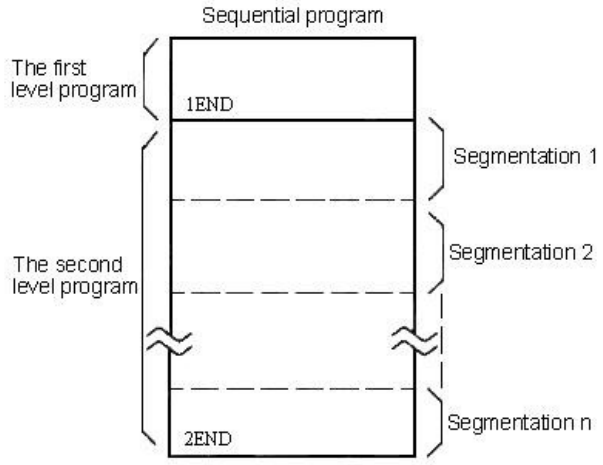
Prior execution

Sequential program consists of three parts: initialization program, the first level program, the second level program.

The initialization program is performed once when system starts.

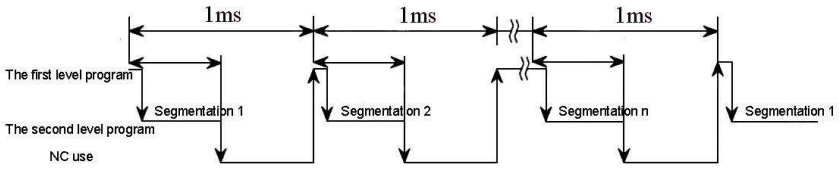
The first level program executes every 1ms.

If the first level is longer, the total execution time will be longer. Therefore, you should document as short program of the first level as possible. The second level program can be automatically separated into n parts for execution, and executes every n ms.



Segmentation of the second level program

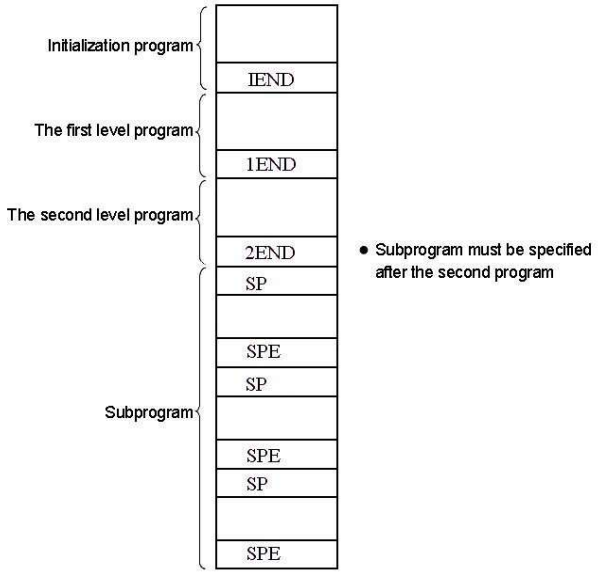
Segmentation of the second level program is to execute the first level program. When the number of segmentations is n, the implementation process is as below diagram:



When the last part of the second level program has been executed, the program starts from the beginning again. In the event of n segmentations, the time for one loop execution is n ms (1ms X n). The first level program executes every 1ms, and the second level program executes every n X 1ms. If the steps of the first level program increase, the steps of the second level program will correspondingly decrease within 1ms, then more segmentations will be gotten, and program processing time will be longer. For this reason, the first level program should be documented as briefly as possible.

The first level program only handles short pulse signal, which includes emergency stop, axis over-travel, and the like.

When subprogram is used, sequential program consists of:



1.5 Sequential Program Composition

Composition

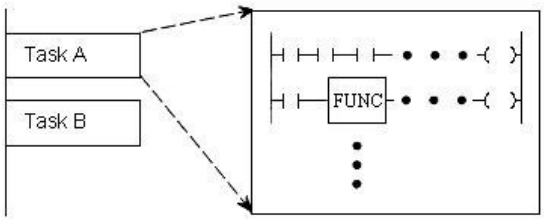
For traditional PLC, ladder diagram can be only documented sequentially. The ladder diagram language, which allows structured program, has the following advantages:

- The program is easily to be understood and documented.
- It is more convenient to find out programming errors.
- It is easy to find out reasons causing errors.

There are three main structured programming means

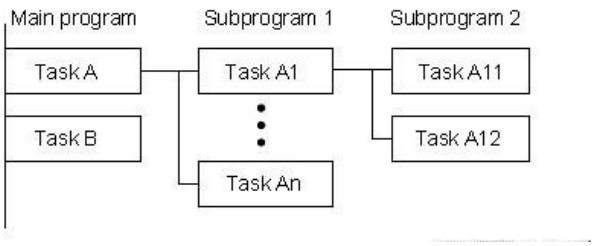
Subprogram

Subprogram regards the ladder diagram block as the processing unit.



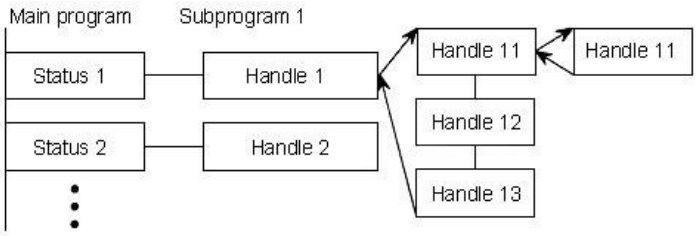
Nesting

The combination of documented subprograms forms structured program.



Conditional branch

Main program executes recurrently and detects whether condition is satisfied or not. If condition is satisfied, corresponding subprogram is executed; if condition is not satisfied, corresponding subprogram is not executed.


Main program
Subprogram 1
Status 1
Handle 1
Status 2
Handle 2
Handle 11
Handle 11
Handle 12
Handle 13

1.6 Address

Address Definition

Address is used to differentiate signals. Various of addresses correspond to input and output signals of machine and CNC, internal relay, counter and the like. The address is composed of address No. and bit No.

Address format

A word must be specified to the left of address No. to express the signal types as below table:

Register	Signal	Range
X	Input signal from machine	X0~~X512
Y	Signal output from PLC to machine	Y0~~Y512
F	Input signal from NC	F0~~F3119
G	Signal output from PLC to NC	G0~~G3119
R	Internal relay of single byte	R0~~R399
W	Internal relay of double-byte	W0~~W199
D	Internal relay of four-byte	D0~~D99
B	Power-off delay relay	B0~~B49
P	User-defined parameter	P0~~P199
C	Counter (Those after C300 is for the power-off delay.)	C0~~C127 C300~~C427
T	Timer (Those after T300 is for the power-off delay.)	T0~~T127 T300~~T427
L	Label number	——
S	Subprogram number	——

2 Basic Instruction

Sequential program is mainly composed of coil, contact, symbol and functional block. The segments, by which elements of ladder diagram are jointed, form the logical relationship of sequential program. Sequential program can be described by ladder diagram language, as well as statement list language which is written by mnemonics (LD, AND, OR, etc.) and register address. Ladder diagram is written by coil contact of relay and functional block.

As the international standard of IEC61131-3 lays out, ladder diagram language and statement list language can convert each other logically, and ambiguity caused by this can be avoided through some programming methods. In HNC_LADDER_WIN(V1.0) editing software, user can see that the two languages can mutually be compiled.

To better understand documenting and inner-making of sequential program, and to avoid errors in logic or understanding, please see the explanation of basic concepts as following:

Type:

PLC instruction of series 8 is divided into basic instruction and functional instruction.

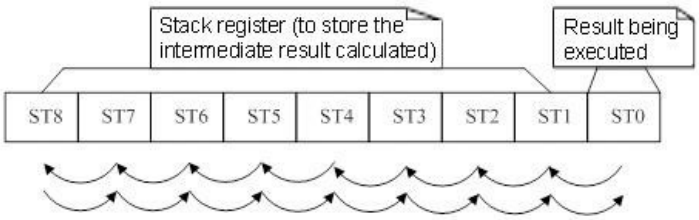
Basic instruction

Basic instruction is the most basic and most common part of sequential program, with a total of 19. It executes one-bit manipulation.

Functional instruction

Functional instruction can perform the functions that is hard to be done by basic elements, and it can simplify programming.

Storage of logical outcome (ST)



Storage of logical outcome is a stack-like structure. Result of the current instruction

is saved in ST0. When reading instruction such as LD and LDI appears, result of current execution is needed to be saved in stack. When ANB or ORB instruction is encountered, the storage makes ST1 result out stack and logically calculate with result in ST0, which then is saved in ST0. Therefore, when sequential program is documented with statement list instruction, ANB and ORB must correspond to the input instructions after the first instruction, one to one; otherwise, errors may occur.

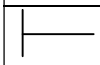
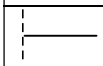
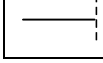
Storage of multi-output logical outcome

The role of this storage is similar with that of logical outcome storage. It is to save result of current node, and is usually used for multi-output instruction with conditional judgements (see detailed command instruction for the usage of MPS, MRD, MPP). What differs from storage of logical outcome is that, it permits reading result of node without stack out of the result. Only when the embedded use of multi-output function is needed is stack operation of storage performed. Similarly, MPS and MPP instructions must be used correspondingly; otherwise, logical errors may occur.

Pre and Post

Pre indicates that other elements can be connected to the front of the element, and post indicates that other elements can be connected to the back of the element.

Here are constraint rules about the graphics in this manual:

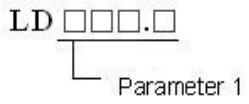
Graphics	Meaning
○	Can be used or not
√	Must be used
×	Cannot be used
	Can use pre component or not
	Must use pre component
	Cannot use pre component
	Can use post component or not
	Must use post component
	Cannot use post component

Detailed basic instructions are listed below:

No.	Instruction	Function
1	LD	Read in specified element signal status
2	LDI	Read in inverted status of specified element signal
3	LDT	Read in true element signal status
4	OUT	Output result of logical operation to specified address
5	OOUT	Output inverted result of logical operation to specified address
6	SET	After Logic OR the line calculation result to signal in specified address, return the result to this address.
7	RST	After Logic AND the inverted calculation result to signal in specified address, return the result to this address.
8	AND	Logic AND
9	ANI	Logic AND the inverted specified signal
10	OR	Logic OR
11	ORI	Logic OR the inverted specified signal
12	LDP	Read in rising edge of signal
13	LDF	Read in falling edge of signal
14	ANDP	Logic AND rising edge of specified signal
15	ANDF	Logic AND falling edge of specified signal
16	ORP	Logic OR rising edge of specified signal
17	ORF	Logic OR falling edge of specified signal
18	ORB	Block logic OR
19	ANB	Block logic AND
20	MPS	Node result push
21	MRD	Node result read
22	MPP	Node result pull

2.1 LD

Format

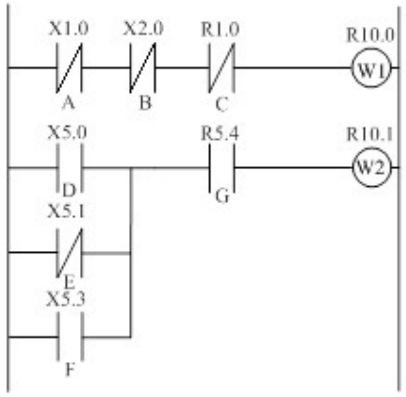


Function Read out status signal (1 or 0) of specified address, and save that signal in ST0. It is used for the situation in which programming starts from the normally-open node.

Parameter Register point parameter

Example

Ladder Diagram



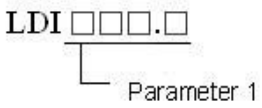
Operation Process

No.	Instruction	Address	Bit No.	Reference	ST2	ST1	ST0
1	LDI	X1 .	0	A			*A*
2	ANI	X2 .	0	B			*A.B*
3	ANI	R1 .	0	C			*A.B.C*
4	OUT	R10 .	0	Output W1			*A.B.C*

	5	LD	X5 . 0	D			D
	6	ORI	X5 . 1	E			$D+E$
	7	OR	X5 . 3	F			$D+E+F$
	8	AND	X5 . 4	G			$(D+E+F)\overline{G}$
	9	OUT	R10 . 0	Output W2			$(D+E+F)\overline{G}$

2.2 LDI

Format

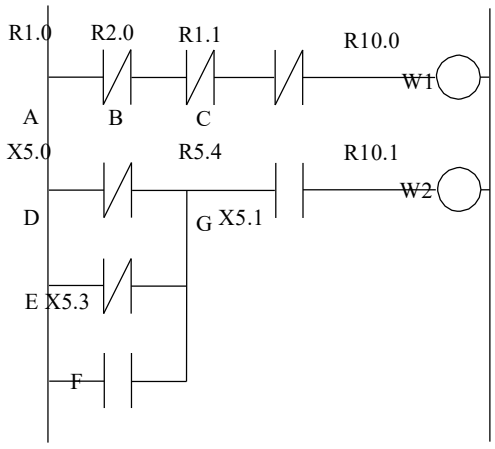


Function Read out status signal (1 or 0) of specified address, and save that inverted signal in ST0. It is used for the situation in which programming starts from normally-closed node.

Parameter Register point parameter

Example

Ladder Diagram



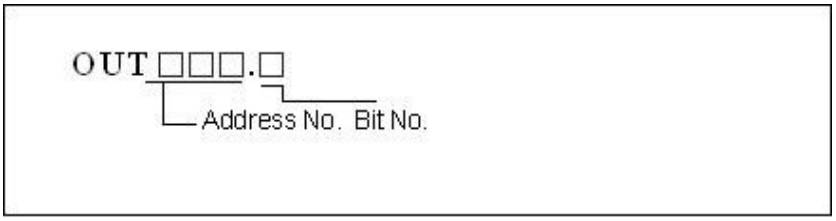
Operation Process

No.	Instruction	Address	Bit No.	Reference	ST2	ST1	ST0
1	LDI	R1 .	0	A			A
2	ANI	R2 .	0	B			$A.B$
3	ANI	R1 .	1	C			$A.B.C$
4	OUT	R10 .	0	W1			$A.B.C$

	5	LDI	X5 . 0	D			$\overline{D}$
	6	ORI	X5 . 1	E			$D\overline{+}E\overline{\ }$
	7	OR	X5 . 3	F			$\overline{D}+\overline{E}+F$
	8	AND	R5 . 4	G			$\left(\overline{D}+\overline{E}+F\right)G$
	9	OUT	R10 . 1	W2			$\left(\overline{D}+\overline{E}+F\right)G$

2.3 OUT

Format

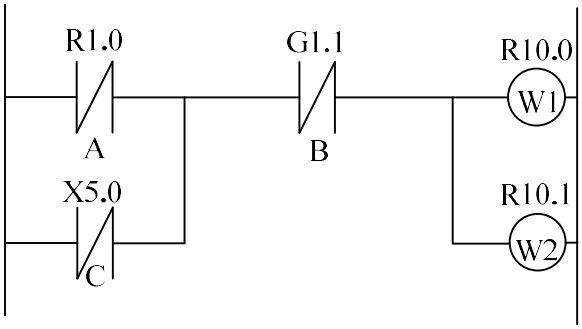


Function Output result of logic operation (status of ST0) to the specified address. It is used to output the result to one or more than one address.

Parameter Register point parameter

Example

Ladder Diagram



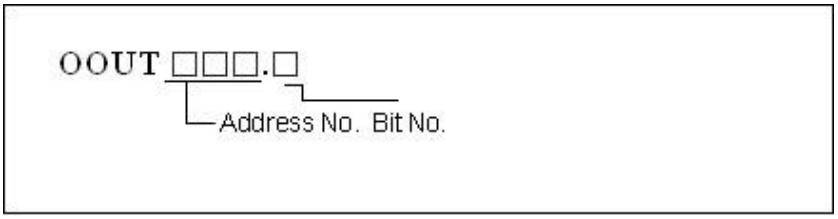
Operation Process

No.	Instruction	Address	Bit No.	Reference	ST2	ST1	ST0
1	LDI	R1 . 0		A			$\overline{A}$
2	ORI	X5 . 0		C			$\overline{A}+\overline{C}$
3	ANI	G1 . 1		B			$(\overline{A}+\overline{C})\overline{B}$
4	OUT	R10 . 0		W1			$\overline{(\overline{A}+\overline{C})\overline{B}}$
5	OUT	R10 . 1		W2			$\overline{(\overline{A}+\overline{C})\overline{B}}$

Description	Cases about series circuit and parallel circuit

2.4 OOUT

Format



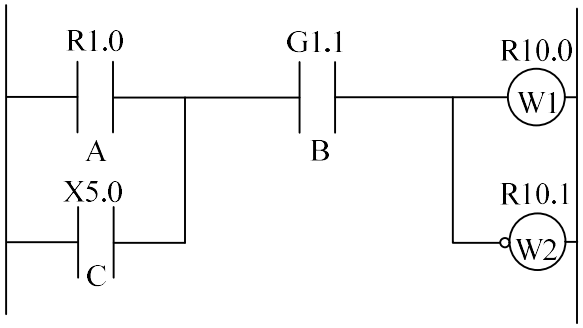
Function Output inverted result of logic operation (status of ST0) to the specified address.

It is used to output the result to one or more than one address.

Parameter Register point parameter

Example

Ladder Diagram

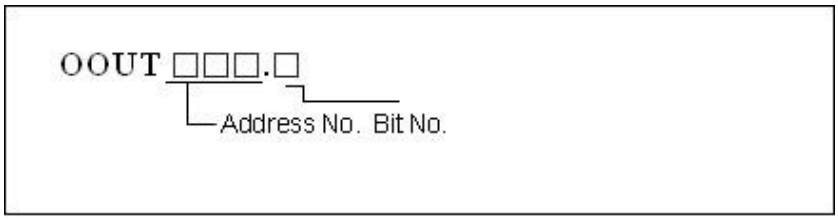


Operation Process

No.	Instruction	Address	Bit No.	Reference	ST2	ST1	ST0
1	LD	R1	0	A			A
2	OR	X5	0	C			$A+C$
3	AND	G1	1	B			$(A+C).B$
4	OUT	R10	0	W1			$(A+C).B$
5	OOUT	R10	1	W2			$(A+C).B$

2.5 SET

Format

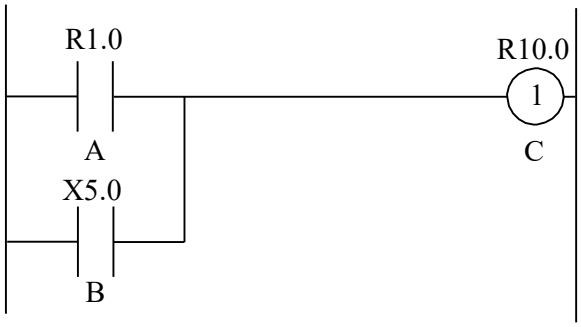


Function Logic OR the result of logic operation (ST0) to the specified address, which then is output to the same address.

Parameter Register point parameter

Example

Ladder Diagram

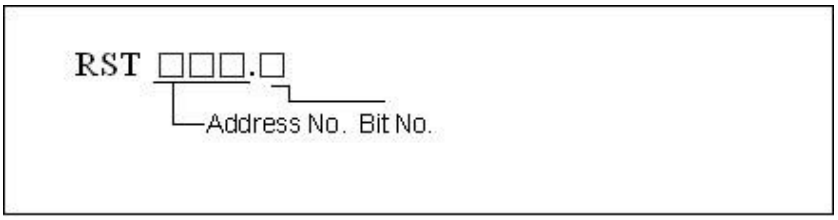


Operation Process

No.	Instruction	Address	Bit No.	Reference	ST2	ST1	ST0
1	LD	R1 .	0	A			A
2	OR	X5 .	0	B			$A+B$
3	SET	R10 .	0	C			$A+B$

2.6 RST

Format

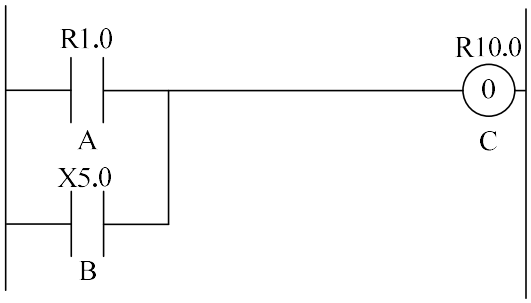


Function Logic AND the inverted result of logic operation (ST0) to the specified address, which then is output to the same address.

Parameter Register point parameter

Example

Ladder Diagram



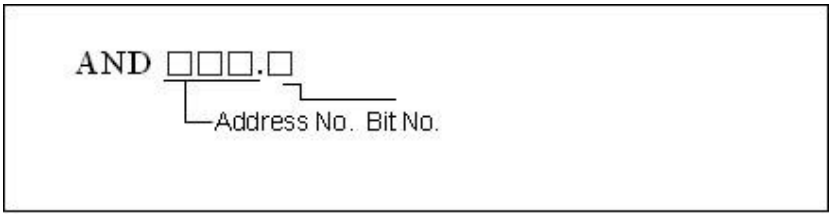
Operation Process

No.	Instruction	Address	Bit No.	Reference	ST2	ST1	ST0
1	LD	R1 . 0		A			A
2	OR	X5 . 0		B			$A+B$
3	RST	R10 . 0		C			$A+B$

Descriptio

2.7 AND

Format



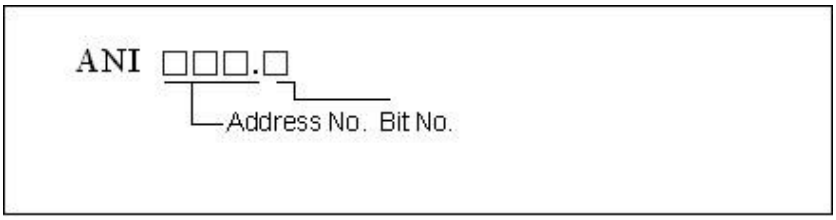
Function Logic AND

Parameter Register point parameter

Example See the example for LD instruction

2.8 ANI

Format



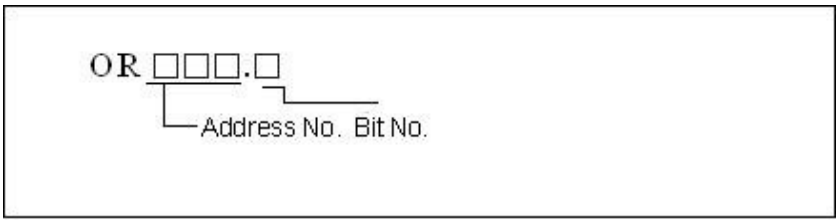
Function Logic AND NOT

Parameter Register point parameter

Example See the example for LD instruction

2.9 OR

Format



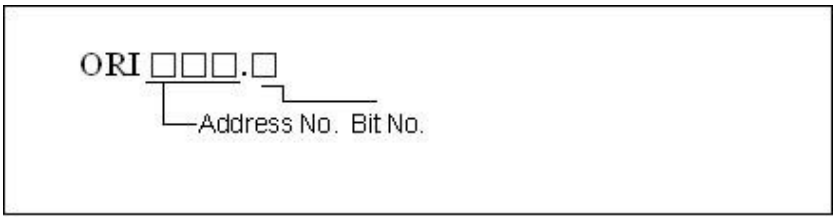
Function Logic OR

Parameter Register point parameter

Example See the example for LDI instruction

2.10 ORI

Format



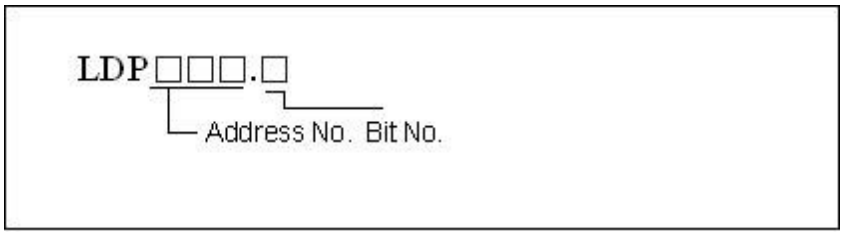
Function Logic OR NOT

Parameter Register point parameter

Example See the example for LDI instruction

2.11 LDP

Format



Function Get rising edge of trigger element signal, and save the signal in ST0.

Set input signal to 1 in the next scanning period of the rising edge of input signal.

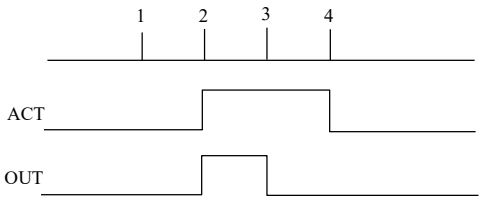
It is used for the situation in which programming starts from elements of rising edge.

Parameter Register point parameter

Control condition Input signal: Set output signal to 1 at the rising edge of signal（0->1）.

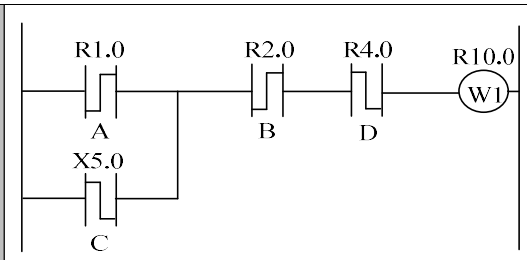
Output signal: During operation, input signal keeps 1 within one PLC scanning period.

Operation



Example

Ladder Diagram

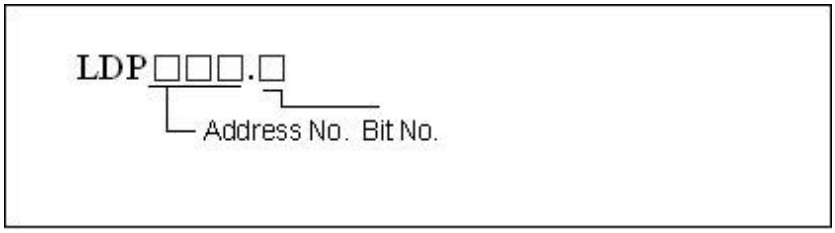


Operation Process

No.	Instruction	Address	Bit No.	Reference
1	LDP	R1	0	Rising edge of A
2	ORF	X5	0	Falling edge of B
3	ANDP	R2	0	Rising edge of C
4	ANDF	R4	0	Falling edge of D
5	OUT	R10	1	Output W1

2.12 LDF

Format



Function Get falling edge of trigger element signal, and save the signal in ST0.

Set input signal to 1 in the scanning period of the falling edge of input signal.

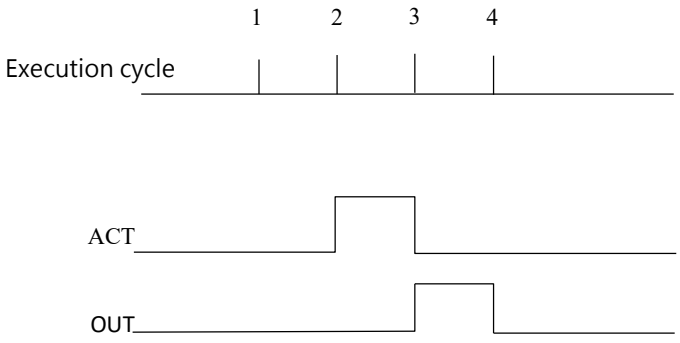
It is used for the situation in which programming starts from elements of falling edge.

Parameter Register point parameter

Control condition Input signal: Set output signal to 1 at the falling edge of signal（1->0）.

Output signal: During operation, input signal keeps 1 within one PLC scanning period.

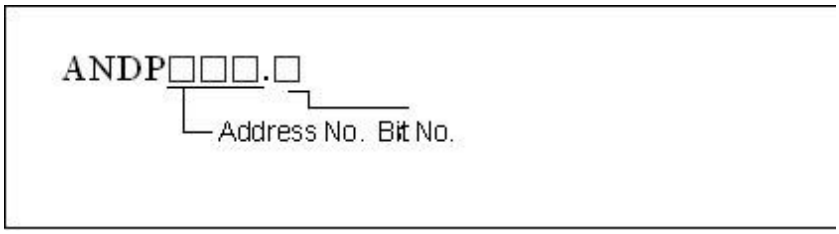
Operation



Example See the example for LDP instruction

2.13 ANDP

Format



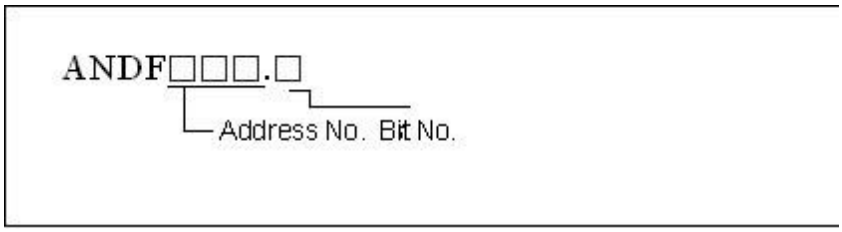
Function Logic AND rising edge

Parameter Register point parameter

Example See the example for instruction LDP

2.14 ANDF

Format



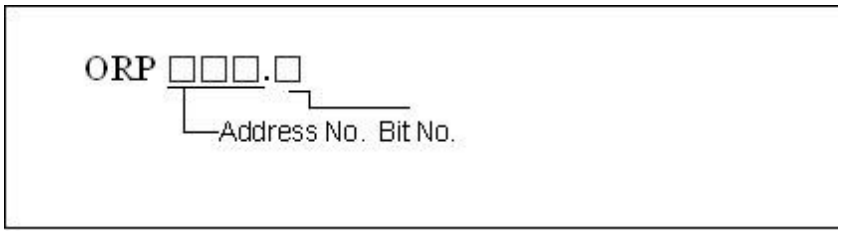
Function Logic AND falling edge

Parameter Register point parameter

Example See the example for instruction LDP

2.15 ORP

Format



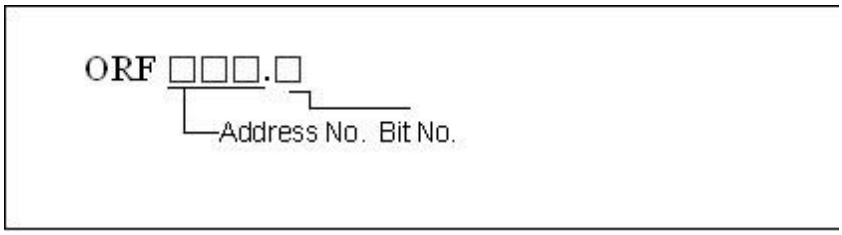
Function Logic OR rising edge

Parameter Register point parameter

Example See the example for instruction LDP

2.16 ORF

Format



Function Logic OR falling edge

Parameter Register point parameter

Example See the example for instruction LDP

2.17 ORB

Format

ORB

Function

1) ORB is independent, and doesn't need to connect to other elements or functional blocks.

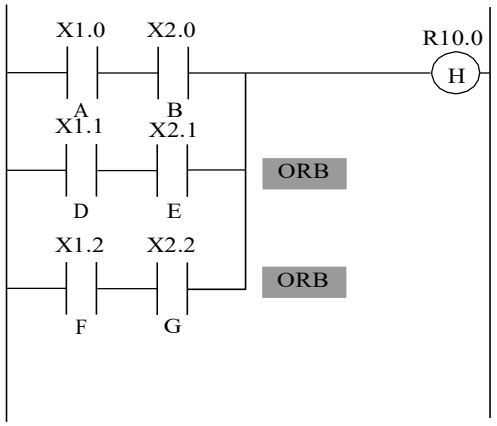
2) ORB is to connect two or more series circuits that contain more than one series block or contain the series ANB blocks.

3) Start the programming with LD or LDI, and have all series blocks being in parallel via ORB.

Parameter No parameter

Example

Ladder Diagram



Operation Process

No.	Instruction	Address	Bit No.	Reference	ST2	ST1	ST0
1	LD	X1 .	0	A			A
2	AND	X2 .	0	B			$A.B$
3	LD	X1 .	1	D		$A.B$	D
4	AND	X2 .	1	E		$A.B$	$D.E$
5	ORB						$A.B+D.E$
6	LD	X1.	2	F		$A.B+D.E$	F
7	AND	X2.	2	G		$A.B+D.E$	$F.G$
8	ORB						$A.B+D.E+F.G$
9	OUT	R10.	1	H			$A.B+D.E+F.G$

2.18 ANB

Format

ANB

Function

1) ANB is independent, and doesn't need to connect to other elements or functional blocks.

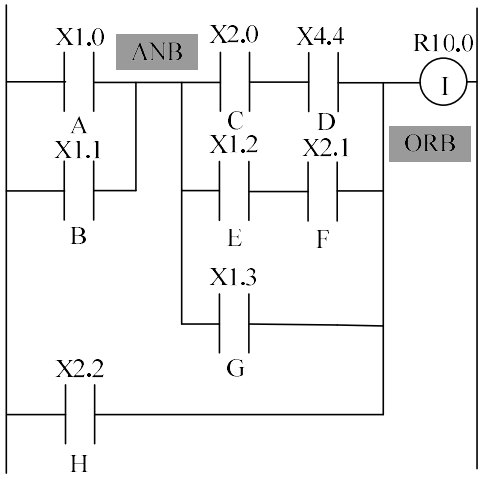
2) ANB is to connect two or more parallel circuits that contain more than one parallel-connected block or contain the parallel ORB blocks.

3) Starts programming with LD or LDI, and have all series blocks being in parallel via ANB.

Parameter No parameter

Example

Ladder Diagram



Operation

No.	Instruction	Address	Bit No.	Reference	ST2	ST1	ST0
1	LD	X1 . 0		A			*A.*

	2	OR	X1 . 1	B			$A+B$
	3	LD	X2 . 0	C		$A+B$	C
	4	AND	X4. 4	D		$A+B$	$C.D$
	5	LD	X1 . 2	E	$A+$	$C.D$	E
	6	AND	X2 . 1	F	$A+$	$C.D$	$E.F$
	7	ORB				$A+B$	$C.D+E.F$
	8	OR	X1. 3	G		$A+B$	$C.D+E.F+G$
	9	ANB					$(A+B).(C.D+E.F+G)$
	10	OR	X2. 2	H			$(A+B).(C.D+E.F+G)+H$
	11	OUT	R10. 0	I			$(A+B).(C.D+E.F+G)+H$
Description							

2.19 MPS, MRD, MPP

Format

MPS

MRD

MPP

Function

1) MPS Stores signal states of this point, waiting to be used when other lines are output.

2) MRD reads signal from last storage point, connects to the next node, of which signal status is always the same.

3) MPP brings up signal status from this storage point, connects to the next node, and removes the status of this node.

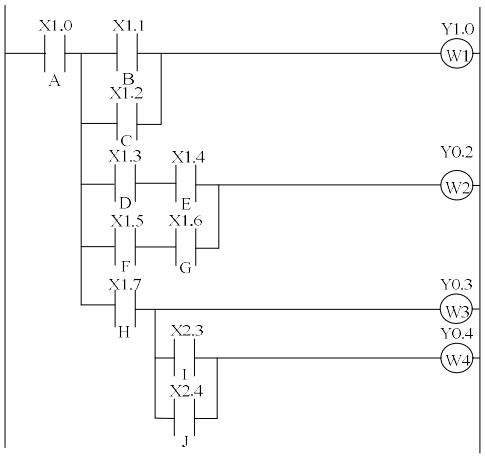
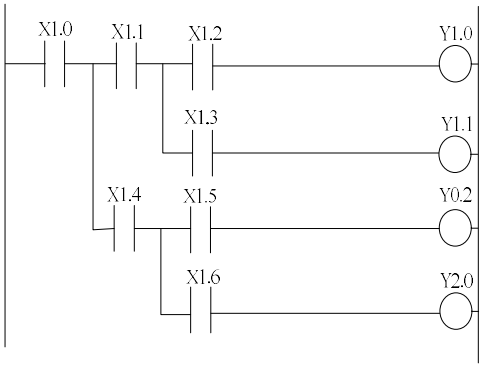
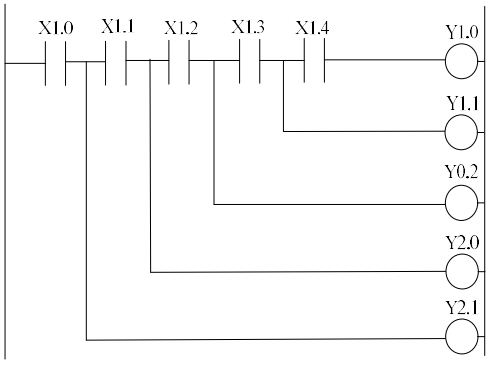
4) Every MPS must ends with MPP.

5) The last connection line must be ended with MPP.

Parameter

No parameter

Example

Ladder Diagram	Statement List	
	LD X1.0 MPS LD X1.1 OR X1.2 ANB OUT Y1.0 MRD LD X1.3 AND X1.4 LD X1.5 (followed by the right)	AND X1.6 ORB ANB OUT Y0.2 MPP AND X1.7 OUT Y0.3 LD X2.3 OR X2.4 ANB OUT Y0.4
	LD X1.0 MPS AND X1.1 MPS AND X1.2 OUT Y1.0 MPP AND X1.3 (followed by the right)	OUT Y1.1 MPP AND X1.4 MPS AND X1.5 OUT Y0.2 MPP AND X1.6 OUT Y2.0
	LD X1.0 MPS AND X1.1 MPS AND X1.2 MPS AND X1.3 MPS AND X1.4 (followed by the right)	OUT Y1.0 MPP OUT Y1.1 MPP OUT Y0.2 MPP OUT Y2.0 MPP OUT Y2.1

3 Basic Element

This chapter includes the sections as following:

3.1 Normally-open Contact

3.2 Normally-closed Contact

3.3 True Contact

3.4 Rising Edge of Contact

3.5 Falling Edge of Contact

3.6 Logic Output

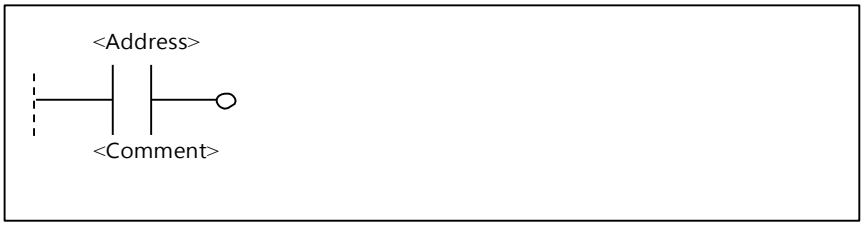
3.7 Inverted Logic Output

3.8 Setting Output

3.9 Reset Output

3.1 Normally-open Contact

Symbol

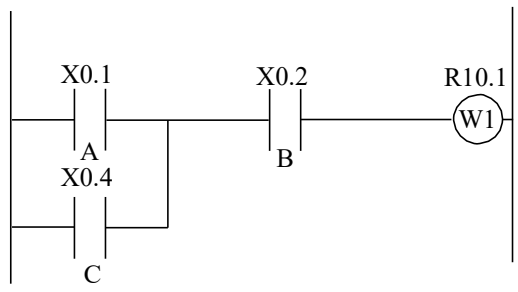


Parameter	Parameter form	Data type	Storage area	Explanation	Properties
<Address>	□□□□。□	BOOL	X, Y, F, G, R, W, D, P, T, C, B	Register bit to be checked	Pre ○ Post √

Function When the bit saved in the specified address is "1", the normally-open contact is closed; If the contact is closed, the signal will flow through this contact.

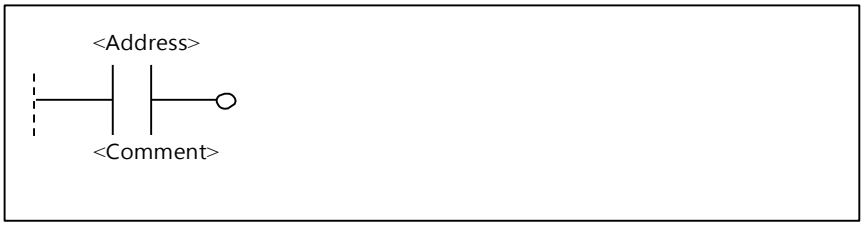
Parameter Parameter 1: register point parameter, in the form of X0.1.

Example

Ladder Diagram	
Description	When signal of X0.1 or X0.4 is "1", and X0.2 signal is "1", the current is conducted, with R10.1 being output.

3.2 Normally-closed Contact

Symbol

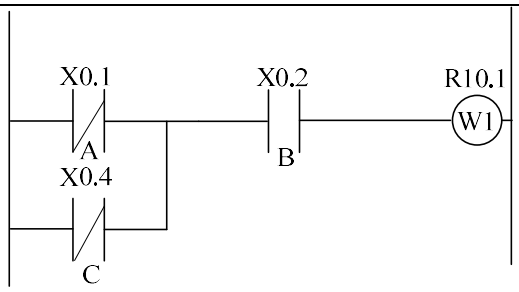


Parameter	Parameter form	Data type	Storage area	Explanation	Properties
<Address>	□□□□。□	BOOL	X, Y, F, G, R, W, D, P, T, C, B	Register bit to be checked	Pro ○ Post √

Function When the bit saved in the specified address is "0", the normally-closed contact is open; If the contact is open, the signal will flow through this contact.

Parameter Parameter 1: register point parameter, in the form of X0.1.

Example

Ladder Diagram	
Description	When signal of X0.1 or X0.4 is "0", and X0.2 signal is "1", the current is conducted, with R10.1 being output.

3.3 True Contact

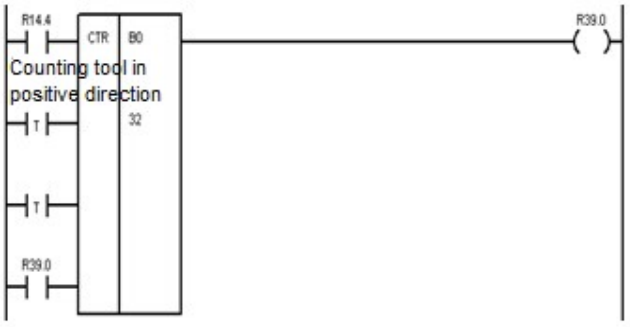
Symbol

Parameter	Parameter form	Data type	Storage area	Explanation	Properties
None	None	None	None	None	Pre ○
					Post √

Function When PLC is turned on, the signal on the left of an element can always reach the right through it. This function is usually used as the switch setting of functional module input, and used for those which need constantly valid input.

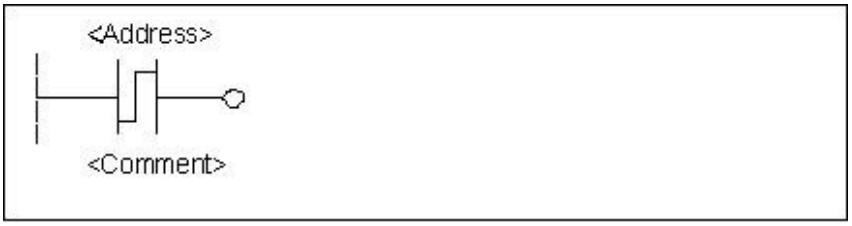
Parameter No parameter.

Example

Ladder Diagram	
Description	When the second input of counter uses true contact, the counting starts with 1 after counter is reset; when the third input uses true contact, the counter counts in continuous subtraction.

3.4 Rising edge of Contact

Symbol



Parameter	Parameter form	Data type	Storage area	Explanation	Properties
<Address>	□□□□。□	BOOL	X, Y, F, G, R, W, D, P, T、C, B	Contact of rising edge detection	Pre ○ Post √

Function When signal is changed from “0” to “1”, this contact is turned on.

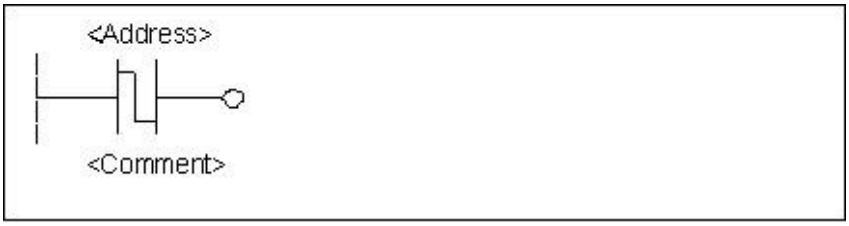
Parameter Parameter 1: register bit.

Example

Ladder Diagram	See the example for LDP instruction.
Description	

3.5 Falling Edge of Contact

Symbol



Parameter	Parameter form	Data type	Storage area	Explanation	Properties
<Address>	□□□□。□	BOOL	X, Y, F, G, R, W, D, P, T, C, B	Contact of falling edge detection	Pre ○ Post √

Function When signal is changed from "1" to "0", this contact is turned on.

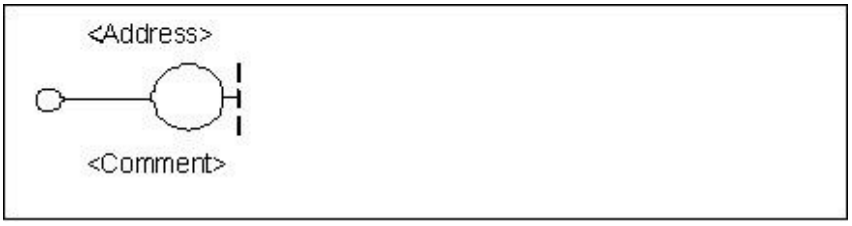
Parameter Parameter 1: register bit.

Example

Ladder	See the example for LDF instruction.
Description	

3.6 Logic Output

Symbol



Parameter	Form	Data type	Storage area	Explanation	Properties
<Address>	□□□□ ◦ □	BOOL	Y, G, R, W, D, B	Output coil	Pre ○
					Post ×

Function Output result of logical operation to output register.

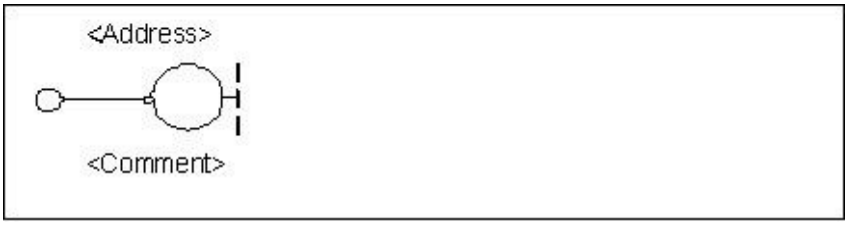
Parameter Parameter 1: register bit.

Example

Ladder	See the example for OUT instruction.
Description	

3.7 Inverted Logic Output

Symbol



Parameter	Parameter form	Data type	Storage area	Explanation	Properties
<Address>	□□□□ ◦ □	BOOL	Y, G, R, W, D, B	Inverted output coil	Pre ○
					Post ×

Function Output inverted result of logical operation to output register.

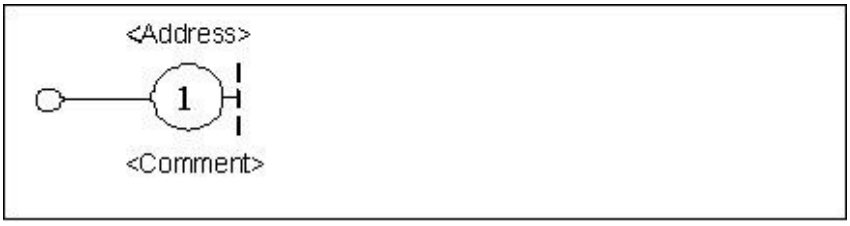
Parameter Parameter 1: register bit.

Example

Ladder	See the example for OOUT instruction.
Description	

3.8 Setting Output

Symbol



Parameter	Parameter form	Data type	Storage area	Explanation	Properties
<Address>	□□□□ ◦ □	BOOL	Y, G, R, W, D, B	Setting output coil	Pre ○
					Post ×

Function When result of logical operation is “1”, set output coil to output status, until this coil is reset by other functions.

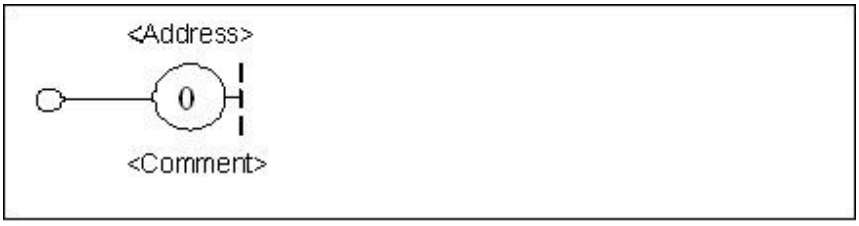
Parameter Parameter 1: register bit.

Example

Ladder	See the example for SET instruction.
Description	

3.9 Reset Output

Symbol



Parameter	Parameter form	Data type	Storage area	Explanation	Properties
<Address>	□□□□ ◦ □	BOOL	Y, G, R, W, D, B	Reset output coil	Pre ○
					Post ×

Function When result of logical operation is "1", reset output coil, until this coil is set by other functions.

Parameter Parameter 1: register bit.

Example

Ladder diagram	See the example for RST instruction.
Description	

4 Basic Function Module

This chapter includes:

4.1 Control Instruction

4.2 Mathematical Operation

4.3 Counter

4.4 Timer

4.5 Process Control

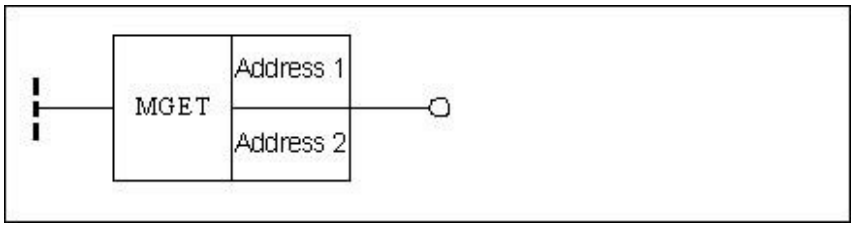
4.6 Comparison

4.7 Data Manipulation

4.1 Control Instruction

4.1.1 Instruction M Get MGET

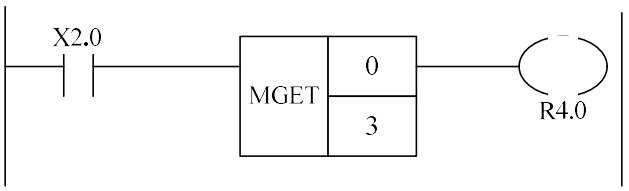
Format



Parameter	Parameter form	Data type	Storage area	Explanation	Properties
<Address 1>	□□□□	INT	Constant	Channel No.	Pre ○
<Address 2>	□□□□	INT	Constant	M code No.	Post √

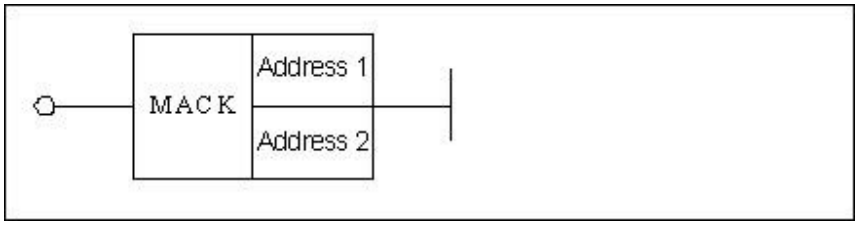
Function Through the channel selected by parameter 1, parameter 2 selects M code number which needs to be determined. When this channel gets this M code, the output is “1”; otherwise, the output is “0”.

Example

Ladder Diagram	
Statement	LD X2.0 MGET 0 3 WRT R4.0
Description	When M3 is executed in channel 0, R4.0 is set.

4.1.2 M Instruction Response MACK

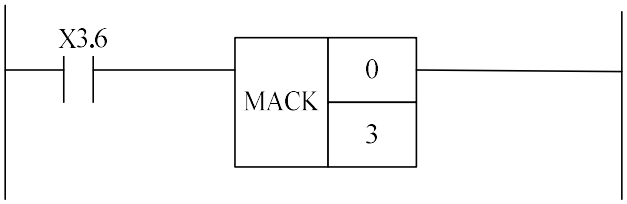
Format



Parameter	Parameter form	Data type	Storage area	Explanation	Properties
<Address 1>	□□□□	INT	Constant	Channel No.	Pre ○ Post ×
< Address 2>	□□□□	INT	Constant	M code No.	

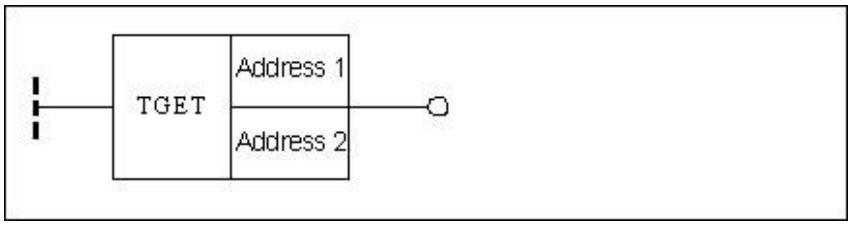
Function When M Code has been implemented in this channel, it is necessary to reply to M code. After the reply, this M instruction can continue the next instructions.

Example

Ladder Diagram	
Stateme	LD X3.6 MACK 0 3
Description	When X3.6 is effective, M3 in channel 0 is responded.

4.1.3 T Instruction Get TGET

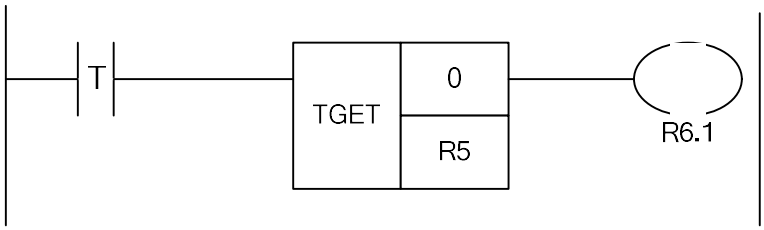
Format



Parameter	Parameter form	Data type	Storage area	Explanation	Properties
<Address 1>	□□□□	INT	Constant	Channel No.	Pre ○ Post √
<Address 2>	□□□□	INT	Constant, Y, G, R, W, D, B	T code No.	

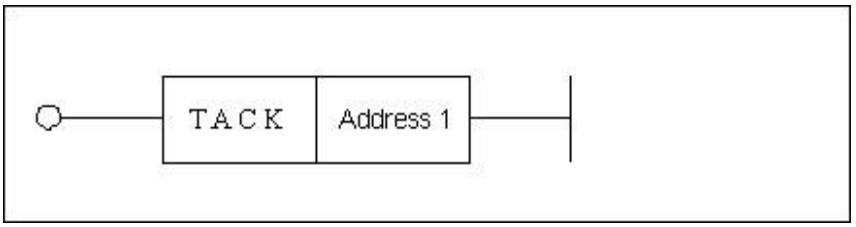
Function Through the channel selected by parameter 1, parameter 2 is where the gotten T code is stored in. When this channel gets T code, the output is 1; otherwise, the output is 0.

Example

Ladder Diagram	
Statement	LDT TGET 0 R5 OUT R6.1
Description	When channel 0 executes T instruction, T instruction parameter is sent to R5 register, and R6.1 is reset.

4.1.4 T Instruction Response TACK

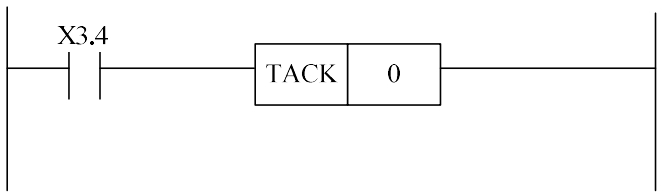
Format



Parameter	Parameter form	Data type	Storage area	Explanation	Properties
<Address 1>	□□□□	INT	Constant	Channel No.	Pre ○ Post ×

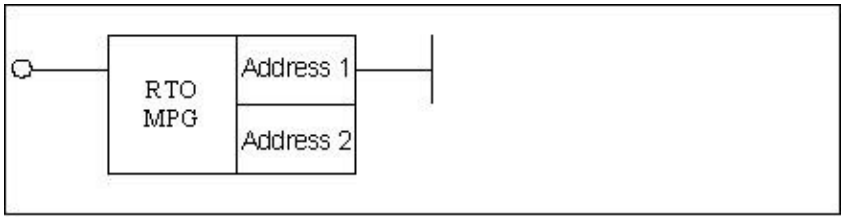
Function Through the channel selected by parameter 1, set to T code response in this channel.

Example

Ladder Diagram	
Stateme	LD X3.4 TACK 0
Description	When X3.4 is turned on, T instruction in this channel is replied.

4.1.5 Handwheel Control RTOMPG

Format



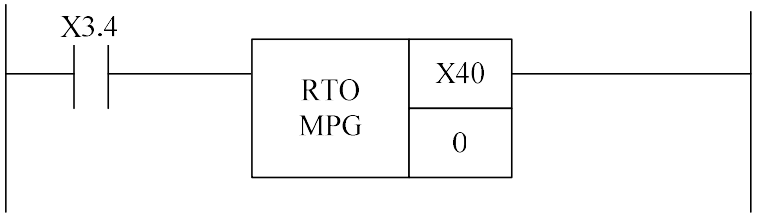
Parameter	Parameter form	Data type	Storage area	Explanation	Properties
<Address 1>	□□□□	INT	Constant, X, Y, F, G, R, W, D, P, B		Pre ○ Post ×
<Address 2>	□□□□	INT	Constant		

Function Handwheel control (only for series HNC8)

Parameter Parameter 1: the register for the handwheel pulse increment input. (The default register for handwheel of series HNC8 is X490)

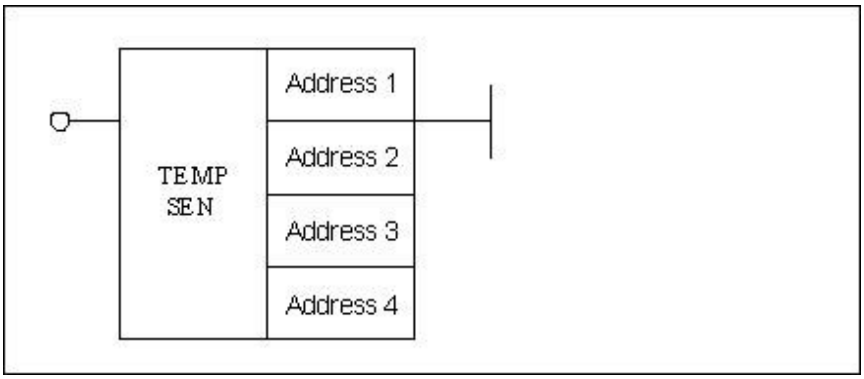
Parameter 2: MPG number, this parameter is for handwheel numbering. When there are more than one handwheels, they are distinguished by this parameter.

Example

Ladder Diagram	
Stateme	LD X3.4 RTOMPG X40 0
Descri	When X3.4 is turned on, enable the handwheel control. Handwheel pulse control register X40 is set into handwheel 0.

4.1.6 Thermal Error Compensation Module TEMPSEN

Format



Parameter	Parameter form	Data type	Storage area	Explanation	Properties
<Address 1>	□□□□	INT	Constant		Pre ○ Post ×
<Address 2>	□□□□. □	BOOL	X		
<Address 3>	□□□□	INT	Constant		
<Address 4>	□□□□. □	BOOL	P		

Function

Analog signal of temperature sensor is converted to digital signal by AD of IO module, and is input to a position (group number) of X register which is determined by IO module device parameter.

Parameter

Parameter 1: number of temperature sensor (number of temperature register). HNC8 CNC system is limited to input of 20 temperature acquisition signals. Therefore, the range of values for temperature sensor number is zero to nineteen.

Parameter 2: group number of X register corresponding to the digital signal of temperature acquisition.

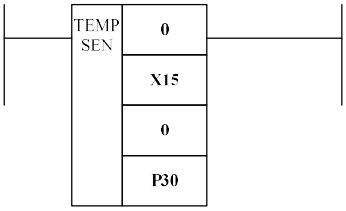
Parameter 3: thermocouple grid type (its default value is 0. 1: the corresponding model is built to calculate temperature by the user parameter specified by “parameter 4” which includes the lowest and highest (the temperature corresponding to the voltage of 6.7 V) temperatures; 2: temperature

sensor of PT100 is supported, and thermocouple grid of HIO-1075 is connected; 3: temperature sensor of KTY84-300 is supported, and thermocouple grid of HIO-1076 is connected; 4: the relationship between the measured temperature and the resistance calculated by the entered DA value is linear. The corresponding model is built to calculate temperature by the user parameter (P parameter) specified by "parameter 4" which includes the lowest and highest temperatures, as well as minimum and maximum resistances (unit: 0.01Ω).

Note: The thermocouple grid type of 2 and 3 are standard configurations, where the corresponding bus thermocouple grid can be connected, there are corresponding temperature models in system, and the value of P parameter doesn't need to be set.

Parameter 4: set the range of acquisition temperature for temperature sensor by user parameter (P parameter). As shown in the figure below, P30 specifies the acquisition of the lowest temperatures, and P31 specifies the acquisition of the highest temperatures (the temperature corresponding to the voltage 6.7 V, unit: degree). If the thermocouple grid type is 2 or 3, set value of P parameter will not be read.

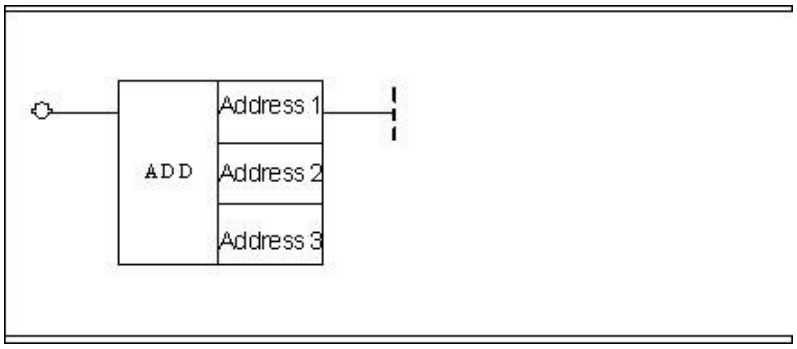
Example

Ladder Diagram	
Statement List	TEMPSEN 0 X15 0 P30
Description	The temperatures which have been gathered by No. 0 temperature sensor, is put into X15 register. P30 specifies the lowest acquisition temperatures.

4.2 Mathematical Operation

4.2.1 Addition ADD

Format



Parameter	Parameter form	Data type	Storage area	Explanation	Properties
<Address 1>	□□□□	INT	Constant, X, Y, F, G, R, W, D, P, B		Pre √ Post ○
< Address 2>	□□□□	INT	Constant, X, Y, F, G, R, W, D, P, B		
< Address 3>	□□□□	INT	Y, G, R, W, D, B		

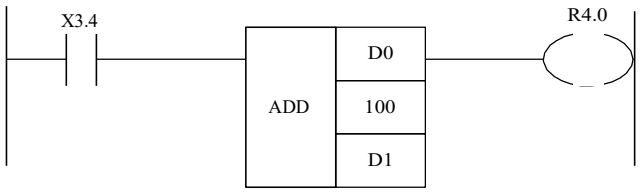
Function Perform addition operation.

Parameter Parameter 1: augend

Parameter 2: addend

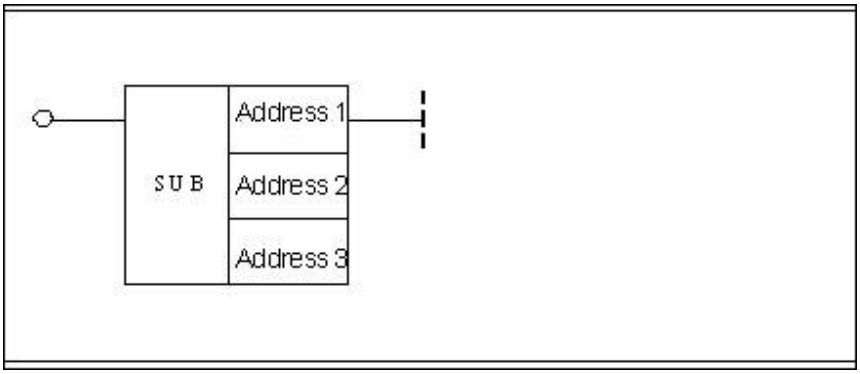
Parameter 3: operation result output address.

Example

Ladder Diagram	
Statement List	LD X3.4 ADD D0 100 D1 OUT R4.0
Description	When X3.4 is turned on, D1=D0+100 is implemented.

4.2.2 Subtraction SUB

Format



Parameter	Parameter form	Data type	Storage area	Explanation	Properties
<Address 1>	□□□□	INT	Constant, X, Y, F, G, R, W, D, P, B		Pre √ Post ○
<Address 2>	□□□□	INT	Constant, X, Y, F, G, R, W, D, P, B		
<Address 3>	□□□□	INT	Y, G, R, W, D, B		

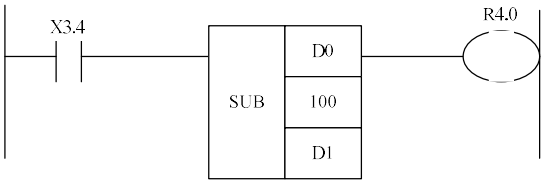
Function Perform subtraction operation.

Parameter Parameter 1: minuend

Parameter 2: subtrahend

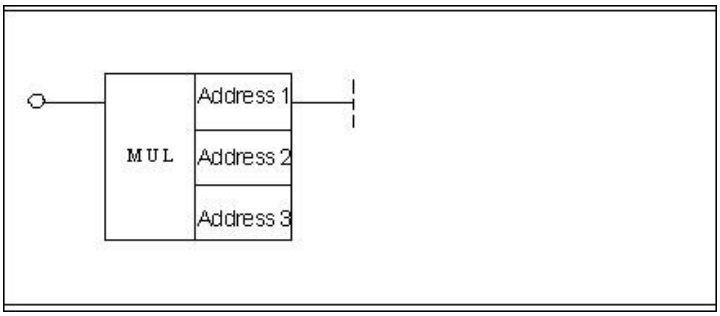
Parameter 3: operation result output address.

Example

Ladder Diagram	
Statement List	LD X3.4 SUB D0 100 D1 OUT R4.0
Description	When X3.4 is turned on, D1=D0-100 is implemented.

4.2.3 Multiplication MUL

Format



Parameter	Parameter form	Data type	Storage area	Properties
<Address 1>	□□□□	INT	Constant, X, Y, F, G, R, W, D, P, B	Pre √ Post ○
<Address 2>	□□□□	INT	Constant, X, Y, F, G, R, W, D, P, B	
<Address 3>	□□□□	INT	Y, G, R, W, D, B	

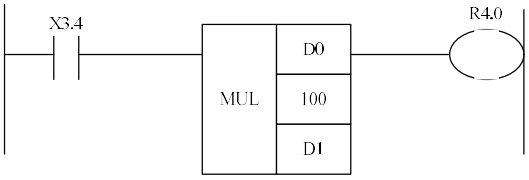
Function Perform multiplication operation.

Parameter Parameter 1: multiplicand

Parameter 2: multiplier

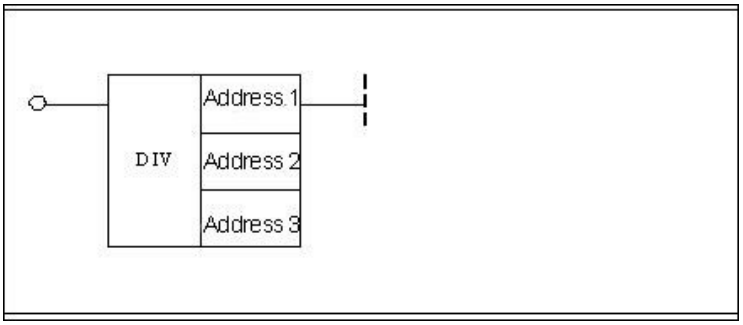
Parameter 3: operation result output address.

Example

Ladder Diagram	
Statement List	LD X3.4 MUL D0 100 D1 OUT R4.0
Description	When X3.4 is turned on, D1=D0*100 is implemented.

4.2.4 Division DIV

Format



Parameter	Parameter form	Data type	Storage area	Properties
<Address 1>	□□□□	INT	Constant, X, Y, F, G, R, W, D, P, B	Pre √ Post ○
<Address 2>	□□□□	INT	Constant, X, Y, F, G, R, W, D, P, B	
<Address 3>	□□□□	INT	Y, G, R, W, D, B	

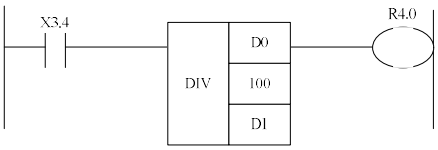
Function Perform division operation.

Parameter Parameter 1: dividend

Parameter 2: divisor, cannot be 0

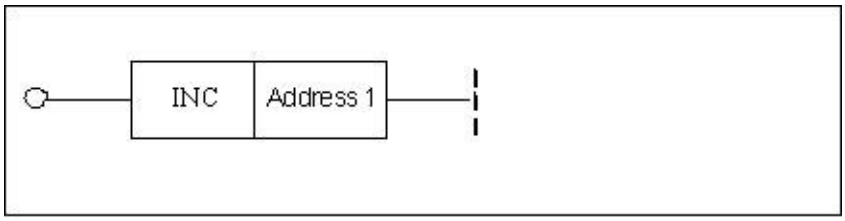
Parameter 3: operation result output address.

Example

Ladder Diagram	
Statement List	LD X3.4 DIV D0 100 D1 OUT R4.0
Description	When X3.4 is turned on, D1=D0/100 is implemented.

4.2.5 Increase One INC

Format

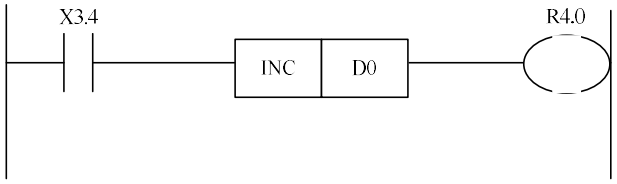


Parameter	Parameter form	Data type	Storage area	Properties
<Address 1>	□□□□	INT	Y, G, R, W, D, B	Pre √ Post ○

Function Perform plus-one operation.

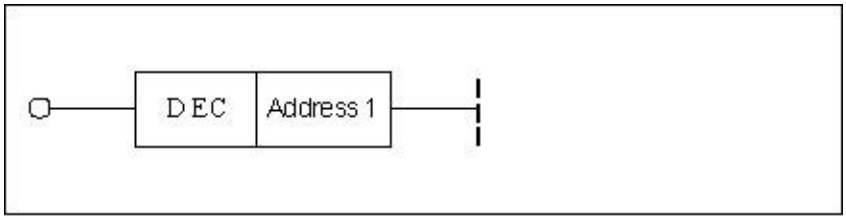
Parameter Parameter 1: operand.

Example

Ladder Diagram	
Statement List	LD X3.4 INC D0 OUT R4.0
Description	When X3.4 is turned on, D0=D0+1 is implemented.

4.2.6 Decrease One DEC

Format

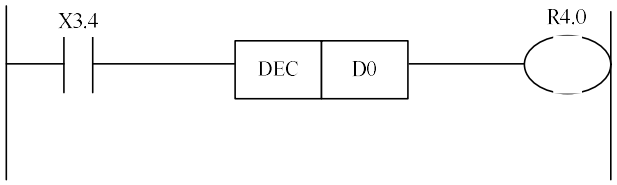


Parameter	Parameter form	Data type	Storage area	Properties
<Address 1>	□□□□	INT	Y, G, R, W, D, B	Pre √ Post ○

Function Perform minus-one operation.

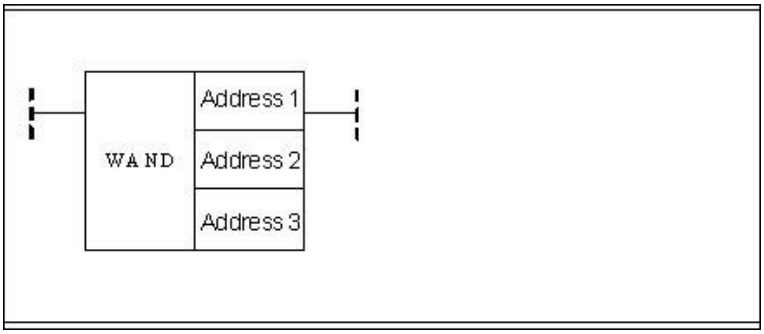
Parameter Parameter 1: operand.

Example

Ladder Diagram	
Statement List	LD X3.4 DEC D0 OUT R4.0
Description	When X3.4 is turned on, D0=D0-1 is implemented.

4.2.7 Logic AND WAND

Format



Parameter	Parameter form	Data type	Storage area	Properties
<Address 1>	□□□□	INT	Constant, X, Y, F, G, R, W, D, P, B	Pre √ Post ○
<Address 2>	□□□□	INT	Constant, X, Y, F, G, R, W, D, P, B	
<Address 3>	□□□□	INT	Y, G, R, W, D, B	

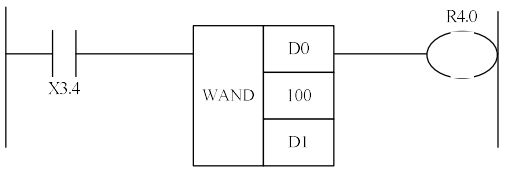
Function Perform logic AND.

Parameter Parameter 1: the number being operated.

Parameter 2: operand.

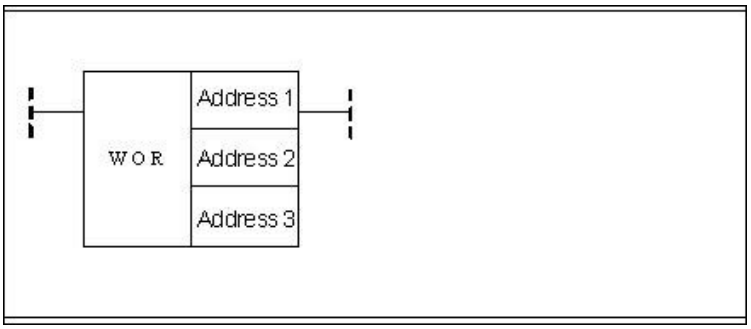
Parameter 3: operation result output address.

Example

Ladder Diagram	
Statement List	LD X3.4 WAND D0 100 D1 OUT R4.0
Description	When X3.4 is turned on, D0=D0&100 is implemented.

4.2.8 Logic OR WOR

Format



Parameter	Parameter form	Data type	Storage area	Properties
<Address 1>	□□□□	INT	Constant, X, Y, F, G, R, W, D, P, B	Pre √ Post ○
<Address 2>	□□□□	INT	Constant, X, Y, F, G, R, W, D, P, B	
<Address 3>	□□□□	INT	Y, G, R, W, D, B	

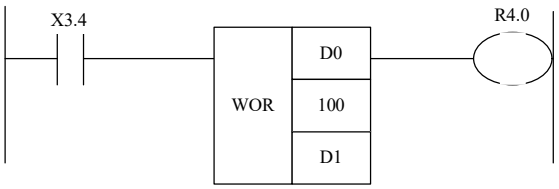
Function Perform logic OR

Function Parameter 1: the number being operated.

Parameter 2: operand.

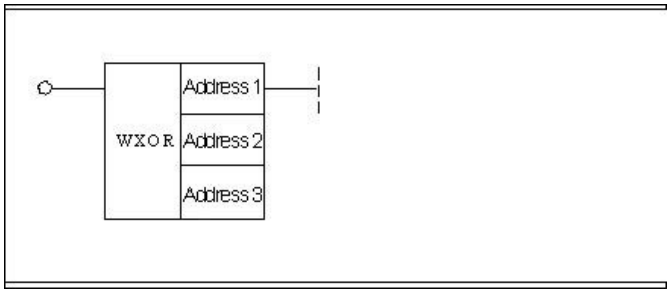
Parameter 3: operation result output address.

Example

Ladder Diagram	
Statement List	LD X3.4 WOR D0 100 D1 OUT R4.0
Description	When X3.4 is turned on, D0=D0|100 is implemented.

4.2.9 Logic XOR WXOR

Format



Parameter	Parameter form	Data type	Storage area	Properties
<Address 1>	□□□□	INT	Constant, X, Y, F, G, R, W, D, P, B	Pre √ Post ○
<Address 2>	□□□□	INT	Constant, X, Y, F, G, R, W, D, P, B	
<Address 3>	□□□□	INT	Y, G, R, W, D, B	

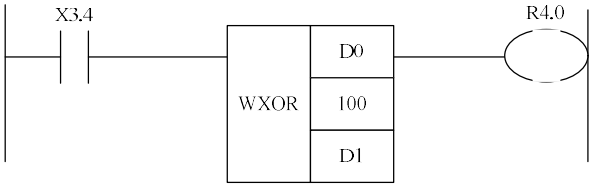
Function Perform logic XOR.

Parameter Parameter 1: the number being operated.

Parameter 2: operand

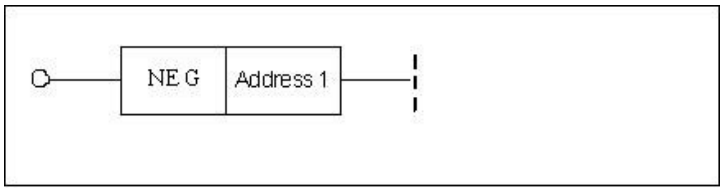
Parameter 3: operation result output address.

Example

Ladder Diagram	
Statement List	LD X3.4 WXOR D0 100 D1 OUT R4.0
Description	When X3.4 is turned on, D0=D0^100 is implemented.

4.2.10 Complement NEG

Format

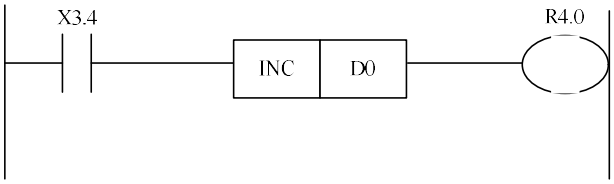


Parameter	Parameter form	Data type	Storage area	Properties
<Address 1>	□□□□	INT	Y, G, R, W, D, B	Pre √ Post ○

Function Perform complement operation.

Parameter Parameter 1: operand

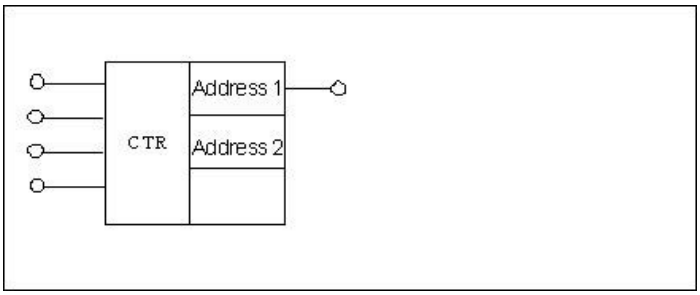
Example

Ladder Diagram	
Statement List	LD X3.4 NEG D0 OUT R4.0
Description	When X3.4 is turned on, D0=-D0 is implemented.

4.3 Counter

4.3.1 Up/Down Counter CTR

Format



Parameter	Parameter form	Data type	Storage area	Properties
<Address 1>	□□□□. □	BOOL	R, W, D, B	Pre √ Post √
<Address 2>	□□□□	INT	Constant, R, W, D, B, P	

Function Common up/down counter.

Parameter Parameter 1: current value of counter. This function is used to get the current value of the counter.

Parameter 2: preset value of counter.

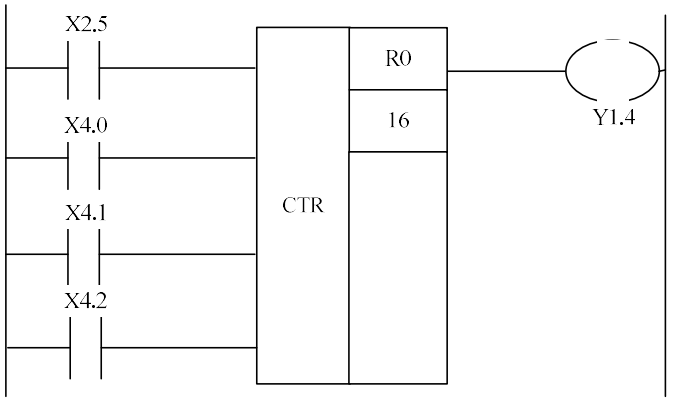
Input Input 1: control input

Input 2: start value after counter is reset. When the register is turned on, the counting starts with 1; when the register is not satisfied, the counting starts with 0.

Input 3: up/down input. When the register is turned on, count-down is performed; when the register is not turned on, count-up is performed.

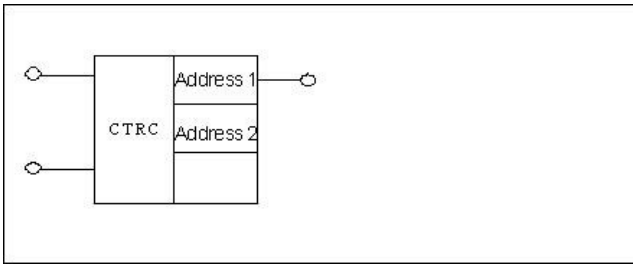
Input 4: reset input

Example

Ladder Diagram	
Statement List	LD X2.5 LD X4.0 LD X4.1 LD X4.2 CTR R0 16 OUT Y1.4
Description	If X4.0 is turned on, the counting will start with 1. If X4.1 is turned on, count-down is performed. When X2.5 has been turned on sixteen times, and output to Y1.4, X4.2 is the reset signal to clear the counter output. When X2.5 has been turned on five times, the value of R0 is 6

4.3.2 Counter CTRC

Format



Parameter	Parameter form	Data type	Storage area	Properties
<Address 1>	□□□□	INT	Constant	Pre √ Post √
<Address 2>	□□□□	INT	Constant, R, W, D, B, P	

Function Fixed counter

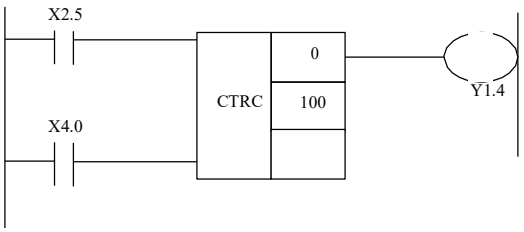
Parameter Parameter 1: counter number

Parameter 2: preset value of counter

Input Input 1: control input

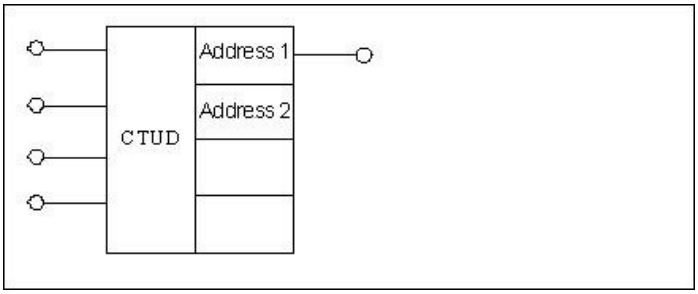
Input 2: reset input

Example

Ladder Diagram	
Statement List	LD X2.5 LD X4.0 CTRC 0 100 OUT Y1.4
Description	When X2.5 is switched on and then off 100 times, the counter is on. When X4.0 is switched on, the counter is reset, and the signal is output to Y1.4.

4.3.3 Custom Up/down Counter CTUD

Format



Parameter	Parameter form	Data type	Storage area	Properties
<Address 1>	□□□□	INT	Constant	Pre √ Post √
<Address 2>	□□□□	INT	Constant, R, W, D, B, P	

Function Up/down counter with custom starting value.

Parameter Parameter 1: counter number

Parameter 2: preset value of counter

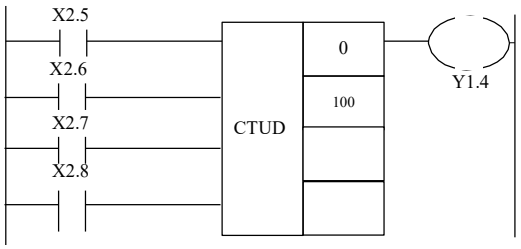
Input Input 1: control input

Input 2: start value after reset. When the register is turned on, the counting starts with 1; when the register is turned off, the counting starts with 0.

Input 3: Up/down input. When the register is turned on, count-down is performed; when the register is turned off, count-up is performed.

Input 4: Reset input

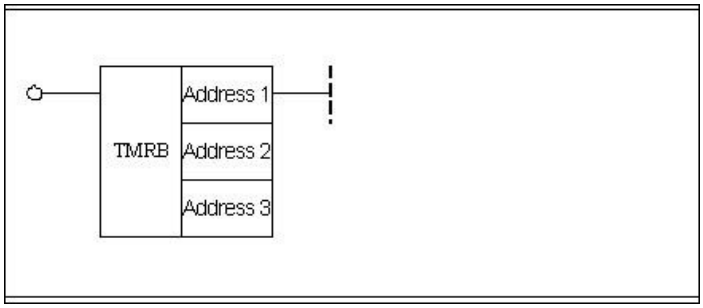
Example

Ladder Diagram	
Statement List	LD X2.5 LD X2.6 LD X2.7 LD X2.8 CTUD 0 100 OUT Y1.4
Description	When X2.5 is switched on and off 100 times, counter 0 is on, and the signal is output to Y1.4. When X2.6 is switched on, the counter starts count with 1 after being reset; otherwise, the counter starts count with 0. When X2.7 is switched off, the counter is incremented; otherwise, the counter is decremented. When X2.8 is switched on, the counter is reset.

4.4 Timer

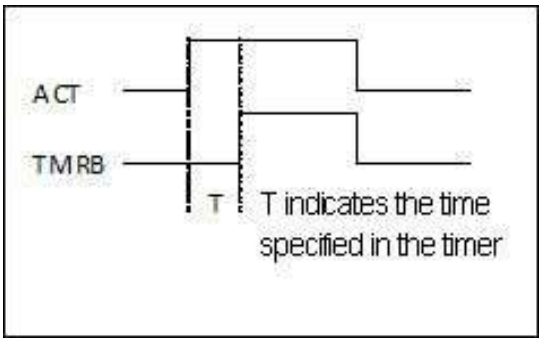
4.4.1 On-delay Timer TMRB

Format



Parameter	Parameter form	Data type	Storage area	Properties
<Address 1>	□□□□	INT	Constant	Pre √ Post ○
<Address 2>	□□□□	INT	Constant	
<Address 3>	□□□□	INT	Constant, R, W, D, P	

Sequence diagram



Function On-delay timer

Parameter Parameter 1: timer number

Parameter 2: time unit, the details are as following:

Time unit is hour, in the event of the value being 3;

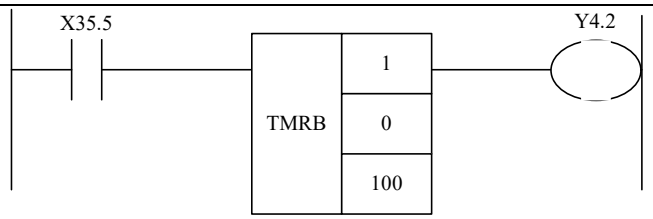
Time unit is minute, in the event of the value being 2;

Time unit is second, in the event of the value being 1;

Time unit is millisecond, in the event of the value being 0.

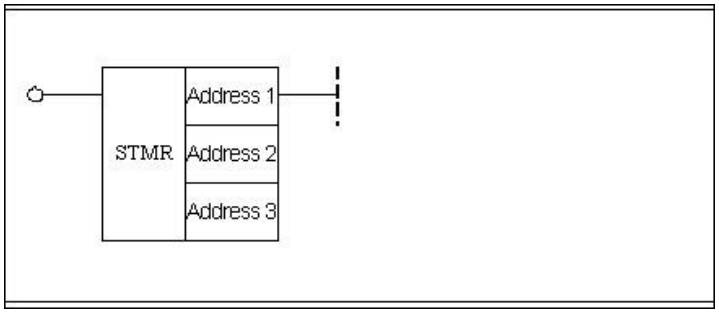
Parameter 3: Length of timing.

Example

Ladder Diagram	
Statement List	LD X35.5 TMRB 1 0 100 OUT Y4.2
Description	After X35.5 has been on for 100ms, timer 1 is switched on, and the signal is output to Y4.2.

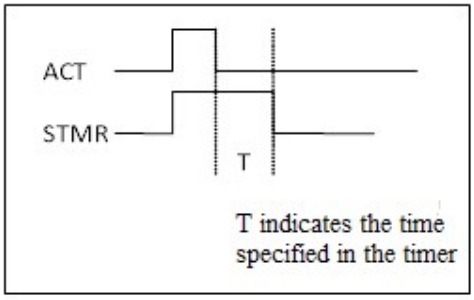
4.4.2 Off-delay Timer STMR

Format



Parameter	Parameter form	Data type	Storage area	Properties
<Address 1>	□□□□	INT	Constant	Pre √ Post ○
<Address2>	□□□□	INT	Constant	
<Address 3>	□□□□	INT	Constant, R, W, D, P	

Sequence diagram



Function Off-delay timer

Parameter

Parameter 1: timer number

Parameter 2: time unit, the details are as following:

Time unit is hour, in the event of the value being 3;

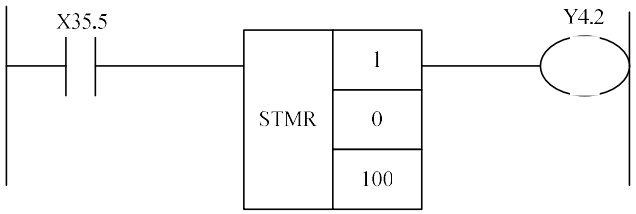
Time unit is minute, in the event of the value being 2;

Time unit is second, in the event of the value being 1;

Time unit is millisecond, in the event of the value being 0.

Parameter 3: Length of timing.

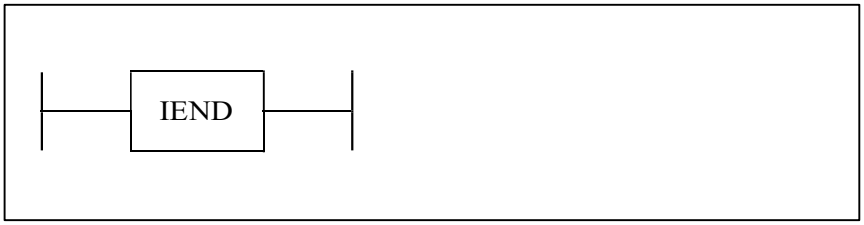
Example

Ladder Diagram	
Statement List	LD X35.5 STMR 1 0 100 OUT Y4.2
Descri	After X35.5 has been off for 100ms, timer 1 is off, and the output of Y4.2 is cut off.

4.5 Process Control

4.5.1 Initialization Module End IEND

Format



Parameter	Parameter form	Data type	Storage area	Properties
None	None	None	None	Pre × Post ×

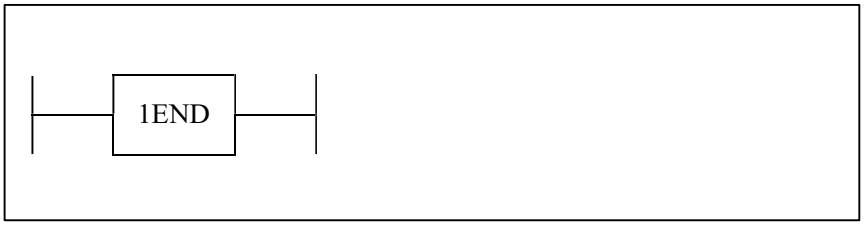
Function To define the end of the initialization module. Program generally is preceded by initialization module which is performed only once after the system is powered on.

Example

Ladder Diagram	
Statement List	IEND
Description	Initializer is ended.

4.5.2 PLC1 Module End 1END

Format



Parameter	Parameter form	Data type	Storage area	Properties
None	None	None	None	Pre × Post ×

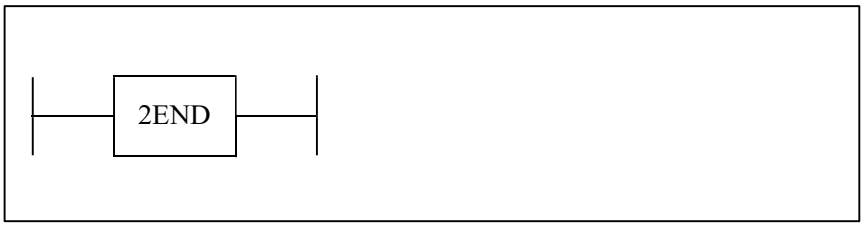
Function PLC1 module is ended.

Example

Ladder Diagram	
Statement List	1END
Description	PLC1 program is ended.

4.5.3 PLC2 Module End 2END

Format



Parameter	Parameter form	Data type	Storage area	Properties
None	None	None	None	Pre × Post ×

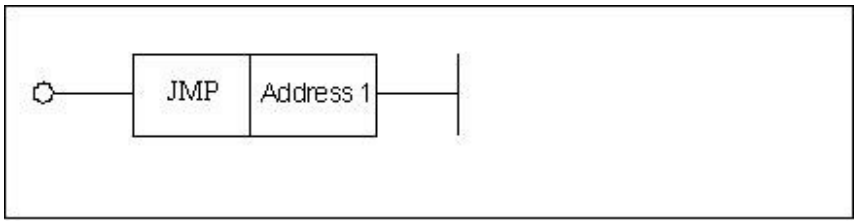
Function PLC2 module is finished.

Example

Ladder Diagram	
Statement List	2END
Description	PLC2 program is ended.

4.5.4 Jump JMP

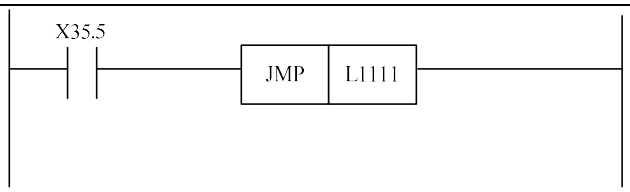
Format



Parameter	Parameter form	Data type	Storage area	Properties
<Address 1>	□□□□	INT	L	Pre √ Post ×

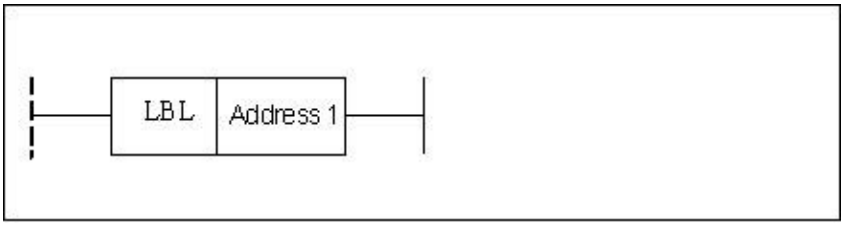
Function Follow the label to jump.

Example

Ladder Diagram	
Stateme	LD X35.5 JMP L1111
Descri	If x35.5 is on, go to the position labeled L1111 to continue the execution.

4.5.5 Label LBL

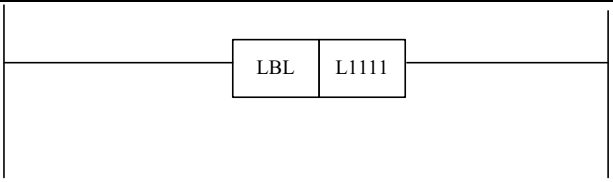
Format



Parameter	Parameter form	Data type	Storage area	Properties
<Address1>	□□□□	INT	L	Pre ○ Post ×

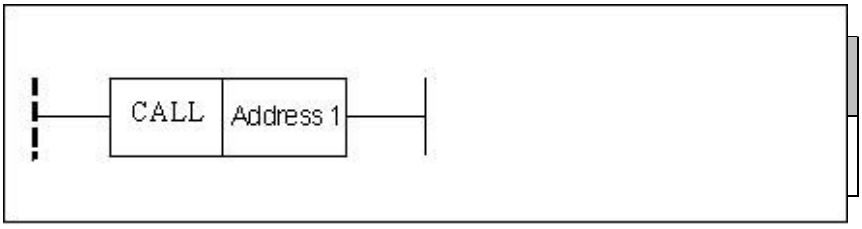
Function Label, follow the label to jump. It is used with JMP.

Example

Ladder Diagram	
Statement List	LBL L1111
Description	Set label L1111.

4.5.6 Call Subprogram CALL

Format

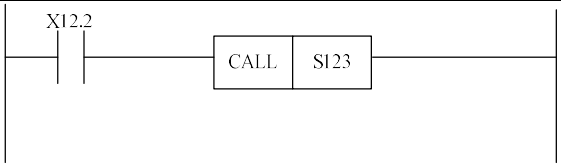


Parameter	Parameter form	Data type	Storage area	Properties
<Address 1>	□□□□	INT	S	Pre ○ Post ×

Function Call subprogram.

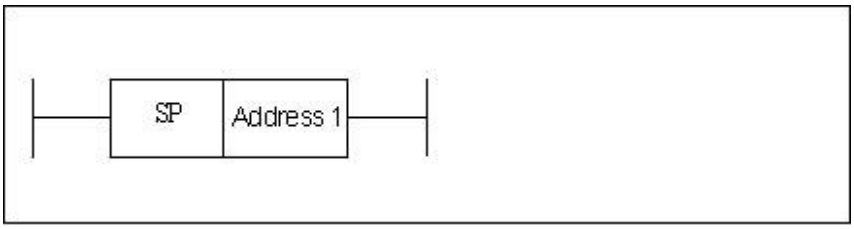
Parameter Subprogram number.

Example

Ladder Diagram	
Statement List	LD X12.2 CALL S123
Description	When X12.2 input is valid, jump to the subprogram of No. S123 to execute.

4.5.7 Subprogram Start SP

Format

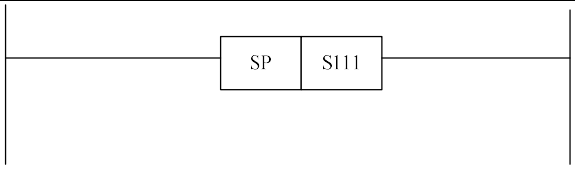


Parameter	Parameter form	Data type	Storage area	Properties
<Address 1>	□□□□	INT	S	Pre × Post ×

Function To start subprogram.

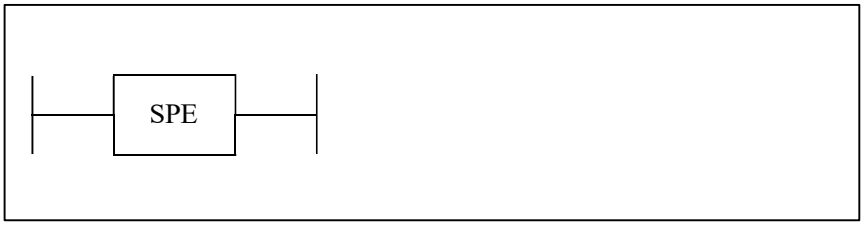
Parameter Number (up to 512 numbers of subprogram is supported).

Example

Ladder Diagram	
Statement List	SP S111
Description	Set subprogram number S111.

4.5.8 Subprogram End SPE

Format

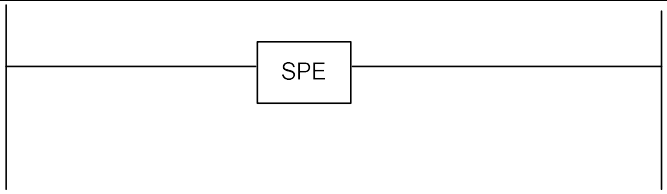


Parameter	Parameter form	Data type	Storage area	Properties
None	None	None	None	Pre × Post ×

Function To end Subprogram.

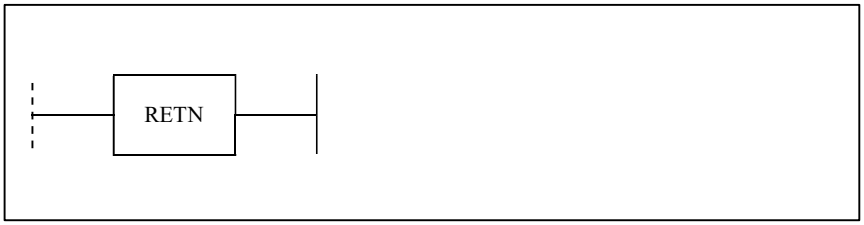
Parameter

Example

Ladder Diagram	
Statement List	SPE
Description	Subprogram is ended.

4.5.9 Subprogram Return RETN

Format

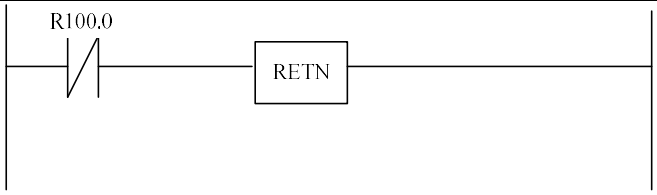


Parameter	Parameter form	Data type	Storage area	Properties
None	None	None	None	Pre ○ Post ×

Function Subprogram return. If this instruction is encountered in the subprogram, the execution will jump out of the subprogram, and continue the rest.

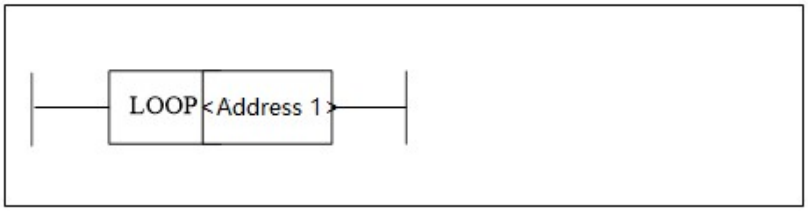
Parameter

Example

Ladder Diagram	
Statement List	LDI R100.0 RETN
Description	If normally-closed point R100.0 is valid, the subprogram will return.

4.5.10 Loop LOOP

Format

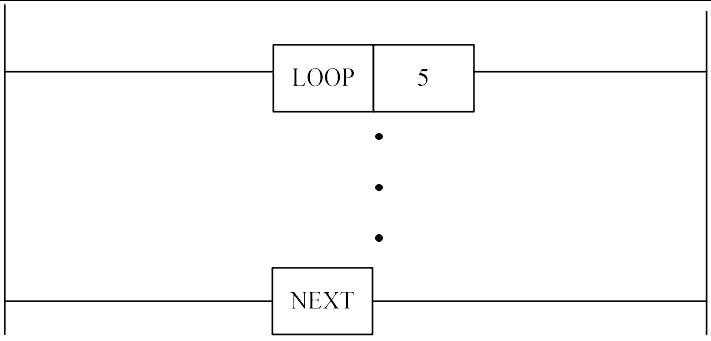


Parameter	Parameter form	Data type	Storage area	Properties
<Address 1>	□□□□	INT	Constant	Pre × Post ×

Function To start the loop. The statement within the body of each loop will be executed. When the number of loops is reached, the rest statement will be continued. This instruction must be used in conjunction with NEXT instruction. The statement between LOOP and NEXT is called the loop body.

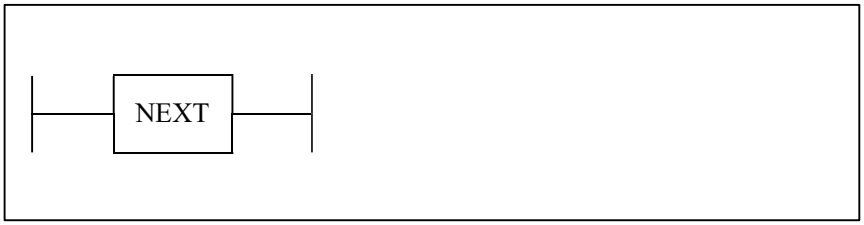
Parameter Number of loops, constant and register can be used.

Example

Ladder Diagram	
Statement List	LOOP 5 NEXT
Description	Loop 5 times

4.5.11 Next Loop NEXT

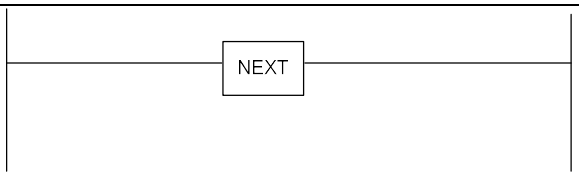
Format



Parameter	Parameter form	Data type	Storage area	Properties
None	None	None	None	Pre × Post ×

Function Enter the next loop.

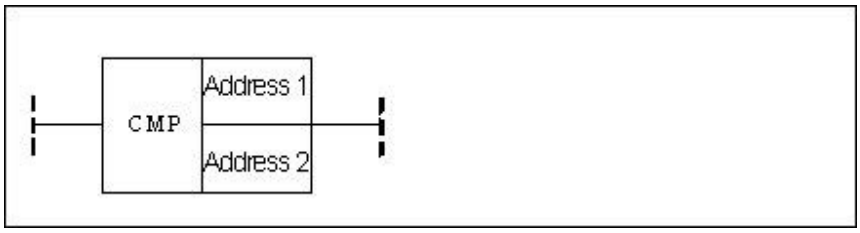
Example

Ladder Diagram	
Statement List	NEXT
Description	Enter the next loop. It is used with the instruction LOOP.

4.6 Comparison

4.6.1 Comparison CMP

Format

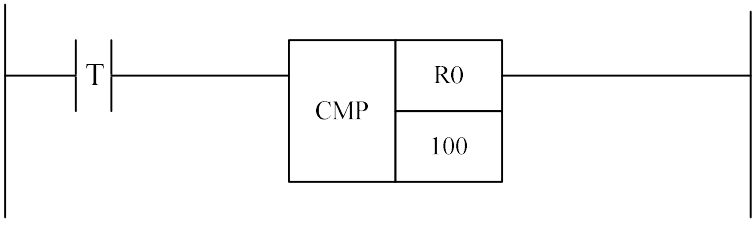


Parameter	Parameter form	Data type	Storage area	Explanation	Properties
<Address 1>	□□□□	INT	Constant, X, Y, F, G, R, W, D, P, B	When address 1 is larger than address 2, the output is 0, when address 1 is smaller than or equal to address 2, the output is 1.	Pre ○ Post √
<Address 2>	□□□□	INT	Constant, X, Y, F, G, R, W, D, P, B		

Function To compare. When the address 1 is larger than address 2, the output is 0, when the address 1 is lower than or equal to the address 2, the output is 1.

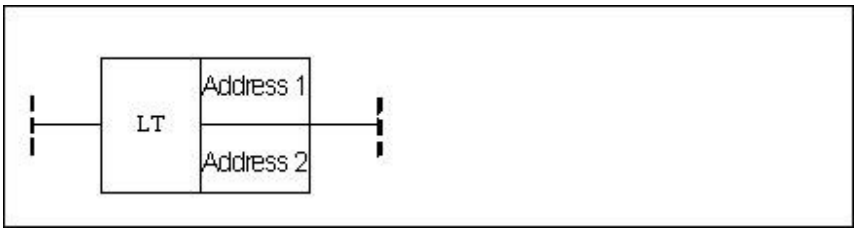
Parameter Parameter 1: comparing data, can be constant and register.
Parameter 2: data being compared, can be constant and register.

Example

Ladder Diagram	
Description	When R0<=100, the condition is satisfied.

4.6.2 Lower Than LT

Format



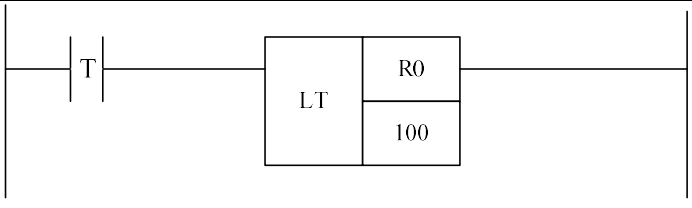
Parameter	Parameter form	Data type	Storage area	Explanation	Properties
<Address 1>	□□□□	INT	Constant, X, Y, F, G, R, W, D, P, B	When address 1 is larger than or equal to address 2, the output is 0, when address 1 is smaller than address 2, the output is 1.	Pre ○ Post √
<Address 2>	□□□□	INT	Constant, X, Y, F, G, R, W, D, P, B		

Function To compare. When the address 1 is larger than or equal to the address 2, the output is 0, when the address 1 is smaller than the address 2, the output is 1.

Parameter Parameter 1: comparing data, can be constant and register.

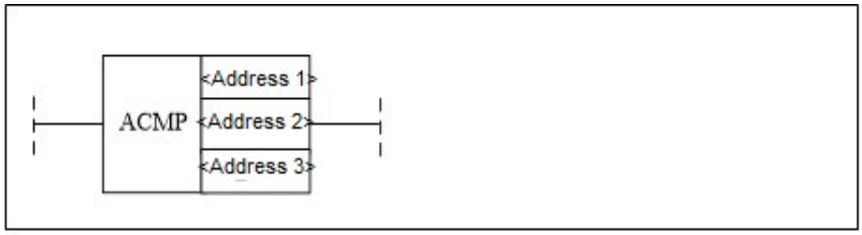
Parameter 2: data being compared, can be constant and register.

Example

Ladder Diagram	
Description	When R0<100, the register is turned on.

4.6.3 Area Comparison ACMP

Format



Parameter	Parameter form	Data type	Storage area	Explanation	Properties
<Address1>	□□□□	INT	Constant, X, Y, F, G, R, W, D, P, B	When data of address 3 is larger than that of address 1, and smaller than that of address2, the output is 1.	Pre ○ Post √
<Address 2>	□□□□	INT	Constant, X, Y, F, G, R, W, D, P, B		
<Address 3>	□□□□	INT	Constant, X, Y, F, G, R, W, D, P, B		

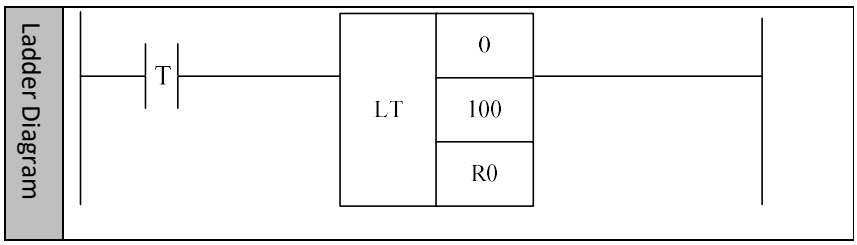
Function Area comparison. When data of address 3 is larger than that of address 1, and smaller than that of address2, the output is 1.

Parameter Parameter 1: the lower limit of comparison range, can be constant or register.

Parameter 2: the upper limit of comparison range, can be constant or register.

Parameter 3: Comparing data, can be constant or register.

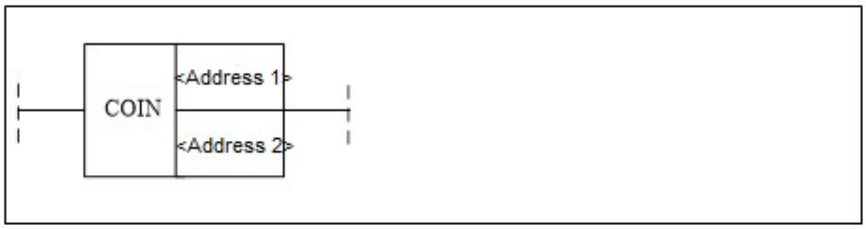
Example



Description	When R0<100 and R0>0, the register is turned on.

4.6.4 Consistency Comparison COIN

Format



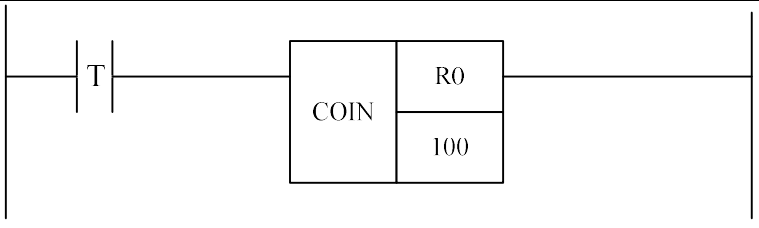
Parameter	Parameter form	Data type	Storage area	Explanation	Properties
<Address 1>	□□□□	INT	Constant, X, Y, F, G, R, W, D, P, B	When the data of address 1 and address 2 are the same, the output is 1; when they are not the same, the output is 0.	Pre ○ Post √
< Address 2>	□□□□	INT	Constant, X, Y, F, G, R, W, D, P, B		

Function Consistency comparison, When the data of address 1 and address 2 are the same, the output is 1; when they are not the same, the output is 0.

Parameter Parameter 1: benchmark data, can be constant and register.

Parameter 2: comparing data, can be constant and register.

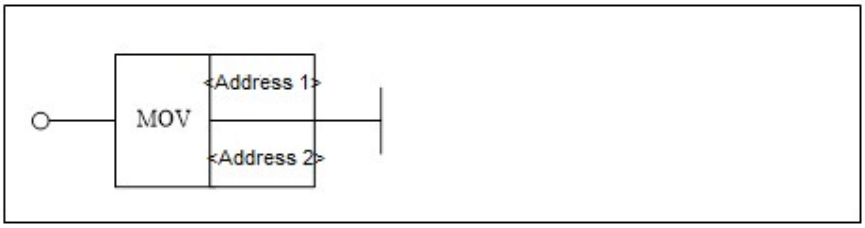
Example

Ladder Diagram	
Description	When R0=100, the register is turned on.

4.7 Data Manipulation

4.7.1 Moving Data MOV

Format



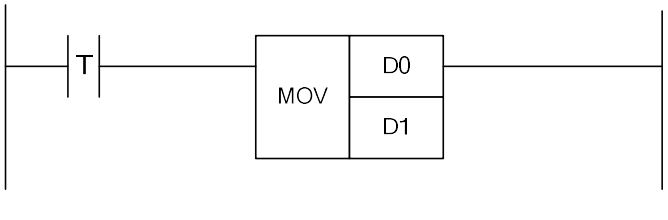
Parameter	Parameter form	Data type	Storage area	Explanation	Properties
<Address 1>	□□□□	INT	Constant, X, Y, F, G, R, W, D, P, B	Moving data	Pre ○
<Address 2>	□□□□	INT	Y, G, R, W, D, B		Post √

Function To move data. To transfer source data to destination address.

Parameter Parameter 1: source data, can be constant and register.

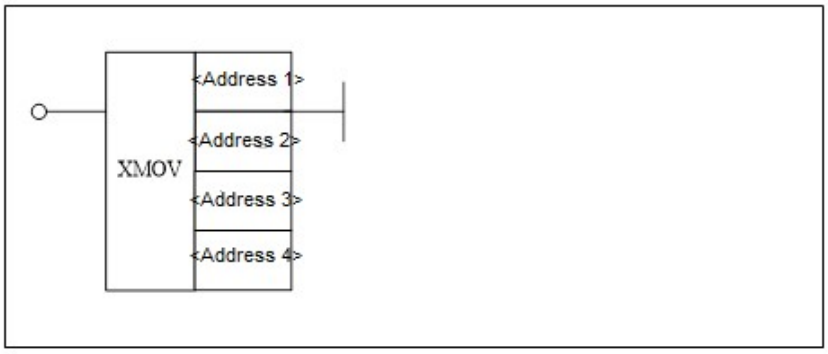
Parameter 2: destination data, can be register.

Example

Ladder Diagram	
Description	Data in D0 is assigned to D1.

4.7.2 Relative Moving Data XMOV

Format



Parameter	Parameter form	Data type	Storage area	Explanation	Properties
<Address 1>	□□□□	INT	Constant	Relative moving data	Pre ○ Post √
<Address 2>	□□□□	INT	G, R, W, D, B		
<Address 3>	□□□□	INT	Constant		
<Address 4>	□□□□	INT	G, R, W, D, B		

Function To move data. To transfer source data to the destination address.

Parameter

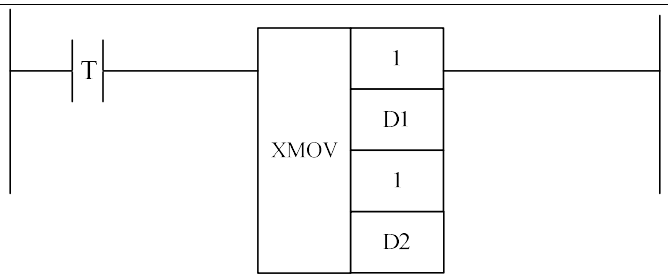
Parameter 1: the format of operand 1. 0 represents register, 1 represents register B, 2 represents register P. For example, the parameter 1 being 0 and the parameter 2 being R10 represents R10 address; the parameter 1 being 1 and the parameter 2 being R10 represents the register B, and the group number of register B is the data registered by R10; the parameter 1 being 2 and the parameter 2 being R10 represent the register P, and P register group number is the data stored in R10

Parameter 2: the address of operand 1.

Parameter 3: the address of operand 2.

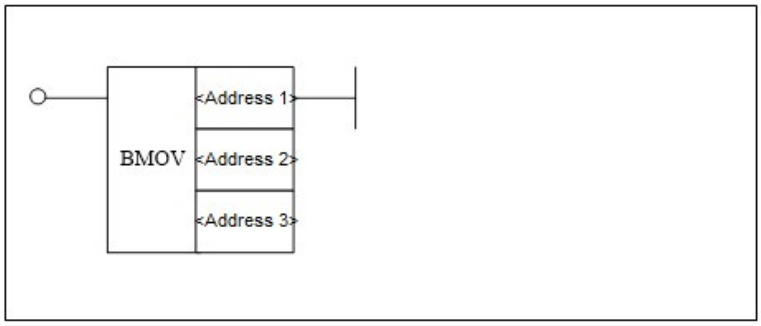
Parameter 4: the address of operand 2.

Example

Ladder Diagram	
Description	Assign the data shifted to D1 in register B to the position shifted to D2 in register B.

4.7.3 Batch Moving BMOV

Format



Parameter	Parameter form	Data type	Storage area	Explanation	Properties
<Address 1>	□□□□	INT	Constant, X, Y, F, G, R, W, D, P, B	Moving the data in batch.	Pre ○
<Address 2>	□□□□	INT	Constant, X, Y, F, G, R, W, D, P, B		
<Address 3>	□□□□	INT	Constant		Post √

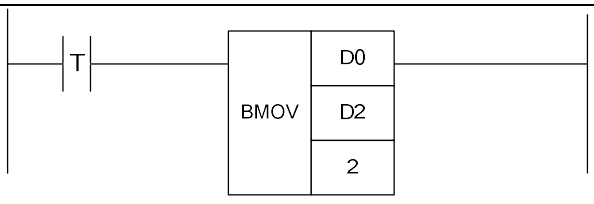
Function To move data in batch. Multiple data of which the addresses start with source is transferred to starting address of destination.

Parameter Parameter 1: Starting address of source data

Parameter 2: Starting address of destination

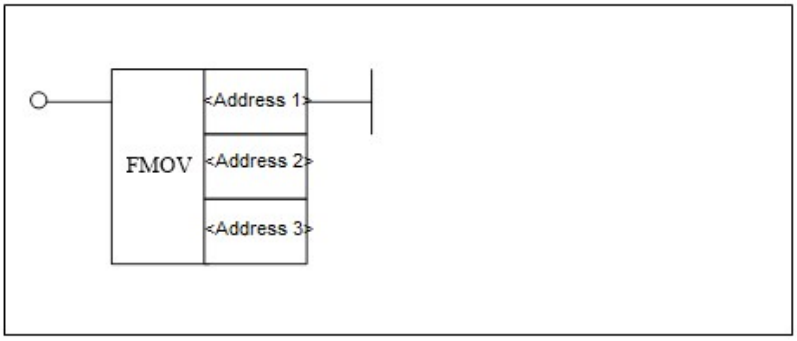
Parameter 3: The number of moves, can only be constant.

Example

Ladder Diagram	
Description	Two data starting from D0 is assigned to two positions starting from D2, that is, D0 is assigned to D2, and D1 is assigned to D3.

4.7.4 Multiple Moves FMOV

Format



Parameter	Parameter form	Data type	Storage area	Explanation	Properties
<Address 1>	□□□□	INT	Y, G, R, W, D, B	Multiple movement data.	Pre ○ Post ×
<Address 2>	□□□□	INT	Y, G, R, W, D, B		
<Address 3>	□□□□	INT	Constant, X, Y, F, G, R, W, D, P, B		

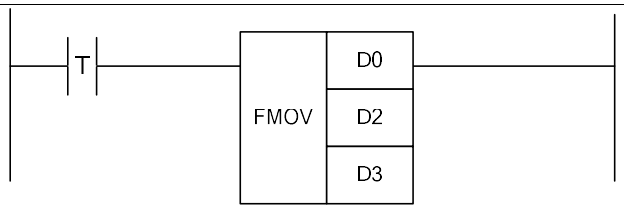
Function Multiple movement data. Source data is transferred to a space that is from the starting address of destination to ending address of destination.

Parameter Parameter 1: starting address of destination

Parameter 2: ending address of destination

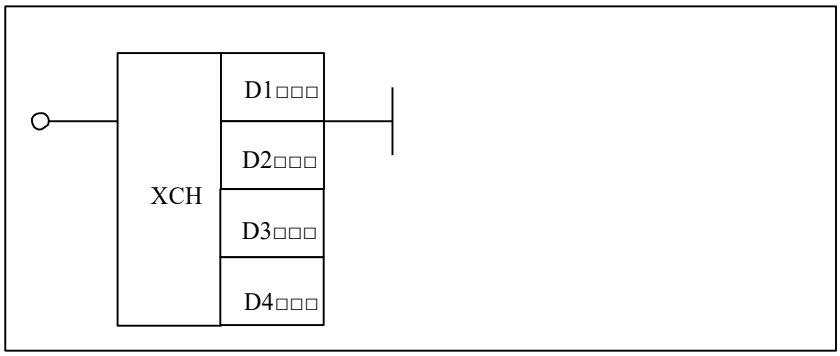
Parameter 3: source data

Example

Ladder Diagram	
Description	D3 data is assigned to the position that from D0 to D2.

4.7.5 Data Exchange XCH

Format

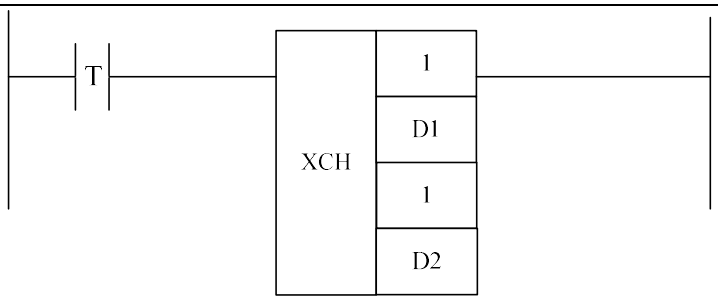


Parameter	Parameter form	Data type	Storage area	Explanation	Properties
<Address 1>	□□□□	INT	Constant	It is used to exchange data.	Pre ○
<Address 2>	□□□□	INT	G, R, W, D, B		
<Address 3>	□□□□	INT	Constant		
<Address 4>	□□□□	INT	G, R, W, D, B		Post ×

Function Data exchange. Address of operand 2 is exchanged with address of operand 4. The format of operand 2 can be represented by the value of address 1. 0 indicates the default register which is used in address 2, 1 indicates that B register is used in address 2. In the same way, the format of operand 4 can be represented by the value of address 3.

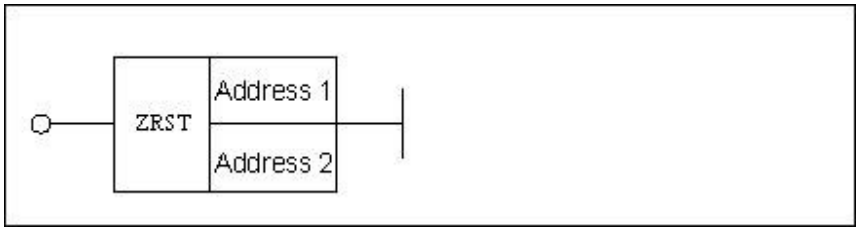
Parameter Parameter 1: the format of operand 1. 0 indicates register, 1 indicates B register, and 2 indicates P register. For example, parameter 1 is 0 and parameter 2 is R10, which represent the address is R10. Parameter 1 of 1, and parameter 2 of R10, represent B register, and B register group number is the data stored by R10. Parameter 1 of 2, and parameter 2 of R10, represent P register, and P register group number is the data stored by R10.
Parameter 2: address of operand 1
Parameter 3: format of operand 2
Parameter 4: address of operand 2

Example

Ladder Diagram	
Description	Exchange the data shifted to D1 with the data shifted to D2 in register B.

4.7.6 Data Reset ZRST

Format



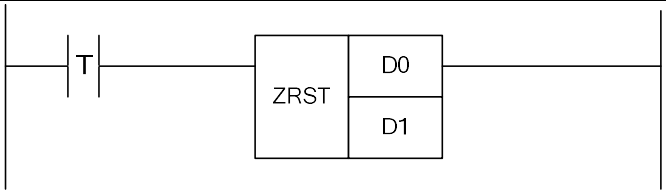
Parameter	Parameter form	Data type	Storage area	Explanation	Properties
<Address 1>	□□□.□	BOOL	Y, G, R, W, D, B	Data reset	Pre ○
<Address 2>	□□□.□	BOOL	Y, G, R, W, D, B		Post √

Function Data reset. Reset all the data from starting address of operand to ending address of operand.

Parameter Parameter 1: starting address of operand;

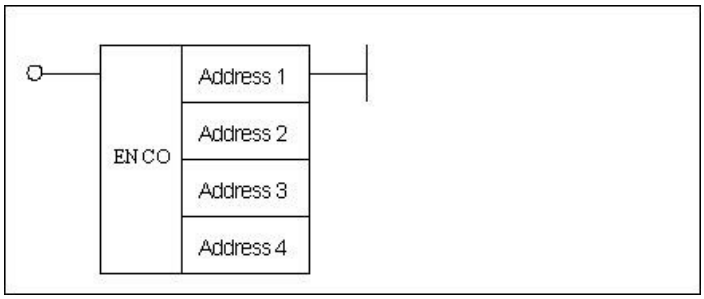
Parameter 2: ending address of operand.

Example

Ladder Diagram	
Description	0 is assigned to the position that from D0 to D1

4.7.7 Encoding ENCO

Format



Parameter	Parameter form	Data type	Storage area	Explanation	Properties
<Address 1>	□□□□. □	BOOL	X, Y, F, G, R, W, D, P, B	It is used for override value conversion	Pre ○
<Address 2>	□□□□	INT	Constant		
<Address 3>	□□□□. □	BOOL	X, Y, F, G, R, W, D, P, B		Post ×
<Address 4>	□□□□. □	BOOL	Y, G, R, W, D, P, B		

Function Coding. When there are 5 data bits (3, 5, 7, 8, 9) from the starting position of encoded data, if source data is 3, the output is 00000001B; if source data is 5, the output is 00000010B; if source data is 7, the output is 00000100B.

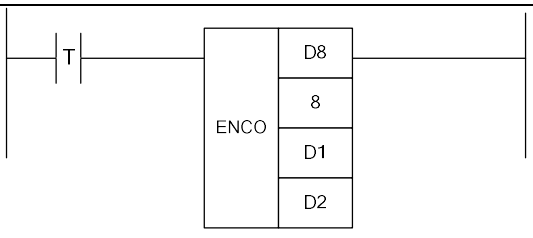
Parameter Parameter 1: the starting position of encoded data, can be register D.

Parameter 2: number of coded data, can be constant.

Parameter 3: source data, can be register R and D.

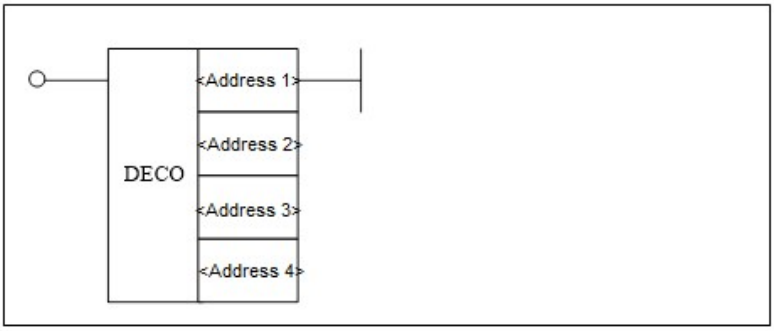
Parameter 4: Output address of destination data, can be register R and D.

Example

Ladder Diagram	
Description	After being encoded, data in D1 is output to D2.

4.7.8 Decoding DECO

Format



Parameter	Parameter form	Data type	Storage area	Explanation	Properties
<Address 1>	□ □ □ □ . □	BOOL	X, Y, F, G, R, W, D, P, B	It is used for override value conversion.	Pre√
<Address 2>	□□□□	INT	Constant		
<Address 3>	□ □ □ □ . □	BOOL	X, Y, F, G, R, W, D, P, B		
<Address 4>	□ □ □ □ . □	BOOL	Y, G, R, W, D, P, B		Post×

Function Decoding, which is reversed to encoding.

Parameter

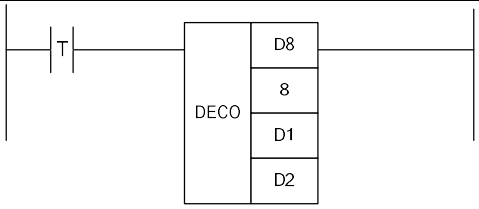
Parameter 1: the starting position of decoded data, which can be register D

Parameter 2: number of decoded data, can be constant.

Parameter 3: source data, can be register R and D

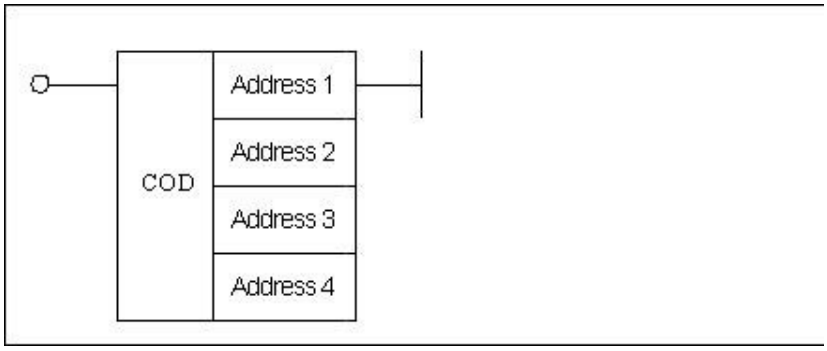
Parameter 4: Output address of destination data, can be register R and D.

Example

Ladder Diagram	
Description	After being decoded, the data in D1 is output to D2

4.7.9 Transformation COD

Format



Parameter	Parameter form	Data type	Storage area	Explanation	Properties
<Address 1>	□□□□. □	BOOL	X, Y, F, G, R, W, D, P, B	It is used for override value conversion	Pre √
<Address 2>	□□□□	INT	Constant		Post ×
<Address 3>	□□□□. □	BOOL	X, Y, F, G, R, W, D, P, B		
<Address 4>	□□□□. □	BOOL	Y, G, R, W, D, P, B		

Function Code transformation. It is mainly used for override value conversion. Take spindle override as an example, there are 8 data bits (50, 60, 70, 80, 90, 100, 110, 120) from D0; when source data is 0, the data transformed is 50; when source data is 1, the data transformed is 60; when source data is 2, the data transformed is 70.

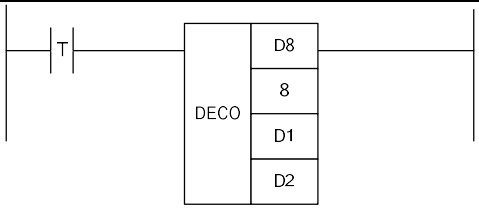
Parameter Parameter 1: the starting position for transforming data, can be register D.

Parameter 2: the number of data being transformed, which can be constant.

Parameter 3: source data, can be register R and D.

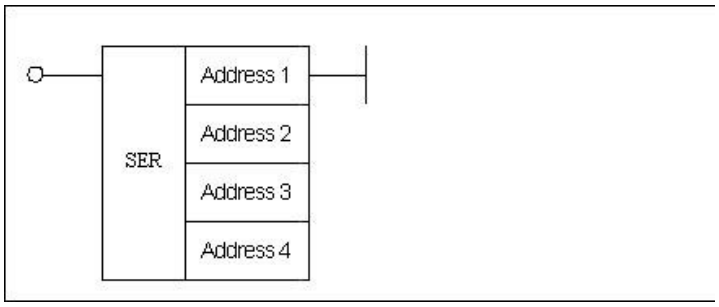
Parameter 4: output address of the target data, can be register R and D.

Example

Ladder Diagram	
Description	After being decoded, the data in D1 is output to D2.

4.7.10 Data Search SER

format



Parameter	Parameter form	Data type	Storage area	Explanation	Properties
<Address 1>	□□□□	INT	X, Y, F, G, R, W, D, P, B	When data is found, the output is 1; when data is not found, the output is 0.	Pre √
<Address 2>	□□□□	INT	Constant		
<Address 3>	□□□□	INT	X, Y, F, G, R, W, D, P, B		Post ×
<Address 4>	□□□□	INT	Y, G, R, W, D, P, B		

Function To search data. Search a data in a statement list. When the data is found, the output is 1; when the data is not found, the output is 0.

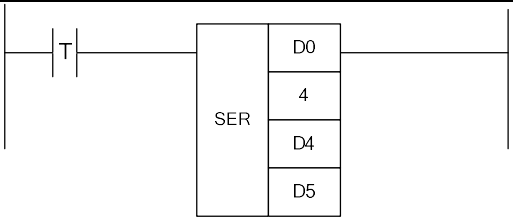
Parameter Parameter 1: searching address, can only be register D.

Parameter 2: searching range, can be constant.

Parameter 3: the data to be searched, can be constant and register X, Y, K, L, F, G, R, D.

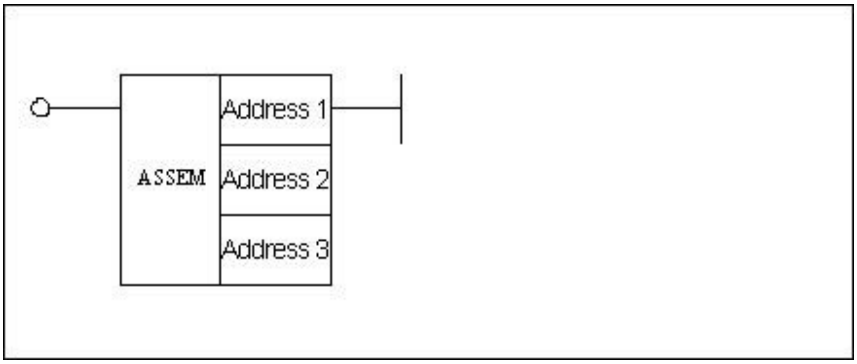
Parameter 4: the output address of searched result, can only be register D.

Example

Ladder Diagram	
Description	Search the data in D4 among the 4 data starting from D0, and output the position where the data is found to D5.

4.7.11 Register Merging ASSEM

Format



Parameter	Parameter form	Data type	Storage area	Explanation	Properties
<Address 1>	□□□□	INT	X, Y, F, G, R, W, D, P, B	To merge the data of several registers into one register.	Pre√
<Address 2>	□□□□	INT	Constant		
<Address 3>	□□□□	INT	G, W, D, B		Post×

Function To merge several register data into one register.

ddress

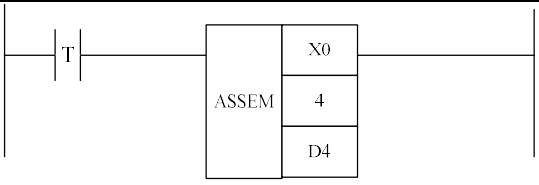
parameter

Parameter 1: source address.

Parameter 2: quantity of source registers, can only be constant.

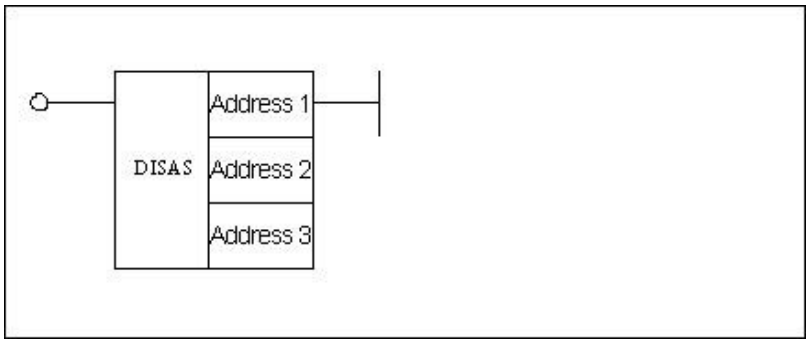
Parameter 3: target address, can be register G, W, D, B.

Example

Ladder Diagram	
Descri	Merge the four data starting from X0 into one data (i.e. 4 8-bit data are combined into one 32-bit data), and output to D4

4.7.12 Register Decomposition DISAS

Format



Parameter	Parameter form	Data type	Storage area	Explanation	Properties
<Address 1>	□□□□	INT	F, G, W, D, P, B	To break up the data of one register into several registers.	Pre√
<Address 2>	□□□□	INT	Constant		
<Address 3>	□□□□	INT	Y, G, R, W		Post×

Function To break up the data of one register into several registers.

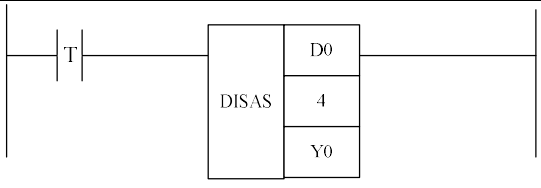
Parameter

Parameter 1: source address.

Parameter 2: number of source registers, can only be constant.

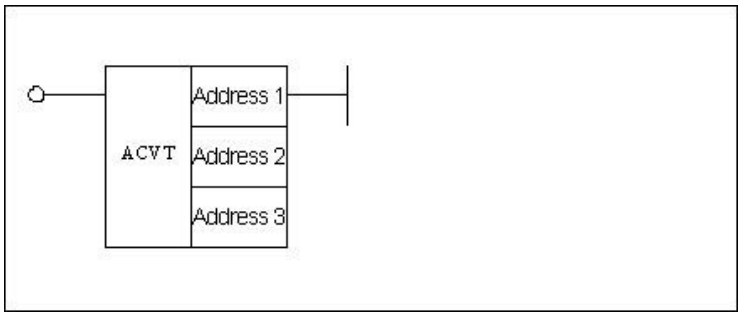
Parameter 3: target address, can be register Y, G, R, W.

Example

Ladder Diagram	
Description	Break up the data of D0 into the four data starting from Y0 (that is, one 32-bit data is broken up into 4 8-bit data).

4.7.13 Area Conversion ACVT

Format



Parameter	Parameter form	Data type	Storage area	Explanation	Properties
<Address 1>	□□□□	INT	P	Convert source data which follows a certain proportional relationship To target data.	Pre√
<Address 2>	□□□□	INT	X, Y, F, G, R, W, D, P, B		
<Address 3>	□□□□	INT	Y, G, R, W, D, B		Post×

Function Convert source data into the target data according to a certain ratio.

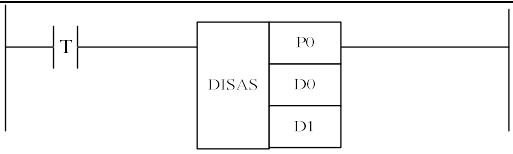
parameter Parameter 1: Address of proportional relationship.

0	Minimum value of source data
1	Maximum value of source data
2	Minimum value of target data
3	Maximum value of target data

Parameter 2: number of source registers.

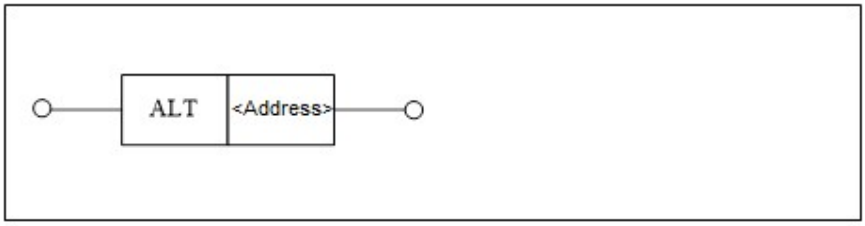
Parameter 3: target address, can be register Y, G, R, W, D, B ;

Example

Ladder Diagram	
Description	Convert D0 data which follows a certain proportional relationship to D1. D1=（D0-P0）*（P3-P2）/（P1-P0）+P0;

4.7.14 Alternate Output ALT

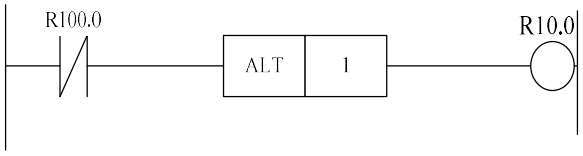
Format



Parameter	Parameter form	Data type	Storage area	Explanation	Properties
<Address 1>	□□□□	INT	Constant	Number	Pre √ Post √

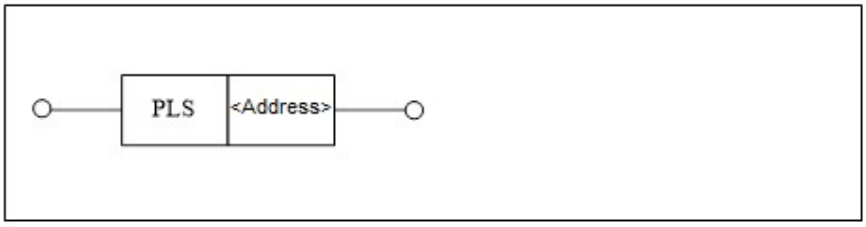
Function Alternate output. The component keeps its output status, until it encounters the rising edge, then the output status changes (change from 0 to 1, or 1 to 0).

Example

Ladder Diagram	
Description	When R100.0 is turned off from turned on, the module 1 changes the output status.

4.7.15 Fetch Rising Edge PLS

Format

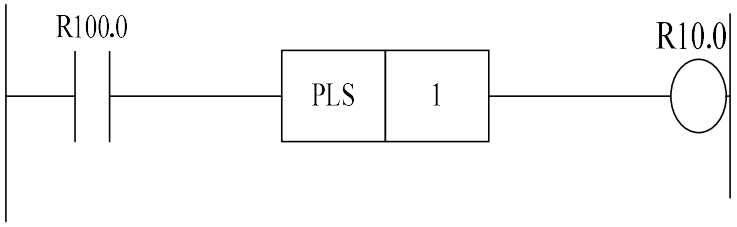


Parameter	Parameter form	Data type	Storage area	Explanation	Properties
<Address 1>	□□□□	INT	Constant	Rising edge module number	Pre ○ Post √

Function Get the status of the current line or current position, and get its trigger signal of the rising edge.

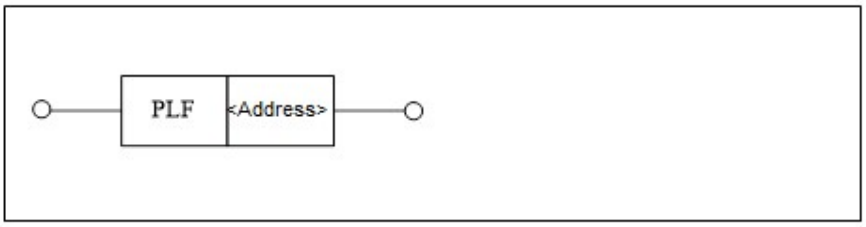
Set the input signal to 1 in the current scan cycle of the rising edge signal. (Note the difference between the trigger component of rising edge for basic component and this function). This function is suitable for the situations where the rising edge status needs to be detected.

Example

Ladder Diagram	
Description	When R100.0 status is changed from 0 to 1, its rising edge status is obtained and R10.0 is output as 1.

4.7.16 Fetch Falling edge PLF

Format

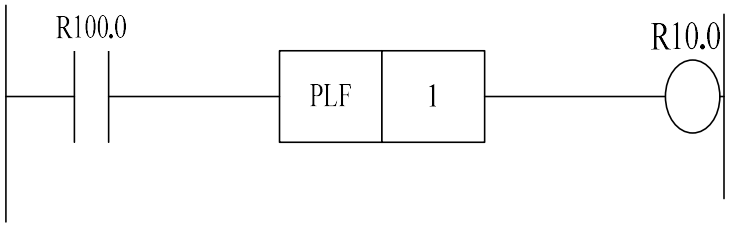


Parameter	Parameter form	Data type	Storage area	Explanation	Properties
<Address 1>	□□□□	INT	Constant	Falling edge module number	Pre ○ Post √

Function Get the status of the current line or current position, and get its trigger signal of the falling edge.

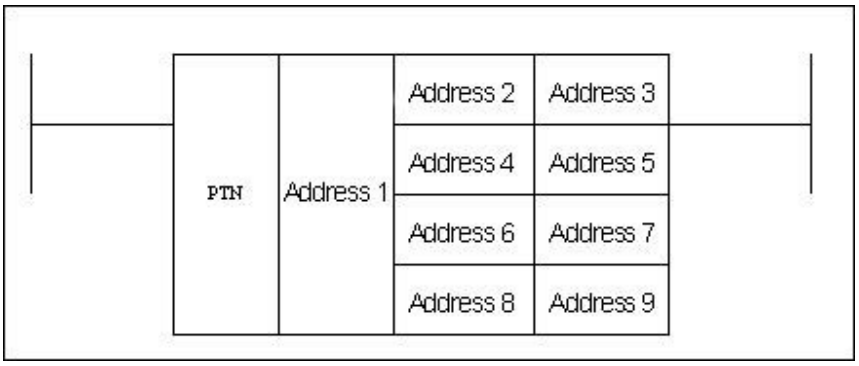
Set the input signal to 1 in the current scan cycle of the falling edge signal. (Note the difference between the function here and the trigger component of falling edge in basic component). This function is suitable for the situations where the falling edge status needs to be detected.

Example

Ladder Diagram	
Descri	When R100.0 status is changed from 1 to 0, its falling edge status and is obtained and R10.0 is output as 1.

4.7.17 Points Transformation PTN

Format



Parameter	Parameter form	Data type	Storage area	Explanation	Properties
<Address 1>	□□□□	BOOL	Y, G, R, W, D, B	When the point is effective, the corresponding number is generated.	Pre ○
<Address 2>	□ □ □ □ . □	BOOL	X, Y, F, G, R, W, D, P, T, C, B		
<Address 3>	□□□□	INT	Constant		
<Address 4>	□ □ □ □ . □	BOOL	X, Y, F, G, R, W, D, P, T, C, B		
<Address 5>	□□□□	INT	Constant		Post ×
<Address 6>	□ □ □ □ . □	BOOL	X, Y, F, G, R, W, D, P, T, C, B		
<Address 7>	□□□□	INT	Constant		
<Address 8>	□ □ □ □ . □	BOOL	X, Y, F, G, R, W, D, P, T, C, B		
<Address 9>	□□□□	INT	Constant		

Function To build the corresponding relationship between points and numbers. When the point is effective, the corresponding number is generated.

Parameter

Parameter 1: the destination address.

Parameter 2: point 1

Parameter 3: number 1

Parameter 4: point 2

Parameter 5: number 2

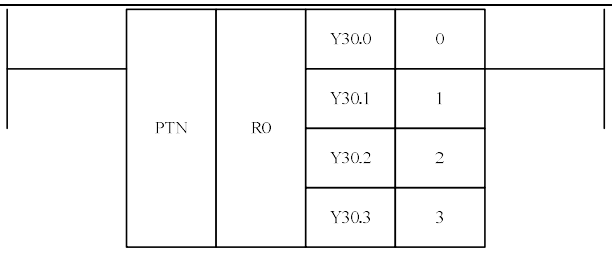
Parameter 6: point 3

Parameter 7: number 3

Parameter 8: point 4

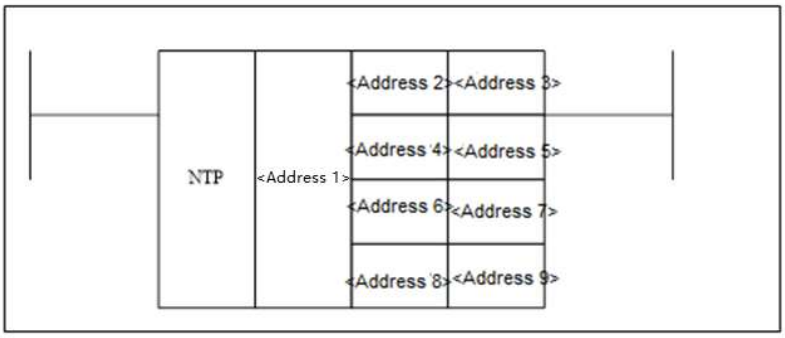
Parameter 9: number 4

Example

Ladder Diagram	
Description	When Y30.0 is effective, R0=0. When Y30.1 is effective, R0=1. When Y30.2 is effective, R0=2. When Y30.3 is effective, R0=3.

4.7.18 Number Conversion NTP

Format



Parameter	Parameter form	Data type	Storage area	Properties
<Address 1>	□□□□	BOOL	Y, G, R, W, D, B	Pre ○
<Address 2>	□□□□	INT	Constant	
<Address 3>	□□□□. □	BOOL	X, Y, F, G, R, W, D, P, T, C, B	
<Address 4>	□□□□	INT	Constant	
<Address 5>	□□□□. □	BOOL	X, Y, F, G, R, W, D, P, T, C, B	Post ×
<Address 6>	□□□□	INT	Constant	
<Address 7>	□□□□. □	BOOL	X, Y, F, G, R, W, D, P, T, C, B	
<Address 8>	□□□□	INT	Constant	
<Address 9>	□□□□. □	BOOL	X, Y, F, G, R, W, D, P, T, C, B	

Function To build the corresponding relationship between numbers and points. The point signal corresponding to the number in Parameter 1 is generated.

Parameter 1: the address of source data

Parameter 2: number 1

Parameter 3: point 1

Parameter Parameter 4: number 2

Parameter 5: point 2

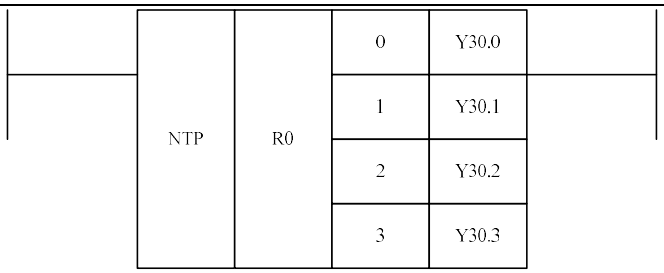
Parameter 6: number 3

Parameter 7: point 3

Parameter 8: number 4

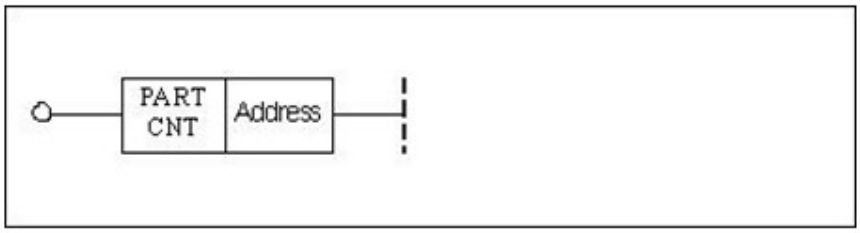
Parameter 9: point 4

Example

Ladder Diagram	
Description	When R0=0, Y30.0 is effective. When R0=1, Y30.1 is effective. When R0=2, Y30.2 is effective. When R0=3, Y30.3 is effective.

4.7.19 Parts Count PARTCNT

Format

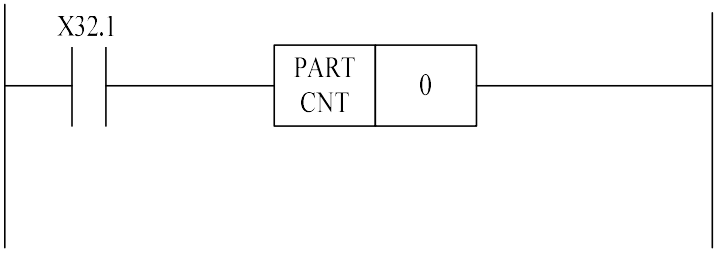


Parameter	Parameter form	Data type	Storage area	Explanation	Properties
<Address 1>	□□□□	INT	Constant	When condition is satisfied, the parts count of <Address 1> Channel will increase by 1.	Pre ○ Post √

Function To count machined parts.

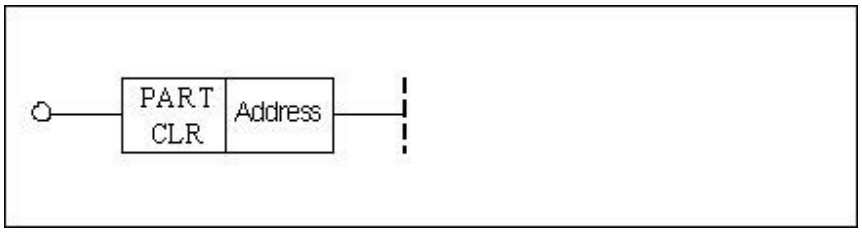
Parameter Parameter 1: channel number

Example

Ladder Diagram	
Description	When 32.1 is turned on, 1 is added to the parts count of channel 0.

4.7.20 Parts-counting Clear PARTCLR

Format



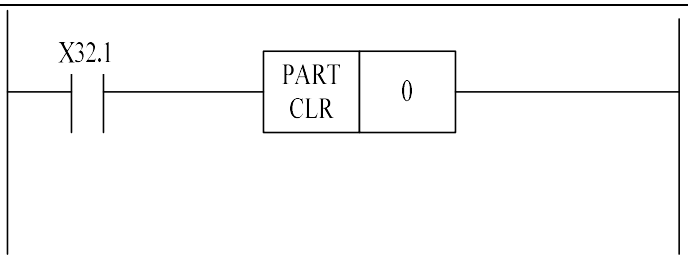
Parameter	Parameter form	Data type	Storage area	Explanation	Properties
<Address 1>	□□□□	INT	Constant	When condition is satisfied, parts count of <Address > channel is cleared.	Pre ○ Post √

Function Clear the count of the parts.

Parameter Parameter 1: channel number.

Example

Ladder Diagram

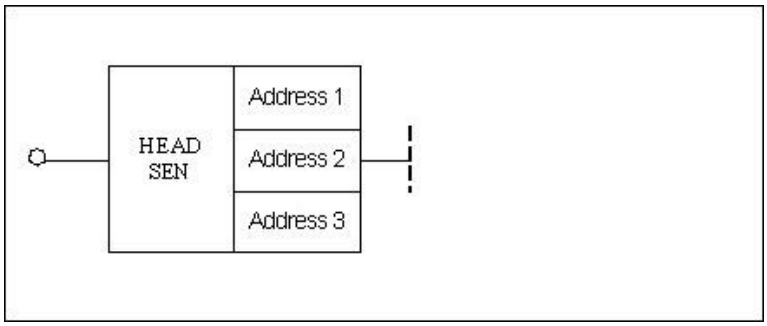


Description

When X32.1 is turned on, the parts count in channel 0 is cleared.

4.7.21 Temperature Collection Module HEADSEN

Format



<Address 1>	□□□□.□	BOOL	X, Y, F, R, W, D, P, T, C, B	When <address 2> is 0, the temperature collection module starts to count, and the temperature data in <address 1> is stored in the starting address given by <address 3>.	Pre ○ Post ×
<Address 2>	□□□□	INT	Constant		
<Address 3>	□□□□	INT	Constant		

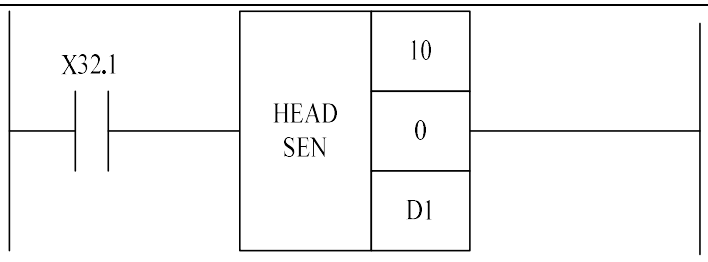
Function Temperature collection module.

Parameter

Parameter 1: total quantity of temperature collections, can be constant. Parameter 2: enable switch of temperature collection module, 0 indicates counting starts, other values indicate the module is not enabled.

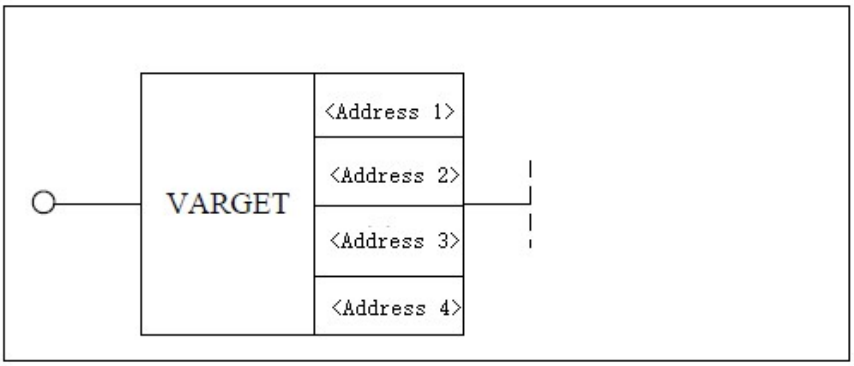
Parameter 3: the initial location where the temperature collection data is stored, and it can be register D.

Example

Ladder Diagram	
Description	When X32.1 is turned on, the temperature collection module starts to count, and 10 temperature data is stored in the initial location given by D1.

4.7.22 Variable Reading Module VARGET

Format



Parameter	Format	Type	Storage area	Description	Properties
<Address 1>	□□□□.	INT	Constant (0~9)	If the offset number address variable value of <address 2> corresponding to variable type is an integer, this value will be read to <address 4>; if the variable value is a floating point, it will be read to <address 4> after enlarging the exponential times of 10	Pre √
<Address 2>	□□□□	INT	Constant, X, Y, F, G, R, W, D, P, B		
<Address 3>	□□□□	INT	Constant (0~4)		
<Address 4>	□□□□		Y, G, R, W, D, B		

Function To read variable values of system.

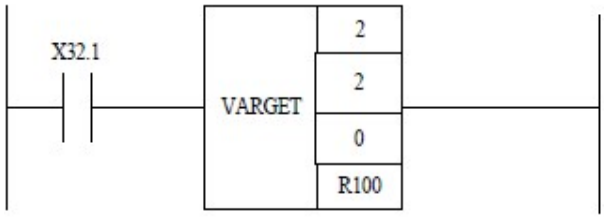
Parameter Parameter 1: type of variable.

Parameter 2: offset number of variable address which is read

Parameter 3: floating point variable which is increased by power of 10 times.

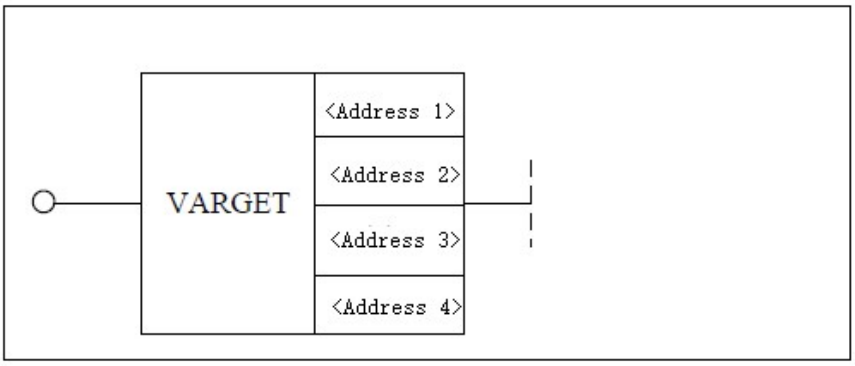
Parameter 4: result address

Example

Ladder Diagram	
Description	When X32.1 is turned on, system reads the value of the user variable #501 into the register R100.

4.7.23 Variable Writing Module VARSET

Format



Parameter	Format	Type	Storage area	Description	Properties
<Address 1>	□□□□	INT	Constant (0-9)	If the value in <Address 4> is an integer, directly assign the value to <Address 2> corresponding to the variable type; if the value of <Address 4> is a floating point type, set it to <Address 2> after magnifying the exponential multiple of <Address 3> of 10	Pre √
<Address 2>	□□□□	INT	Constant, X, Y, F, G, R, W, D, P, B		
<Address 3>	□□□□	INT	Constant (0-4)		
<Address 4>	□□□□		Y, G, R, W, D, B		

Function To set variable values of system.

Parameter

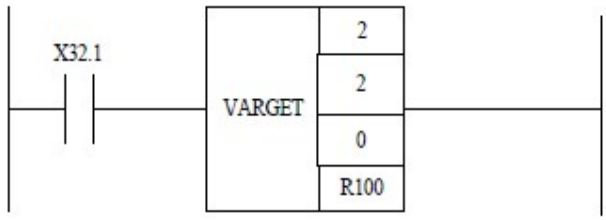
Parameter 1: type of variable.

Parameter 2: address offset number corresponding to the variable type.

Parameter 3: floating point variable which is increased by power of 10 times.

Parameter 4: data address

Example

Ladder Diagram	
Description	When X32.1 is turned on, system sets the values in register R100 to the user variable #501.

Variable type

Variable type value (address 1)	Variable offset number (address 2)
0: user variable	0-4999 corresponding to #50000-#54999
1: extension user variable, compatible for FANUC #500-#999	0-499 corresponding to #500-#999
2: 32-bit integer system variable	0-9999
3: 64-bit integer system variable	0-4999
4: floating point system variable	0-4999
5: 32-bit integer channel variable	2000*ch+0-1999
6: floating point channel variable	1000*ch+0-999
7: 32-bit integer axis channel	100*ax+0-99
8: 64-bit integer axis channel	50*ax+0-49
9: tool variable	200*t+0-199 corresponding to #(70000+200*t+0) ~ #(70000+200*t+199)

5 Status Word and Control Word Programming

This chapter includes:

5.1 Introduction on Status Word and Control Word

5.2 Example of Status Word and Control Word Programming

5.1 Introduction on Status Word and Control Word

Overview

The status word and control word are the most direct way of the interaction between CNC and PLC. The status data of system can be obtained through the status word, and user can write control word to change the system state. In the HNC8 system, F represents status word with its property being read-only, G represents control word with its property being read-write.

However, to limit the use of some key functions of system, some control words are restricted, or are invisible to user. Please read following constraints of status word and control word carefully.

Usage restrictions of status word and control word

	can be used
	Reserved for future expansion
	Not allowed to be used by user

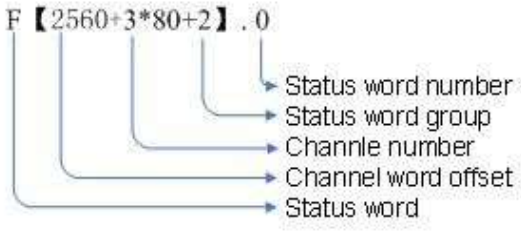
Range of application

Status words and control words can be divided into three types for each of its function in the system. They all have serviceable range based on the system model, refer to the configuration manual for details.

- Status words and control words of axis
- Status words and control words of channel
- Status words and control words of system

Symbol form

Take the words of channel as an example:



This example shows the format: channel 3, the second group of status word, and

No.0 status word. 2560 is the offset of the channel status word. The format of other types of words is similar.

5.1.1 Axis Status Word

Overview

80 status words are configured for each axis. Each status word has 16-bit bytes. The first row indicates the bits from 0 to 7, and the second row indicates the bits from 8 to 15. The axis status words need to be used with the logical number offset of axis.

Axis status word

D7	D6	D5	D4	D3	D2	D1	D0
D15	D14	D13	D12	D11	D10	D9	D8

F0

Slave axis follow	Slave axis zero	Homing of slave axis	Homing completion	Homing failure	1st reference point return	2nd reference point return	Axis motion
Axis reset	Axis lock	Axis parameter OK	Axis overload	4th reference point	3rd reference point	2nd reference point	1st reference point

F1

SPD arrival	Spindle zero speed	Orientation completion	Rapid traverse feed	Reserved	Reserved	Spindle mode	PMC enable
Index axis lock	Index position	Index axis unlock	Reserved	Reserved	Reserved	Reserved	Reserved

F2

Servo parameter	Zero position capture	Reserved	Servo homing	2Enc zero	Reserved	Servo ready	First Z capture
Zero speed of spindle	SPD arrival	Gain switching	Z pulse capture	Torque control	Speed control	Position control	SV ready

F3

Reserved	Reserved	Reserved	Reserved	Reserved	Servo prompt	Servo alarm	Servo normal
Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Orientation completion

Details

【F0.0】During the axis movement, when the axis is moving, the value is 1; when the axis is not moving, the value is 0.

【F0.1】The first step of homing: when the axis is homing without meeting home block, the value is 1; otherwise, the value is 0.

【F0.2】The second step of homing: when Z pulse is being looked for, the value is 1; otherwise, the value is 0.

【F0.3】Unsuccessful homing: when the axis homing is not completed, the value is 1; otherwise, the value is 0.

【F0.4】Successful homing: When the axis has been to zero, the value is 1; otherwise, the value is 0.

【F0.5】Slave axis is returning to reference point.

【F0.6】Reference position for slave axis has been checked.

【F0.7】Following status of slave axis has been lifted.

【F0.8】Comfirm the first reference: when the axis is at the first reference point, the value is 1; otherwise, the value is 0.

【F0.9】Comfirm the second reference: when the axis is at the second reference point, the value is 1; otherwise, the value is 0.

【F0.10】Comfirm the third reference: when the axis is at the third reference point, the value is 1; otherwise, the value is 0.

【F0.11】Comfirm the forth reference: when the axis is at the forth reference point, the value is 1; otherwise, the value is 0.

【F0.13】Axis parameter take effect.

【F0.14】Axis has been locked.

【F0.15】Axis has been repositioned.

【F1.0】PMC control enable. When PMC control has been enabled, the value is 1; otherwise, the value is 0.

【F1.1】Feed spindle mode. 1 is position mode, and 0 is speed mode.

【F1.5】Orientation of feed spindle has been finished.

【F1.6】Feed spindle is at zero speed.

【F1.7】Feed spindle speed arrival.

【F1.13】Index axis is unlocked. 1 indicates that system notifies PLC to unlock index axis, and the index axis is enabled.

【F1.14】Index axis is at index position.

【F1.15】Index axis is locked. 1 indicates that system notifies PLC to lock index axis. The index axis is disabled.

【F2.0】When Z pulse is captured once during homing of axis, the value is 1; otherwise, the value is 0.

【F2.1】When servo ready flag is 0, servo can receive incremental data.

【F2.3】Capture Z pulse of the second encoder, which is mainly used for homing of distance-coded grating scale.

【F2.4】When servo has been back to zero, the output is 1.

【F2.6】Zero capture, which is mainly used for spindle. When spindle meets the first Z pulse at rotating time, set the value to 1. In the event of CS switching, the value needs to be set to 1.

【F2.7】Servo parameter switching. 0: default parameter. 1: switch to the second set of servo parameter.

【F2.8】When bus servo is ready, the value is 1; otherwise, the value is 0.

【F2.9】When servo is in position control mode, the value is 1; otherwise, the value is 0.

【F2.10】When servo is in speed control mode, the value is 1; otherwise, the value is 0.

【F2.11】When servo is in torque control mode, the value is 1; otherwise, the

value is 0.

【F2.12】When Z pulse is encountered, the value is 1; otherwise, the value is 0.

【F2.13】

【F2.14】When the spindle speed reaches, the value is 1, otherwise, the value is 0.

【F2.15】Spinde stop: when spindle stops, the value is 1; otherwise, the value is 0.

【F3.0】When servo is normal, the value is 1.

【F3.1】When servo alarms, the value is1.

【F3.2】When servo prompts, the value is 1.

【F3.8】Spindle orientation completion. After spindle orientation is set, spindle starts to orient. After the orientation is completed, servo returns the signal of completing spindle orientation, and the value is 1; otherwise, the value is 0.

【F4】Number of channel which the axis belongs to. (Channel number is in decimal.)

【F5】Number of slave axes which are guided. (Number of slave axes is in decimal.)

【F[6/7]】Real-time output command increment, motor coordinate.

【F[8/9/10/11]】Real-time output command position, motor coordinate. (metric unit)

【F[12/13/14/15]】Output command pulse position, unit: pulse.

【F[16/17]】Command pulse per cycle. Number of command pulses which is sent to servo each cycle.

【F[18/19]】Output command torque.

【F[20/21/22/23]】Actual feedback position of encoder 1. (metric unit)

【F[24/25/26/27]】Actual feedback position of encoder 2. (metric unit)

【F[28/29/30/31]】Command position of machine. (metric unit)

【F[32/33/34/35]】Actual position of machine. (metric unit)

【F[36/37]】Axis alarm

【F36.2】Plus software limit switch is reached.

【F36.3】Minus software limit switch is reached

【F36.4】Actual speed is overspeed.

【F36.6】Overspeed

【F36.7】Ultra acceleration.

【F36.8】Z pulse cannot be found.

【F36.9】Connection has been aborted.

【F36.10】Reference point in not returned.

【F36.11】Sync position out-of-tolerance

【F36.12】Slave axis zero check is aborted

【F36.13】Sync speed out-of-tolerance

【F37.0】Plus software limit is exceeded.

【F37.2】Minus software limit is exceeded.

【F37.2】Acceleration does not match maximum speed.

【F[38/39]】Axis prompt

【F38.0】Max compensation ratio is exceeded.

【F38.1】Max compensation is exceeded.

【F38.2】Zero offset parameter is too small.
【F38.4】Software limit is too large.
【F38.5】The second software limit is too large.
【F38.6】Absolute encoder cycle digits are illegal.
【F38.7】Position overflow.
【F38.8】Target is outside plus software limit.
【F38.9】Target is outside minus software limit.
【F38.10】Mask angle of Z pulse needs to be adjusted.
【F38.11】Reference point needs to be adjusted.
【F38.12】Tracking error is too large.
【F[70]】Current mode of axis.

5.1.2 Axis Control Word

Overview

80 control words are configured for each axis. Each control word has 16-bit bytes. The first row indicates the bits from 0 to 7, and the second row indicates the bits from 8 to 15. The axis control words need to be used with the logical number offset of axis.

Axis control word

D7	D6	D5	D4	D3	D2	D1	D0
D15	D14	D13	D12	D11	D10	D9	D8

G0

Axis enable	Axis lock	Homing block	Homing start	Inhibition in minus direction	Inhibition in plus direction	Minus limit	Plus limit
Axis reset	Compensation extension	Reserved	Reserved	Slave axis follow	Reserved	Reserved	Reserved

G1

*SP rotation CCW	*SP rotation CW	*SP orientation	*SP Jog	Extension software limit	Second software limit	Relative pmc motion	Absolute pmc motion
Response locking	Response unlocking	Reserved	CS response	Reserved	Reserved	Reserved	Reserved

G2

Servo Parameter	Reserved	Reserved	Reserved	Capture Z pulse of encoder 2	Reserved	Reserved	Capture Z pulse
Spindle current-limiting	Orientation gear-shift	Reserved	Spindle orientation	Torque control	Speed control	Position control	Servo gain

G3

Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Servo enable
Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved

Details

【G0.0】Plus limit switch of axis.

【G0.1】Minus limit switch of axis.

【G0.2】No axis movement in positive direction.

【G0.3】No axis movement in negative direction.

【G0.4】Set to start homing.

【G0.5】Set homing block.

【G0.6】Set to lock the axis.

【G0.7】Set to axis enable

【G0.11】Set to disable function of slave axis following

【G0.14】Compensation expansion

【G0.15】Single-axis reset

【G1.0】Absolute PMC axis motion is enabled.

【G1.1】Relative PMC axis motion is enabled.

【G1.2】The second software limit is enabled.

【G1.3】Extension software limit is enabled.

【G1.4】Feed-spindle JOG.

【G1.5】Feed-spindle orientation.

【G1.6】Feed -spindle rotates in clockwise direction.

【G1.7】Feed -spindle rotates in counter clockwise direction.

【G1.12】Response flag of PLC to spindle C/S switching.

【G1.14】Response flag of PLC to signal of unlocking index axis.

【G1.15】Response flag of PLC to signal of locking index axis.

【G2.0】Z pulse flag. (when motor is at the position of Z pulse, this flag is 1.)

【G2.1】Wait for zero pulse

【G2.2】Turn off function of searching zero pulse.

【G2.3】Capture zero pulse of the second encoder.

【G2.7】Servo parameter switching. 0: Default parameter, 1: switch to the second set of parameters.

【G2.8】Servo gain switching.

【G2.9】Switch to position control mode.

【G2.10】Switch to speed control mode.

【G2.11】Switch to torque control mode.

【G2.12】Spindle orientation start.

【G2.14】Directional gear-shift of spindle.

【G2.15】Spindle current limiting.

【G3.0】Servo enable switch.

【G4】Axis jog flag. When the axis is manual, or returning to zero, or the spindle is rotating, this flag is effective.

【G5】Increment flag of axis. When axis is moving incrementally, this flag is effective.

【G[6/7]】Jog speed. 0: stop; 1: Jog speed in parameter; 2: Rapid traverse speed in parameter; >2: Self-defined speed.

【G8】Incremental magnification.

【G9】Handwheel magnification.

【G[10/11]】handwheel pulse.

【G[12/13/14/15]】Axis feedback position, unit: pulse

【G[16/17/18/19]】Axis feedback position 2, unit: pulse

【G[20/21]】Actual speed of axis, unit: pulse. Actual axis-speed is the incremental value per cycle of the actual feedback position of axis (G12-G15).

【G[22/23]】Actual speed 2 of axis

【G[24/25]】Actual torque of axis

【G[26/27]】Tracking error. (Tracking error of axis is the difference between the actual axis feedback position (G12-G15) and the axis command position (F12-F15).)

【G[28/29/30/31]】Counter value of encoder 1

【G[32/33/34/35]】Counter value of encoder 2

【G[36/37]】Real-time compensation value.

【G[38/39]】Sample timestamp

【G[40/41/42/43]】Latch position 1 (when the first encoder has Z pulse, the current position is latched, which is used for homing of G31 or distance code.

【G[44/45/46/47]】Latch position 2 (when the second encoder has Z pulse, the current position is latched, which is used for homing of G31 or distance code.

【G[48/49/50/51]】Target position of absolute movement for PMC axis.

【G[52/53/54/55]】Incremental movement of PMC axis

【G[56/57]】Servo alarm code.

【G[58/59]】Servo prompt code.

【G60】Axis control mode switching (2 is handwheel interruption, and 103 is PMC mode)

【G61】Override value of PMC axis.

【G62.0】PMC axis stop.

【G62.1】Handwheel interruption reset.

【G62.2】Turn on function of tangent following.

【G62.4】Index axis switch.

【G62.5】Synchronize the axis position when the slave axis coupling is restored

【G62.8】Spindle control, write actual rotation speed to instruction.

【G62.9】Start spindle rotation speed of gear shift.

【G64】Current axis gear.

【G66/67】Gear shift of spindle.

【G68/69】Z pulse position.

【G70/71】Z pulse interval 1.

【G72/73】Z pulse interval 2.

【G74】Gear shift of spindle.

【G78/79】Sample data of servo

5.1.3 Channel Status Word

Overview 80 control words are configured for each channel. Each control word has 16-bit bytes. The first row indicates the bits from 0 to 7, and the second row indicates the bits from 8 to 15. The axis control words need to be used with the logical number offset of channel.

Axis status word

D7	D6	D5	D4	D3	D2	D1	D0
D15	D14	D13	D12	D11	D10	D9	D8

F2560

User intervention	Motion at the time of non-auto	Cycle start	Feedhold	Mode #3	Mode #2	Mode #1	Mode #0
Search Z pulse	Resetting	Dwell request	Reset flag	Verify	Reserved	Thread turning	Cutting

F2561

reserved	reserved	Await completion	Interruption skip	Interruption completion	Program completion	Program start	Program selected
reserved	reserved	reserved	reserved	reserved	reserved	Non-empty completion	Non-empty instruction

F2562

reserved	reserved	reserved	reserved	reserved	reserved	reserved	reserved
4S instruction	3S instruction	2S instruction	1S instruction	Constant linear speed of spindle	Index instruction	Tool offset flag	reserved

Details

【F2560.0 ~F2560.3】To get mode

0: Reset mode 1: Auto mode 2: JOG mode
3: Incremental mode 4: Handwheel mode 5: Homing mode
6: PMC mode 7: Single block mode 8: MDI mode

【F2560.4】Feedhold: channel is in state of feedhold.

【F2560.5】Cycle start: channel is in state of cycle start.

【F2560.6】There is movement in non-auto mode.

【F2560.7】There is user movement intervention.

【F2560.8】Cutting.

【F2560.9】Thread-cutting: channel is in state of cutting thread, and feedhold is not allowed.

【F2560.11】Verification state.

【F2560.12】Channel reset: in the event of channel reset or reset button on panel being pressed, channel reset is effective, until channel reset response is set.

【F2560.13】Suspend request.

【F2560.14】Channel is resetting.

【F2560.15】When axis is homing to look for Z pulse, switching mode is not allowed.

【F2561.0】Program is selected, which is set by interpreter.

【F2561.1】Program start, which is set by channel control.

【F2561.2】Program is completed, which is set by channel control.

【F2561.3】Interrupt instruction G28/G31 is completed.

【F2561.4】Skip interrupt instruction.

【F2561.5】Wait for completing instruction.

【F2561.8】There are non-empty instruction flags in channel.

【F2561.9】Non-empty instruction flag is completed in channel.

【F2562.9】Tool offset mark [tool offset number is in T instruction]

【F2562.10】PLC index instruction flag.

【F2562.11】Constant linear speed of spindle.

【F2562.12】The first S instruction.

【F2562.13】The second S instruction.

【F2562.14】The third S instruction.

【F2562.15】The forth S instruction.

【F2569】Tool offset number, which is in T instruction.

【F[2570/2571】The first S instruction. Unit: 0.001 revolution/ minute.

【F[2572/2573】The second S instruction. Unit: 0.001 revolution/minute.

【F[2574/2575】The third S instruction. Unit: 0.001 revolution/minute.

【F[2576/2577】The forth S instruction. Unit: 0.001 revolution/minute.

【F2578/79】G31 number which is currently waiting signal.

【F2580】The currently running coordinate system

【F[2581/2589]】Axis number of 9 axes in channel

【F[2590/2593]】Axis number of 4 spindles in channel.

【F[2594/2595]】Alarm code for syntax error.

【F[2596/2599]】Channel alarm code.

【F[2600/2603]】Channel prompt number.

【F[2604/2607]】User output.

【F[2608/2615]】M codes which run in channel, with a maximum of 8.

【F2616】T instruction in channel. When T code is executing in channel, the value of T code is in register; otherwise, the output is -1.

【F2617】B instruction in channel. B axis in boring machine is executed by PLC, and indexing is executed by B instruction.

【F2632】Number of tool which is alarmed for the maximum life span being reached.

【F2636.0】Channel is resetting.

【F2632.1】Program has been stopped exactly.

【F2632.2】Flag of inclined axis

【F2632.3】Interpolation instruction runs in channel.

【F2632.4】Flag of spindle synchronization.

【F2632.5】Handwheel feed direction.

【F2637.0】Subprogram process start.

【F2637.1】Subprogram waits for feedhold, and saves breakpoint.

【F2637.2】Break point flag.

【F2637.3】Start to load subprogram.

【F2637.4】Complete loading.

【F2637.5】Start running.

【F2637.6】Complete running.

【F2637.7】Breakpoint has been restored.

【F2637.8】Process ends.

【F2637.9】Process error.

【F2637.10】Process reset.

【F2637.11】Process waits for interpreter to complete reset.

【F2638.0】Cumulative flag of tool changing in tool life

5.1.4 Channel Control Word

Overview 80 control words are configured for each channel. Each control word has 16-bit bytes. The first row indicates the bits from 0 to 7, and the second row indicates the bits from 8 to 15. The axis control words need to be used with the logical number offset of channel.

Axis control word

D7	D6	D5	D4	D3	D2	D1	D0
D15	D14	D13	D12	D11	D10	D9	D8

G2560

Measurement interruption	Dry run	Cycle start	Feedhold	Work mode	Work mode	Work mode	Work mode
Data save	Data recovery	Reset	Buff clear	Emergency stop	Panel reset	Reset response	Verify

G2561

Data recovery	Any line	Rerun	Interpretation reset	Optional stop	Block skip mark	Rerun 2	Interpreter startup
Reserved	Program modification	Reserved	Handwheel interruption	External interruption	User motion	Reserved	Save interpretation

G2562

Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
Reserved	Reserved	Rotation speed arrival	No spindle	MST lock	Speed check	Reserved	Reserved

G2620

Panel enable	PMC	Handwheel	Homing	Increment	JOG	Single block	Auto
Reserved	Reserved	Reserved	Reserved	Reserved	Rapid traverse	Incremental magnification	

G2621

Handwheel 1				Handwheel 0	
Reserved	Reserved	Reserved	Handwheel 1 enable	Handwheel 1 magnification	Handwheel 0 magnification

G2622

Axis 7+	Axis 6+	Axis 5+	Axis 4+	Axis 3+	Axis 2+	Axis 1+	Axis 0+
Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Axis 8+

G2623

Axis 7-	Axis 6-	Axis 5-	Axis 4-	Axis 3-	Axis 2-	Axis 1-	Axis 0-
Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Axis 8-

Details

【G2560.0/1/2/3】Work mode. 0: reset mode, 1: auto mode, 2: manual mode, 3: increment mode, 4: handwheel mode, 5: homing mode, 6: PMC mode, 7: single block mode, 8: MDI mode

【G2560.4】Feedhold: Set feedhold in channel.

【G2560.5】Cycle start: set cycle start in channel.

【G2560.6】Dry run: set to dry run in channel.

【G2560.7】Mearsurement interruption flag. When this flag is set to 1, system interrupts ongoing G31 instruction. It is used with G2582.

【G2560.8】Verification

【G2560.9】PLC reset response: when PLC has been reset, set this flag to 1.

【G2560.10】Panel reset flag. Through detecting this flag, PLC determines whether the system is resetting.

【G2560.11】Emergency stop flag. This flag is set for emergency stop of machine.

【G2560.12】Flag of channel buffering clear.

【G2560.13】This flag is set when resetting machine.

【G2560.14】Flag of channel data recovery.

【G2560.15】Channle data save.

【G2561.0】Flag of interpreter startup.

【G2561.1】Program reruns the second step.

【G2561.2】Flag of skipping block. When this flag is set to 1, system skips block.

【G2561.3】Optional stop flag. When this flag is set to 1, system performs optional stop.

【G2561.4】Flag of interpreter reset.

【G2561.5】Flag of program rerun.

【G2561.6】MDI resets to program header

【G2561.7】Flag of interpreter data recovery.

【G2561.8】Flag of interpreter data save.

【G2561.9】Exact-stop check.

【G2561.10】Flag of user motion control.

【G2561.11】Flag of external interruption.

【G2561.12】Turn on handwheel interruption.

【G2561.13】When rapid traverse override is 0, use the feed override to control G00, up to 25%.

【G2561.14】Flag of program modification.

【G2561.15】Coordinate of workpiece or tool changes, re-interpretation is requested.

【G2562.0】S instruction response word of No.1 spindle.

【G2562.1】S instruction response word of No.2 spindle.

【G2562.2】S instruction response word of No.3 spindle.

【G2562.3】S instruction response word of No.4 spindle.

【G2562.8】Feed direction of precutting by handwheel. 0 is moving forward, 1 is retracting.

【G2562.10】Spindle speed check.

【G2560.10】Flag of panel reset. PLC determines whether system is resetting by detecting this flag.

【G2560.11】Emergency stop, to set channel of emergency stop.

【G2562.11】MST lock.

【G2562.12】Spindle is not started.

【G2562.13】Spindle speed doesn’t reach.

【G2562.14】Following start.

【G2562.15】Precutting by handwheel. Use handwheel magnification.

【G2563】T instruction.

【G2564】Feedrate override.

【G2565】Rapid traverse override.

【G2566/67/68/69】Spindle override. Override of four spindles in channel.

【G2570/71/72/73/74/75/76/77】Spindle output instruction, output instructions of four spindles in channel. After obtaining spindle rotation speed【F2570-F2577】, PLC calculates spindle override, and outputs spindle instruction. In servo spindle, the output is the spindle speed, and in PMW spindle, the output is DA value.

【G2578】F2578.1 imaginary axis control.

【G2579】Machined-parts count.

【G2580/81】Protected area mask

【G2582】G31 number. When G31 execution is interrupted, the number of interrupted G31.

【G2584/85/86/87】User bit input

【G2588~2607】User value input.

【G2608~2615】M code response of channel. When PLC is not executing M code, set to -1; when PLC is executing M code, set to -2; when PLC has executed M code, set to the currently executed M code.

【G2616】T code response of channel. When PLC has executed T code, set to the currently executed T code; otherwise, set to -1.

【G2617】Tangent following of tool.

【G2636.0】Channel reset (PLC sets register, and notifies HMI to reset channel.)

【G2636.3】IRQ control.

【G2636.4】Channel resetting is not allowed. 【Reset button is invalid】

【G2636.5】Life timekeeping/counting pause.

【G2560.15】Channel data save

【G2561.0】Interpreter start.

【G2637】Subprogram calling start.

【G2638】Counting of tool changing.

【G2970】Flag of system activity channel.

【G2978】Control word of system activity control channel

【G2980~2989】Control word of handwheel 【previous axis selection】

【G2990~2999】Display output of handwheel.

【G3010~3025】External alarm of PLC (External alarm of PLC, and 8*32=256 external alarms exist concurrently).

【G3040~3055】External event of PLC (external event of PLC, and 8*32=256 external events exist concurrently).

【G3056~3070】External reminder of PLC (external reminder of PLC and 8*32 256 external events exist concurrently)

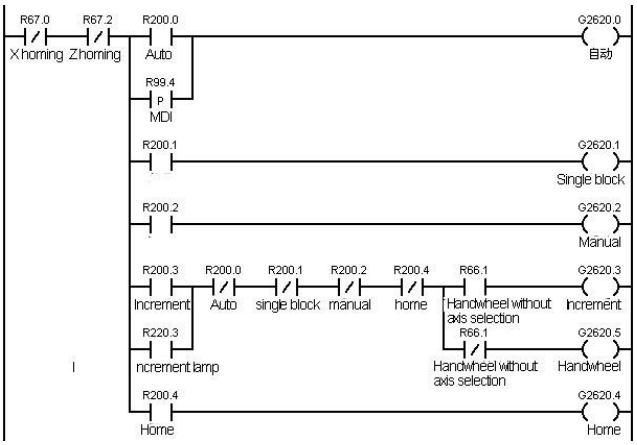
【G3080~3099】Temperature sensor value

5.2 Example of Status Word and Control Word Programming

5.2.1 Working Mode Setting

Example

Ladder diagram



Function Set the status in the working mode of channel. When the axis is in the position control mode, set the working mode in the current channel to auto, single block, JOG, increment, handwheel or home.

5.2.2 Working Mode Obtaining

Example

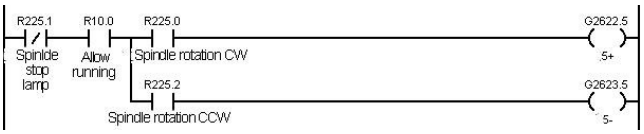
Ladder diagram

Function Get the status in the working mode of channel, which can be auto, single block, JOG, increment, handwheel or home.

5.2.3 Control of Feed Axis and Spindle

Example

Ladder diagram

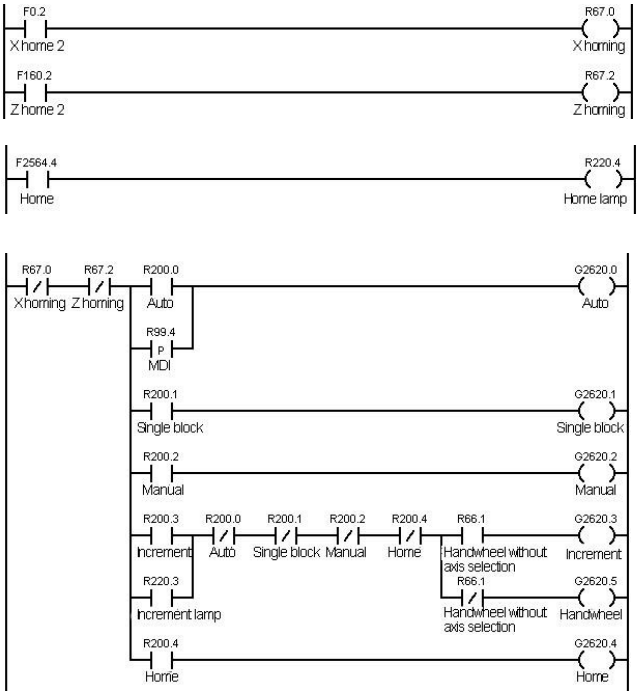


Function It is used to control the movement of feed axis and the spindle rotation. Set the current channel mode to JOG mode, if you press Axis selection, and positive or negative movement buttons, the moving status of the current axis will be set, thus the axis will move; if you press spindle rotation (CW or CCW) button, the rotation direction of the spindle will be set.

5.2.4 Home

Example

Ladder diagram

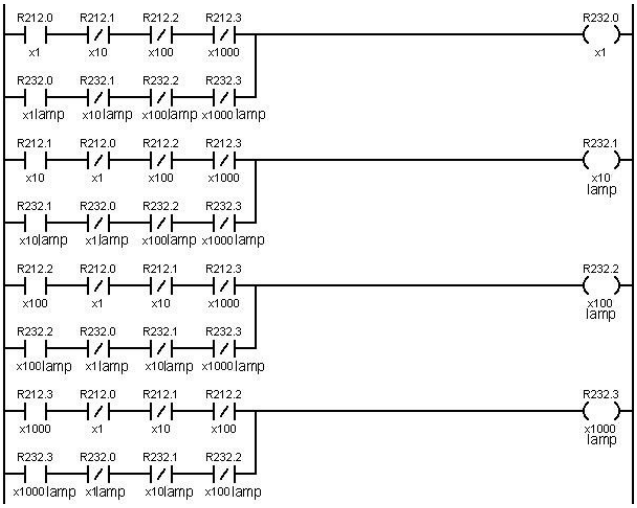


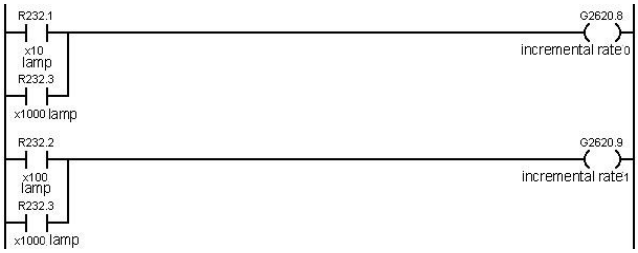
Function Obtain whether the current channel is returning home through the status register. During the process of meeting the home block, which is the first process of homing, switching to other statuses is allowed; During the process of researching Z pulse, which is the second process of homing, switching to other statuses is not allowed.

5.2.5 Incremental Magnification Override

Example

Ladder diagram



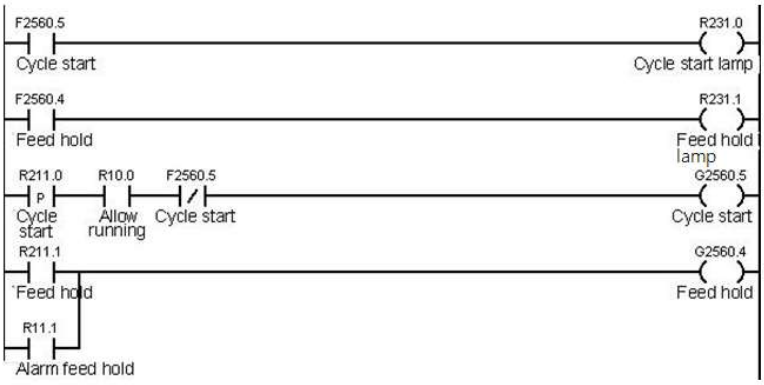


Function Incremental magnification consumes two bits. 00 represents x1, 01 represents x10, 10 represents x100, and 11 represents x1000. The axis movement is controlled by the setting of above axis register.

5.2.6 Cycle Start and Feed Hold

Example

Ladder diagram

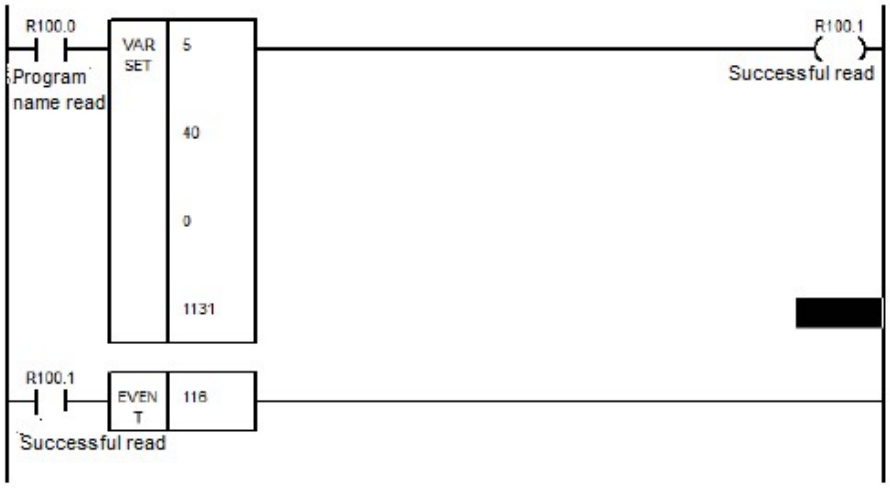


Function When the working mode in channel is auto or single block, and is not at cycle start, set to cycle start. Set to feed hold under the cycle start. If the setting is successful, the system will be at the state of feed hold.

5.2.7 Program Name Specified by Loaded Variable

Example

Ladder diagram



Function Write the program number to be loaded into the variable in channel 1131, the PLC sends the event 116, and the system automatically loads the program corresponding to the variable program name in channel 1131 after receiving the event 116.

6 Extension Function Module

This chapter includes:

6.1 NC Function

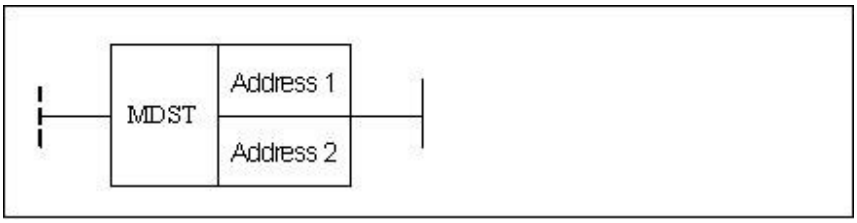
6.2 Functional Unit of Axis

6.3 System Function

6.1 NC Function

6.1.1 Channel Mode Setting MDST

Format



Parameter	Parameter form	Data type	Storage area	Explanation	Properties
<Address 1>	□□□□	INT	Constant	Channel No.	Pre ○
<Address 2>	□□□□	INT	Constant, F, G, R, W, D, P, B	Work mode value	Post ×

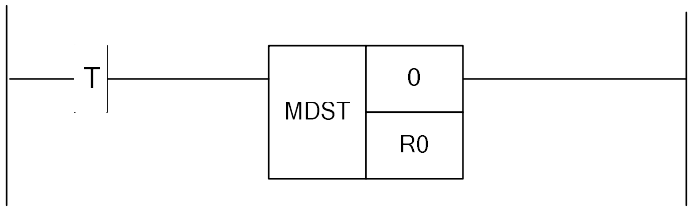
Function Set the working mode in the current channel (Auto, Single-block, JOG, Increment, Reference home, Handwheel, PMC)

Parameter

Working mode / Parameter	Auto	Single-block	JOG	Increm-ent	Home	Hand wheel	PMC
D2□□□	1	2	4	8	16	32	64

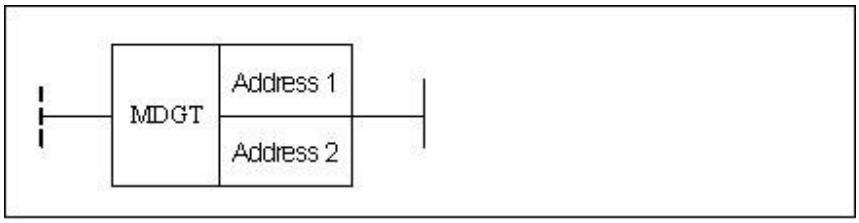
Supplementary note If the axis homing is the status in channel, do not switch mode.

Example

Ladder Diagram	
Statement List	MDST 0 R0
Description	Set the working mode in channel 0 based on the value of R0.

6.1.2 Channel Mode Getting MDGT

Format



Parameter	Parameter form	Data type	Storage area	Explanation	Properties
<Address 1>	□□□□	INT	Constant	Channel No.	Pre ○
<Address 2>	□□□□	INT	Constant, F, G, R, W, D, P, B	Working mode value	Post ×

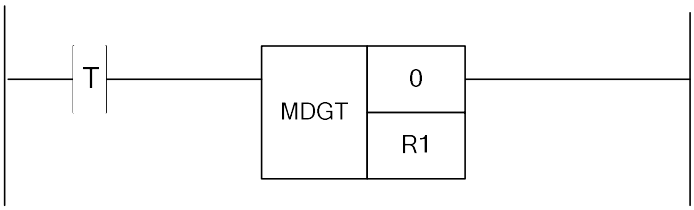
Function To get the working mode value of the current channel.

Parameter

Work mode / Parameter	Auto	Single-block	JOG	Incre-ment	home	Hand wheel	PMC
D2□□□	1	2	4	8	16	32	64

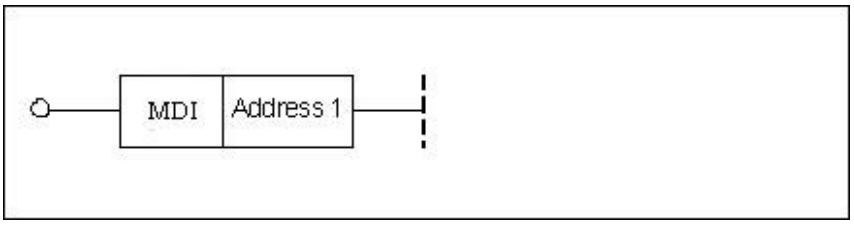
Supplementary note If the axis homing is the status in channel, do not switch mode.

Example

Ladder Diagram	
Statement List	MDGT 0 R1
Description	Take the working mode in Channel 0 into R1 register.

6.1.3 Mode MDI

Format

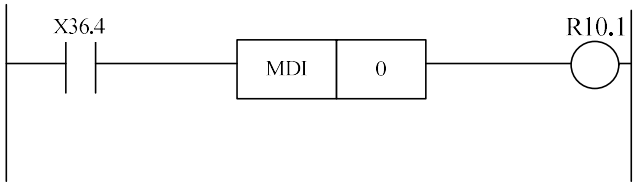


Parameter	Parameter form	Data type	Storage area	Explanation	Properties
<Address 1>	□□□□	INT	Constant	Channel No.	Pre ○ Post ×

Function To get MDI mode in the channel.

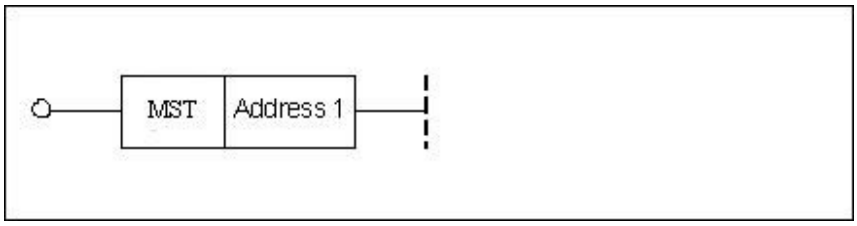
Parameter Parameter 1: channel No.

Example

Ladder Diagram	
Statement List	MDI 0
Description	When X36.4 is turned on, channel 0 is in MDI mode.

6.1.4 Locking Channel MST

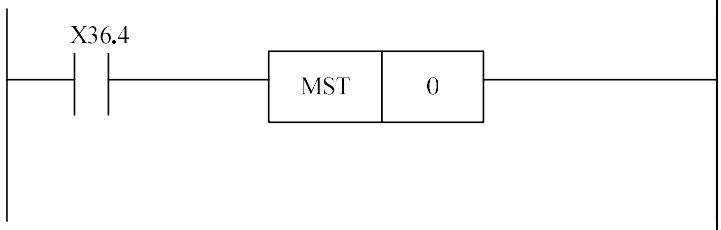
Format



Parameter	Parameter form	Data type	Storage area	Explanation	Properties
<Address 1>	□□□□	INT	Constant	Channel No.	Pre √ Post ×

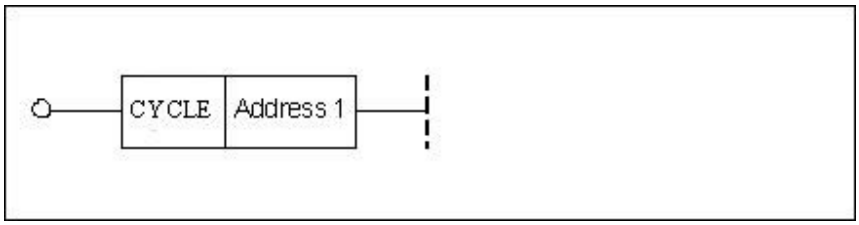
Function Lock the channel MST. When this function is turned on, all MST instructions in this channel are not available, and are directly skipped.

Example

Ladder Diagram	
Statement List	MST 0
Description	When X36.4 is turned on, channel 0 is locked.

6.1.5 Cycle Start CYCLE

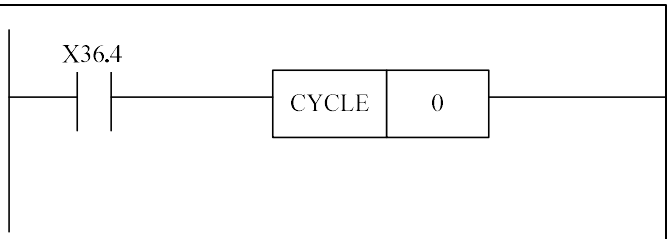
Format



Parameter	Parameter form	Data type	Storage area	Explanation	Properties
<Address 1>	□□□□	INT	Constant	Channel No.	Pre √ Post ×

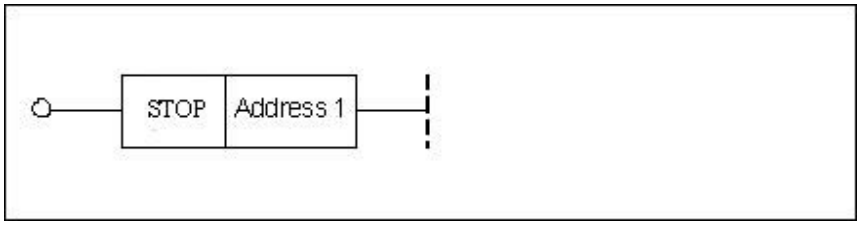
Function Set the channel which needs cycle start by parameter, and perform cycle start via ACT signal.

Example

Ladder Diagram	
Statement List	CYCLE 0
Description	When X36.4 is turned on, set the channel 0 to cycle start.

6.1.6 Emergency Stop STOP

Format

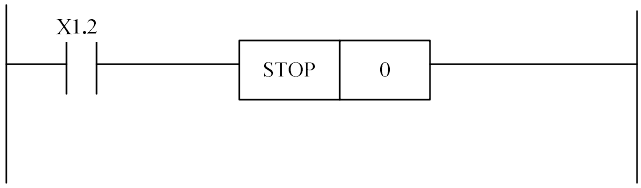


Parameter	Parameter form	Data type	Storage area	Explanation	Properties
<Address 1>	□□□□	INT	Constant	Channel No.	Pre √ Post ×

Function Set the channel which needs emergency stop by parameter, and start emergency stop via ACT signal.

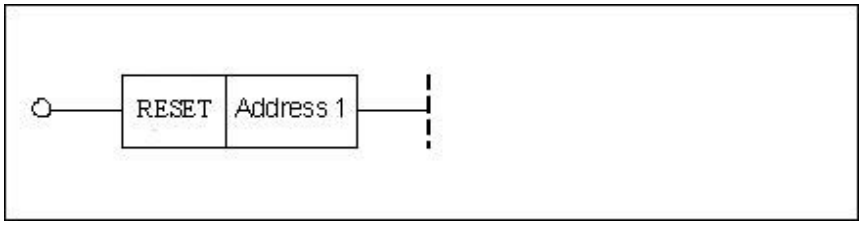
Example

Reset

Ladder Diagram	
Statement List	LD X1.2 STOP 0
Description	When X1.2 is turned on, set the channel 0 to emergency stop.

6.1.7 RESET

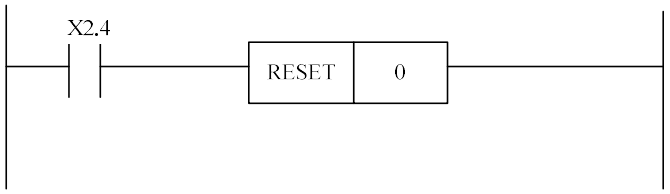
Format



Parameter	Parameter form	Data type	Storage area	Explanation	Properties
<Address 1>	□□□□	INT	Constant	Channel No.	Pre √ Post ×

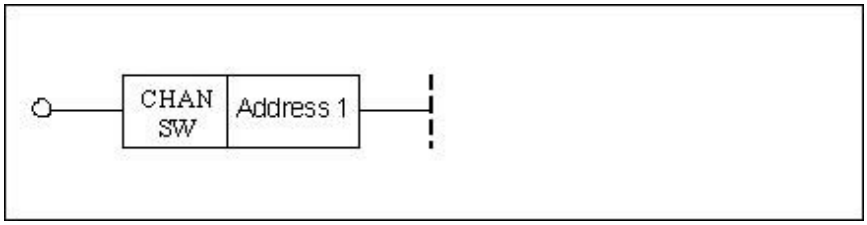
Function Set the channel which needs reset by parameter, and activate reset via ACT signal.

Example

Ladder Diagram	
Statement List	LD X2.4 RESET 0
Description	When X2.4 is turned on, the channel 0 is set to reset.

6.1.8 Channel Exchange CHANSW

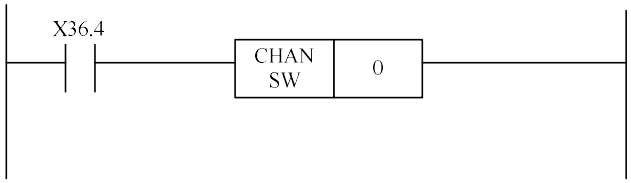
Format



Parameter	Parameter form	Data type	Storage area	Explanation	Properties
<Address 1>	□□□□	INT	Constant	Set the channel of feedhold	Pre √ Post ×

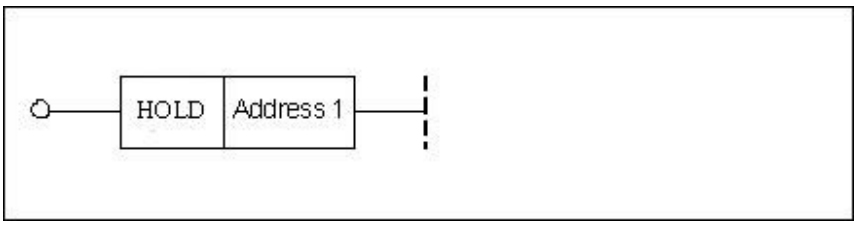
Function Set the channel which needs to be exchanged by parameter, and enable the channel exchange via ACT signal.

Example

Ladder Diagram	
Statement List	LD X36.4 CHANSW 0
Description	When X36.4 is turned on, set the channel 0 to the active channel.

6.1.9 Feed Hold Start HOLD

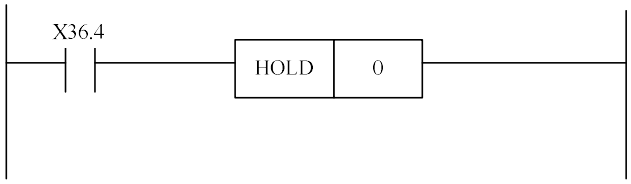
Format



Parameter	Parameter form	Data type	Storage area	Explanation	Properties
<Address 1>	□□□□	INT	Constant	Set the channel of feedhold	Pre √ Post ×

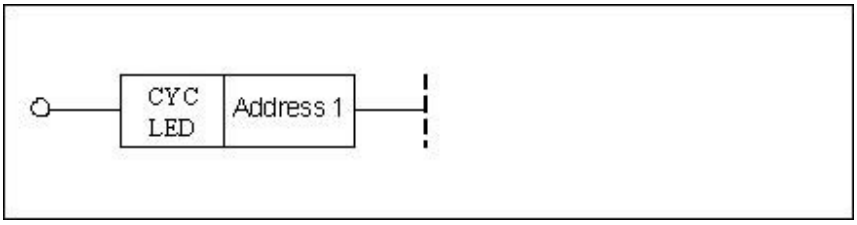
Function Set the channel which needs feed hold by parameter, and enable feed hold through ACT signal.

Example

Ladder Diagram	
Statement List	LD X36.4 HOLD 0
Description	When X36.4 is turned on, set the channel 0 to feed hold.

6.1.10 Cycle Start LED CYCLED

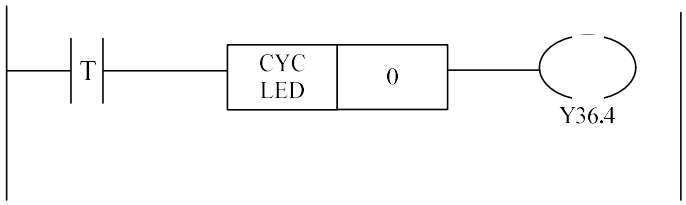
Format



Parameter	Parameter form	Data type	Storage area	Explanation	Properties
<Address 1>	□□□□	INT	Constant	Obtain the channel of cycle start	Pre ○ Post ×

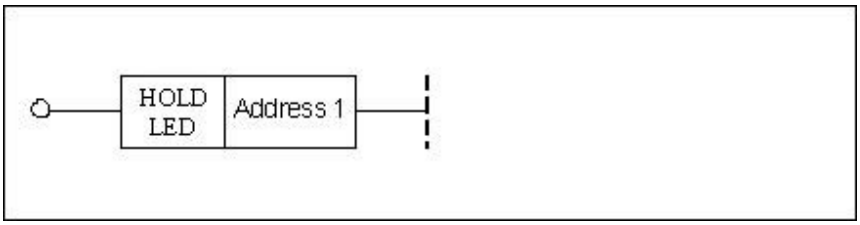
Function Set the channel where needs feed hold via parameter. If the cycle start is successful, the output will light up the cycle start light

Example

Ladder Diagram	
Statement List	LDT CYCLED 0 OUT Y36.4
Description	Get the cycle start state of channel 0, and the cycle start LED is lit.

6.1.11 Feed Hold LED HOLDLED

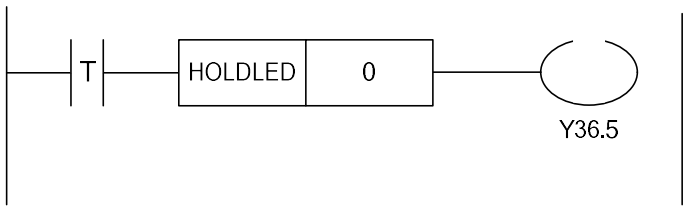
Format



Parameter	Parameter form	Data type	Storage area	Explanation	Properties
<Address 1>	□□□□	INT	Constant	*To get the channel of feedhold state.*	Pre ○ Post √

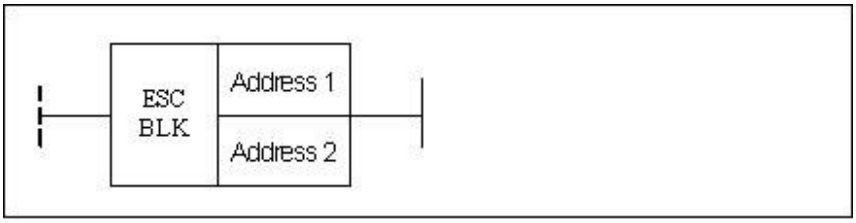
Function Set the channel where the feed hold LED needs to be lit by parameter, and light feed hold LED through ACT signal.

Example

Ladder Diagram	
Statement List	LDT HOLDLED 0 OUT Y36.5
Description	Control the feed hold LED based on the feed hold status in the channel 0.

6.1.12 Block Skip（G31）ESCBLK

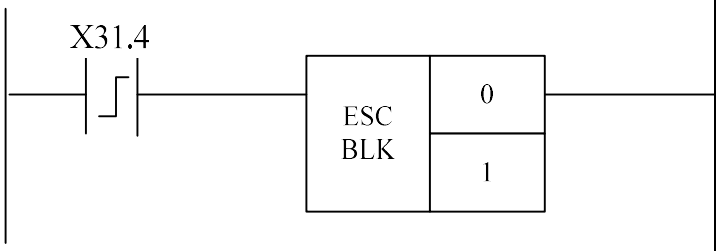
Format



Parameter	Parameter form	Data type	Storage area	Explanation	Properties
<Address 1>	□□□□	INT	Constant	The channel where the block skip function needs to be activated.	Pre ○ Post ×
<Address 2>	□□□□	INT	Constant	The number of G31	

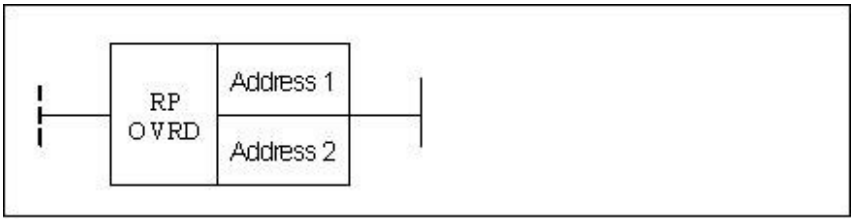
Function Set the channel where the block skip function needs to be activated by parameter, and enable this function through ACT signal.

Example

Ladder Diagram	
Statement List	LDP X31.4 ESCBLK 0 1
Description	When the rising edge of X31.4 is turned on, the first G31 statement (G31.1) in the channel 0 is activated.

6.1.13 Rapid Traverse Override RPOVRD

Format

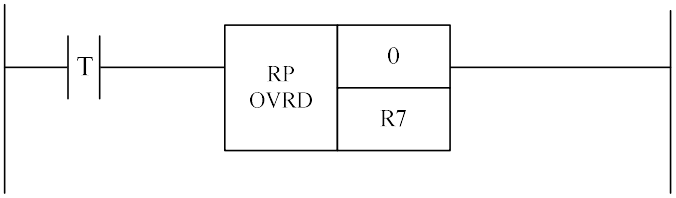


Parameter	Parameter form	Data type	Storage area	Explanation	Properties
<Address 1>	□□□□	INT	Constant	Channel No.	Pre ○ Post ×
<Address 2>	□□□□	INT	Constant, X, Y, F, G, R, W, D, P, B	Override value	

Function The channel is selected by parameter 1, override value is passed via register by parameter 2, and this function is enabled through ACT signal.

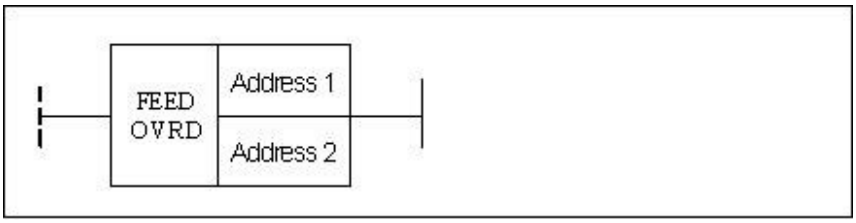
Supplementary note The override value cannot be changed at the time of thread-cutting.

Example

Ladder Diagram	
Statement List	LDT RPOVRD 0 R7
Description	Set the rapid override in the channel 0 by the value of R7

6.1.14 Feedrate Override FEEDOVRD

Format

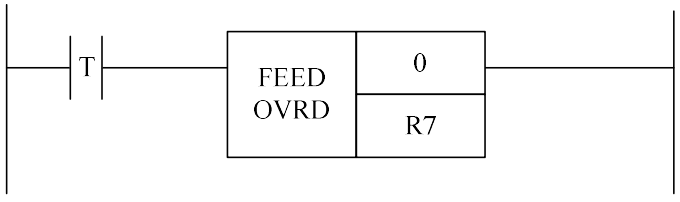


Parameter	Parameter form	Data type	Storage area	Explanation	Properties
<Address 1>	□□□□	INT	Constant	Channel No.	Pre ○ Post ×
<Address 2>	□□□□	INT	Constant, X, Y, F, G, R, W, D, P, B	Override value	

Function The channel is selected by parameter 1, override value is passed via register by parameter 2, and this function is enabled through ACT signal.

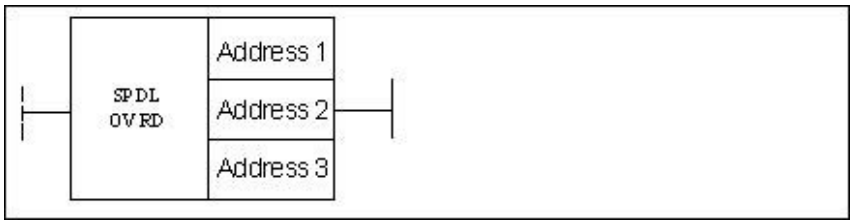
Supplementary Note The override value cannot be changed at the time of thread-cutting.

Example

Ladder Diagram	
Statement List	LDT FEEDOVRD 0 R7
Description	Set the feedrate override in the channel 0 with the value of R7.

6.1.15 Spindle Override SPDLOVRD

Format

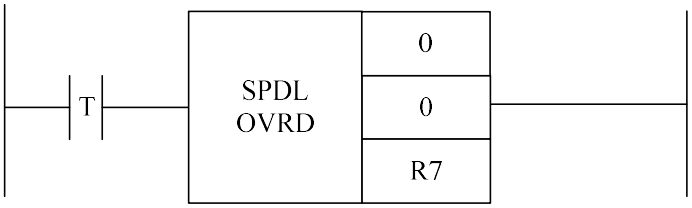


Parameter	Parameter form	Data type	Storage area	Explanation	Properties
<Address 1>	□□□□	INT	Constant	Channel No.	Pre ○ Post ×
<Address 2>	□□□□	INT	Constant	Spindle No.	
<Address 3>	□□□□	INT	Constant, X, Y, F, G, R, W, D, P, B	Override value	

Function The channel is selected by parameter 1, spindle No. is selected by parameter 2, and the override value is passed by parameter 3 via register. The federate override function is enabled by ACT.

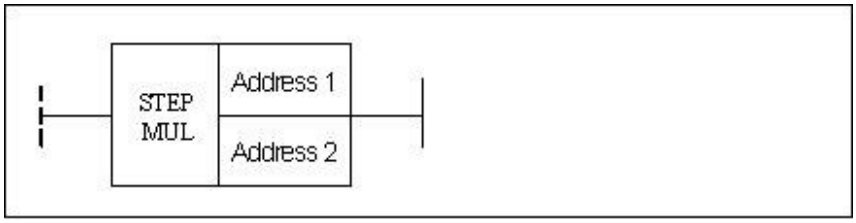
Supplementary note The override value cannot be changed at the thread-cutting time.

Example

Ladder Diagram	
Statement List	LDT SPDLOVRD 0 0 R7
Description	Set the No. 0 spindle override value in the channel 0 by the value of R7.

6.1.16 Incremental (Stepping) Magnification STEPMUL

Format

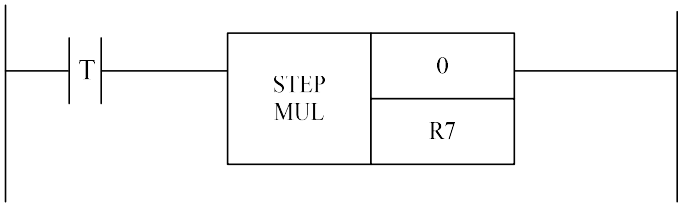


Parameter	Parameter form	Data type	Storage area	Explanation	Properties
<Address 1>	□□□□	INT	Constant	Axis No.	Pre ○ Post ×
< Address 2>	□□□□	INT	Constant, X, Y, F, G, R, W, D, P, B	Magnificati on value	

Function The axis number is set by the parameter 1, the magnification value is passed by the parameter 2 via register, and ACT enables feedrate override function.

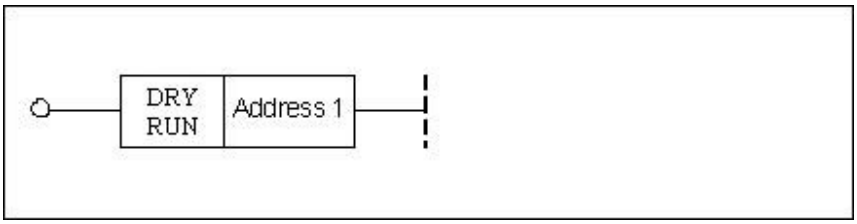
Supplementary note This function can only be used in the incremental (stepping) status.

Example

Ladder Diagram	
Statement List	LDT STEPMUL 0 R7
Description	Set the incremental magnification in the channel 0 by the R7 value.

6.1.17 Dryrun DRYRUN

Format



Parameter	Parameter form	Data type	Storage area	Properties
<Address 1>	□□□□	INT	Constant	Pre √ Post ×

Function

Press Dryrun button in the auto mode on the control panel, its indicator lamp is lit, and CNC is in dryrun status, where the feed rate specified by the program is ignored, and the coordinate axis moves at the maximum rapid traverse speed.

The actual cutting cannot be performed in the dryrun model. The purpose is to confirm the cutting path and the program.

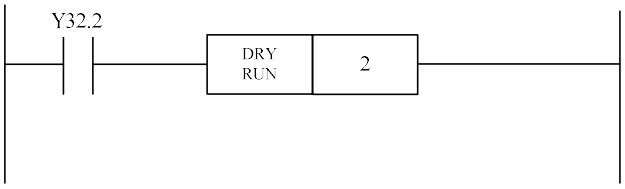
During the process of actual cutting, this function must be turned off; otherwise, it may cause danger.

This function cannot work on thread cutting.

Parameter

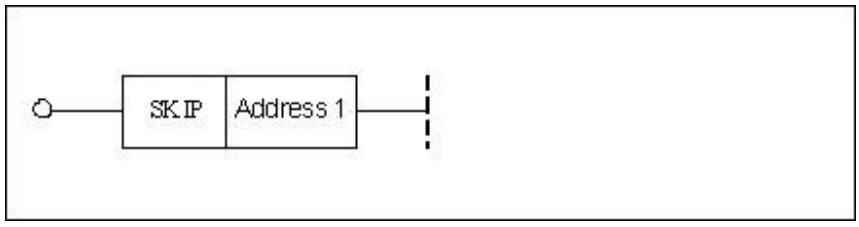
Parameter 1: channel No.

Example

Ladder Diagram	
Statement List	DRYRUN 2
Description	When Y32.2 is turned on, the channel 2 is in dry run.

6.1.18 Block Skip SKIP

Format

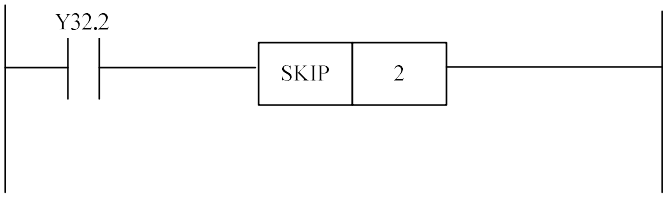


Parameter	Parameter form	Data type	Storage area	Properties
<Address 1>	□□□□	INT	Constant	Pre √
				Post ×

Function The system can skip some specified blocks in the auto mode. If "/" is put at the start of a program block, press Block skip, then this block will be skipped and not be executed in the auto mode; if Block skip is released, "/" will not work, and this block will be implemented.

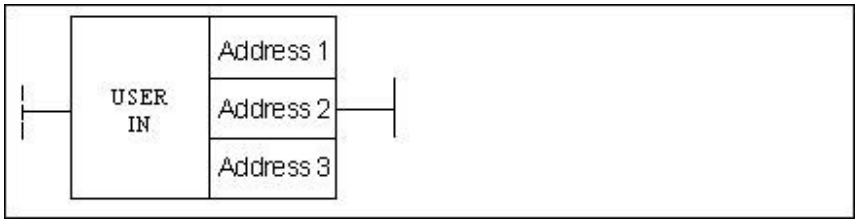
Parameter Parameter 1: channel No.

Example

Ladder Diagram	
Statement List	SKIP 2
Description	When Y32.2 is turned on, block-skip in channel 2 is valid.

6.1.19 User Input USERIN

Format



Parameter	Parameter form	Data type	Storage area	Explanation	Properties
<Address 1>	□□□□	INT	Constant		Pre √
<Address 2>	□□□□	INT	Constant		Post ×
<Address 3>	□□□□	INT	Constant		

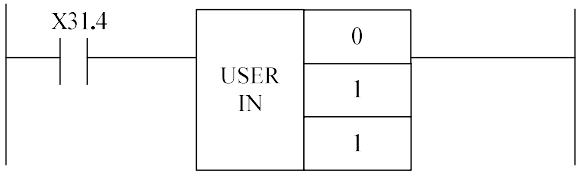
Function Set the user input. When ACT is effective, set user-defined group and bit in the channel to 1, and the macro-variable changes accordingly.

Parameter Parameter 1: channel No.

Parameter 2: not available at present

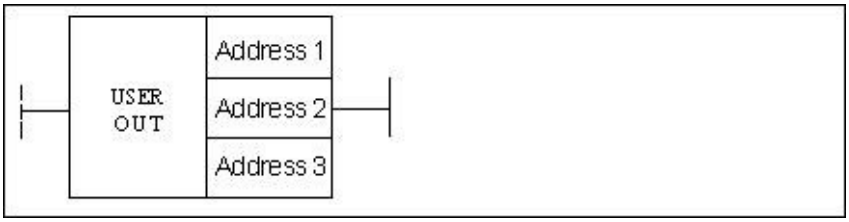
Parameter 3: power of 2. For example, 17 means that the value of #1190 is 2^17=131072.

Example

Ladder Diagram	
Statement List	USERIN 0 1 1
Description	When X31.4 is turned on, the macro-variable #1190 of user input group which corresponds to the channel 0 is 2.

6.1.20 User Output USEROUT

Format



Parameter	Parameter form	Data type	Storage area	Properties
<Address 1>	□□□□	INT	Constant	Pre ○
<Address 2>	None	None	None	Pre ×
<Address 3>	□□□□	INT	Y, R	

Function

Set the user output. Set the value of macro-variable #1191in the program, which will determine the group number and position number of user-defined output. 32-bit output is defined, and four groups of 8-bit outputs are obtained. The start address of output is defined by parameter 3.

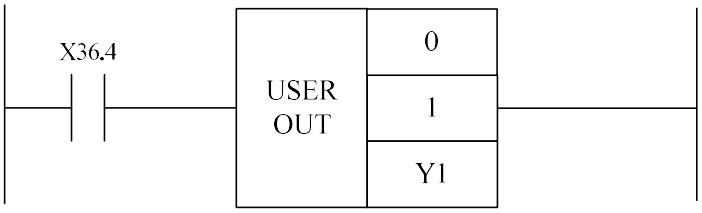
Parameter

Parameter 1: channel No.

Parameter 2: not available at present.

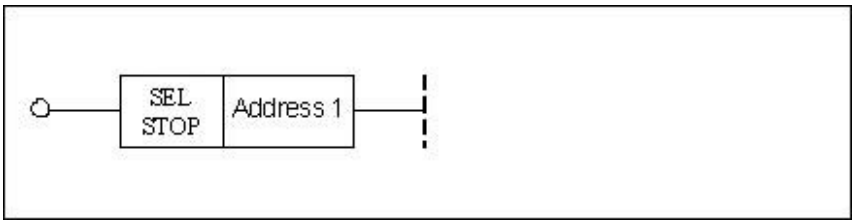
Parameter 3: the start address of output register. The output value is 32-bit. Therefore, for y register of 8-bit, four consecutive y registers are used.

Example

Ladder Diagram	
Statement List	USEROUT 0 1 Y1
Description	Y1.2 and Y1.3 will be output, and other bits of Y1 to Y4 are 0.

6.1.21 Optional Stop SELSTOP

Format

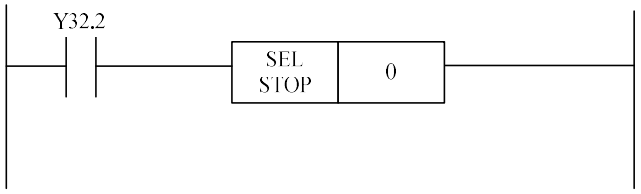


Parameter	Parameter form	Data type	Storage area	Explanation	Properties
<Address 1>	□□□□	INT	Constant		Pre √ Post ×

Function When "optional stop" is enabled (the indicator light is on), the program stops at M01 in the auto mode.

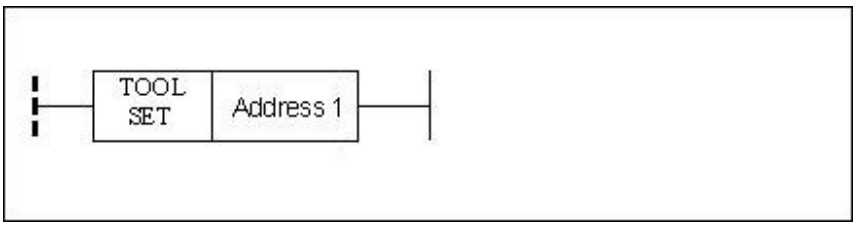
Parameter Parameter 1: channel No.

Example

Ladder Diagram	
Statement List	SELSTOP 0
Description	When Y32.2 is turned on, the optional stop in channel 0 is effective.

6.1.22 Vector Tool Direction Setting TOOLSET

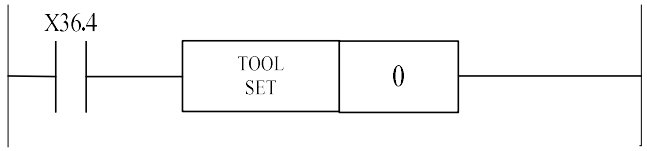
Format



Parameter	Parameter form	Data type	Storage area	Explanation	Properties
<Address 1>	□□□□	INT	Constant	Channel No.	Pre ×
					Post √

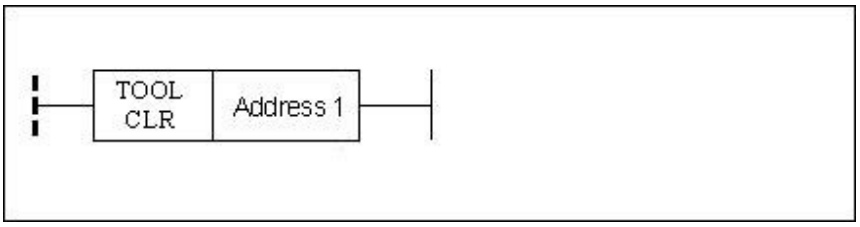
Function This function is generally used for 5-axis machining. Set the vector direction of the current tool in this channel to Z direction, enable this function, manually feed and retract the tool along the vector direction.

Example

Ladder Diagram	
Statement List	TOOLSET 0
Description	When Y32.2 is turned on, the tool direction in channel 0 is effective.

6.1.23 Vector Tool Direction Clear TOOLCLR

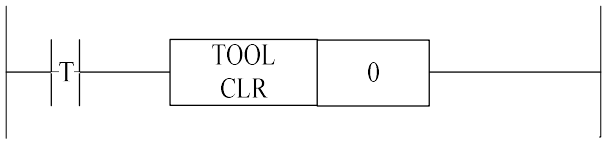
Format



Parameter	Parameter form	Data type	Storage area	Explanation	Properties
<Address 1>	□□□□	INT	Constant	Channel No.	Pre ×
					Post √

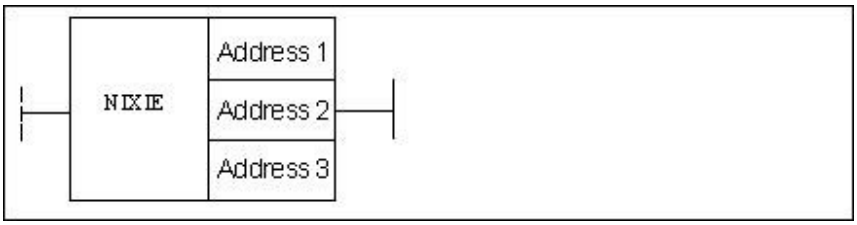
Function This function is generally used for 5-axis machining. In this channel, Z direction is cancelled as the vector direction of the current tool. This function is used in conjunction with TOOLSET function.

Example

Ladder Diagram	
Statement List	TOOLCLR 0
Description	The tool direction in channel 0 is set to be invalid.

6.1.24 8-bit Nixie Tube NIXIE

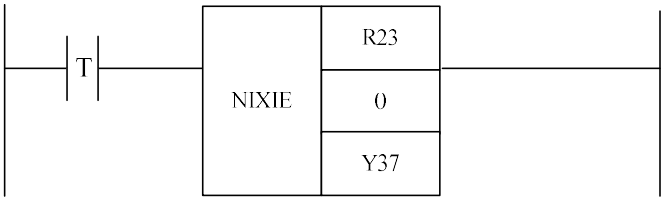
Format



Parameter	Parameter form	Data type	Storage area	Explanation	Properties
<Address 1>	□□□□	INT	Constant, X, Y, F, G, R, W, D, P, B	The number to be shown by the digital tube	Pre √
<Address 2>	□□□□	INT	Constant	”0” indicates single byte, and “1” indicates double-byte.	Post √
<Address 3>	□□□□	BOOL	Y, R, W, D, B	Set the 8-bit digital tube on the panel	

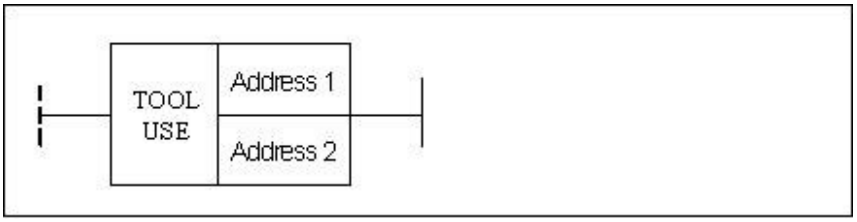
Function The 8-bit digital tube on the panel displays the number of the current tool.

Example

Ladder Diagram	
Statement List	NIXIE R23 0 Y37
Description	The tool number in R23 is displayed to the digital tube.

6.1.25 Tool Display TOOLUSE

Format



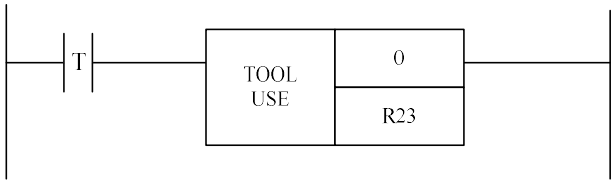
Parameter	Parameter form	Data type	Storage area	Properties
<Address 1>	□□□□	INT	Constant, X, Y, F, G, R, W, D, P, B	Pre ○
<Address 2>	□□□□	INT	Constant, X, Y, F, G, R, W, D, P, B	Post ×

Function Display the tool number in the currently executed T code to the interface of CNC.

Parameter Parameter 1: channel number

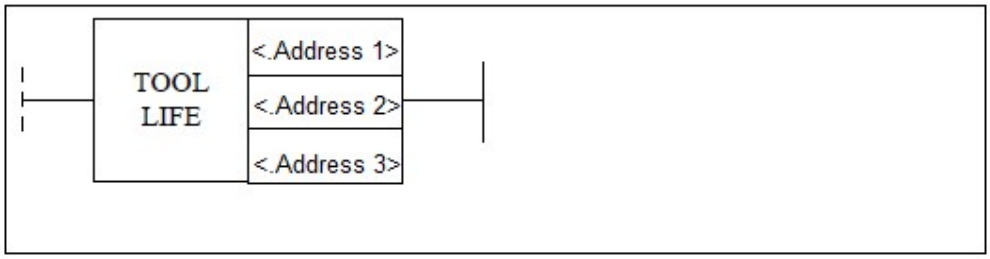
Parameter: tool number

Example

Ladder Diagram	
Statement List	TOOLUSE 0 R23
Description	The tool number in channel 0 is displayed on the interface.

6.1.26 Tool Life TOOLLIFE

Format



Parameter	Parameter form	Data type	Storage area	Explanation	Properties
<Address 1>	□□□□	INT	Constant (0-3)	In the channel of <address 1>, when the times that the tool is installed reaches the times of <address 2>, the register of <address 3> responses once.	Pre ○
<Address 2>	□□□□	INT	Constant		
<Address 3>	□□□□	INT	R register		Post ×

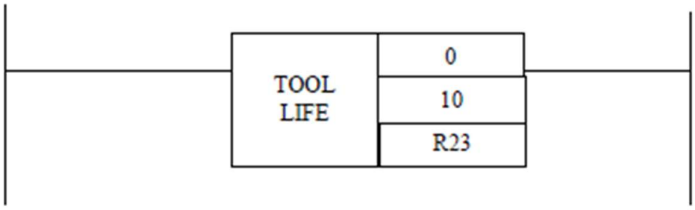
Function Cumulative tool installation times

Parameter Parameter 1: channel number

Parameter 2: number of installations

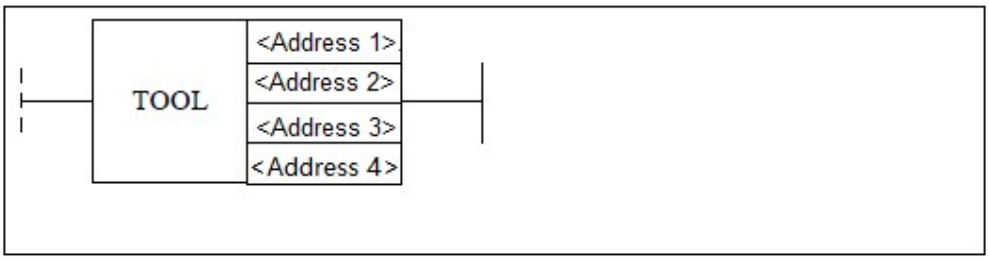
Parameter 3: counting response point

Example

Ladder Diagram	
Statement List	TOOLIFE 0 10 R23
Description	When the times that the tool is installed in channel 0 reaches 10 in channel 0, R23 responses once.

6.1.27 Tool Selection Module TOOL

Format



Parameter	Parameter form	Data type	Storage area	Explanation	Properties
<Address 1>	□□□□	INT	Constant	It is turned on after the target tool position number in <address 4> is found; otherwise, it is not turned on.	Pre ○
<Address 2>	□□□□	INT	Constant		
<Address 3>	□□□□	INT	Constant		Post ×
<Address 4>	□□□□	INT	Constant		

Function It is turned on after the target tool position number in <address 4> is found; otherwise, it is not turned on. (The relevant parameters on magazine interface need to be set before running.)

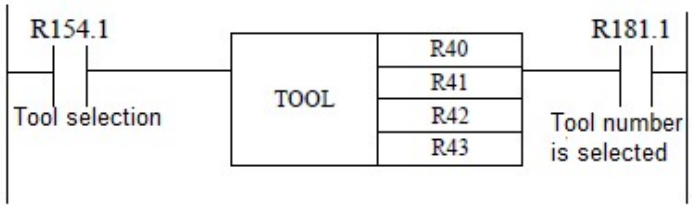
Parameter Parameter 1: spindle tool number

Parameter 2: specified tool number

Parameter 3: alarm number

Parameter 4: target tool position number

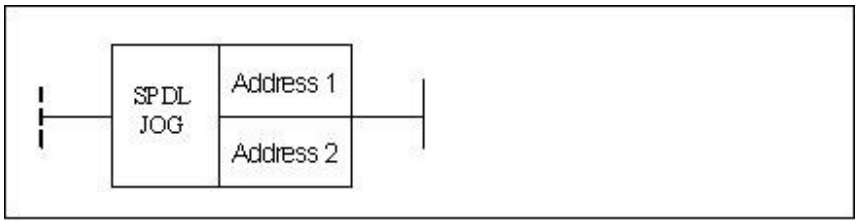
Example

Ladder Diagram	
Statement List	TOO R40 R41 R42 R43
Description	When tool selection R154.1 is turned on, if the target number R43 is chosen, R181.1 is turn on.

6.2 Functional Unit of Axis

6.2.1 Spindle JOG SPDLJOG

Format



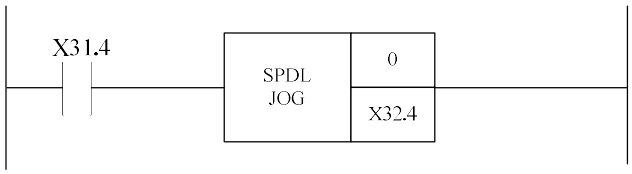
Parameter	Parameter form	Data type	Storage area	Properties
<Address 1>	□□□□	INT	Constant	Pre ○
<Address 2>	□□□□	BOOL	X, Y, F, G, R, W, D, P, B	Post ×

Function Control spindle manually.

Parameter Parameter 1: spindle number

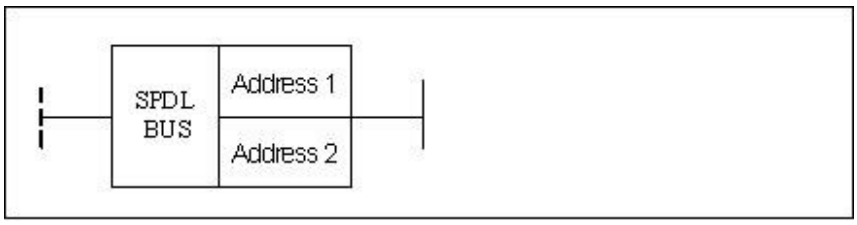
Parameter 2：point of CW rotation

Example

Ladder Diagram	
Statement List	SPDLJOG 0 X32.4
Description	When X31.4 is effective, if X32.4 is valid, the 0 axis will rotate at the default speed in the clockwise direction.

6.2.2 Spindle Control【Servo Spindle】SPDLBUS

Format



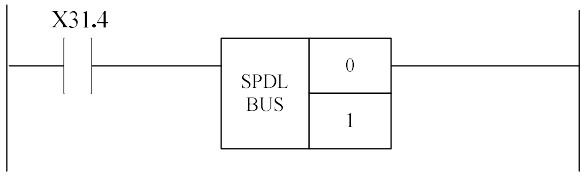
Parameter	Parameter form	Data type	Storage area	Explanation	Properties
<Address 1>	□□□□	INT	Constant		Pre ○
<Address 2>	□□□□	INT	Constant		Post ×

Function A spindle in a channel is set to be valid. Set the device which is set to be associated with the spindle number by the channel parameter to the spindle. For example, the logical axis number of the 0 axis in the current channel 0 is 5, (suppose No.5 axis is enabled), then the logical axis 5 is regarded as the first spindle in the current channel. The spindle is enabled to be effective by this functional module.

Parameter Parameter 1: channel number

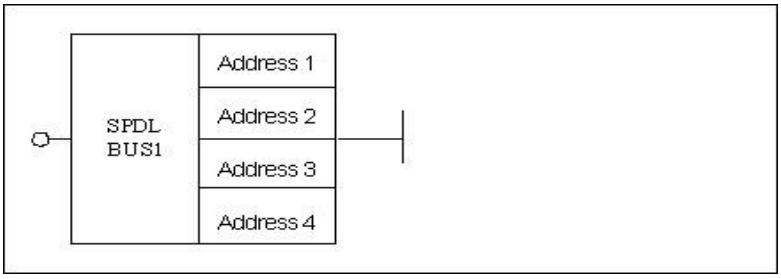
Parameter 2: spindle number

Example

Ladder Diagram	
Statement List	SPDLBUS 0 1
Description	When X31.4 is effective, the No.1 spindle in channel 0 is controlled.

6.2.3 Spindle Control with Gear【Servo Spindle】SPDLBUS1

Format



Parameter	Parameter form	Data type	Storage area	Properties
<Address 1>	□□□□	INT	Constant	Pre ○
<Address 2>	□□□□	INT	Constant	
<Address 3>	□□□□	INT	Constant, Y, G, R, W, D, B	
<Address 4>	□□□□	INT	P	

Function Bus type spindle control with gear.

Parameter Parameter 1: channel number.

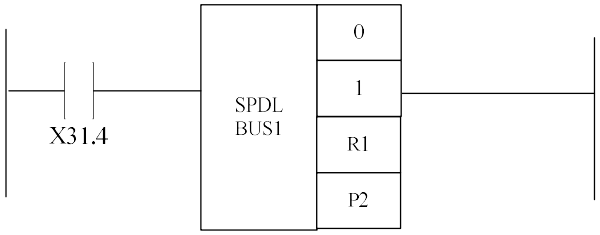
Parameter 2: spindle number.

Parameter 3: gear register, starting with 1.

Parameter 4: control parameter. The specified parameter holds the data such as maximum speed of spindle motor, initial speed and the like. Spindle control value for parameter 4 includes:

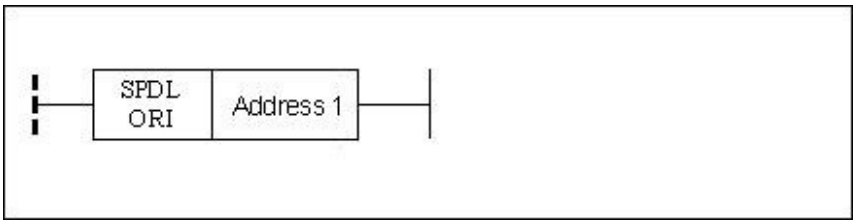
0	Maximum speed of motor
1	Minimum speed of actual measurement
2	Maximum speed of actual measurement
3	Numerator of current transmission ratio
4	Denominator of current transmission ratio

Example

Ladder Diagram	
Statement List	
Description	When X31.4 is effective, the current gear for No. 1 spindle override in channel 0 is in R1 register. Control parameter is in user parameters from P2. Refer to the Parameter Manual for filling in parameters, according to the actual condition of machine.

6.2.4 Spindle Orientation Enable SPDLORI

Format



Parameter	Parameter form	Data type	Storage area	Explanation	Properties
<Address 1>	□□□□	INT	Constant	Axis number	Pre ○
					Post ×

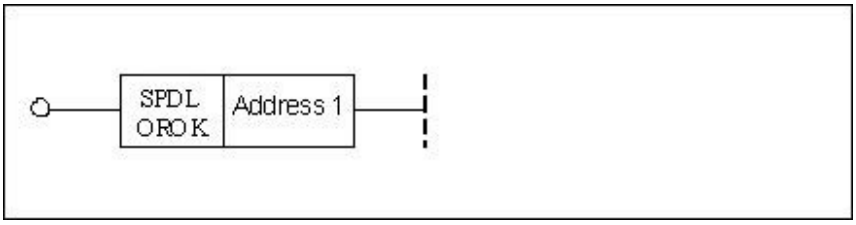
Function Spindle orientation enable. The spindle needs to be oriented to a specified angle at the beginning of tool changing and rigid tapping. Perform spindle orientation via this function. The orientation angle is set by the parameter in the servo drive.

Example

Ladder Diagram	
Statement List	SPDLORI 0
Description	The spindle orientation for No.0 spindle.

6.2.5 Spindle Orientation Completion SPDLOROK

Format



Parameter	Parameter form	Data type	Storage area	Properties
<Address 1>	□□□□	INT	Constant	Pre ○ Post √

Function The spindle orientation is completed, which indicates that the spindle has been at the specified orientation angle.

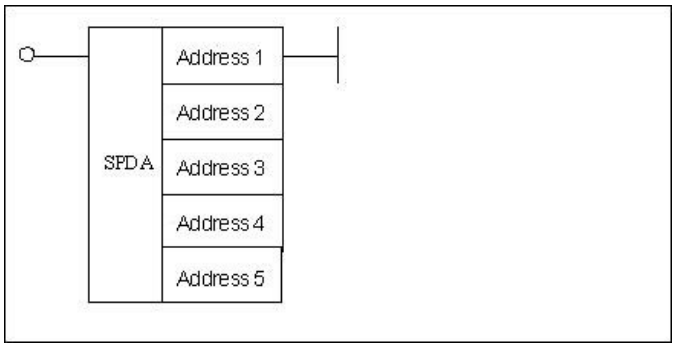
Parameter Parameter 1: axis number

Example

Ladder Diagram	
Statement List	SPDLOROK 0
Description	When No.0 spindle orientation has been enabled, set R10.1 to be effective.

6.2.6 Spindle Control【DA】SPDA

Format



Parameter	Parameter form	Data type	Storage area	Properties
<Address 1>	□□□□	INT	Constant	Pre ○
<Address 2>	□□□□	INT	Constant	Post ×
<Address 3>	□□□□	INT	Constant, Y, G, R, W, D, B	
<Address 4>	□□□□	BOOL	Y, G, R, W, D, B	
<Address 5>	□□□□	BOOL	P	

Function DA control of spindle. It is used to control the analog spindle.

Parameter Parameter 1: channel number

Parameter 2: spindle number

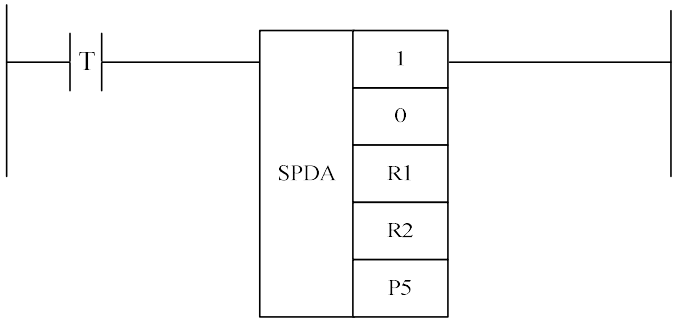
Parameter 3: gear register (gear starts from 1)

Parameter 4：invalid

Parameter 5： spindle control value includes:

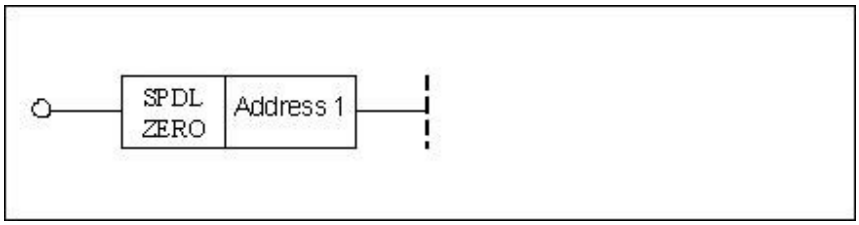
0	Maximum speed of motor
1	Minimum speed of actual measurement
2	Maximum speed of actual measurement
3	Numerator of current transmission ratio
4	Denominator of current transmission ratio

Example

Ladder Diagram	
Statement List	SPDA 1 0 R1 R2 P5
Description	Channel 1. The current gear for spindle 0 in channel 1 is in R1 register. The reference value of spindle control is in R2. Control parameter is in P5.

6.2.7 Zero Speed Detection for Spindle SPDLZERO

Format

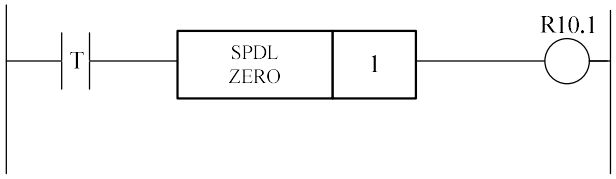


Parameter	Parameter form	Data type	Storage area	Properties
<Address 1>	□□□□	INT	Constant	Pre ○
				Post ×

Function Zero speed detection for spindle.

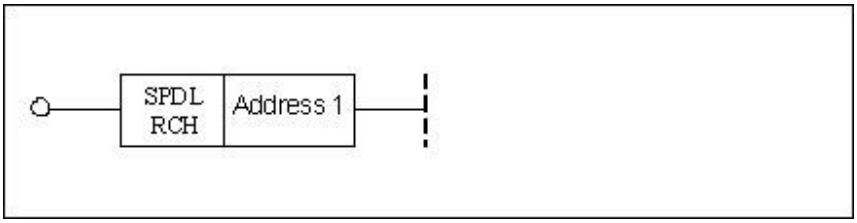
Parameter Parameter 1: axis number

Example

Ladder Diagram	
Statement List	SPDLZERO 1
Description	Zero speed detection for the No.1 spindle.

6.2.8 Spindle Speed Arrival SPDLRCH

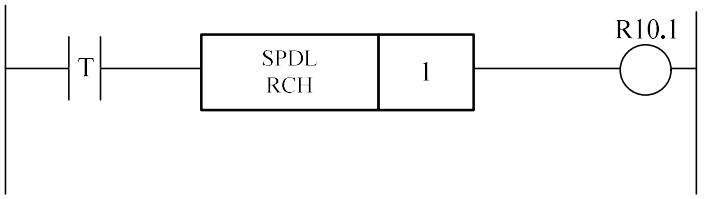
Format



Parameter	Parameter form	Data type	Storage area	Explanation	Properties
<Address 1>	□□□□	INT	Constant	Channel No.	Pre ○ Post ×

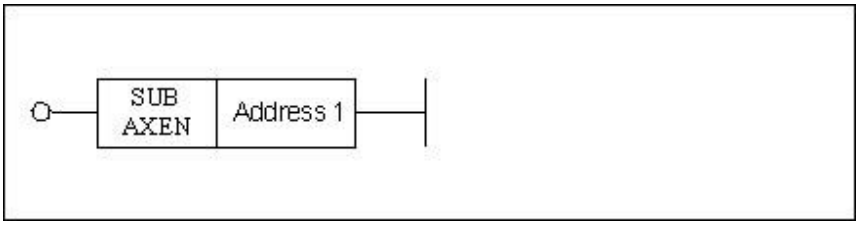
Function To detect whether the spindle speed reaches the command speed.

Example

Ladder Diagram	
Statement List	SPDLRCH 1
Description	Speed arrival of the No.1 spindle.

6.2.9 Slave Axis Home SUBAXEN

Format

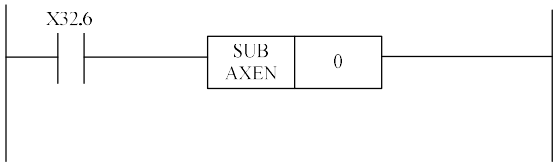


Parameter	Parameter form	Data type	Storage area	Explanation	Properties
<Address 1>	□□□□	INT	Constant	Channel number	Pre √ Post ×

Function Enable the slave axis to return to zero. When this function is turned on, the slave axis performs reference point return to search Z pulse. Z pulse has been found, which means the reference position return of slave axis is completed. Then the master axis continues to return to zero.

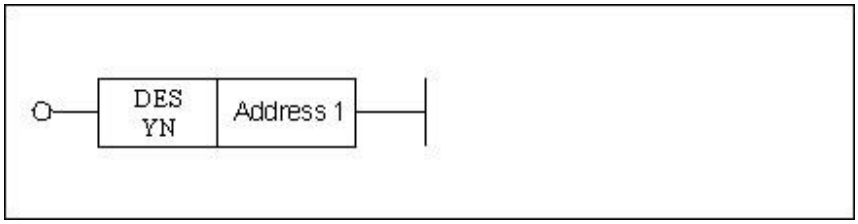
Parameter Parameter 1: slave axis number

Example

Ladder Diagram	
Statement List	SUBAXEN 0
Description	When X32.6 is valid, the slave axis is enabled to return to zero.

6.2.10 Release Slave Axis DESYN

Format

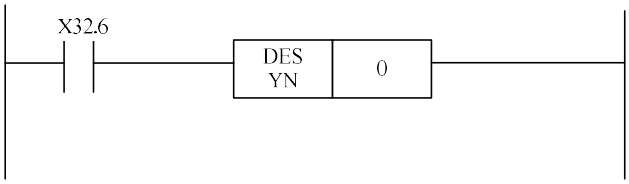


Parameter	Parameter form	Data type	Storage area	Explanation	Properties
<Address 1>	□□□□	INT	Constant		Pre √ Post ×

Function Release slave axis. An axis is set to the slave axis of another one by parameter.
If some instructions are sent to master axis, slave axis will also get those. When the function of slave axis release is turned on, the slave axis is disassociated with the master axis, and doesn't receive instruction pulse of the master axis.

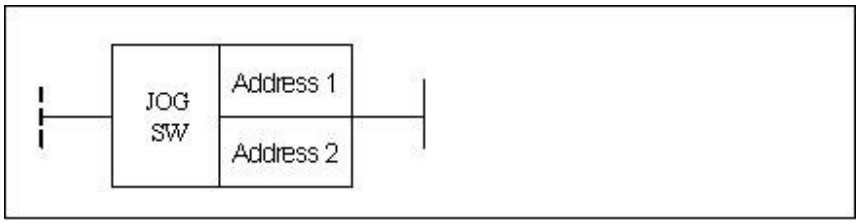
Parameter Parameter 1: slave axis number

Example

Ladder Diagram	
Statement List	DESYN 0
Description	When X32.6 is valid, the No.0 slave axis is released.

6.2.11 Axis Jog JOGSW

Format



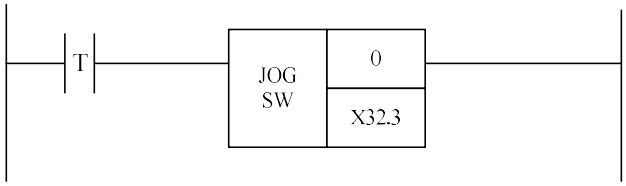
Parameter	Parameter form	Data type	Storage area	Properties
<Address 1>	□□□□	INT	Constant, X, Y, F, G, R, W, D, P, B	Pre ○
<Address 2>	□□□□	BOOL	X, Y, F, G, R, W, D, P, B	Post ×

Function Manually control the axis JOG Enable

Parameter Parameter 1: axis number

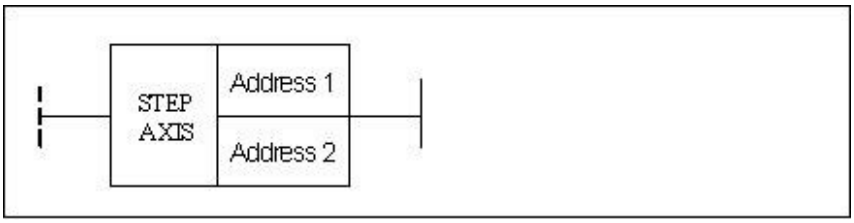
Parameter 2: positive JOG direction of axis. When 1 is set, it indicates the positive direction; when 0 is set, it indicates the negative direction.

Example

Ladder Diagram	
Statement List	JOGSW 0 X32.3
Description	When X32.3 is turned on, axis 0 is disabled to move manually in positive direction, and axis 0 is enabled to move manually in negative direction.

6.2.12 Axis Stepping STEPAXIS

Format



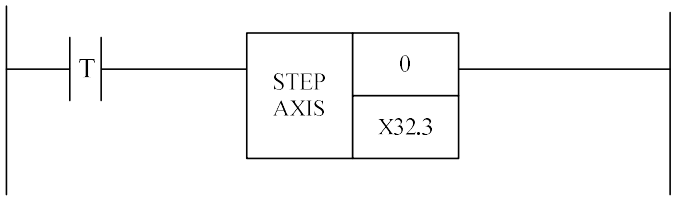
Parameter	Parameter form	Data type	Storage area	Properties
<Address 1>	□□□□	INT	Constant, X, Y, F, G, R, W, D, P, B	Pre ○
<Address 2>	□□□□	BOOL	X, Y, F, G, R, W, D, P, B	Post ×

Function Enable the stepping of stepping.

Parameter Parameter 1: axis number.

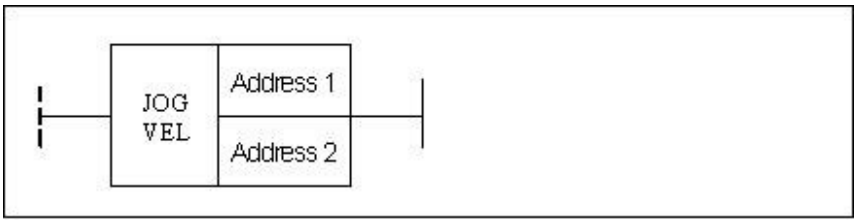
Parameter 2: the direction of axis stepping. "0" represents jog in the positive direction, and "1" represents jog in the negative direction.

Example

Ladder Diagram	
Statement List	STEPAXIS 0 X32.3
Description	When X32.3 is turned on, axis 0 is enabled to jog in negative direction; when X32.3 is not turned on, axis 0 is enabled to jog in positive direction.

6.2.13 Jog Velocity JOGVEL

Format



Parameter	Parameter form	Data type	Storage area	Properties
<Address 1>	□□□□	INT	Constant, X, Y, F, G, R, W, D, P, B	Pre √
<Address 2>	□□□□	INT	Constant, X, Y, F, G, R, W, D, P, B	Post ×

Function Manually control the jog speed.

Parameter Parameter 1: axis number

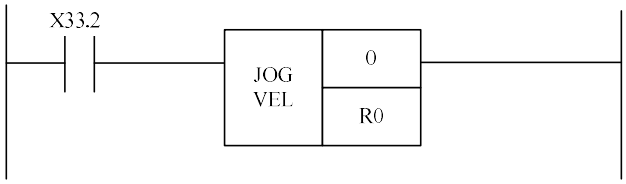
Parameter 2: axis speed, and its value can be as below:

1: Jog speed of axis

2: rapid traverse speed of axis

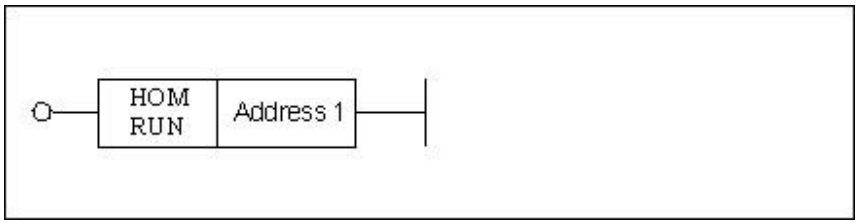
>2: speed (pulse/rev)

Example

Ladder Diagram	
Statement List	JOGVEL 0 R0
Description	When X33.2 is turned on, the 0 axis runs at the speed specified in R0.

6.2.14 Home Start HOMRUN

Format

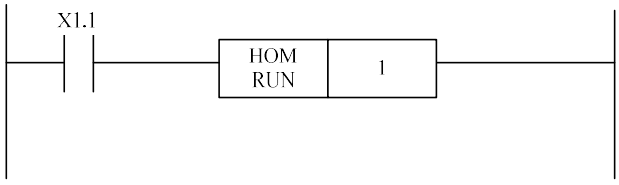


Parameter	Parameter form	Data type	Storage area	Explanation	Properties
<Address 1>	□□□□	INT	Constant		Pre √
					Post ×

Function To start the reference point return.

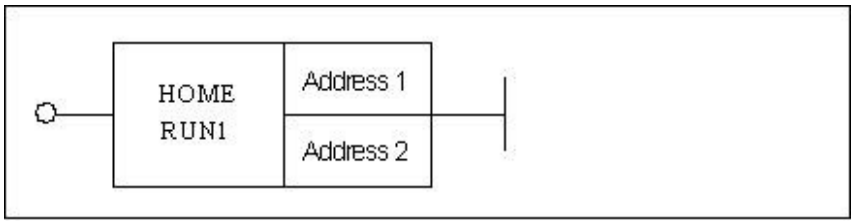
Parameter Parameter 1: axis number

Example

Ladder Diagram	
Statement List	HOMRUN 1
Description	When X1.1 is turned on, the reference position return of axis 1 starts.

6.2.15 Home Start 1 HOMERUN1

Format



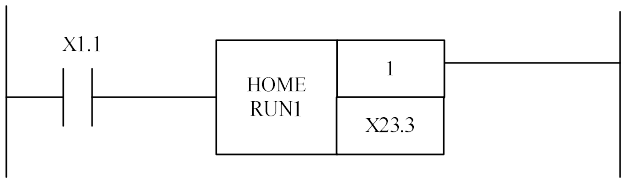
Parameter	Parameter form	Data type	Storage area	Properties
<Address 1>	□□□□	INT	Constant	Pre √
<Address 2>	□□□□	BOOL	X, Y, F, G, R, W, D, P, B	Post ×

Function To start the reference point return.

Parameter Parameter 1: axis number

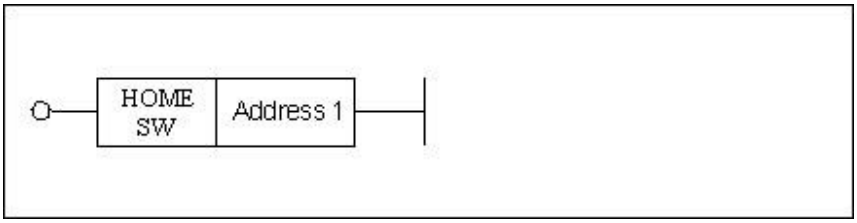
Parameter 2: the direction of reference point return.

Example

Ladder Diagram	
Statement List	HOMERUN1
Description	When X1.1 is turned on, the axis 1 starts to return to reference point; when X23.3 is turned on, the Jog of axis 0 in positive direction is enabled; when X23.3 is turned off, the jog of axis 0 in negative direction is enabled.

6.2.16 Home Approaching Switch HOMESW

Format

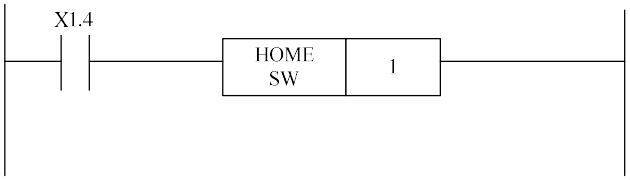


Parameter	Parameter form	Data type	Storage area	Explanation	Properties
<Address 1>	□□□□	INT	Constant		Pre √ Post ×

Function The axis meets the home block.

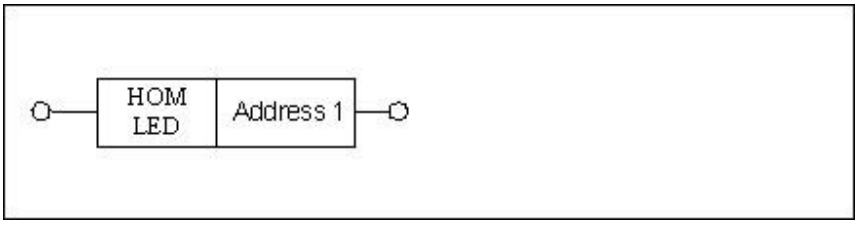
Parameter Parameter 1: axis

Example

Ladder Diagram	
Statement List	HOMESW 1
Description	X1.4 is turned on, which means the axis 1 meets the home block.

6.2.17 Homing Completion HOMLED

Format

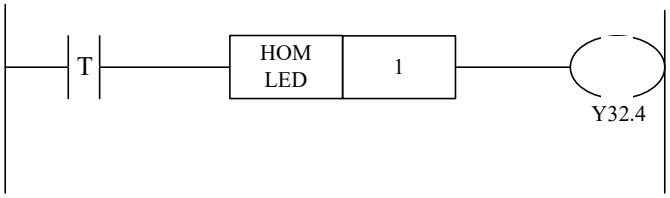


Parameter	Parameter form	Data type	Storage area	Explanation	Properties
<Address 1>	□□□□	INT	Constant		Pre √
					Post √

Function Homing is completed

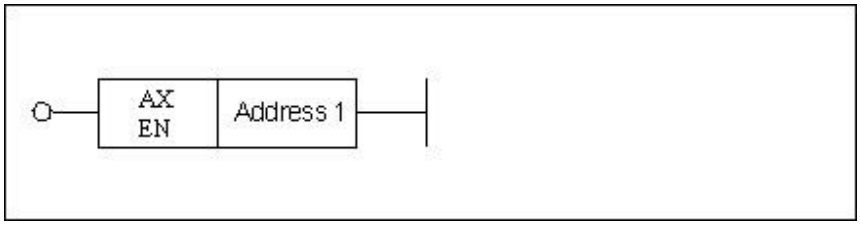
Parameter Parameter 1: axis number

Example

Ladder Diagram	
Statement List	HOMLED 1
Description	Axis 1 has completed homing, and the lamp of axis 1 is lit.

6.2.18 Axis Enable AXEN

Format

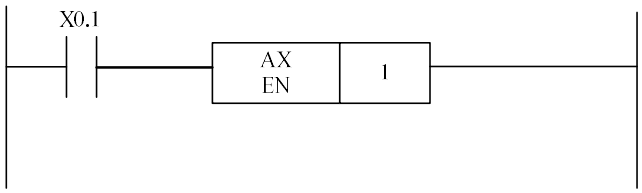


Parameter	Parameter form	Data type	Storage area	Properties
<Address 1>	□□□□	INT	Constant	Pre √ Post ×

Function Axis enable.

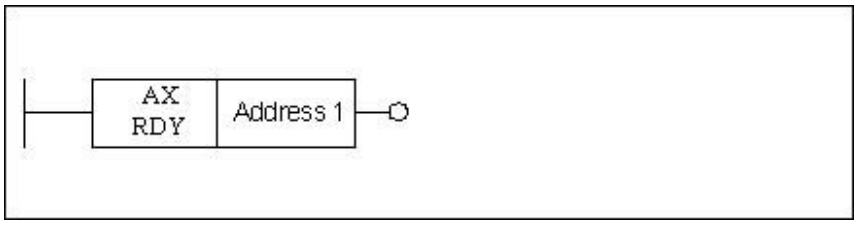
Parameter Parameter 1: axis number, can be constant and register.

Example

Ladder Diagram	
Statement List	AXEN 1
Description	When X0.1 is turned on, axis 1 is enabled.

6.2.19 Axis Ready (Bus) AXRDY

Format

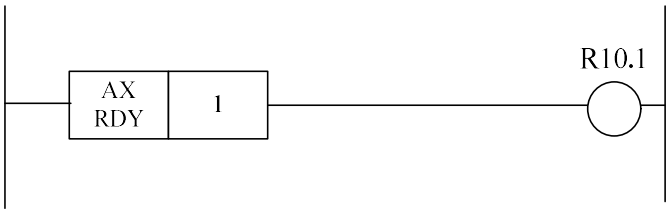


Parameter	Parameter form	Data type	Storage area	Properties
<Address 1>	□□□□	INT	Constant	Pre ×
				Post √

Function Axis is ready.

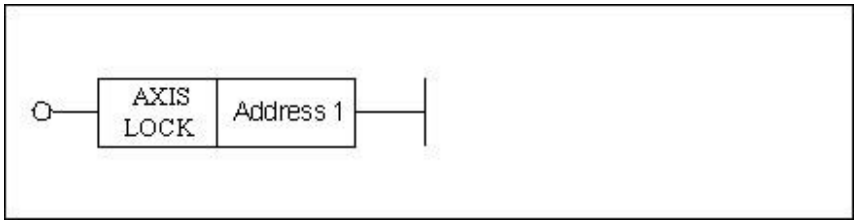
Parameter Parameter 1: axis number.

Example

Ladder Diagram	
Statement List	AXRDY 1
Description	When the axis 1 is ready, R10.1 is set to 1.

6.2.20 Axis Lock AXISLOCK

Format

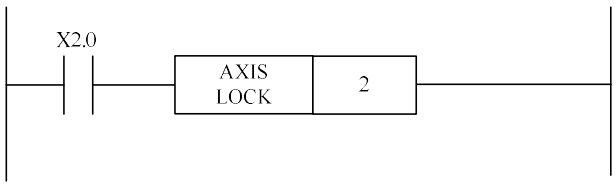


Parameter	Parameter form	Data type	Storage area	Properties
<Address 1>	□□□□	INT	Constant	Pre √ Post ×

Function The axis is locked.

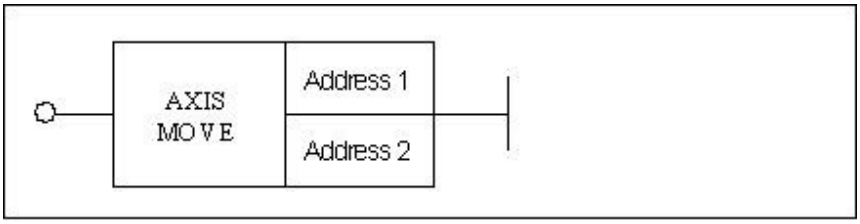
Parameter Parameter 1: axis number, can be constant and register.

Example

Ladder Diagram	
Statement List	AXISLOCK 2
Description	When X2.0 is turned on, the axis 2 is locked.

6.2.21 Relative PMC Axis Traverse AXISMOVE

Format



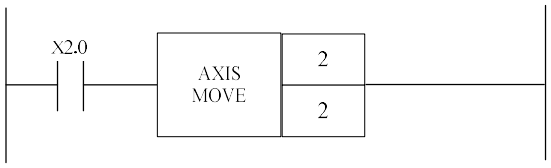
Parameter	Parameter form	Data type	Storage area	Properties
<Address 1>	□□□□	INT	Constant, X, Y, F, G, R, W, D, P, B	Pre √
<Address 2>	□□□□	INT	Constant, X, Y, F, G, R, W, D, P, B	Post ×

Function PMC axis is a special traverse axis, which cannot be moved by the command, and cannot be used for the interpolation. The PMC axis can only be moved by the PLC program. This instruction is used to move the PMC axis, and specify the relative moving distance.

Parameter Parameter 1: axis number,

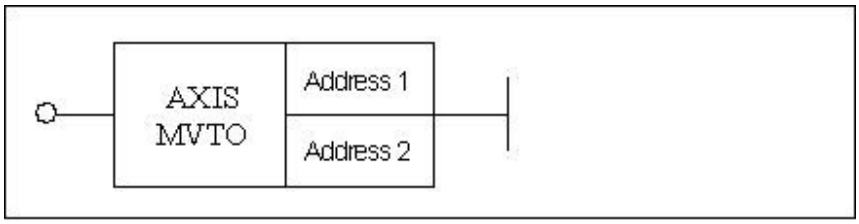
Parameter 2: axis movement amount（unit 1/1000mm, or 1/1000degree）.

Example

Ladder Diagram	
Statement List	AXISMOVE 2 2
Description	When X2.0 is turned on, axis 2 relatively moves t2 units of the distance.

6.2.22 Absolute PMC Axis Movement AXISMVTO

Format



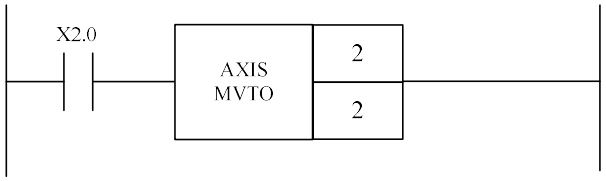
Parameter	Parameter form	Data type	Storage area	Properties
<Address 1>	□□□□	INT	Constant, X, Y, F, G, R, W, D, P, B	Pre √
<Address 2>	□□□□	INT	Constant, X, Y, F, G, R, W, D, P, B	Post ×

Function This instruction is used to move the PMC axis to an absolute position.

Parameter Parameter 1: axis number

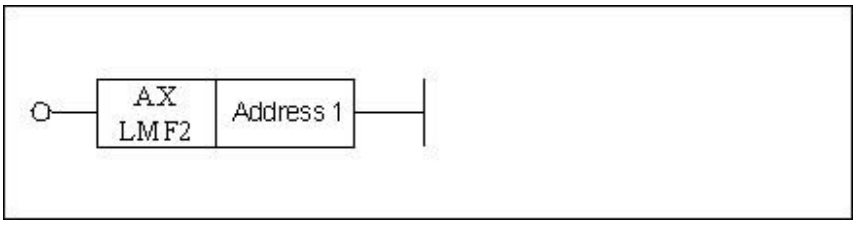
Parameter 2: the position that the axis moves (unit: 1/1000mm, or 1/1000 degree).

Example

Ladder Diagram	
Statement List	AXISMVTO 2 2
Description	When X2.0 is turned on, the axis 2 moves to the position 2.

6.2.23 The Second Soft Limit of Axis AXLMF2

Format

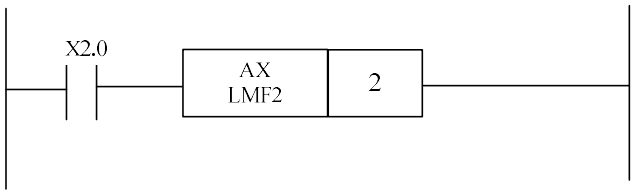


Parameter	Parameter form	Data type	Storage area	Properties
<Address 1>	□□□□	INT	Constant, X, Y, F, G, R, W, D, P, B	Pre √ Post ×

Function The second soft limit of the axis.

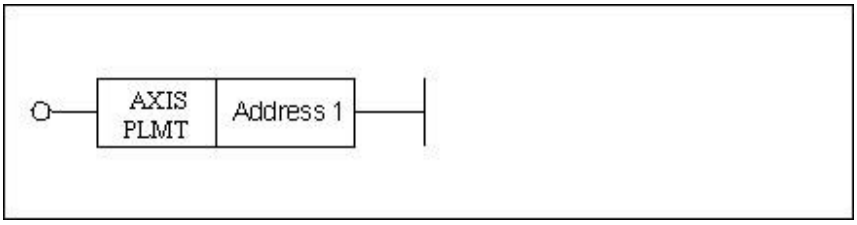
Parameter Parameter 1: axis number

Example

Ladder Diagram	
Statement List	AXLMF2 2
Description	When X2.0 is turned on, the second soft limit of the axis 2 is effective.

6.2.24 Block Switch in Positive Limit Direction AXISPLMT

Format

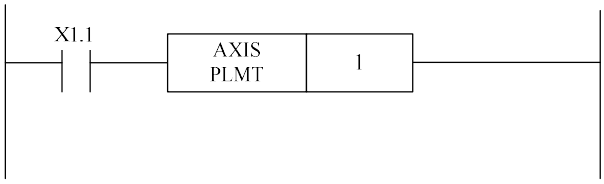


Parameter	Parameter form	Data type	Storage area	Properties
<Address 1>	□□□□	INT	Constant	Pre √
				Post ×

Function The positive limit of the axis.

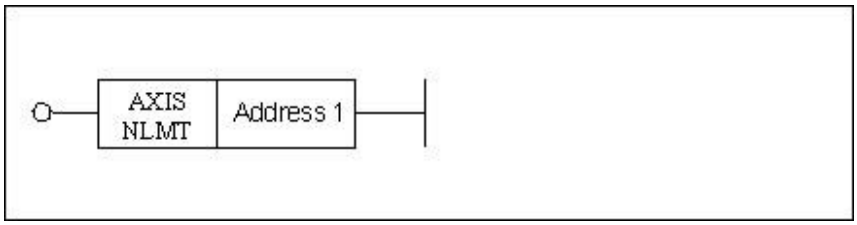
Parameter Parameter 1: axis number

Example

Ladder Diagram	
Statement List	AXISNLMT 1
Description	X1.1 being effective indicates the positive limit.

6.2.25 Block Switch in Negative Limit Direction AXISNLMT

Format

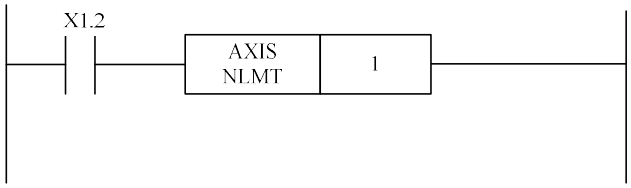


Parameter	Parameter form	Data type	Storage area	Properties
<Address 1>	□□□□	INT	Constant	Pre √ Post ×

Function The negative limit of the axis.

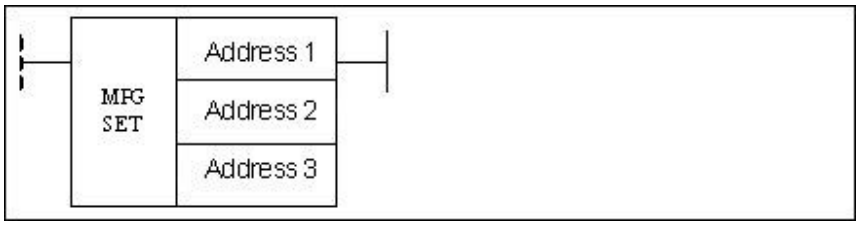
Parameter Parameter 1: axis number

Example

Ladder Diagram	
Statement List	AXISNLMT 1
Description	X1.2 being effective indicates the axis 1 meets the negative limit point.

6.2.26 Handwheel MPGSET

Format



Parameter	Parameter form	Data type	Storage area	Properties
<Address 1>	□□□□	INT	Constant	Pre √
<Address 2>	□□□□	INT	Constant, X, Y, F, G, R, W, D, P, B	Post ×
<Address 3>	□□□□	INT	Constant, X, Y, F, G, R, W, D, P, B	

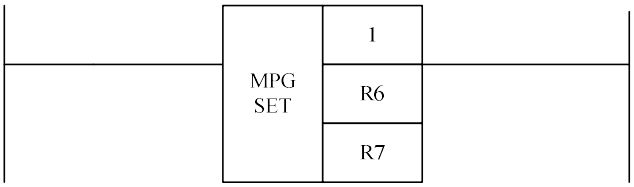
Function To set handwheel.

Parameter Parameter 1: handwheel number.

Parameter 2: axis number.

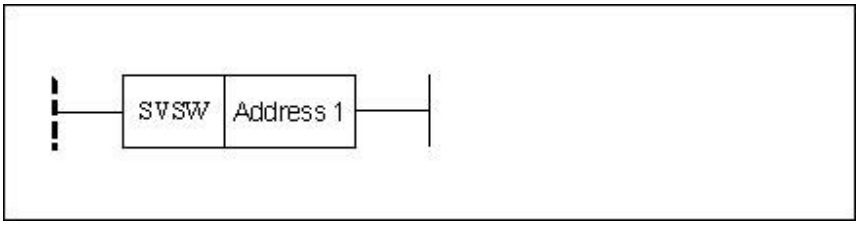
Parameter 3: override value.

Example

Ladder Diagram	
Statement List	MPGSET 1 R6 R7
Description	Handwheel 1 gets its incremental value. The axis number selected by the handwheel 1 is stored in R6. Override value of the handwheel 1 is stored in R7.

6.2.27 Servo Enable (Bus) SVSW

Format

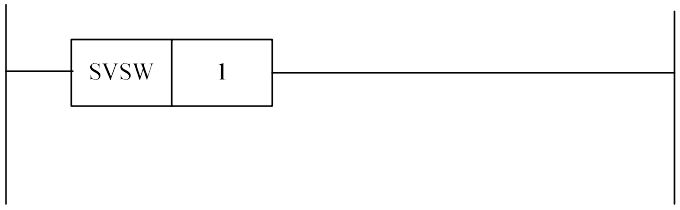


Parameter	Parameter form	Data type	Storage area	Properties
<Address 1>	□□□□	INT	Constant	Pre ○
				Post ×

Function Servo enable.

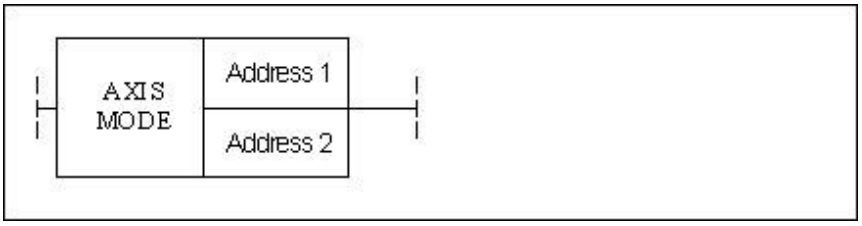
Parameter Parameter 1: axis number.

Example

Ladder Diagram	
Statement List	SVSW 1
Description	Servo enable of the axis 1.

6.2.28 Axis Working Mode AXISMODE

Format



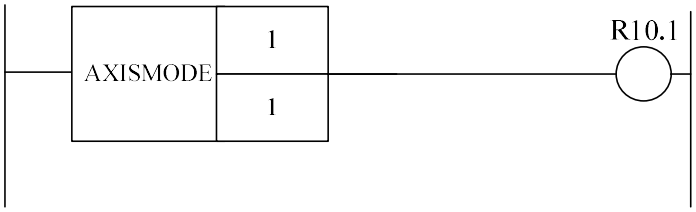
Parameter	Parameter form	Data type	Storage area	Properties
<Address 1>	□□□□	INT	Constant	Pre ○
<Address 2>	□□□□	INT	Constant	Post ×

Function To select the working mode of the axis.

Parameter Parameter 1: axis number.

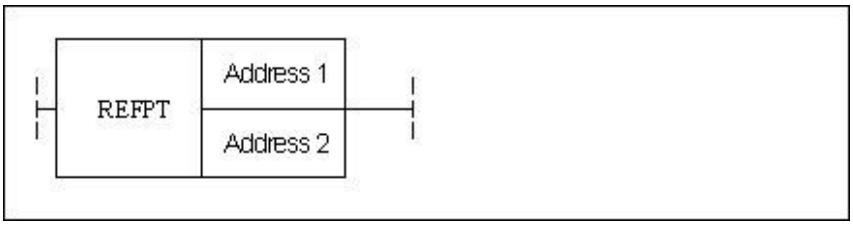
Parameter 2: "0" is position, "1" is speed, and "2" is torque.

Example

Ladder Diagram	
Statement List	AXISMODE 1 1
Description	The working mode of axis 1 is set to the speed mode.

6.2.29 Axis Reference REFPT

Format



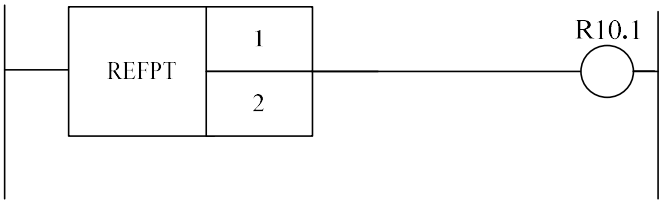
Parameter	Parameter form	Data type	Storage area	Properties
<Address 1>	□□□□	INT	Constant	Pre ○
<Address 2>	□□□□	INT	Constant	Post √

Function To confirm the reference position of axis.

Parameter Parameter 1: axis number.

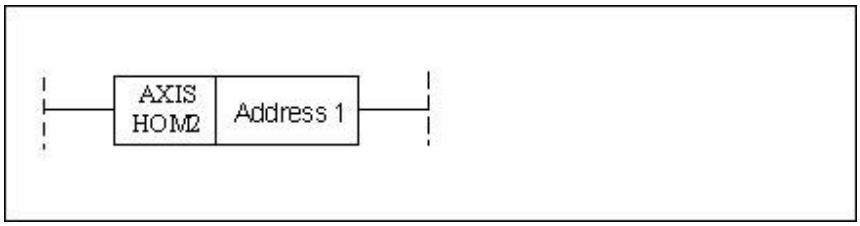
Parameter 2: "2" indicates that the second reference is valid, "3" indicates that the third reference is valid, "4" indicates that the forth reference is valid, and "5" indicates that the fifth reference is valid.

Example

Ladder Diagram	
Statement List	REFPT 1 2
Description	The second reference of the axis 1 is valid, and R10.1 is output.

6.2.30 During Axis Home AXISHOM2

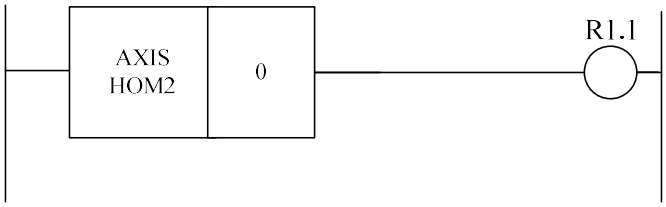
Format



Parameter	Parameter form	Data type	Storage area	Explanation	Properties
<Address 1>	□□□□	INT	Constant	Axis number	Pre ○
					Post ○

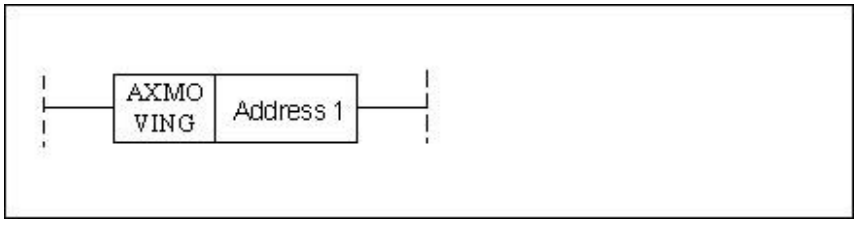
Function To get the home status while the axis is returning home. In the process of axis home, some operations cannot be performed, in which case the home status must be judged. The corresponding F status word is F0.2.

Example

Ladder Diagram	
Statement List	AXISHOM2 0
Description	While the axis is returning home, R1.1 is output, where other manual operations are not allowed.

6.2.31 During Axis Moving AXMOVING

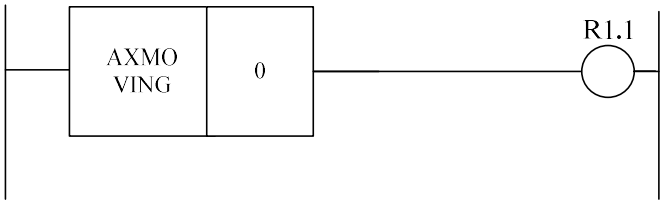
Format



Parameter	Parameter form	Data type	Storage area	Explanation	Properties
<Address 1>	□□□□	INT	Constant	Axis number	Pre ○ Post ○

Function To get the axis status during its movement. In the process of axis moving, some operations cannot be performed, in which case the status must be judged. The corresponding F status word is F0.0.

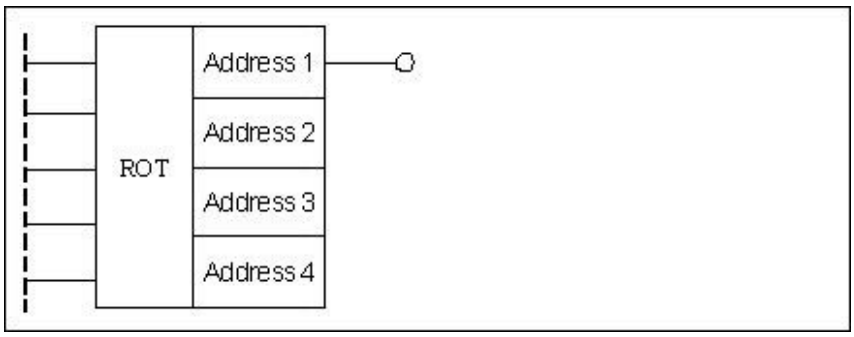
Example

Ladder Diagram	
Statement List	AXMOVING 0
Description	The moving sign of axis 0 is valid. While the axis is moving, R1.1 is output.

6.3 System Function

6.3.1 Rotation ROT

Format



Parameter	Parameter form	Data type	Storage area	Properties
<Address 1>	□□□□	INT	Constant	Pre ○
<Address 2>	□□□□	INT	X, Y, F, G, R, W, D, P, B	Post √
<Address 3>	□□□□	INT	X, Y, F, G, R, W, D, P, B	
<Address 4>	□□□□	INT	X, Y, F, G, R, W, D, P, B	

Function Rotation control, which is used for tool rest and the like. The output is 0 for the rotation in the clockwise direction, and the output is 1 for the rotation in the counter clockwise direction.

Parameter Input 1: enable on/off

Input 2: starting number. If the number is 0, the position number of rotational equipment starts from 0; if the number is 1, the position number of rotational equipment starts from 1.

Input 3: Whether to select a cutter nearby. If it is 0, the cutter will be selected in the clockwise direction; if it is 1, the cutter will be selected nearby.

Input 4: target location type. When the value is 0, the current target location is counted; when the value is 1, the previous location of target is counted.

Input 5: type of counting result. The value of 0 represents the number of count locations, and the value of 1 represents the count steps.

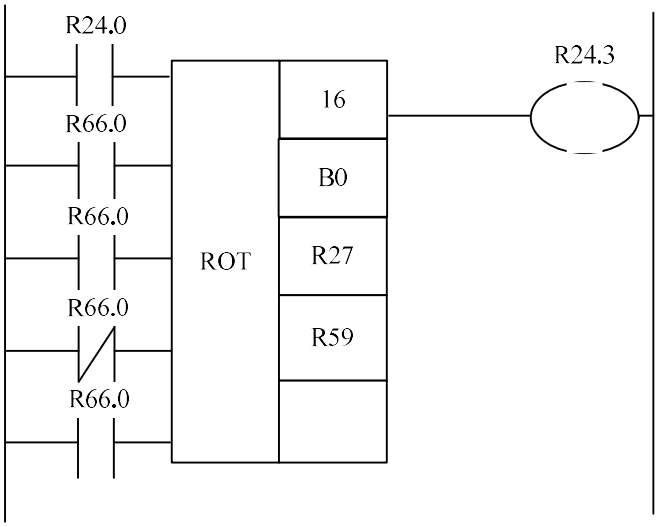
Parameter 1: maximum quantity of tool posts.

Parameter 2: address of current position.

Parameter 2: address of target position

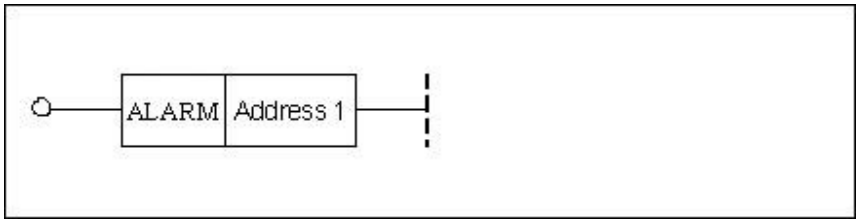
Parameter 4: address of count result. The meaning of count result is determined by input 4 and input 5.

Example

Ladder Diagram	
Statement List	ROT 16 B0 R27 R59
Description	R66.0 is a true contact, where the setting status of ROT module is: the counting for the starting number is started from 1, use the function of short-path selecting tool, count the current target location, output the counting steps, and output the counting value to R59. The current tool location is saved in B0, and the target tool location is saved in R27.

6.3.2 Alarm ALARM

Format

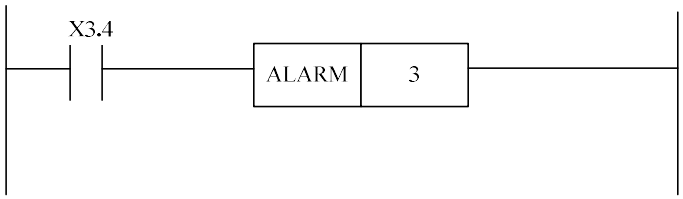


Parameter	Parameter form	Data type	Storage area	Properties
<Address 1>	□□□□	INT	Constant	Pre ○
				Post ×

Function To generate alarm.

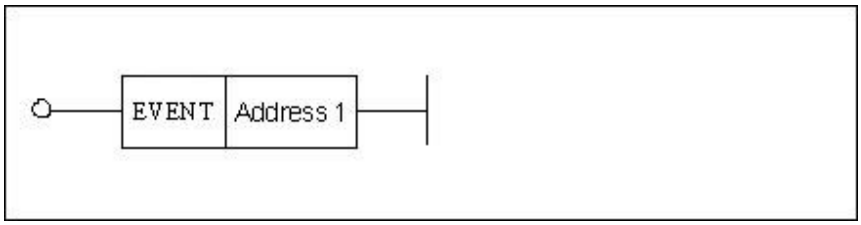
Parameter Parameter 1: alarm code. The PLC alarm code is from 1 to 256, and the prompt number of PLC is from 501 to 884.

Example

Ladder Diagram	
Statement List	ALARM 3
Description	When X3.4 is turned on, the alarm 3 is generated.

6.3.3 Event EVENT

Format

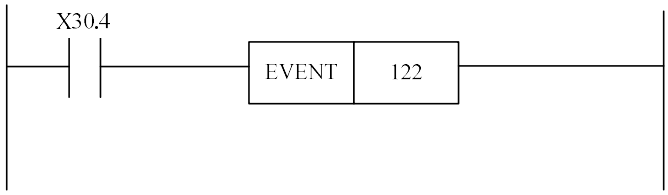


Parameter	Parameter form	Data type	Storage area	Properties
<Address 1>	□□□□	INT	Constant	Pre √ Post ×

Function To create the event object.

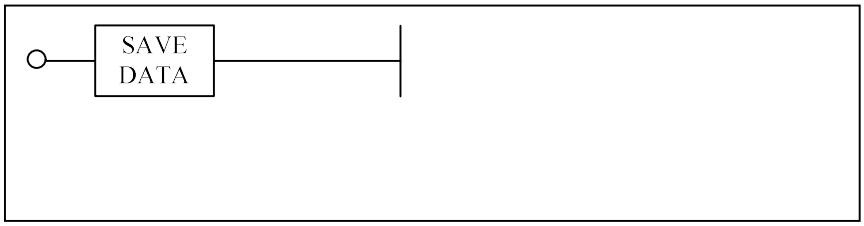
Parameter Parameter 1: event number

Example

Ladder Diagram	
Statement List	EVENT 122
Description	When 30.4 is turned on, the event 122 is generated.

6.3.4 Save Data SAVEDATA

Format



Parameter	Parameter form	Data type	Storage area	Properties
None				Pre √
				Post ×

Function To save all data.

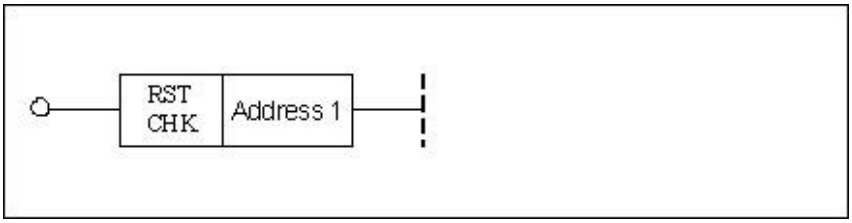
Parameter None

Example

Ladder Diagram	
Statement List	SAVEDATA
Description	When X30.4 is turned on, the data which hasn't been saved before outage can be saved.

6.3.5 Reset Setting Output RSTCHK

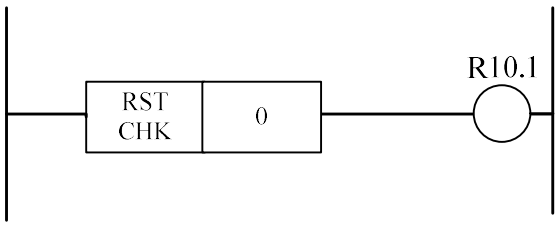
Format



Parameter	Parameter form	Data type	Storage area	Explanation	Properties
<Address 1>	□□□□	INT	Constant	Channel No.	Pre √ Post √

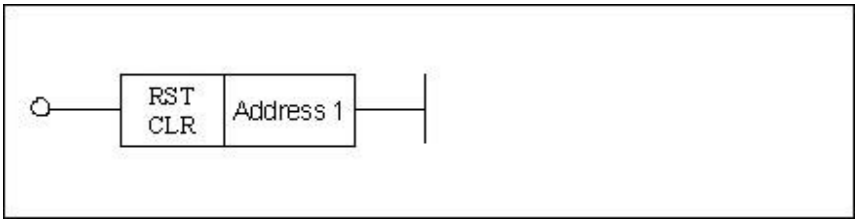
Function To get panel reset signal (must be used with RSTCLR simultaneously). If some reset actions in PLC need to be performed after the reset button on the panel is pressed, use this function module. At this point, "Resetting…" will be shown on the CNC interface.

Example

Ladder Diagram	
Statement List	RSTCHK 0
Description	The output of setting reset is 1

6.3.6 Reset Clear RSTCLR

Format

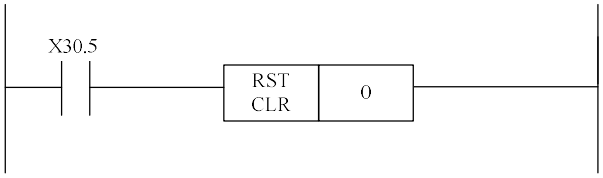


Parameter	Parameter form	Data type	Storage area	Explanation	Properties
<Address 1>	□□□□	INT	Constant	Use it together with RSTCHK.	Pre √ Post ×

Function After the reset actions in PLC are completed, the reset must be cleared (must be used with RSTCHK simultaneously), and the signal of completing reset is transmitted to CNC. “Reset done” will be shown on the CNC interface.

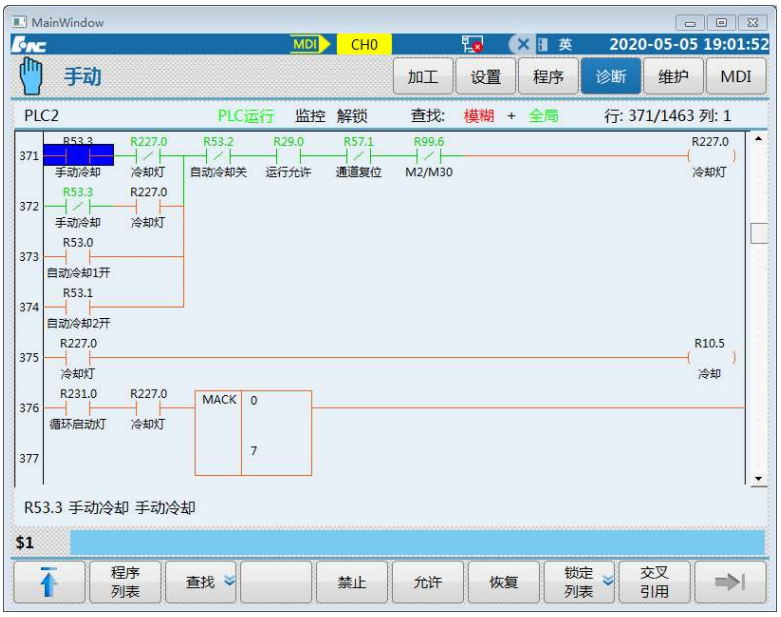
Parameter Channel number.

Example

Ladder Diagram	
Statement List	RSTCLR 0
Description	When X30.5 is turned on, the reset is cleared, and the output is 0.

7 Operational Monitoring and Online Modification for Ladder Diagram

The function of operational monitoring and online modification for the ladder diagram, which is provided by PLC edit function, will monitor changes of the status of each component in the ladder diagram, and force a modification of a component status to achieve the goals of debugging.

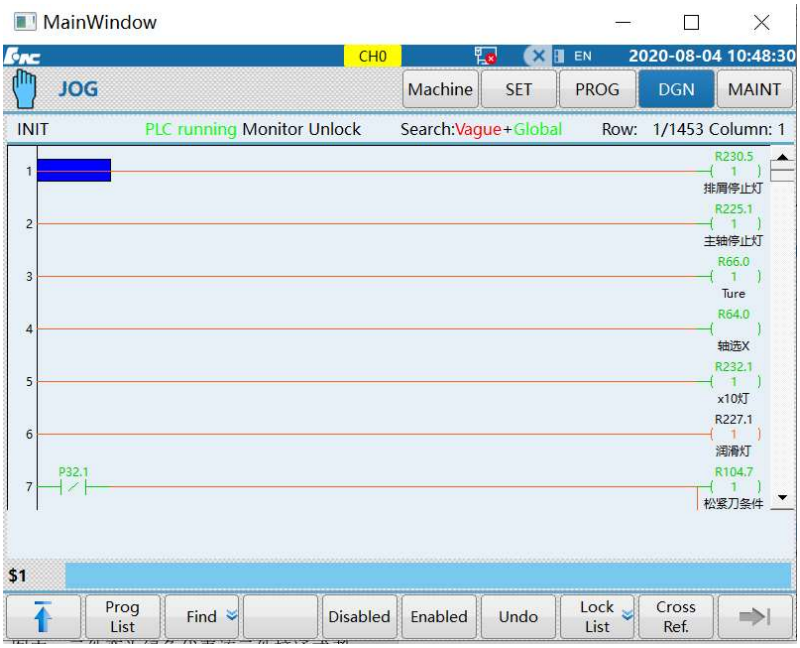


Press “Ladder monitoring” on the diagnosis interface, to access the ladder diagram monitoring interface, as seen above. The buttons on this interface include Program list, Find, Disable, Enable, Undo, Lock list, Cross reference, and Return.

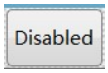
	Ladder diagram diagnosis: view the value of each variable, and perform intervention operations of component.
Program list	Display all subprogram lists
Find	Search the component or the register.
Disable	Turn off a register
Enable	Turn on a register
Undo	Restore “disable” or “enable”
Lock list	Fix the value of a register
Cross reference	View all multiplexed registers

7.1 Ladder Monitoring

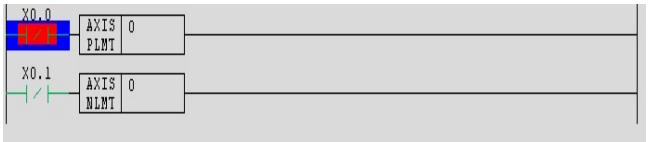
Press Ladder Monitoring button to access the corresponding interface, as seen in below figure. The buttons include: Disable, Enable, Undo, Return, etc.

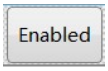


Press “DGN→Status” to view the value of each variable. User can move the cursor up and down to view variables. As seen above, the component in green indicates that this component is turned on or is valid, then user can execute the operations such as disable, enable, undo and the like.

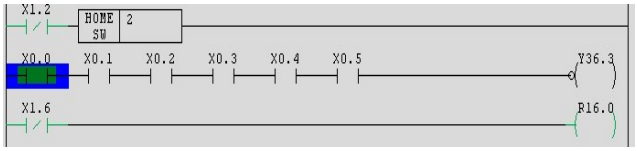


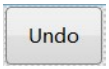
Disable function button. Move the cursor on the component, and press Disable button to shield the component. As shown in the below figure, press disable, the component turns red, which indicated that the component is shielded. Press Undo button to restore the function of the component, which will be covered later.



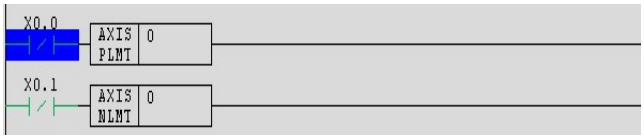


Enable function button. Move the cursor on the component, press Enable button, and the component is enabled. As demonstrated below, the cursor has been moved on the component, press Enable button, the component turns green, which indicates that the component is enabled. In the below figure, X0.0 is normally open. Move the cursor on X0.1, press Enable button, the component turns green, and is switched off.



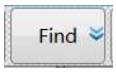


Undo function button. Move the cursor on the component, and press Undo, to undo the shielding or enabling operations described above. Press this button after Disable function is enabled, the red color on the component disappears, which indicates that the function of component is restored, as shown in the figure below.

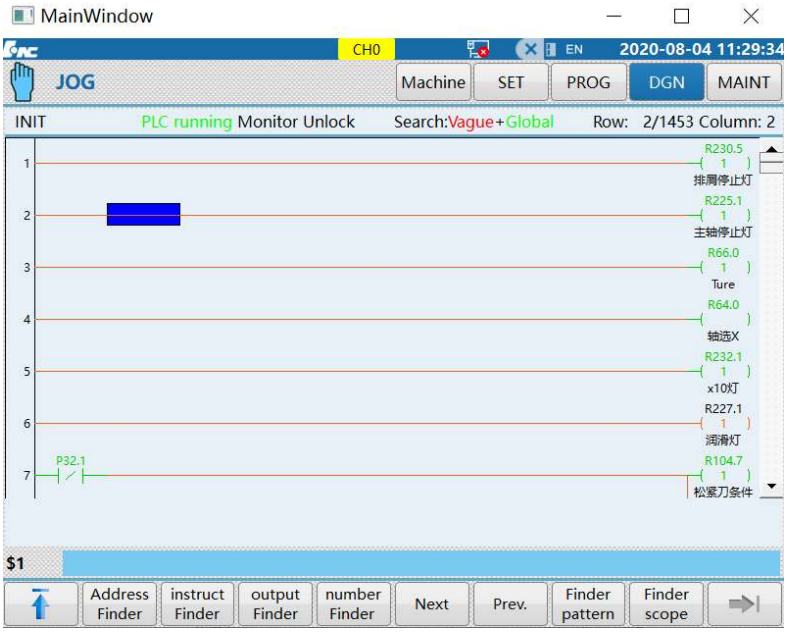


Return function button. Press this button to return to the interface of ladder diagram monitoring for performing other operations.

7.2 Find



Press Find, then the operation interface as shown in the figure below appears, where the component can be looked up.

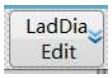


For example, type X0.0, press “Enter”, the first X0.0 of the program under the cursor line can be found. See the figure as below:

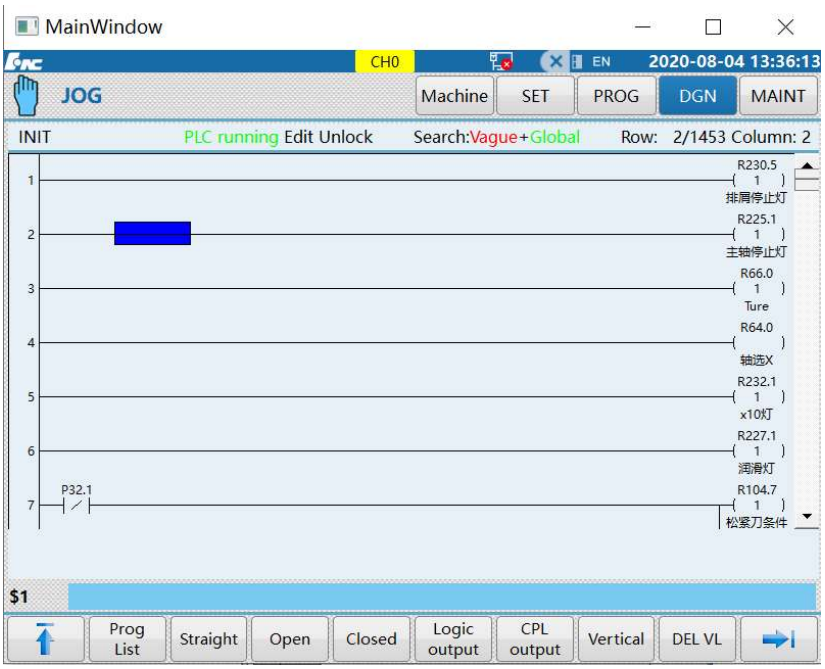
The found component is covered by blue curser. If you want to continue to search, press Next or Prev., then other components with same names can be found.

The functional button of Return. Press this button to return to the interface of the ladder diagram monitoring.

7.3 Edit



User presses the corresponding functional button of Ladder Diagram Edit menu, to perform operations on the new component.

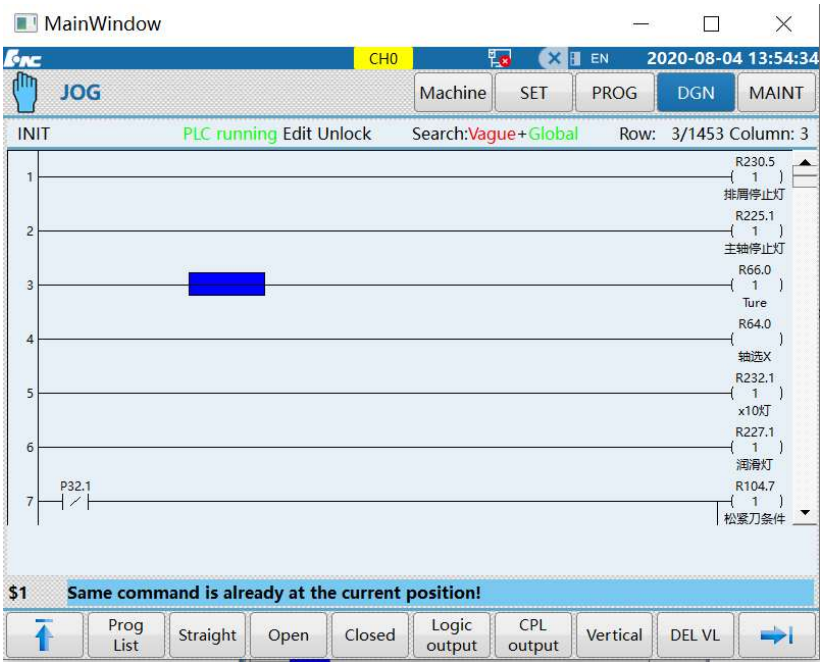


Straight	Straight line: insert a straight line
Open	Normally open: insert a normal-open contact
Closed	Normally closed: insert a normal-closed contact
Logic output	Logical output: insert an output
CPL output	Inverted output: insert an inverted output
Vertical	Vertical line: insert a vertical line
DEL VL	Delete vertical line: Delete a vertical line
Find	Search a component or register

DEL element	Delete component: delete a component or register
FM	Functional module: add an instruction module
Edit gird	Edit grid: block operation of PLC program
List edit	List edit: edit the subprogram list of PLC
Two-coil	Dual coil: edit or search coil
Update Mod.	Update modification: update after PLC modification
Undo Mod.	Undo modification: abandon the previous editing of the PLC

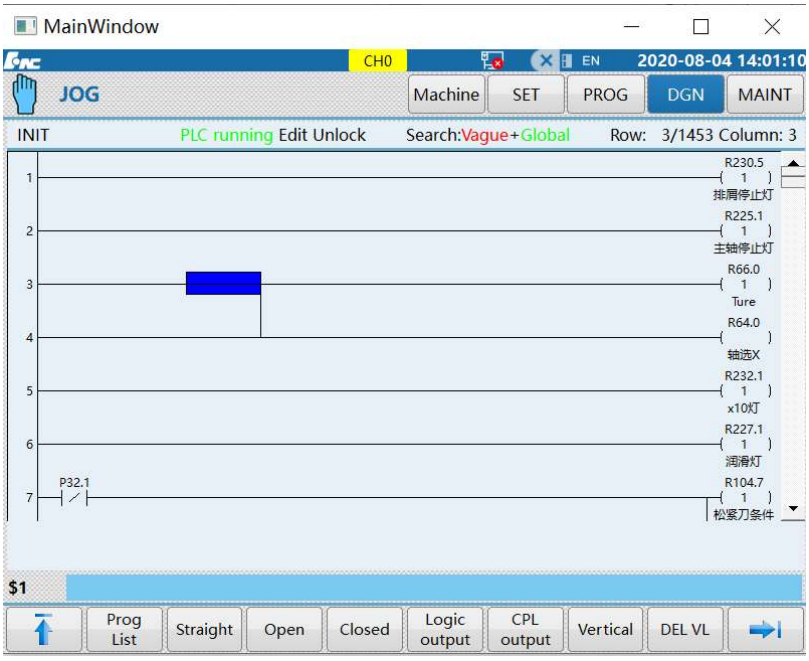
7.3.1 Insert Straight Line

Press the functional button of Straight Line to insert a straight line in the ladder Diagram as shown below:



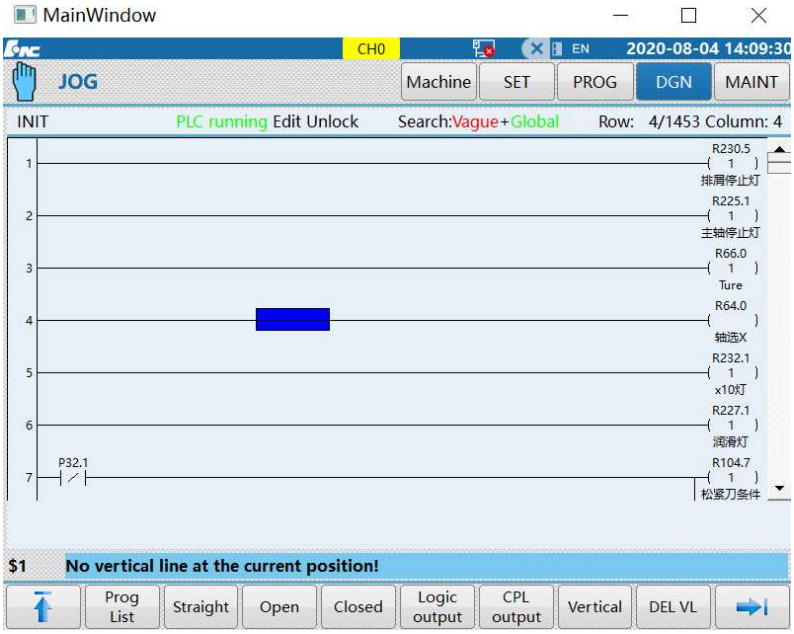
7.3.2 Insert Vertical Line

Press the functional button of Vertical Line to insert a vertical line after the cursor, as shown in the figure below:

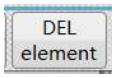


7.3.3 Delete Vertical Line

Press the functional button of “Delete Vertical Line” to delete the vertical line after the cursor, as demonstrated below:



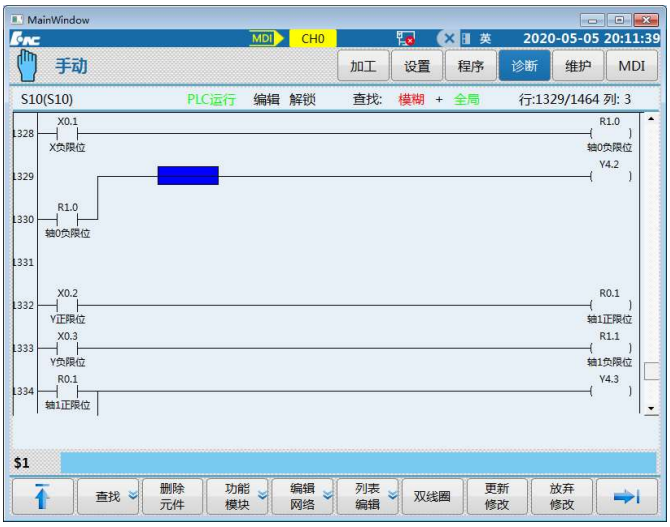
7.3.4 Delete Component



Move the cursor on the component to be deleted, press the functional button of to delete the component in the ladder diagram.

- Before the deletion

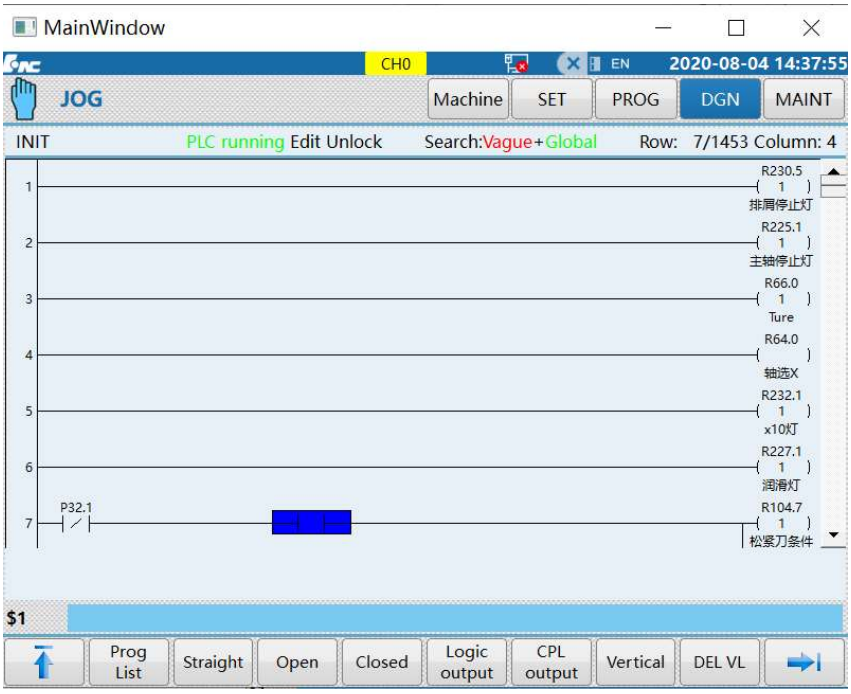
- After the deletion



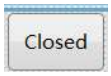
7.3.5 Normally-open

Open

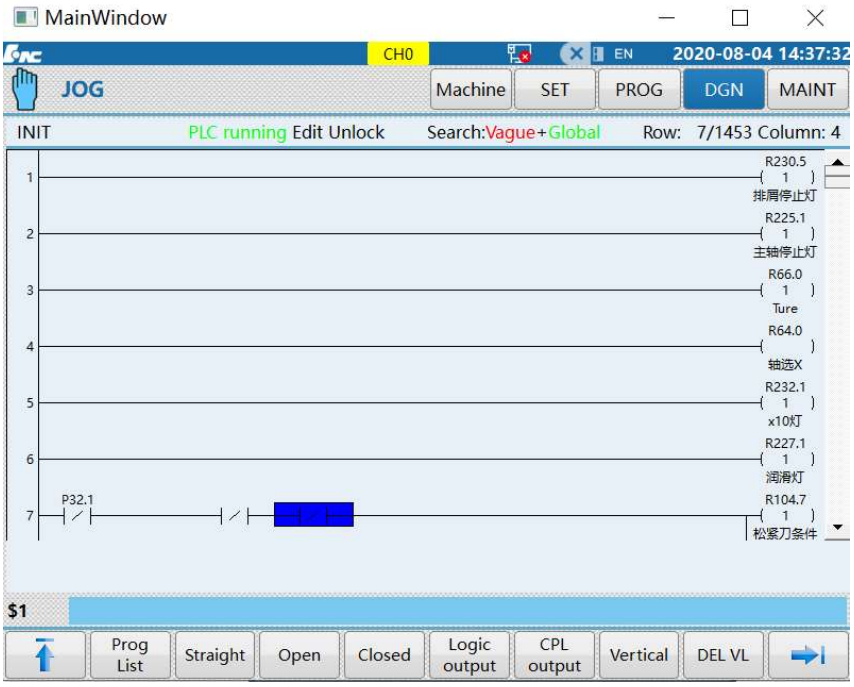
Move the cursor to the position where the normally-open contact is to be inserted, press the functional button of "Normally open" to insert the normally-open contact at the specified position.



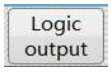
7.3.6 Normally-closed



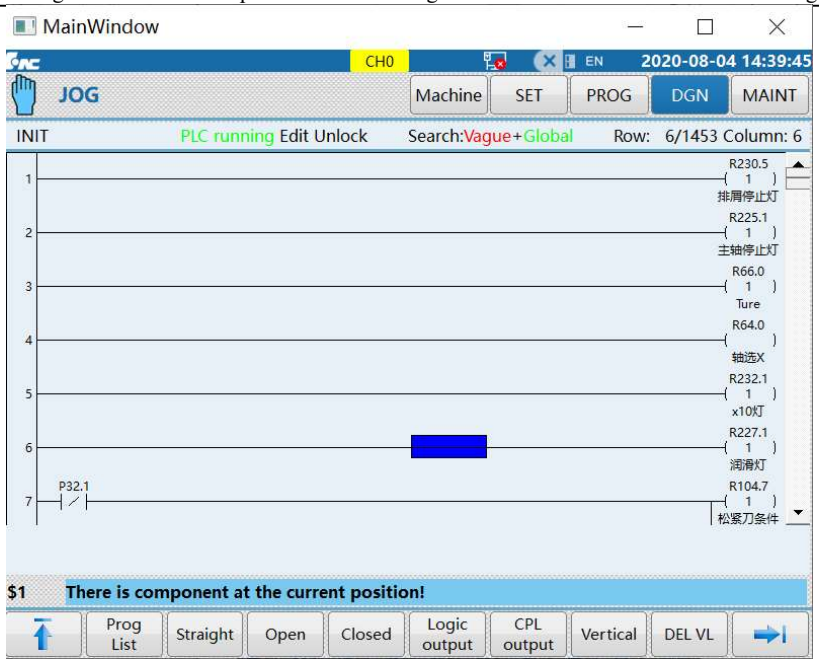
Move the cursor to the position where the normally-closed contact is to be inserted, press the functional button of "Normally-closed" to insert the "normally-closed" contact at the specified position, as shown in the figure below:



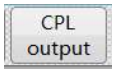
7.3.7 Logical Output



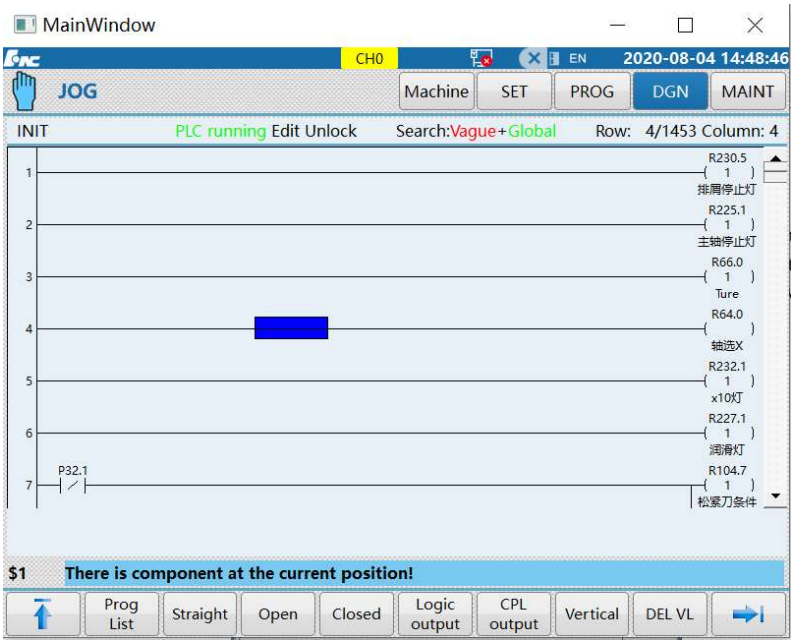
Move the cursor to the position where the logical output needs to be inserted, press the functional button of "Logical output" to insert the logical output at the specified position in the ladder diagram, as shown in the figure below. It is important to note that pre can be added to the logical output, but post cannot. Refer to the section of programming for details.



7.3.8 Inverted Output

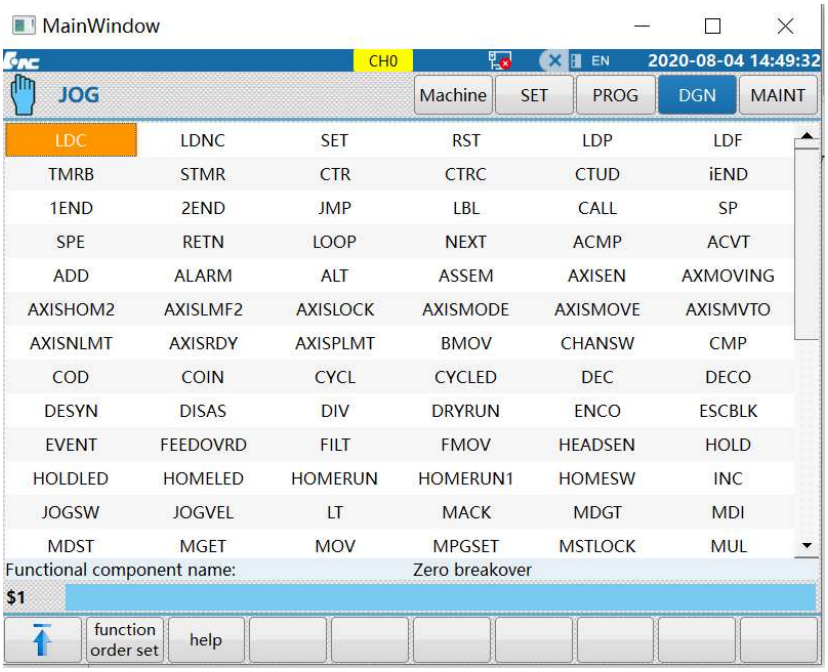


Move the cursor to the position where the inverted output needs to be inserted, press the functional button “Inverted output” to insert the inverted output at the specified position in the ladder diagram, which is illustrated by the following figure.

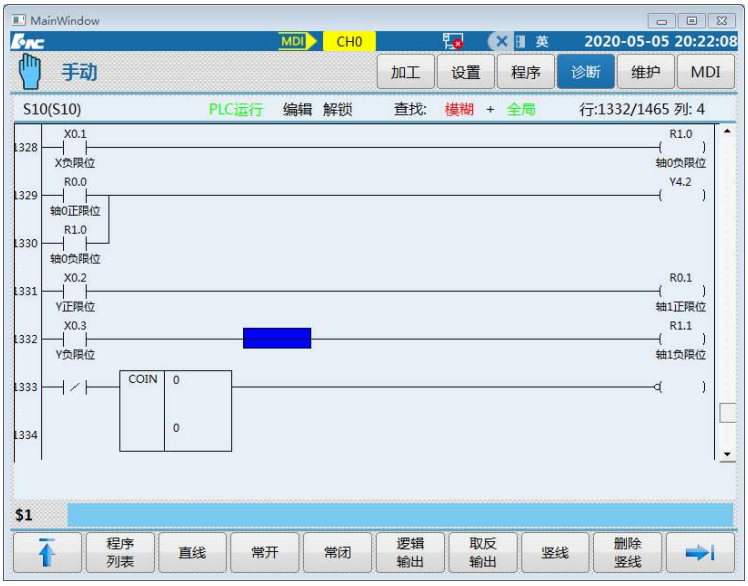


7.3.9 Functional Module

Press the functional module button to access the operation interface shown as below figure, and select the functional module needed.



Then hit Enter to enter the selected functional module into the ladder diagram. User can press the initial word of the component to select the relevant component.



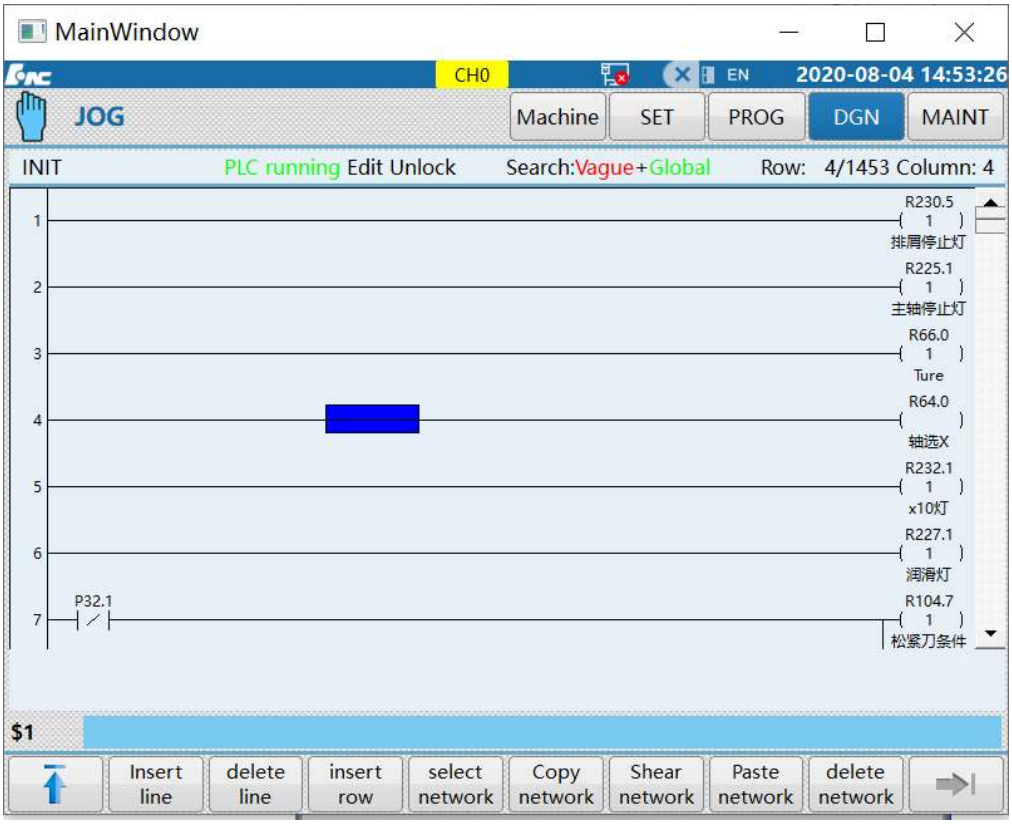
Press functional module button again to return to the interface of operation modification.

7.3.10 Return

Press “Return” to return to the previous operation interface.

7.4 Edit Network

User can press the buttons listed in the below table to edit the ladder diagram.

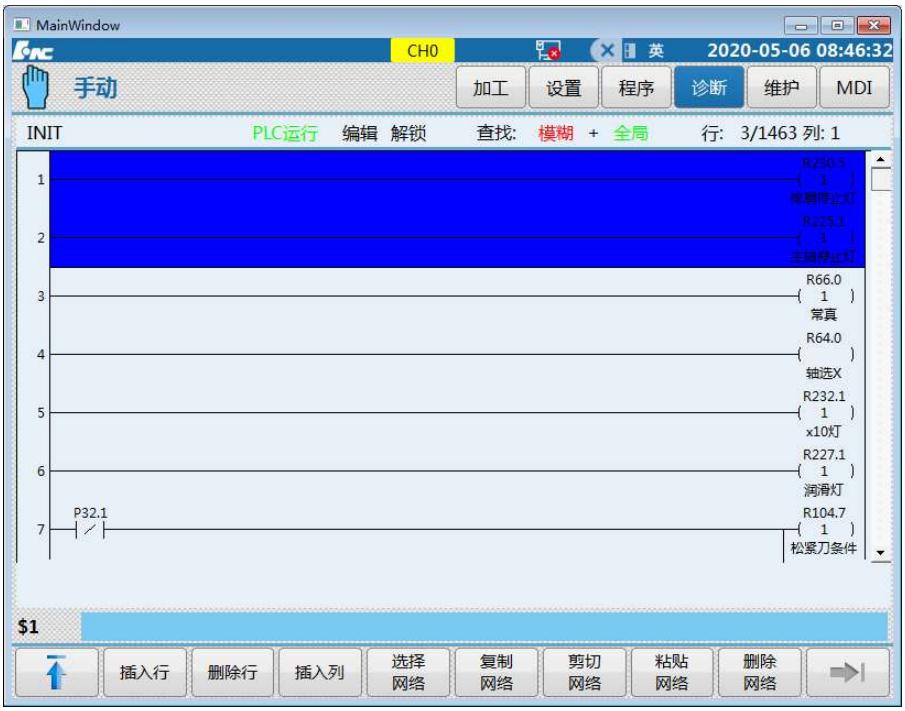


Insert line	Insert line: insert a line before the cursor line
Delete line	Delete line: delete the cursor line
Insert column	Insert column: insert a column before the cursor column
Select network	Select network: enlarge the area the cursor covers
Copy network	Copy network: copy the PLC the cursor covers
Paste network	Paste network: paste the PLC copied or cut
Cut network	Cut network: cut the PLC the cursor covers
Delete network	Delete network: delete the PLC the cursor covers

7.4.1 Select Network

选择
网络

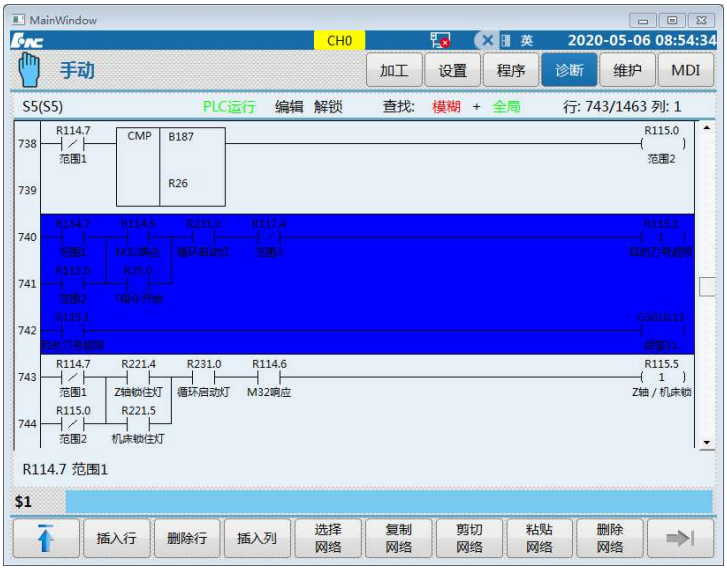
Move the cursor to the line that you want to select, press the functional button of Select Network, the selected line turns blue, and then press Select Network again to select the next line of the current line. It is illustrated by the following figure. You can perform the operations such as delete after selecting the line you want.



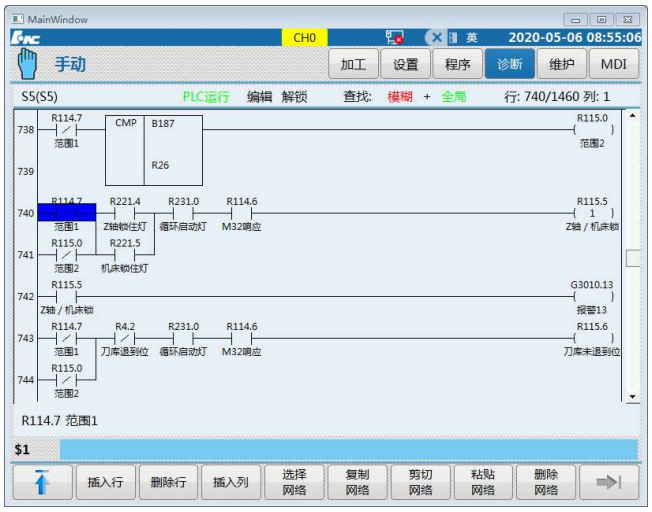
7.4.2 Delete Network

Move the cursor to the line to be deleted, press “Select” button, the line turns blue, then press “Delete” to delete this line.

- Before the deletion



- After the deletion



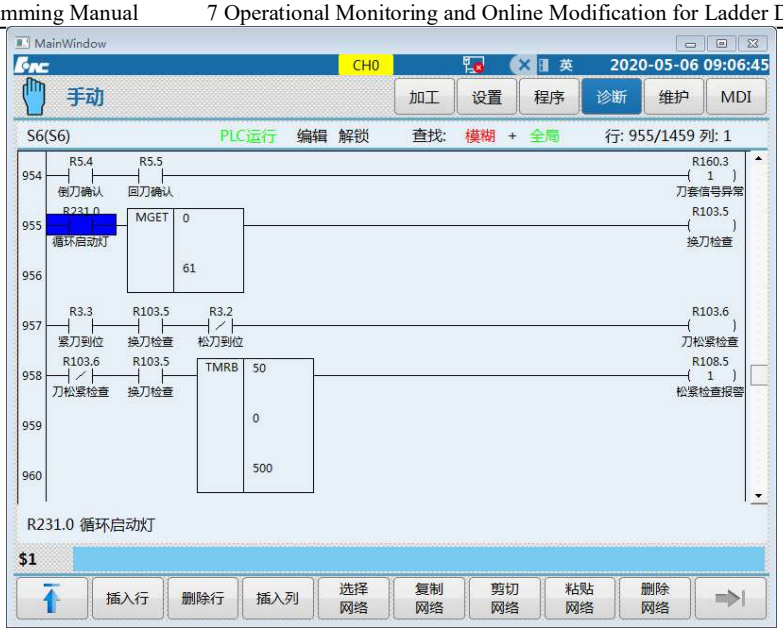
7.4.3 Move

选择网络

First move the cursor to the line to be moved, then press Select Network button, this line turns blue.

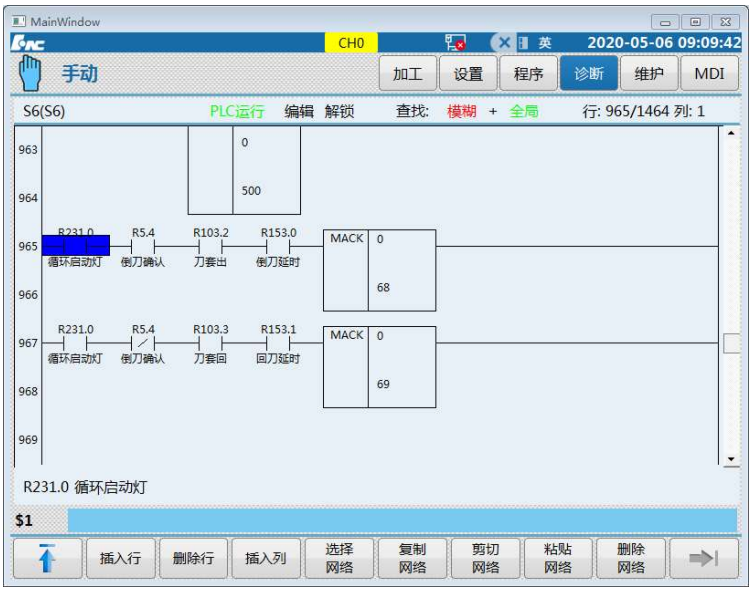
剪切网络

Press the functional button Paste Network to access the interface which is shown in the below figure, and the selected line disappears.



粘贴网络

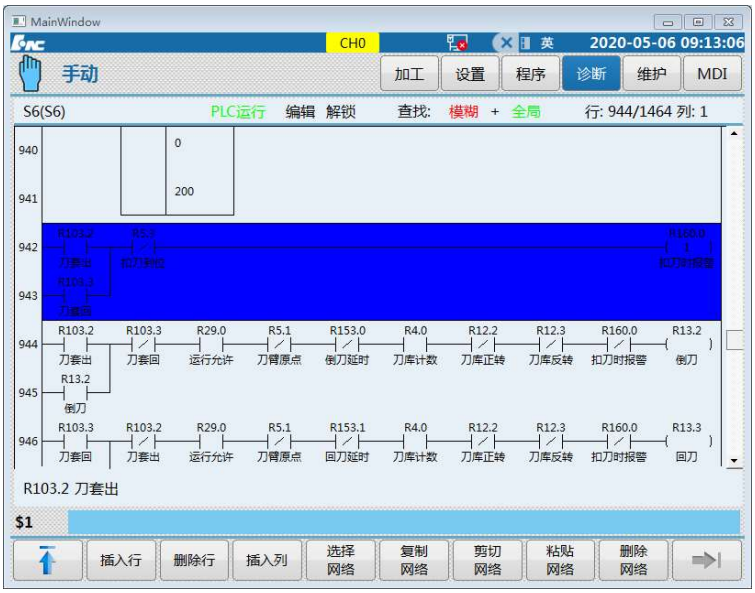
Move the cursor to the target line, and press Paste to move the selected line to the target line.



7.4.4 Copy

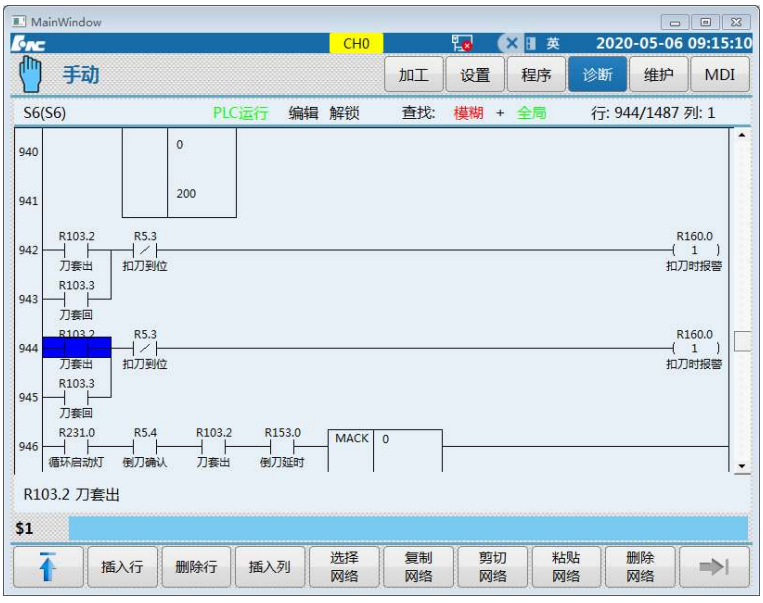
复制网络

Move the cursor to the position of the line that needs to be copied, then press the functional buttons Copy Network. See as below:



粘贴网络

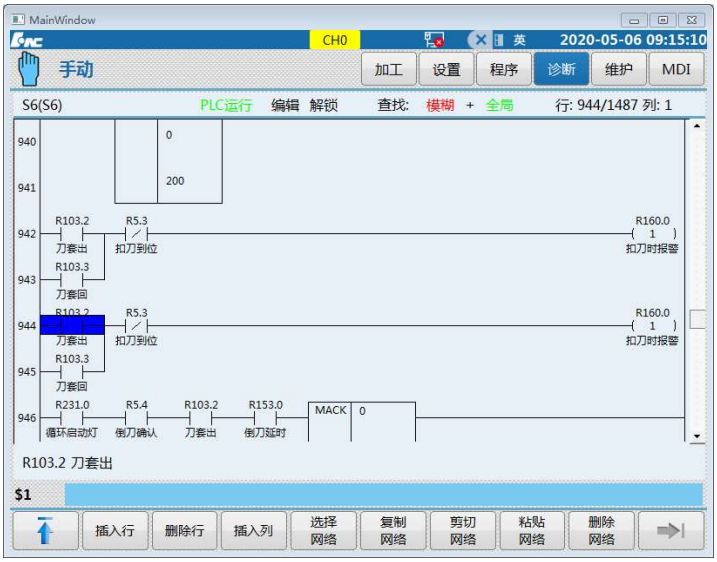
Move the cursor to the target line, and press the functional button of paste to paste the copied line.



7.4.5 Paste Network

粘贴
网络

The functional button Paste Network has been applied in section 7.4.3 and 7.4.4. Refer to the two sections for the detailed operations.

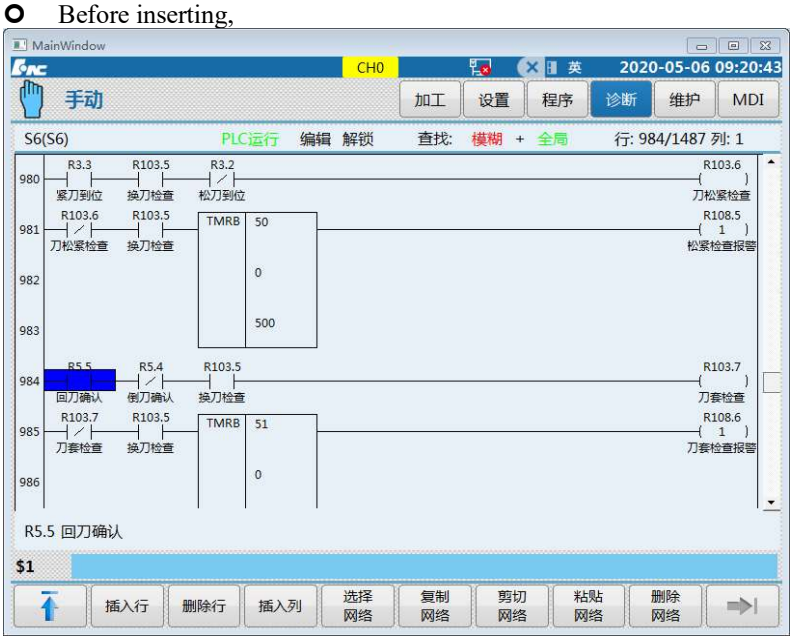


7.4.6 Insert Line

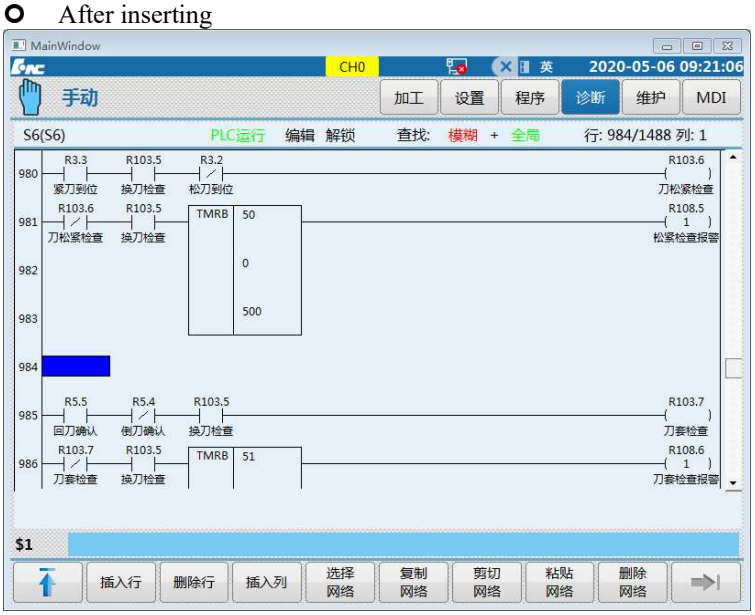
插入行

As shown in the figure below, move the cursor to the next line of the line to be inserted, press the functional button Insert Line, then the line is inserted. Note that the line is generally inserted above the cursor line.

- Before inserting,



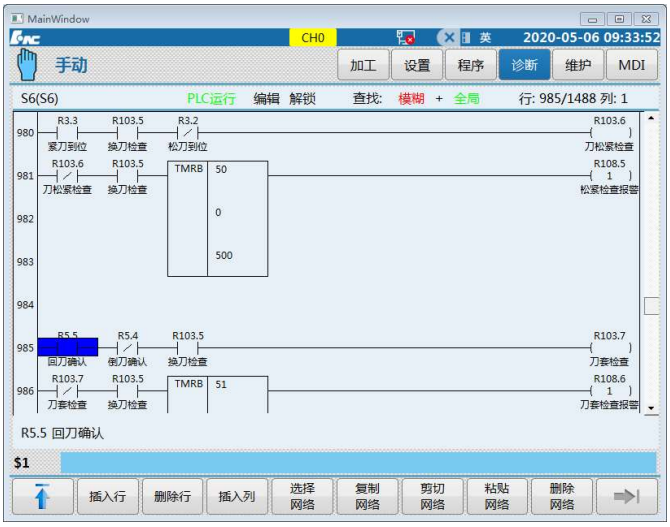
- After inserting



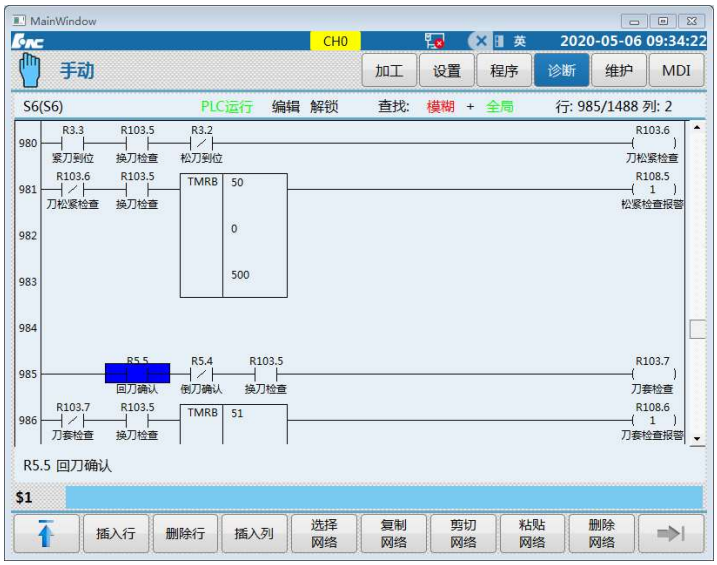
7.4.7 Insert Column

The functional button of Insert Column is in contrast to Insert Line where the column is added before the cursor column, as shown in the figure below, press the functional button of Insert Column, then a new column is added before the cursor column.

- Before adding



- After adding



7.4.8 Return

Press the functional button of Return to back to the previous interface.

7.4.9 Update Modification

After the ladder diagram is compiled, the current ladder diagram will be loaded into the current ladder diagram after pressing the Update Modification function key after checking.

7.4.10 Undo modification

After editing the ladder diagram, if you need to edit again, you can press this function key to cancel the editing operation of the ladder diagram.

7.5 Return

Press the functional button of Return, to return to the diagnosis interface.

8 Instruction on PLC Development Environment

This chapter includes:

7.1 Overview

7.2 Installation of Development Environment

7.3 Development Environment Interface

7.4 Development Environment Operation

8.1 Overview

HNC-LADDER-WIN Numerical Ladder Diagram Programming Software is a latest software of PLC program development environment for Series HNC8 system. This software runs on Windows XP operating system, and can easily set up ladder diagrams by visual graph programming. It is compatible with various of PLC languages that are compliant to IEC61131-3 international standard, and is a simple, efficient, and reliable PLC development tool.

8.2 Installation of Development Environment

Take the full installation of ladder diagram development environment from the CD-ROM under the Chinese version of WindowsXP as an example to illustrate the installation of the ladder diagram development environment

(1) Boot into Chinese WindowsXP.

(2) Place the disk of ladder diagram development environment in the CD-ROM drive.

(3) Double click Setup.exe file under the directory of HCNC ladder diagram, the installation program may run automatically, and then the installation wizard appears.

(4) The greeting window appears after the installation wizard interface.

(5) Click “Next (N)”, and the selection dialog displays on the screen.

(6) After doing some necessary modifications to the installation path on the selection dialog, click “Next(N)”.

(7) Then the formal installation starts, with the above displaying on the screen.

(8) After the installation is complete, the installation completion prompt box appears.

8.3 Development Environment Interface

Menu

Development Environment of ladder diagram is divided into four parts: menu, ladder diagram, statement list, and symbol table.

The bar at the top of the development environment is called menu, where every pulldown menus of ladder diagram interface are listed. Clicking a menu item shows the command options in the pulldown menu. Click a command to handle relevant operation.

There are six items in the development environment menu: file, view, tool, window, and help, which are discussed in the following:

File

The “File” menu contains the command items working on files, which mainly provide operations on files of ladder diagram with user.

New	This item is for creating a new project.
Open	This item is to open an existing dft file.
Save	This item is for saving files of current window as dft files.
Save as	The function of this item likes “save ladder diagram” item, which is to save opened files, and the difference is that this item is to save the opened files with new names.
Close	This item is for closing current ladder diagram interface.
Load dit file	This item is for opening existing dit files.
Print	This item is for printing current window contents.
Print preview	This item is for previewing print effect.
Print setup	This item is to set printing parameters.
Exit	When you select this item, the program exits.

Edit

The “Edit” menu contains rapid operations of copy, paste and the like, of which the purpose is to improve the efficiency of writing the ladder diagram.

Cut	Cut string and element.
Copy	Copy string and element
Paste	Paste string and element
Insert row	Insert a row at the current cursor location
Delete row	Delete the row which the current cursor locates.

View

"View" menu is to control the subwindow displaying in the main window.

Ladder diagram	To open (close) ladder diagram view.
Statement list	To open (close) statement list view.
Symbol table	To open (close) symbol table view.
Primitive tree	To open (close) primitive tree view on the left.
Message box	To open (close) message box at the bottom.
Toolbar	To open (close) toolbar.
Status bar	To open (close) status bar.

Tool

The function of "Tool" menu is to find/replace.

Find	To search the specified string.
Find next	To continue to search the specified string.
Replace	To replace the specified string.

Window

"Window" menu is used to open each window.

Overlap	To arrange subwindows in overlap.
Tile	To arrange subwindows in tiling.
REG	To display symbol table window.
STL	To display statement list window.
LADDER	To display ladder diagram window.

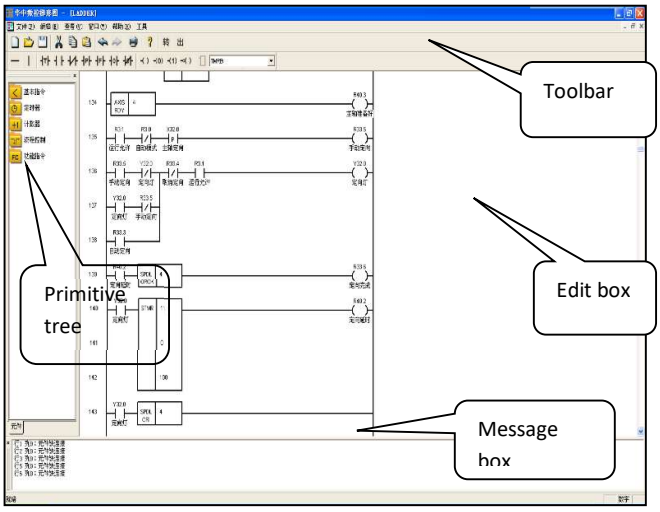
Help

About NEWPLC: to display the software version.

Ladder Diagram Interface

Four parts including toolbar, primitive tree, edit window, and message box are in the ladder diagram window.

Toolbar and primitive tree can dock freely, that means, they can be put on any of the four side walls of the main window. Toolbar can be located anywhere on the desktop.

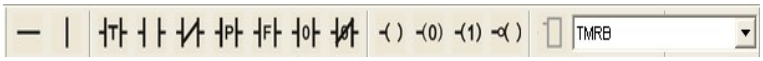


Toolbar

There are two toolbars including action bar and component bar, in the ladder diagram interface.

(1) Action bar is used to manipulate new files quickly, such as zooming, undo, redo, and so on.

(2) Component bar is used to fast add the basic input/output cell and select function module.

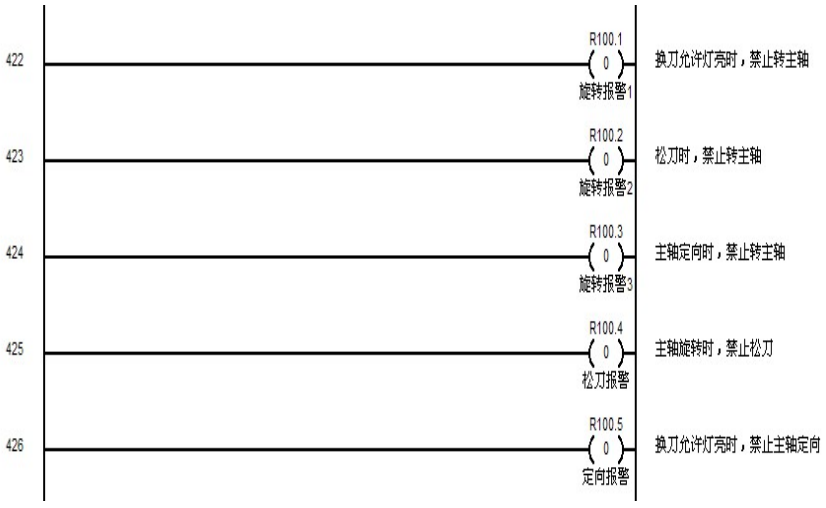


Primitive Tree

Primitive tree is used to seletct function module. Double-clicking the icon can expand and collapse the instruction tree. Select the instruction icon needed from the instruction tree.

Edit Window

Edit window is for displaying and editing the ladder diagram. The area between the left busbar and the right busbar is the editing domain of the ladder diagram, the row number you are currently editing displays on the left of the left busbar, and the comments to the meaning of output status for the current line displays on the right of the right busbar.

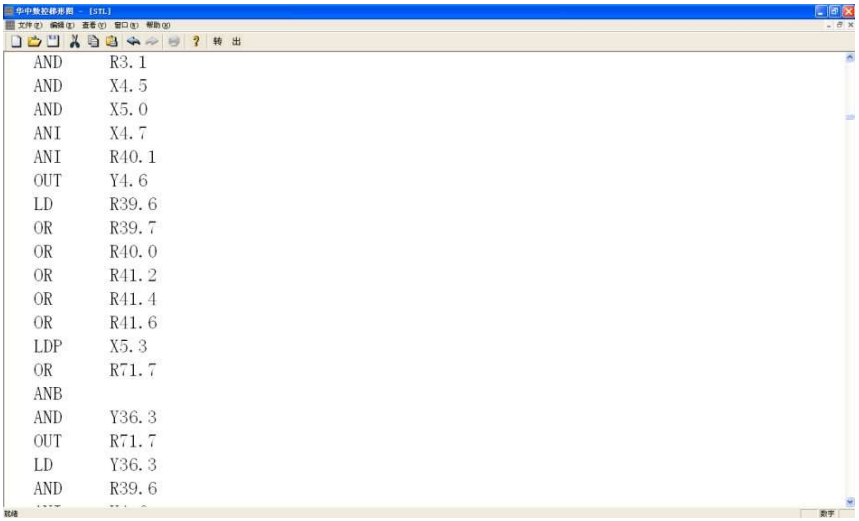


Message Box

While the ladder diagram is being compiled, if the statement error and syntax error which can be recognized may appear in the message box, then a message box is needed to display the errors in the conversion and output.

Statement List Interface

Toolbar and edit window are in the statement list interface.

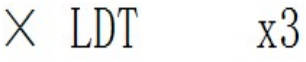


Toolbar

An operation toolbar is in the statement list interface.

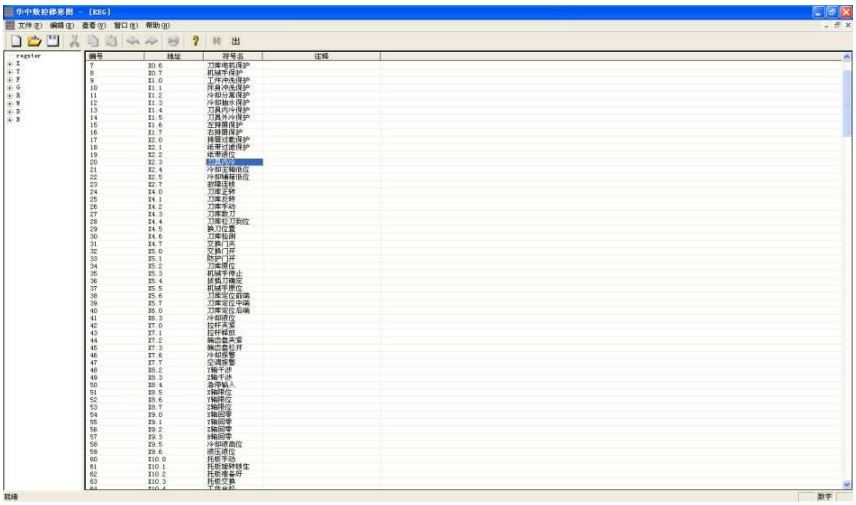
Edit Window

Edit window is for displaying and editing statement list, and can judge the current row when the statement list is being edited.



Symbol List Interface

Symbol names and comments of relevant addresses can be defined in symbol list interface.



Selection box for register is on the left of edit window of the symbol list, and edit box for register is on the right.

The edit box for register includes number, address, symbol name and comments.

- Number: automatically generate the number of the current symbol name in all the symbol names.
- Address: the specified address.
- Symbol name: the symbol name corresponding to the specified address.
- Comments: the comments corresponding to the specified address.

8.4 Development Environment Operation

Before editing PLC, first define the symbol name for the address to be used, and annotate the address, then edit PLC in the way of ladder diagram or statement list.

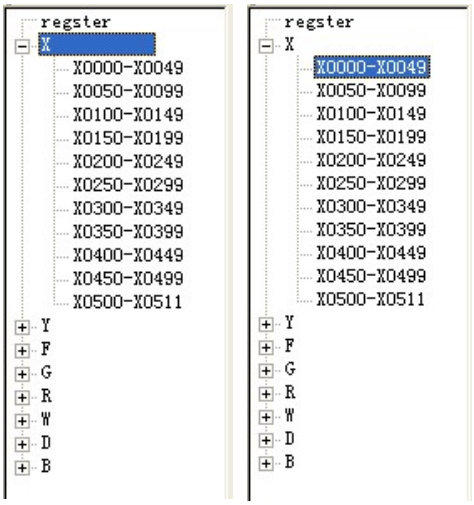
Symbol List Operation

Symbol list is used to define the symbol name for the specified address, and annotate the address.

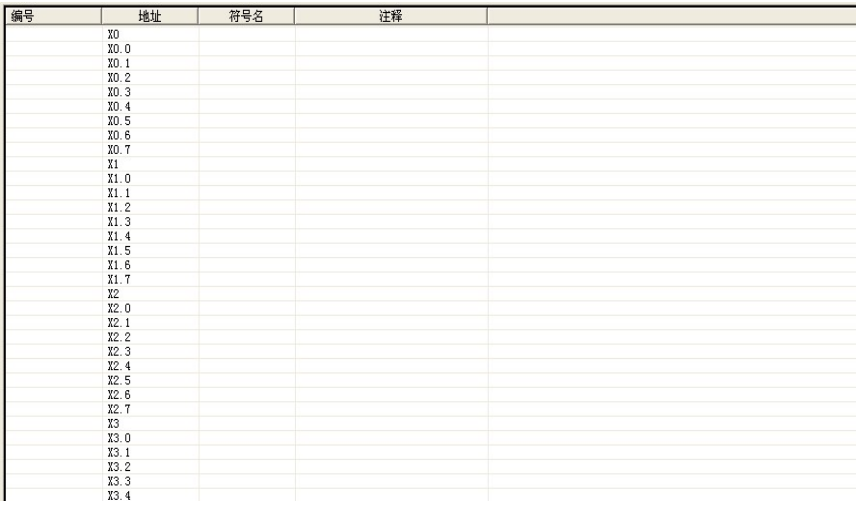
Add Symbol List

Here in X10.0 (the positive limit direction of X axis) as an example to introduce.

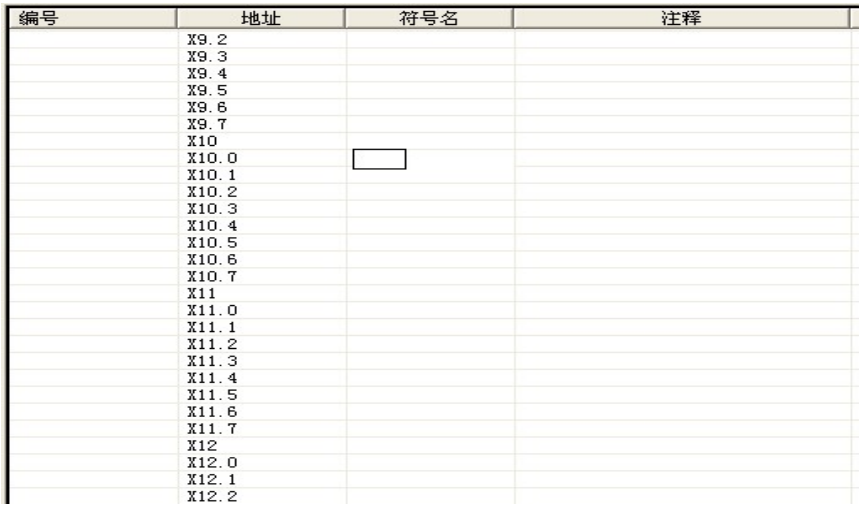
X10.0 is in X register. Select X register in the selection box of register. X10.0 is in X000-X0049.



All the register bits and points from X0000 and X0049 may appear in the edit box of register.



Click “Symbol name” item at the X10.0 row three times, then the edit box pops up.



Type “positive X limit”, and hit Enter button.

After typing the symbol name, annotate the address. The edit box will pop up, with the three-click on the “comments” item at the X10.0 row.

编号	地址	符号名	注释
	X9.2		
	X9.3		
	X9.4		
	X9.5		
	X9.6		
	X9.7		
	X10		
0	X10.0	X正限位	
	X10.1		
	X10.2		
	X10.3		
	X10.4		
	X10.5		
	X10.6		
	X10.7		
	X11		
	X11.0		
	X11.1		

Type “positive X limit, active high” in the edit box, and hit Enter button.

编号	地址	符号名	注释
	X9.2		
	X9.3		
	X9.4		
	X9.5		
	X9.6		
	X9.7		
	X10		
0	X10.0	X正限位	X正限位，高电平有效
	X10.1		
	X10.2		
	X10.3		
	X10.4		
	X10.5		
	X10.6		
	X10.7		
	X11		
	X11.0		
	X11.1		

Then defining of the symbol name of X10, and annotating of X10.0 are complete.

Delete Symbol List

When the symbol name and comments of X10.0 are not needed, delete them.

Select “X10.0” in the column of address, and hit Delete button, to delete X10.0 from the list.

编号	地址	符号名	注释
	X9.2		
	X9.3		
	X9.4		
	X9.5		
	X9.6		
	X9.7		
	X10		
0	X10.0	X正限位	X正限位，高电平有效
	X10.1		
	X10.2		
	X10.3		
	X10.4		
	X10.5		
	X10.6		
	X10.7		
	X11		
	X11.0		
	X11.1		
	X11.2		
	X11.3		

Ladder Diagram Operation

The ladder diagram is composed of rows which themselves have up to 10 cells.

Inserting Component

Inserting components can be separated into two types: inserting basic components and inserting functional components.

- **Inserting basic components**

(1) When you want to insert basic component, first select a position on the ladder diagram.

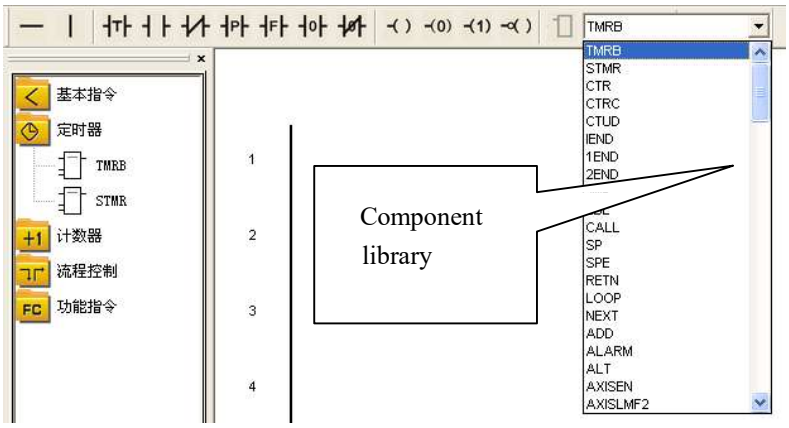
(2) Click the basic component to be inserted on the toolbar.

(3) Then the basic component is inserted in the ladder diagram.

- **Inserting functional components**

(1) Select the functional components needed from the primitive tree.

Or select it from the selection box of component on the toolbar.



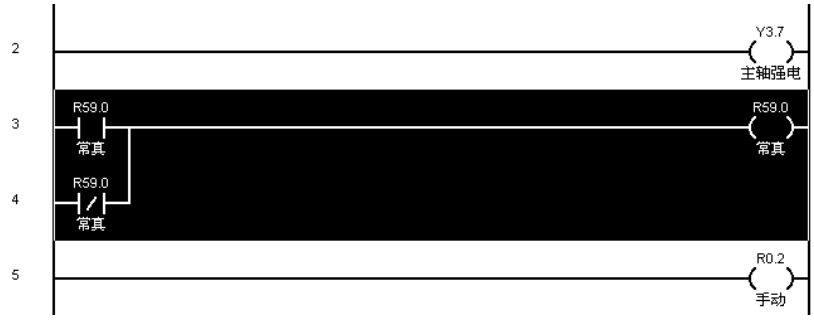
(2) Double-left click the ladder diagram, then the functional component is inserted.

Deleting Component

Select the component to be deleted in the ladder diagram, and hit Delete button to delete it.

Deleting Multi-row

Select the rows needed to be deleted. (Drag the mouse to select the area to be deleted)



Hit Delete button to delete the selected area.

Cutting, Copying and Pasting Component

First select a component in the ladder diagram.

Then choose “Cut” or “Copy” in the “Edit” menu. Or right-click the component to be cut or copied, and select “Cut” or “Copy”.

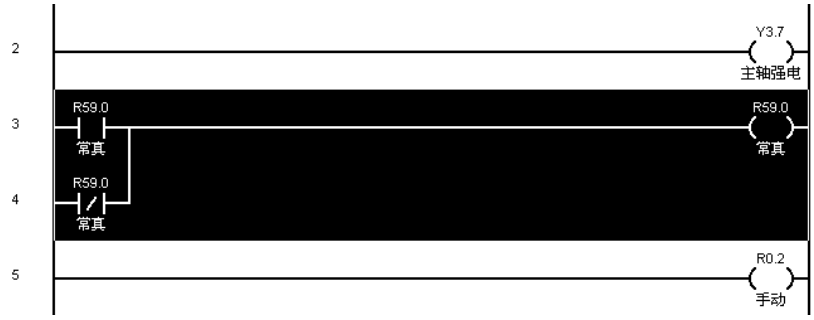
- The first way

- The second way

At last, paste the component on other locations.

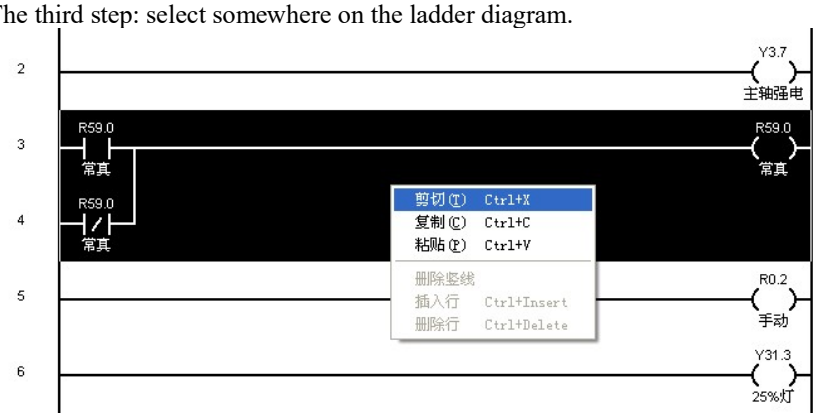
Cutting, Copying and Pasting Multi-row

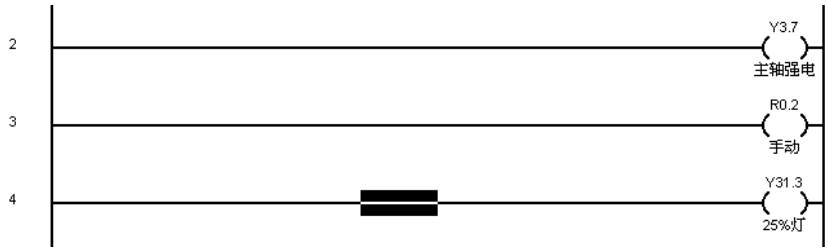
The first step: drag the mouse to select the rows to be cut or copied.



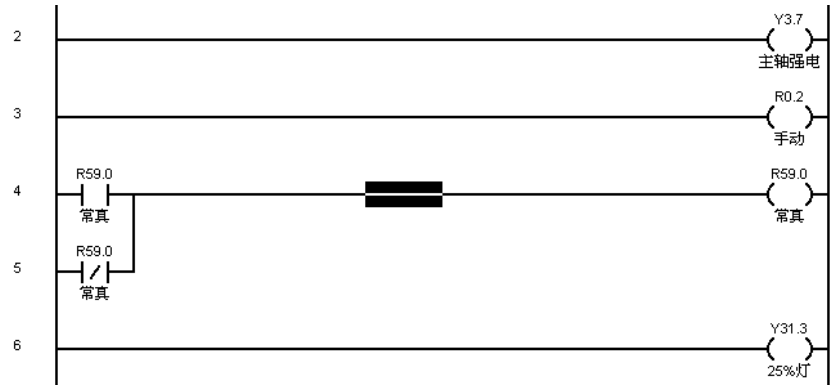
The second step: click “Cut” or “Paste” in the menu. Or right-click the component to be cut or copied, and click “Cut” or “Paste”.

The third step: select somewhere on the ladder diagram.



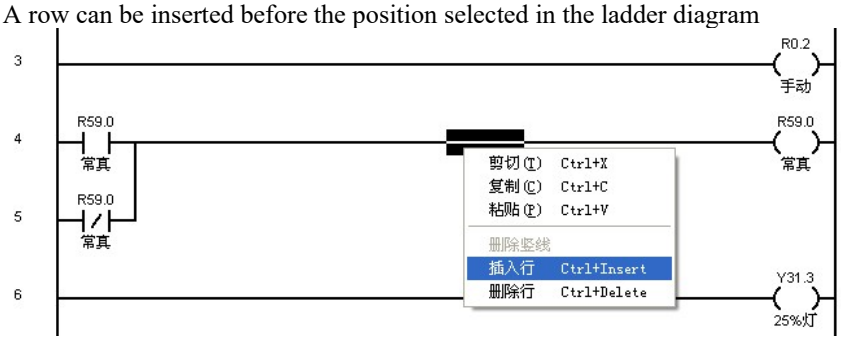


The forth step: Click “Paste”



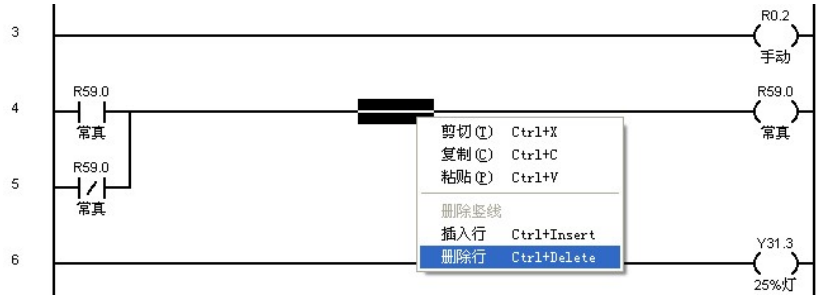
Insert Row

A row can be inserted before the position selected in the ladder diagram



Delete Row

Can delete a row selected in the diagram.



Undo

Using this button on the toolbar to undo the previous operations

Redo

Select this button on the toolbar to recover the undone operations.

Conversion

Using this button to convert the current ladder diagram to the corresponding statement list. If there are errors in the ladder diagram, the message box showing error information will pop up.

Output

Using this button to convert the current ladder diagram to the corresponding statement list, and output the plc.dit file (execution file of ladder diagram). If there are errors in the ladder diagram, the message box showing error information will pop up.

Statement List Operation

Edit

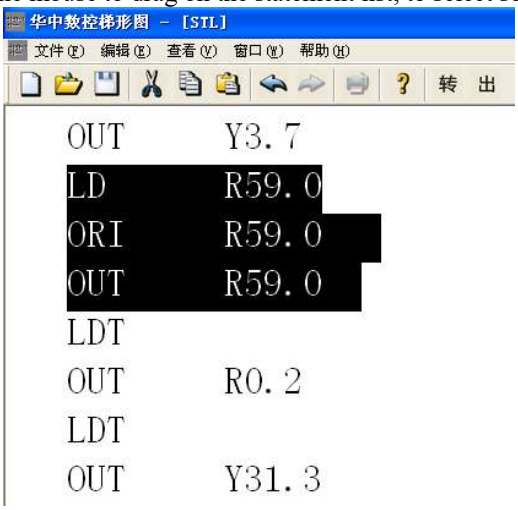
In the statement list, type characters directly to edit the statement list.

After a line of statement has been typed and the cursor is moved away, the system will check and arrange the line.

If there are errors in this line, the statement list will annotate the errors.

Cut, Copy and Paste

Use the mouse to drag on the statement list, to select some statements.



Then use Cut, Copy and Paste in the menu, to work accordingly.

Conversion

Select this button on the toolbar, to convert the current statement list to the corresponding ladder diagram.

Output

Hit this button on the toolbar to convert the current ladder diagram to the corresponding statement list, and output the plc.dit file (execution file of ladder diagram).

Appendix A

- Panel of 818A lathe system

	0	1	2	3	4	5	6	7
X480	Auto	Single block	JOG	Increment	Reference point return	Chuck release/ clamping	Internal/ External clamping	Dry run
X481	Block skip	Optional stop	MST Lock	Machine Lock	Tailstock loosening/ tightening	Hydraulic start	Feed hold Ⅱ	Manual tool change
X482		—X		x1	x10	x100	x1000	Lamp
X483	Protective door	—Z	Fast forward	+Z	Spindle Jog	Cooling	Lubrication	Spindle upshift
X484	Chip removal CW	Chip removal CCW		+X		Spindle CW	Spindle stop	Spindle CCW
X485	Spindle downshift		Overtravel release					
X486	Rapid traverse override				Cycle start	Feed hold		
X487	Spindle override							
X488	Handwheel emergency stop, Handwheel axis selection, and Handwheel magnification							
X489	Feedrate override							
X490	Incremental pulse per cycle by handwheel							
X491								

- Panel of 818A milling system

	0	1	2	3	4	5	6	7
X480	Auto	Single block	JOG	Increment	Reference position return	Tool change permission	Tool clamping	Dry run
X481	Block skip	Optional stop	Z-axis lock	Machine Lock	Protective door	Lamp	Feed hold Ⅱ	Manual tool change
X482	+4	+Z	—Y	x1	x10	x100	x1000	F1
X483	F2	+X	Fast forward	—X	Spindle orientation	Spindle Jog	Spindle brake	Cooling
X484	F3	F4	+Y	—Z	—4	Spindle CW	Spindle stop	Spindle CCW
X485	Lubrication		Overtravel release					
X486	Rapid traverse override				Cycle start	Feed hold		
X487	Spindle override							
X488	Handwheel emergency stop, Handwheel axis selection and Handwheel magnification							
X489	Feedrate override							
X490	Incremental pulse per cycle for handwheel							
X491								

- Panel of 818B lathe system

	0	1	2	3	4	5	6	7
X480	Auto	Single block	JOG	Increment	Reference position return	Chuck clamping	Tailstock loosening /tightening	Dry run
X481	Block skip	Optional stop	MST Lock	Machine Lock	Center rest	Tailstock joint	Feed hold Ⅱ	Manual tool change
X482				0%	25%	Spindle CW	Spindle stop	Spindle CCW
X483	Lamp	+C	—Y		50%	100%	Spindle Jog	Spindle upshift
X484	Spindle downshift	Protective door	—X	Fast forward	+X	F1	F2	Cooling
X485	Lubrication	Hydraulic start	Auto power off		+Y	—C	F3	F4
X486	Chip removal CW	Chip removal stop	Chip removal CCW	Overtravel release	Cycle start	Feed hold		
X487	Spindle override							
X488	Handwheel emergency stop, Handwheel axis selection, and Handwheel magnification							
X489	Feedrate override							
X490	Incremental pulse per cycle for handwheel							
X491								

- Panel of 818B milling system

	0	1	2	3	4	5	6	7
X480	Auto	Single block	JOG	Increment	Reference position return	Tool change permission	Tool clamping	Dry run
X481	Block skip	Optional stop	Z-axis lock	Machine Lock			Magazine CW	Magazine CCW
X482	X	Y	Z	0%	25%	Spindle CW	Spindle stop	Spindle CCW
X483	Lamp	A	B	C	50%	100%	Spindle orientation	Spindle Jog
X484	Spindle brake	Protective door	7	8	9	F1	F2	Cooling
X485	Lubrication	Chip blowing	Auto power off	—	Fast forward	+	F3	F4
X486	Chip removal CW	Chip removal stop	Chip removal CCW	Overtravel release	Cycle start	Feed hold		
X487	Spindle override							
X488	Handwheel emergency stop, Handwheel axis selection, and Handwheel magnification							
X489	Feedrate override							
X490	Incremental pulse per cycle for handwheel							
X491								